

The Good, the Bad, and the Random

Stochastic Games with Lossy Channels



Parash
Abdulla



Noomene
Ben Henda



Richard
Mayr



Sven
Sandberg

Uppsala University

NC State University, USA

Slide 1

Overview

- Background
- Model
- Algorithm
- Conclusion

Slide 2

Overview

- Background
- Model
- Algorithm
- Conclusion

Slide 3

Context

- Reactive System**
 - Never terminates
 - Input & output during execution
- Program Verification**
 - Communication protocols
 - Hardware

One approach to verification:

- Model checking**
 - Input: model + formula
 - Output: "yes, formula true" / "no, formula false"

Slide 4

Context

- Reactive System**
 - Never terminates
 - Input & output during execution
- Program Verification**
 - Communication protocols
 - Hardware

One approach to verification:

- Model checking**
 - Input: model + formula
 - Output: "yes, formula true" / "no, formula false"

Slide 5

Stochastic Infinite-State Games

- Stochastic**
 - Unreliable communication
 - Fault-Tolerant systems
 - Randomized algorithms
- Infinite-State**
 - Unbounded queues, stacks, counters, ...
 - Time
 - Unbounded parallelism
- Games**
 - Model interacts with environment
 - Alternation in logics
 - Abstraction

Slide 6

Repeated Reachability

- F – set of "target states"
- F reached infinitely often?

Example:

- Will it always return to the green state?



Slide 9

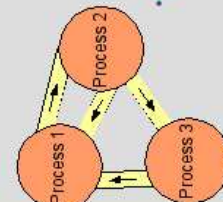
Stochastic Infinite-State Games

- Stochastic**
 - Unreliable communication
 - Fault-Tolerant systems
 - Randomized algorithms
- Infinite-State**
 - Unbounded queues, stacks, counters, ...
 - Time
 - Unbounded parallelism
- Games**
 - Model interacts with environment
 - Alternation in logics
 - Abstraction

Slide 7

Lossy Channel Systems (LCS)

- Model:**
 - Finite state processes
 - Unbounded lossy channels
 - Send & receive operations
- Motivation:**
 - Models of communication protocols

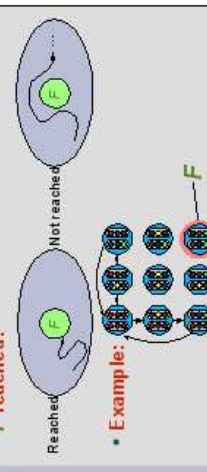


Slide 11

Reachability

- F – set of "target states"
- F reached?

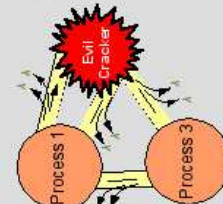
Example:



Slide 8

Game Probabilistic Game Channel Systems (GPLCS)

- Game:**
 - Interaction with evil cracker
- Probabilistic:**
 - Messages lost randomly (probability λ)



Slide 12

Background Model Algorithm Conclusion

Game Probabilistic Lossy Channel Systems (GPLCS)

Sufficient:

- One channel
- One process
- Each state controlled by a player

Properties:

- Infinite state space
- Perfect channel = Turing machine

Slide 13

Background Model Algorithm Conclusion

Game Probabilistic Lossy Channel Systems (GPLCS)

State: $s = (A, msgn)$

Slide 14

Background Model Algorithm Conclusion

Game Probabilistic Lossy Channel Systems (GPLCS)

Transitions:

Send: $A \xrightarrow{1/m} B$, $m \ p \ p \ m \dots$

Receive: $B \xrightarrow{1/n} A$, $m \ p \ p \ m \dots$

Slide 15

Background Model Algorithm Conclusion

Game Probabilistic Lossy Channel Systems (GPLCS)

Transitions:

Send: $A \xrightarrow{1/m} B$, $m \ p \ p \ m \dots$

Receive: $B \xrightarrow{1/n} A$, $m \ p \ p \ m \dots$

Slide 16

Background Model Algorithm Conclusion

Game Probabilistic Lossy Channel Systems (GPLCS)

Transitions:

Send: $A \xrightarrow{1/m} B$, $m \ p \ p \ m \dots$

Receive: $B \xrightarrow{1/n} A$, $m \ p \ p \ m \dots$

No-op: $A \xrightarrow{1/n} A$, $m \ p \ p \ m \dots$

Slide 17

Background Model Algorithm Conclusion

Game Probabilistic Lossy Channel Systems (GPLCS)

Probabilistic message loss:

$p \ p \ n \ p \dots$

$p \ n \ n \dots$

Each transition: each message lost with prob $\lambda > 0$, independently

Slide 18

Background Model Algorithm Conclusion

Stochastic Games

Every GPLCS induces a stochastic game

3 types of states:

