CoNICE: Consensus in Intermittently-Connected Environments by Exploiting Naming with Application to Emergency Response

Mohammad Jahanian *and K. K. Ramakrishnan*
(University of California, Riverside, USA)

IEEE ICNP 2020

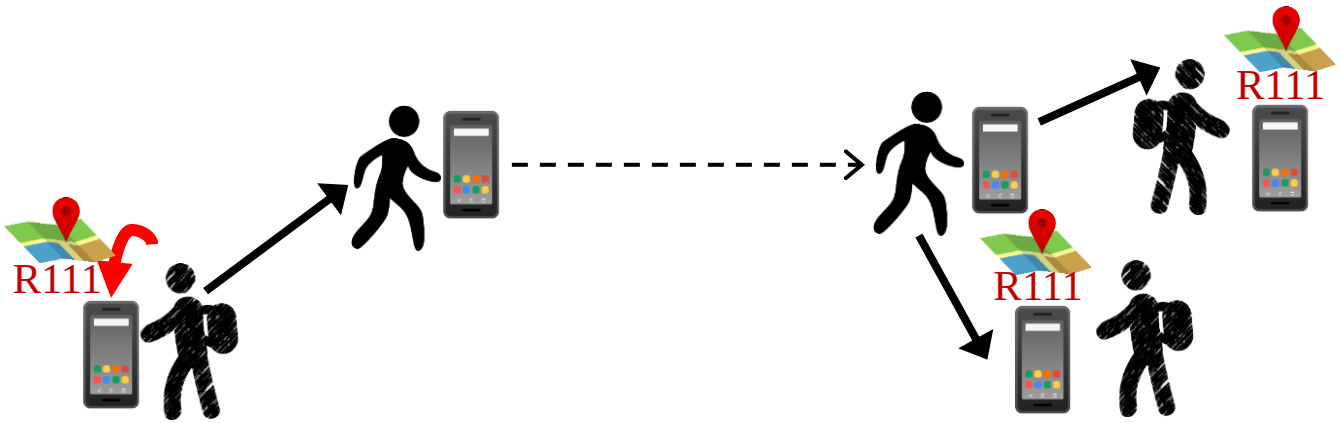
UC RIVERSIDE

Overview

- In many scenarios, e.g., during disasters: information dissemination in intermittently-connected environments where infrastructure damaged
- Many users, e.g., first responders, create updates to a shared dataset, e.g., map data
- Consistency and order of applying update important, challenging, and complex

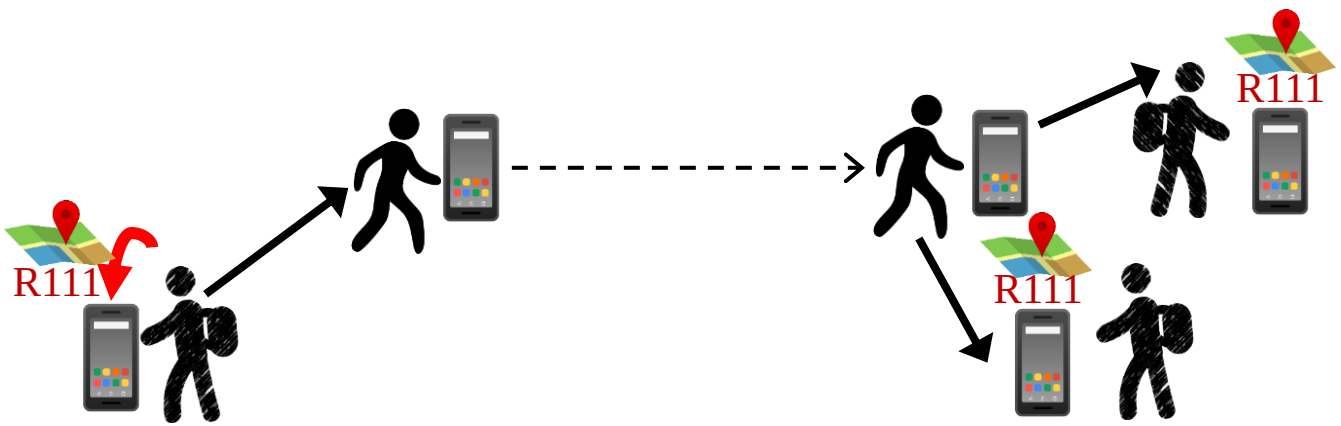
Overview

- In many scenarios, e.g., during disasters: information dissemination in intermittently-connected environments where infrastructure damaged
- Many users, e.g., first responders, create updates to a shared dataset, e.g., map data
- Consistency and order of applying update important, challenging, and complex
- CoNICE: a framework to ensure consistent dissemination of updates among users in intermittently-connected, infrastructure-less environments



Overview

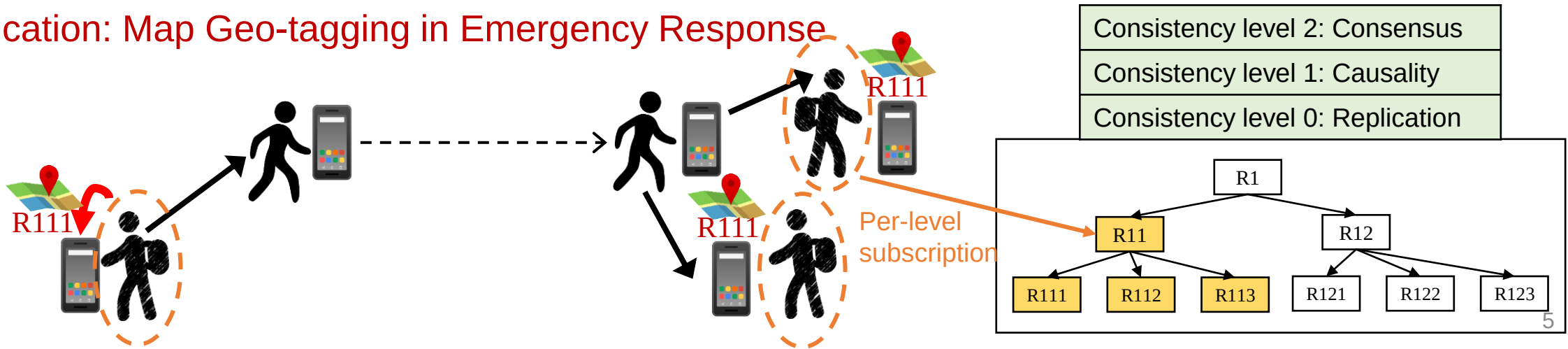
- In many scenarios, e.g., during disasters: information dissemination in intermittently-connected environments where infrastructure damaged
- Many users, e.g., first responders, create updates to a shared dataset, e.g., map data
- Consistency and order of applying update important, challenging, and complex
- CoNICE: a framework to ensure consistent dissemination of updates among users in intermittently-connected, infrastructure-less environments
  - Multiple consistency levels, support both causal ordering and consensus



Consistency level 2: Consensus
Consistency level 1: Causality
Consistency level 0: Replication

Overview

- In many scenarios, e.g., during disasters: information dissemination in intermittently-connected environments where infrastructure damaged
- Many users, e.g., first responders, create updates to a shared dataset, e.g., map data
- Consistency and order of applying update important, challenging, and complex
- CoNICE: a framework to ensure consistent dissemination of updates among users in intermittently-connected, infrastructure-less environments
  - Multiple consistency levels, support both causal ordering and consensus
  - Integration of consistent dissemination with naming of information for two purposes:
    1. Enhance relevancy of information dissemination (typical benefit in ICNs)
    2. Enhance the degree of information consistency among relevant users
  - Application: Map Geo-tagging in Emergency Response



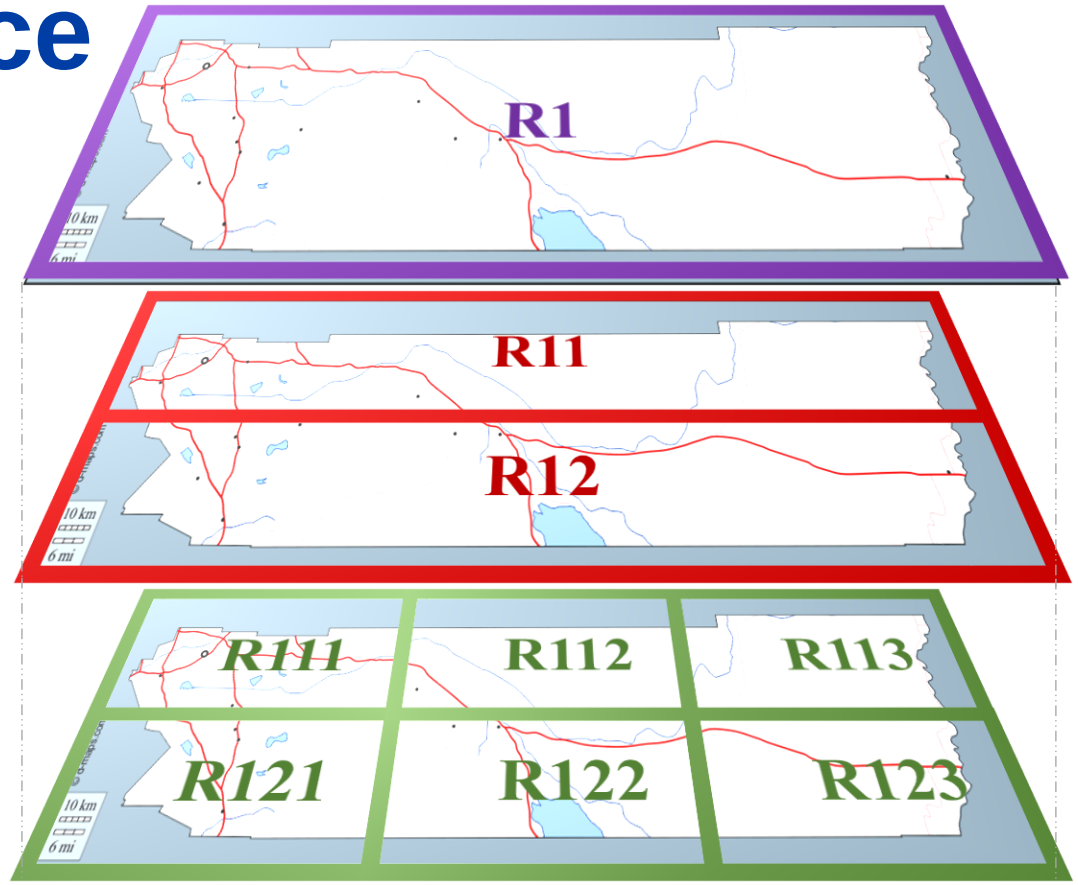
Map Regions and Namespace

- Emergency response: map divided into regions with emergency response tasks
  - Similar approach in online gaming, Augmented Reality, etc.

Map: base layer

Map Regions and Namespace

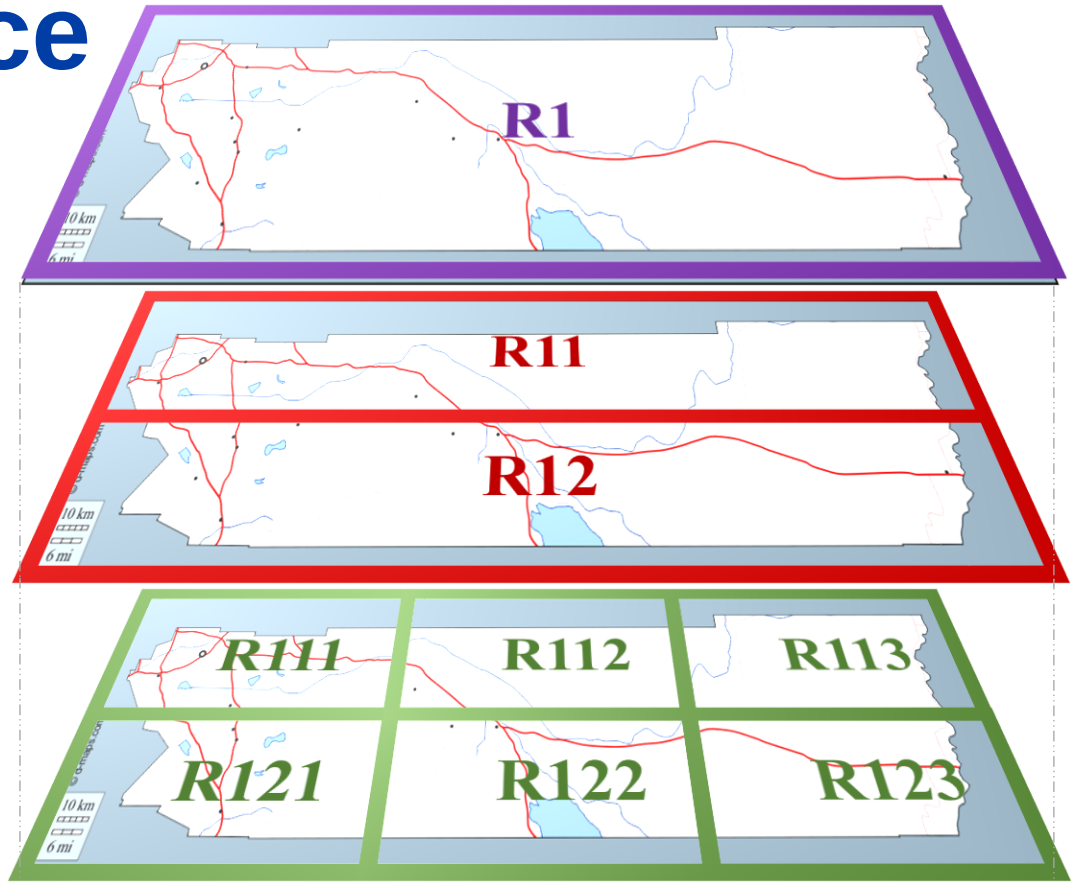
- Emergency response: map divided into regions with emergency response tasks
- Hierarchical region-ing
  - Multiple levels of geographical view; zoom in&out



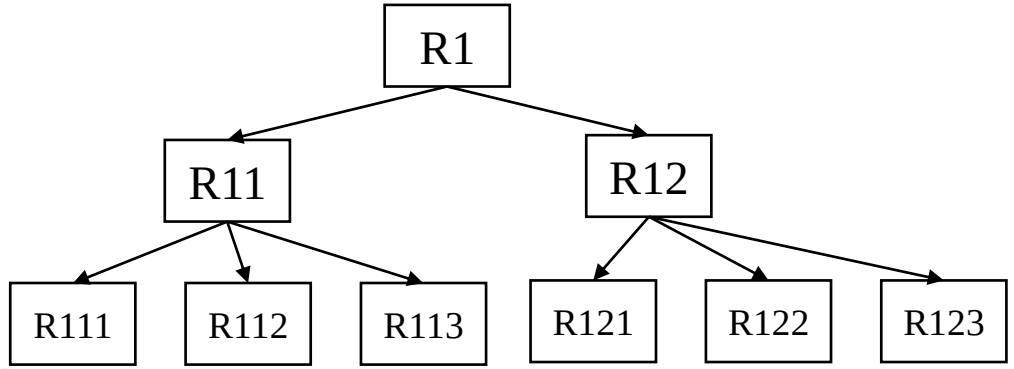
Map: base layer

Map Regions and Namespace

- Emergency response: map divided into regions with emergency response tasks
- Hierarchical region-ing
- Namespace: hierarchical graph
  - First responders indicate fine-grained interest (subscription)
    - ‘R11’ includes ‘R111’, ‘R112’ and ‘R113’ too
    - Subscription to ‘R11’ means implicit subscription to all its descendants too, i.e., ‘R111’, etc.



