Abstraction in Quantitative Probabilistic Model Checking

Gethin Norman

University of Oxford

Two Decades of Probabilistic Verification (November 2007)

Abstraction – Motivation

- Even employing efficient model checking algorithms, state of the art data structures model checking is hard
- Even more important in the probabilistic setting
  - algorithms more complex
  - require numerical computation
- Required for model checking infinite state systems
- Abstraction is an approach to reduce the complexity of model checking
- A number of different approaches
  - abstract the model/property/satisfaction relation
  - automated/require user interaction

In this talk...

- Quantitative probabilistic verification
  - DTMCs, CTMCs and MDPs
- For simplicity consider reachability probabilities
  - basis of model checking algorithms for temporal logic
  - results extends to until and globally properties
- Approaches also extend to reward structures
  - expected reward cumulated before reaching a target set
  - expected reward at time t/cumulated by time t
  - probability reach a target set before the reward reaches...

Overview

- Notation
- Exact approaches
  - bisimulation minimisation
  - probabilistic timed automata
  - symmetry reduction/partial order reduction
- Approximate approaches
  - algorithm-based
  - model-based
    - models
    - model checking
    - refinement
    - implementations
- Conclusions

Notation

- DTMC = (S,**P**)
  - S set of states
  - **P** : $S \times S \rightarrow [0,1]$ such that $\Sigma_{s' \in S} P(s,s') = 1$ for all $s \in S$
- Probabilistic reachability
  - F set of target states
  - $p_{DTMC}(s,F)$ probability of reaching F
- MDP = (S,Steps)
  - **Steps** : $S \rightarrow \mathrm{dist}(S)$
  - **Steps**(s) set of distributions/choices available in s
- Minimum/Maximum probabilistic reachability
  - $p_{MDP}^{min}(s,F)$ minimum probability of reaching F
  - $p_{MDP}^{max}(s,F)$ maximum probability of reaching F

Notation

- CTMC = (S,**R**)
  - S set of states
  - **R** : S×S → ℝ rate matrix
- Time-bounded reachability probabilities
  - $p_{CTMC}(s,t,F)$ probability of reaching F by time t
  - $p_{CTMC}(s,t,F)$ probability of reaching F by time t
- In each case assume where necessary
  - initial state $\underline{s}$
  - set of atomic propositions AP
  - labelling function L : S → AP
    - L(s) is the set of atomic propositions that hold in state s

Overview

- Notation
- Exact approaches
  - bisimulation minimisation
  - probabilistic timed automata
  - symmetry reduction/partial order reduction
- Approximate approaches
  - algorithm-based
  - model-based
    - models
    - model checking
    - refinement
    - implementations
- Conclusions

(Exact) Abstraction

- Basic idea: construct a smaller "equivalent" model
  - preserves satisfaction of all/some temporal logic properties
  - e.g. yields same reachability probability
  - e.g. yields same transient/steady state probabilities
- State-level algorithms
  - work directly on the states, optimal reduction
  - e.g. bisimulation
- Model-level algorithms
  - based on higher-level description, non-optimal reduction
  - e.g. symmetry reduction (based on state representation)
- Automated techniques (no user interaction required)

Probabilistic bisimulation

- Equivalence: (strong) probabilistic bisimulation
  - also known as lumping
  - applicable to DTMCs, MDPs and CTMCs
  - preserves the satisfaction of PCTL, CSL, LTL, CTL*...
  - optimal for branching time logics
    - states equivalent if and only if they satisfy the same formulae
  - feasible algorithms for computing “smallest” bisimilar model

- Abstraction: the quotient model
  - abstract states are the equivalence classes of the relation

Probabilistic bisimulation – DTMCs

- Probabilistic bisimulation (DTMCs) [Larsen & Skou 91]
- The relation $R \subseteq S \times S$ is a strong bisimulation if for any $(s_1,s_2) \in R$:
  - $L(s_1) = L(s_2)$ (the same atomic propositions hold)
  - $\mathbf{P}(s_1,C) = \mathbf{P}(s_2,C)$ for all $C \in S/R$

    (S/R set of equivalence classes under R)

Probabilistic bisimulation – CTMCs

- Probabilistic bisimulation (CTMCs) [Buchholz 94]
- The relation $R \subseteq S \times S$ is a strong bisimulation if for any $(s_1,s_2) \in R$:
  - $L(s_1) = L(s_2)$ (the same atomic propositions hold)
  - $\mathbf{R}(s_1,C) = \mathbf{R}(s_2,C)$ for all $C \in S/R$

    (S/R set of equivalence classes under R)
- Also backwards probabilistic bisimulation
  - $\mathbf{R}(C,s_1) = \mathbf{R}(C,s_2)$ for all $C \in S/R$ (and $\mathbf{R}(s_1,S) = \mathbf{R}(s_2,S)$)
  - preserves CSL without nested probabilistic/steady state operators [Sproston & Donatelli 04]

Probabilistic bisimulation – MDPs

- Probabilistic bisimulation (MDPs) [Segala & Lynch 94]
- The relation $R \subseteq S \times S$ is a strong bisimulation if for any $(s_1,s_2) \in R$:
  - $L(s_1) = L(s_2)$ (the same atomic propositions hold)
  - for any $\mu_1 \in Steps(s_1)$ there exists $\mu_2 \in Steps(s_2)$ such that $\mu_1(C) = \mu_2(C)$ for all $C \in S/R$
  - for any $\mu_2 \in Steps(s_2)$ there exists $\mu_1 \in Steps(s_1)$ such that $\mu_2(C) = \mu_1(C)$ for all $C \in S/R$

Bisimulation minimisation – Algorithm

- Basic algorithm (partition refinement) is based on splitting
  - suppose $P=\{S_1,\ldots,S_n\}$ is some initial partition of S
  - a splitter for some block S_i is an element S_p of the partition such that $\mathbf{P}(s,S_p) \neq \mathbf{P}(s',S_p)$ for some $s,s' \in S_i$
    - the probability to enter S_p is not the same for each state of S_i
  - algorithm splits S_i into sub-blocks for which probabilities agree
    - i.e. $\mathbf{P}(s,S_p)$ is the same for all states s in the sub-block
  - repeat until there are no more splitters
- Returns the coarsest bisimulation
  - dependent on the initial partition
  - states not in same set of initial partition will not be equivalent

Bisimulation minimisation – Algorithm

- Complexity for DTMCs and CTMCs
  - as for non-probabilistic bisimulation
  - logarithmic in the number of states
  - linear in the number of transitions
- Complexity for MDPs
  - O(NM(log(N)+log(M))
  - N number of states and M number of transitions
- Optimisations
  - exploit compositionality – reduce sub-components separately e.g. [Hermanns & Katoen 00]
  - symbolic implementations, e.g. MTBDDs [Derisavi 07]
  - base initial partition on only atomic propositions of interest, use qualitative precomputation algorithms [Katoen et. al. 07]

Probabilistic bisimulation – Summary

- Been shown to be successful in practice
- Limitation: time to construct the bisimulation quotient
  - can exceed the model checking time for the concrete system
  - less true in the probabilistic setting (model checking is harder)
  - reduced if checking a number of properties
- Limitation: requires construction of the concrete system
  - compositional approach (perform abstraction of parallel components separately and then compose)
  - symbolic data-structures (allow representation of larger state spaces)
- Use coarser equivalence to improve reduction?
  - e.g. for LTL use trace distribution equivalence – no feasible algorithms

Weak probabilistic bisimulation

- Equivalent up to "internal" computation (τ actions)
  - for example updating/modifying views in the Gossip protocol
  - preservation of temporal logics without next operator
    - "stuttering equivalent"
  - coarser than probabilistic bisimulation
  - minimisation algorithm more complex
    - requires computation of reachability probabilities

- Complexity
  - DTMCs: cubic in the number of states [Baier & Hermanns 97]
  - MDPs: exponential in the number of states [Cattani & Segala 04]

Probabilistic timed automata

- Semantics inherently infinite state (real-time)
  - several verification approaches [Kwiatkowska .et al. 99-07]
- Region graph
  - preserves PTCTL but prohibitively large for even small examples
- Digital clocks
  - restricted to probabilistic/expected reachability
  - efficient (employ finite state model checking techniques)
- Zones
  - forwards: bounds on reachability probabilities
  - backwards: PTCTL
  - yields small models but complex operations
  - requires construction of MDP for each quantitative check

Symmetry reduction

- Exploits presence of replication within a model
  - requires models to have a certain structure
  - model level bisimulation
  - cheaper than (state level) bisimulation reduction
  - not necessarily optimal quotient
- Two approaches developed for PRISM
  - both based on component symmetry
  - symbolic [Kwiatkowska et. al. CAV 06]
    - reduction performed on the MTBDD representing the system
  - language level – GRIP tool [Donaldson & Miller ATVA 06]
    - reduction performed on the PRISM language syntax

Component symmetry

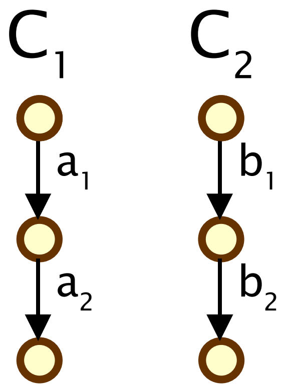
- System of N symmetric components
  - exchanging a pair of components has no effect on behaviour
  - system states $(s_1,s_2,\ldots,s_n)$ where s_i local state of component i
- Reduction gives (up to) factorially smaller quotient model
  - for example 4 components each with local states {A,B,C}
  - (A,A,C,B) = (A,A,B,C) = (C,A,B,A) = ···
- Require atomic propositions also "symmetric"
  - allowed: "some/all/K components have received a request"
  - not allowed: "component i has received a request"
- Essentially corresponds to counting number of components in the different possible local state
  - e.g. "population model" used in systems biology

Symmetry Reduction – Summary

- Successful in practice
- Two approach complementary
  - MTBDD level appropriate for models with small number of complex components
  - syntax level appropriate for models with large number of simple components
- Many other forms of symmetry
  - e.g. rotational symmetry for ring networks

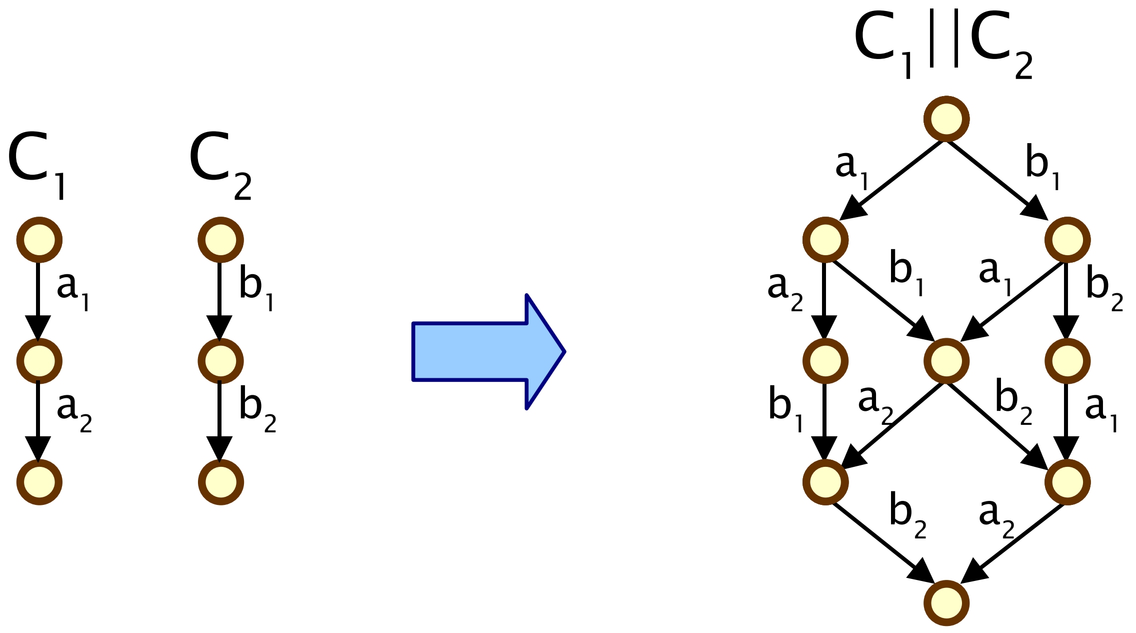
Partial order reduction

- State space explosion used by the interleaving of parallel components



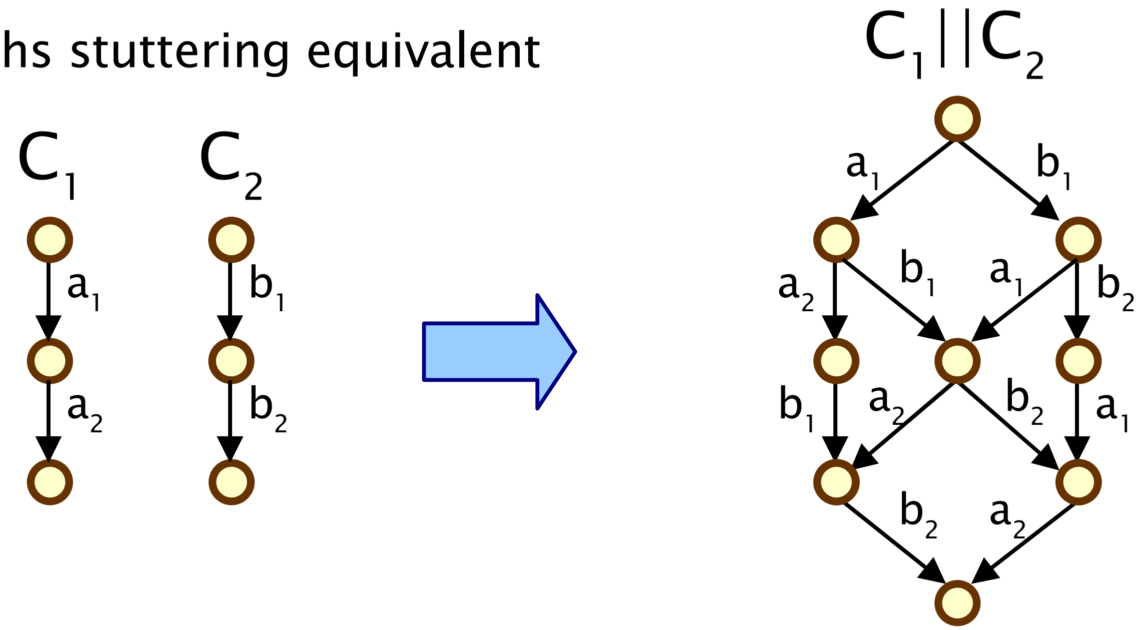
Partial order reduction

- State space explosion used by the interleaving of parallel components



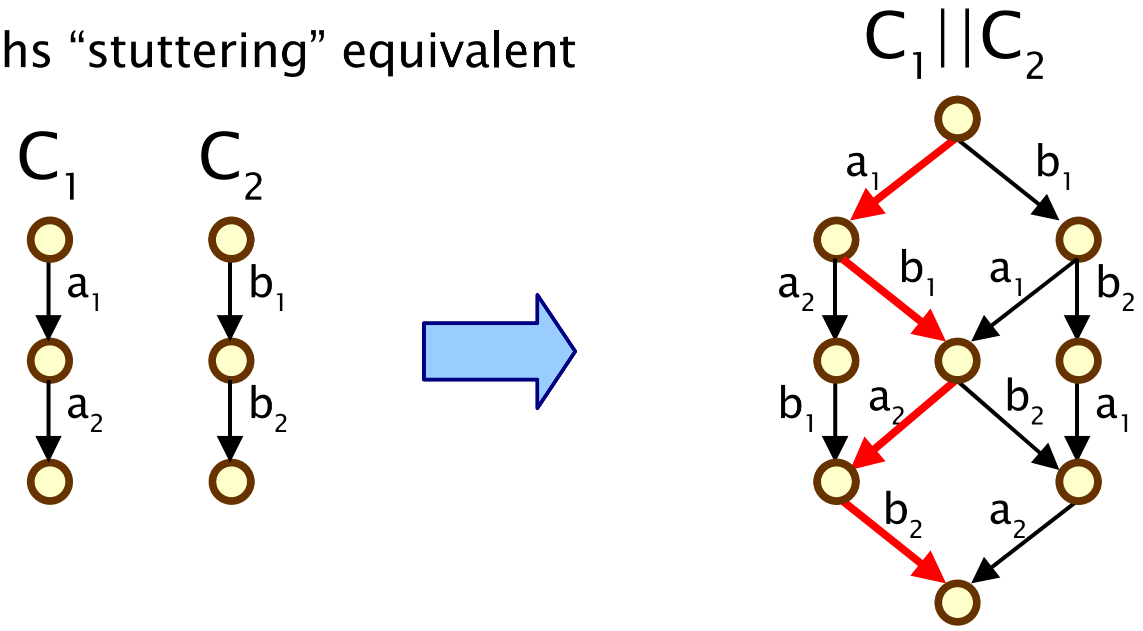
Partial order reduction

- State space explosion used by the interleaving of parallel components
  - paths stuttering equivalent



Partial order reduction

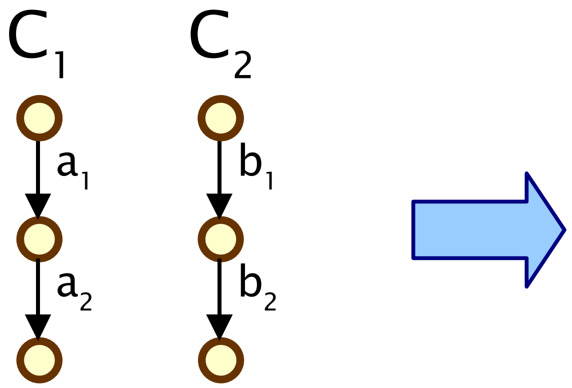
- State space explosion used by the interleaving of parallel components
  - paths "stuttering" equivalent

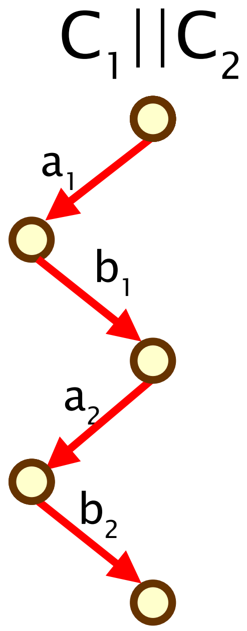


- Partial order reduction – include only one representative

Partial order reduction

- State space explosion used by the interleaving of parallel components
  - all paths stuttering equivalent





- Partial order reduction – include only one representative
  - reduction in state space

Partial order reduction – Probabilistic

- Probabilistic extension for MDP models
  - POR based on interleaving (asynchronous composition) of subcomponents
    - i.e. nondeterministic choice as to which component moves
- Extensions of Peled's ample set method for MDPs
  - linear time [Baier et. al. 04] [D'Argenio & Niebert 04]
  - branching time [Baier et. al. 05]
  - preservation of temporal logical properties without next
- Implemented in the tool LiQuor
- Many different non-probabilistic approaches to investigate/extend
  - e.g. stubborn sets, persistent sets

Exact techniques – Summary

- These techniques have been shown to be very successful in practice, however may still not yield a sufficient gain
  - reductions do not exploit states with “similar” behaviour
  - all states considered equally (do not ignore state which can be reached with a very small probability)
  - reductions may not exploit the single/small set of properties of interest (bisimulation preserves all of PCTL/CSL)
    - e.g. bisimulation minimisation algorithm will preserve all formulae for atomic propositions encoded in the initial partition

- Alternative is to employ approximate abstractions...