- Player GOOD
- Player BAD
- Player RANDOM

Slide 20

Background Model Algorithm Conclusion

Stochastic Games

Strategy = selection of outgoing transitions

Strategies for GOOD & BAD

Only probabilistic choices remain

"Prob(event)" well-defined

Slide 21

Background Model Algorithm Conclusion

Stochastic Games

Strategy = selection of outgoing transitions

Strategies for GOOD & BAD

Only probabilistic choices remain

"Prob(event)" well-defined

Slide 22

Background Model Algorithm Conclusion

Overview

- Background
- Models
- Algorithm
- Conclusion

Slide 23

Background Model Algorithm Conclusion

Repeated Reachability for GPLCS

Input:

- GPLCS
- Set F of final states

Output: Partition of states:

- Winning for BAD
- Winning for GOOD

GOOD forces $\text{Prob}(\text{reach } F \text{ inf. often}) = 1$

BAD forces $\text{Prob}(\text{reach } F \text{ inf. often}) < 1$

Slide 24

Background
Model
Algorithm
Conclusion

Subroutine: Force-set Correctness

- Force-set**
 - Given target Q , compute the set of states where Q can force Q_0 by construction
- Backward** (and I'll show why)
 - $Q_1 = 1$ step before Q_0
 - $Q_2 = 1$ step before Q_1
 - $Q_3 = 1$ step before Q_2
 - $Q_4 = 1$ step before Q_3
 - $Q_5 = 1$ step before Q_4
 - $Q_6 = Q_0$

Slide 37

Background
Model
Algorithm
Conclusion

Subroutine: Force-set Correctness

- Force-set**
 - Given target Q , compute the set of states where Q can force Q_0 by construction
- Backward** search
 - $Q_1 = 1$ step before Q_0
 - $Q_2 = 1$ step before Q_1
 - $Q_3 = 1$ step before Q_2
 - $Q_4 = 1$ step before Q_3
 - $Q_5 = 1$ step before Q_4
 - $Q_6 = Q_0$

Slide 38

Background
Model
Algorithm
Conclusion

Subroutine: Force-set Correctness

- Force-set**
 - Given target Q , compute the set of states where Q can force Q_0 by construction
- Backward** search
 - $Q_1 = 1$ step before Q_0
 - $Q_2 = 1$ step before Q_1
 - $Q_3 = 1$ step before Q_2
 - $Q_4 = 1$ step before Q_3
 - $Q_5 = 1$ step before Q_4
 - $Q_6 = Q_0$

Slide 39

Background
Model
Algorithm
Conclusion

Algorithm Idea

- A world of islands and water
- No food ($\neq F$) on islands
- In water, may float to island (possibly elsewhere, too)
- Find **BA** islands and water!

Slide 41

Background
Model
Algorithm
Conclusion

Algorithm Idea

- A world of islands and water
- No food ($\neq F$) on islands
- In water, may float to island (possibly elsewhere, too)
- Find **BA** islands and water!

Slide 42

Background
Model
Algorithm
Conclusion

Algorithm Idea

- A world of islands and water
- No food ($\neq F$) on islands
- In water, may float to island (possibly elsewhere, too)
- Find **BA** islands and water!

Slide 43

Background
Model
Algorithm
Conclusion

Algorithm Idea

- A world of islands and water
- No food ($\neq F$) on islands
- In water, may float to island (possibly elsewhere, too)
- Find **BA** islands and water!

Slide 45

Background
Model
Algorithm
Conclusion

Algorithm Idea

- A world of islands and water
- No food ($\neq F$) on islands
- In water, may float to island (possibly elsewhere, too)
- Find **BA** islands and water!

Slide 46

Background
Model
Algorithm
Conclusion

Algorithm Idea

- A world of islands and water
- No food ($\neq F$) on islands
- In water, may float to island (possibly elsewhere, too)
- Find **BA** islands and water!

Slide 47

Background
Model
Algorithm
Conclusion

Algorithm Idea

- A world of islands and water
- No food ($\neq F$) on islands
- In water, may float to island (possibly elsewhere, too)
- Find **BA** islands and water!

Slide 40

Background
Model
Algorithm
Conclusion

Algorithm Idea

- A world of islands and water
- No food ($\neq F$) on islands
- In water, may float to island (possibly elsewhere, too)
- Find **BA** islands and water!

Slide 44

Background
Model
Algorithm
Conclusion

Algorithm Idea

- A world of islands and water
- No food ($\neq F$) on islands
- In water, may float to island (possibly elsewhere, too)
- Find **BA** islands and water!

