

Outline

Self-stabilization

Hypotheses

Atomicity Scheduling

Composition

Fair Composition Crossover Composition

Proof Techniques

Transfer Function Convergence stairs

Conclusion



Example

Example

- $U_0 = a$
- $U_{n+1} = \frac{U_n}{2}$ if U_n is even
- $U_{n+1} = \frac{3U_n + 1}{2}$ if U_n is odd

- ▶ $U_0 = a$
- ▶ $U_{n+1} = \frac{U_n}{2}$ if U_n is even
- ▶ $U_{n+1} = \frac{3U_n + 1}{2}$ if U_n is odd

n	0	1	2	3	4	5	6	7	8	9	10	11
U_n	7	11	17	26	13	20	10	5	8	4	2	1



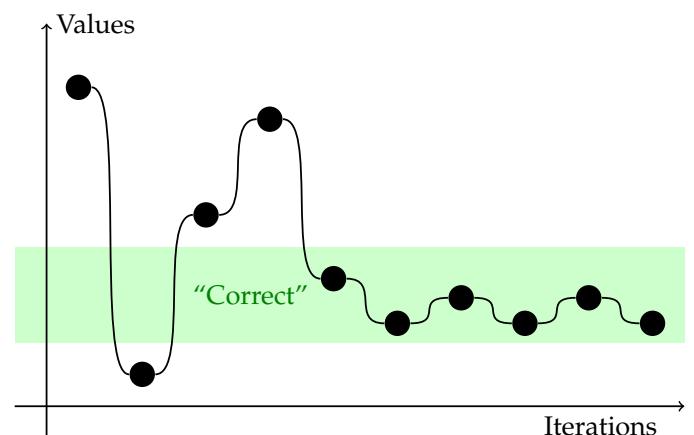
Example

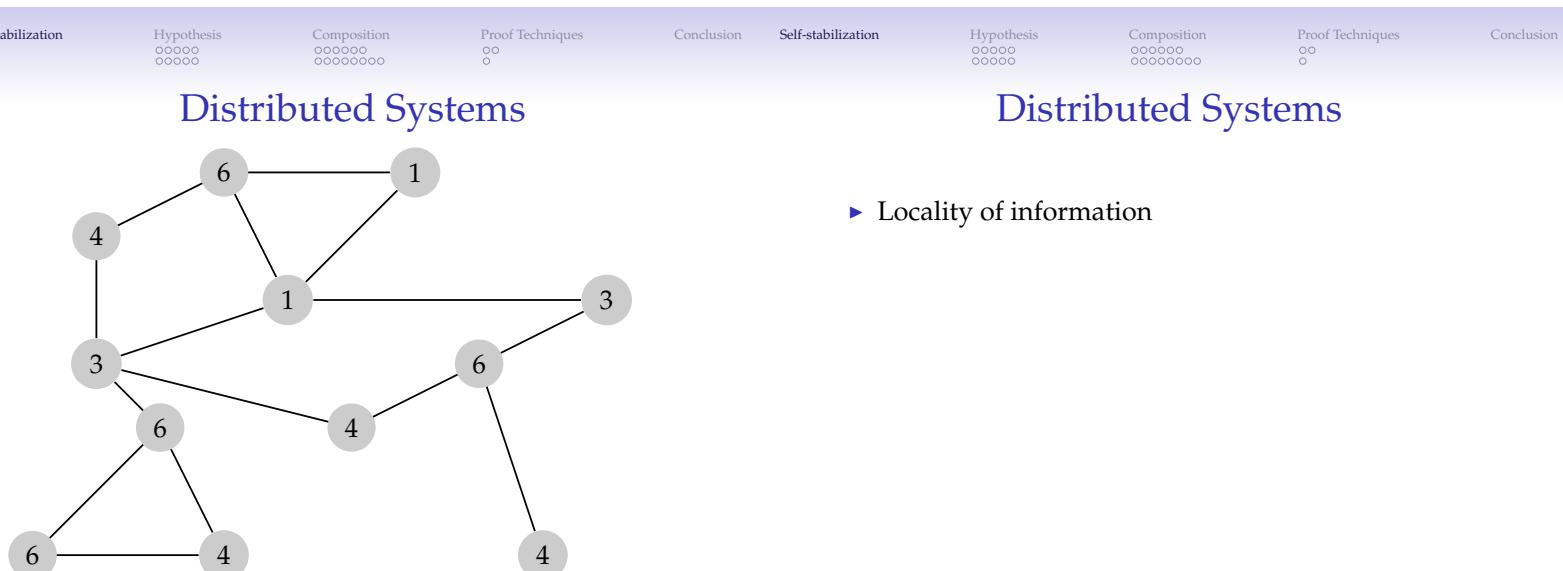
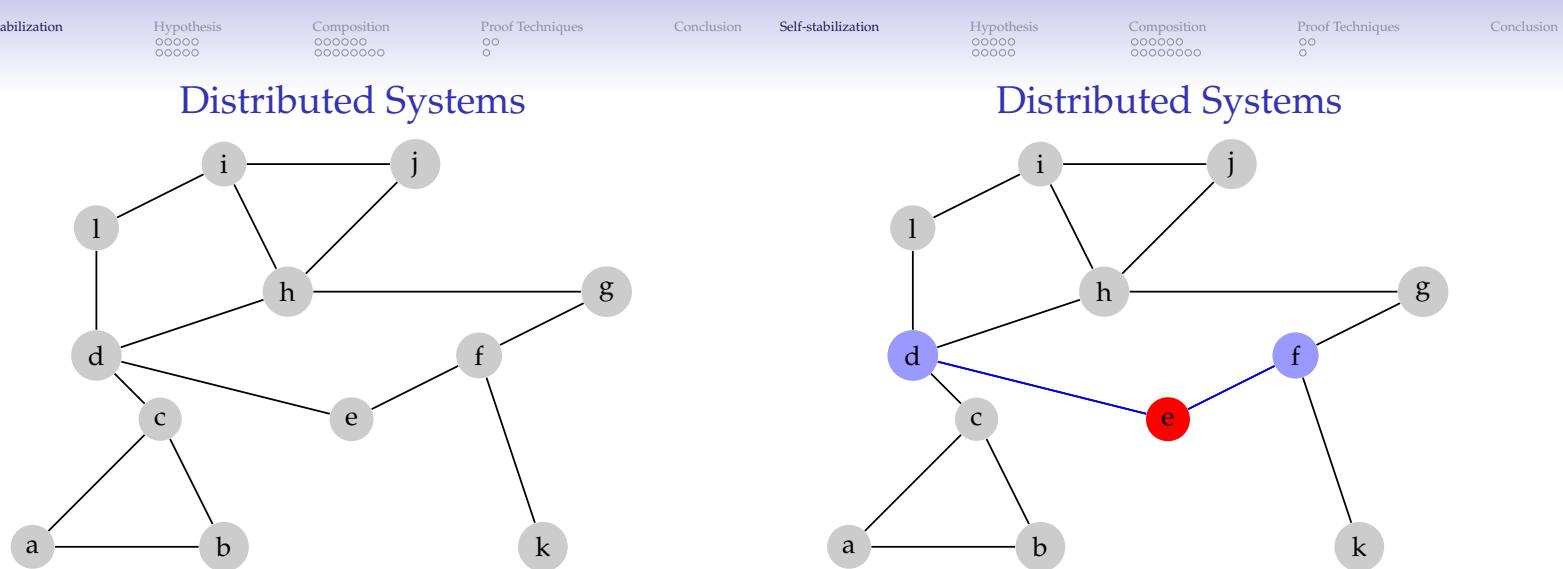
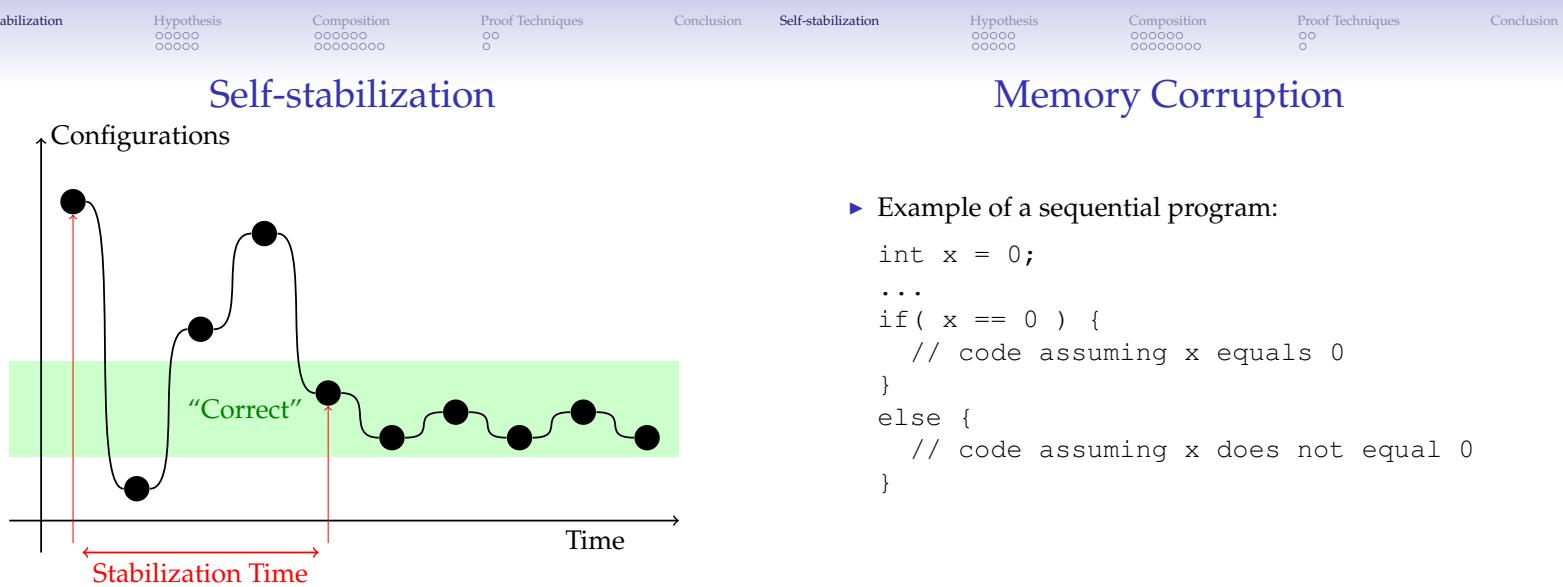
Example

