Toward Systematic Detection and Resolution of Network Control Conflicts

Dennis Volpano

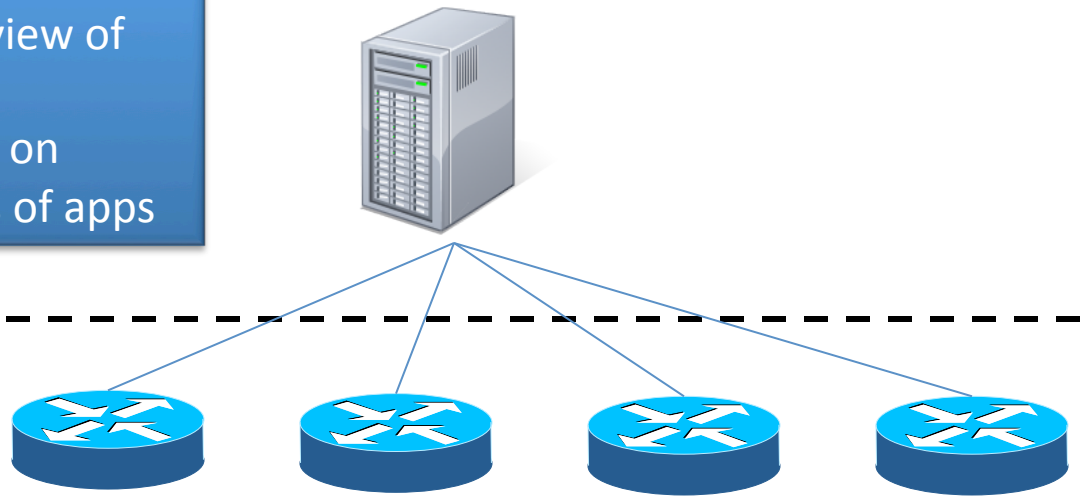
Joint work with Xin Sun and Geoffrey Xie

What's the problem?

Control logic:
- Implements application, e.g., load balancing or power management

Control logic | Control logic | Control logic

Controller:
- Collects & provides a global view of network for control logics
- Runs control logics which act on switches to achieve the goals of apps

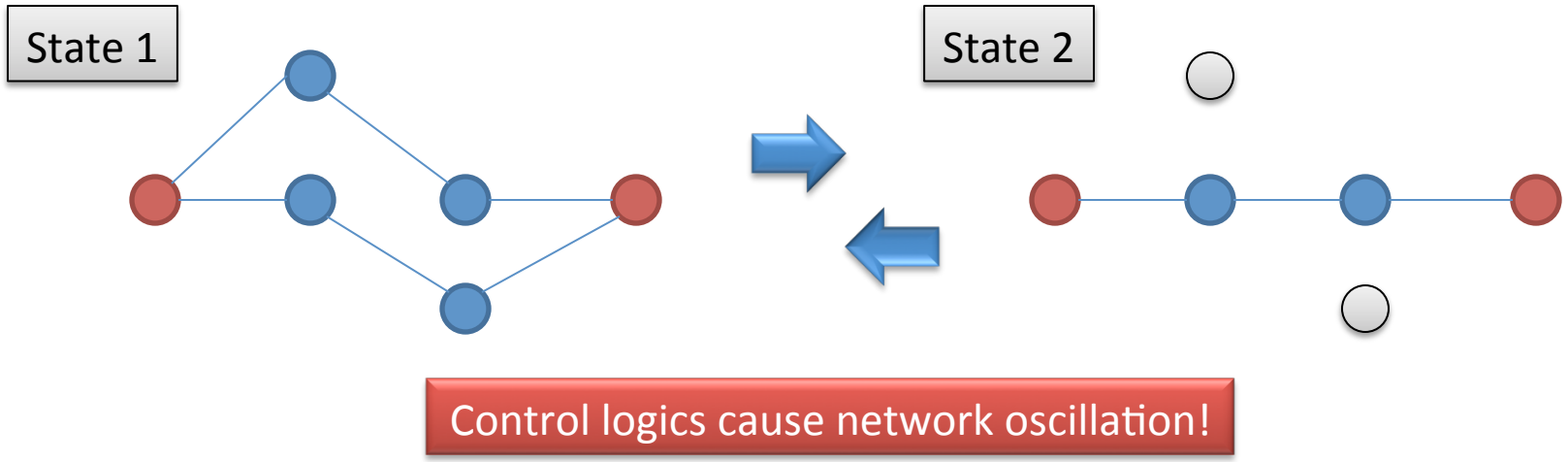


Control logics can take conflicting actions resulting in network instability

Example due to Franck Le

Control logic 1: (fast reroute)
- Need at least two disjoint paths between two critical nodes

Control logic 2: (energy saving)
- Reroute traffic to turn off network devices as much as possible

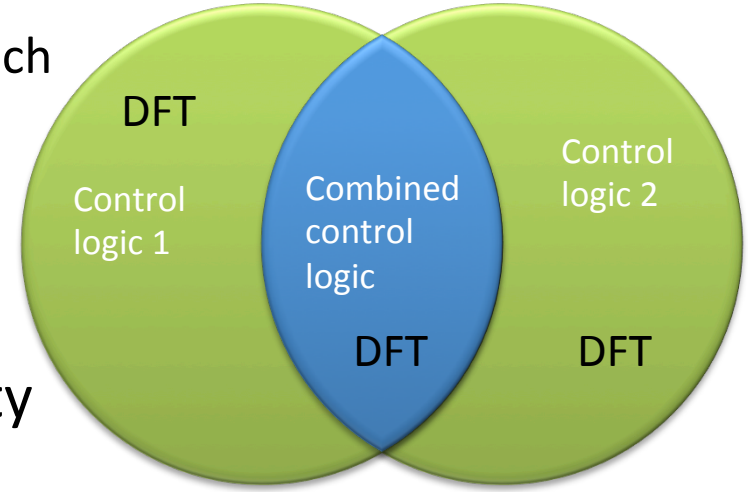


- Want to know effect of combining logics BEFORE they're deployed
- This has been the focus of our research

Solution

Use Det Finite State Transducers to express control logics; a DFT has 3 major advantages:

1) Reveals a *stable operating region*
   - network operating conditions under which the control logic never acts
2) Their intersection is computable
   - intersection is also a control logic that can run on an SDN controller
3) Whether an intersection has an empty stable region is decidable
   - not even semi-decidable for Python
   - nonempty implies an operating condition under which *all* control logics are dormant
   - can inspect the stable region BEFORE deployment
   - make decision to use based on region's narrowness



Conclusion

- We want alternatives to Python for defining control logics that admit formal processing
- We want expressiveness and decidability
  - control logic equivalence
  - emptiness of intersection's stable region
- We want scalable formal reasoning
  - with DFTs, the invariant of the conjunction of states is the conjunction of invariants

- DFTs are limited to regular properties
  - problem in practice? Needs more study