Overview

- Notation
- Exact approaches
  - bisimulation minimisation
  - probabilistic timed automata
  - symmetry reduction/partial order reduction
- Approximate approaches
  - algorithm-based
  - model-based
    - models
    - model checking
    - refinement
    - implementations
- Conclusions

Approximate model checking

- Use an approximate model checking algorithm
- Number of different approaches
  - magnifying lens abstraction
  - approximate LTL model checking for MDPs
  - also relevant: sampling based approaches
    - APMC, YMER and VESTA
  - approaches from performance

Magnifying lens abstraction

- Magnifying Lens Abstraction (MLA) [de Alfaro & Roy 07]
  - model checking algorithm for approximating minimum and maximum reachability probabilities of MDPs
  - returns upper and lower bounds on property of interest
  - i.e minimum/maximum probability within the interval $[p_1,p_2]$

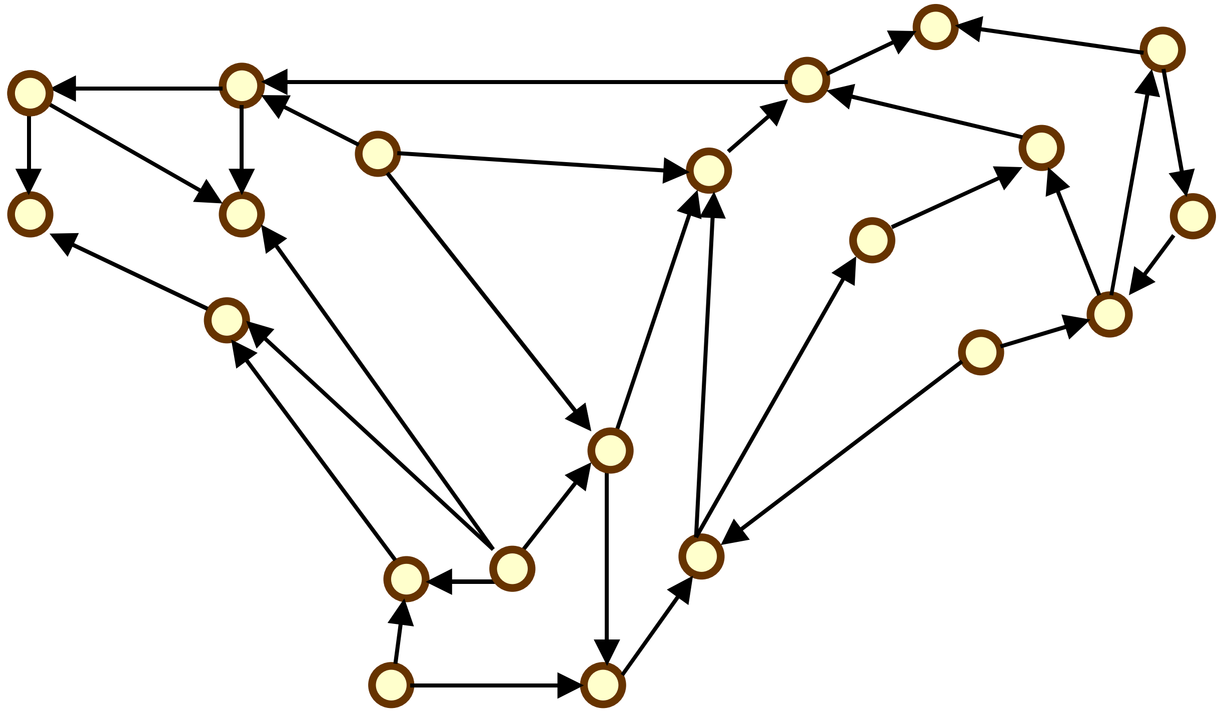
- Magnification:
  - partition state space into regions and analyse region separately
  - analysis examines individual states in “magnified” region
  - (“semi-abstract” since involves analysing concrete states)

Magnifying lens abstraction (MLA)

- Based on the fact the major problem in probabilistic model checking is storing the vector of probabilities for states
  - efficient methods for storing very large transition systems
- Method includes refinement
  - can return interval up to any prescribed degree of accuracy
  - if returned intervals are too large then split regions and compute new intervals
- Approach is based on clustering states based on value
  - different from model based abstraction approaches which are based on transition structure

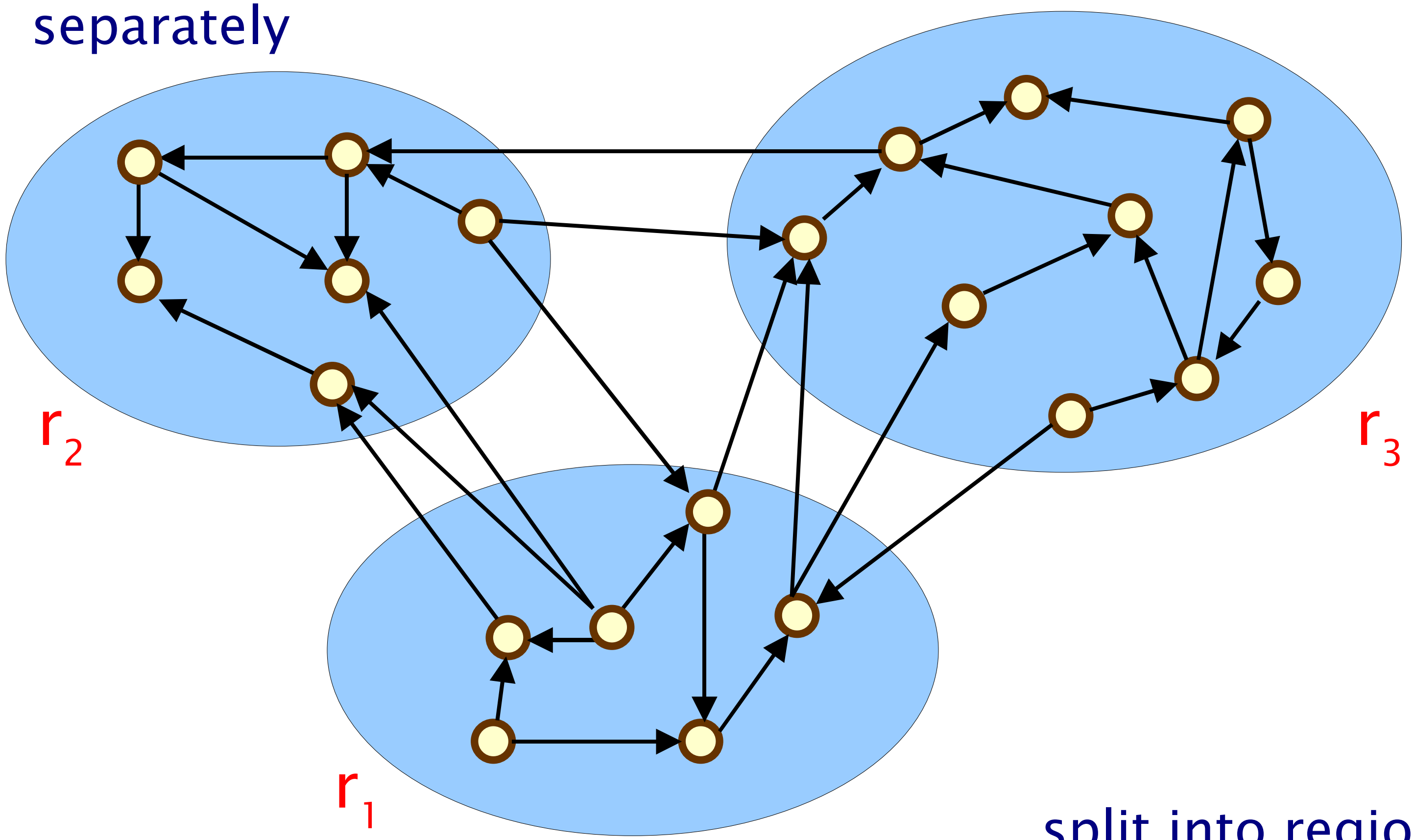
MLA – Example

- Basic idea is to split into individual regions and analyse separately



MLA – Example

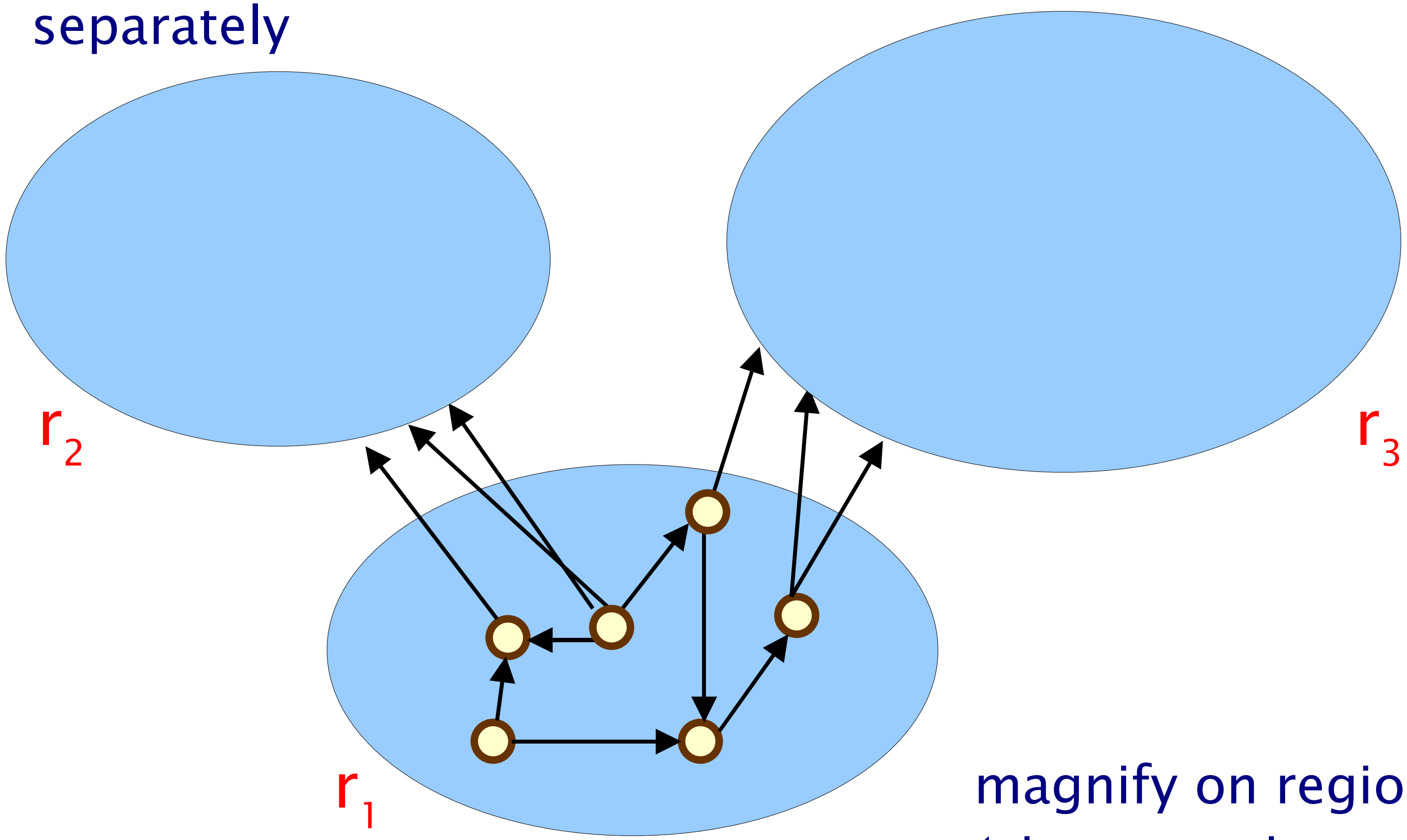
- Basic idea is to split into individual regions and analyse separately



split into regions

MLA – Example

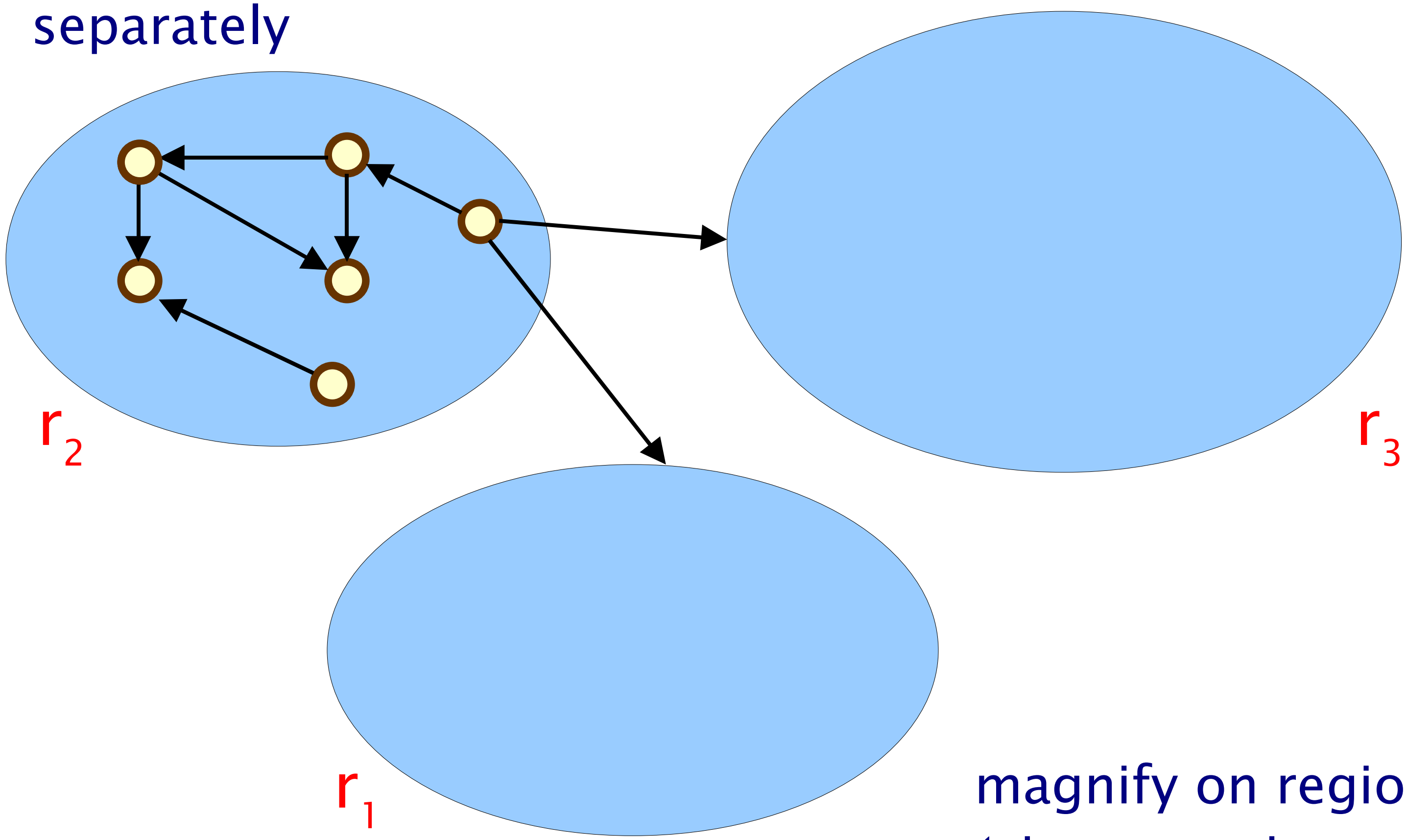
- Basic idea is to split into individual regions and analyse separately



magnify on region r_1
(abstract other regions)