- ▶ $U_0 = a$
- ▶ $U_{n+1} = \frac{U_n}{2}$ if U_n is even
- ▶ $U_{n+1} = \frac{3U_n + 1}{2}$ if U_n is odd

n	0	1	2	3	4	5	6	7	8	9	10	11
U_n	7	11	17	26	13	20	10	5	8	4	2	1

- ▶ Converges towards a “correct” behavior
 - ▶ 12121212121212121212121212121212...
 - ▶ Independent from the initial value





Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	---------------------	-------------------------	-----------------------------	------------	--------------------	---------------------	-------------------------	-----------------------------	------------

Distributed Systems

- ▶ Locality of information
- ▶ Locality of time

- ▶ Locality of information
- ▶ Locality of time
- ▶ ⇒ **non-determinism**

Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	---------------------	-------------------------	-----------------------------	------------	--------------------	---------------------	-------------------------	-----------------------------	------------

Distributed Systems

- ▶ Locality of information
- ▶ Locality of time
- ▶ ⇒ **non-determinism**

Definition (Configuration)

Product of the local states of the system components.

Definition (Execution)

Interleaving of the local executions of the system components.

Definition (Classical System, a.k.a. Non stabilizing)

Starting from a **particular** initial configuration, the system **immediately** exhibits correct behavior.

Definition (Self-stabilizing System)

Starting from **any** initial configuration, the system **eventually** reaches a configuration from with its behavior is correct.

Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	---------------------	-------------------------	-----------------------------	------------	--------------------	---------------------	-------------------------	-----------------------------	------------

Self-stabilization

Definition (Self-stabilizing System)

Starting from **any** initial configuration, the system **eventually** reaches a configuration from with its behavior is correct.

- ▶ defined by Dijkstra in 1974

Self-stabilization

Definition (Self-stabilizing System)

Starting from **any** initial configuration, the system **eventually** reaches a configuration from with its behavior is correct.

- ▶ defined by Dijkstra in 1974
- ▶ advocated by Lamport in 1984 to address fault-tolerant issues

Self-stabilization	Hypothesis ○○○○○ ○○○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ●○○○○ ○○○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	------------------------------	-----------------------------------	-----------------------------	------------	--------------------	------------------------------	-----------------------------------	-----------------------------	------------

Self-stabilization

Hypothesis
Atomicity
Scheduling

Composition
Fair Composition
Crossover Composition

Proof Techniques
Transfer Function
Convergence stairs

Conclusion

Atomicity

- ▶ Example of “stabilizing” sequential program

```
int x = 0;
...
while( x == x ) {
    x = 0;
    // code assuming x equals 0
}
```

Self-stabilization	Hypothesis ●○○○○ ○○○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ●○○○○ ○○○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	------------------------------	-----------------------------------	-----------------------------	------------	--------------------	------------------------------	-----------------------------------	-----------------------------	------------

Atomicity

Atomicity

- ▶ Example of “stabilizing” sequential program

```
0  iconst_0
1  istore_1
2  goto 7
5  iconst_0
6  istore_1
7  iload_1
8  iload_1
9  if_icmpeq 5
```

- ▶ Example of “stabilizing” sequential program

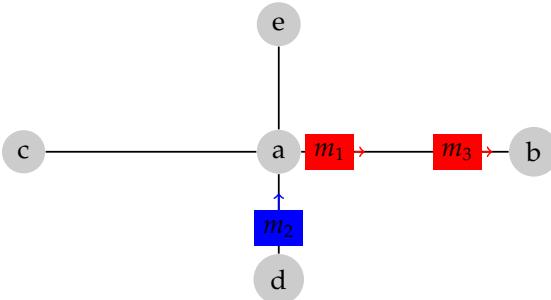
```
0  iconst_0
1  istore_1
2  goto 7
5  iconst_0
6  istore_1
7  iload_1
8  iload_1
9  if_icmpeq 5
```

Self-stabilization	Hypothesis ○○○○○ ○○○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○ ○○○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	------------------------------	-----------------------------------	-----------------------------	------------	--------------------	------------------------------	-----------------------------------	-----------------------------	------------

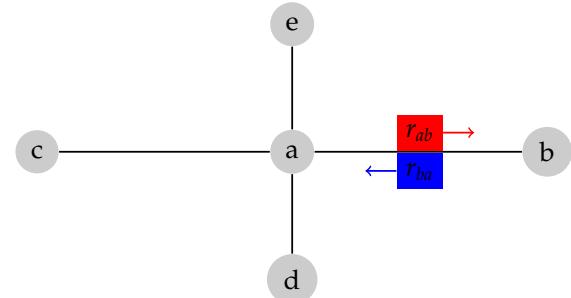
Communications

Communications

- ▶ Message Passing

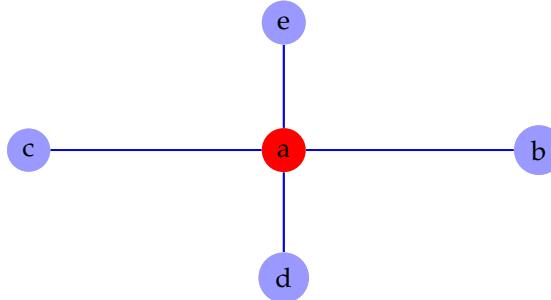


- ▶ Shared Registers





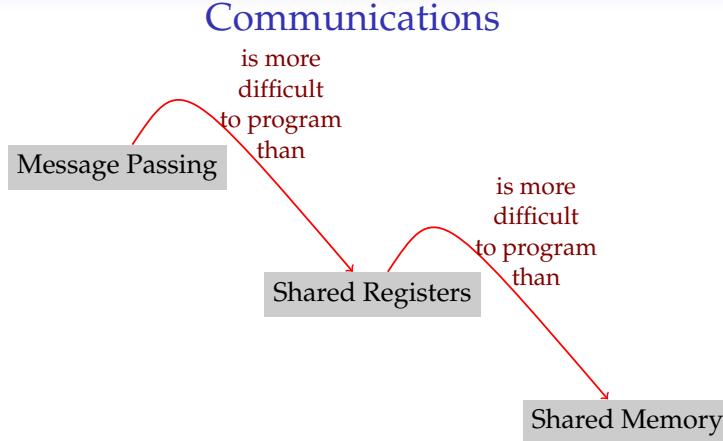
► Shared Memory



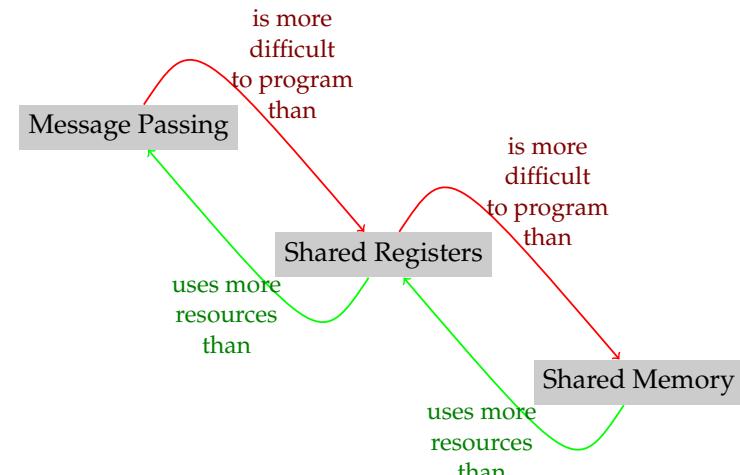
Message Passing

Shared Registers

Shared Memory



Communications



Example

Definition (Shared Memory)

In one atomic step, read the states of all neighbors and write own state

Definition (Guarded command)

► Guard → Action

Definition (Shared Memory)

In one atomic step, read the states of all neighbors and write own state

Definition (Guarded command)