Map: base layer



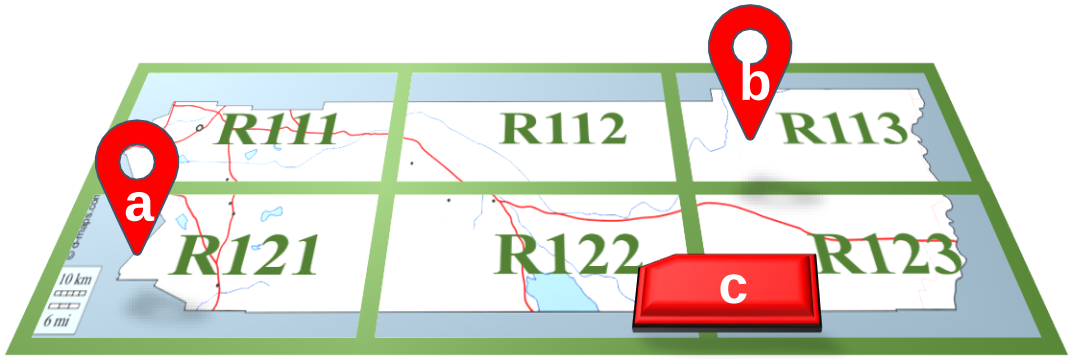
Namespace

Map Regions and Namespace

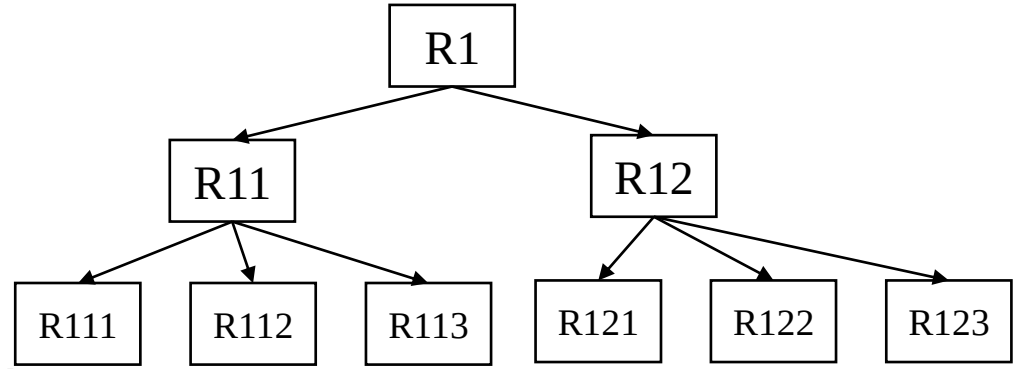
- Emergency response: map divided into regions with emergency response tasks
- Hierarchical region-ing
- Namespace: hierarchical graph
- Users create region-bound updates: Map geo-tagging
  - Bootstrap: every first responder has the background map (base layer) and namespace (offline)
  - Goal: updates (data layer) to be created and disseminated dynamically to relevant recipient, according to their namespace subscription (online)
  - Can be an easy task in normal situation, but challenging in disaster situation: no central coordination, no network infrastructure, no time synchronization

Data: Pins ('a', 'b') and shapes ('c') with information associated with them; e.g.:

- *This house marked as search & rescue completed.*
- *This building is 50% evacuated.*
- *This area needs a firefighting team.*



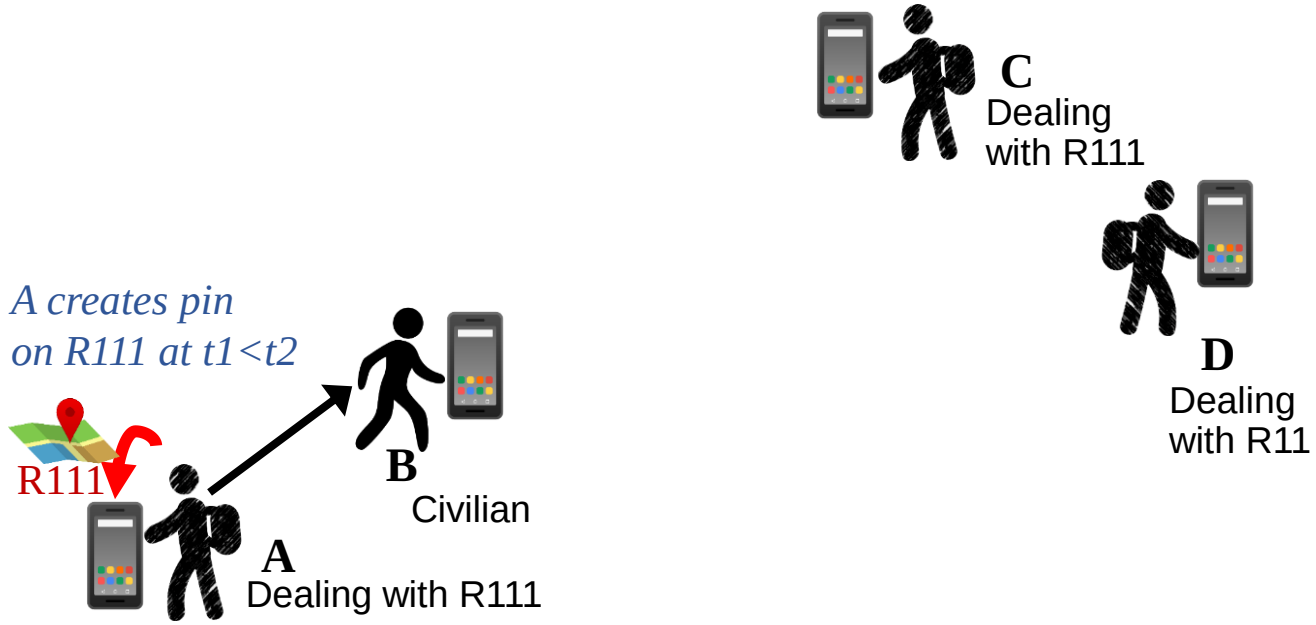
Map: base layer + data layer

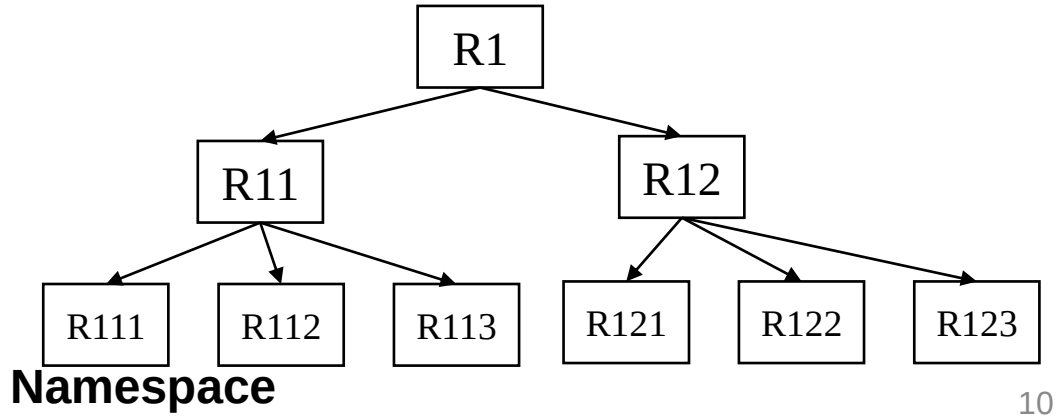


Namespace

Intermittently-Connected Environment

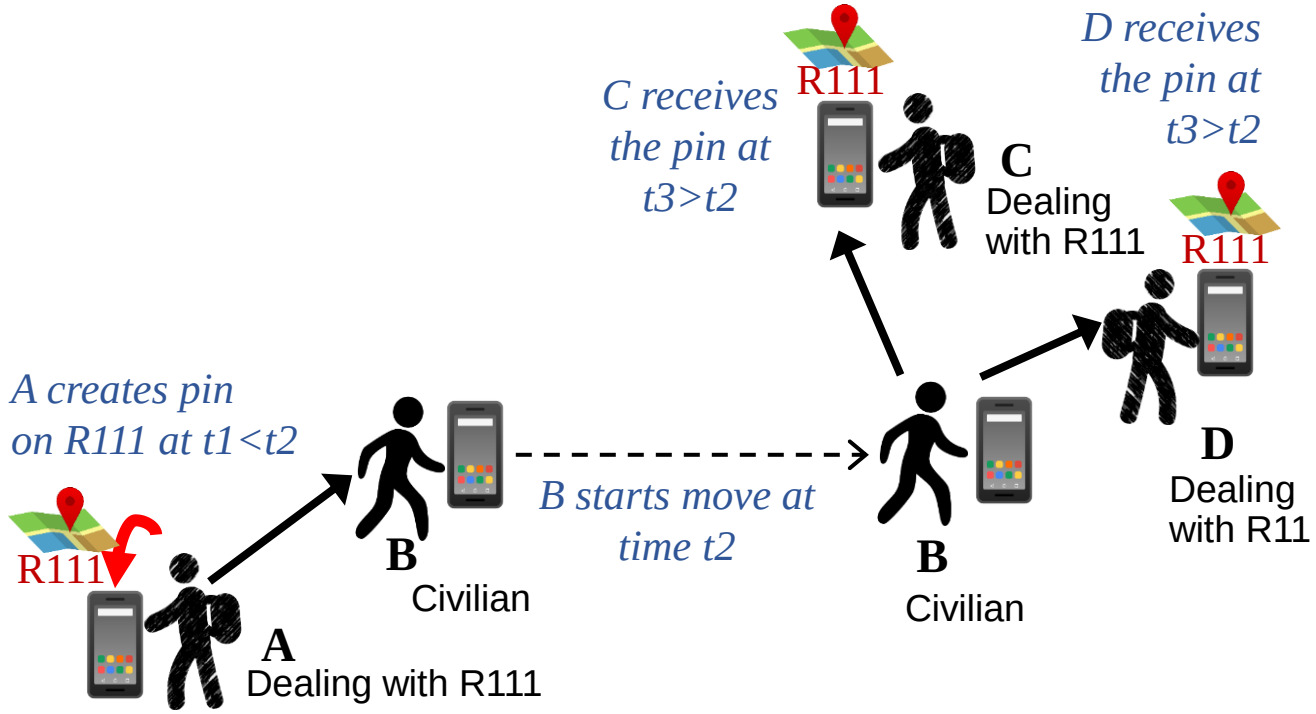
- Network is fragmented (not always a path); relies on users D2D and opportunistic exchanges
  - Users A and B in one fragment; users C and D in another
  - User A creates update about R111; how to reach C and D?

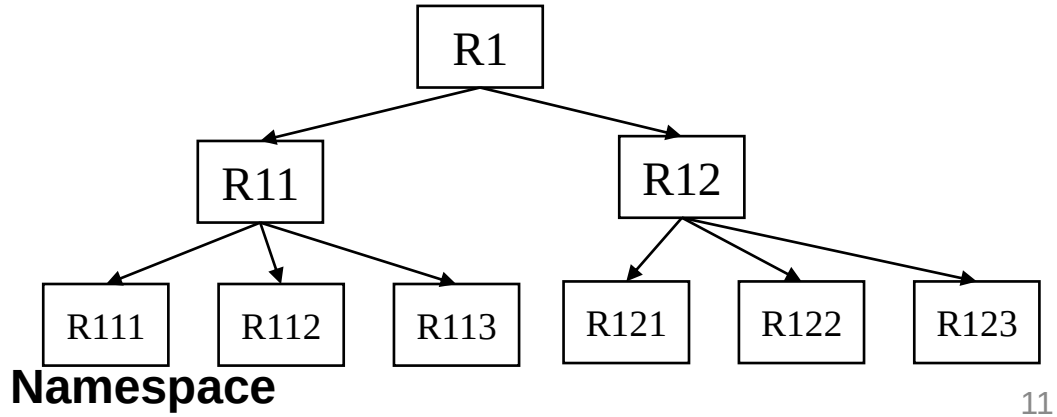




Intermittently-Connected Environment

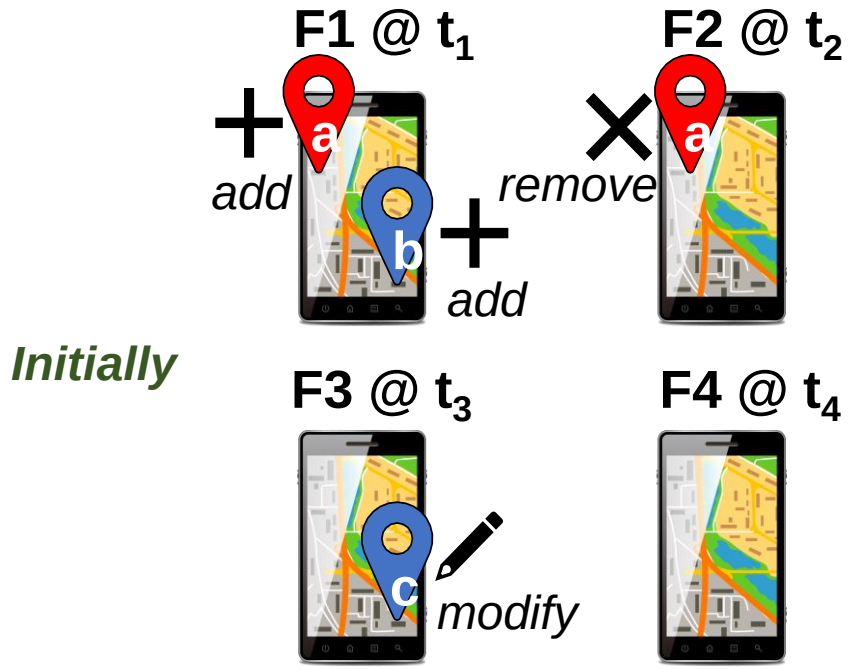
- Network is fragmented (not always a path); relies on users D2D and opportunistic exchanges
  - Users A and B in one fragment; users C and D in another
  - User A creates update about R111; how to reach C and D?
- Thanks to user B's move (acting as a mule), message gets propagated
  - Opportunistic or Delay-Tolerant Networking (DTN)
- The use of namespace makes sure relevant, subscribed users are notified and participate
- Many users create many updates without centralized coordination





