Optimistic Replication

Saito and Shapiro, CSUR 2005

O.R. and CRDTs

- Breakout discussion:
  - How do CRDTs fit into O.R.?
  - What is their “classification”?

O.R. and CRDTs

- How do CRDTs fit into O.R.?
- Multi-master
- op-based and state-based
- commutativity in op-based for conflict handling
- Vclocks for ordering (scheduling): causal broadcast
- Comm topology/prorogation are not discussed

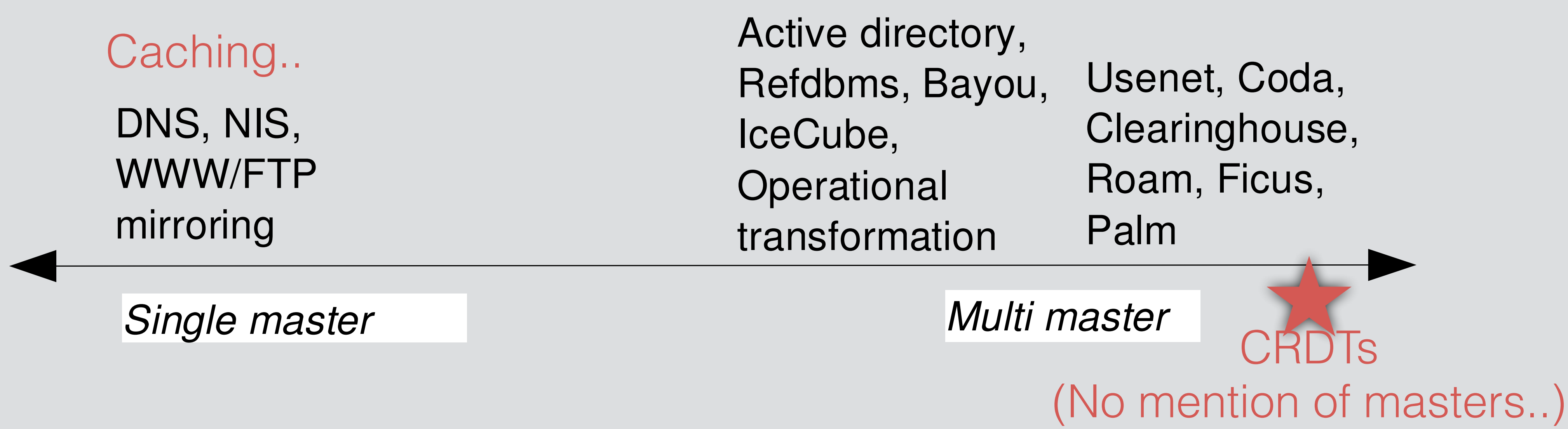
O.R. and CRDTs

- How do CRDTs fit into O.R.?
- No error resolution since they don't need it
- You can introduce errors explicitly as part of your semantics
- A CRDT must encode all the rules for resolving semantic mismatch
- State based CRDT may end up growing indefinitely (tombstones for deletion)
- CmRDTs (op) include many more aspects of O.R. than CvRDTs (state)

O.R. and CRDTs

- How do CRDTs fit into O.R.?
- Consistency guarantees — SEC!
- 5.2.2 and 5.2.3 hint at CRDT design — commutativity and semantic scheduling to resolve “conflicts” automatically with a canonical ordering
- Are CRDTs syntactic or semantic?
    - Semantic by construction and point of view of the O.R. survey. But, internally could be rather syntactic in the definition.
- Contrasting viewpoint: O.R. much broader than CRDT — include more aspects
    - CR**DT** is a ***data type*** abstraction. It doesn’t have any knowledge of the distribution, just knowledge of semantics of the *data.*