Slide 48

Background Model Algorithm Conclusion

Algorithm Overview

Slide 49

Background Model Algorithm Conclusion

Algorithm Overview

Island 0

- Compute Q : G000 can force $\text{Prob}(\text{reach } F) > 0$

Slide 50

Background Model Algorithm Conclusion

Algorithm Overview

Island 0

- Compute Q : G000 can force $\text{Prob}(\text{reach } F) > 0$
- $I_0 = \text{complement}(Q)$
- So BAD can force $\text{Prob}(\text{reach } F) = 0$ on I_0

Slide 51

Background Model Algorithm Conclusion

Algorithm Overview

Water 0

- Compute Q : BAD can force $\text{Prob}(\text{reach } I_0) > 0$

Slide 52

Background Model Algorithm Conclusion

Algorithm Overview

Water 0

- Compute Q : BAD can force $\text{Prob}(\text{reach } I_0) > 0$
- $W_0 = Q$

Slide 53

Background Model Algorithm Conclusion

Algorithm Overview

Island 1

- Compute Q : G000 can force $\text{Prob}(\text{reach } F) > 0$, **avoiding I_0 and W_0**

Slide 54

Background Model Algorithm Conclusion

Algorithm Overview

Island 1

- Compute Q : G000 can force $\text{Prob}(\text{reach } F) > 0$, **avoiding I_0 and W_0**
- $I_1 = \text{complement}(Q)$

Slide 55

Background Model Algorithm Conclusion

Algorithm Overview

Water 1

- Compute Q : BAD can force $\text{Prob}(\text{reach } I_0 \cup W_0 \cup I_1) > 0$

Slide 56

Background Model Algorithm Conclusion

Algorithm Overview

Water 1

- Compute Q : BAD can force $\text{Prob}(\text{reach } I_0 \cup W_0 \cup I_1) > 0$
- $W_1 = Q$

Slide 57

Background Model Algorithm Conclusion

Algorithm Overview

Island 2

- Compute Q : G000 can force $\text{Prob}(\text{reach } F) > 0$, **avoiding I_0, W_0, I_1, W_1**

Slide 58

Background Model Algorithm Conclusion

Algorithm Overview

Island 2

- Compute Q : G000 can force $\text{Prob}(\text{reach } F) > 0$, **avoiding I_0, W_0, I_1, W_1**
- $I_2 = \text{complement}(Q)$

Slide 59

Background Model Algorithm Conclusion

Algorithm Overview

Slide 60

Background Model Algorithm Conclusion

Algorithm Convergence

Converges?

Slide 61

Background Model Algorithm Conclusion

Algorithm Convergence

Converges?

YES! for GPLCS
(well quasi orders, *difficult*)

Slide 62

Background Model Algorithm Conclusion

Algorithm Correctness

Correct?

Slide 63

Background Model Algorithm Conclusion

Algorithm Correctness

Correct?

YES!
(and I'll show why)

Slide 64

Background Model Algorithm Conclusion

Algorithm Correctness

Correct?

Bao can force
Prob(reach F inf. often) < 1

Good can force
Prob(reach F inf. often) $= 1$

Slide 65

Background Model Algorithm Conclusion

Algorithm Correctness

Correct?

Bao can force
Prob(reach F inf. often) < 1

Slide 66

Background Model Algorithm Conclusion

Algorithm Correctness

Correct?

Bao can force
Prob(reach F inf. often) < 1

In I_p^c :
Prob(reach F inf. often) < 1
(since even
Prob(reach F) $= 0$)

In W_p :
Prob(reach F inf. often) < 1
(since Prob(reach I_p) > 0)

Slide 68

Background Model Algorithm Conclusion

Algorithm Correctness

Correct?

Bao can force
Prob(reach F inf. often) < 1

In I_p^c :
Prob(reach F inf. often) < 1
(since even
Prob(reach F) $= 0$)

Good chooses how to die:
• stay in I_p and lose (no F)
• or go to W_p , and lose

In I_p :
Good chooses how to die:
• stay in I_p and lose (no F)
• or go to W_p , and lose

In W_p :
Prob(reach F inf. often) < 1
(since Prob(reach I_p) > 0)

So BAD wins
with prob. > 0
in all W_p , I_p loses how to die:
• stay in I_p and lose (no F)
• or go to W_p , and lose

Slide 69

Background Model Algorithm Conclusion

Algorithm Correctness

Correct?

Bao can force
Prob(reach F inf. often) < 1

In I_p^c :
Prob(reach F inf. often) < 1
(since even
Prob(reach F) $= 0$)

In W_p :
Prob(reach F inf. often) < 1
(since Prob(reach I_p) > 0)

So BAD wins
with prob. > 0
in all W_p , I_p loses how to die:
• stay in I_p and lose (no F)
• or go to W_p , and lose

Slide 70

Background Model Algorithm Conclusion

Algorithm Correctness

Correct?

