Formal Verification of Distributed Algorithms

K. Mani Chandy[1] Brian Go[1] Sayan Mitra[2] Jerome White[1]

[1]California Institute of Technology
[2]University of Illinois at Urbana-Champaign

Overview

Improve distributed software **reliability**

Ease **proof** burden for concrete algorithms

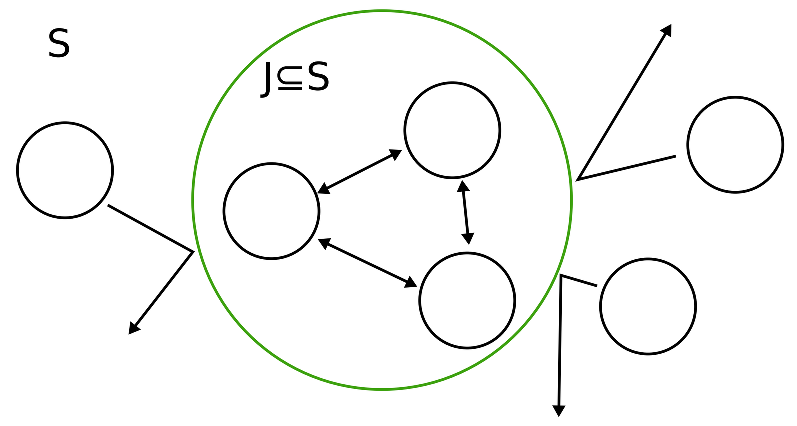
Stepwise **refinement** and program transformation

Organize **library** of theorems for distributed systems (PVS)

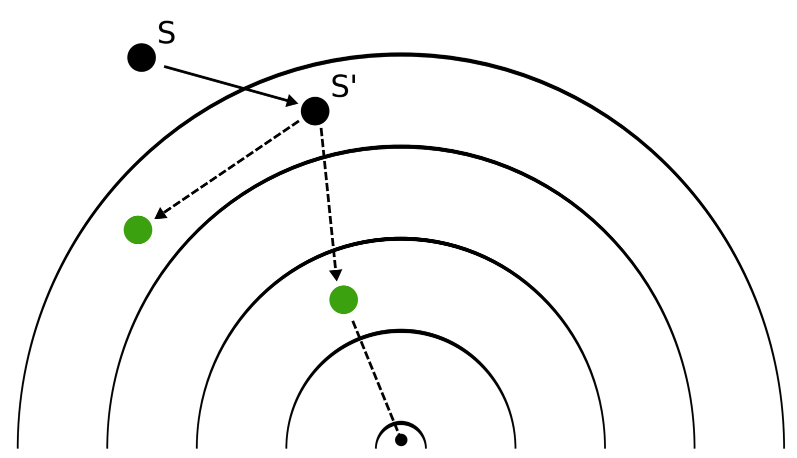
Transform theorem proven code into executable code (Java/Erlang)

Improve **teaching**

Local Interactions...



... Global Implications



Abstraction

Abstractions

Incorporate fundamental abstractions of distributed algorithms that are generic enough to be applicable to a large class of problems

Fundamental Functions

Binary operator	∘: T → T
Composition function	f: S, {K⊆S}, ∘ → T
Transition Predicate	t: S, S → Bool

Correctness Properties

Safety: invariant function with respect to start state

$$t(S^0, S) \Rightarrow f(S, \{K=S\}, \circ) = f(S^0, \{K=S\}, \circ)$$

Progress: variant function to a well founded set

$$\rho: T \rightarrow T \mid T \text{ is well founded}$$

Application: Consensus

Abstractions

Given state S with n agents, want final state to be function of start state S*: $\forall j \in S: S^*(j) = f(S^0)$

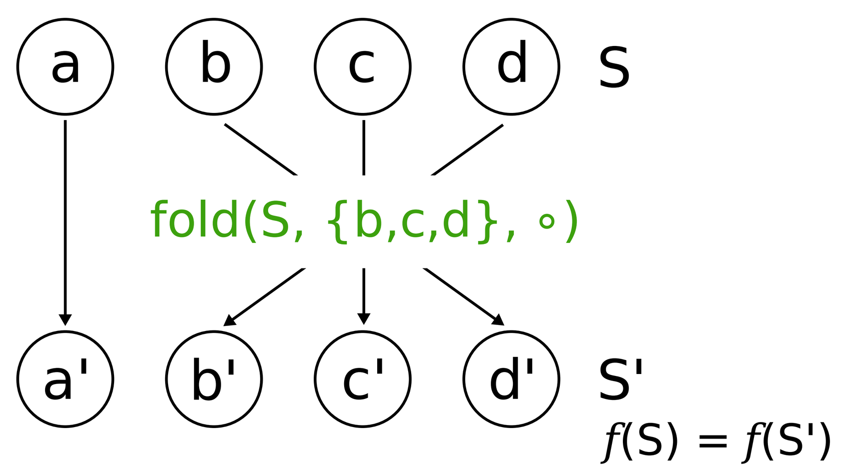
∘: commutative ∧ associative ∧ (idempotent ∨ super idempotent)