Single v. Multi-master

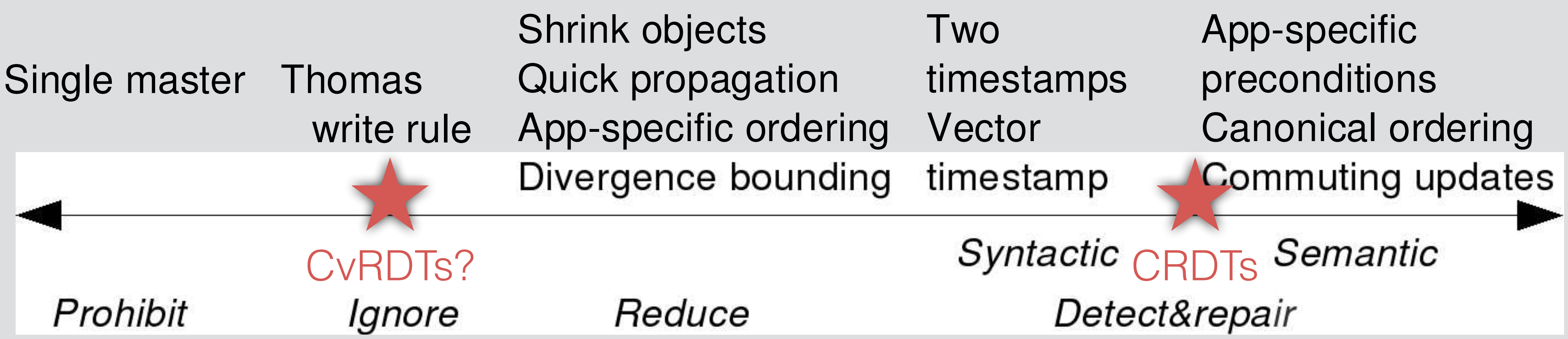


- More masters = more concurrent updates = more conflicts = more divergence between replicas
- Irrelevant to CRDT: commutativity resolve conflicts regardless of number of conflicts.
- Single-master CRDT would provide strong consistency
- SEC consistency would vary in its eventuality depending on # of masters
- *Multi-master has higher availability: CRDTs can live with n-1 failures!*

Definition of operations

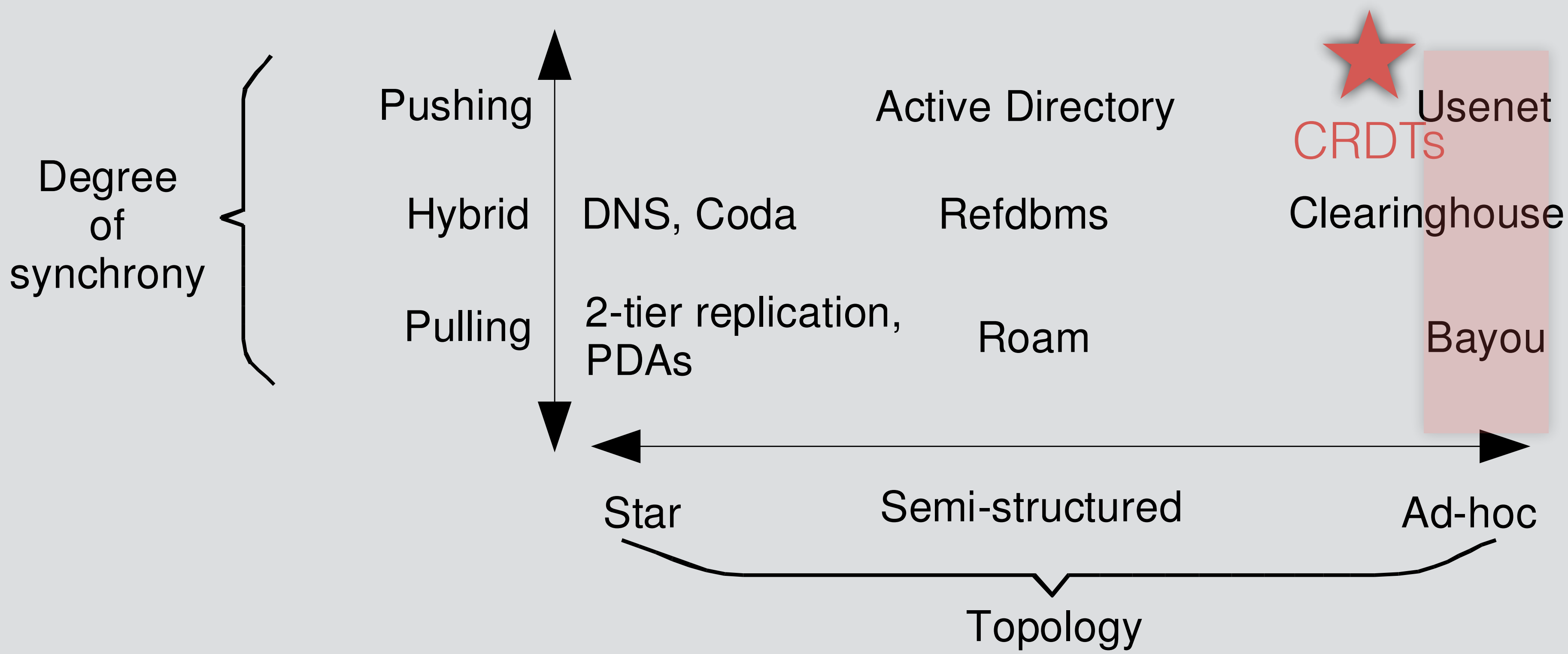
- State Coda is a file system: much larger than CRDTs (many optimizations in O.R for handling large state)
- DT ~ state, but it's abstract and transfer/merging is defined on all of state
- CRDT — can I decompose large state into several CRDTs, and can I coordinate CRDTs to derive SEC across all of my state?
- Immutability helps to freeze state and make it easier to splinter off during conflict resolution/transfers