- ▶ Guard → Action
- ▶ Guard: predicate on the states of the neighborhood

Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

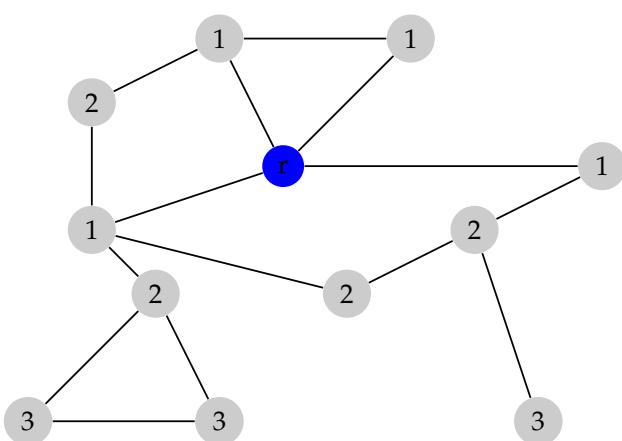
Example

Definition (Shared Memory)

In one atomic step, read the states of all neighbors and write own state

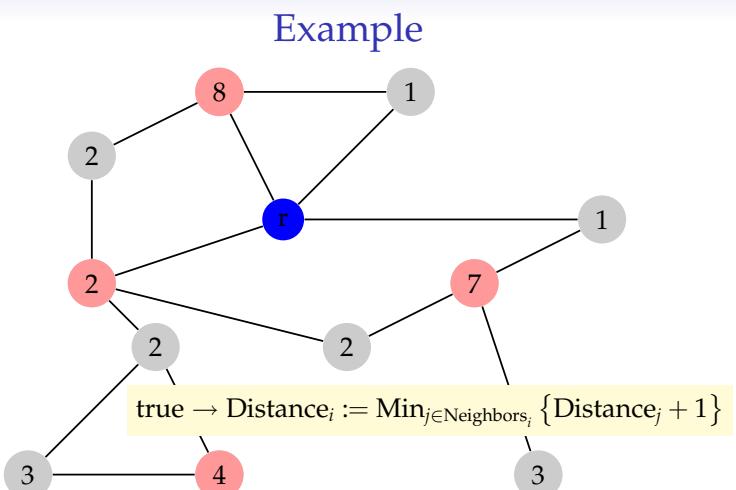
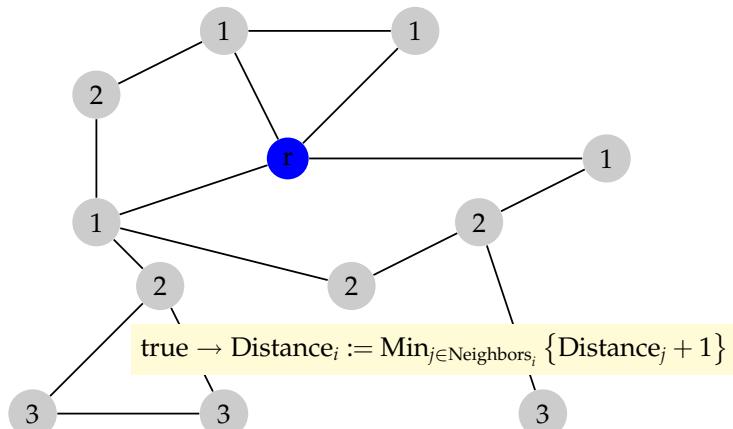
Definition (Guarded command)

- Guard → Action
- Guard: predicate on the states of the neighborhood
- Action: executed if Guard evaluates to true



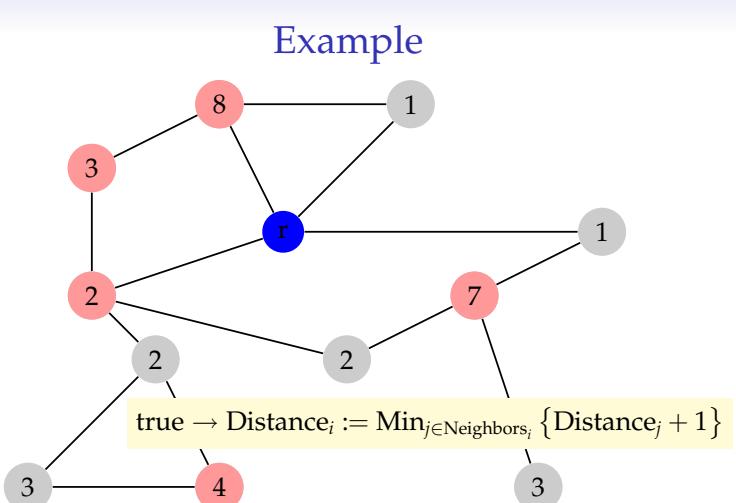
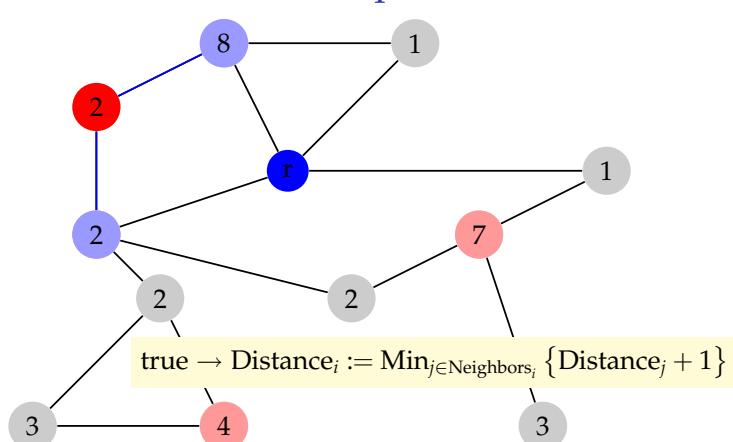
Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

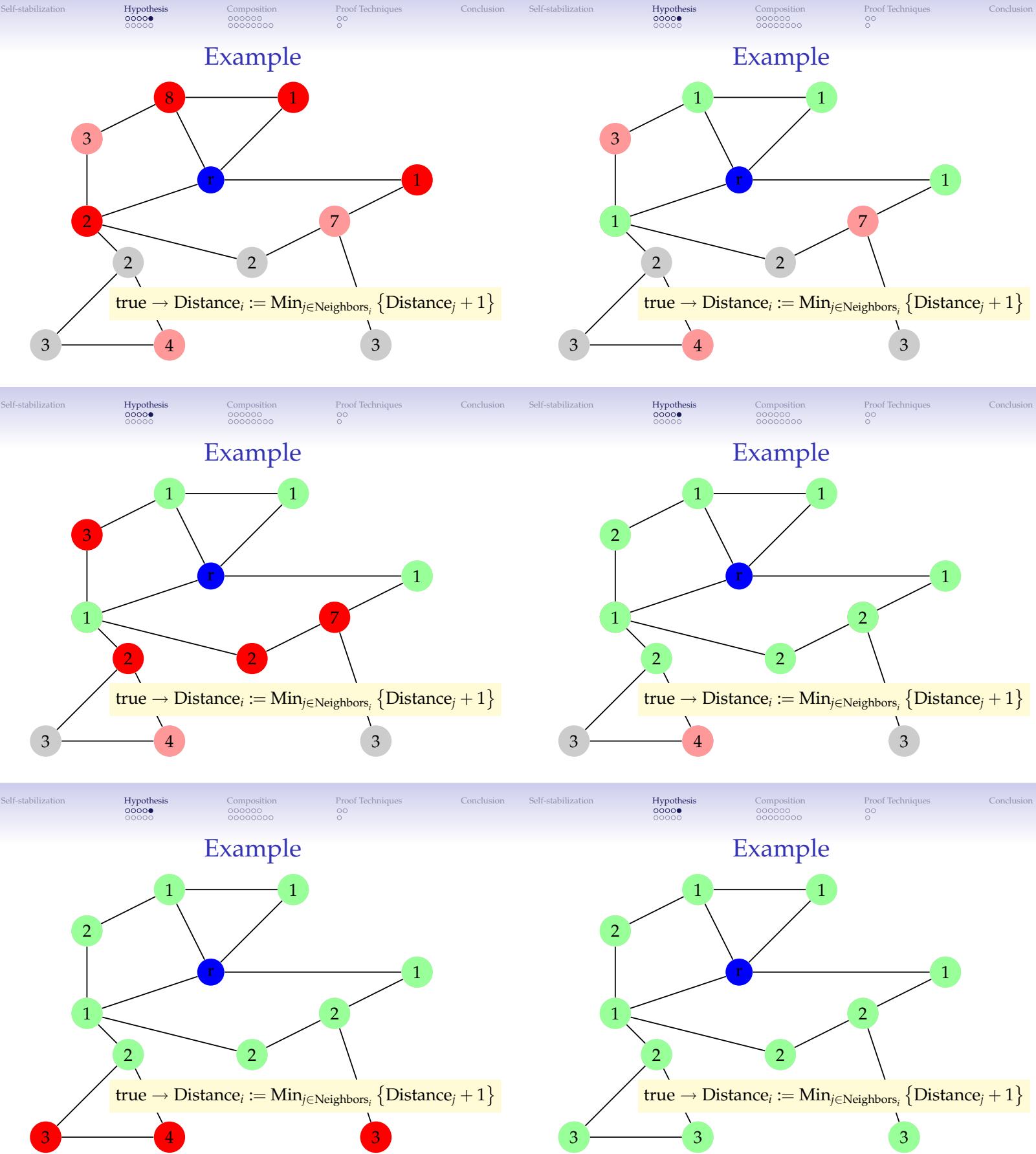
Example



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

Example





Self-stabilization	Hypothesis ○○○○○ ●●○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○ ●●○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	------------------------------	-----------------------------------	-----------------------------	------------	--------------------	------------------------------	-----------------------------------	-----------------------------	------------

Scheduling

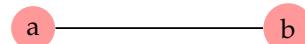
Definition (Scheduler a.k.a. Daemon)

The daemon chooses among activatable processors those that will execute their actions.