f is aggregation (fold) of composable binary operator ∘
$f(S) = S(0) \circ S(1) \circ \ldots \circ S(n)$

$$\equiv \text{fold}(S, \circ) = \begin{cases} S(0) \text{ if } |S| = 1 \\ \text{fold}(S, \{J \setminus S(n)\}, \circ) \circ S(n) \text{ otherwise} \end{cases}$$

Transition	$t(S, S') \equiv \forall j: \exists J: j \in J \wedge S'(j) = \text{fold}(S, J, \circ)$
Safety	$\forall S, S': t(S, S') \Rightarrow \text{fold}(S, J, \circ) = \text{fold}(S', J, \circ)$
Progress	$\lvert\{ a \text{ where } S(a) = \text{fold}(S^0, \{a\}, \circ) \}\rvert$

Visual Transition



Proof Refinement

∘(S, S) → T

fold(S, J, ∘) → T

commutative, associative

min/max — gcd/lcm (idempotent)

Average — Convex Hull (super idempotent)

Object Refinement

Operator

FoldableOperator

Min — GCD — ... — Cvx Hull

Application: Graphs

Abstractions

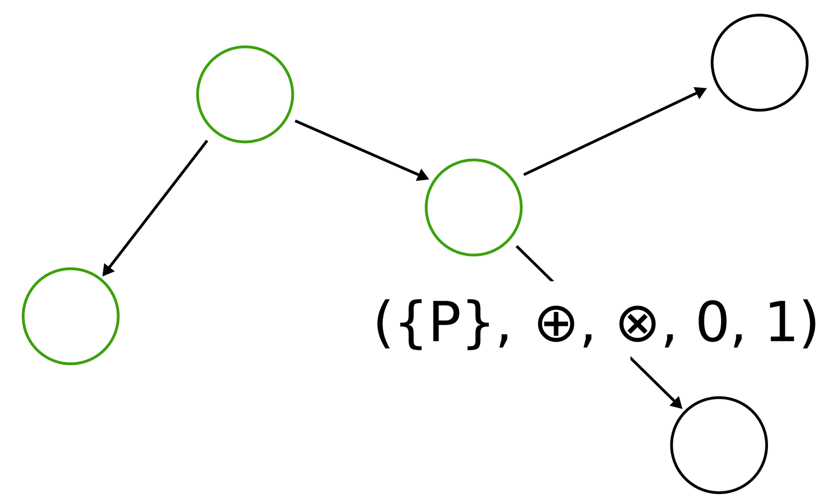
Given source graph G and root vertex R, produce target graph that solves given algebraic path problem

∘: semiring with additional constraints
idempotent ($x \oplus x = x$), and
partial order ≤ over {P}

f is a depth first graph traversal method

Transition	$t(S, S') \equiv G'(v) = G(v) \oplus (G(u) \otimes E[u, v])$
Safety	$\forall S, S': t(S, S') \Rightarrow G'(v) \leq G(v)$
Progress	$\lvert\{E[u,v] \mid \text{untraversed}\}\rvert$

Visual Transition



Proof Refinement

∘({P}, ⊕, ⊗, 0, 1) → T

traverse(S, E[u,v], ∘) → G

idempotent semirings

reachability ({0, 1}, ∨, ∧, 0, 1)

shortest path ({ℝ⁺}, min, +, 0, ∞⁺)

Object Refinement

Semiring

GraphTraverser

Reachability — ShortestPath

Acknowledgements

The Lee Center for Advanced Networking

Infospheres Lab

References

K. Mani Chandy, Brian Go, Sayan Mitra, and Jerome White. "Towards Verified Distributed Software Through Refinement of Formal Archetypes." Second IFIP Working Conference on Verified Software: Workshop on Experiments. October 6-9, 2008.

K. Mani Chandy, Brian Go, Sayan Mitra, Concetta Pilotto, and Jerome White. "Guaranteeing Convergence of Distributed Systems: From Specification to Implementation via Refinenement." Submitted: Formal Aspects of Computing.