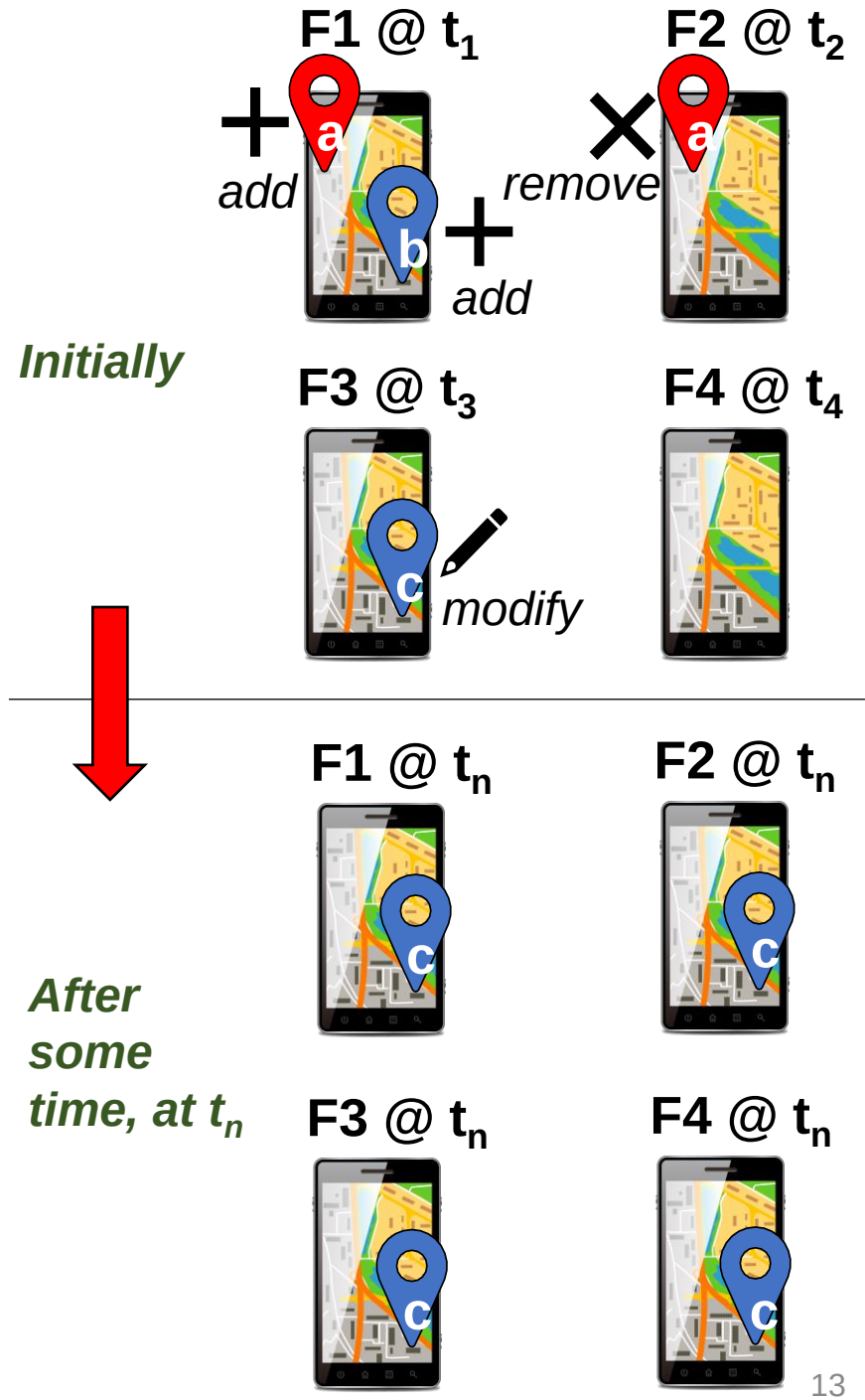
Consistent Dissemination

- Updates on a single, shared dataset (map data layer)
- Each update: add/remove/modify pins/shapes
  - Order of applying matters in result (final map view on individual first responder device)
- Consistency of updates is important and challenging
  - Definitions later



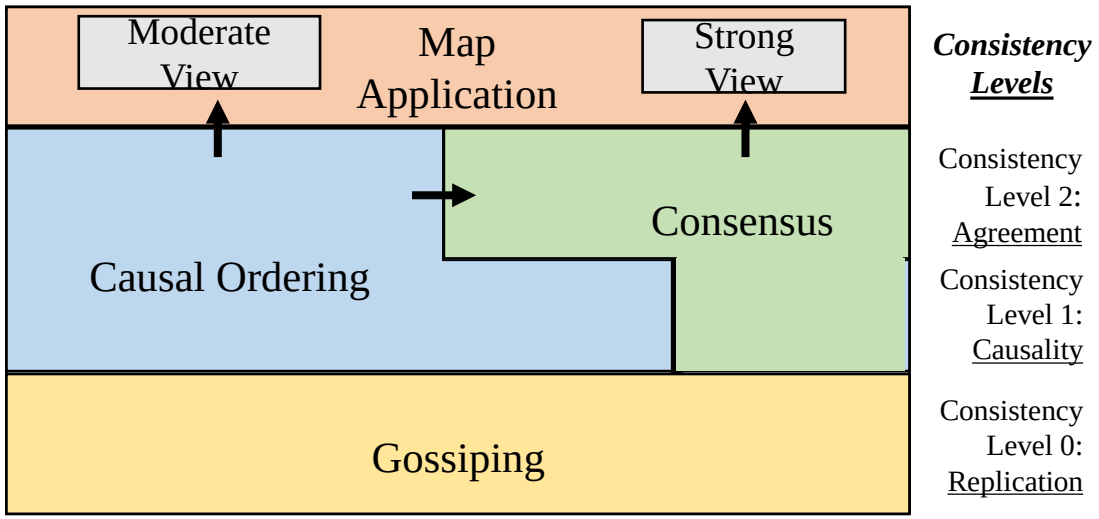
Consistent Dissemination

- Updates on a single, shared dataset (map data layer)
- Each update: add/remove/modify pins/shapes
  - Order of applying matters in result (final map view on individual first responder device)
- Consistency of updates is important and challenging
  - Definitions later
- Goal: eventually, all relevant users have the same view of the map
  - Strong consistency through consensus (agreement) on order of updates in each region
  - Strong consistency requiring complex, time-consuming procedures → CoNICE provides flexibility of multiple consistency levels bound to named regions



CoNICE Architecture

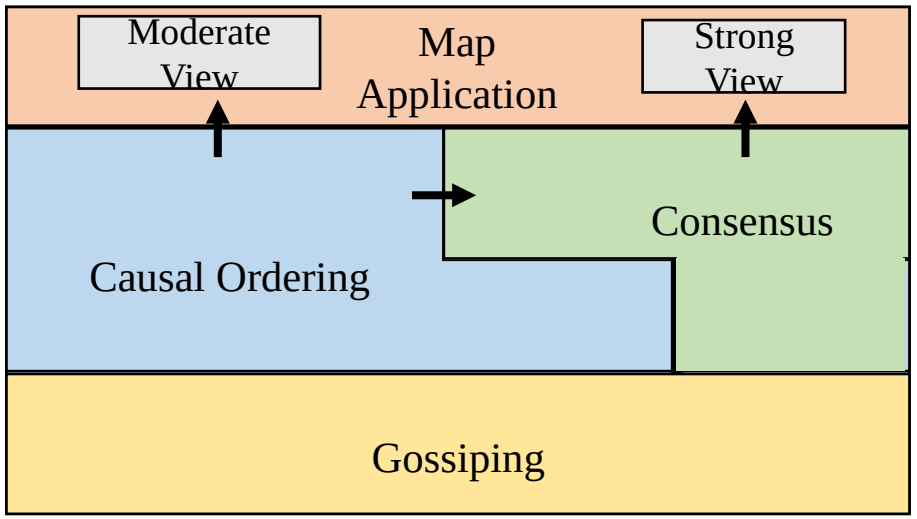
- **Consistency level 0**: **Replication** – make sure users receive updates; ***Gossiping***
- **Consistency level 1: Causality** – make sure users get causally ordered view of updates (Moderate View); ***Causal Ordering***
- **Consistency level 2: Agreement** – make sure users get an agreed upon view of updates (Strong View); ***Consensus***

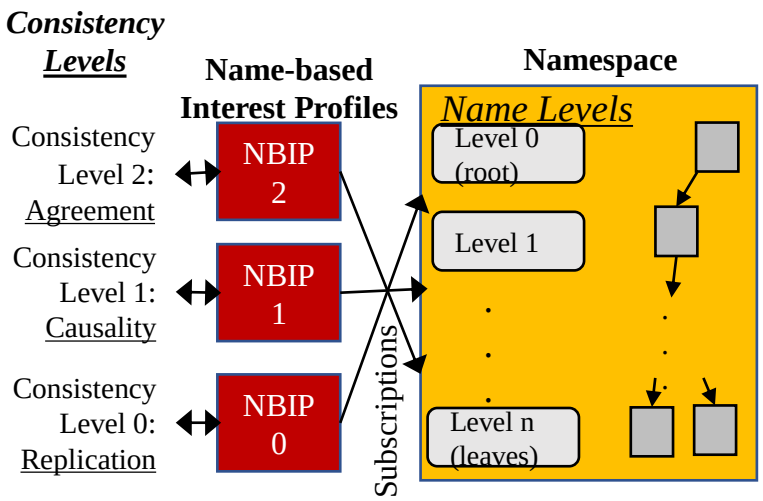


CoNICE Architecture

- **Consistency level 0**: **Replication** – make sure users receive updates; ***Gossiping***
- **Consistency level 1: Causality** – make sure users get causally ordered view of updates (Moderate View); ***Causal Ordering***
- **Consistency level 2: Agreement** – make sure users get an agreed upon view of updates (Strong View); ***Consensus***

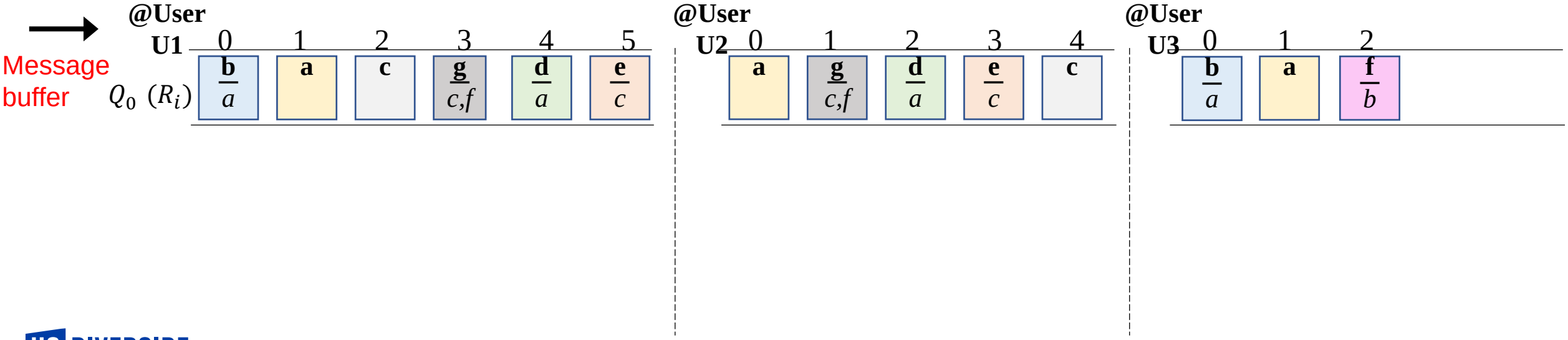
- To achieve selectiveness: Name-based Interest Profiles (NBIPs) for each consistency level (NBIP0,1,2)
- Each NBIP a name subscription, to limits a user's participation according to relevant namespace subsets





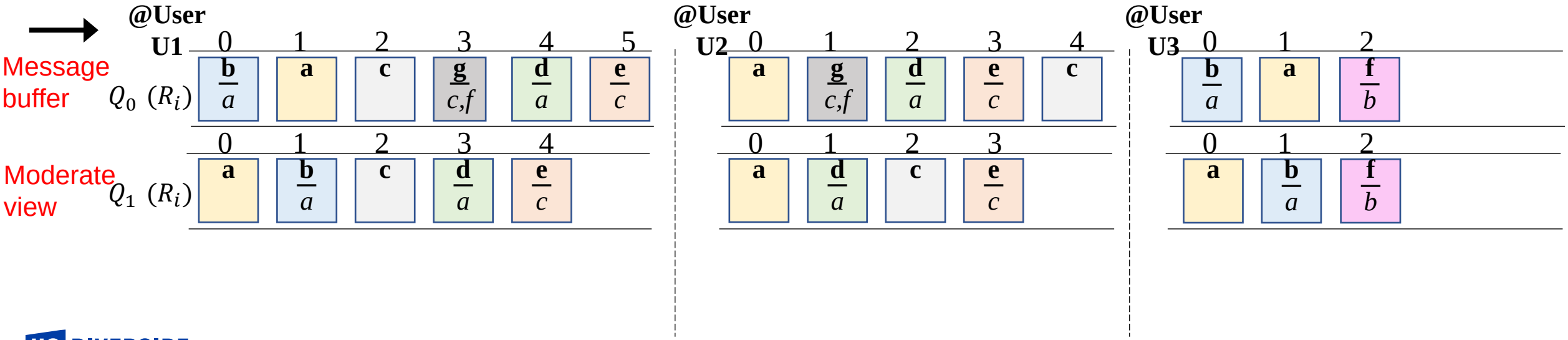
Consistency Levels

- Three levels, each w/ a sequence/queue & slots to fill; bound to regions; incremental
- Three users U1, U2, and U3; updates (a, b, etc.) & dependencies ($\frac{b}{a}$: b depends on a)
- **Level 0: Replication**: Gossiping propagates updates
  - Probabilistic (no guaranteed delivery); filled in order of receipt out of dependency order



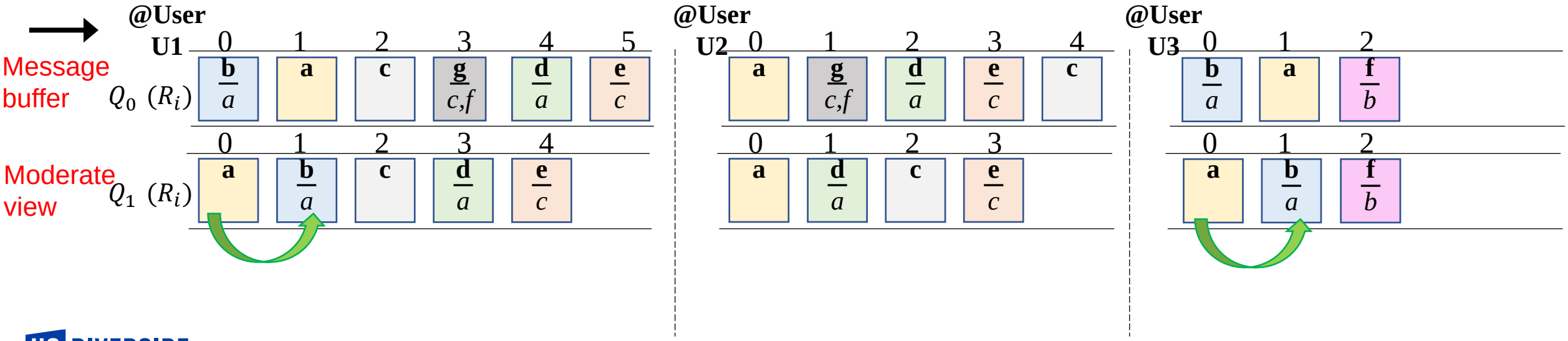
Consistency Levels

- Three levels, each w/ a sequence/queue & slots to fill; bound to regions; incremental
- **Level 0: Replication**: Gossiping propagates updates (a, b, etc.; $\frac{b}{a}$: b depends on a)
  - Probabilistic (no guaranteed delivery); filled in order of receipt out of dependency order
- **Level 1: Causality**: Causal Ordering applies Q_0 in causal order → Moderate View