- The **daemon** can be seen as an adversary whose role is to prevent stabilization

$$\text{true} \rightarrow \text{color}_i := \text{Min} \{ \Delta \setminus \{\text{color}_j | j \in \text{Neighbors}_i\} \}$$

$$\Delta = \{ \textcolor{red}{0}, \textcolor{blue}{1} \}$$

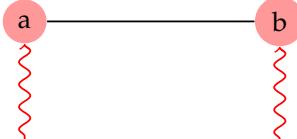


Self-stabilization	Hypothesis ○○○○○ ●●○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○ ●●○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	------------------------------	-----------------------------------	-----------------------------	------------	--------------------	------------------------------	-----------------------------------	-----------------------------	------------

Spatial Scheduling

$$\text{true} \rightarrow \text{color}_i := \text{Min} \{ \Delta \setminus \{\text{color}_j | j \in \text{Neighbors}_i\} \}$$

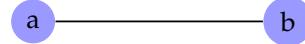
$$\Delta = \{ \textcolor{red}{0}, \textcolor{blue}{1} \}$$



Spatial Scheduling

$$\text{true} \rightarrow \text{color}_i := \text{Min} \{ \Delta \setminus \{\text{color}_j | j \in \text{Neighbors}_i\} \}$$

$$\Delta = \{ \textcolor{red}{0}, \textcolor{blue}{1} \}$$

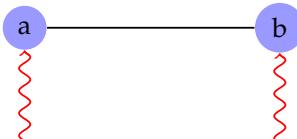


Self-stabilization	Hypothesis ○○○○○ ●●○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○ ●●○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	------------------------------	-----------------------------------	-----------------------------	------------	--------------------	------------------------------	-----------------------------------	-----------------------------	------------

Spatial Scheduling

$$\text{true} \rightarrow \text{color}_i := \text{Min} \{ \Delta \setminus \{\text{color}_j | j \in \text{Neighbors}_i\} \}$$

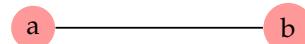
$$\Delta = \{ \textcolor{red}{0}, \textcolor{blue}{1} \}$$



Spatial Scheduling

$$\text{true} \rightarrow \text{color}_i := \text{Min} \{ \Delta \setminus \{\text{color}_j | j \in \text{Neighbors}_i\} \}$$

$$\Delta = \{ \textcolor{red}{0}, \textcolor{blue}{1} \}$$



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

Hypothesis
○○○○○
○●○○○

Composition
○○○○○○
○○○○○○○○

Proof Techniques
○○
○

Conclusion

Self-stabilization

Hypothesis
○○○○○
○●○○○

Composition
○○○○○○
○○○○○○○○

Proof Techniques
○○
○

Conclusion

Spatial Scheduling

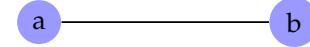
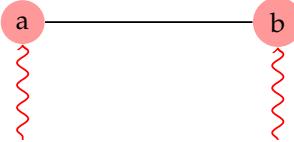
Spatial Scheduling

true \rightarrow $\text{color}_i := \text{Min} \{ \Delta \setminus \{\text{color}_j | j \in \text{Neighbors}_i\} \}$

true \rightarrow $\text{color}_i := \text{Min} \{ \Delta \setminus \{\text{color}_j | j \in \text{Neighbors}_i\} \}$

$$\Delta = \{ \textcolor{red}{0}, \textcolor{blue}{1} \}$$

$$\Delta = \{ \textcolor{red}{0}, \textcolor{blue}{1} \}$$



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

Hypothesis
○○○○○
○●○○○

Composition
○○○○○○
○○○○○○○○

Proof Techniques
○○
○

Conclusion

Self-stabilization

Hypothesis
○○○○○
○●○○○

Composition
○○○○○○
○○○○○○○○

Proof Techniques
○○
○

Conclusion

Spatial Scheduling

Spatial Scheduling

true \rightarrow $\text{color}_i := \text{Min} \{ \Delta \setminus \{\text{color}_j | j \in \text{Neighbors}_i\} \}$

true \rightarrow $\text{color}_i := \text{Min} \{ \Delta \setminus \{\text{color}_j | j \in \text{Neighbors}_i\} \}$

$$\Delta = \{ \textcolor{red}{0}, \textcolor{blue}{1} \}$$

$$\Delta = \{ \textcolor{red}{0}, \textcolor{blue}{1} \}$$



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

Hypothesis
○○○○○
○●○○○

Composition
○○○○○○
○○○○○○○○

Proof Techniques
○○
○

Conclusion

Self-stabilization

Hypothesis
○○○○○
○●○○○

Composition
○○○○○○
○○○○○○○○

Proof Techniques
○○
○

Conclusion

Spatial Scheduling

Spatial Scheduling

Distributed

is more difficult to program than
Distributed

Synchronous

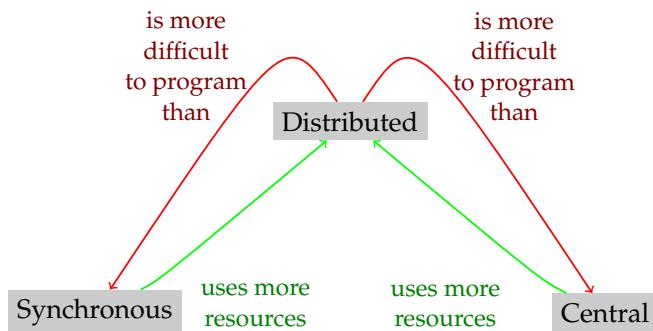
Central

Synchronous

Central

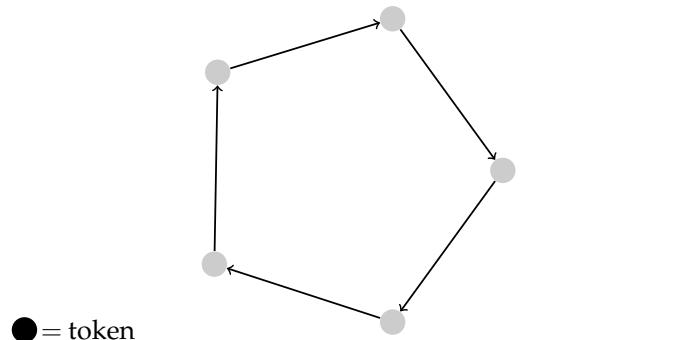
Self-stabilization	Hypothesis ○○○○○ ○○○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○ ○○○○○○	Composition ○○○○○○○○ ○○○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	------------------------------	-----------------------------------	-----------------------------	------------	--------------------	-------------------------------	---------------------------------------	-----------------------------	------------

Spatial Scheduling



Temporal Scheduling

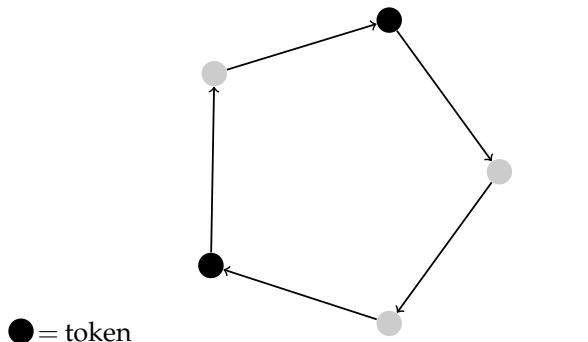
token \rightarrow pass token to left neighbor with probability $\frac{1}{2}$



Self-stabilization	Hypothesis ○○○○○ ○○○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○ ○○○○○○	Composition ○○○○○○○○ ○○○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	------------------------------	-----------------------------------	-----------------------------	------------	--------------------	-------------------------------	---------------------------------------	-----------------------------	------------

