



# Quantitative Abstraction Refinement

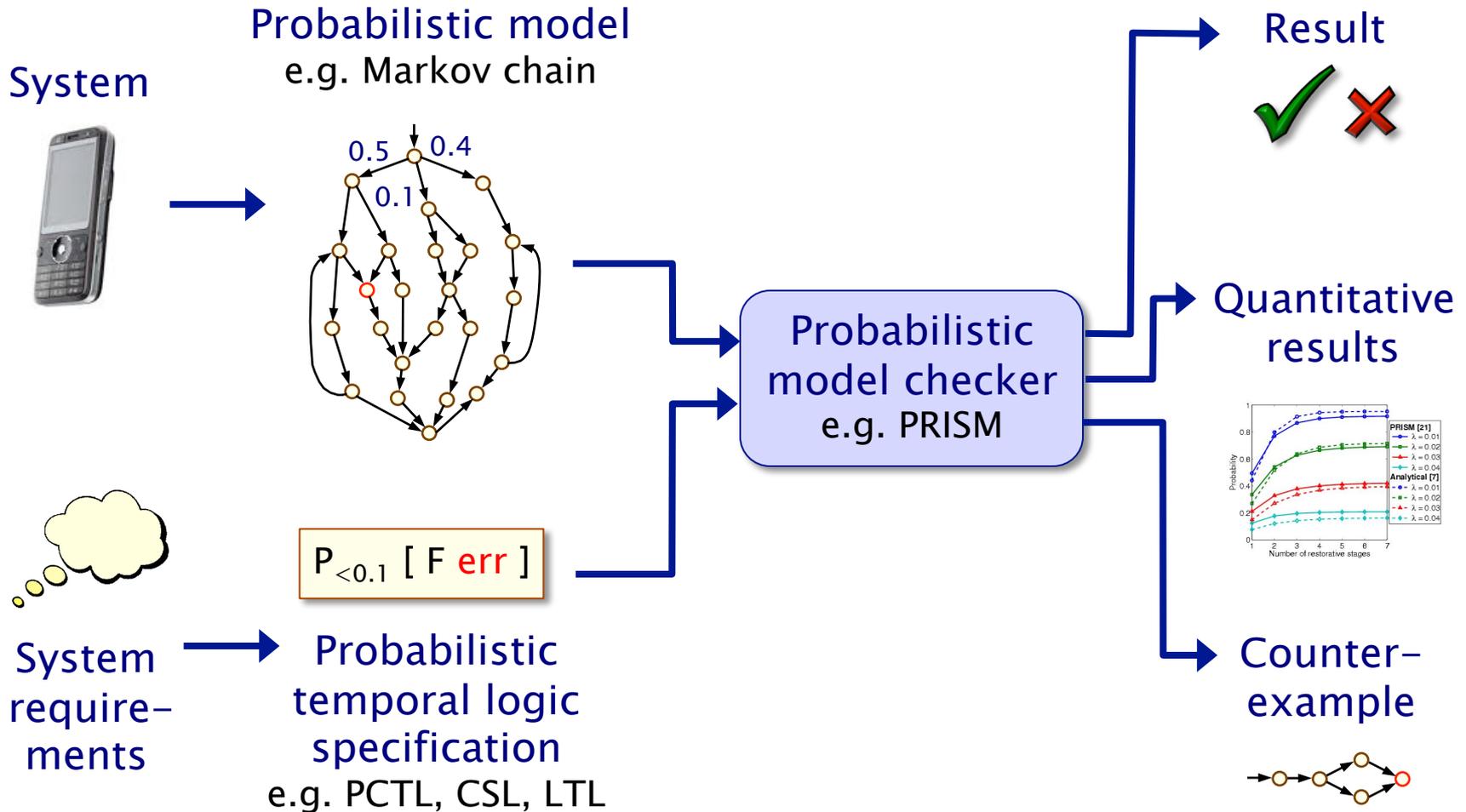
Dave Parker

Oxford University Computing Laboratory

Dagstuhl, July 2010

**Joint work with:** Marta Kwiatkowska, Gethin Norman, Mark Kattenbelt

# Probabilistic model checking



# Why probability?

- Why probability?
  - many systems we want to verify are inherently probabilistic
- **Randomisation**, e.g. in distributed coordination algorithms
  - random delays/back-off in Bluetooth, CSMA/CD, IEEE 802.11
  - randomised algorithms for anonymity, contract signing, ...
- **Uncertainty or unreliable** behaviour
  - e.g. messages losses/delays over wireless communication
  - e.g. component failures
  - e.g. reaction rates in biological systems
- Need formal techniques for **quantitative** guarantees of:
  - safety, reliability, performance, dependability, resource usage, security, privacy, trust, anonymity, fairness, ...

# Probabilistic models

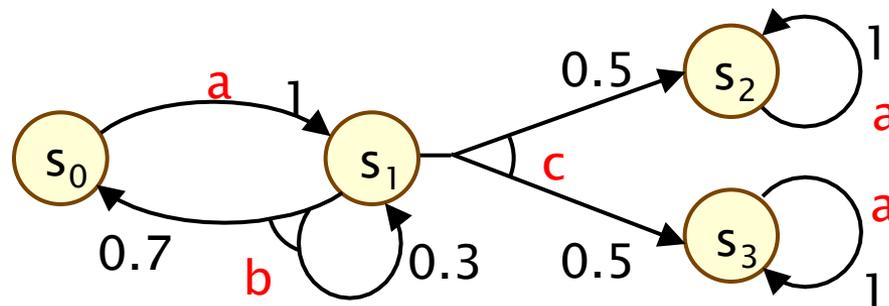
	Fully probabilistic	Nondeterministic
Discrete time	Discrete-time Markov chains (DTMCs) $S \times S \rightarrow [0,1]$	Markov decision processes (MDPs) (probabilistic automata) $S \times \text{Act} \rightarrow \text{Dist}(S)$
Continuous time	Continuous-time Markov chains (CTMCs) $S \times S \rightarrow \mathbb{R}_{\geq 0}$	CTMDPs/IMCs
		Probabilistic timed automata (PTAs)

# Overview

- Probabilistic model checking
  - Markov decision processes (MDPs)
  - probabilistic timed automata (PTAs)
- Abstraction for probabilistic models
  - abstractions of MDPs (stochastic two-player games)
- Quantitative abstraction refinement
  - abstraction-refinement loop
  - probabilistic model checking for PTAs
  - also: verification of probabilistic software
- Conclusions & current/future work

# Markov decision processes (MDPs)

- Model **nondeterministic** as well as **probabilistic** behaviour
  - e.g concurrency, environmental factors, under-specification, ...
- Formally, an MDP is a tuple **(S, Act, Steps)** where:
  - **S** is a set of states
  - **Act** is a set of actions
  - **Steps** :  $S \times \text{Act} \rightarrow \text{Dist}(S)$  is the transition probability function