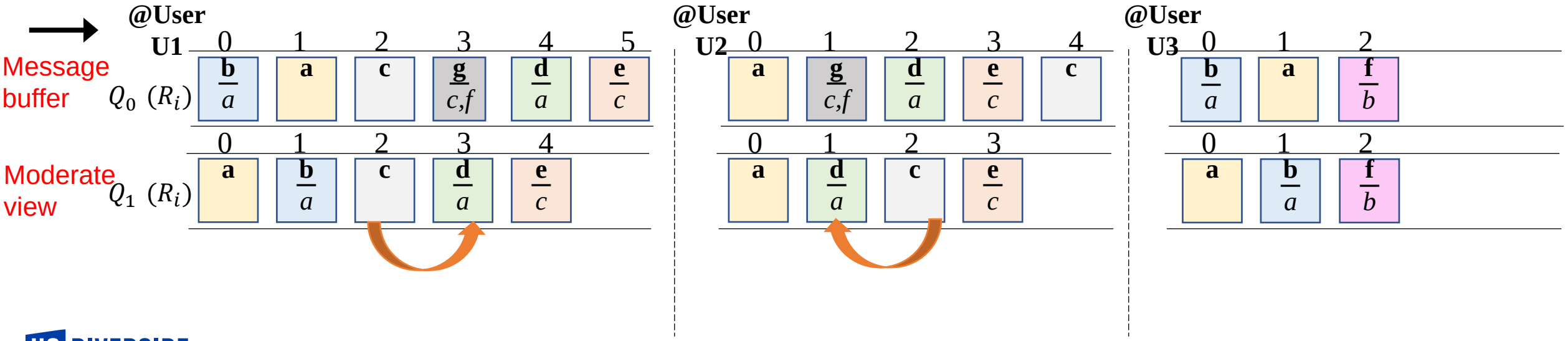
Consistency Levels

- Three levels, each w/ a sequence/queue & slots to fill; bound to regions; incremental
- **Level 0: Replication**: Gossiping propagates updates (a, b, etc.; $\frac{b}{a}$: b depends on a)
  - Probabilistic (no guaranteed delivery); filled in order of receipt out of dependency order
- **Level 1: Causality**: Causal Ordering applies Q_0 in causal order → Moderate View
  - Orders "orderable" updates deterministically (e.g., "a" and "b"); good starting point for a useful view



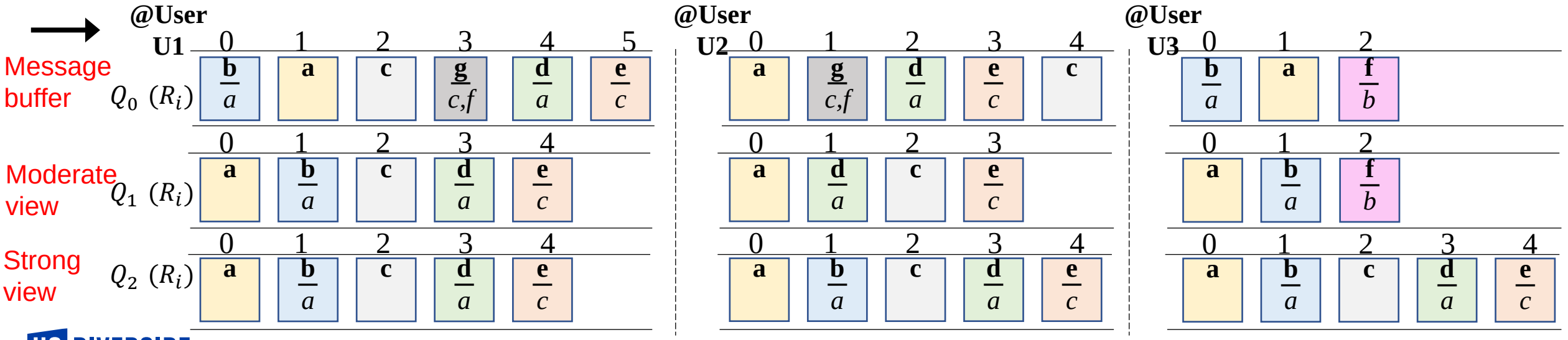
Consistency Levels

- Three levels, each w/ a sequence/queue & slots to fill; bound to regions; incremental
- **Level 0: Replication**: Gossiping propagates updates (a, b, etc.; $\frac{b}{a}$: b depends on a)
  - Probabilistic (no guaranteed delivery); filled in order of receipt out of dependency order
- **Level 1: Causality**: Causal Ordering applies Q_0 in causal order → Moderate View
  - Orders "orderable" updates deterministically (e.g., "a" and "b"); good starting point for a useful view
  - Challenge: "un-orderable" updates (e.g., "c" and "d"); different users create updates concurrently, and may not "see" all potential dependencies (due to no guaranteed delivery)



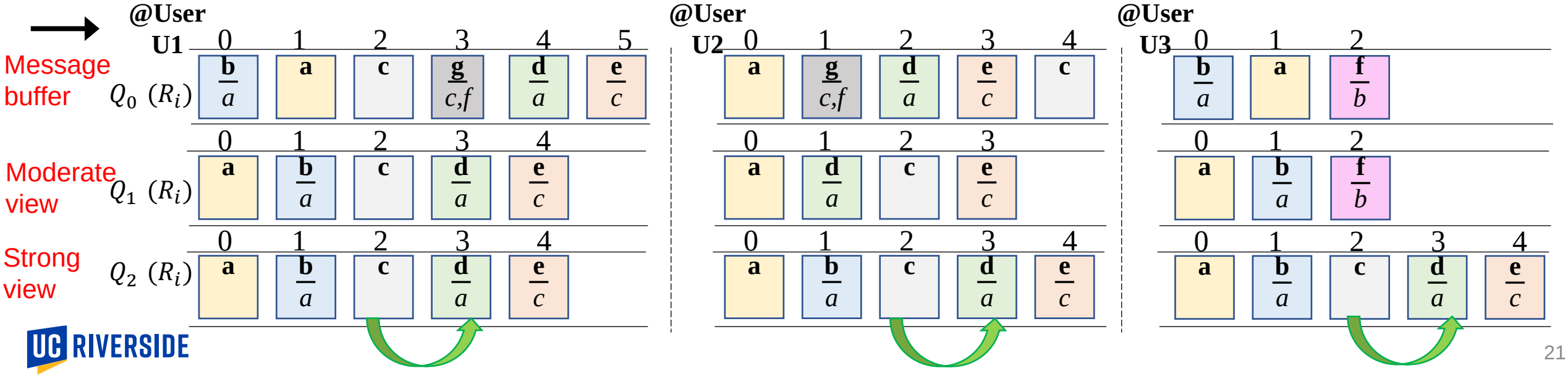
Consistency Levels

- Three levels, each w/ a sequence/queue & slots to fill; bound to regions; incremental
- **Level 0: Replication**: Gossiping propagates updates (a, b, etc.; $\frac{b}{a}$: b depends on a)
  - Probabilistic (no guaranteed delivery); filled in order of receipt out of dependency order
- **Level 1: Causality**: Causal Ordering applies Q_0 in causal order → Moderate View
  - Orders "orderable" updates deterministically (e.g., "a" and "b"); good starting point for a useful view
- **Level 2: Agreement**: Consensus applies Q_1 elements in agreed order → Strong View



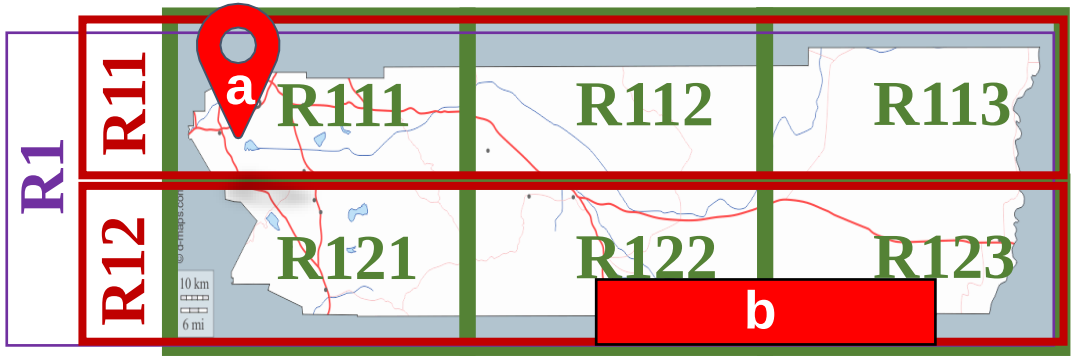
Consistency Levels

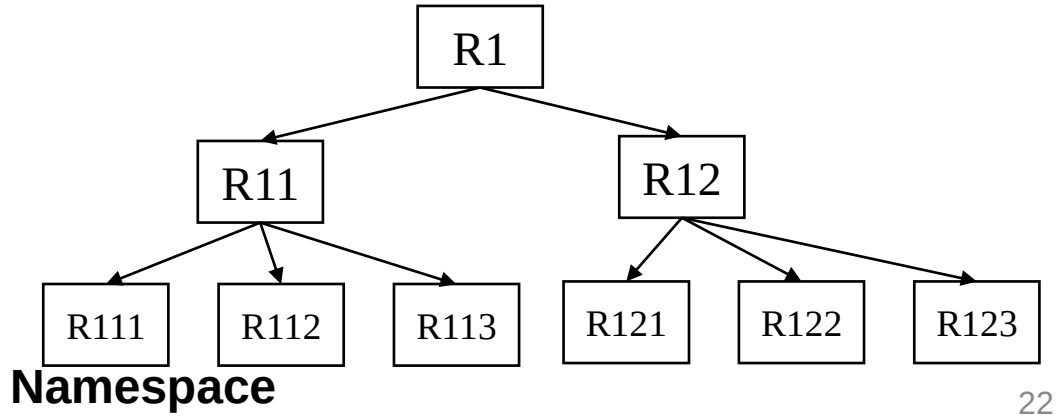
- Three levels, each w/ a sequence/queue & slots to fill; bound to regions; incremental
- **Level 0: Replication**: Gossiping propagates updates (a, b, etc.; $\frac{b}{a}$: b depends on a)
  - Probabilistic (no guaranteed delivery); filled in order of receipt out of dependency order
- **Level 1: Causality**: Causal Ordering applies Q_0 in causal order → Moderate View
  - Orders "orderable" updates deterministically (e.g., "a" and "b"); good starting point for a useful view
- **Level 2: Agreement**: Consensus applies Q_1 elements in agreed order → Strong View
  - Orders even "un-orderable" updates (e.g., "c" and "d"); exactly same across all users



Level 0: Replication – Gossiping

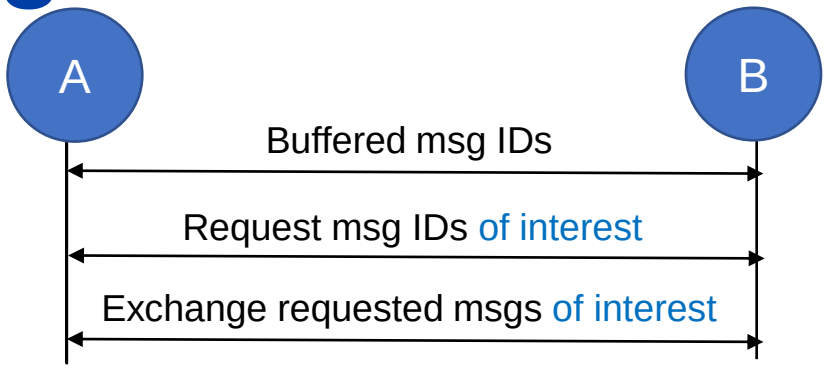
- Design protocols for each level of consistency
- Each update ‘belongs to’ exactly one, (smallest) region large enough to contain update on map
  - ‘a’ belongs to R111, ‘b’ belongs to R12
- Update identified as:
  - **update<userID, regionBelongTo, seqNum>**
  - Can have additional ‘regionsCovered’ field, to include ‘R122’ and ‘R123’ for ‘b’ too



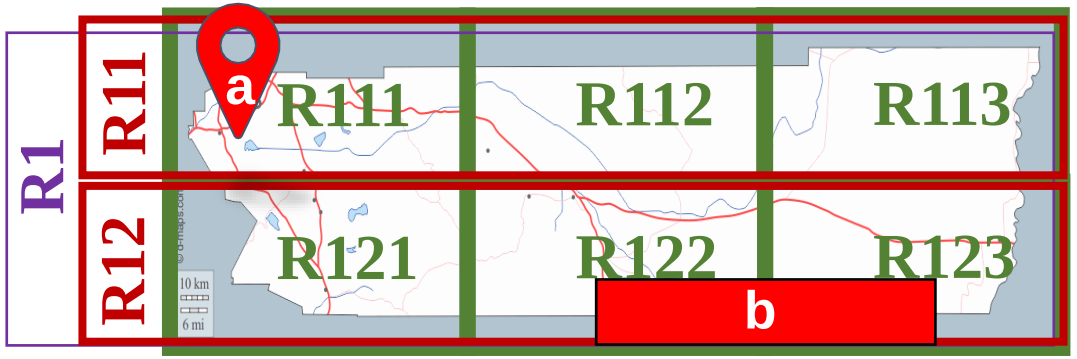


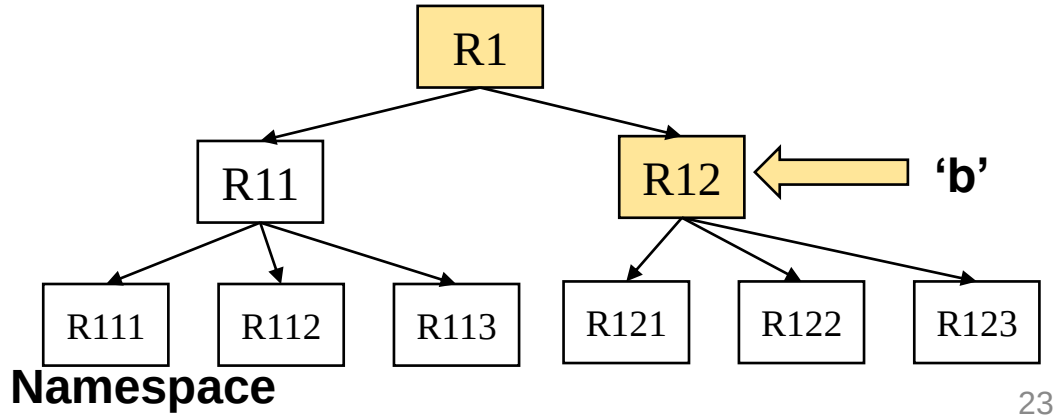
Level 0: Replication – Gossiping

- Design protocols for each level of consistency
- Each update ‘belongs to’ exactly one, (smallest) region large enough to contain update on map
  - ‘a’ belongs to R111, ‘b’ belongs to R12
- Update identified as:
  - **update<userID, regionBelongTo, seqNum>**
- DTN-based Epidemic Routing protocol (Vahdat & Becker '00) for gossiping
  - Users exchange states from their buffers, & messages upon coming into contact
  - Enhance with name-based relevance
  - Use level 0 name-based interest profile (NBIP0)
- Update collected by subscribers of its region-set and above them
  - Subscribers of R1, R12 receive ‘b’