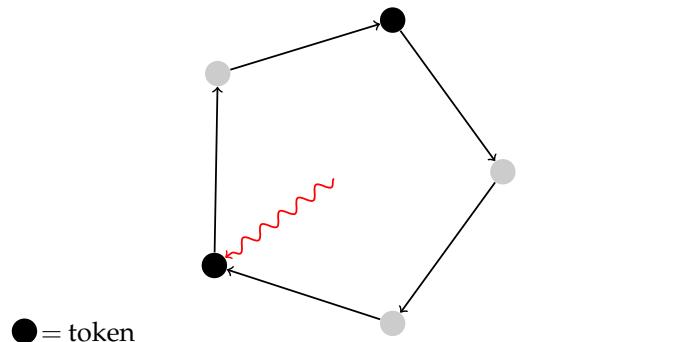
Temporal Scheduling

token \rightarrow pass token to left neighbor with probability $\frac{1}{2}$



Temporal Scheduling

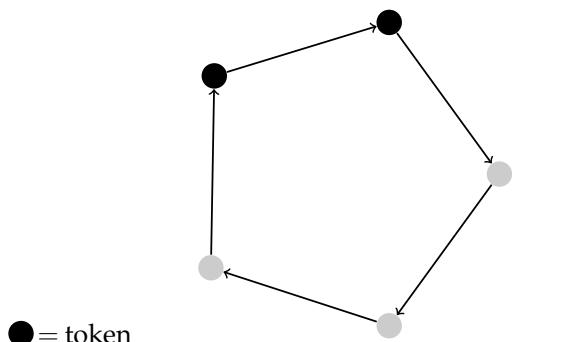
token \rightarrow pass token to left neighbor with probability $\frac{1}{2}$



Self-stabilization	Hypothesis ○○○○○ ○○○○○	Composition ○○○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○ ○○○○○○	Composition ○○○○○○○○ ○○○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	------------------------------	-----------------------------------	-----------------------------	------------	--------------------	-------------------------------	---------------------------------------	-----------------------------	------------

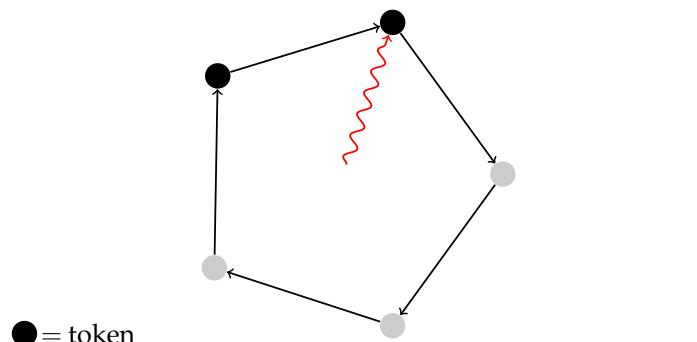
Temporal Scheduling

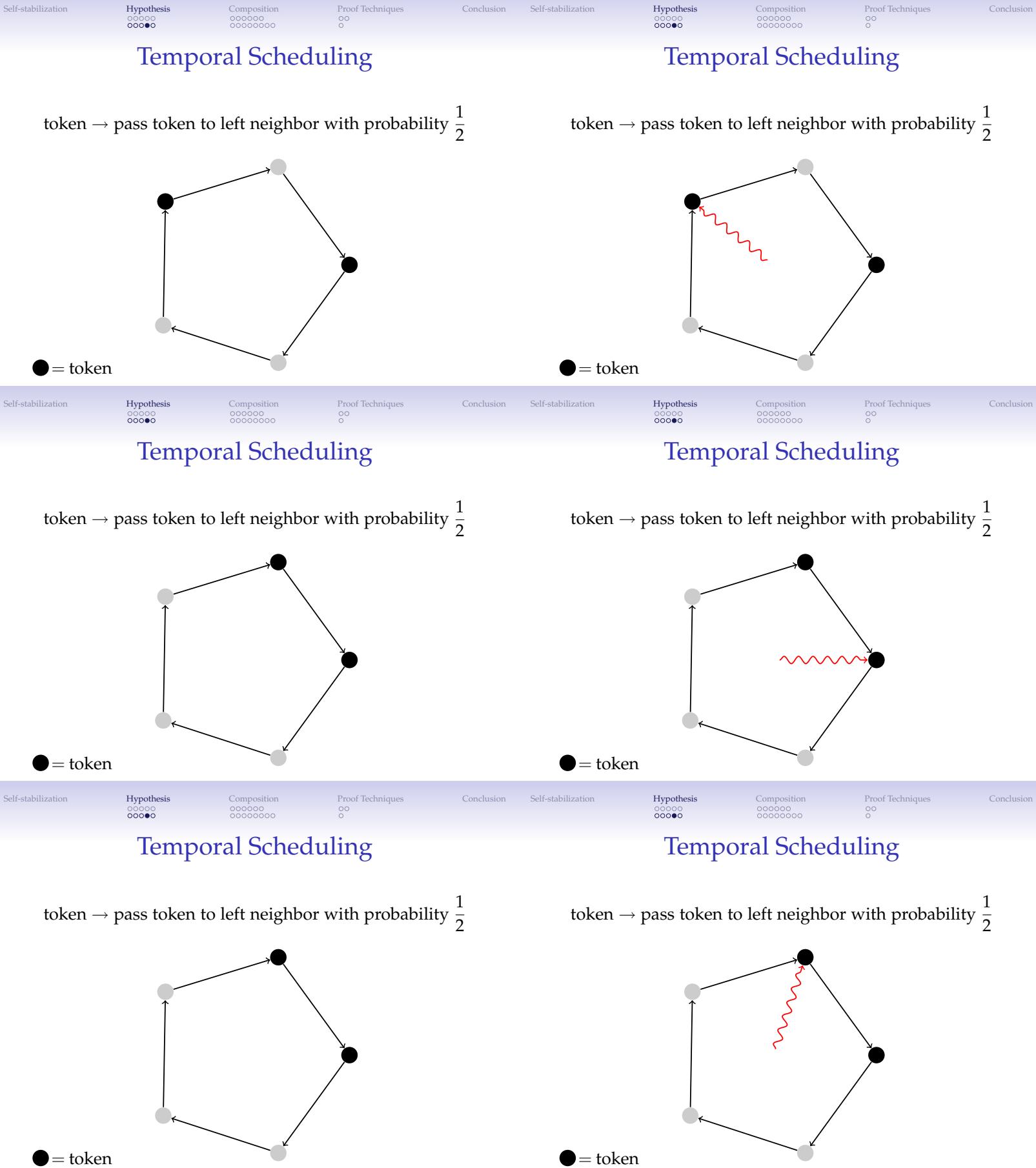
token \rightarrow pass token to left neighbor with probability $\frac{1}{2}$



Temporal Scheduling

token \rightarrow pass token to left neighbor with probability $\frac{1}{2}$





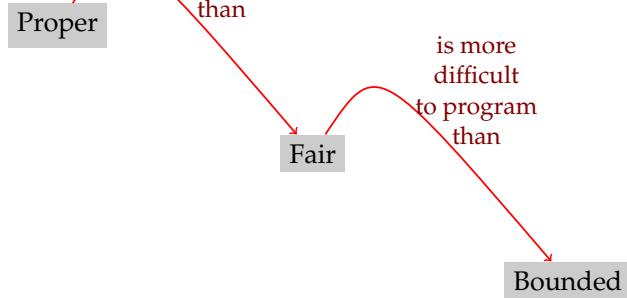
Temporal Scheduling

token → pass token to left neighbor with probability $\frac{1}{2}$



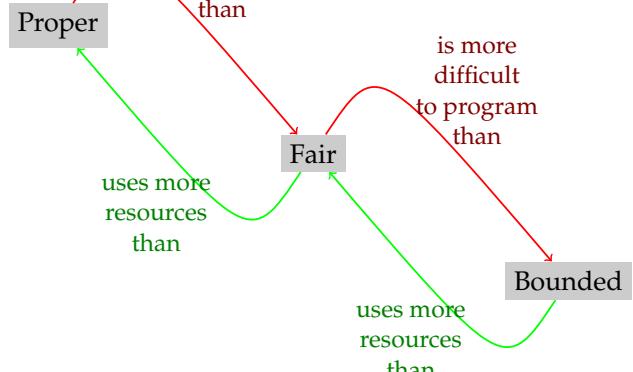
Temporal Scheduling

is more difficult to program than



Temporal Scheduling

is more difficult to program than



Self-stabilization

Hypothesis
Atomicity
Scheduling

Composition
Fair Composition
Crossover Composition

Proof Techniques
Transfer Function
Convergence stairs

Conclusion

Fair Composition

Basic idea

- ▶ Compose several self-stabilizing algorithms Al_1, Al_2, \dots, Al_k such that the results of algorithms Al_1, Al_2, \dots, Al_i can be reused by Al_{i+1}
- ▶ Al_{i+1} can not detect whether algorithms Al_1, Al_2, \dots, Al_i have stabilized, but behaves as if

Self-stabilization	Hypothesis ○○○○○	Composition ●○○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○	Composition ○○●○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	---------------------	----------------------------------	-----------------------------	------------	--------------------	---------------------	-----------------------------------	-----------------------------	------------

Fair Composition

Basic idea

- Compose several self-stabilizing algorithms Al_1, Al_2, \dots, Al_k such that the results of algorithms Al_1, Al_2, \dots, Al_i can be reused by Al_{i+1}
- Al_{i+1} can not detect whether algorithms Al_1, Al_2, \dots, Al_i have stabilized, but behaves as if

Example with $k = 2$

- Two simple algorithms server and client are combined to obtain a more complex algorithm
- The server algorithm ensures that some properties (used by the client) will be eventually verified

Self-stabilization	Hypothesis ○○○○○	Composition ○○●○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○	Composition ○○●○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	---------------------	-----------------------------------	-----------------------------	------------	--------------------	---------------------	-----------------------------------	-----------------------------	------------