MLA – Example

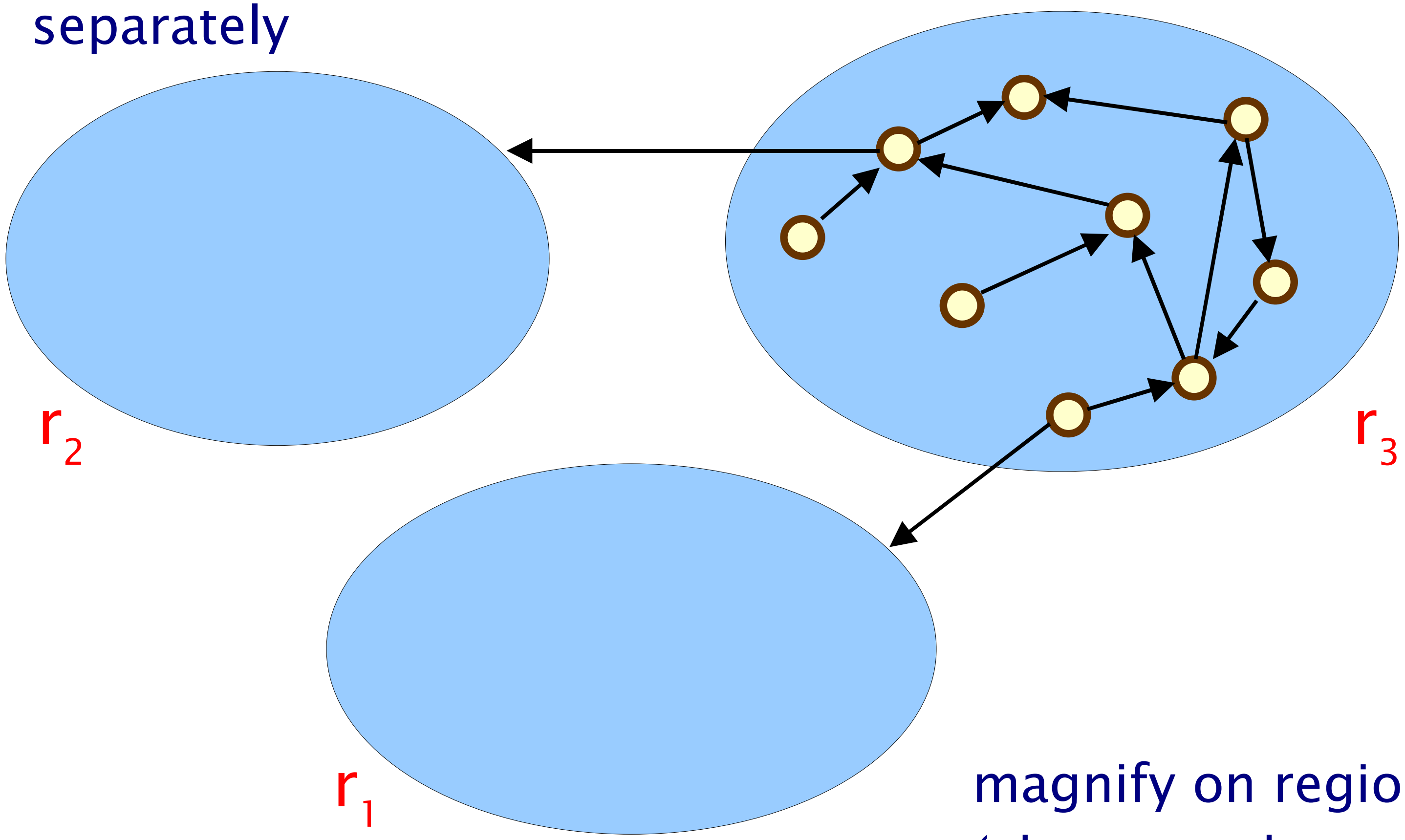
- Basic idea is to split into individual regions and analyse separately



magnify on region r_2
(abstract other regions)

MLA – Example

- Basic idea is to split into individual regions and analyse separately



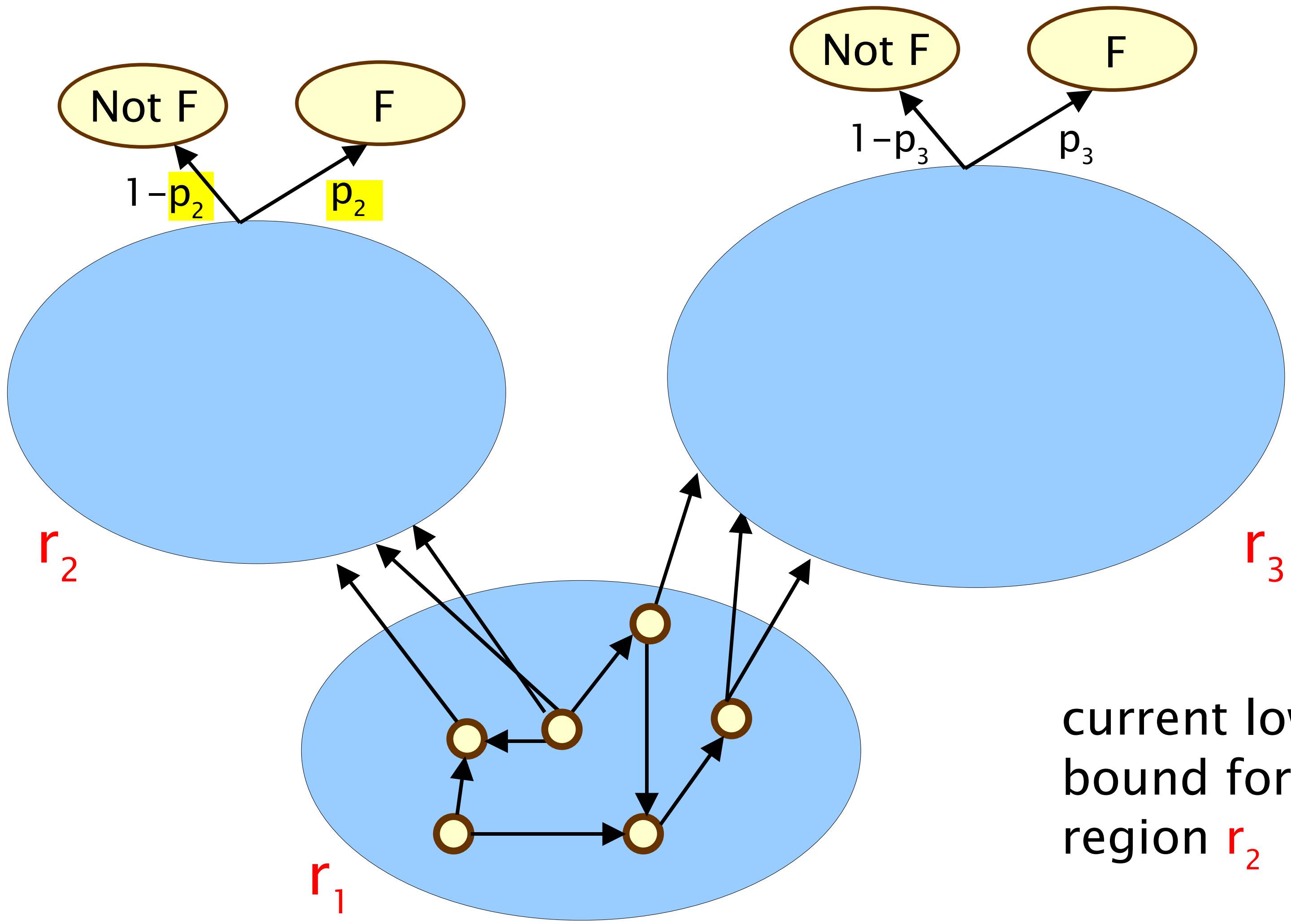
magnify on region r_3
(abstract other regions)

MLA – Algorithm

- Suppose interested in minimum probability of reaching F and partitioned state space into regions $r_1,\ldots,r_n$
- Pseudo-code for computing lower bound:

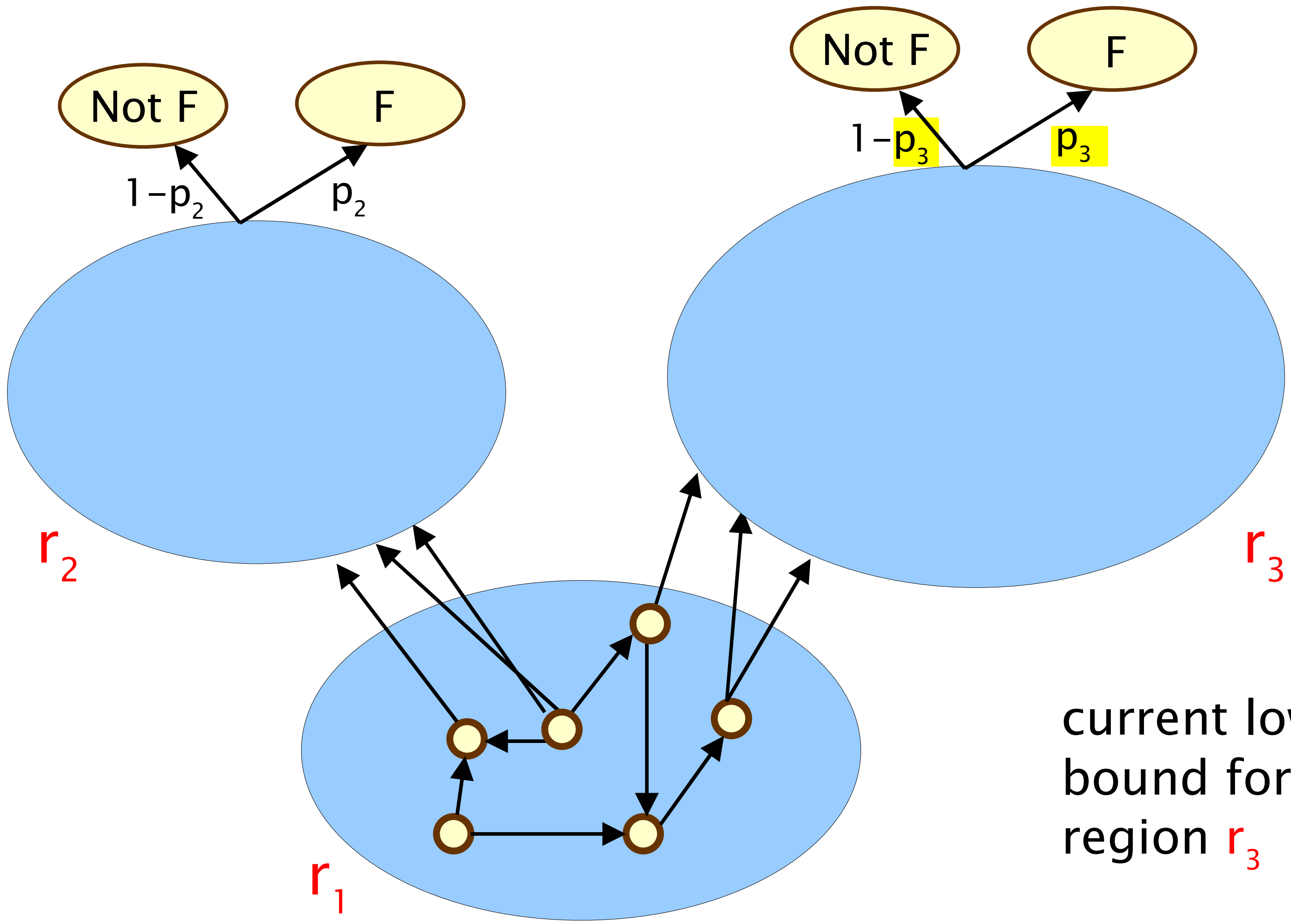
```
for $i=1..n$
  – magnify on region $r_i$
  – abstract each region $r_j$ ($j \neq i$) to single state for which
    probability of reaching F equals lower bound for region $r_j$
```

MLA – Algorithm



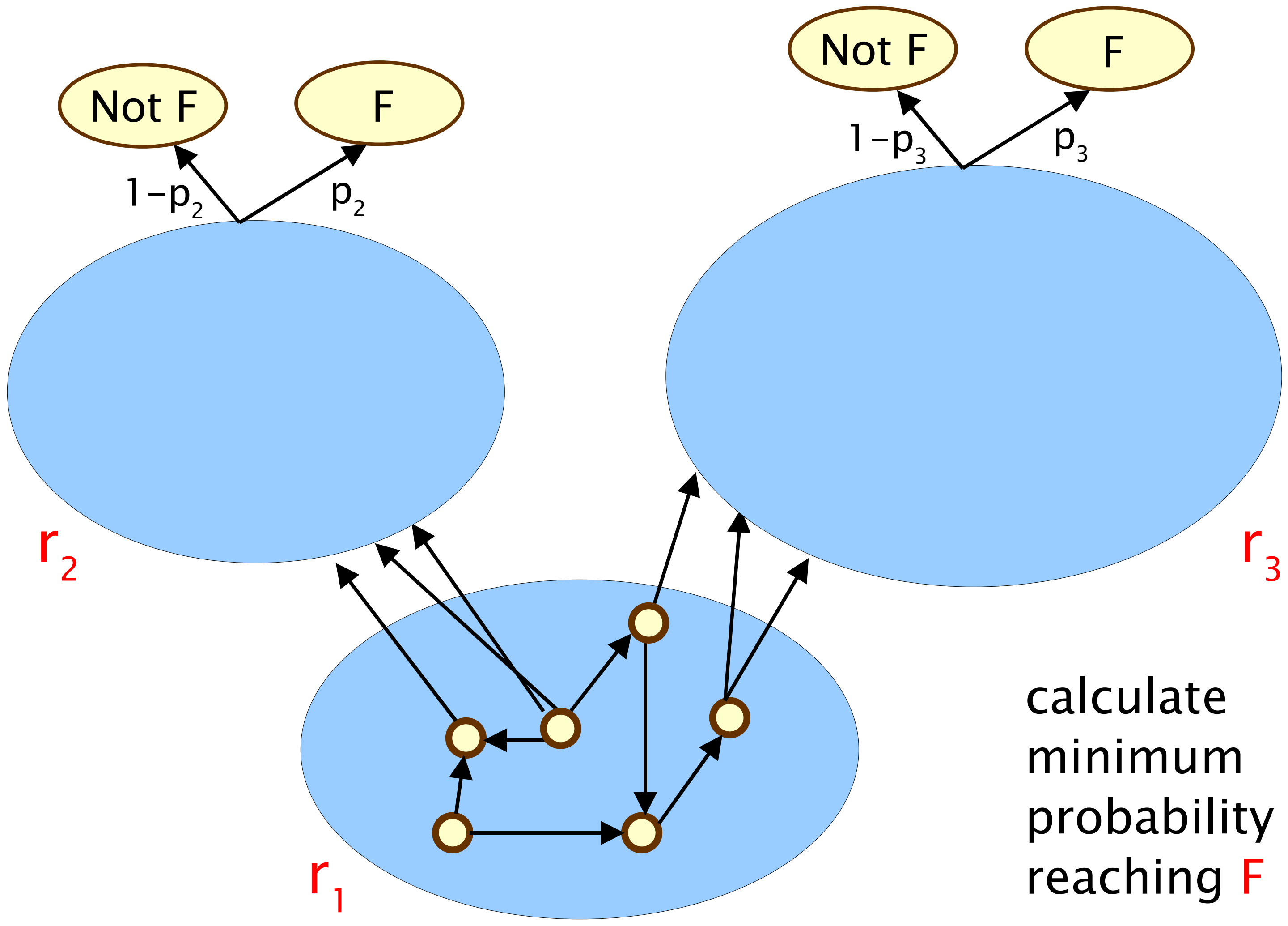
current lower bound for region r_2

MLA – Algorithm



current lower bound for region r_3

MLA – Algorithm



calculate minimum probability of reaching F

MLA – Algorithm

- Suppose interested in minimum probability of reaching F and partitioned state space into regions $r_1,\ldots,r_n$
- Pseudo-code for computing lower bound:

```
for $i=1..n$
  – magnify on region $r_i$
  – abstract each region $r_j$ ($j \neq i$) to single state for which
    probability of reaching F equals lower bound for region $r_j$
  – compute for all state in $r_i$ minimum probability of reaching F
```

MLA – Algorithm

- Suppose interested in minimum probability of reaching F and partitioned state space into regions $r_1,\ldots,r_n$
- Pseudo-code for computing lower bound:

```
for $i=1..n$
  – magnify on region $r_i$
  – abstract each region $r_j$ ($j \neq i$) to single state for which
    probability of reaching F equals lower bound for region $r_j$
  – compute for all state in $r_i$ minimum probability of reaching F
  – take minimum over all states in $r_i$ as new lower bound for $r_i$
```

MLA – Algorithm

- Suppose interested in minimum probability of reaching F and partitioned state space into regions $r_1,\ldots,r_n$
- Pseudo-code for computing lower bound:

```
for $i=1..n$
  – magnify on region $r_i$
  – abstract each region $r_j$ ($j \neq i$) to single state for which
    probability of reaching F equals lower bound for region $r_j$
  – compute for all state in $r_i$ minimum probability of reaching F
  – take minimum over all states in $r_i$ as new lower bound for $r_i$
repeat until bounds do not change
```

MLA – Algorithm

- Suppose interested in minimum probability of reaching F and partitioned state space into regions $r_1,\ldots,r_n$
- Pseudo-code for computing lower bound:

```
for $i=1..n$
  – magnify on region $r_i$
  – abstract each region $r_j$ ($j \neq i$) to single state for which
    probability of reaching F equals lower bound for region $r_j$
  – compute for all state in $r_i$ minimum probability of reaching F
  – take minimum over all states in $r_i$ as new lower bound for $r_i$
repeat until bounds do not change
```

MLA – Algorithm

- Suppose interested in minimum probability of reaching F and partitioned state space into regions $r_1,\ldots,r_n$
- Pseudo-code for computing upper bound:

```
**for** $i=1..n$
    – magnify on region $r_i$
    – abstract each region $r_j$ ($j \neq i$) to single state for which
      probability of reaching F equals upper bound for region $r_j$
    – compute for all state in $r_i$ minimum probability of reaching F
    – take maximum over all states in $r_i$ as new lower bound for $r_i$
**repeat** until bounds do not change
```

MLA – Refinement

- Refinement
  - divide any region from which upper and lower bound differ by more than some prescribed error
    - do not divide all regions
  - attempted more complete refinement schemes but during experiments this simple approach worked best
- How to divide the region?
  - based on the state variables of the concrete system
  - suppose the variables are ordered
  - first split based on first variable in the order, then second, ...
  - dependent on how the user defines the model

MLA – Complexity

- Approach has limited space complexity since during computation need to store
  - upper and lower bounds for all regions
  - values for all concrete states in current magnified region
  - space requirement $2 \cdot |R| + \max_{r \in R} |r|$
  - $O(\sqrt{|S|})$ since $\max_{r \in R} |r| \geq |S|/|R|$
- Not applicable to infinite/very large systems
- There is a trade off employing this approach:
  - small number of regions: many states in each region
  - large number of regions: storage of lower and upper bounds

MLA – Summary

- Limitation in space gains
- Appears to work well in limited experiments
- Potentially appropriate for models not amenable to other (model based) abstraction approaches

- Future work
  - extensions, e.g. develop refinement schemes...
  - combine with other approaches?