- An **adversary** (aka. “scheduler” or “policy”) of an MDP
  - is a resolution of the nondeterminism in the MDP
  - under a given adversary  $\sigma$  the behaviour is fully probabilistic

# Probabilistic reachability for MDPs

- Probabilistic reachability

- fundamental concept in the quantitative verification of MDPs
- $p_s^\sigma(F)$  = probability of reaching  $F$  starting from  $s$  under  $\sigma$
- consider the minimum/maximum values over all adversaries
- $p_s^{\min}(F) = \inf_\sigma p_s^\sigma(F)$  and  $p_s^{\max}(F) = \sup_\sigma p_s^\sigma(F)$



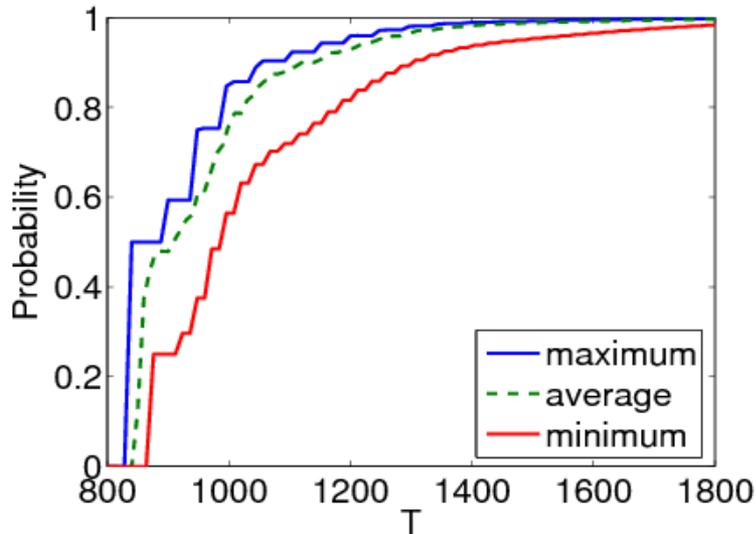
- can be computed efficiently (and corresponding adversaries)

- Allows reasoning about best/worst-case behaviour

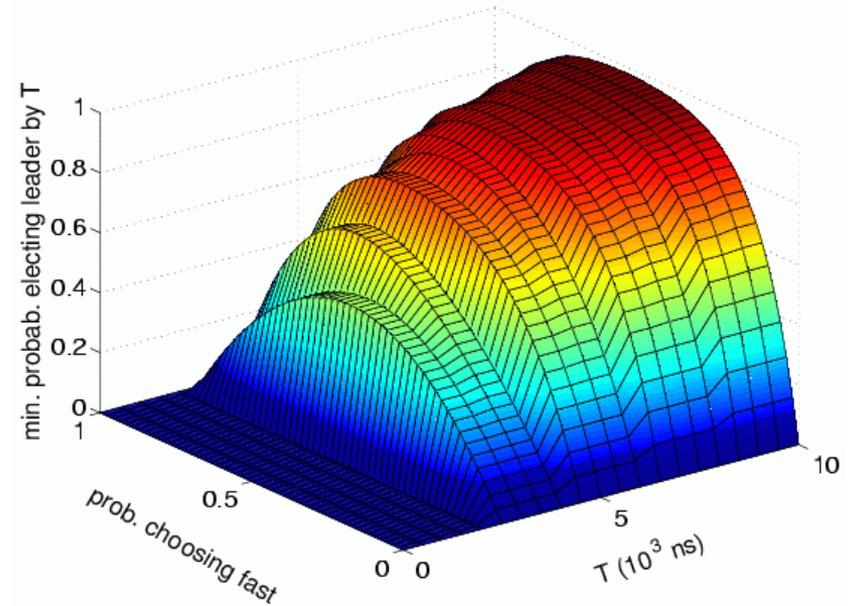
- e.g. minimum probability of the protocol terminating correctly
- e.g. maximum probability of a security breach

# Probabilistic reachability for MDPs

- Often focus on **quantitative** properties:



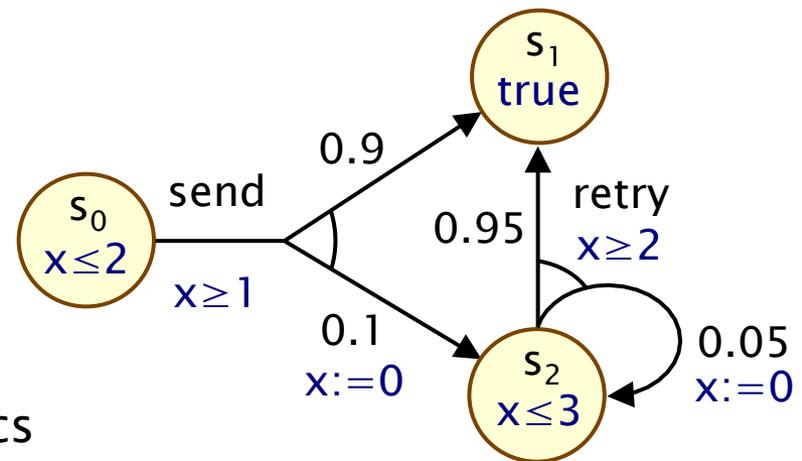
CSMA/CD network protocol:  
Maximum, average and minimum probability that a message is sent successfully by time  $T$



FireWire protocol:  
Worst case (minimum) probability of electing a leader by time  $T$  for various coin biases

# Probabilistic timed automata

- Probabilistic timed automata (PTAs)
  - Markov decision processes + real-valued clocks
  - or: timed automata + discrete probabilistic choice
  - models **timed**, **probabilistic** and **nondeterministic** behaviour
  - essential e.g. for communication protocols such as Zigbee, Bluetooth, which feature **delays**, **randomisation**, **failures** and **concurrency**



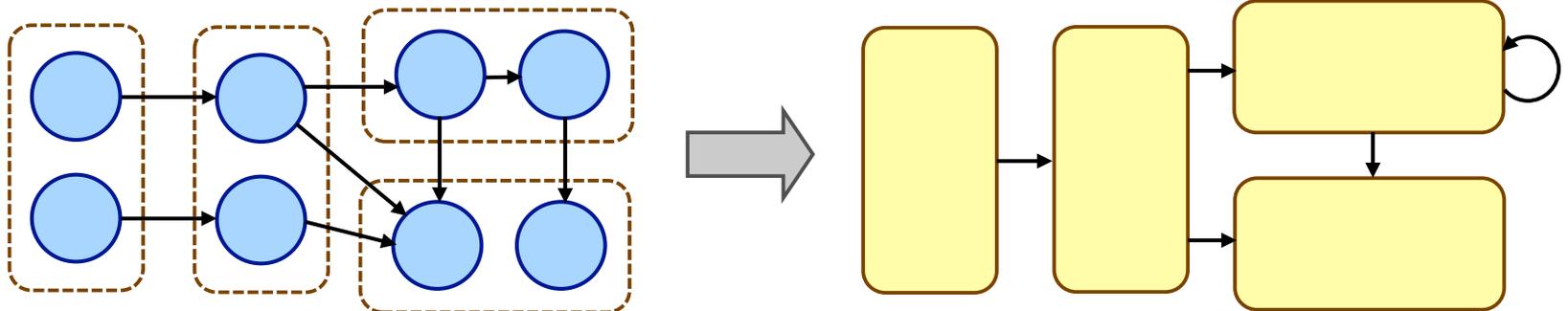
- PTA model checking
  - infinite-state MDP semantics
  - probabilistic (timed) reachability

