ConcoRDanT+STREAMS M24 meeting, Nantes, November 2012

Programming CRDTs on SwiftCloud

Marek Zawirski	UPMC-LIP6 & INRIA
Annette Bieniusa	U. Kaiserslautern
Valter Balegas	UNL
Nuno Preguiça	UNL
Sérgio Duarte	UNL
Marc Shapiro	INRIA & UPMC-LIP6
Carlos Baquero	U. Minho

Programming SwiftCloud – example

```
void befriend(CRDTIdentifier other) {
  TxnHandle txn = swiftSession.beginTxn(ISOLATION_LEVEL,
                                        CACHE_POLICY, READ_ONLY);
  // Obtain new friend's data
  User friend = ((Register<User>) txn.get(other, CREATE)).getValue();
  // Register him as my friend
  Set<Id> friends = (Set<Id>) txn.get(currentUser.friends, CREATE,
                                      updatesSubscriber);
  friends.insert(other);
  // Register me as his friend
  Set<Id> hisFriends = (Set<Id>) txn.get(friend.friends, CREATE,
                                         updatesSubscriber);
  hisFriends.insert(currentUser);
  txn.commit(visibilityHook);
}
```

```
class User implements Copyable {
   CRDTIdentifier userId;
   String fullName; long birthday;
   CRDTIdentifier msgs;
   CRDTIdentifier friends;
   CRDTIdentifier inFriendReq;
   CRDTIdentifier outFriendReq;
}
```

Defining a new CRDT in SwiftCloud

- SwiftCloud currently uses a hybrid state+operation object model

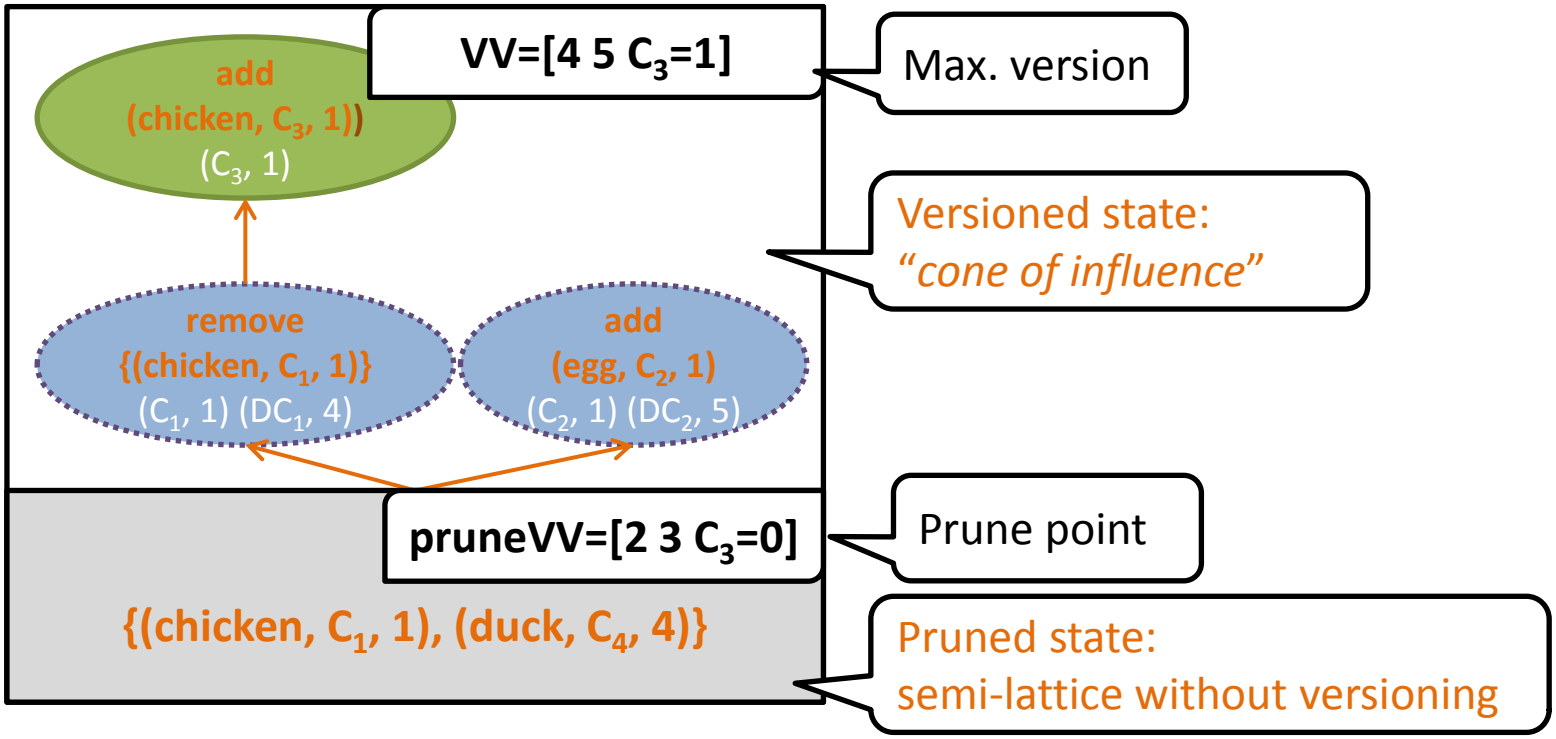
- Base shared functionality implemented by the platform
  - Transactions
  - Timestamp & aliases
  - Causality tracking, updates dissemination
  - Updates idempotence*
  - Pruning stability protocols
  - TODO: generic support for tests?