Approximate LTL semantics for MDPs

- LTL model checking of MDPs is hard
  - doubly exponential in the formula
- PCTL model checking of MDPs is (relatively) easy
  - linear in the formula
- PCTL requires probabilities for "simple" path formulae only
  - reduces to reachability analysis
  - e.g. do not compute probability of $(\phi \; U \; \psi) \wedge (\phi' \; U \; \psi')$
- Approximate conjunction (and disjunction) [Baier et. al. 99]

Sampling based – Monte Carlo

- Uses discrete event simulation and Monte Carlo methods
  - estimates reachability probabilities for DTMCs and CTMCs
  - generates random paths from high-level model
  - number of samples dependent on approximation parameter ϵ and confidence parameter δ such that

$$\mathbf{Prob}(\ |\ ans - p_{DTMC}(s,F)\ |\ \leq \epsilon\ ) \geq 1-\delta$$

  - probability estimation within ϵ of answer is at least $1-\delta$
  - number of samples $\mathbf{O}(1/\epsilon, \log(1/\delta))$
- Only correct for bounded properties
  - generated path must have a finite depth
- Introduced in APMC [Herault. et al. VMCAI 04]
  - also implemented in PRISM

Sampling based – Hypothesis testing

- Based on hypothesis testing [Younes & Simmons CAV 02]
  - checking time bounded until CSL formula for CTMCs
  - requires a probability bound (does not compute an approximate probability instead tests the hypothesis: the probability is above/below a bound)
  - combined with PRISM to verify general CSL formulae
  - extends to general distributions (no increase in complexity)
  - using this approach can quickly learn the result with some error

- Tool support: YMER [Younes & Simmons CAV 02]
  - (formerly called ProVer)
- Similar approach: VESTA [Sen et. al. CAV 04]

Sampling based – Summary

- Two approaches
  - hypothesis testing more efficient than Monte Carlo
  - but require a probability bound (cannot return "probability is approximately..." only "yes" or "no")
  - both can handle infinite state models (samples constructed from high level language description)
  - both amenable to distributed implementations
  - Returns result for a single state
- Statistical approaches for MDPs?
  - non-determinism means techniques no longer applicable
  - not one probability space
  - compute "average"?
    - i.e. adversary that makes choices uniformly at random

Overview

- Notation
- Exact approaches
  - bisimulation minimisation
  - probabilistic timed automata
  - symmetry reduction/partial order reduction
- Approximate approaches
  - algorithm-based
  - model-based
    - models
    - model checking
    - refinement
    - implementations
- Conclusions

Model-based abstraction

- Number of approaches based on the non-probabilistic technique of existential abstraction [Clarke et. al. 91]
  - restricted to CTL* without "E" (∃) operator
- Constructs a "conservative" abstraction
  - if a property holds in the abstract model, then it also holds in the concrete system
  - if the property does not holds in the abstract model, then may or may not be false in the concrete system

Existential abstraction

- Technique based on a partition of the concrete state
  - each element of the partition is an abstract state

- Suppose we are given a concrete system $LTS = (S,\mathbf{T})$
  - S set of states
  - $\mathbf{T} \subseteq S \times S$ transition relation
- and partition of the state space $P = \{S_1,S_2,\ldots,S_n\}$

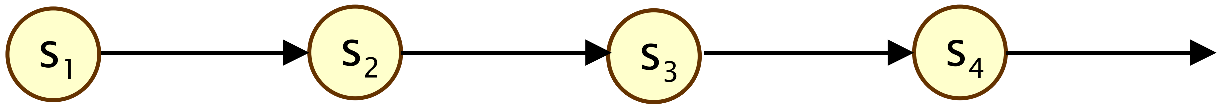
- Abstract transition system $LTS_A = (A,\mathbf{T}_A)$
  - $A = \{S_1,S_2,\ldots,S_n\}$
  - $(a,a') \in \mathbf{T}_A$ if and only if $(s,s') \in \mathbf{T}$ for some $s \in a$ and $s' \in a'$

Existential abstraction – Simulation

- $R \subseteq S \times A$ is a simulation relation $(s,a) \in R$
  - $L(s) = L(a)$ (states satisfy same atomic propositions)
  - for any $(s,s') \in T$ there exists $(a,a') \in T_A$ such that $(s',a') \in R$
- A concrete state s is simulated by the abstract state containing s
  - anything the concrete system can do the abstract model can simulate (but abstraction may do more)

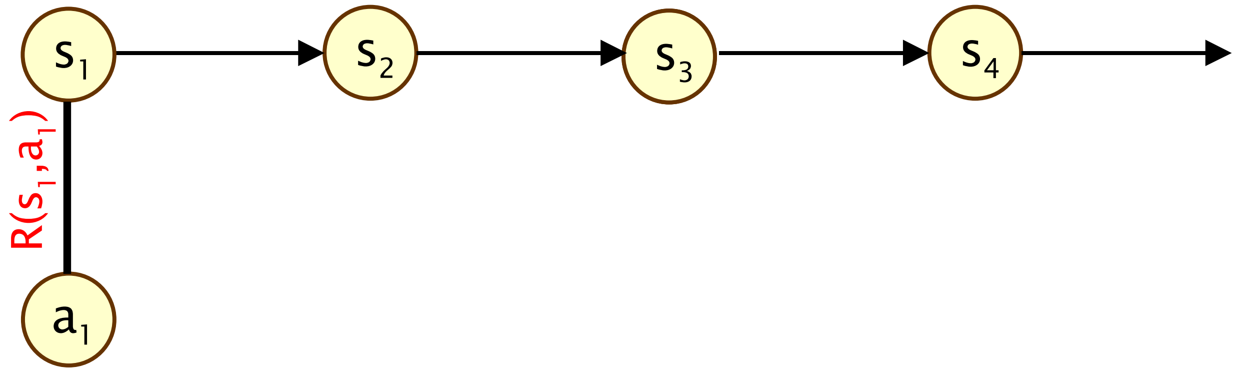
Existential abstraction – Simulation

- $R \subseteq S \times A$ is a simulation relation $(s,a) \in R$
  - $L(s) = L(a)$ (states satisfy same atomic propositions)
  - for any $(s,s') \in T$ there exists $(a,a') \in T_A$ such that $(s',a') \in R$
- A concrete state s is simulated by the abstract state containing s
  - anything the concrete system can do the abstract model can simulate (but abstraction may do more)
- Consider any concrete path



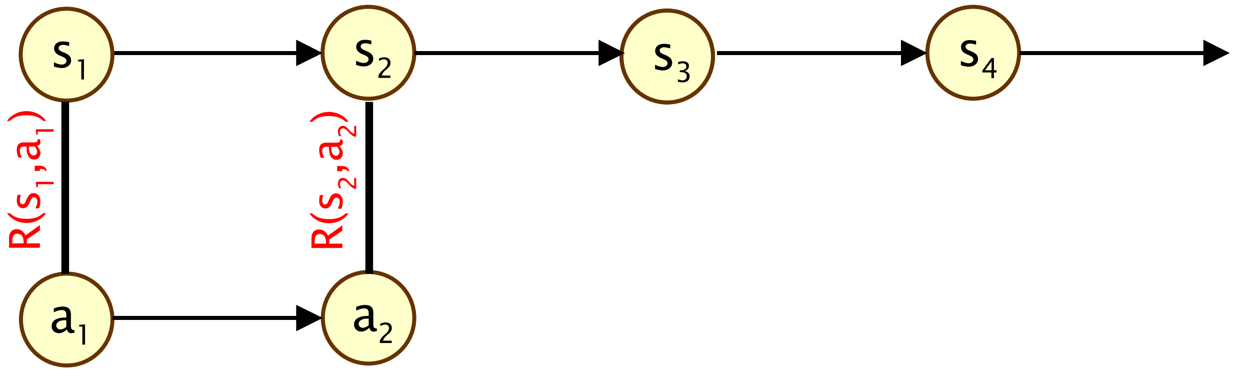
Existential abstraction – Simulation

- $R \subseteq S \times A$ is a simulation relation $(s,a) \in R$
  - $L(s) = L(a)$ (states satisfy same atomic propositions)
  - for any $(s,s') \in T$ there exists $(a,a') \in T_A$ such that $(s',a') \in R$
- A concrete state s is simulated by the abstract state containing s
  - anything the concrete system can do the abstract model can simulate (but abstraction may do more)
- Consider any concrete path



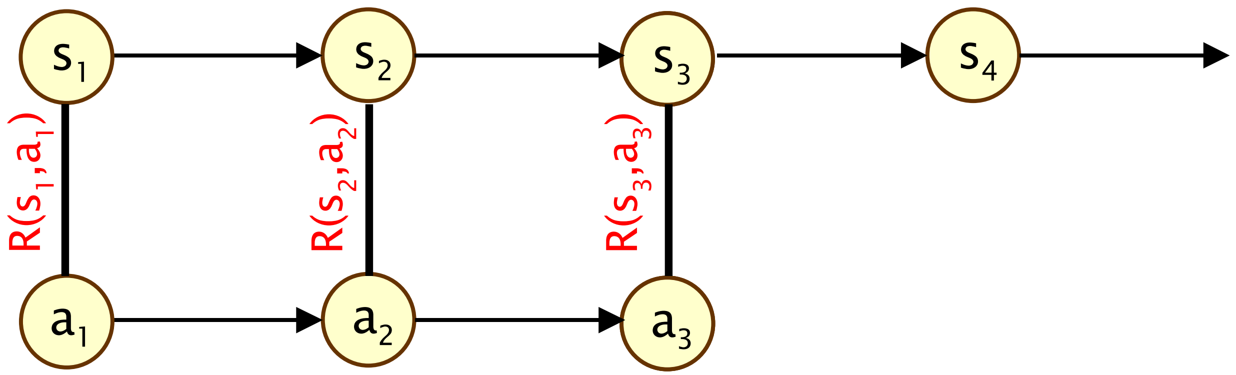
Existential abstraction – Simulation

- $R \subseteq S \times A$ is a simulation relation $(s,a) \in R$
  - $L(s) = L(a)$ (states satisfy same atomic propositions)
  - for any $(s,s') \in T$ there exists $(a,a') \in T_A$ such that $(s',a') \in R$
- A concrete state s is simulated by the abstract state containing s
  - anything the concrete system can do the abstract model can simulate (but abstraction may do more)
- Consider any concrete path


$$\begin{tikzcd}
s_1 \arrow[r] \arrow[d, dash, "R(s_1{,}a_1)"'] & s_2 \arrow[r] \arrow[d, dash, "R(s_2{,}a_2)"'] & s_3 \arrow[r] & s_4 \arrow[r] & {} \\
a_1 \arrow[r] & a_2
\end{tikzcd}$$

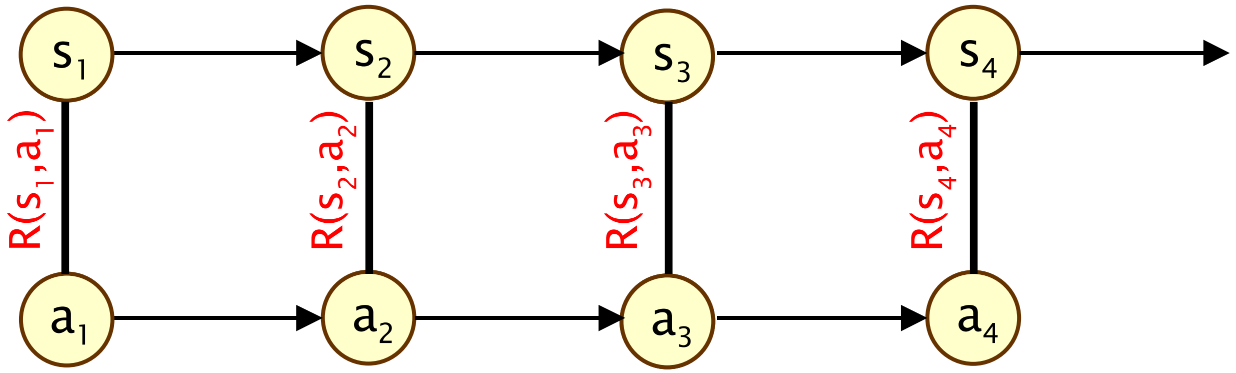
Existential abstraction – Simulation

- $R \subseteq S \times A$ is a simulation relation $(s,a) \in R$
  - $L(s) = L(a)$ (states satisfy same atomic propositions)
  - for any $(s,s') \in T$ there exists $(a,a') \in T_A$ such that $(s',a') \in R$
- A concrete state s is simulated by the abstract state containing s
  - anything the concrete system can do the abstract model can simulate (but abstraction may do more)
- Consider any concrete path



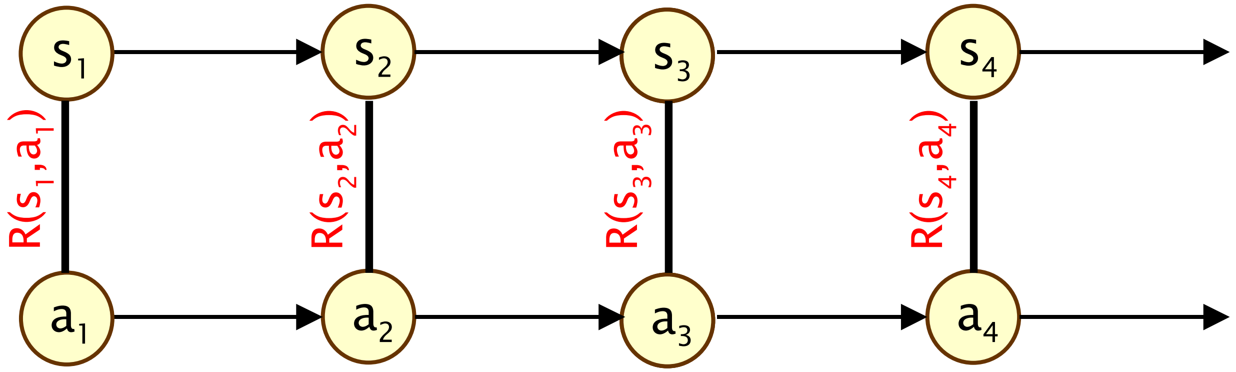
Existential abstraction – Simulation

- $R \subseteq S \times A$ is a simulation relation $(s,a) \in R$
  - $L(s) = L(a)$ (states satisfy same atomic propositions)
  - for any $(s,s') \in T$ there exists $(a,a') \in T_A$ such that $(s',a') \in R$
- A concrete state s is simulated by the abstract state containing s
  - anything the concrete system can do the abstract model can simulate (but abstraction may do more)
- Consider any concrete path



Existential abstraction – Simulation

- $R \subseteq S \times A$ is a simulation relation $(s,a) \in R$
  - $L(s) = L(a)$ (states satisfy same atomic propositions)
  - for any $(s,s') \in T$ there exists $(a,a') \in T_A$ such that $(s',a') \in R$
- A concrete state s is simulated by the abstract state containing s
  - anything the concrete system can do the abstract model can simulate (but abstraction may do more)
- Consider any concrete path



Existential abstraction – Probabilistic

- Existential abstraction in the probabilistic setting
  - can use probabilistic simulation [Segala & Lynch 94]
- What is the probabilistic abstraction (abstract system)?
  - MDPs, abstract Markov chains or two player stochastic games
- What is the model checking approach?
  - three valued logic ("true", "false", "do not know")
  - "probability bounded by p" or "probability in the interval $[p_1,p_2]$"
- How to refine when answers inconclusive?
  - what happens when we get "do not know", probability greater than/less than 0/1 or probability within the interval [0,1]
- How to implement?
  - predicate abstraction

Existential abstraction – Probabilistic

- Existential abstraction in the probabilistic setting
  - can use probabilistic simulation [Segala & Lynch 94]
- What is the probabilistic abstraction (abstract system)?
  - MDPs, Abstract Markov chains or two player stochastic games
- What is the model checking approach?
  - three valued logic (“true”, “false”, “do not know”)
  - “probability bounded by p” or “probability in the interval $[p_1,p_2]$”
- How to refine when answers inconclusive?
  - what happens when we get “do not know”, probability greater than/less than 0/1 or probability within the interval [0,1]
- How to implement?
  - predicate abstraction

Rapture

- Extension of existential abstraction [D'Argenio et. al. 02]
  - Reachability analysis of probabilistic transition systems based on reduction strategies
- Both concrete and abstract model are MDPs
  - abstraction introduces more nondeterminism
- $MDP = (S, Steps)$ and partition $P = \{S_1, S_2, \ldots, S_n\}$
- Quotient model $MDP_P = (A, Steps_A)$
  - $A = \{S_1, S_2, \ldots, S_n\}$ (abstract states are elements of the partition)
  - $\mu_A \in Steps_A(a)$ if and only if there exists $\mu \in Steps(s)$ such that $s \in a$ and $\mu_A(a') = \Sigma \{ \mu(s') \mid s' \in a' \}$ for all $a' \in A$
- Abstract MDP (probabilistically) simulates the concrete MDP
  - extension of non-probabilistic existential abstraction