# Overview

- Probabilistic model checking
  - Markov decision processes (MDPs)
  - probabilistic timed automata (PTAs)
- **Abstraction for probabilistic models**
  - abstractions of MDPs (stochastic two-player games)
- Quantitative abstraction refinement
  - abstraction-refinement loop
  - probabilistic model checking for PTAs
  - also: verification of probabilistic software
- Conclusions & current/future work

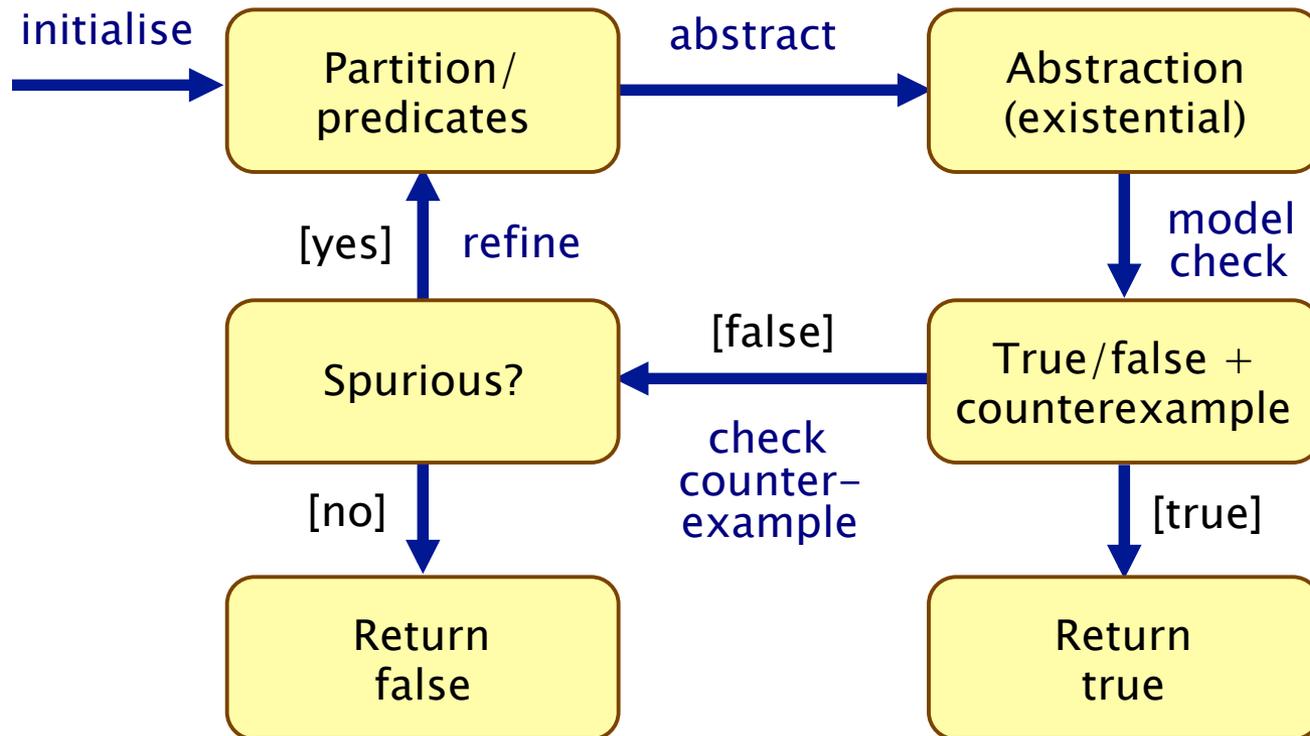
# Abstraction

- Very successful in (non-probabilistic) formal methods
  - essential for verification of large/infinite-state systems
  - hide details irrelevant to the property of interest
  - yields smaller/finite model which is easier/feasible to verify
  - loss of precision: verification can return “don’t know”
- Construct abstract model of a concrete system
  - e.g. based on a partition of the concrete state space
  - an **abstract state** represents a set of **concrete states**



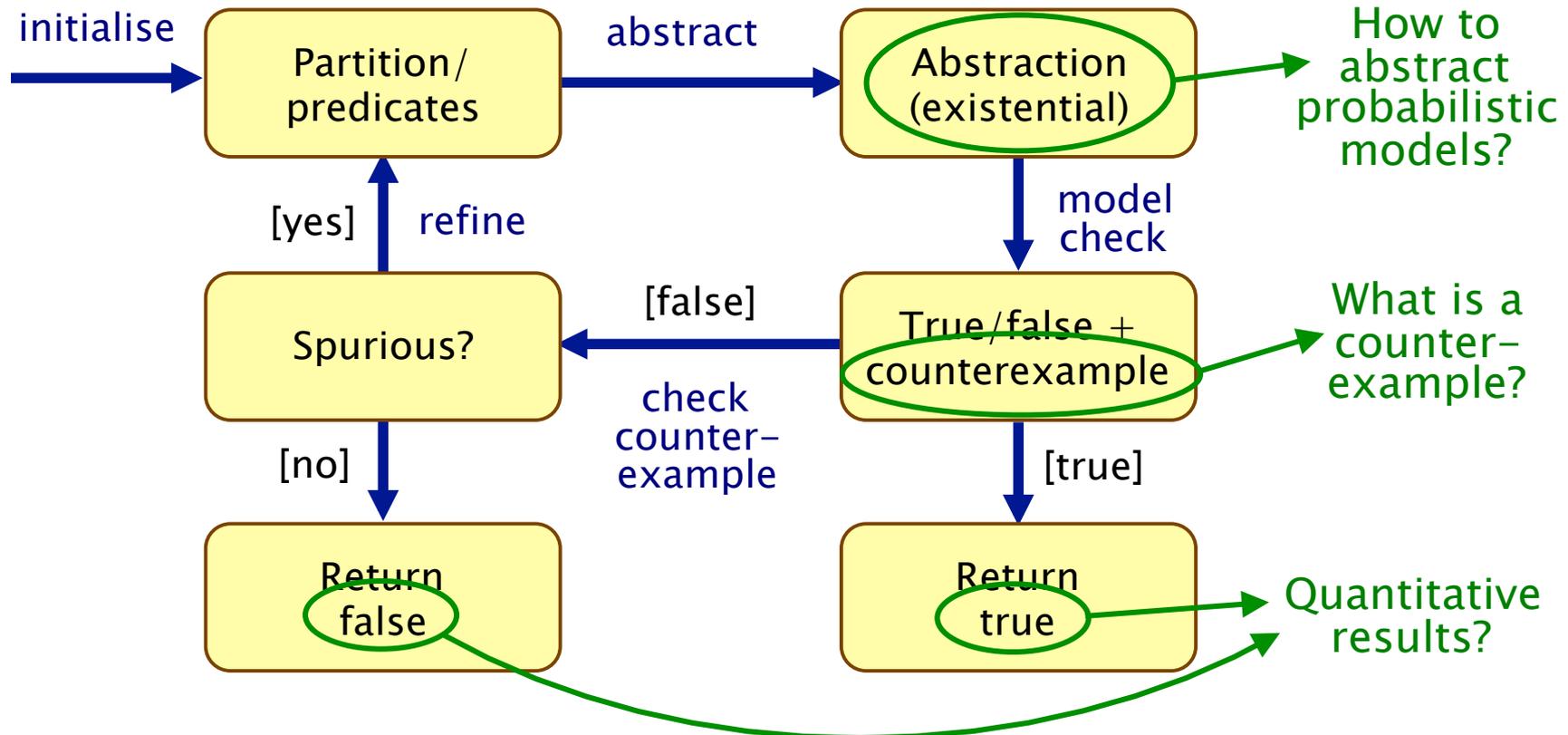
# Abstraction refinement (CEGAR)