- Data type designer's job restricted to type-specific definitions
  - **State and updates definition**
  - **Pruning**
  - **Snapshot**
  - **… and understanding** causality tracking to some extend

CRDT model in more details: OR-Set

Versioned CRDT object

```
class ORSet<V>
implements CRDTVersioned {...
```



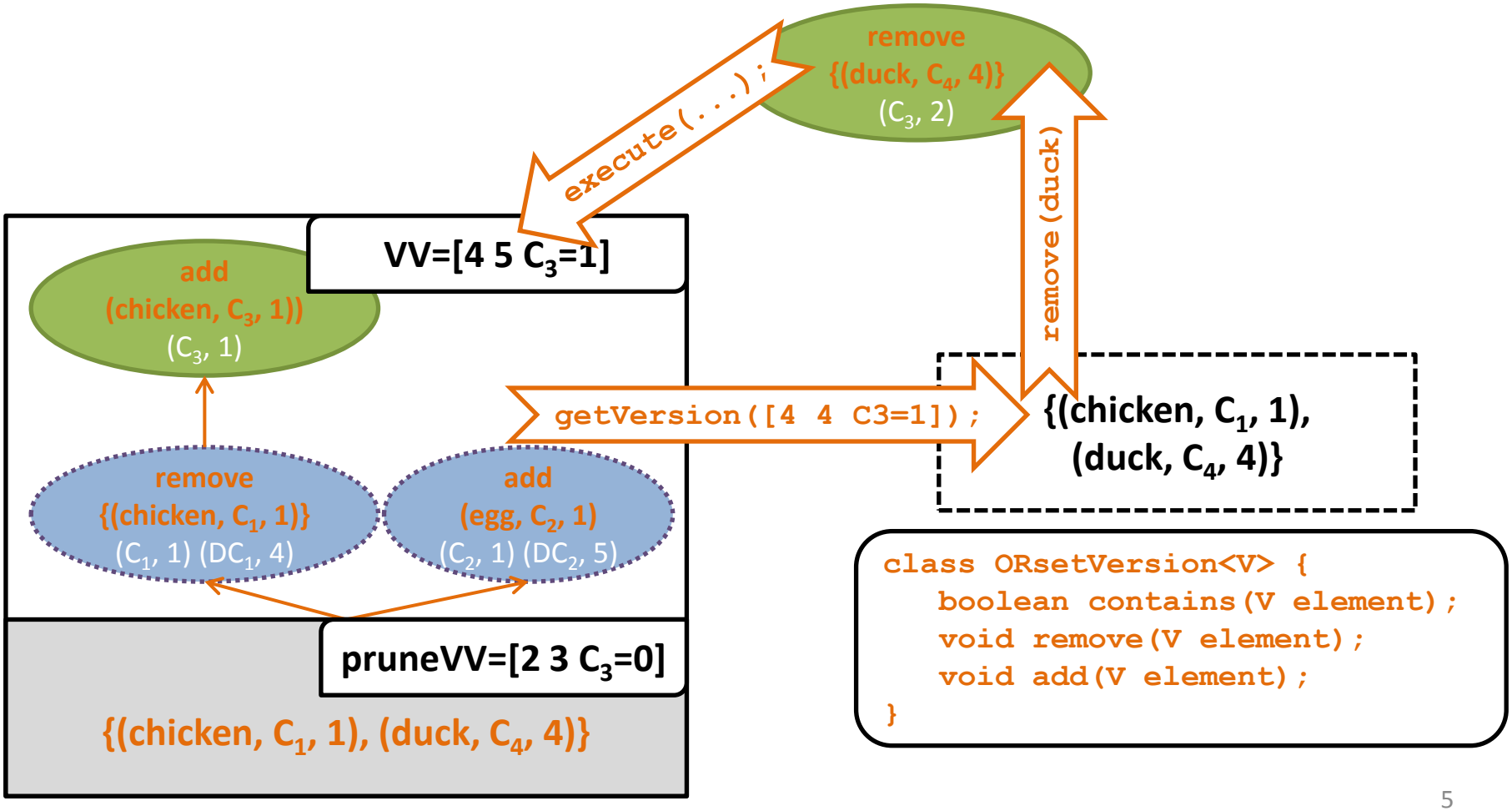
CRDT model in more details: OR-Set

Versioned CRDT object

```
class ORSet<V>
implements CRDTVersioned {...
```

Transaction local object view

```
class ORSetVersion<V>
implements Version<ORSet> {...
```




```
class ORsetVersion<V> {
    boolean contains(V element);
    void remove(V element);
    void add(V element);
}
```