Fair Composition

Definition (A -projection)

For a configuration c of $S_1 \times S_2 \times \dots \times S_n$, the **A -projection** of c is (a_1, \dots, a_n) of $A_1 \times \dots \times A_n$

Definition (Conditional Stabilization)

Al_2 is **self-stabilizing for task T_2 given task T_1** if any fair computation of Al_2 that has an A -projection in T_1 has a suffix in T_2

Self-stabilization	Hypothesis ○○○○○	Composition ○○●○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○	Composition ○○●○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	---------------------	-----------------------------------	-----------------------------	------------	--------------------	---------------------	-----------------------------------	-----------------------------	------------

Fair Composition

Definition (Fair composition)

Al is a **fair composition** of Al_1 and Al_2 if, in Al , every process alternatively executes actions of Al_1 and Al_2

Theorem

If Al_2 is self-stabilizing for T_2 given T_1 , and if Al_1 is self-stabilizing for T_1 , then the fair composition of Al_1 and Al_2 is self-stabilizing for T_2

$A_i \rightarrow B_i$

$Al_1 \quad Al_2$

Example

- Assume the server algorithm Al_1 solves a task defined by a set of legal executions T_1 , and the client algorithm Al_2 solves T_2

A_i

- Let A_i be the set of states of process P_i for Al_1 , and let $S_i = A_i \times B_i$ be the set of states of process P_i for Al_2 , where anytime P_i executes Al_2 , it modifies the B_i part of $A_i \times B_i$

$A_i \rightarrow B_i$

Self-stabilization	Hypothesis ○○○○○	Composition ○○●○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○	Composition ○○●○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	---------------------	-----------------------------------	-----------------------------	------------	--------------------	---------------------	-----------------------------------	-----------------------------	------------

Fair Composition

Definition (Fair composition)

Al is a **fair composition** of Al_1 and Al_2 if, in Al , every process alternatively executes actions of Al_1 and Al_2

Theorem

If Al_2 is self-stabilizing for T_2 given T_1 , and if Al_1 is self-stabilizing for T_1 , then the fair composition of Al_1 and Al_2 is self-stabilizing for T_2

$A_i \rightarrow B_i$

$Al_1 \quad Al_2$

Self-stabilization	Hypothesis ○○○○○	Composition ○○●○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion	Self-stabilization	Hypothesis ○○○○○	Composition ○○●○○○ ○○○○○○○○	Proof Techniques ○○ ○	Conclusion
--------------------	---------------------	-----------------------------------	-----------------------------	------------	--------------------	---------------------	-----------------------------------	-----------------------------	------------

Fair Composition

Definition (Fair composition)

Al is a **fair composition** of Al_1 and Al_2 if, in Al , every process alternatively executes actions of Al_1 and Al_2

Theorem

If Al_2 is self-stabilizing for T_2 given T_1 , and if Al_1 is self-stabilizing for T_1 , then the fair composition of Al_1 and Al_2 is self-stabilizing for T_2

$A_i \rightarrow B_i$

$Al_1 \quad Al_2$

Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

○○○○
○○○○

○○○○●
○○○○○○○

○○
○

Conclusion

Self-stabilization

○○○○
○○○○

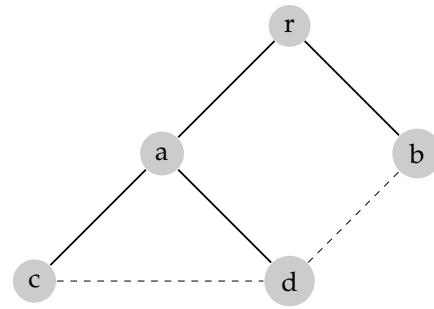
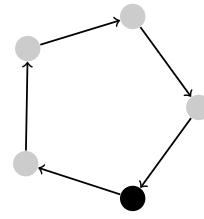
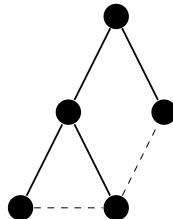
○○○○●
○○○○○○○

○○
○

Conclusion

Example

- We are given two self-stabilizing algorithms, one for constructing a tree, one for mutual exclusion on a ring
- We wish to construct a self-stabilizing mutual exclusion algorithm on general graphs



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

○○○○
○○○○

○○○○●
○○○○○○○

○○
○

Conclusion

Self-stabilization

○○○○
○○○○

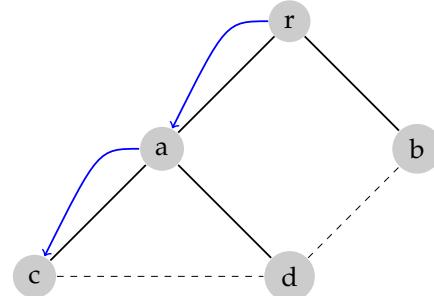
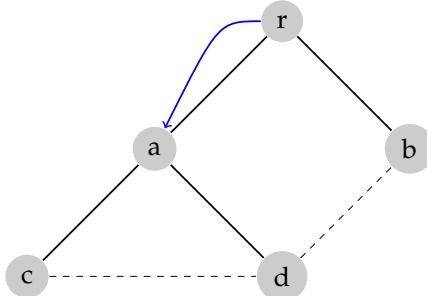
○○○○●
○○○○○○○

○○
○

Conclusion

Example

Example



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

○○○○
○○○○

○○○○●
○○○○○○○

○○
○

Conclusion

Self-stabilization

○○○○
○○○○

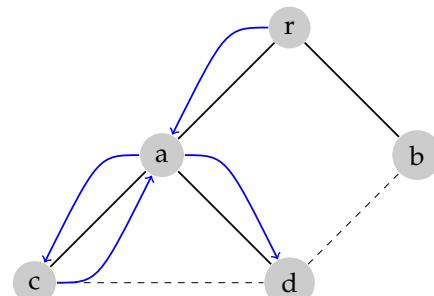
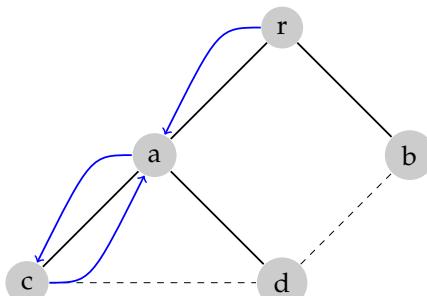
○○○○●
○○○○○○○

○○
○

Conclusion

Example

Example



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

Hypothesis
○○○○○
○○○○○

Composition
○○○○●○
○○○○○○○

Proof Techniques
○○
○

Conclusion Self-stabilization

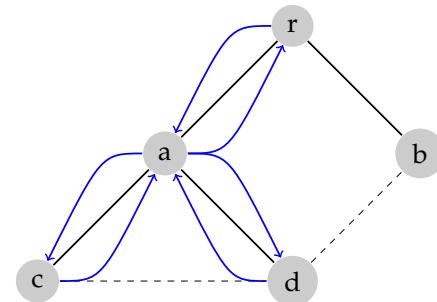
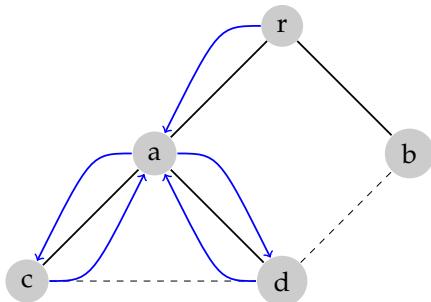
Hypothesis
○○○○○
○○○○○

Composition
○○○○●○
○○○○○○○

Proof Techniques
○○
○

Conclusion

Example



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

Hypothesis
○○○○○
○○○○○

Composition
○○○○●○
○○○○○○○

Proof Techniques
○○
○

Conclusion Self-stabilization

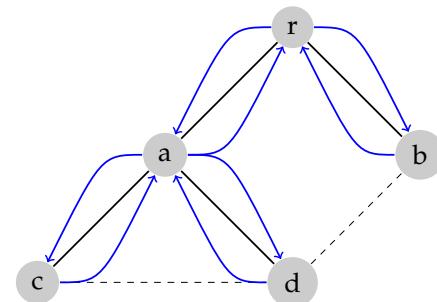
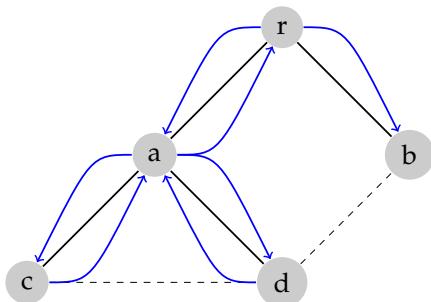
Hypothesis
○○○○○
○○○○○

Composition
○○○○●○
○○○○○○○

Proof Techniques
○○
○

Conclusion

Example



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

Hypothesis
○○○○○
○○○○○

Composition
○○○○●○
○○○○○○○

Proof Techniques
○○
○

Conclusion Self-stabilization

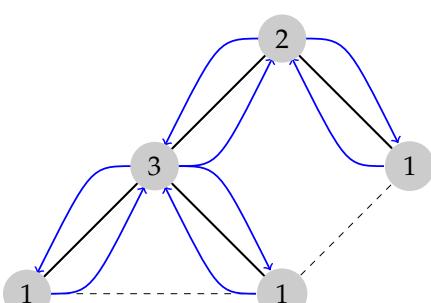
Hypothesis
○○○○○
○○○○○

Composition
○○○○●○
●○○○○○○

Proof Techniques
○○
○

Conclusion

Example



Crossover Composition

Basic Idea

- We are given two algorithms Al_1 and Al_2
- Al_1 is correct with hypothesis H_1 and Al_2 is correct with hypothesis H_2
- H_2 is more restrictive than H_1

