War of the Worlds : Branch Consistency in Distributed Systems

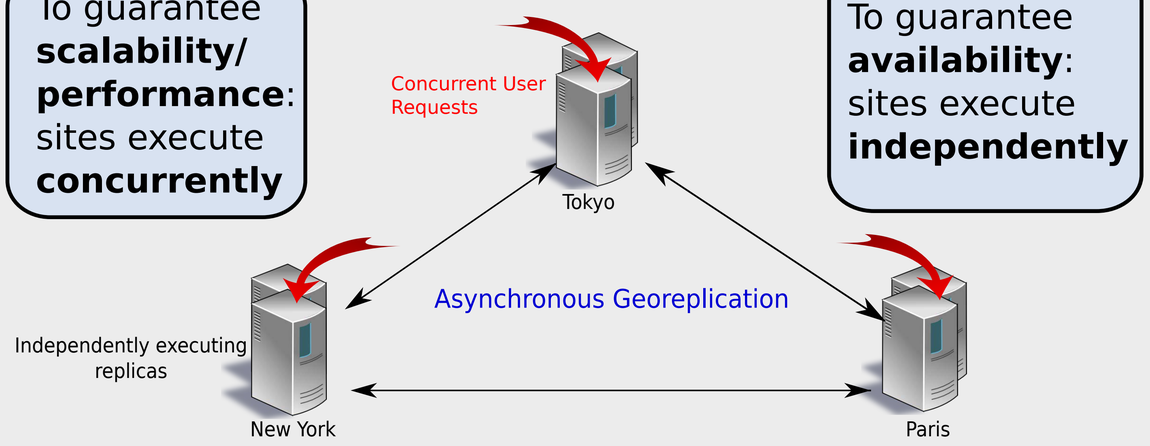
UNIVERSITY OF CAMBRIDGE

Natacha Crooks, Jean Bacon, Steven Hand
Cambridge University Computer Laboratory
natacha.crooks@cl.cam.ac.uk

Reinhard Munz, Allen Clement
MPI-SWS

Concurrent/distributed systems fundamentally consist of multiple independent executions

To guarantee **scalability/ performance**: sites execute **concurrently**

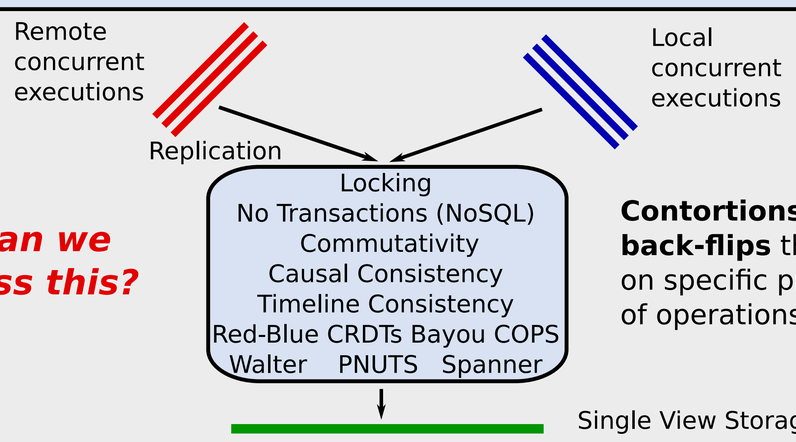


To guarantee **availability**: sites execute **independently**

Unrealistic to expect that a unique system view could or even should exist at a given time.

Problem: Fundamental storage mismatch

Dichotomy between: **distributed reality** and the abstraction of a **single unique view** of the world provided by the local storage



How can we address this?

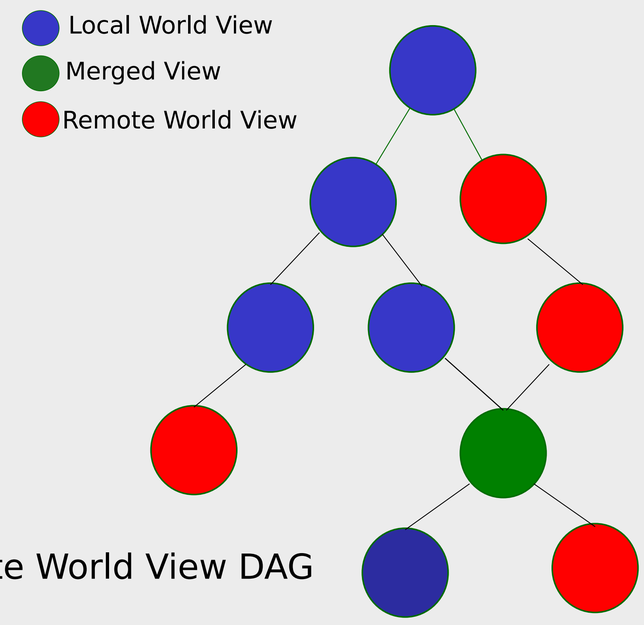
Contortions and back-flips that rely on specific properties of operations or data.

Branch Consistency - a declarative consistency model with branching as a first class primitive

Treat branches as the first-class primitive

- explicity reasons about branches **(world views)**, not independent objects

- guarantees **isolation** between branches



Application-centric

- "consistent". No meaning outside of an application

- Declarative:

=> users specify what a **conflict** is

=> users specify **when/how to merge**

Three pillars of Branch Consistency

Conflict Definition

What is a conflict ?

- Defined by the user.
f: (trxn,world view) → {0,1}

- Determines whether can execute a transaction on this world view.

- Conflict definitions are associated with transactions

Conflict Handling

How should we handle conflicts?

- Computational Time Logic
Determine when/how to branch
Constructs World View DAG

- **Linear Time Logic**
Express properties of individual branches.

Conflict Resolution

How/when/if do we resolve the conflict?

- Via a user-defined resolution function

- **Optional and asynchronous**

- Explicitly merge branches, not objects

Prototype: Transactional storage with parallel snapshots

- **Transactional**

- Supports multiversion concurrency control and branches

- Supports **arbitrary conflict definitions**

- Never forces merging

- Handles conflict through **branching**

- **Non-blocking** (including merging and replication)

- Efficiently models the World View DAG

What branch consistency enables

- No more distinction between **local vs remote storage**

- **No more reliance on properties of data/operations**

- **composition of consistency levels** through varying conflict definition

- **flexibility:** emulates existing consistency models

- **performance**. branching can be made cheap