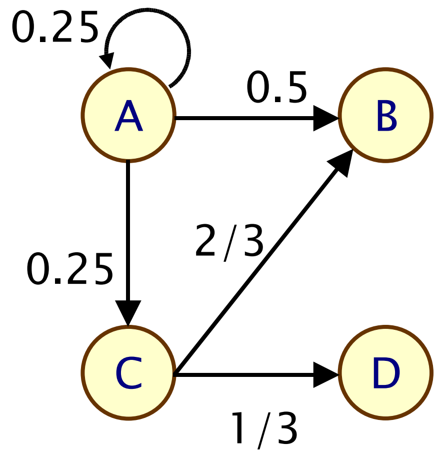
Rapture – Example

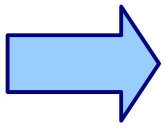
- Partition { {A,C} , {B,D} }

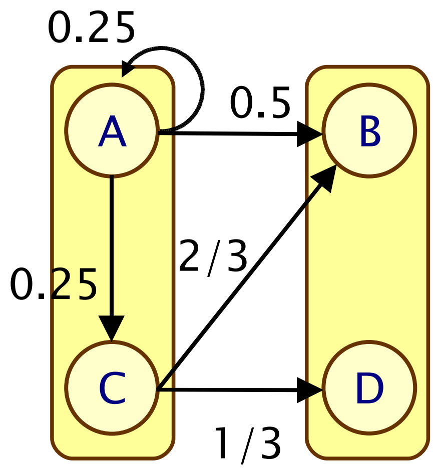
$$\begin{tikzcd}
A \arrow[loop above, "0.25"] \arrow[r, "0.5"] \arrow[d, "0.25"'] & B \\
C \arrow[ur, "2/3"] \arrow[r, "1/3"'] & D
\end{tikzcd}$$

Rapture – Example

- Partition { {A,C} , {B,D} }

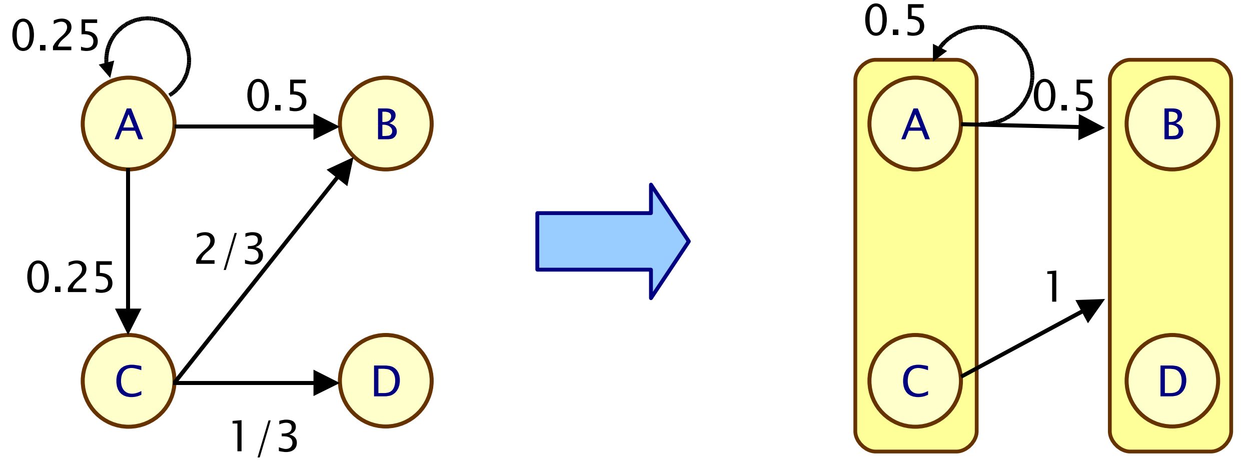






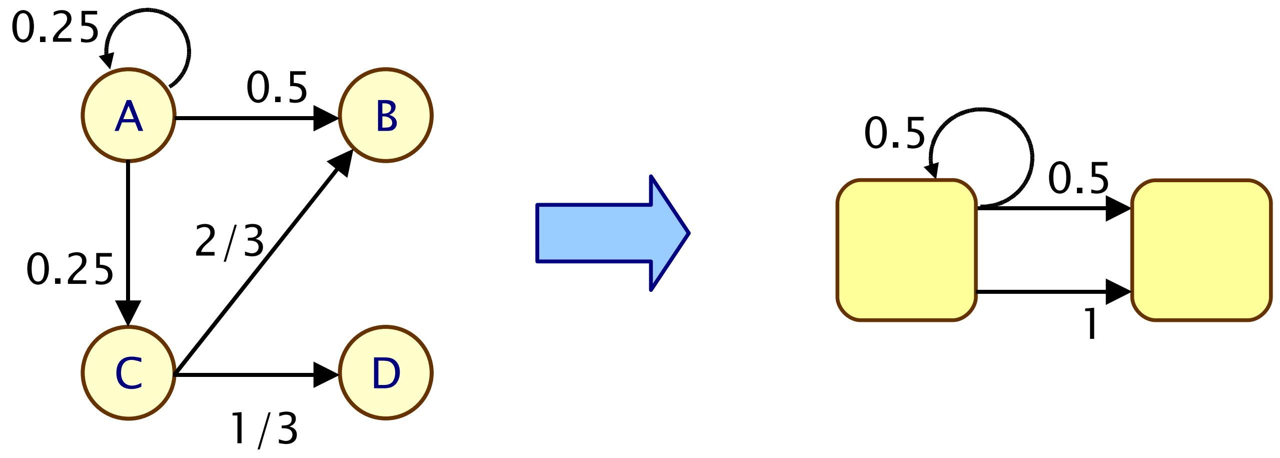
Rapture – Example

- Partition { {A,C} , {B,D} }



Rapture – Example

- Partition { {A,C} , {B,D} }



Rapture – Abstraction

- Concrete model $MDP=(S,Steps)$, partition P & target states F
- Abstract (quotient) model $MDP_P=(A, Steps_A)$
  - for any state $s \in S$, if $s \in a$ then:

$$p_{MDP/P}^{min}(a,F) \leq p_{MDP}^{min}(s,F)$$

$$p_{MDP}^{max}(s,F) \leq p_{MDP/P}^{max}(a,F)$$

Rapture – Abstraction

- Concrete model $MDP=(S,Steps)$, partition P & target states F
- Abstract (quotient) model $MDP_P=(A, Steps_A)$
  - for any state $s \in S$, if $s \in a$ then:

$$p_{MDP/P}^{min}(a,F) \leq p_{MDP}^{min}(s,F)$$

$$p_{MDP}^{max}(s,F) \leq p_{MDP/P}^{max}(a,F)$$

abstract minimum probabilities give lower bounds on minimum reachability probabilities

Rapture – Abstraction

- Concrete model $MDP=(S,Steps)$, partition P & target states F
- Abstract (quotient) model $MDP_P=(A, Steps_A)$
  - for any state $s \in S$, if $s \in a$ then:

$$p_{MDP/P}^{min}(a,F) \leq p_{MDP}^{min}(s,F)$$

$$p_{MDP}^{max}(s,F) \leq p_{MDP/P}^{max}(a,F)$$

  - abstract maximum probabilities give upper bounds on maximum reachability probabilities

Rapture – Abstraction

- Concrete model $MDP=(S,Steps)$, partition P & target states F
- Abstract (quotient) model $MDP_P=(A, Steps_A)$
  - for any state $s \in S$, if $s \in a$ then:

$$p_{MDP/P}^{min}(a,F) \leq p_{MDP}^{min}(s,F)$$
$$p_{MDP}^{max}(s,F) \leq p_{MDP/P}^{max}(a,F)$$

  - no information on the upper/lower bound for minimum/maximum reachability probabilities

Rapture – Abstraction

- Concrete model $MDP=(S,Steps)$, partition P & target states F
- Abstract (quotient) model $MDP_P=(A, Steps_A)$
  - for any state $s \in S$, if $s \in a$ then:

$$p_{MDP/P}^{min}(a,F) \leq p_{MDP}^{min}(s,F)$$
$$p_{MDP}^{max}(s,F) \leq p_{MDP/P}^{max}(a,F)$$

  - no information on the upper/lower bound for minimum/maximum reachability probabilities
  - can use abstract minimum probabilities as a lower bound for concrete maximum probabilities (and vice versa) but bounds can be very coarse
    - no reason for minimum and maximum probabilities to be close

Rapture – Abstraction

- Better suited to DTMCs?
  - in such cases have two sided bounds

$$p_{DTMC/P}^{min}(a,F) \leq p_{DTMC}(s,F) \leq p_{DTMC/P}^{max}(a,F)$$

  - minimum and maximum probabilities agree in the DTMC

Abstract Markov chains

- Abstract Markov Chains (AMCs) [Fecher et. al. 06]
  - abstraction approach for DTMCs
  - "interval valued" DTMCs
  - also considered in [Huth 05]

- Abstract Markov Chain AMC = (S, P^l, P^u)
  - S set of states
  - $P^l, P^u : S \times S \rightarrow [0,1]$ lower and upper bounds on transition probabilities such that for any $s, s' \in S$

$$P^l(s,s') \leq P^u(s,s') \text{ and } P^l(s,S) \leq 1 \leq P^u(s,S)$$

Abstract Markov chains – Abstraction

- Given a DTMC = $(S,\mathbf{P})$ and partition $P = \{S_1,\dots,S_n\}$
- Abstract DTMC given by the $AMC_P = (A,\mathbf{P}^l,\mathbf{P}^u)$ where
  - $A = \{S_1,\dots,S_n\}$ (abstract states are elements of the partition)
  - for any abstract states $a,a' \in A$

$$\mathbf{P}^l(a,a') = \min \{ \Sigma \{ P(s,s') \mid s' \in a' \} \mid s \in a \}$$

$$\mathbf{P}^u(a,a') = \max \{ \Sigma \{ P(s,s') \mid s' \in a' \} \mid s \in a \}$$

Abstract Markov chains – Abstraction

- Given a DTMC = $(S,\mathbf{P})$ and partition $P = \{S_1,\ldots,S_n\}$
- Abstract DTMC given by the $AMC_P=(A,\mathbf{P}^l,\mathbf{P}^u)$ where
  - $A = \{S_1,\ldots,S_n\}$ (abstract states are elements of the partition)
  - for any abstract states $a,a' \in A$

$$\mathbf{P}^l(a,a') = \min \{ \Sigma \{ P(s,s') \mid s' \in a' \} \mid s \in a \}$$

$$\mathbf{P}^u(a,a') = \max \{ \Sigma \{ P(s,s') \mid s' \in a' \} \mid s \in a \}$$

minimum probability of a state in a reaching the set of states a'

Abstract Markov chains – Abstraction

- Given a DTMC = $(S,\mathbf{P})$ and partition $P = \{S_1,\ldots,S_n\}$
- Abstract DTMC given by the $AMC_P=(A,\mathbf{P}^l,\mathbf{P}^u)$ where
  - $A = \{S_1,\ldots,S_n\}$ (abstract states are elements of the partition)
  - for any abstract states $a,a' \in A$

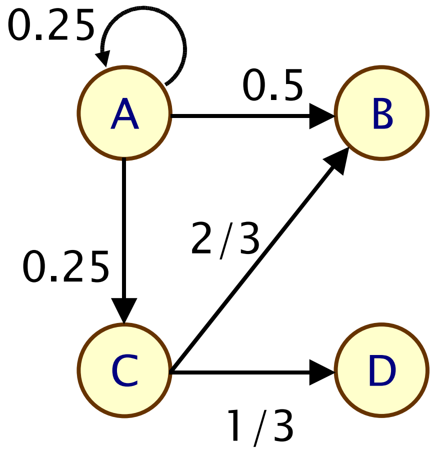
$$\mathbf{P}^l(a,a') = \min \{\ \Sigma\ \{\ P(s,s') \mid s' \in a'\ \} \mid s \in a\ \}$$

$$\mathbf{P}^u(a,a') = \max \{\ \Sigma\ \{\ P(s,s') \mid s' \in a'\ \} \mid s \in a\ \}$$

maximum probability of a state in a reaching the set of states a'

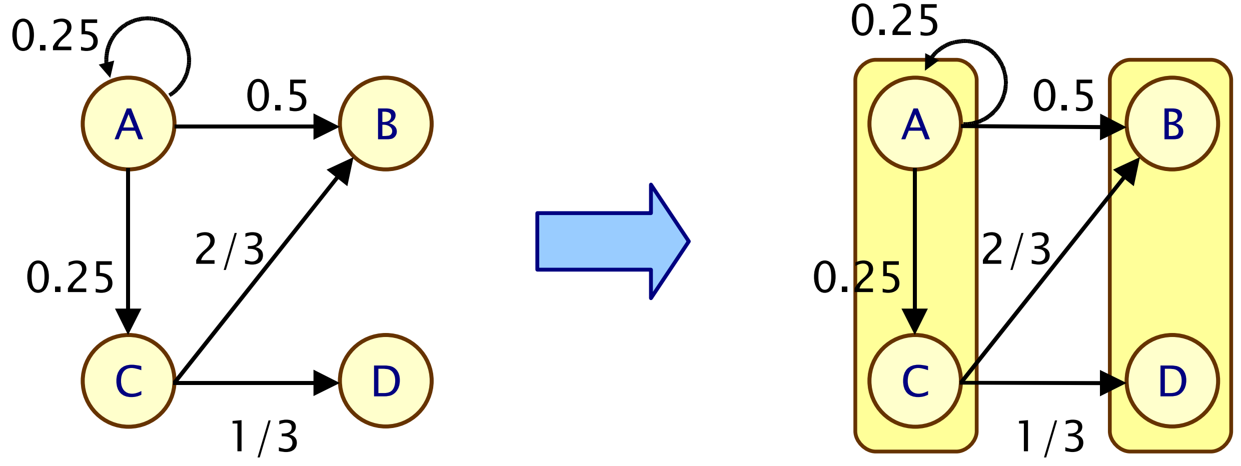
Abstract Markov chains – Example

- Partition { {A,C} , {B,D} }



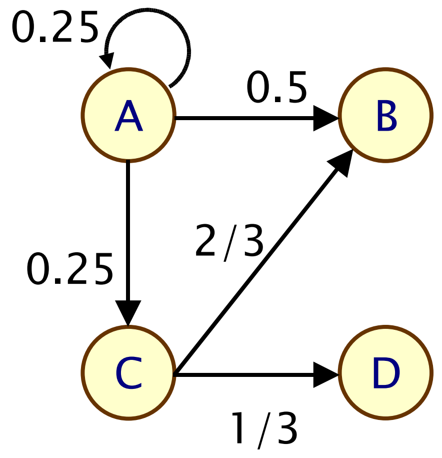
Abstract Markov chains – Example

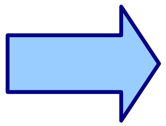
- Partition { {A,C} , {B,D} }

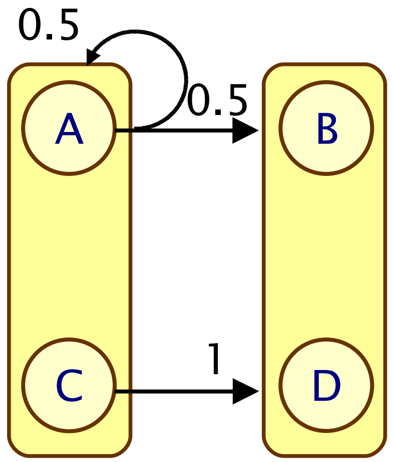


Abstract Markov chains – Example

- Partition { {A,C} , {B,D} }

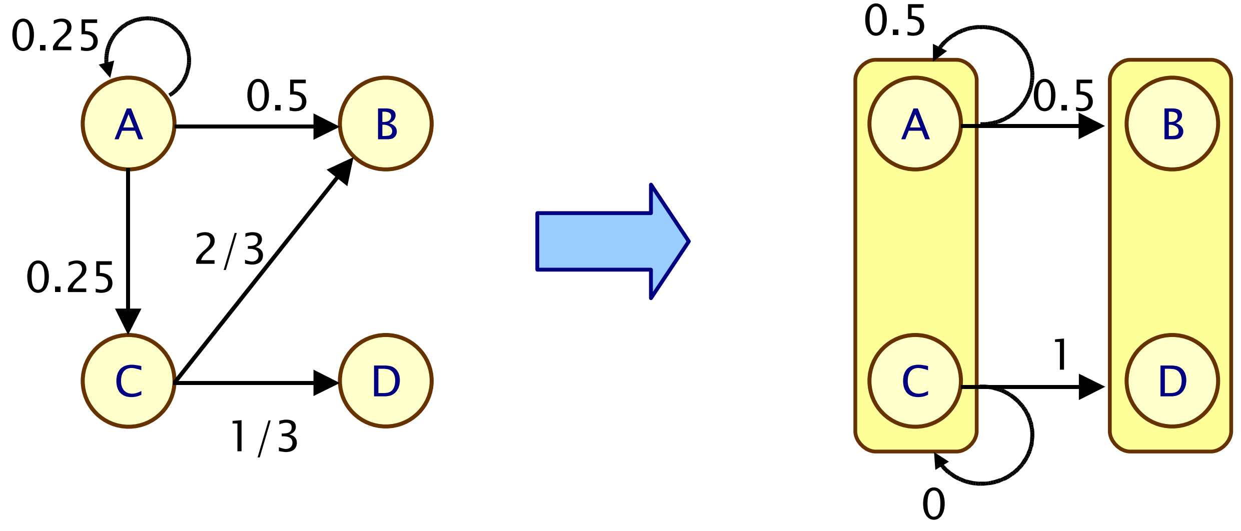






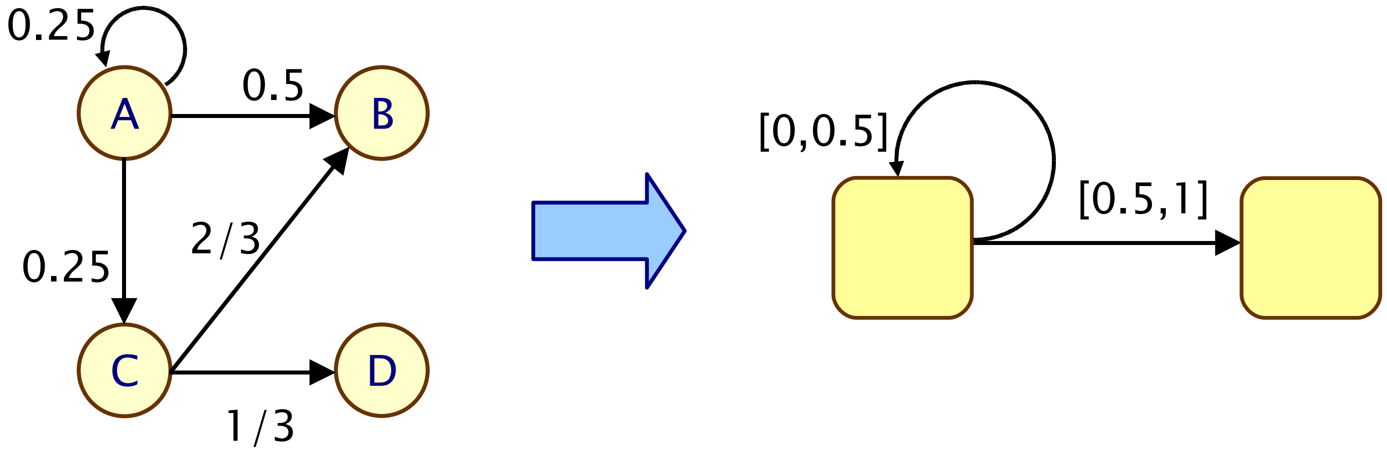
Abstract Markov chains – Example

- Partition { {A,C} , {B,D} }



Abstract Markov chains – Example

- Partition { {A,B} , {B,D} }



Abstract Markov chains – Semantics

- Semantics of AMC (S,P^l,P^u) given by MDP (S,Steps) where

  for any state s we have $\mu \in Steps(s)$ if and only if

  $$P^l(s,s') \leq \mu(s') \leq P^u(s,s') \text{ for all } s' \in S$$

  – probability of reaching any state is within the relevant interval

  – non-trivial intervals yield an infinite number of choices

  – if no non-trivial intervals the AMC is a DTMC

- Sufficient to consider a finite MDP (extremal distributions)