Definition

Crossover Composition The **crossover composition** is such that Al_2 is conditionally executed (only when Al_1 is executed) Al_2 is then correct with hypothesis H_1

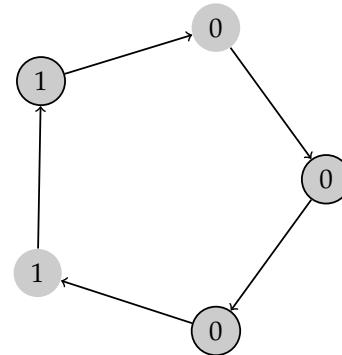
Example

Uniform Unidirectional ring

- Each node i has a variable v_i
- Each node i has a token if $v_i \neq v_{i-1} + 1 \pmod{SND(n)}$
- Each node i passes a token by executing $v_i := v_{i-1} + 1 \pmod{SND(n)}$

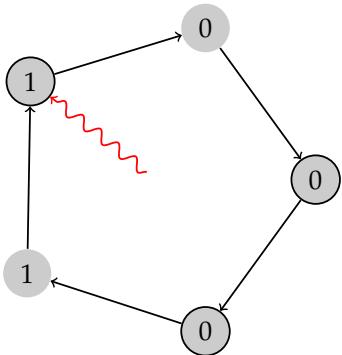
($SND(n)$: smallest non divisor of n)

► $SND(n) = 2$

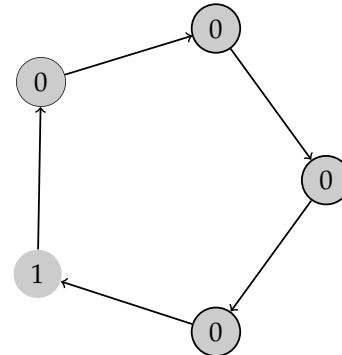


Example

► $SND(n) = 2$

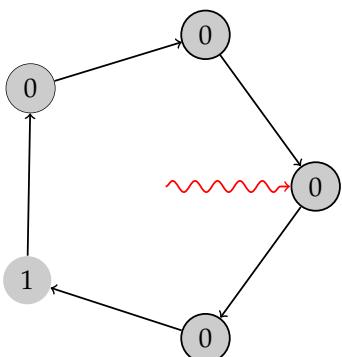


► $SND(n) = 2$

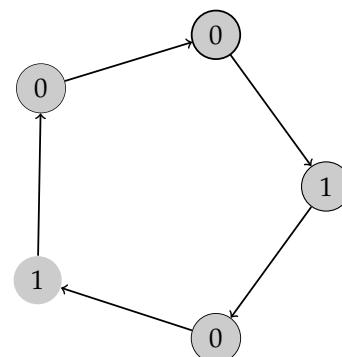


Example

► $SND(n) = 2$



► $SND(n) = 2$





Algorithm Al_1

- ▶ A node with the token is activatable
- ▶ An activated node always transmit the token
- ▶ Al_1 solves the token passing problem with an arbitrary distributed deamon (Hypothesis H_1)

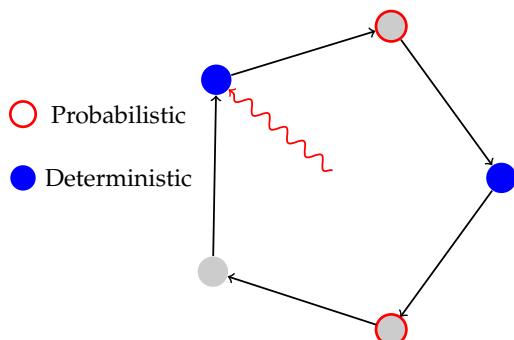
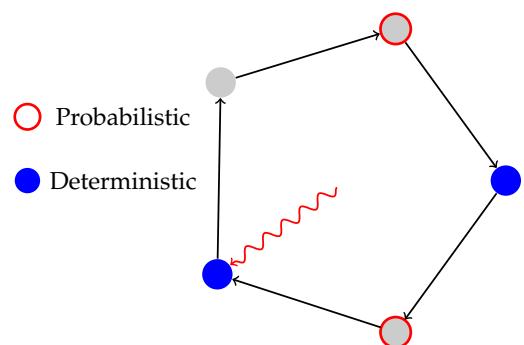
Algorithm Al_2

- ▶ Each node with a token is activatable
- ▶ Each activated node transmits the token with probability $\frac{1}{2}$
- ▶ Al_2 solves the self-stabilizing mutual exclusion problem using token passing and a bounded daemon (Hypothesis H_2)

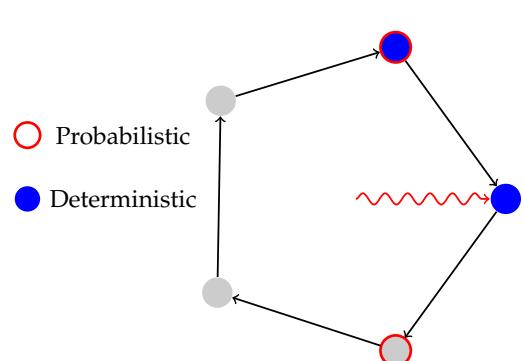


Algorithm Al_2 composed with Al_1

- ▶ A node may have two tokens (one deterministic and one probabilistic)
- ▶ A node with a deterministic token is activatable
- ▶ An activated node passes the deterministic token, and (if it has it) the probabilistic token with probability $\frac{1}{2}$