- Counterexample-guided abstraction refinement
  - (non-probabilistic) model checking of reachability properties



# Abstraction refinement (CEGAR)

- Counterexample-guided abstraction refinement
  - ~~(non-probabilistic)~~ model checking of reachability properties



# Abstraction of MDPs

- Abstraction increases degree of nondeterminism
  - i.e. minimum probabilities are lower and maximums higher



- But what form does the abstraction of an MDP take?
  - (i) an MDP [D'Argenio et al.'01]
    - probabilistic simulation relates concrete/abstract models
  - (ii) a stochastic two-player game [QEST'06]
    - separates nondeterminism from abstraction and from MDP
    - yields separate lower/upper bounds for min/max



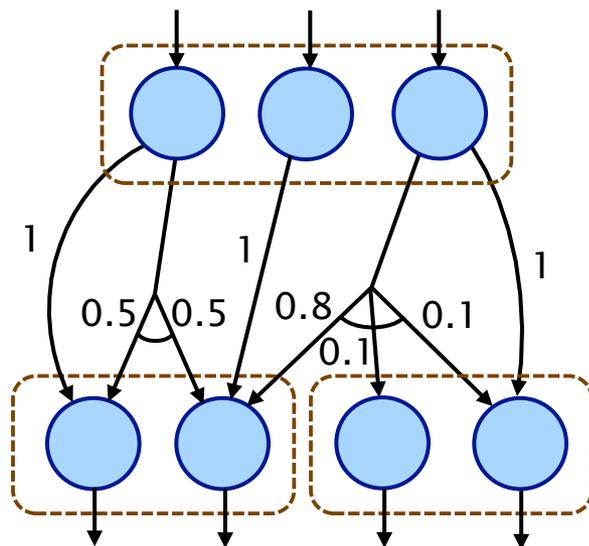
# Stochastic two-player games

- Subclass of simple stochastic games [Shapley, Condon]
  - two nondeterministic players (1 and 2) and probabilistic choice
- Resolution of the nondeterminism in a game
  - corresponds to a pair of **strategies** for players 1 and 2:  $(\sigma_1, \sigma_2)$
  - $p_a^{\sigma_1, \sigma_2}(F)$  probability of reaching **F** from **a** under  $(\sigma_1, \sigma_2)$
  - can compute, e.g. :  $\sup_{\sigma_1} \inf_{\sigma_2} p_a^{\sigma_1, \sigma_2}(F)$
  - informally: “the maximum probability of reaching **F** that player 1 can guarantee no matter what player 2 does”
- Abstraction of an MDP as a stochastic two-player game:
  - **player 1** controls the nondeterminism of the abstraction
  - **player 2** controls the nondeterminism of the MDP

# Game abstraction (by example)

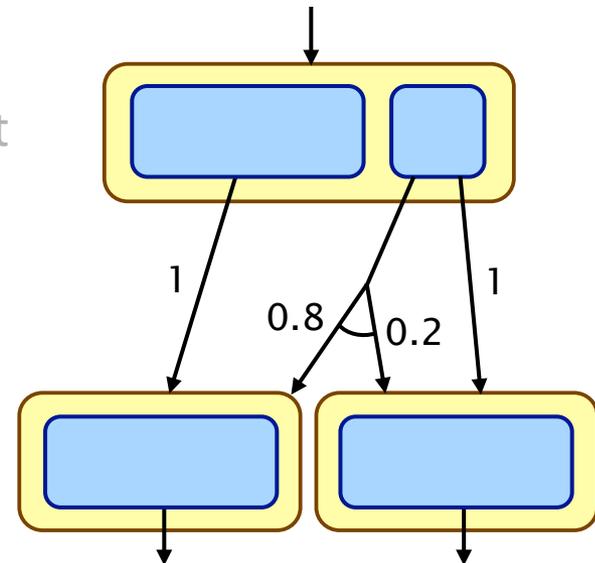
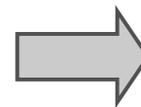
- Player **1** vertices are partition elements (abstract states)
- (Sets of) distributions are lifted to the abstract state space
- States with same (sets of) choices form player **2** vertices

MDP (fragment)



Stochastic game (fragment)

abstract



# Properties of the abstraction

- Analysis of game yields lower/upper bounds:
  - for target  $F \in A$ ,  $s \in S$  and  $a \in A$  with  $s \in a$

$$\inf_{\sigma_1, \sigma_2} p_a^{\sigma_1, \sigma_2}(F) \leq p_s^{\min}(F) \leq \sup_{\sigma_1} \inf_{\sigma_2} p_a^{\sigma_1, \sigma_2}(F)$$

$$\inf_{\sigma_1} \sup_{\sigma_2} p_a^{\sigma_1, \sigma_2}(F) \leq p_s^{\max}(F) \leq \sup_{\sigma_1, \sigma_2} p_a^{\sigma_1, \sigma_2}(F)$$