Good can force
Prob(reach F inf. often) $= 1$

Slide 71

Background Model Algorithm Conclusion

Algorithm Correctness

Correct?

Good can force
Prob(reach F inf. often) $= 1$

• **Convergence means:**
– No more water
– No more island

Slide 72

Background Model Algorithm Conclusion

Algorithm Correctness

Correct?

So Good can force:
Always $\text{Prob}(\text{reach } F) > 0$

For GPLCS, this implies
 $\text{Prob}(\text{reach } F \text{ inf. often}) = 1$
(using attractors, difficult)

- Convergence means:
 - No more water $\Rightarrow$ Good can stay outside I_n, W_n
 - No more island $\Rightarrow$ Good can force $\text{Prob}(\text{reach } F) > 0$

Slide 73

Background Model Algorithm Conclusion

Algorithm Correctness

Correct?

So Good can force:
Always $\text{Prob}(\text{reach } F) > 0$

Convergence means:

- No more water $\Rightarrow$ Good can stay outside I_n, W_n
- No more island $\Rightarrow$ Good can force $\text{Prob}(\text{reach } F) > 0$

Slide 74

Background Model Algorithm Conclusion

Algorithm Correctness

Correct?

So Good can force:
Always $\text{Prob}(\text{reach } F) > 0$

Convergence means:

- No more water $\Rightarrow$ Good can stay outside I_n, W_n
- No more island $\Rightarrow$ Good can force $\text{Prob}(\text{reach } F) > 0$

Slide 75

Background Model Algorithm Conclusion

Overview

- Background
- Models
- Algorithm
- Conclusion

Slide 77

Background Model Algorithm Conclusion

The Problem We Studied

- GPLCS: Protocol vs. Cracker
- Repeated reachability: reach "good" state infinitely many times

Slide 78

Background Model Algorithm Conclusion

Extensions

- Construct winning strategies
- Concurrent games: Simultaneous moves
- Reachability games: Can Good force $\text{Prob}(\text{reach } F) = 1$?
- Parity games: Generalization of repeated reachability
- Unclear

Future

Future

Background Model Algorithm Conclusion

The End

Questions?

Slide 81

Background Model Algorithm Conclusion Bonus Slide

Convergence: Well Quasi Orders

- Recall: $\text{State: } s = (A, \text{repp})$
- Subword ordering: $\text{repp} \sqsubseteq \text{repp}'$
- Upward closure: all "bigger" states
- Message loss $\Rightarrow$ subword

Slide 82

Background Model Algorithm Conclusion Bonus Slide

Convergence: Well Quasi Orders

If we can lose some messages...

Slide 83

Background Model Algorithm Conclusion Bonus Slide

Convergence: Well Quasi Orders

If we can lose some messages...
...then we can lose more!

Slide 84

Background Model Algorithm Conclusion Bonus Slide!

Convergence: Well Quasi Orders

If we can lose some messages...
...then we can lose more!

Conclusion:
RANDOM states one step before are upward closed

Slide 85

Background Model Algorithm Conclusion Bonus Slide!

Convergence of Force-sets: Well Quasi Orders

- Theorem (Higman 1952):**
Any sequence of increasing, upward closed sets converges.
- We apply it:**

Player states
Random states

Slide 86

Background Model Algorithm Conclusion Bonus Slide!

Convergence of Force-sets: Well Quasi Orders

- Theorem (Higman 1952):**
Any sequence of increasing, upward closed sets converges.
- We apply it:**

Player states
Random states

Slide 87

Background Model Algorithm Conclusion Bonus Slide!

Convergence of Force-sets: Well Quasi Orders

- Theorem (Higman 1952):**
Any sequence of increasing, upward closed sets converges.
- Every 2nd step is a loss $\Rightarrow$ terminates!**
- Upward closed $\Rightarrow$ terminates!**

Player states
Random states

Slide 88

Background Model Algorithm Conclusion Bonus Slide!

Convergence of Algorithms: Well Quasi Orders

- Argument is much more difficult!
- RANDOM** states in water upward closed
- Sequence of waters terminates
- Sequence of islands terminates

Player states
Random states

Slide 89

Background Model Algorithm Conclusion Bonus Slide!

Finite-Memory Strategies

The Fine Print:

- We assume BAD is restricted to finite-memory strategies**
- BAD cannot remember** unbounded amount of history
- Intuition:** Evil cracker has a finite hard disk
- Needed to prove correctness

Slide 90

Background Model Algorithm Conclusion Bonus Slide!

Leftover

Slide 92

Background Model Algorithm Conclusion

Repeated Reachability

- F** – set of “target states”
- Is F reached infinitely often?**
- Example:**
– Will it always return to the green state?

Slide 93