Conflict handling



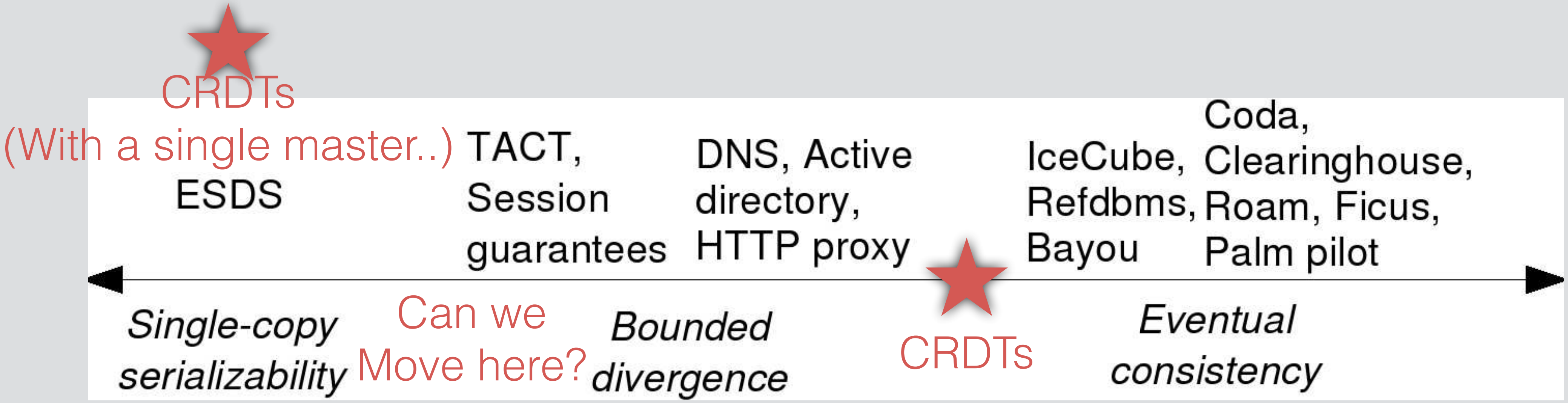
- CRDTs define conflict *away;* by defn. no conflict could occur
- Do CRDTs detect + repair?
- CRDTs to some degree *ignore* conflicts like Thomas write rule

Operation propagation



- CRDT paper: updates are generated by replicas that distribute them ~ push
- CRDT topology is ad-hoc (undefined): works for any topology
- CRDTs could use a different synchrony model if you want to get something stronger/weaker than just push
- CR**DT** permissive of multiple implementations/realizations (env)

Consistency guarantees



- CRDTs provide Strong eventual consistency
- CRDTs choose AP as a baseline, so strongly available. Therefore have to sacrifice some of C(onsistency)
- Could design “CRDTs” that live further to the left of SEC, but they wouldn’t be CRDTs any longer
- *Note: Single-master CRDT would provide strong consistency*

O.R. and CRDTS

- View: CRDT is a mechanism, so fits into survey
  - Could rewrite the survey to accommodate CRDTs
  - They don't fit neatly into the existing dimensions outlined in the paper
- View: CRDT abstraction, therefore doesn't fit into survey
  - CRDT are data types; while paper surveys systems that are specific instantiations
- View: CRDT abstraction, *would fit into survey once implemented*

Optimistic Replication

- CVS -> SVN -> Git
- Git is multimaster, with automatic conflict handling (to an extent)

Optimistic v. Pessimistic

- Faulty duality? (Conflict focused)
  - Weird choice of wording
  - Could also reframe in terms of *availability (CAP)*
  - What is the relationship between CAP and O. v. P split?
  - CAP = it's complicated; So, is O v. P reductionist?
    - Pessimistic: **C**A, or **C**P; Optimistic: **A**P
  - *"Optimism about partitions"* — CA system most optimistic?
- Which ones are more realistic/usable/practical/…?
- When should you use one versus other?
- Scale: global scale pushes design towards high availability => optimistic design
- CAP and O. vs. P encode assumptions about env/use of the systems => design choices

Next: Distributed Hash Tables (DHTs)

- How do we achieve global **scale** in distributed systems? How to coordinate/manage nodes?
- DHTs (overlay networks) provide an answer
  - Will read about the Chord DHT
  - One of the most cited papers in Computer Science
- Will follow-up with loosely structured P2P systems
  - BitTorrent and BitCoin