# Properties of the abstraction

- Analysis of game yields lower/upper bounds:
  - for target  $F \in A$ ,  $s \in S$  and  $a \in A$  with  $s \in a$

$$\inf_{\sigma_1, \sigma_2} p_a^{\sigma_1, \sigma_2}(F) \leq p_s^{\min}(F) \leq \sup_{\sigma_1} \inf_{\sigma_2} p_a^{\sigma_1, \sigma_2}(F)$$
$$\inf_{\sigma_1} \sup_{\sigma_2} p_a^{\sigma_1, \sigma_2}(F) \leq p_s^{\max}(F) \leq \sup_{\sigma_1, \sigma_2} p_a^{\sigma_1, \sigma_2}(F)$$

min/max reachability probabilities for original MDP



# Properties of the abstraction

- Analysis of game yields lower/upper bounds:
  - for target  $F \in A$ ,  $s \in S$  and  $a \in A$  with  $s \in a$

$$\begin{array}{ccccc}
 \inf_{\sigma_1, \sigma_2} p_a^{\sigma_1, \sigma_2}(F) & \leq & p_s^{\min}(F) & \leq & \sup_{\sigma_1} \inf_{\sigma_2} p_a^{\sigma_1, \sigma_2}(F) \\
 \inf_{\sigma_1} \sup_{\sigma_2} p_a^{\sigma_1, \sigma_2}(F) & \leq & p_s^{\max}(F) & \leq & \sup_{\sigma_1, \sigma_2} p_a^{\sigma_1, \sigma_2}(F)
 \end{array}$$

optimal probabilities for player 1, player 2 in game

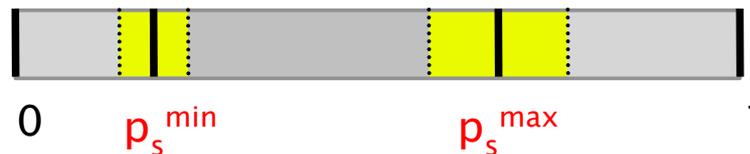


# Properties of the abstraction

- Analysis of game yields lower/upper bounds:
  - for target  $F \in A$ ,  $s \in S$  and  $a \in A$  with  $s \in a$

$$\inf_{\sigma_1, \sigma_2} p_a^{\sigma_1, \sigma_2}(F) \leq p_s^{\min}(F) \leq \sup_{\sigma_1} \inf_{\sigma_2} p_a^{\sigma_1, \sigma_2}(F)$$
$$\inf_{\sigma_1} \sup_{\sigma_2} p_a^{\sigma_1, \sigma_2}(F) \leq p_s^{\max}(F) \leq \sup_{\sigma_1, \sigma_2} p_a^{\sigma_1, \sigma_2}(F)$$

min/max reachability probabilities, treating game as MDP  
(i.e. assuming that players 1 and 2 cooperate)

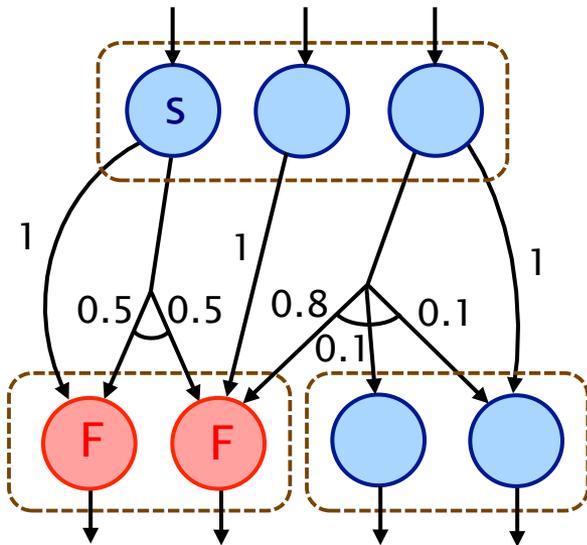


# Example

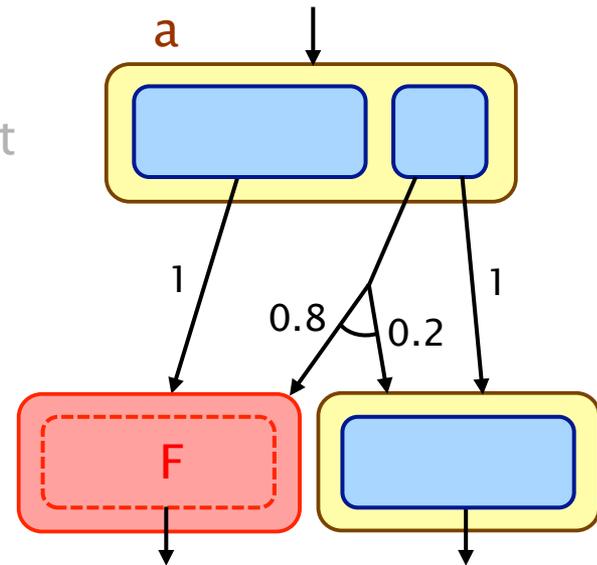
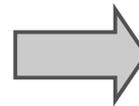
$$p_s^{\max} (F) = 1 \in [0.8, 1]$$

$$\inf_{\sigma_1} \sup_{\sigma_2} p_a^{\sigma_1, \sigma_2} (F) = 0.8$$

$$\sup_{\sigma_1, \sigma_2} p_a^{\sigma_1, \sigma_2} (F) = 1$$

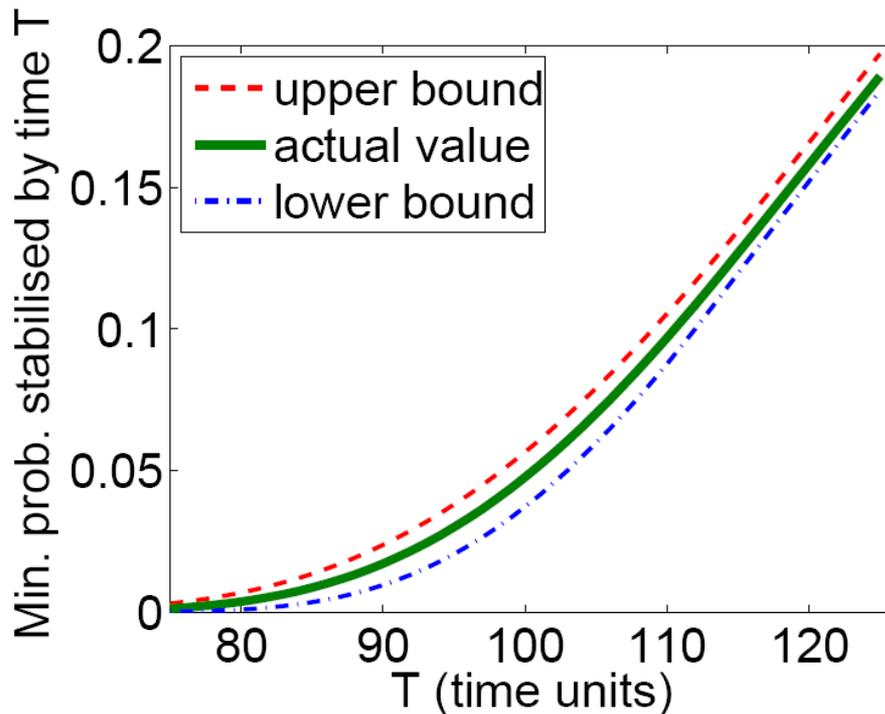


abstract



# Abstraction: Example results

- Israeli & Jalfon's Self Stabilisation [IJ90]
  - protocol for obtaining a stable state in a token ring
  - minimum probability of reaching a stable state by time  $T$



concrete states: 1,048,575  
abstract states: 627

# Nondeterministic abstractions

- We can consider a general class of “nondeterministic” abstractions for probabilistic models