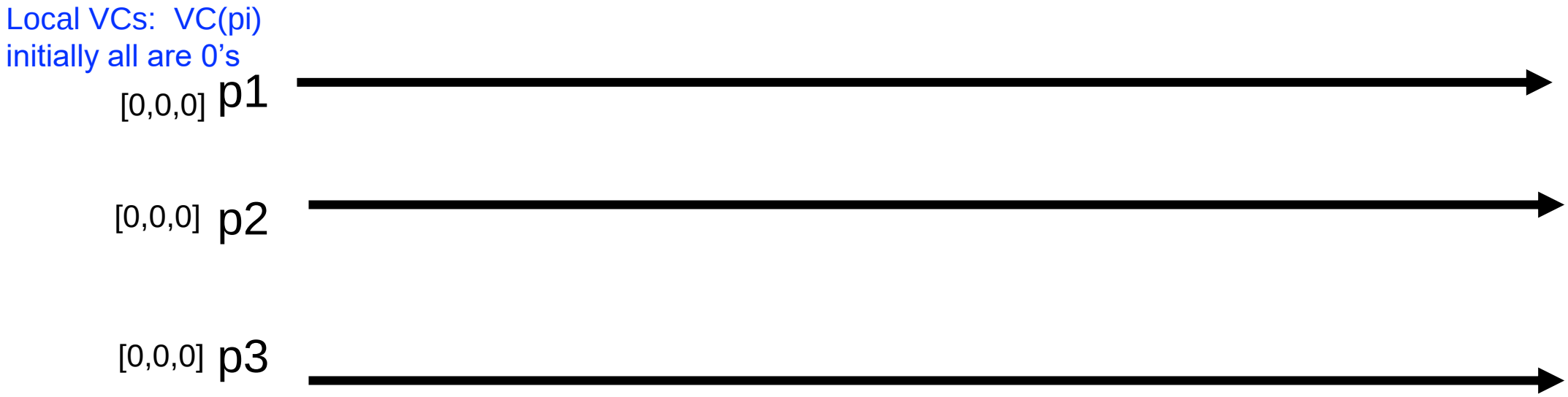
Mutual Replication of relevant updates





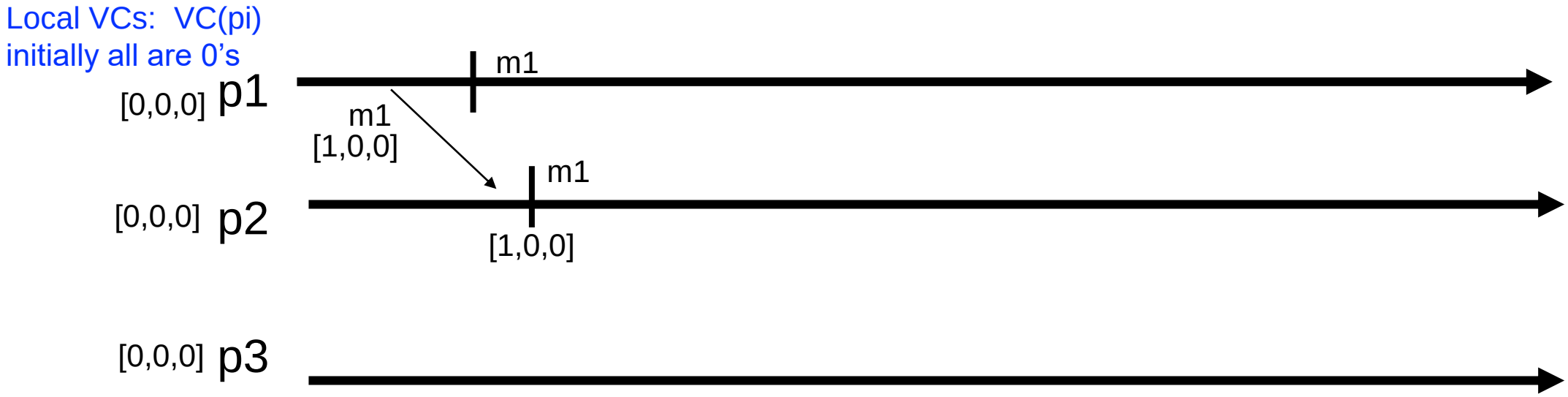
Level 1: Causality – Causal Ordering

- Capture causal relations of updates → moderate consistency/view
- Causality (Lamport ‘78): msg m1 “happened before” m2 (m1→m2) iff,
  - Some user sends m1 and then sends m2 (FIFO order), or
  - Some user receives m1 and then sends m2 (local order), or
  - There exists some message m3 such that m1→m3 and m3→m2 (transitivity rule)
- Logical clock, and its extension, Vector clock: carrying causal history of a message for causal ordering



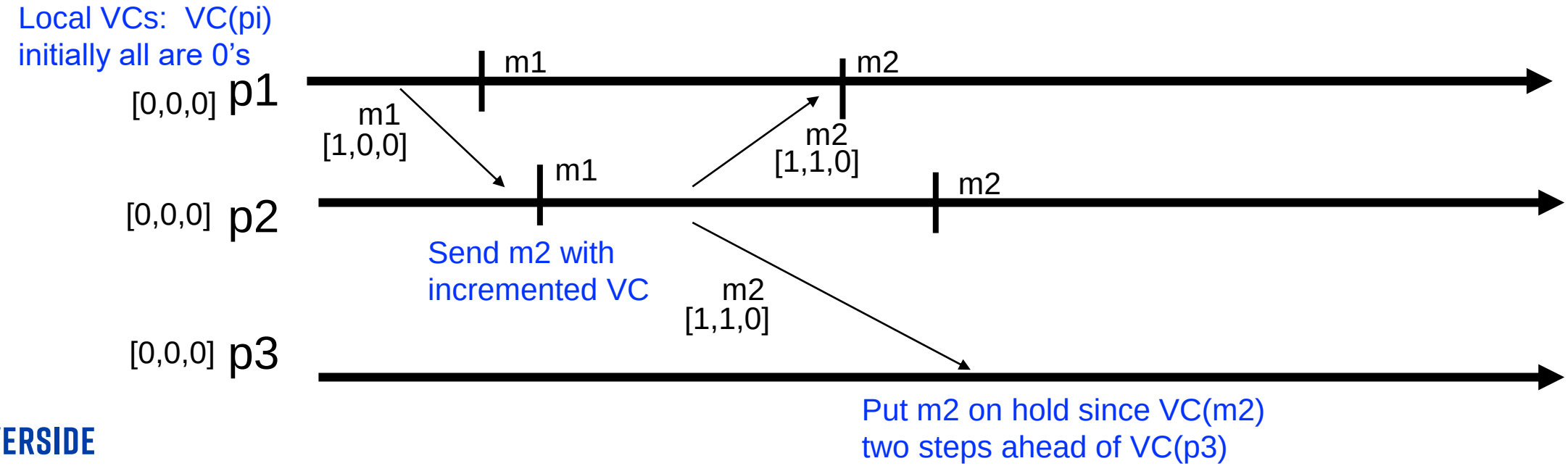
Level 1: Causality – Causal Ordering

- Capture causal relations of updates → moderate consistency/view
- Causality (Lamport ‘78): msg m1 “happened before” m2 (m1→m2) iff,
  - Some user sends m1 and then sends m2 (FIFO order), or
  - Some user receives m1 and then sends m2 (local order), or
  - There exists some message m3 such that m1→m3 and m3→m2 (transitivity rule)
- Logical clock, and its extension, Vector clock: carrying causal history of a message for causal ordering



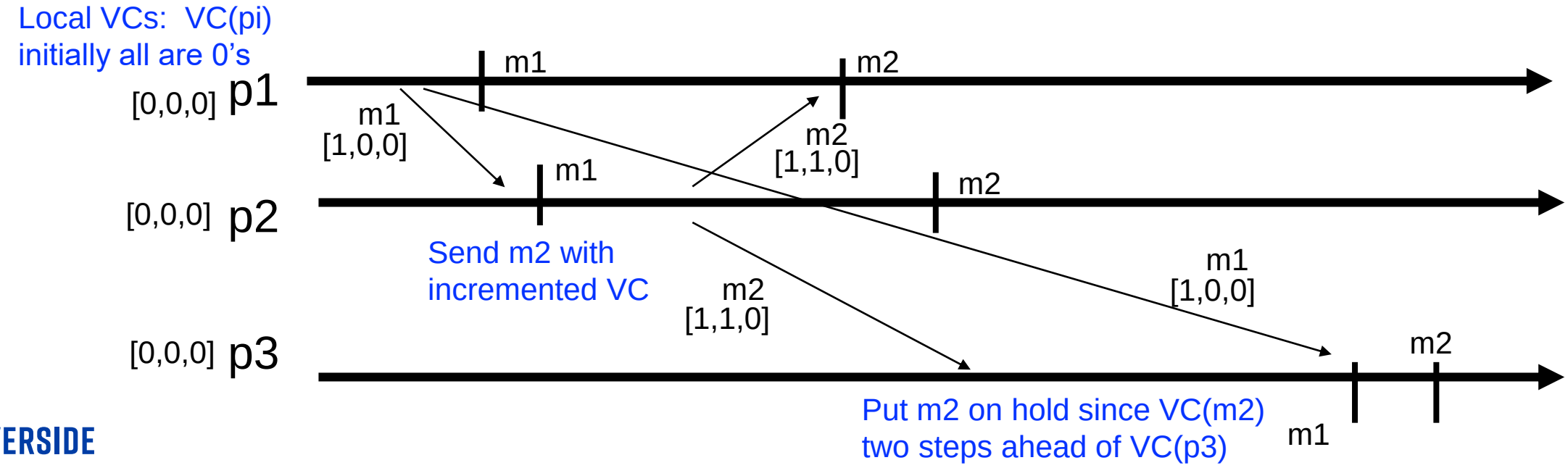
Level 1: Causality – Causal Ordering

- Capture causal relations of updates → moderate consistency/view
- Causality (Lamport '78): msg m1 "happened before" m2 (m1→m2) iff,
  - Some user sends m1 and then sends m2 (FIFO order), or
  - Some user receives m1 and then sends m2 (local order), or
  - There exists some message m3 such that m1→m3 and m3→m2 (transitivity rule)
- Logical clock, and its extension, Vector clock: carrying causal history of a message for causal ordering



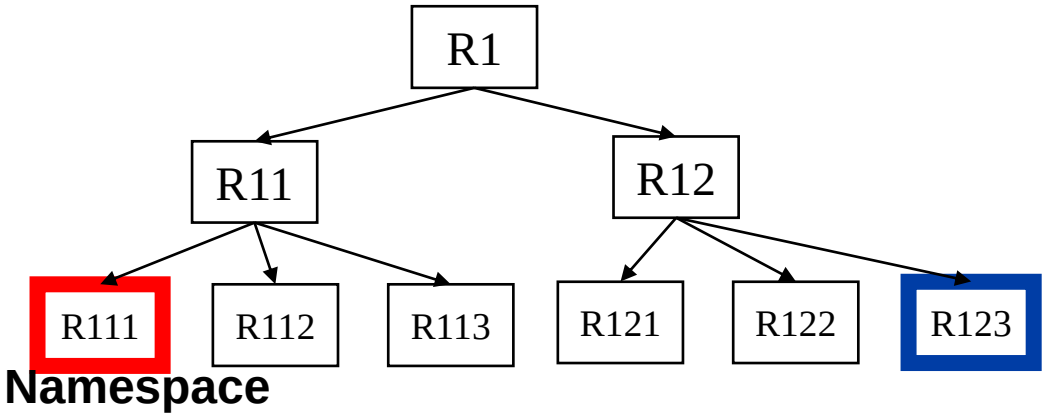
Level 1: Causality – Causal Ordering

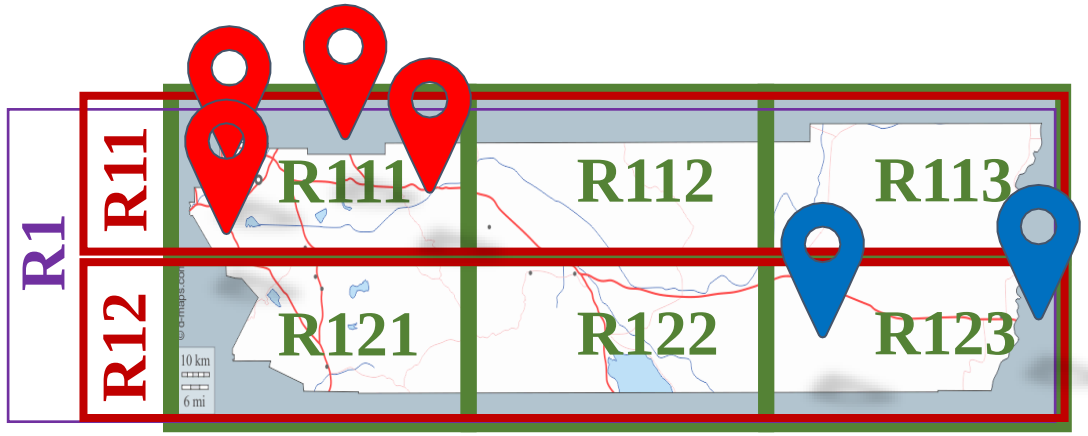
- Capture causal relations of updates → moderate consistency/view
- Causality (Lamport '78): msg m1 "happened before" m2 (m1→m2) iff,
  - Some user sends m1 and then sends m2 (FIFO order), or
  - Some user receives m1 and then sends m2 (local order), or
  - There exists some message m3 such that m1→m3 and m3→m2 (transitivity rule)
- Logical clock, and its extension, Vector clock: carrying causal history of a message for causal ordering



Level 1: Causality – Causal Ordering

- We optimize the causality relation definition, and the causal history being carried
- Causality (CoNICE): msg m1 “happened before in the same region” as m2 (m1→m2); additional condition: m1 and m2 have the same region ID
- CoNICE causal ordering: carrying causal history; only include potentially dependent messages, rather than full vector (using namespace subscription)
  - Users include same-region dependencies that they have seen



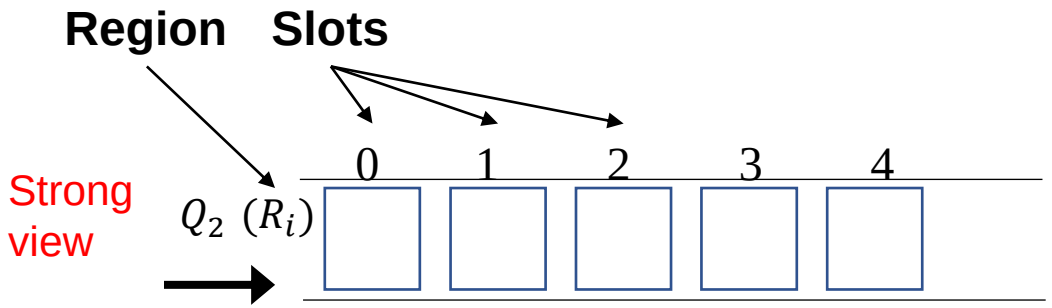


Level 1: Causality – Causal Ordering

- We optimize the causality relation definition, and the causal history being carried
- Causality (CoNICE): msg m1 “happened before in the same region” as m2 (m1→m2); additional condition: m1 and m2 have the same region ID
- CoNICE causal ordering: carrying causal history; only include potentially dependent messages, rather than full vector (using namespace subscription)
- Update: **update<userID, regionBelongTo, seqNum> w References**
  - Implicit dependency: seqNums (for <user,region> pairs) (same user, same region)
    - **[U1, R1, 3]** precedes **[U1, R1, 4]** (FIFO ordering)
  - Explicit dependency: References (identify other users’ updates on same region)
    - **[U1, R1, 3]** precedes **[U2, R1, 1] w Ref [U1, R1, 3]** (Local ordering)
- A reactive mode at recipients to detect causal gaps and requesting them
  - Example: if receive [U1, R1, 4] but don’t have [U1, R1, 3] ⟹ request [U1, R1, 3]
  - Any user with the info’ can respond
  - Only participate for relevant regions, according to level 1 name-based interest profile (NBIP1)

