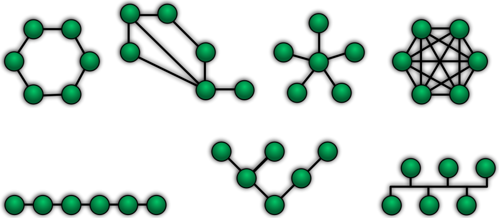
An Abstraction Technique for Parameterized Model Checking of Leader Election Protocols: Application to FTSP

Ocan Sankur, Jean-Pierre Talpin

Irisa, CNRS, Rennes

Parameterized Abstraction for Leader Election Protocols

Goal: Model check a leader election protocol on **arbitrary** network topologies



Verify that for all network topologies and initial states, a unique leader is eventually elected

(Actually, we will verify all network topologies **with given diameter**)

Content of this Work

1. An abstraction technique for parameterized model checking such protocols
2. Application to a specific protocol

Content of this Work

1. An abstraction technique for parameterized model checking such protocols
2. Application to a specific protocol

Case Study: Flooding-Time Synchronization Protocol (FTSP)

Fault-tolerant distributed protocol for maintaining time in wireless sensor networks.

Has two features:

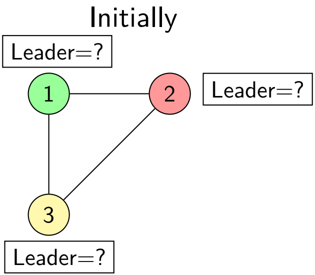
- Maintains a unique leader, recovers in case of link/node failures
- Smoothly synchronizes the clocks over the network with the clock of the leader

Today, we consider the leader election part of FTSP: Verify that a unique leader is eventually elected

Flooding-Time Synchronization Protocol

Flooding-Time Synchronization Protocol (FTSP)

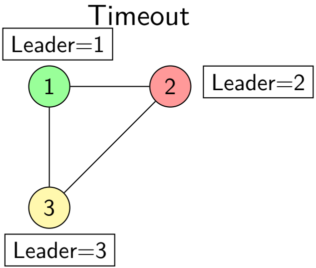
– Nodes have unique identifiers but execute the same program
– The network eventually elects the node with the least ID as the **leader**
– Fault tolerant: any node that hasn't heard from the leader for a while timeouts and declares itself leader



Flooding-Time Synchronization Protocol

Flooding-Time Synchronization Protocol (FTSP)

– Nodes have unique identifiers but execute the same program
– The network eventually elects the node with the least ID as the **leader**
– Fault tolerant: any node that hasn't heard from the leader for a while timeouts and declares itself leader

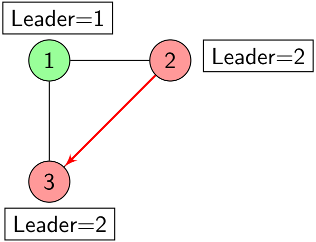


Flooding-Time Synchronization Protocol

Flooding-Time Synchronization Protocol (FTSP)

– Nodes have unique identifiers but execute the same program
– The network eventually elects the node with the least ID as the **leader**
– Fault tolerant: any node that hasn't heard from the leader for a while timeouts and declares itself leader

2 communicates with 3

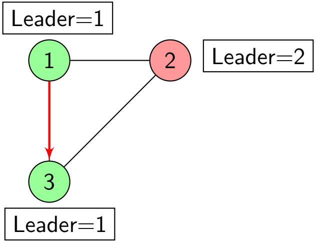


Flooding-Time Synchronization Protocol

Flooding-Time Synchronization Protocol (FTSP)

– Nodes have unique identifiers but execute the same program
– The network eventually elects the node with the least ID as the **leader**
– Fault tolerant: any node that hasn't heard from the leader for a while timeouts and declares itself leader

1 communicates with 3

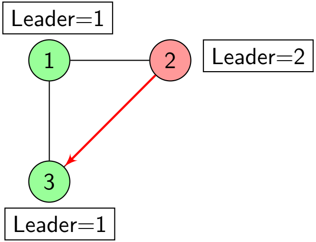


Flooding-Time Synchronization Protocol

Flooding-Time Synchronization Protocol (FTSP)

– Nodes have unique identifiers but execute the same program
– The network eventually elects the node with the least ID as the **leader**
– Fault tolerant: any node that hasn't heard from the leader for a while timeouts and declares itself leader

2 communicates with 3: Ignored!

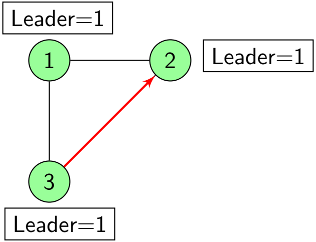


Flooding-Time Synchronization Protocol

Flooding-Time Synchronization Protocol (FTSP)

– Nodes have unique identifiers but execute the same program
– The network eventually elects the node with the least ID as the **leader**
– Fault tolerant: any node that hasn't heard from the leader for a while timeouts and declares itself leader

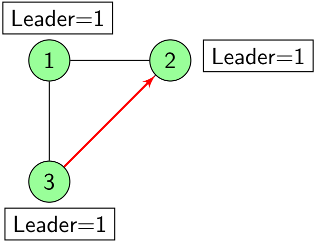
3 communicates with 2: Convergence!