Concrete model:		Abstraction:
DTMC	→	MDP
MDP	→	STPG
CTMC	→	CTMDP
CTMDP	→	CTSTPG

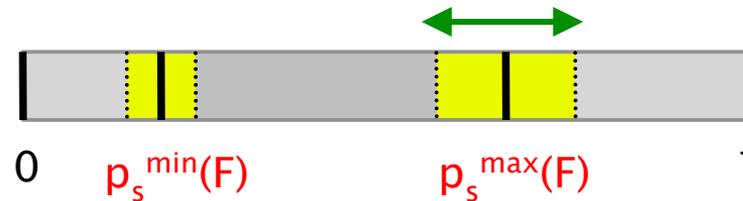
- CTMDP = continuous-time Markov decision process
- CTSTPG = continuous-time stochastic two-player game

# Overview

- Probabilistic model checking
  - Markov decision processes (MDPs)
  - probabilistic timed automata (PTAs)
- Abstraction for probabilistic models
  - abstractions of MDPs (stochastic two-player games)
- **Quantitative abstraction refinement**
  - abstraction-refinement loop
  - probabilistic model checking for PTAs
  - also: verification of probabilistic software
- Conclusions & current/future work

# Abstraction refinement

- Consider (max) difference between lower/upper bounds
  - gives a **quantitative measure** of the abstraction's **precision**

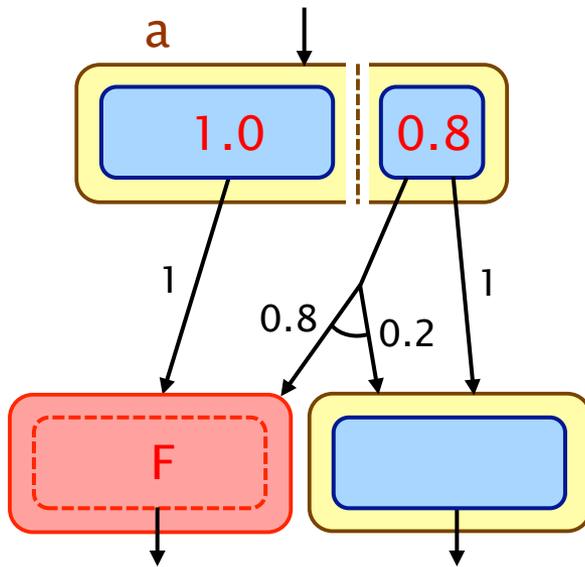


- If the difference (“error”) is too great, **refine** the abstraction
  - a finer partition yields a more precise abstraction
  - lower/upper bounds can tell us **where** to refine (which states)
  - (memoryless) strategies can tell us **how** to refine