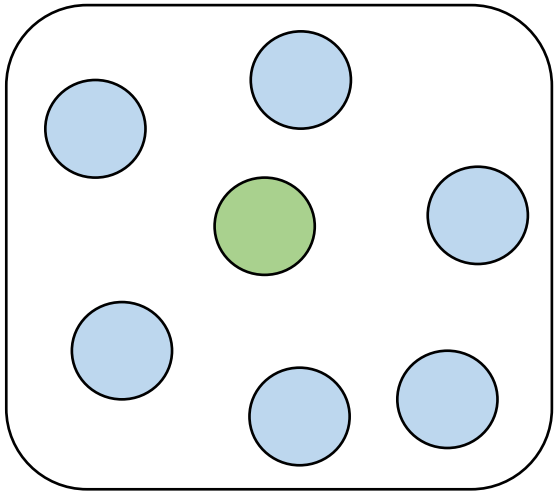
Level 2: Agreement – Consensus

- Consensus protocol to use in Total Ordering for Strong Consistency
  - Goal of consensus: achieve agreement between a group of nodes on a value
  - The value is the next update to apply for each <region, slot> (to order un-orderable updates)



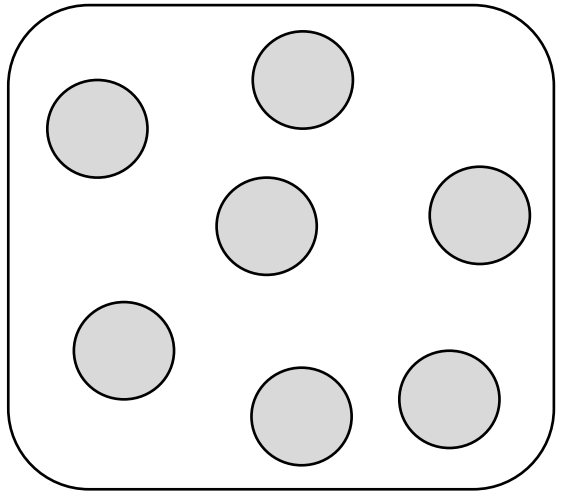
Level 2: Agreement – Consensus

- Consensus protocol to use in Total Ordering for Strong Consistency
  - Achieve agreement between a group of nodes on a value
  - The value is the next update to apply for each <region, slot> (to order un-orderable updates)
- Classic consensus algorithms: Paxos [TOCS'98] (and Raft [ATC'14])
  - Multiple rounds of leader election, voting, deciding, disseminating decision
  - Common case is connected network with reliable links, and (eventually) synchronous environment
  - Not suitable for highly transient networks
  - Simply using it in intermittently-connected network results in many frequent re-attempts of sessions



Level 2: Agreement – Consensus

- Consensus protocol to use in Total Ordering for Strong Consistency
  - Achieve agreement between a group of nodes on a value
  - The value is the next update to apply for each <region, slot> (to order un-orderable updates)
- Classic consensus algorithms: Paxos [TOCS'98] (and Raft [ATC'14])
  - Multiple rounds of leader election, voting, deciding, disseminating decision
  - Common case is connected network with reliable links, and (eventually) synchronous environment
- Consensus algorithms that tolerate loss and unreliable links
  - Assume asynchronous environment but with "good periods", where a message is eventually reachable to any node except for permanently-crashed ones
- One-Third Rule (OTR) algorithm [ICDCN'15]
  - Coordinator-less; nodes contribute (vote) and decide (two msg types)
  - Decide on 2/3 majority rule needed (population need to be known by all)
  - Allows 1/3 of nodes fail in one round
  - Decision same across all users, i.e., agreement property
  - Multiple rounds, but has the potential to finish in a single round
  - May take a long time due to frequent disconnections

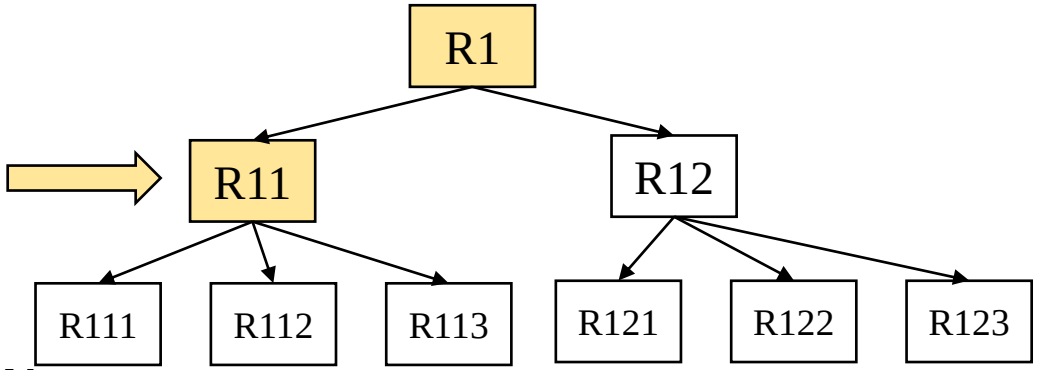


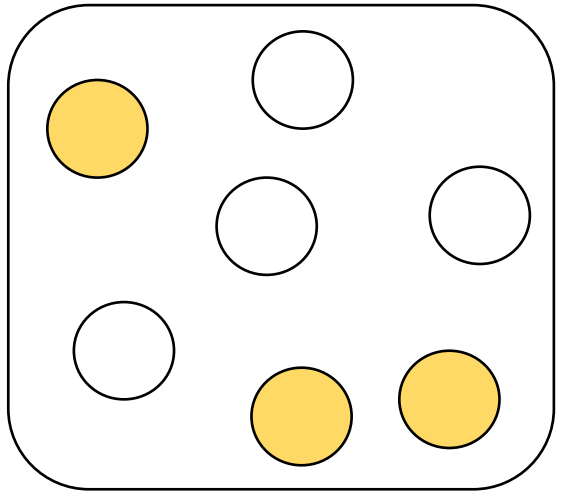
Contributors and Deciders

Level 2: Agreement – Consensus

- CoNICE uses OTR and enhances it in a number of ways

- Name-based selective consensus participation
  - Rather than all users in the network, use level 2 name-based interest profile (NBIP2) at every user to determine participation
  - Example: for slots of R11, only subscribers of R1 and R11
    - Others can still help with relaying, if allowed by their NBIP0





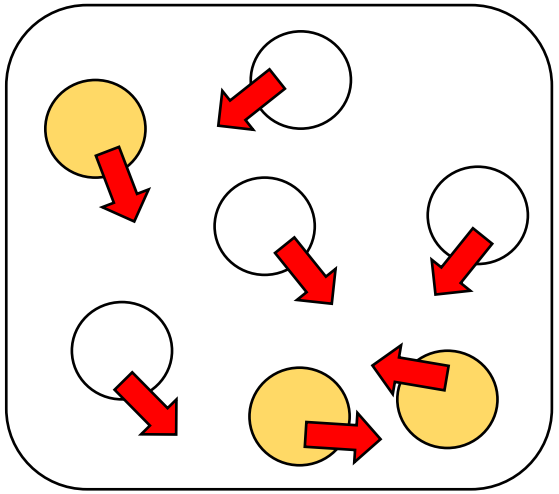
Name-based Group

Level 2: Agreement – Consensus

- CoNICE uses OTR and enhances it in a number of ways

- Name-based selective consensus participation

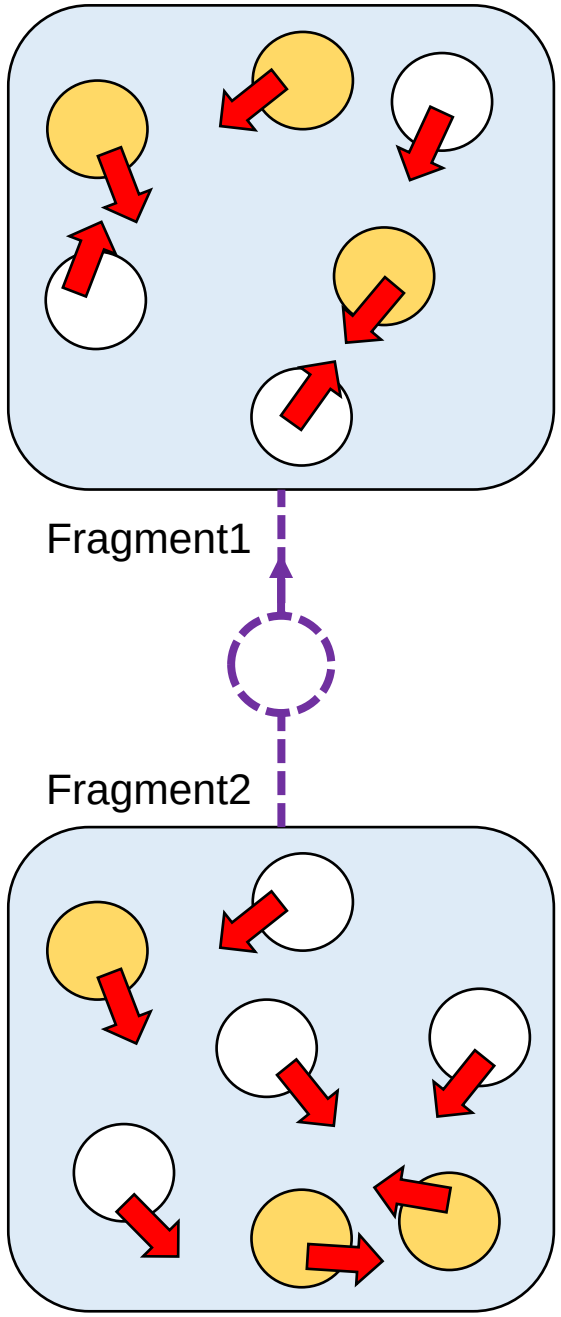
- Periodic reachability beaconing for population count estimation
  - Rather than a priori known, fixed count of all users (to calculate majority constraints), periodically announce self, and NBIP2 subscription
  - Each user estimates # of subscribers of each name-based group



Reachability Beacons

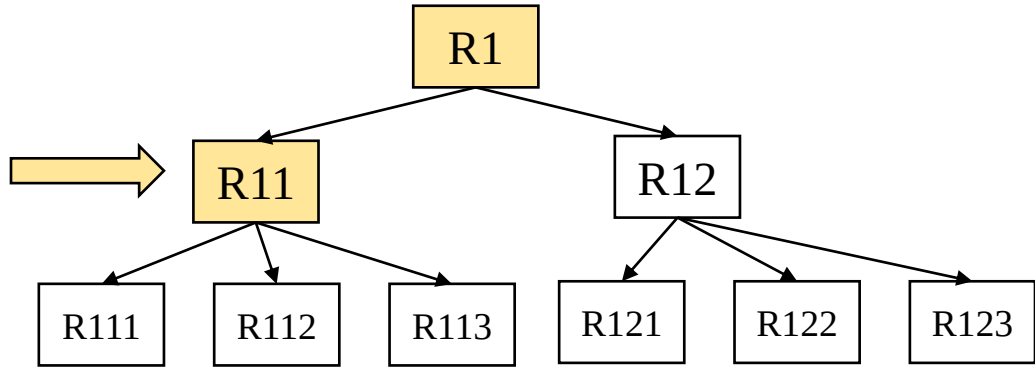
Level 2: Agreement – Consensus

- CoNICE uses OTR and enhances it in a number of ways
- Name-based selective consensus participation
- Periodic reachability beaconing for population count estimation
- Support decision invalidation in long-term fragmentation
  - Isolated network fragments (e.g., shelters), connected after a very a long-time → different decisions for same <region, slot> pairs (violating agreement property)
  - Rather than good period assumption in the whole network, support good period within fragments and decision invalidation
  - To break tie, decision from the higher populated fragment wins, the other should be updated and follow



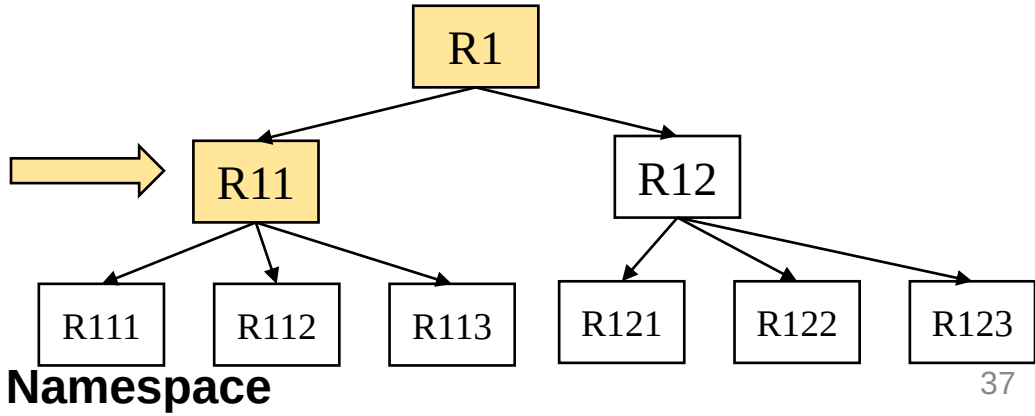
Level 2: Agreement – Consensus

- User X's Contribution message for <R11, 0>, proposes 'a' to fill it; first round, for n participants:
  - **contribution<X, R11, slot=0, round=1, value='a', population='n'>**
  - Disseminated for subscribers of R11 and above
  - Users create contribution: via initiation or are triggered by receiving others' contributions
  - When entering new round, delete contributions of previous rounds from buffer (less storage)



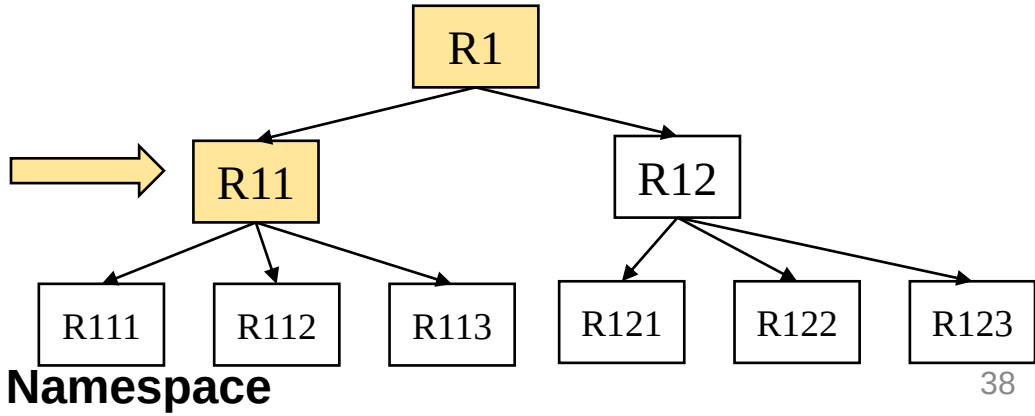
Level 2: Agreement – Consensus

- User X's Contribution message for <R11, 0>, proposes 'a' to fill it; first round, for n participants:
  - **contribution<X, R11, slot=0, round=1, value='a', population='n'>**
  - Disseminated for subscribers of R11 and above
  - Users create contribution: via initiation or are triggered by receiving others' contributions
  - When entering new round, delete contributions of previous rounds from buffer (less storage)
- Decision message:
  - **decision<X, R11, slot=0, value='a', population='n'>**
  - Disseminated for subscribers of R11 and above
  - Users decide and create decision message by reaching the 2/3 majority locally, or receiving decision from others