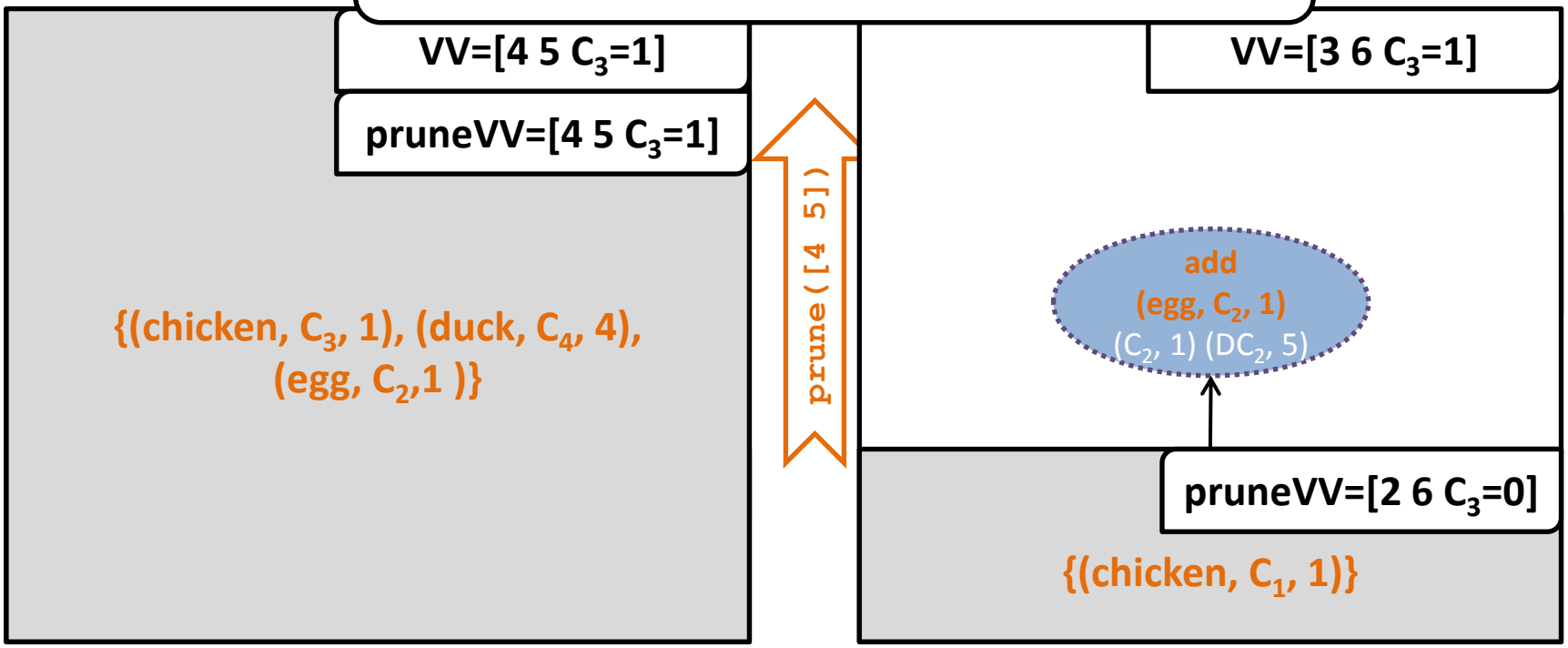
CRDT model in more details: OR-Set

Versioned CRDT object

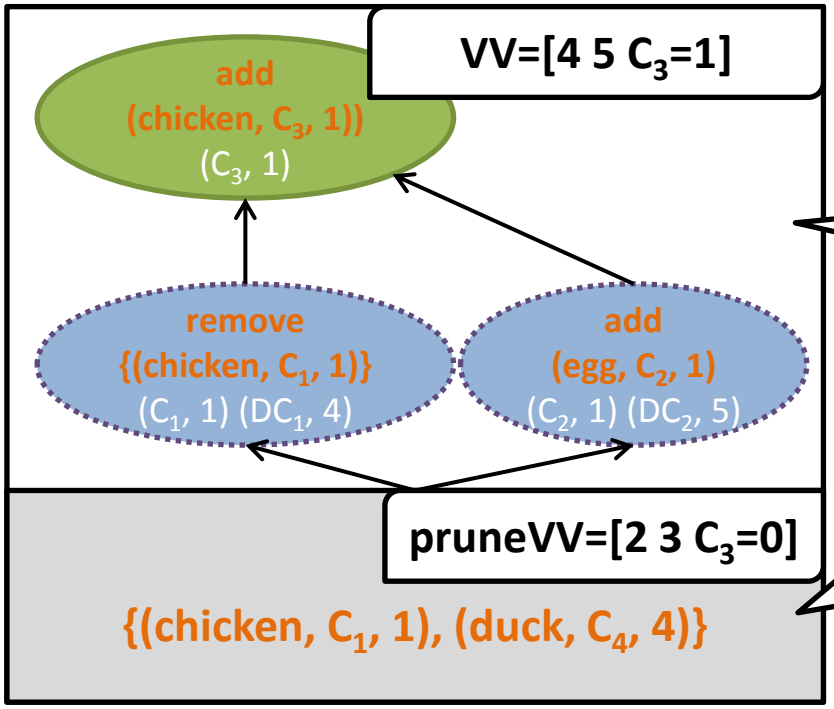
```
class ORSet<V>
implements CRDTVersioned {...
```

Merge definition (a bit involved due to aliasing):

```
void mergePayload(anotherObject);
```



WIP: Simplified CRDT model



Versioned part:
log of updates managed by the platform

Pruned part – two alternatives
1) semi-lattice – mergeable, more difficult
2) state – not mergeable, simpler
Can we live without a merge?

Open problem: using linear extensions for log