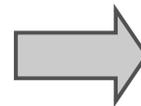
# Example

$$p_s^{\max}(F) = 1 \in [0.8, 1]$$

“error” = 0.2

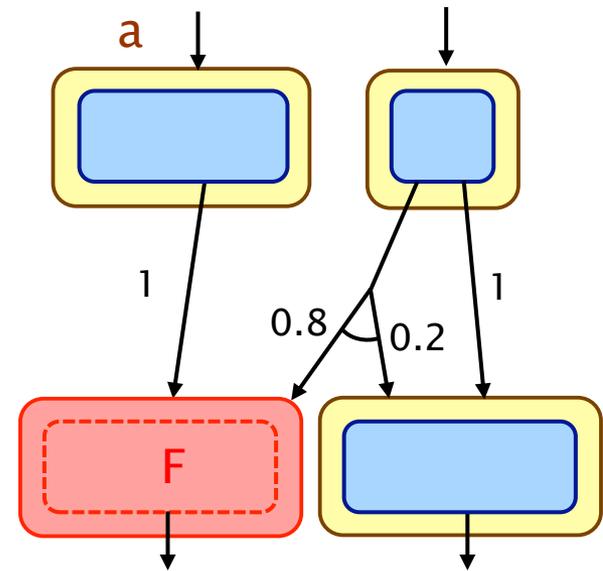


refine



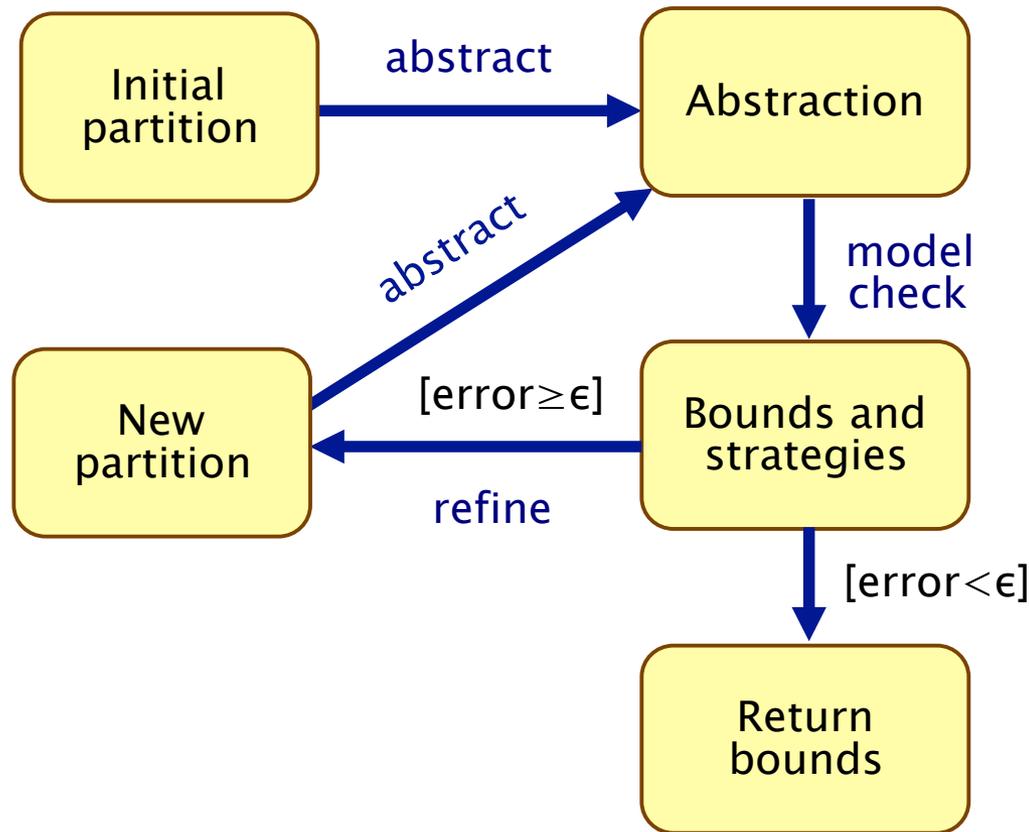
$$p_s^{\max}(F) = 1 \in [1, 1]$$

“error” = 0



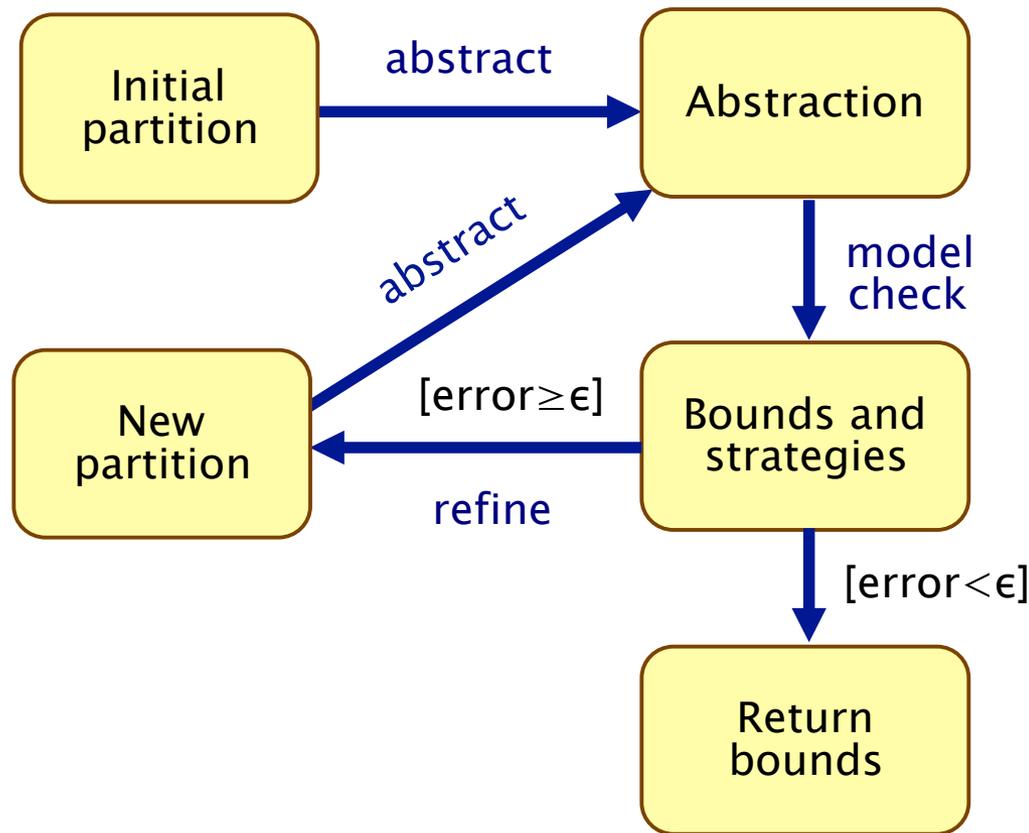
# Abstraction-refinement loop

- Quantitative abstraction-refinement loop for MDPs



# Abstraction–refinement loop

- **Quantitative** abstraction–refinement loop for MDPs



- Refinements yield strictly finer partition

- Guaranteed to converge for finite models

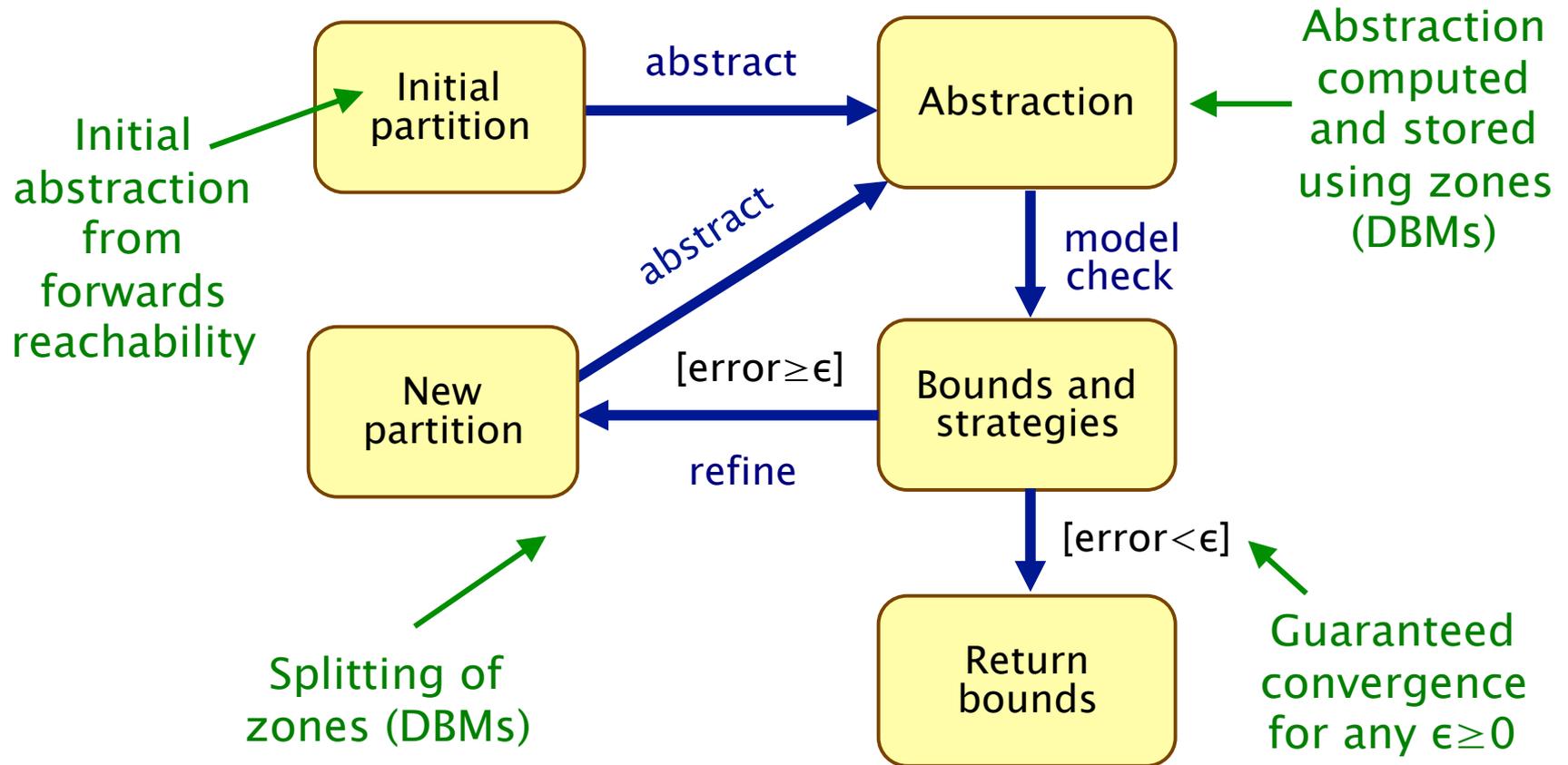
- Guaranteed to converge for infinite models with finite bisimulation