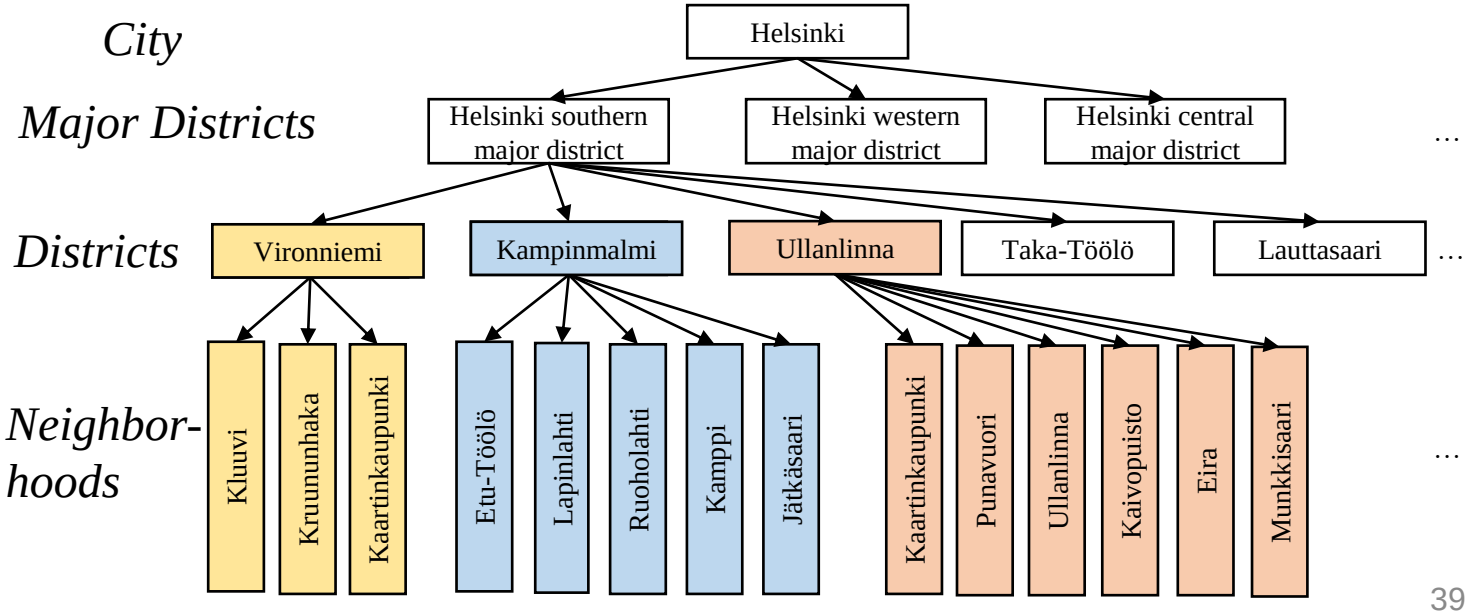
Level 2: Agreement – Consensus

- User X's Contribution message for <R11, 0>, proposes 'a' to fill it; first round, for n participants:
  - **contribution<X, R11, slot=0, round=1, value='a', population='n'>**
  - Disseminated for subscribers of R11 and above
  - Users create contribution: via initiation or are triggered by receiving others' contributions
  - When entering new round, delete contributions of previous rounds from buffer (less storage)
- Decision message:
  - **decision<X, R11, slot=0, value='a', population='n'>**
  - Disseminated for subscribers of R11 and above
  - Users decide and create decision message by reaching the 2/3 majority locally, or receiving decision from others
- Users build up their 'snapshots' based on region-slot decisions
- This protocol ensures correct total ordering across relevant users eventually
  - More details and proofs in the paper!



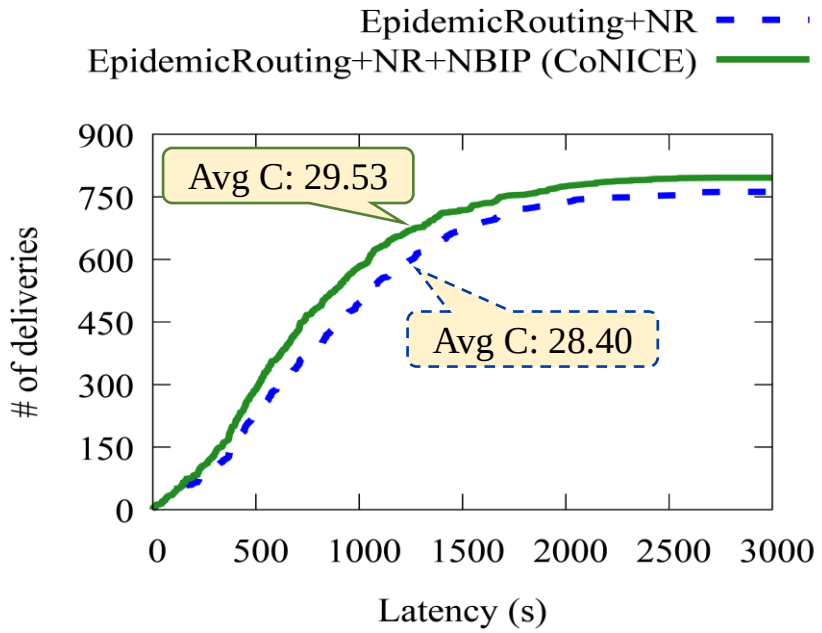
Experimentation

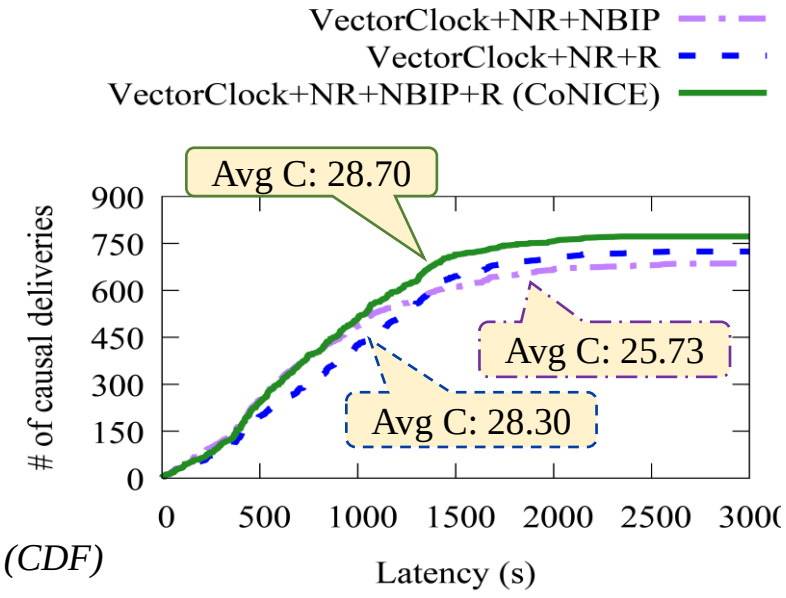
- Subset of City of Helsinki; Namespace:
  - city → major district → district → neighborhood
- Subset (in southern major district): 3 districts; world size: 4500x3400 meters
- 30 mobile first responders; 10 responsible per district
- Additional benevolent mules: 500 civilians and 50 vehicles for helping delivery (not participants of consensus sessions)
- Each first responder creates 3 updates (1KB msgs)
- Experiments: implemented in the ONE (Opportunistic Network Environment) simulator



Simulation results: Gossiping & Causal Ordering

- With use of naming (named region-ing (NR), name-based interest profiling (NBIP)), and reactive causal ordering (R), CoNICE
  - Higher replication completeness (C) of relevant info (deliveries) and causal completeness at first responders (~5-10%)
  - Achieves better average latencies in both

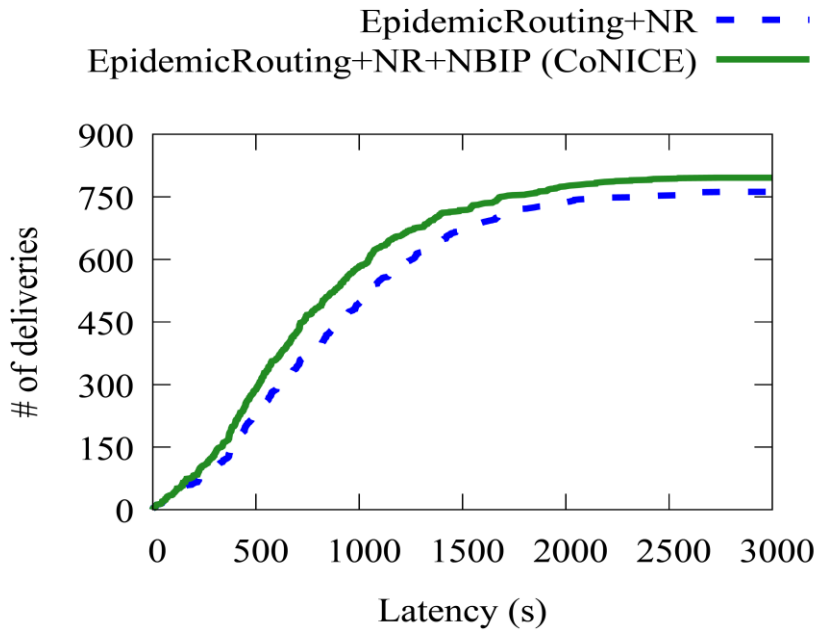




All diagrams are cumulative (CDF)

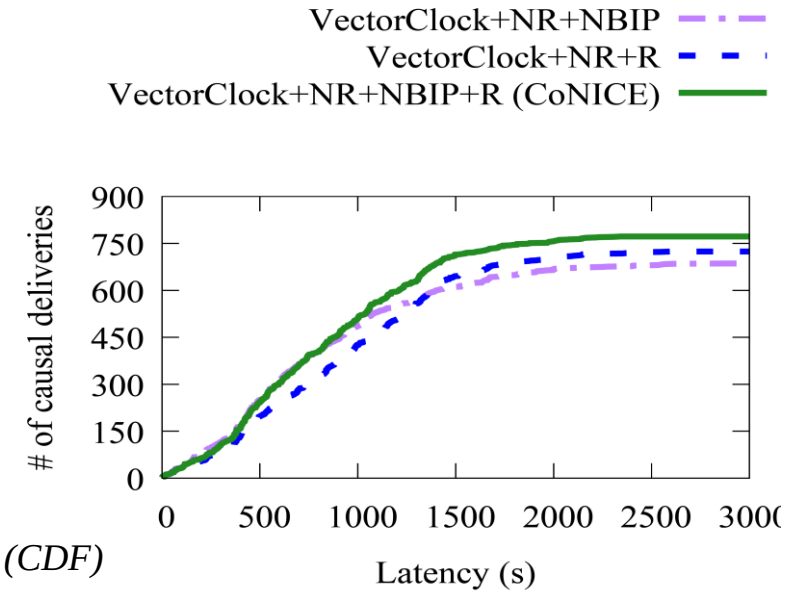
Simulation results: Gossiping & Causal Ordering

- With use of naming (named region-ing (NR), name-based interest profiling (NBIP)), and reactive causal ordering (R), CoNICE
  - Higher replication completeness (C) of relevant info (deliveries) and causal completeness at first responders (~5-10%)
  - Achieves better average latencies in both
  - Similar number of total relays, i.e., total network traffic
  - CoNICE's naming more efficient gossiping and causal ordering



Approach	Total Relays
EpidemicRouting	49,612
EpidemicRouting+ NR	50,123
EpidemicRouting+ NR+NBIP (CoNICE)	48,612

Approach	Total Relays
VectorClock+NR+NBIP	98,485
VectorClock+R	108,289
VectorClock+NR+R	88,134
VectorClock+NR+NBIP+ R (CoNICE)	89,792

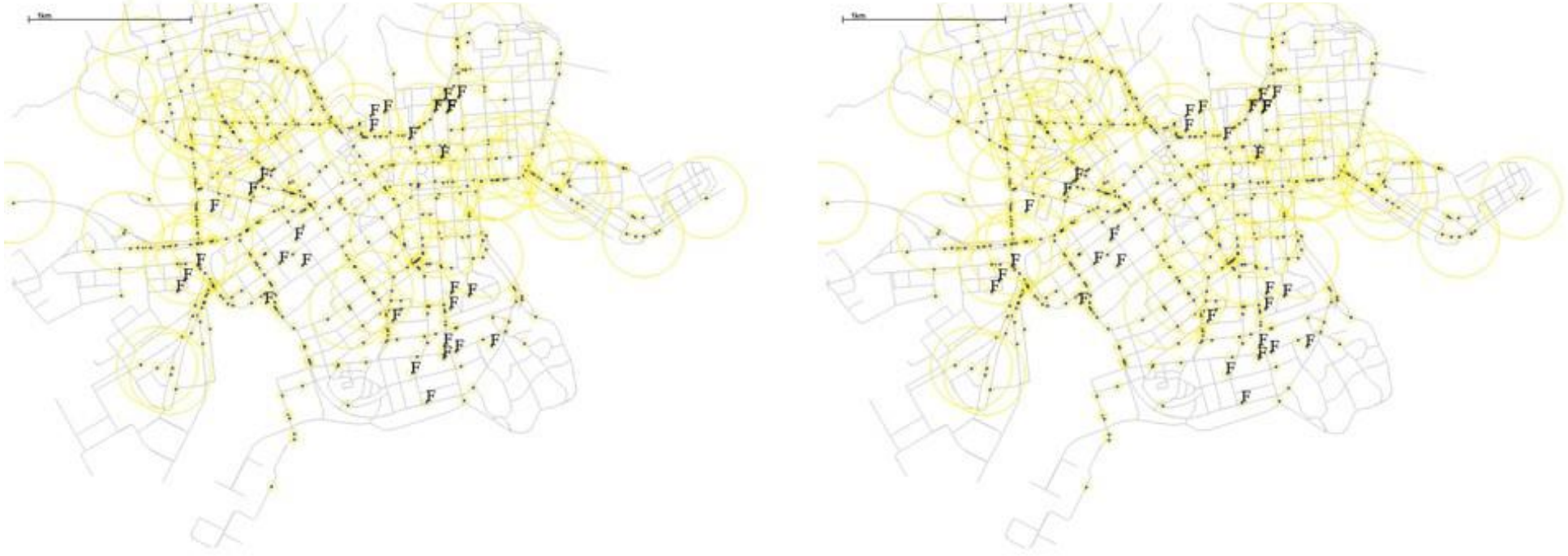


All diagrams are cumulative (CDF)

Simulation results on Consensus

OTR+NR

OTR+NR+NBIP (CoNICE)