Example



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

○○○○○
○○○○○

○○○○○○
○○○○○○●○

○○
○

Conclusion Self-stabilization

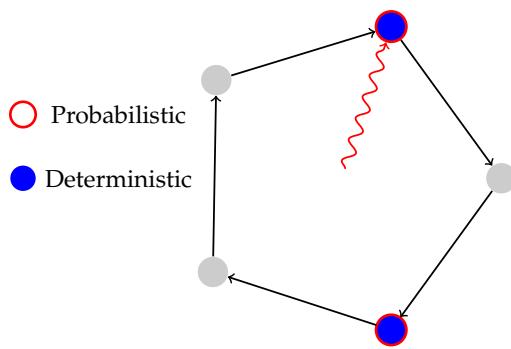
○○○○○
○○○○○

○○○○○○
○○○○○○●○

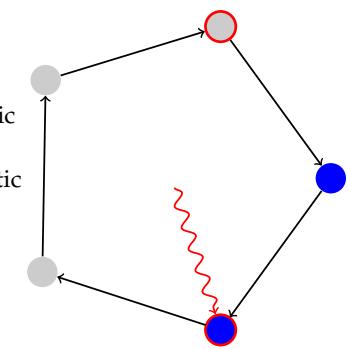
○○
○

Conclusion

Example



Example



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

○○○○○
○○○○○

○○○○○○
○○○○○○●○

○○
○

Conclusion Self-stabilization

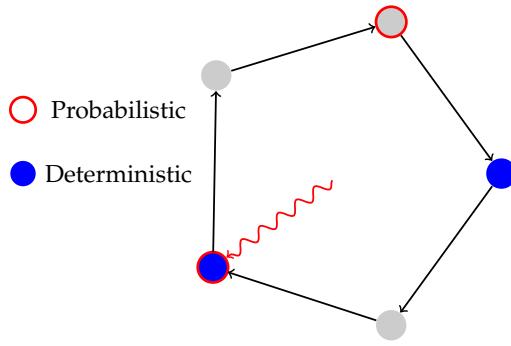
○○○○○
○○○○○

○○○○○○
○○○○○○●○

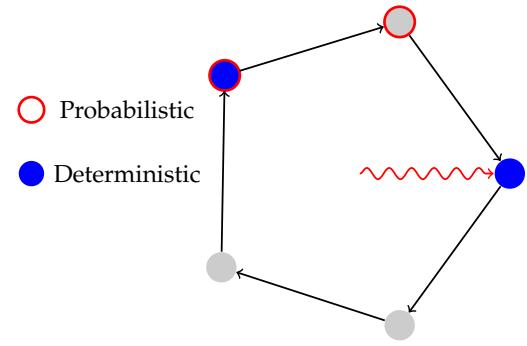
○○
○

Conclusion

Example



Example



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

○○○○○
○○○○○

○○○○○○
○○○○○○●○

○○
○

Conclusion Self-stabilization

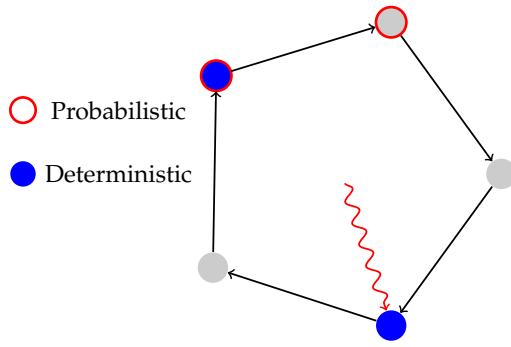
○○○○○
○○○○○

○○○○○○
○○○○○○●○

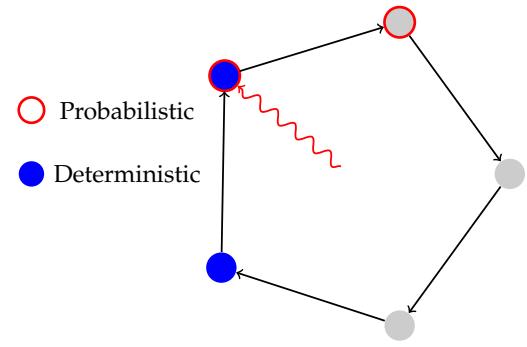
○○
○

Conclusion

Example



Example



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

○○○○○
○○○○○

○○○○○○
○○○○○○●○

○○
○

Conclusion

Self-stabilization

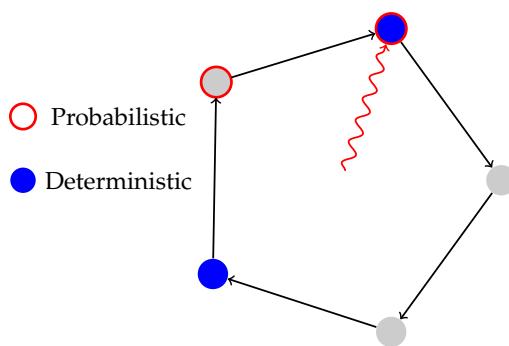
○○○○○
○○○○○

○○○○○○
○○○○○○●○

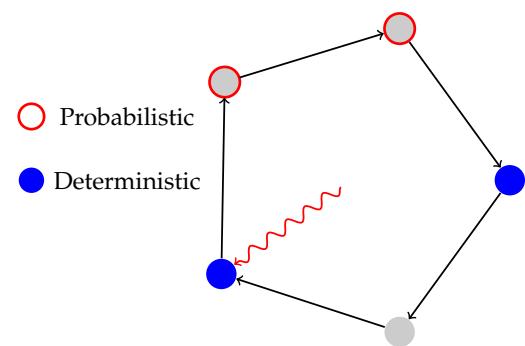
○○
○

Conclusion

Example



Example



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

○○○○○
○○○○○

○○○○○○
○○○○○○●○

○○
○

Conclusion

Self-stabilization

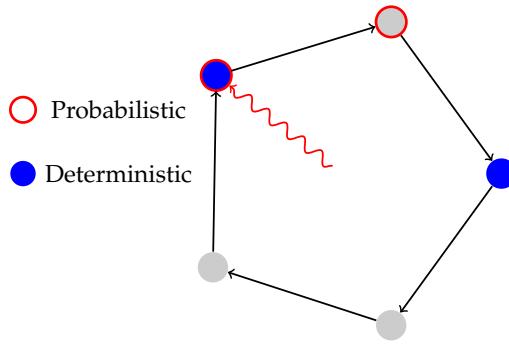
○○○○○
○○○○○

○○○○○○
○○○○○○●○

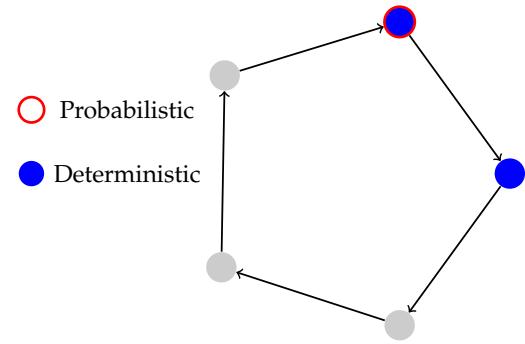
○○
○

Conclusion

Example



Example



Self-stabilization Hypothesis Composition Proof Techniques Conclusion Self-stabilization Hypothesis Composition Proof Techniques Conclusion

○○○○○
○○○○○

○○○○○○
○○○○○○●○

○○
○

Conclusion

Self-stabilization

○○○○○
○○○○○

○○○○○○
○○○○○○●○

○○
○

Conclusion

Example

- ▶ Al_2 composed with Al_1 solves the self-stabilizing mutual exclusion problem with an arbitrary distributed deamon (H_1)
- ▶ Al_2 composed with Al_1 does not solve a more complex problem, but handles less restrictive hypothesis

Self-stabilization

Hypothesis
Atomicity
Scheduling

Composition
Fair Composition
Crossover Composition

Proof Techniques
Transfer Function
Convergence stairs

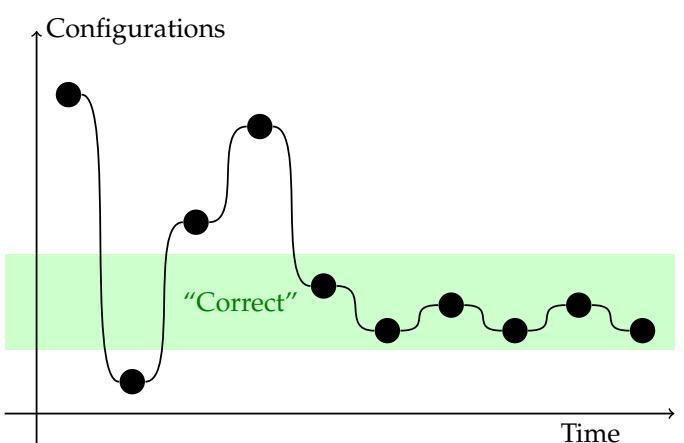
Conclusion

Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○	Proof Techniques ○●○	Conclusion	Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○○○	Proof Techniques ○○●○	Conclusion
--------------------	---------------------	-------------------------	-------------------------	------------	--------------------	---------------------	---------------------------	--------------------------	------------

Transfer Function

Basic Idea

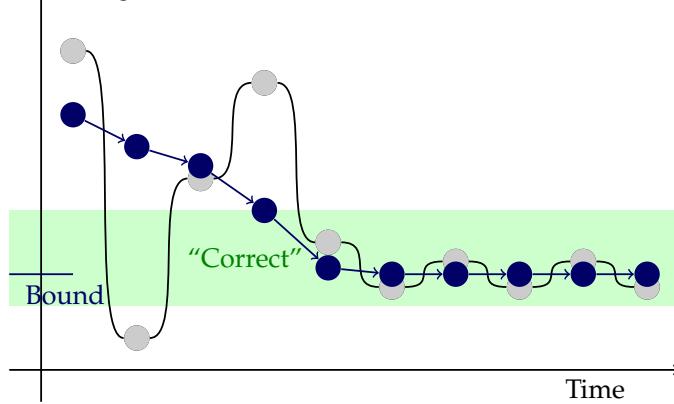
- $c_1 \rightarrow c_2 \rightarrow c_3 \rightarrow c_4 \rightarrow \dots \rightarrow c_i$
- $FP(c_1) > FP(c_2) > FP(c_3) > \dots > FP(c_i) = \text{bound}$
- Used to prove convergence
- Can be used to compute the number of steps to reach a legitimate configuration



Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○	Proof Techniques ○○	Conclusion	Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○○○	Proof Techniques ○○●○	Conclusion
--------------------	---------------------	-------------------------	------------------------	------------	--------------------	---------------------	---------------------------	--------------------------	------------

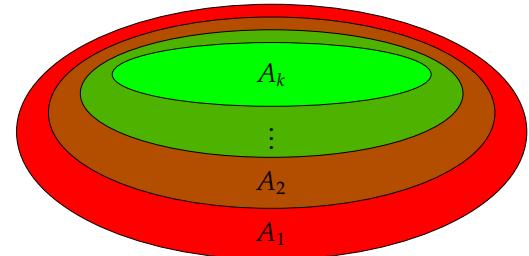
Transfer Function

Configurations



Convergence stairs

- A_i is a predicate
- A_k is legitimate
- For any i between 1 and k , A_{i+1} is a refinement of A_i



Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○	Proof Techniques ○○	Conclusion	Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○○○	Proof Techniques ○○	Conclusion
--------------------	---------------------	-------------------------	------------------------	------------	--------------------	---------------------	---------------------------	------------------------	------------

Self-stabilization

Hypothesis

Atomicity
Scheduling

Composition

Fair Composition
Crossover Composition

Proof Techniques

Transfer Function
Convergence stairs

Conclusion

Self-stabilization

Pros

- The network need not be initialized
- When a fault is diagnosed, it is sufficient to identify, then remove or restart the faulty components
- The self-stabilization property does not depend on the nature of the fault
- The self-stabilization property does not depend on the extent of the fault

Self-stabilization	Hypothesis ○○○○○	Composition ○○○○○○○○	Proof Techniques ○○○	Conclusion
--------------------	---------------------	-------------------------	-------------------------	------------

Self-stabilization

Cons

- ▶ “Eventually” does not give any bound on the stabilization time
- ▶ A single failure may trigger a correcting action at every node in the network
- ▶ Faults must be sufficiently rare that they can be considered are transient
- ▶ Nodes never know whether the system is stabilized or not