Flooding-Time Synchronization Protocol

Flooding-Time Synchronization Protocol (FTSP)

– Nodes have unique identifiers but execute the same program
– The network eventually elects the node with the least ID as the **leader**
– Fault tolerant: any node that hasn't heard from the leader for a while timeouts and declares itself leader



+ Several local variables, message numbers, etc.

Previous Verification Results

Previous work: Model checking that a unique leader is eventually elected:

- a few fixed topologies (max: 7 nodes in ~ 1 hour)
- perfectly synchronized clocks
- synchronous message broadcast

Kusy, Abdelwahed 2006, McInnes 2009, Tan, Zhao, Wang 2010

Previous Verification Results

Previous work: Model checking that a unique leader is eventually elected:

- a few fixed topologies (max: 7 nodes in $\sim$ 1 hour)
- perfectly synchronized clocks
- synchronous message broadcast

Kusy, Abdelwahed 2006, McInnes 2009, Tan, Zhao, Wang 2010

Parameterized verification is difficult:

Arbitrary topology (not a complete graph), distinct node identifiers: no symmetry

Previous Verification Results

Previous work: Model checking that a unique leader is eventually elected:

- a few fixed topologies (max: 7 nodes in ~ 1 hour)
- perfectly synchronized clocks
- synchronous message broadcast

Kusy, Abdelwahed 2006, McInnes 2009, Tan, Zhao, Wang 2010

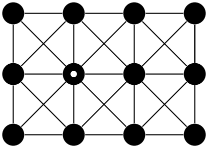
Parameterized verification is difficult:
Arbitrary topology (not a complete graph), distinct node identifiers: no symmetry

Present work

- Arbitrary network topology within given diameter K
  e.g. we can check a grid network with 169 nodes in 15 minutes
- Deviating clocks
- Synchronous or asynchronous broadcast

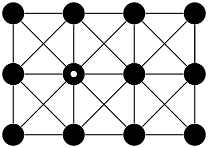
Abstraction Idea for Parameterized Verification

How the leader is propagated:



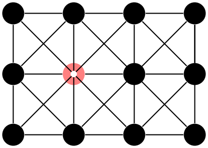
Abstraction Idea for Parameterized Verification

How the leader is propagated:



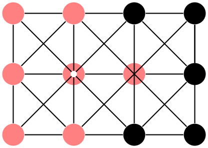
Abstraction Idea for Parameterized Verification

How the leader is propagated:



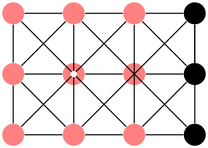
Abstraction Idea for Parameterized Verification

How the leader is propagated:



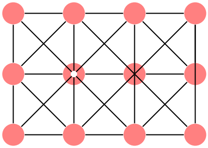
Abstraction Idea for Parameterized Verification

How the leader is propagated:



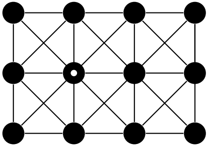
Abstraction Idea for Parameterized Verification

How the leader is propagated:



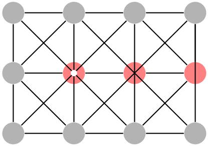
Abstraction Idea for Parameterized Verification

Abstracting the network:



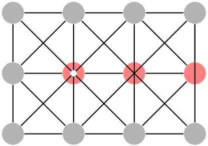
Abstraction Idea for Parameterized Verification

Abstracting the network:

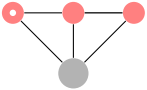


Pick a shortest path from the future leader to some node

Abstracting the network:



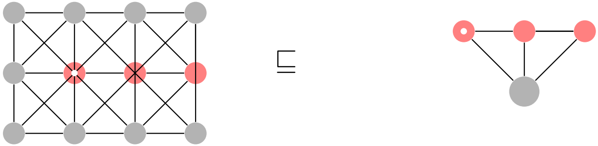
$\sqsubseteq$



Model the path concretely but all other nodes very abstractly

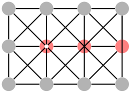
Abstraction Idea for Parameterized Verification

Abstracting the network:

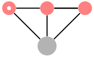


Model the path concretely but all other nodes very abstractly

$+$ Apply data abstraction to local variables and node identifiers



$\sqsubseteq$



Verification results: Topologies with "diameter"[1] up to 7 (13 minutes). With clock rates within 1 ± 10^{-2}.

E.g. 2D grids with 169 nodes, or 3D grids in 2197 nodes.

A custom algorithm implemented within NuSMV

	synchronous		asynchronous	
K	*N*	time	*N*	time
1	8	0s	8	0s
2	14	1s	14	1s
3	23	1s	25	28s
4	35	3s	39	130s
5	54	16s	63	65mins
6	67	76s	TO	TO
7	107	13mins	TO	TO

K: Diameter

N: Number of steps to convergence

Future work: Model checking time synchronization

[1]Max distance from the future leader