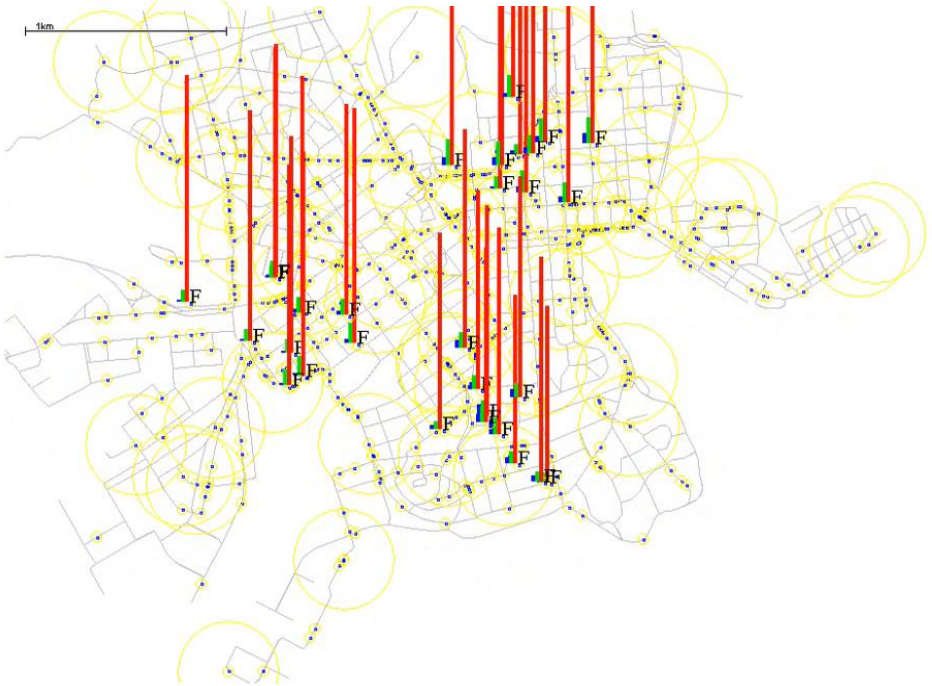
How many strong view slots filled for regions relevant for the first responder

How many strong view slots filled for regions at the first responder

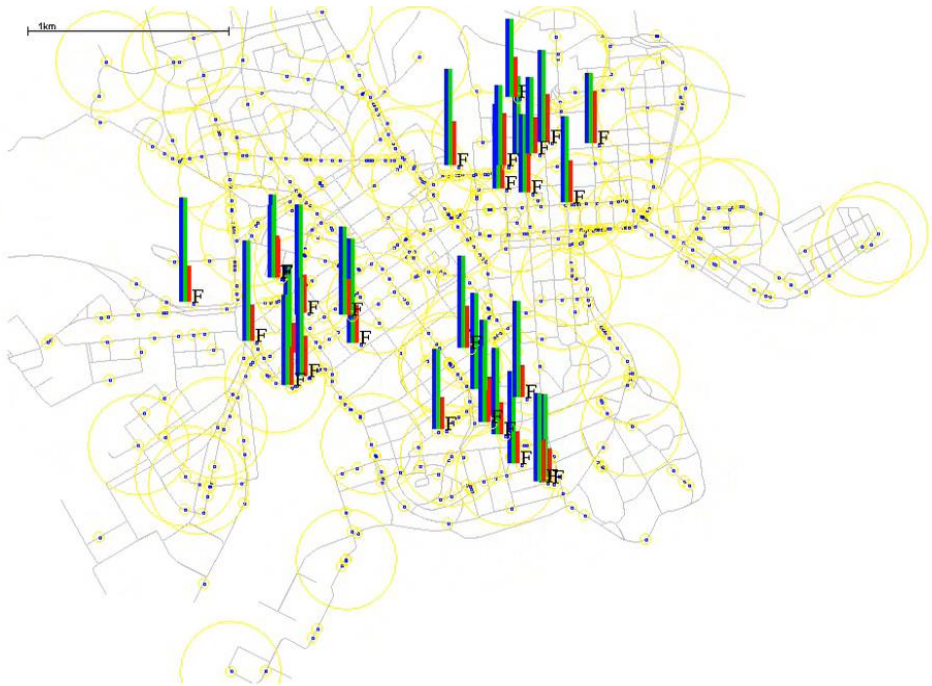
How many MB of buffer occupied at first responder

Simulation results on Consensus

OTR+NR



OTR+NR+NBIP (CoNICE)



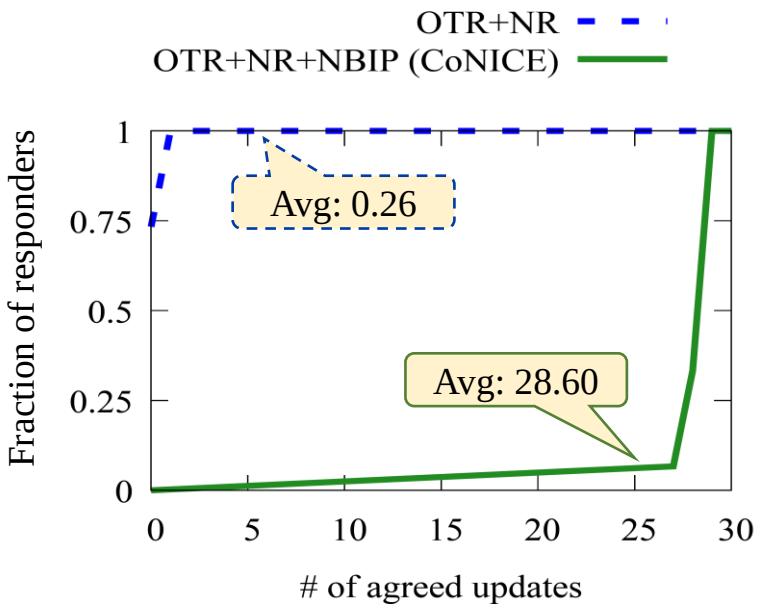
How many strong view slots filled for regions relevant for the first responder

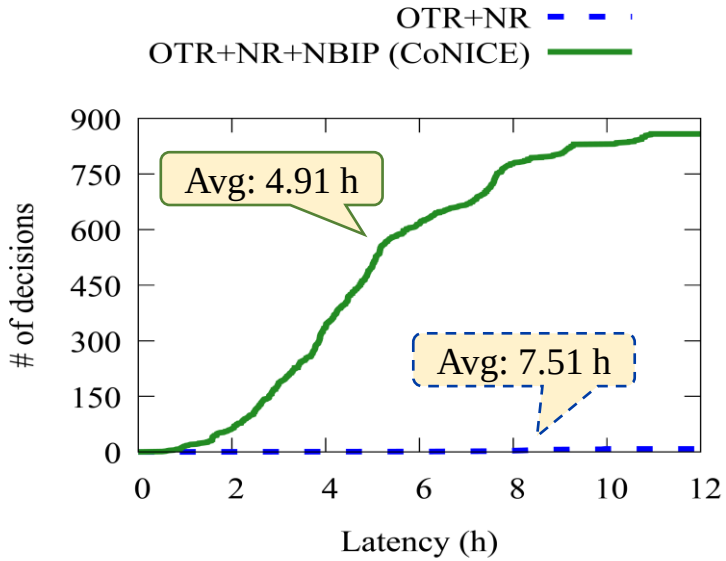
How many strong view slots filled for regions at the first responder

How many MB of buffer occupied at first responder

Simulation results on Consensus

- With use of naming (named region-ing (NR), name-based interest profiling (NBIP)), our solution
  - Achieves higher agreement completeness of relevant info at first responders (~100X using NBIP to identify groups)
  - More relevant decisions deliveries, better latency at first responders (~2X using NBIP)
    - High absolute values justify having a moderate view in the meantime

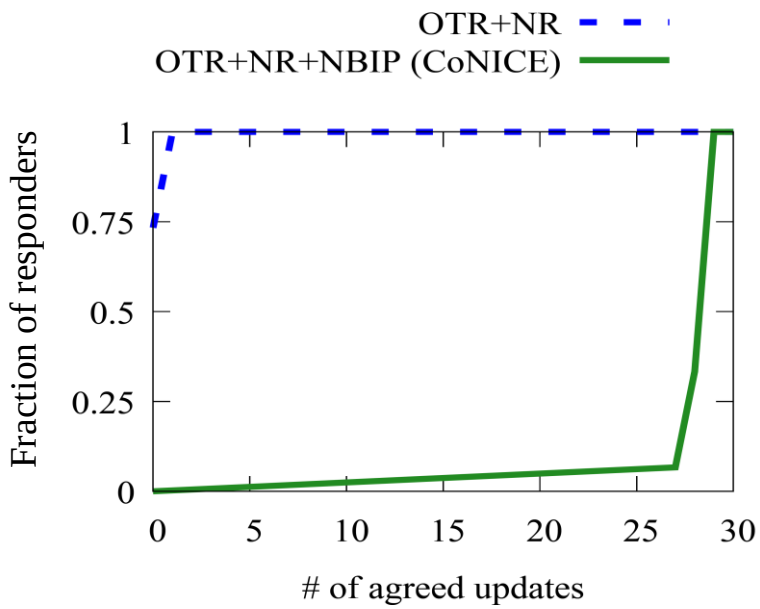




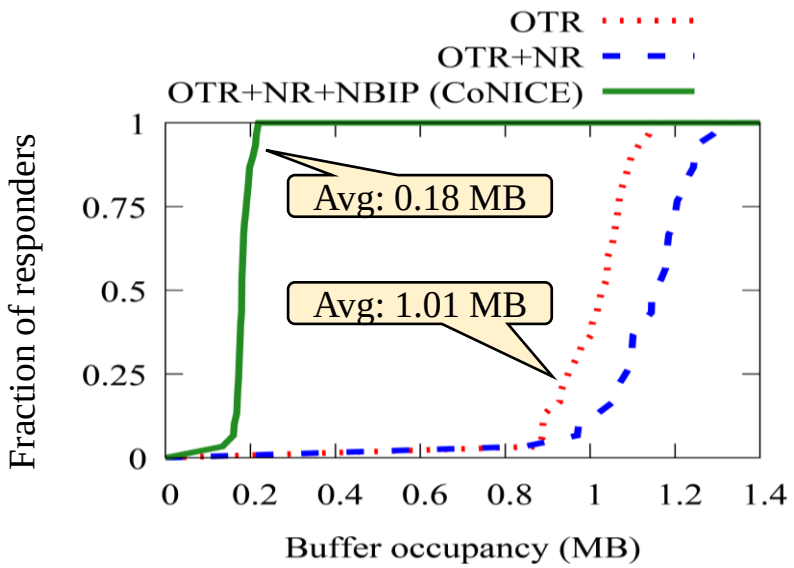
All diagrams are cumulative (CDF)

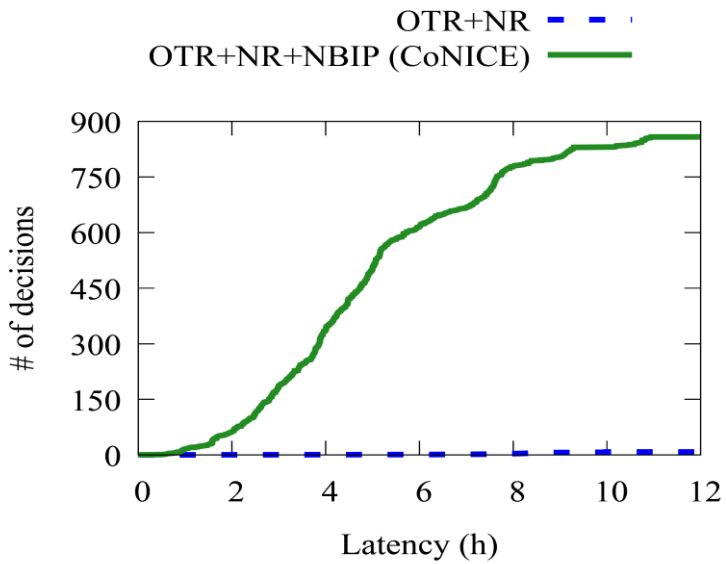
Simulation results on Consensus

- With use of naming (named region-ing (NR), name-based interest profiling (NBIP)), our solution
  - Achieves higher agreement completeness of relevant info at first responders (~100X using NBIP to identify groups)
  - More relevant decisions deliveries, better latency at first responders (~2X using NBIP)
  - Lowers buffer consumption at first responders (~5X with naming)
  - Similar number of total relays in the network
  - CoNICE's name-based grouping achieve agreement in a more efficient and effective manner.



Approach	Total Relays
OTR	3,489,035
OTR+NR	3,512,598
OTR+NR+NBIP (CoNICE)	3,504,557

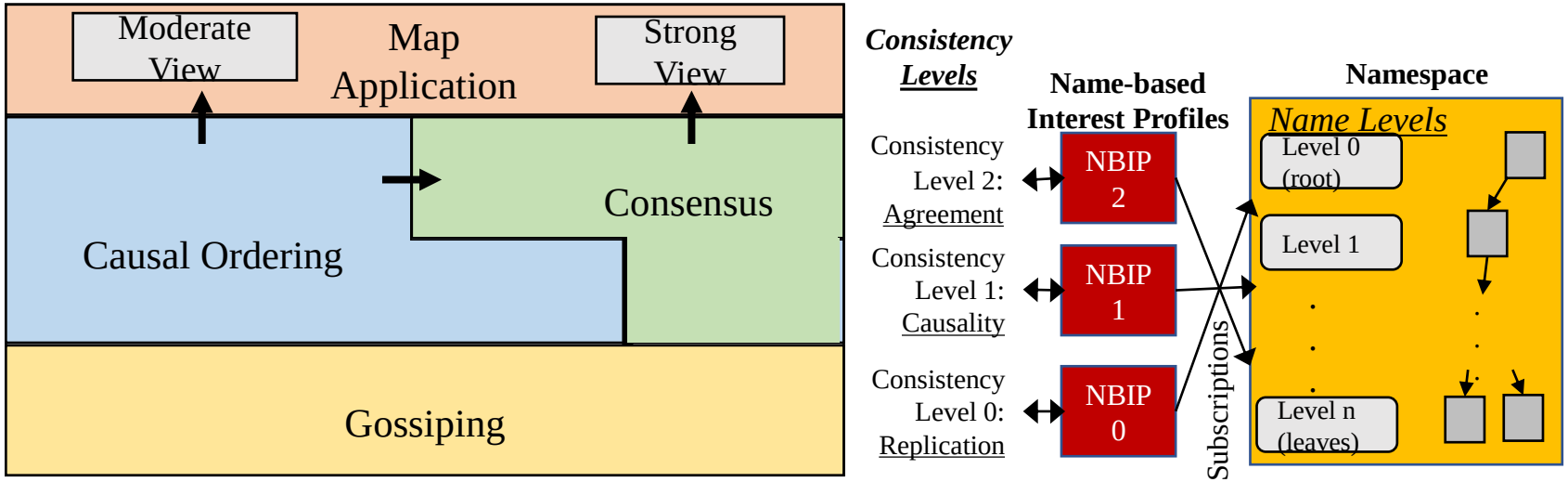




All diagrams are cumulative (CDF)

Summary

- CoNICE: a framework to ensure consistent dissemination of updates among users in intermittently-connected, infrastructure-less environments, e.g., emergency response
- Multi-level consistency supporting causal ordering and consensus
  - Flexible trade-off between completion time and degree of consistency during disasters
- Multi-level naming schema for fine-grained subscription
  - Reduces user storage usage for D2D buffering
  - Achieves higher completeness of strong consistency and faster consensus convergence across relevant users



Supported by US NIST (award 70NANB17H188) and US NSF (grant CNS-1818971)