  – try and minimise or maximise reaching each states

  – leads to a MDP possibly exponentially larger than the AMC

Abstract Markov chains – Abstraction

- Reachability probabilities for AMCs
  - minimum and maximum probabilities (as for MDPs)
- Abstract AMC "simulates" the concrete DTMC
  - gives bounds on probabilities in the concrete DTMC

- Concrete model DTMC=(S,**P**), partition P & target states F
- Abstract model $\text{AMC}_P=(A,\mathbf{P}^l,\mathbf{P}^u)$
  - for any state $s \in S$, if $s \in a$ then:

$$p_{AMC}^{min}(a,F) \leq p_{DTMC}(s,F) \leq p_{AMC}^{max}(a,F)$$

Abstract Markov chains – Abstraction

- Reachability probabilities for AMCs
  - minimum and maximum probabilities (as for MDPs)
- Abstract AMC "simulates" the concrete DTMC
  - gives bounds on probabilities in the concrete DTMC

- Concrete model DTMC=(S,**P**), partition P & target states F
- Abstract model $AMC_P=(A,\mathbf{P}^l,\mathbf{P}^u)$
  - for any state $s \in S$, if $s \in a$ then:

$$p_{AMC}^{min}(a,F) \leq p_{DTMC}(s,F) \leq p_{AMC}^{max}(a,F)$$

  - the minimum reachability probability is an lower bound

Abstract Markov chains – Abstraction

- Reachability probabilities for AMCs
  - minimum and maximum probabilities (as for MDPs)
- Abstract AMC "simulates" the concrete DTMC
  - gives bounds on probabilities in the concrete DTMC

- Concrete model DTMC=(S,**P**), partition P & target states F
- Abstract model AMC_P=(A,$\mathbf{P}^l$,$\mathbf{P}^u$)
  - for any state s ∈ S, if s ∈ a then:

$$p_{\text{AMC}}^{\min}(a,F) \leq p_{\text{DTMC}}(s,F) \leq p_{\text{AMC}}^{\max}(a,F)$$

  - the maximum reachability probability is an upper bound

AMCs vs Rapture (MDPs)

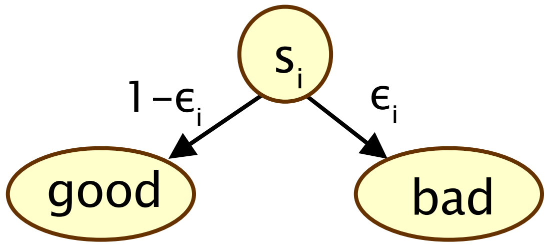
- AMCs lead to “smaller” abstractions
  - states s_i for $i=1,\ldots,n$
  - where $\epsilon_i < \epsilon_{i+1}$ for all $i=1,\ldots,n-1$

$$\begin{tikzcd}
& s_i \arrow[dl, "1-\epsilon_i"'] \arrow[dr, "\epsilon_i"] & \\
\text{good} & & \text{bad}
\end{tikzcd}$$

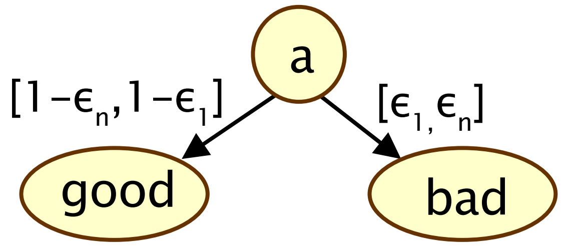
- Abstracting states $s_1,\ldots,s_n$
  - Rapture abstraction will have n different distributions
  - ith distribution gives probability $1-\epsilon_i$ of reaching “good”

AMCs vs Rapture (MDPs)

- AMCs lead to “smaller” abstractions
  - states s_i for $i=1,\ldots,n$
  - where $\epsilon_i < \epsilon_{i+1}$ for all $i=1,\ldots,n-1$



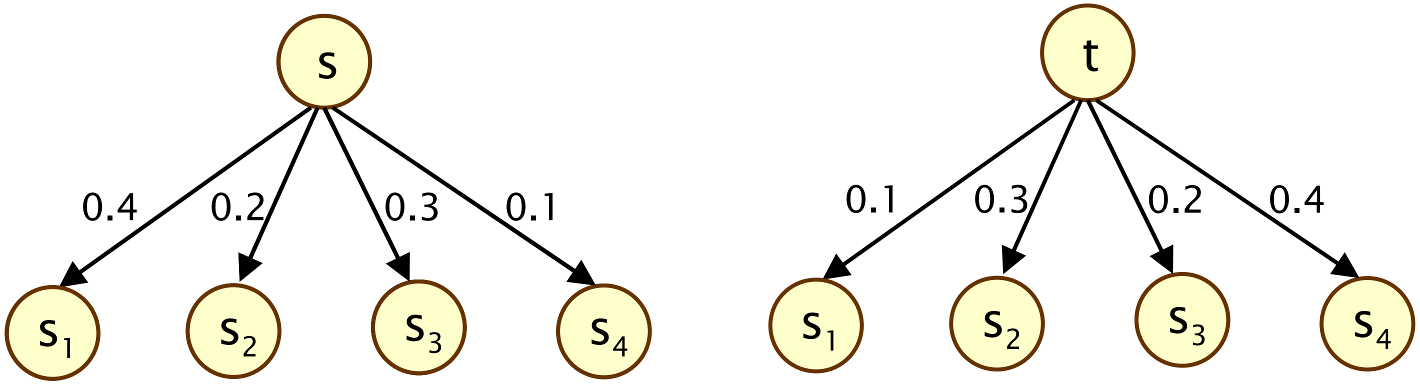
- Abstracting states $s_1,\ldots,s_n$
  - Rapture abstraction will have n different distributions
  - AMC abstraction is independent of n



  - abstractions will give same results with respect to minimum and maximum probabilities of reaching “good”/”bad” states

AMCs vs Rapture (MDPs)

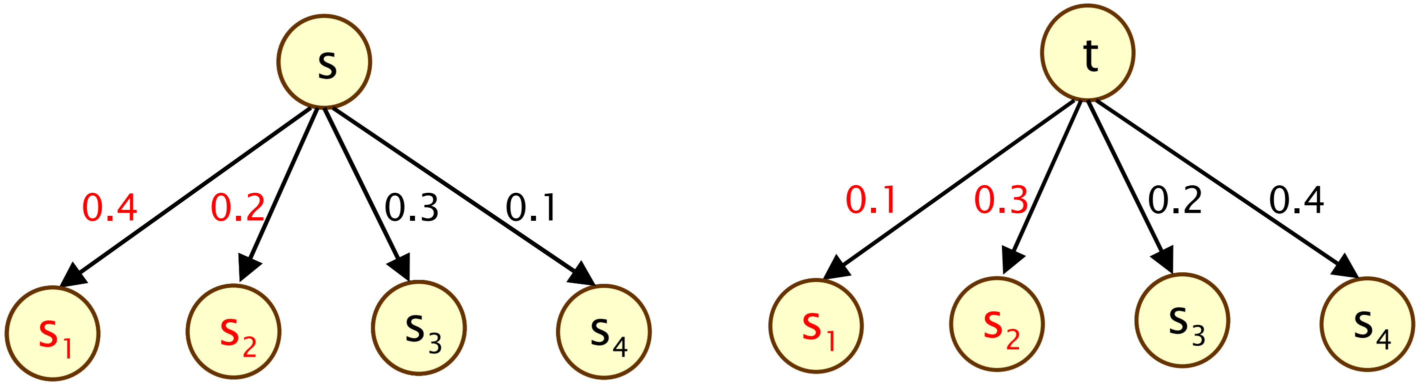
- AMCs are also less “precise”:



- Abstract s and t using the Rapture approach
  - choice between two distributions in the abstract state {s,t}
  - corresponding to choices in the concrete states s and t

AMCs vs Rapture (MDPs)

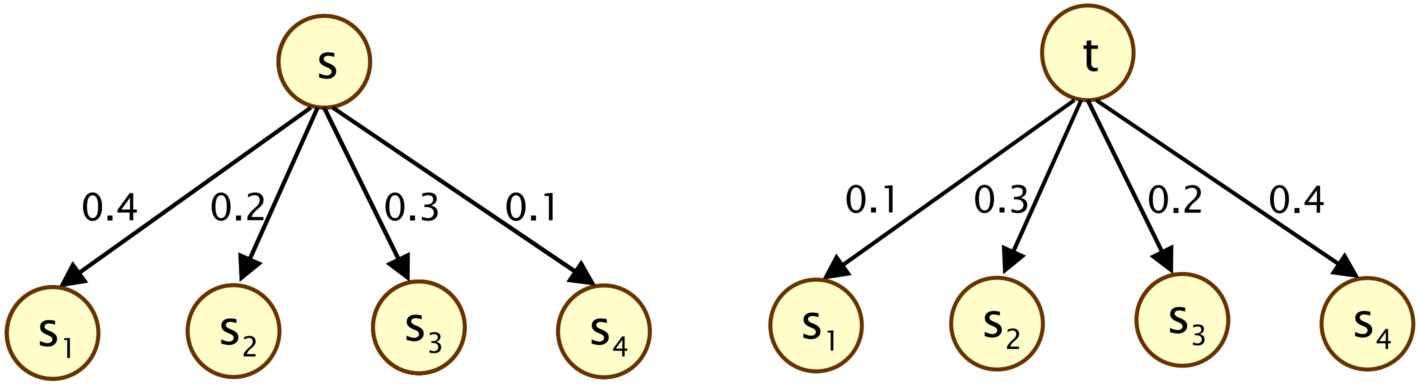
- AMCs are also less "precise":



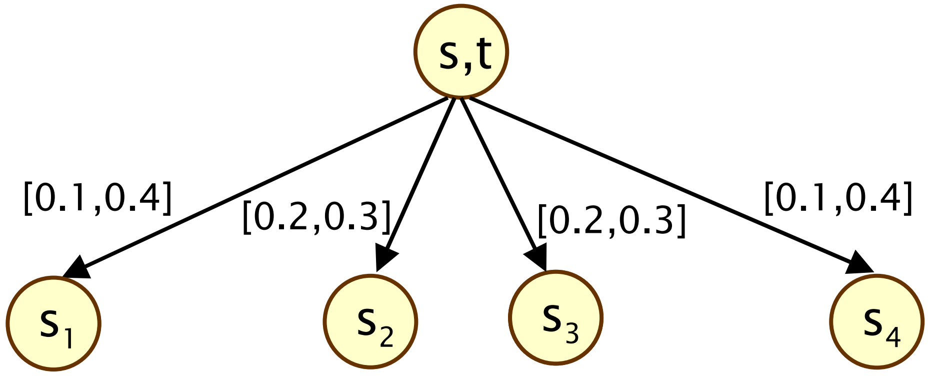
- Abstract s and t using the Rapture approach
  - choice between two distributions in the abstract state {s,t}
  - corresponding to choices in the concrete states s and t
  - maximum probability of reaching either s_1 or s_2 is 0.6
  - since probability from s_1 is 0.6 and from s_2 is 0.4

AMCs vs Rapture (MDPs)

- AMCs are also less “precise”:

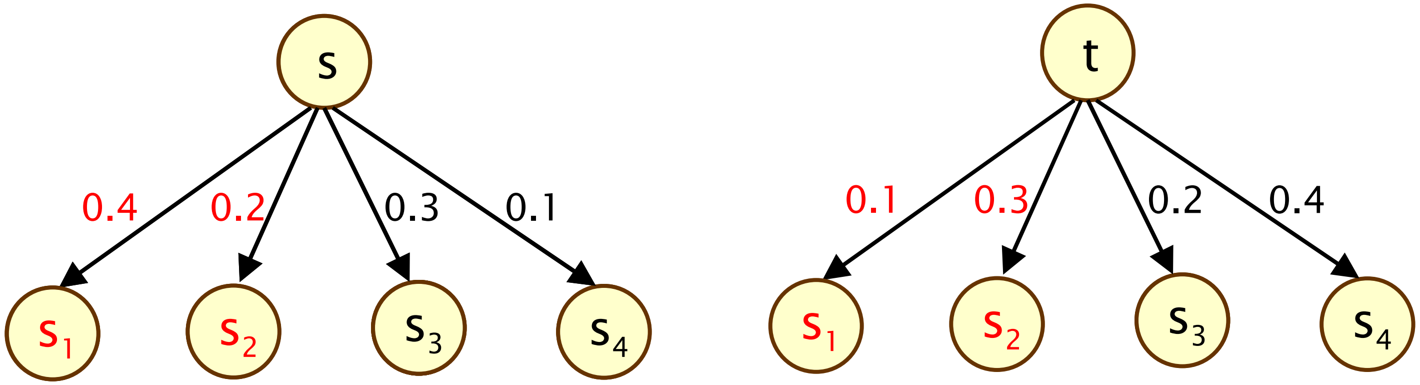


- Abstract s and t using the AMC approach

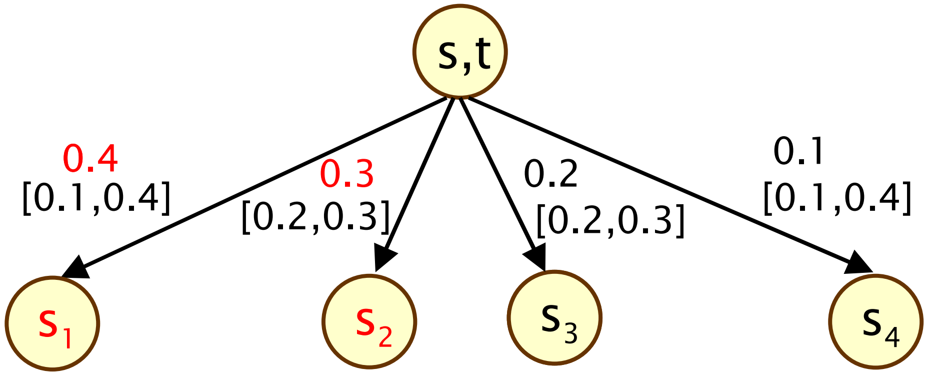


AMCs vs Rapture (MDPs)

- AMCs are also less “precise”:



- Abstract s and t using the AMC approach



  – maximum probability of reaching s_1 or s_2 is now 0.7 not 0.6

AMCs vs Rapture (MDPs) - Summary

- Rapture and AMC abstractions have same abstract states
  - the size of the partition
- AMCs more compact (number of transitions)
- AMCs more abstract (less precise bounds)

- Any practical examples of problems abstracting with MDP?
  - i.e. abstraction blows-up due to the number of transitions
  - otherwise why use a more abstract model?
  - systems constructed from high level language means states will have the same structure?
  - maybe not if one has parametrised distributions
  - need experimental results

Abstract Markov chains – CTMCs

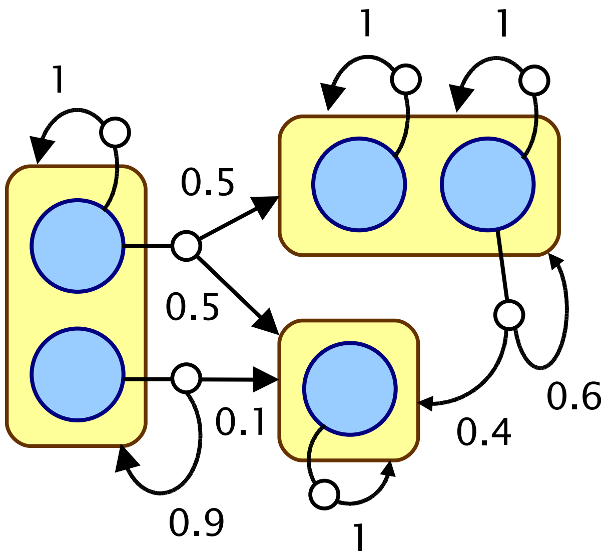
- Extension to AMC approach to CTMCs
  - [Katoen et. al. CAV 07]
- Can express a CTMC as (S,**P**,E) where
  - (S,**P**) is a DTMC
  - $E : S \to \mathbb{R}$ (E(s) is the exit rate from state s)
- Basic approach first translate to uniformised CTMC
  - the exit rates from all states are the same (adds loops to states)
- Perform abstraction on uniformised CTMC
  - essentially now abstracting a DTMC as all exit rates the same
  - using AMC abstraction approach can compute upper and lower bounds on time-bounded reachability
- Could use rapture approach
  - again will be less compact but more precise