# Abstraction–refinement loop

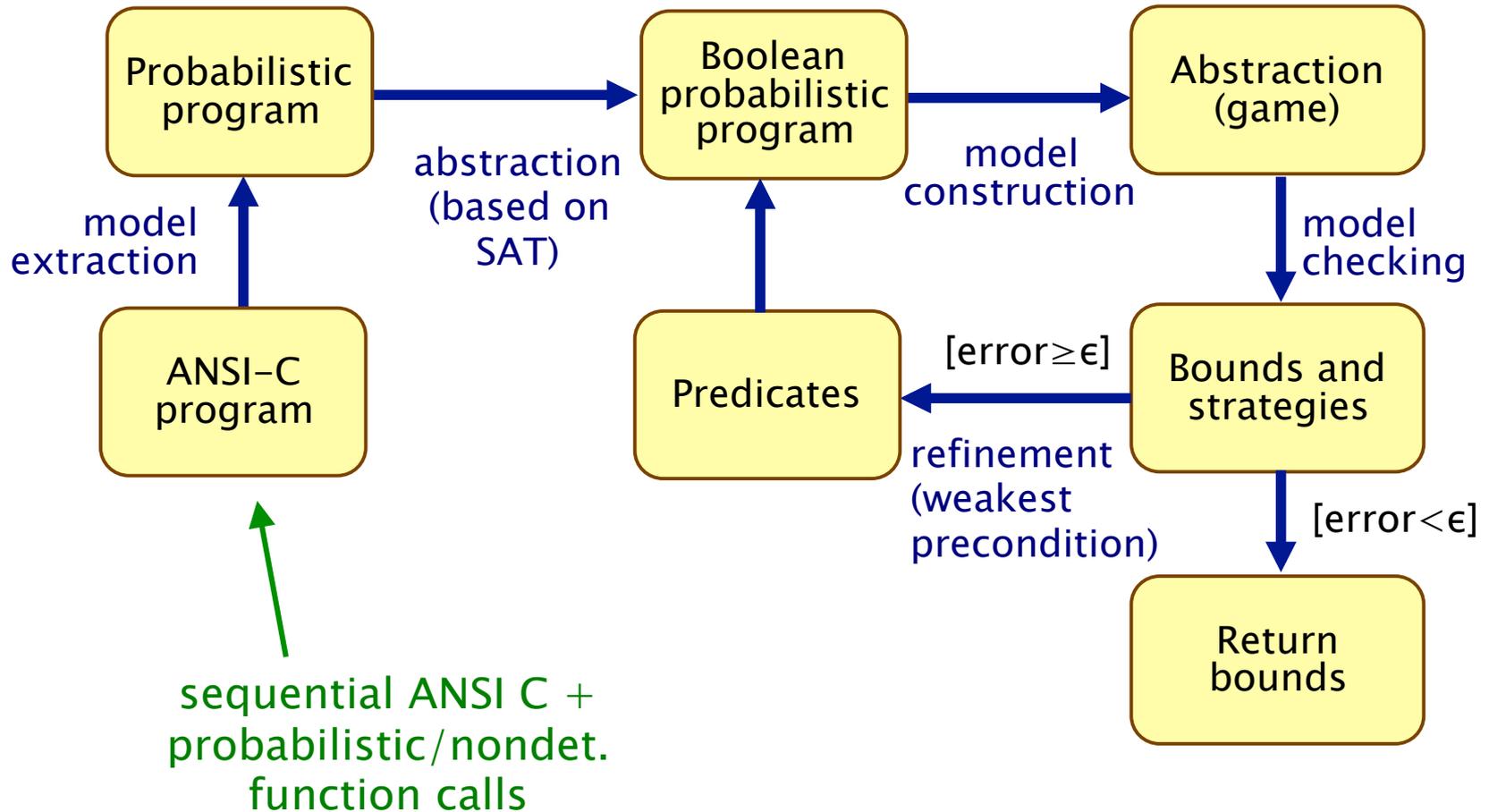
- Implementations of **quantitative abstraction refinement...**
- Verification of **probabilistic timed automata** [FORMATS'09]
  - zone-based abstraction/refinement using DBMs
  - implemented in (next release of) **PRISM**
  - outperforms existing PTA verification techniques
- Verification of **probabilistic software** [VMCAI'09]
  - predicate abstraction/refinement using SAT solvers
  - implemented in tool **qprover**: components of PRISM, SATABS
  - analysed real network utilities (ping, tftp) – approx 1KLOC
- Verification of **concurrent PRISM models** [Wachter/Zhang'10]
  - implemented in tool **PASS**; infinite-state PRISM models

# Verification of PTAs

- Probabilistic model checking of PTAs



# Verification of probabilistic software



# Overview

- Probabilistic model checking
  - Markov decision processes (MDPs)
  - probabilistic timed automata (PTAs)
- Abstraction for probabilistic models
  - abstractions of MDPs (stochastic two-player games)
- Quantitative abstraction refinement
  - abstraction-refinement loop
  - probabilistic model checking for PTAs
  - also: verification of probabilistic software
- **Conclusions & current/future work**

# Related work

- **Abstraction for Markov chains:**
  - DTMCs: probability intervals (MDPs) [Fecher/Leucker/Wolf] [Huth]
  - CTMCs: using CTMDPs [Katoen/Klink/Leucker/Wolf]
  - CTMCs: sliding window abstraction [Henzinger/Mateescu/Wolf]
  - and more...
- **Abstraction refinement for MDPs:**
  - RAPTURE [D'Argenio/Jeannet/Jensen/Larsen]
  - probabilistic CEGAR [Hermanns/Wachter/Zhang]
  - magnifying lens abstraction [de Alfaro/Roy]
  - MDP-based abstractions [Chadha/Viswanathan]
  - and more...

# Conclusions

- **Abstraction for probabilistic models**
  - MDPs (and PTAs) abstracted as stochastic two-player games
  - abstraction yields lower/upper bounds on probabilities
- **Quantitative abstraction refinement**
  - bounds give quantitative measure of utility of abstraction
  - bounds/strategies can be used to guide refinement
  - quantitative abstraction-refinement loop (for error  $< \epsilon$ )
  - fully automatic generation of abstraction
  - works in practice: probabilistic timed automata & software
- **Current & future work**
  - improved refinement heuristics, imprecise abstractions
  - software + time + probabilities
  - CTMCs, timed properties
  - probabilistic/stochastic hybrid systems