Stochastic games

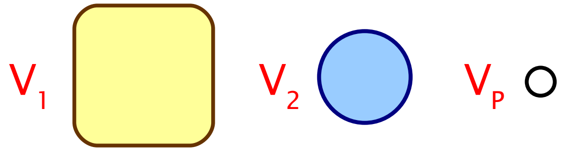
- Abstraction approach for MDPs based on stochastic two player games [Kwiatkowska et. al. 06]
- Abstraction increases degree of nondeterminism
- Key idea: separate two forms of nondeterminism
  - (a) from abstraction and (b) from original MDP
  - can then generate separate lower/upper bounds for min/max reachability probabilities
- For DTMCs reduces to MDPs (same as RAPTURE)

Stochastic games – Definition

- Simple stochastic games [Condon 02]
- Game $G = ((V,E),(V_1,V_2,V_P), \delta)$
  - (V,E) is a finite directed graph
  - (V_1,V_2,V_P) is a partition of V: 'player 1', 'player 2', 'probabilistic'
  - $\delta : V_P \rightarrow Dist(V)$ is a probabilistic transition function



- Execution of G: successor vertex chosen:
  - by player 1/2 for V_1/V_2 vertices
  - at random (δ) for V_P vertices



- MDPs can be thought of as stochastic two-player games with no V_2 vertices and strict alternation between V_1/V_P

Stochastic games – Definition

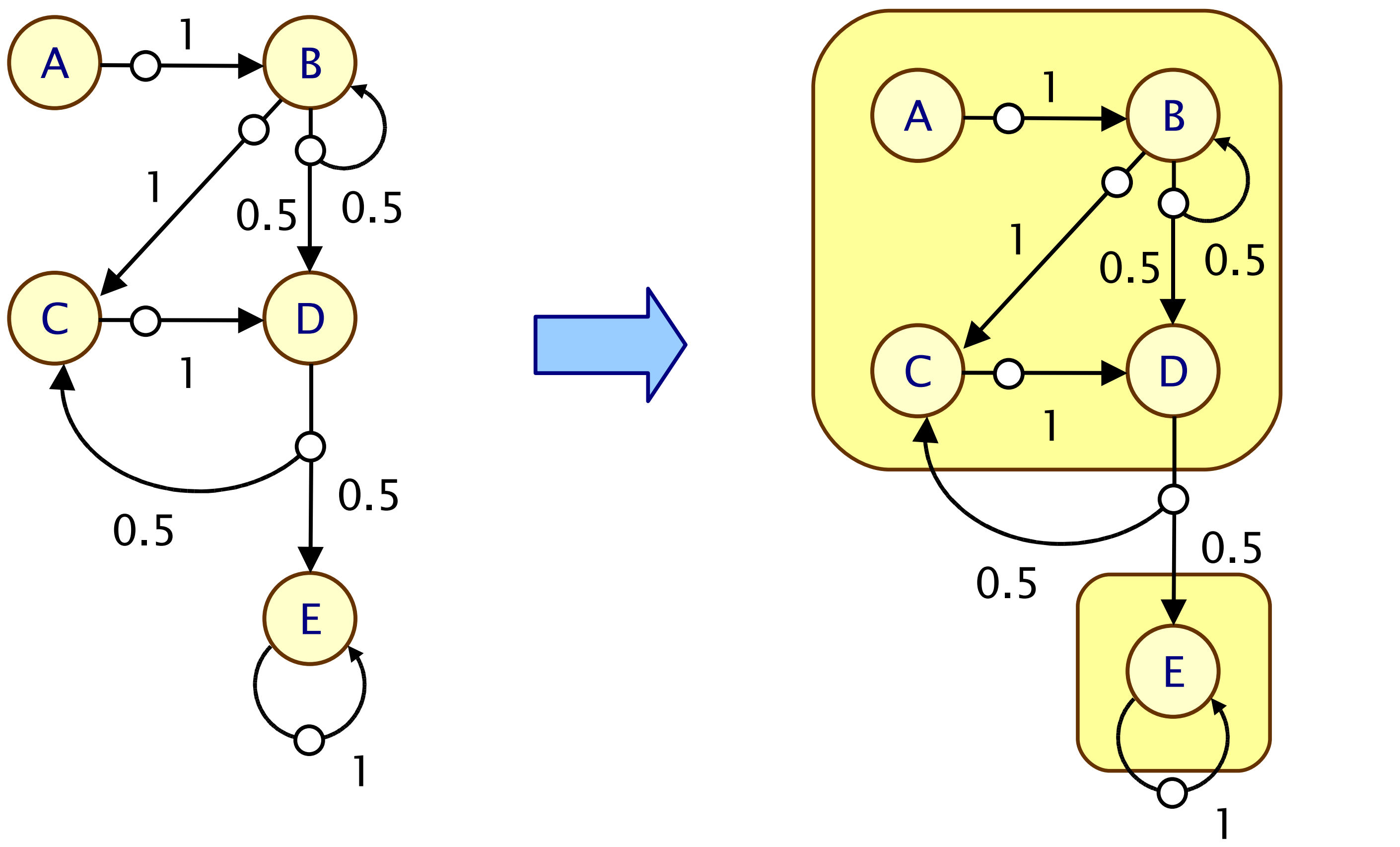
- Resolution of nondeterminism in a stochastic game
  - is done by a pair of strategies for players 1 and 2: (σ_1,σ_2)
  - under which the behaviour of the game is fully probabilistic
- Probabilistic reachability of vertex goal set F
  - $p_v^{\sigma1,\sigma2}(F)$ probability of reaching F from vertex v under (σ_1,σ_2)
- Optimal probabilities for player 1 and player 2
  - $\sup_{\sigma1} \inf_{\sigma2} p_v^{\sigma1,\sigma2}(F)$ and $\sup_{\sigma2} \inf_{\sigma1} p_v^{\sigma1,\sigma2}(F)$
  - computable via simple iterative methods, similar to MDPs

Stochastic games – Abstraction

- Abstract MDP is a two-player stochastic game
  - based on a partition P of MDP state space S
  - V_1 vertices are elements of P (subsets of S)
  - V_2 vertices are sets of prob. distributions (“states of MDP”)
  - V_P vertices are single probability distributions (over V_1)
  - strict alternation between V_1, V_2, V_P vertices
- Player 1 controls nondeterminism from abstraction
  - selects a state of the original MDP from a subset of S (in P)
- Player 2 controls nondeterminism from original MDP
  - selects a single probability distribution from a set

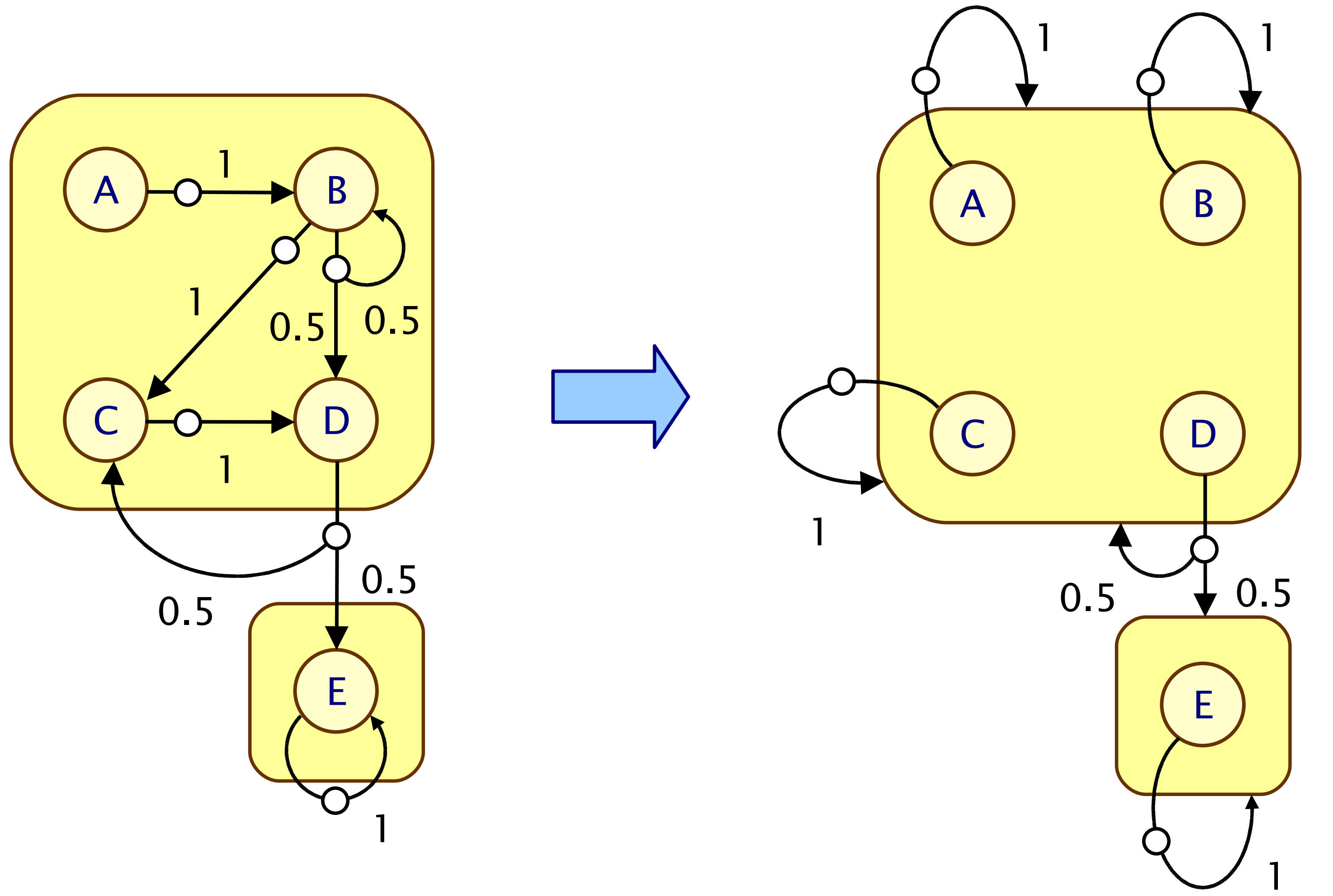
Stochastic games – Example

- Player $\boxed{1}$ vertices are partition elements (abstract states)



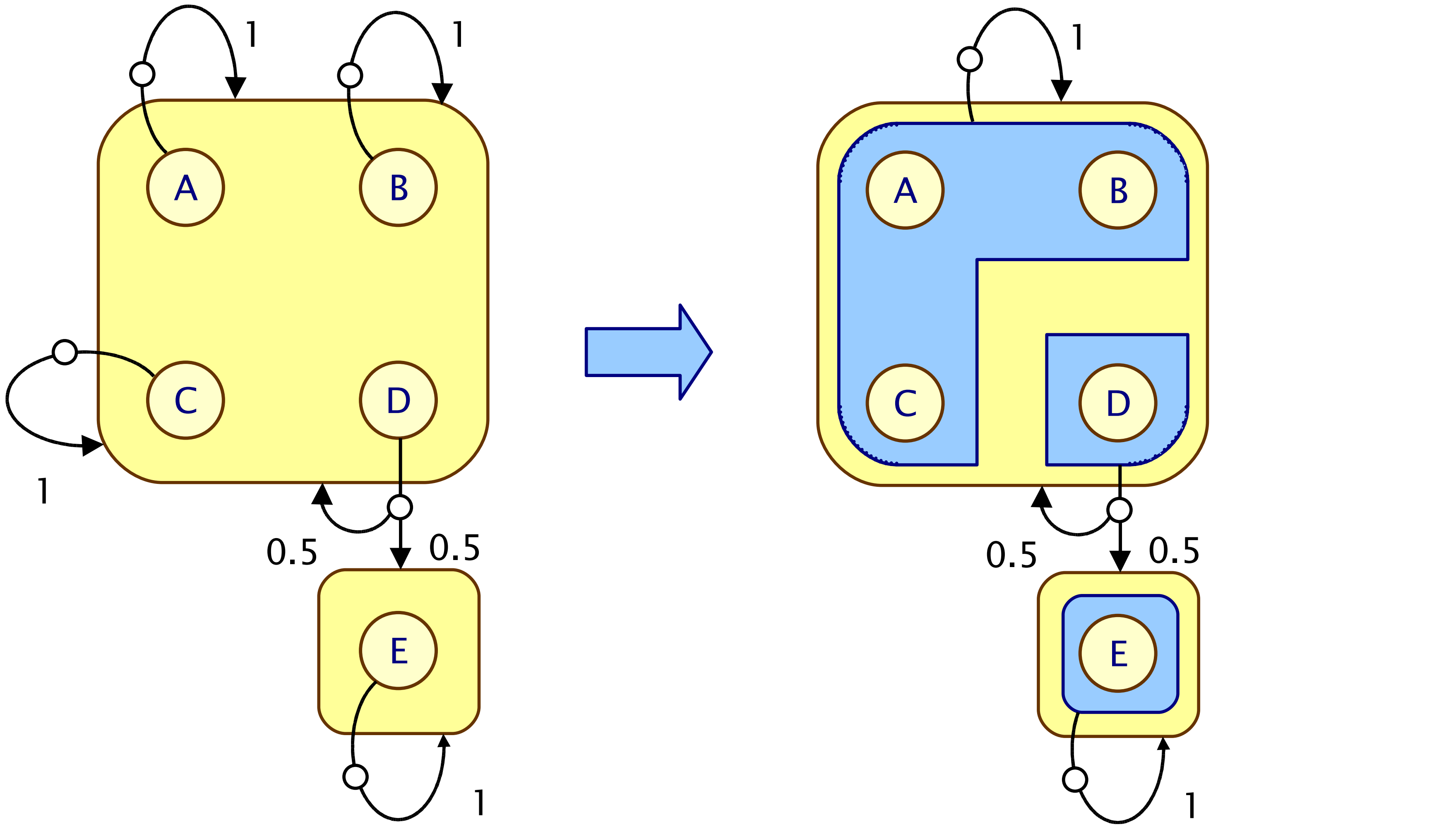
Stochastic games – Example

- (Sets of) distributions are lifted to the abstract state space



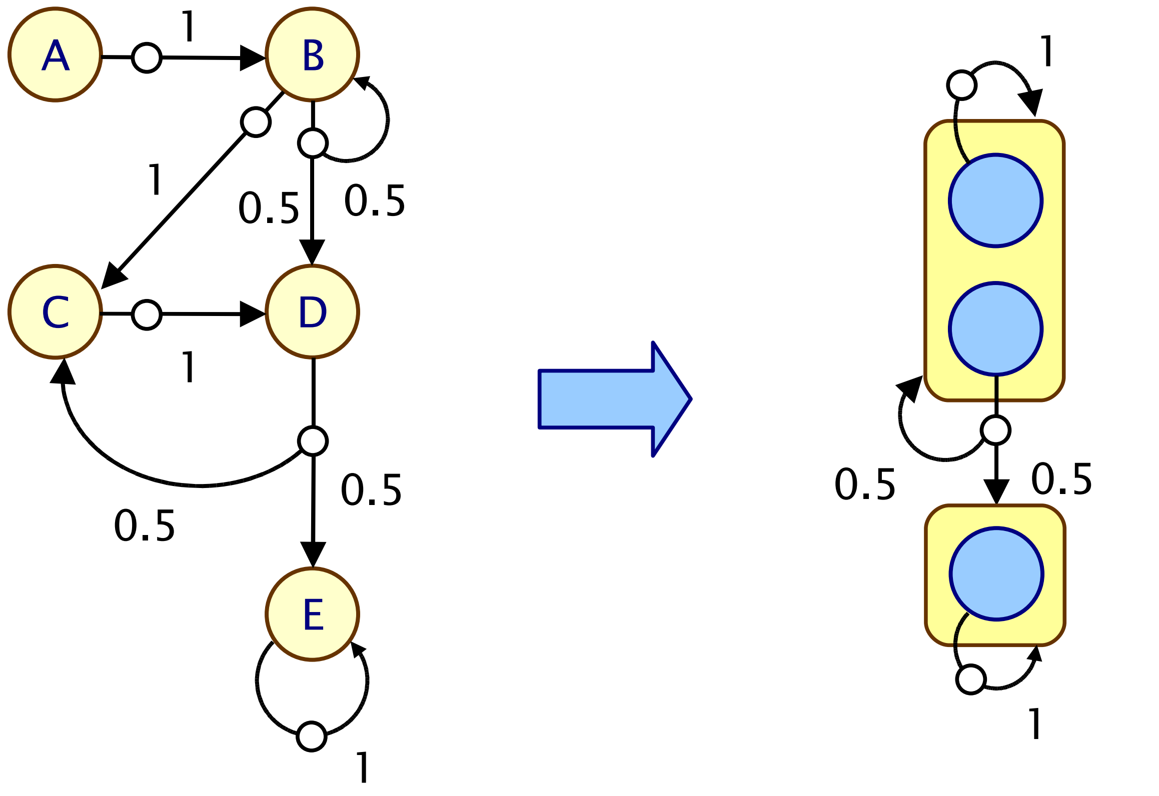
Stochastic games – Example

- States with same (sets of) choices form player $\boxed{2}$ vertices



Stochastic games – Example

- Complete transformation:



MDP

Abstract MDP

Stochastic games – Abstraction

- For a stochastic game built from an MDP and partition P
- Let $s \in S$ be an MDP state, $v \in V$ the corresponding game vertex (i.e. $s \in v$) and $F \in P$ a set of goal states
- Analysis of game yields lower/upper bounds for MDP:

$$\inf_{\sigma1,\sigma2} p_v^{\sigma1,\sigma2}(F) \quad \leq \quad p_s^{\min}(F) \;\leq\; \sup_{\sigma1} \inf_{\sigma2} p_v^{\sigma1,\sigma2}(F)$$

$$\sup_{\sigma2} \inf_{\sigma1} p_v^{\sigma1,\sigma2}(F) \quad \leq \quad p_s^{\max}(F) \;\leq\; \sup_{\sigma1,\sigma2} p_v^{\sigma1,\sigma2}(F)$$

Stochastic games – Abstraction

- For a stochastic game built from an MDP and partition P
- Let $s \in S$ be an MDP state, $v \in V$ the corresponding game vertex (i.e. $s \in v$) and $F \in P$ a set of goal states
- Analysis of game yields lower/upper bounds for MDP:

$$\inf_{\sigma1,\sigma2} p_v^{\sigma1,\sigma2}(F) \leq p_s^{min}(F) \leq \sup_{\sigma1} \inf_{\sigma2} p_v^{\sigma1,\sigma2}(F)$$

$$\sup_{\sigma2} \inf_{\sigma1} p_v^{\sigma1,\sigma2}(F) \leq p_s^{max}(F) \leq \sup_{\sigma1,\sigma2} p_v^{\sigma1,\sigma2}(F)$$

min/max reachability probabilities for original MDP

Stochastic games – Abstraction

- For a stochastic game built from an MDP and partition P
- Let $s \in S$ be an MDP state, $v \in V$ the corresponding game vertex (i.e. $s \in v$) and $F \in P$ a set of goal states
- Analysis of game yields lower/upper bounds for MDP:

$$\inf_{\sigma 1,\sigma 2} p_v^{\sigma 1,\sigma 2}(F) \leq p_s^{min}(F) \leq \sup_{\sigma 1} \inf_{\sigma 2} p_v^{\sigma 1,\sigma 2}(F)$$

$$\sup_{\sigma 2} \inf_{\sigma 1} p_v^{\sigma 1,\sigma 2}(F) \leq p_s^{max}(F) \leq \sup_{\sigma 1,\sigma 2} p_v^{\sigma 1,\sigma 2}(F)$$

optimal probabilities for player 1, player 2 in abstract MDP

Stochastic games – Abstraction

- For a stochastic game built from an MDP and partition P
- Let $s \in S$ be an MDP state, $v \in V$ the corresponding game vertex (i.e. $s \in v$) and $F \in P$ a set of goal states
- Analysis of game yields lower/upper bounds for MDP:

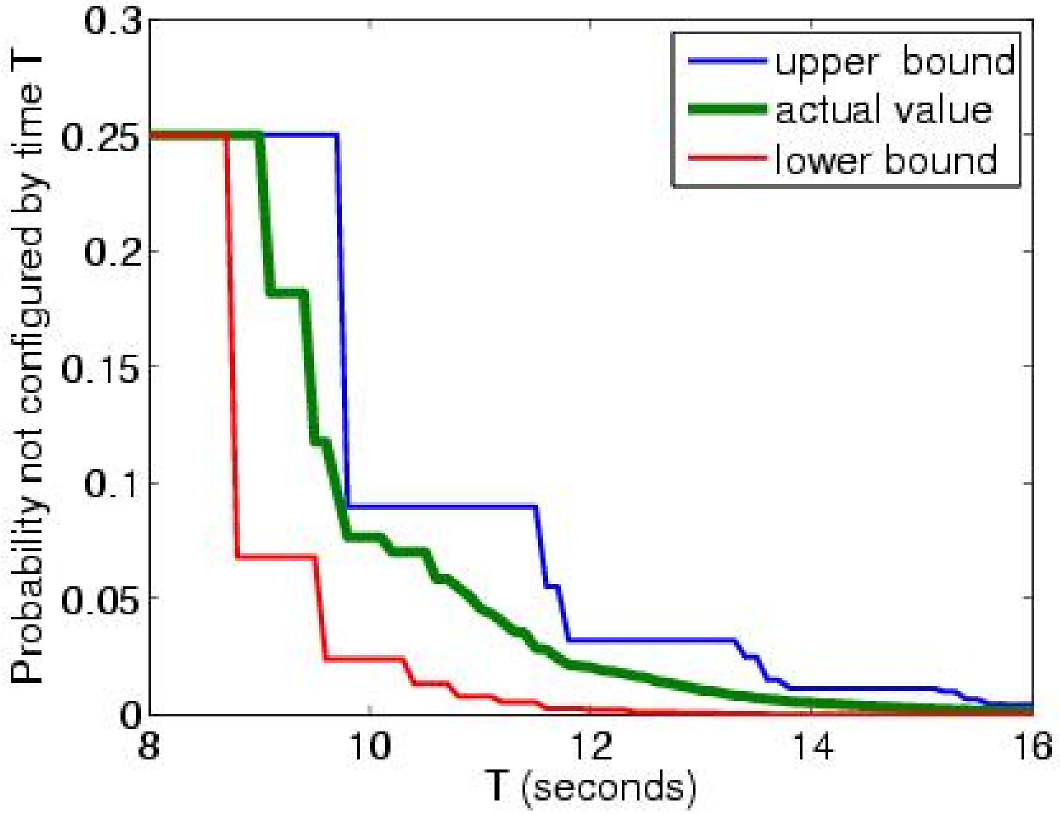
$$\inf_{\sigma_1,\sigma_2} p_v^{\sigma_1,\sigma_2}(F) \quad \leq \quad p_s^{\min}(F) \quad \leq \quad \sup_{\sigma_1} \inf_{\sigma_2} p_v^{\sigma_1,\sigma_2}(F)$$

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

like minimum/maximum reachability probabilities on MDPs (but performed on abstract MDP)

Stochastic games – Results

- N=8, M=32: MDP = 432,185 states, game = 881 vertices
- “Maximum probability not configured by time T”



Stochastic games – Summary

- Promising experimental results
  - but limited number of case studies
- Requires transitions to have the same structure
  - similar to comparison between MDPs and AMCs
- Compare with MDPs with intervals?
  - based on AMCs
  - will also separate two forms of nondeterminism

Existential abstraction – Probabilistic

- Existential abstraction in the probabilistic setting
  - can use probabilistic simulation [Segala & Lynch 94]
- What is the probabilistic abstraction (abstract system)
  - MDPs, Abstract Markov chains or two player stochastic games
- What is the model checking approach?
  - three valued logic ("true", "false", "do not know")
  - "probability bounded by p" or "probability in the interval $[p_1,p_2]$"
- How to refine when answers inconclusive?
  - what happens when we get "do not know", probability greater than/less than 0/1 or probability within the interval [0,1]
- How to implement?
  - predicate abstraction

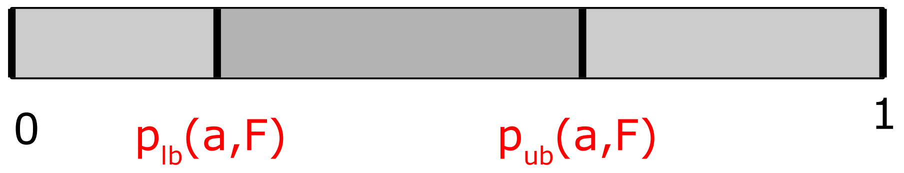
Model checking the abstract models

- Computing reachability probabilities...
- For MDPs can use value iteration
- For AMCs use algorithm based on value iteration
  - could reduce to MDP model checking but MDP possibly exponential in the size of the AMC
- For stochastic games use methods similar to value iteration

- In each case can reuse existing technology

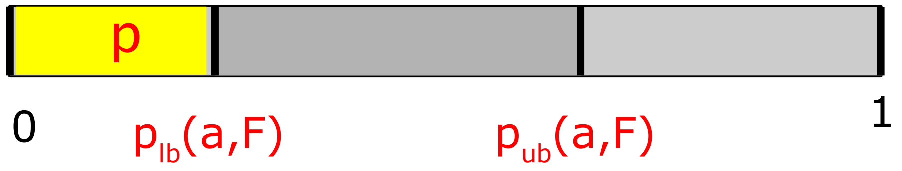
Model checking the abstract models

- Each model produces approximate results
  - upper and lower bounds on the actual probability
  - for Rapture and AMCs bounds on reachability probability
  - for games bounds on either the minimum or maximum probability reachability
- Suppose the verification problem is:
  - is the (min/max) probability of reaching F greater than p?
- Given a partition P calculate bounds for the abstraction



Model checking the abstract models

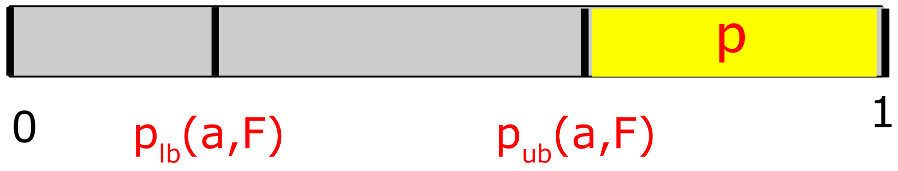
- Each model produces approximate results
  - upper and lower bounds on the actual probability
  - for Rapture and AMCs bounds on reachability probability
  - for games bounds on either the minimum or maximum probability reachability
- Suppose the verification problem is:
  - is the (min/max) probability of reaching F greater than p?
- Given a partition P calculate bounds for the abstraction



- Return "yes"/true (p is smaller than lower bound)

Model checking the abstract models

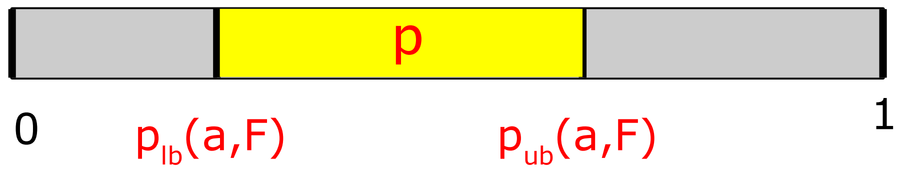
- Each model produces approximate results
  - upper and lower bounds on the actual probability
  - for Rapture and AMCs bounds on reachability probability
  - for games bounds on either the minimum or maximum probability reachability
- Suppose the verification problem is:
  - is the (min/max) probability of reaching F greater than p?
- Given a partition P calculate bounds for the abstraction



- Return "no"/false (p is larger than upper bound)

Model checking the abstract models

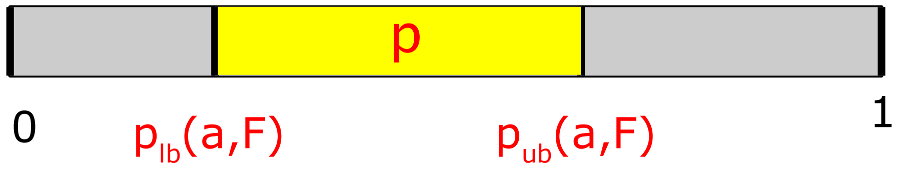
- Each model produces approximate results
  - upper and lower bounds on the actual probability
  - for Rapture and AMCs bounds on reachability probability
  - for games bounds on either the minimum or maximum probability reachability
- Suppose the verification problem is:
  - is the (min/max) probability of reaching F greater than p?
- Given a partition P calculate bounds for the abstraction



- Do not know so...

Model checking the abstract models

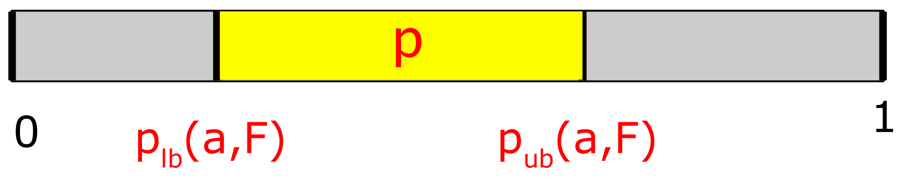
- Each model produces approximate results
  - upper and lower bounds on the actual probability
  - for Rapture and AMCs bounds on reachability probability
  - for games bounds on either the minimum or maximum probability reachability
- Suppose the verification problem is:
  - is the (min/max) probability of reaching F greater than p?
- Given a partition P calculate bounds for the abstraction



- Do not know so...
  - use three-valued logic (return “do not know”)

Model checking the abstract models

- Each model produces approximate results
  - upper and lower bounds on the actual probability
  - for Rapture and AMCs bounds on reachability probability
  - for games bounds on either the minimum or maximum probability reachability
- Suppose the verification problem is:
  - is the (min/max) probability of reaching F greater than p?
- Given a partition P calculate bounds for the abstraction



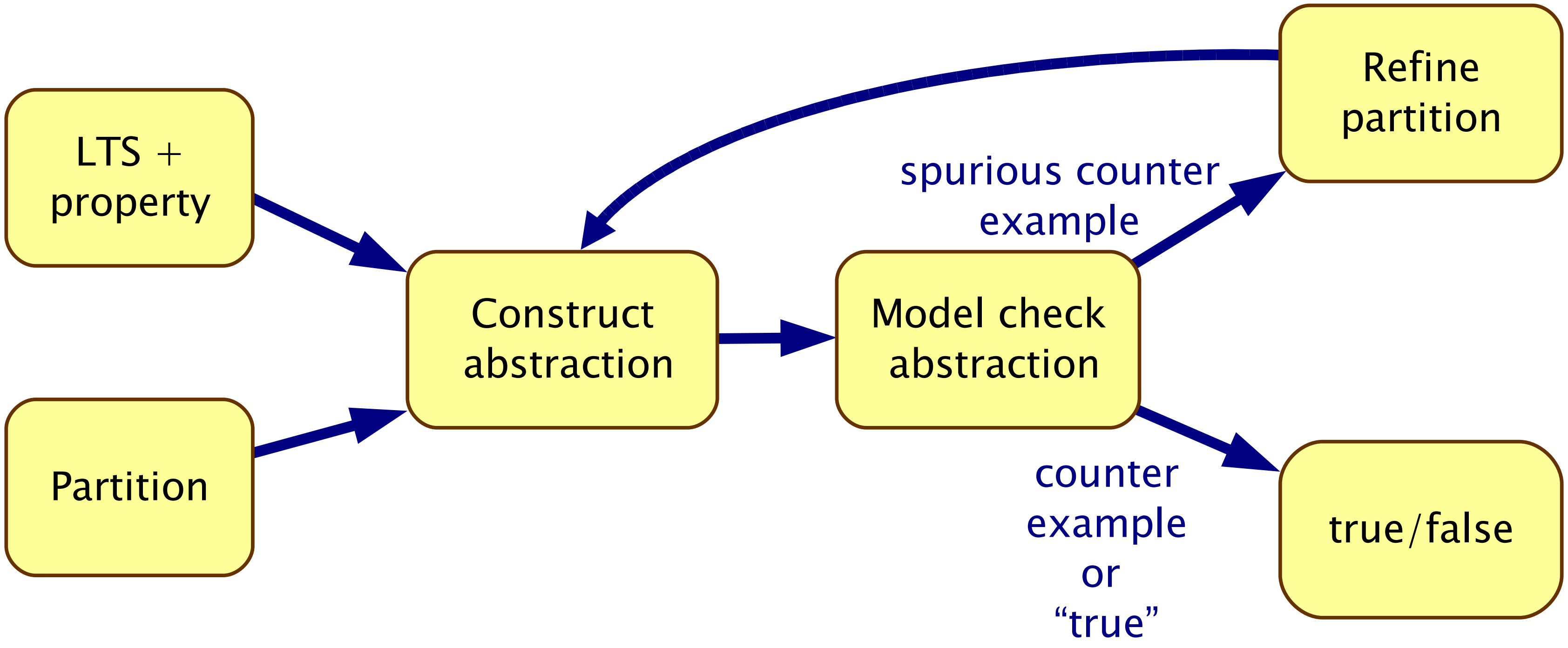
- Do not know so...
  - or refine the abstraction ...

Existential abstraction – Probabilistic

- Existential abstraction in the probabilistic setting
  - can use probabilistic simulation [Segala & Lynch 94]
- What is the probabilistic abstraction (abstract system)
  - MDPs, Abstract Markov chains or two player stochastic games
- What is the model checking approach?
  - three valued logic (“true”, “false”, “do not know”)
  - “probability bounded by p” or “probability in the interval $[p_1,p_2]$”
- How to refine when answers inconclusive?
  - what happens when we get “do not know”, probability greater than/less than 0/1 or probability within the interval [0,1]
- How to implement?
  - predicate abstraction

Refinement – CEGAR

- In the non-probabilistic setting....
  - counterexample-guided abstraction refinement (CEGAR)



Refinement – Probabilistic

- For all approaches a “finer” partition yields tighter bounds
- How to refine?
  - probabilistic model checking algorithms do not return counterexamples
- What is a counterexample?
  - no single path implies probability above/below a bound
  - find paths with largest probability mass
    - time bounded reachability in CTMCs [Aljazzar et. al. 05]
    - reachability in DTMCs (and CTMCs) [Han & Katoen 07]

Refinement – Probabilistic

- In (almost) all cases the upper and lower bounds give us information as to the quality of the abstraction

- This also gives a possible method for refinement
  - exists adversaries which obtain the upper and lower bounds
  - one of these bounds cannot be equal to the correct probability
  - therefore the choices made by ones of these adversaries must be “spurious”
  - such “extremal” adversaries are computed during computation of the probabilities therefore no extra work in finding the adversaries

Refinement – Rapture

- Start with an initial coarse abstraction including
  - the set of initial states and the set of target states
  - sets of states for which minimum/maximum probability is 1/0
    - computed through qualitative precomputation (graph analysis)
- Refinement
  - splitter: set of states with the same abstract transitions
- Heuristics
  - partition based on the control structure
    - e.g. abstract data variables not program counters
  - allow user to specify variables to abstract/not abstract
  - either refine all partitions (fast) or refine one partition at a time (smaller models to verify)

Existential abstraction – Probabilistic

- Existential abstraction in the probabilistic setting
  - can use probabilistic simulation [Segala & Lynch 94]
- What is the probabilistic abstraction (abstract system)
  - MDPs, Abstract Markov chains or two player stochastic games
- What is the model checking approach?
  - three valued logic (“true”, “false”, “do not know”)
  - “probability bounded by p” or “probability in the interval $[p_1,p_2]$”
- How to refine when answers inconclusive?
  - what happens when we get “do not know”, probability greater than/less than 0/1 or probability within the interval [0,1]
- How to implement?
  - predicate abstraction

Model-based abstraction – Tools

- Construct abstraction for language level description through predicate abstraction [Graf & Saïdi 97]
- Idea: given set of predicates $\{\phi_1,\ldots,\phi_n\}$
  - formulas describing properties of system states
- Abstract State Space: tuples of Boolean variables $(b_1,\ldots,b_n)$
  - representing sets of concrete states
  - b_i = true implies all states in the set satisfy ϕ_i
- Galois Connection between concrete and abstract systems
  - concretisation function $\gamma : A \rightarrow 2^S$ where

$$\gamma(b_1,\ldots,b_n) = \{s \in S \mid \phi_1(s)=b_1 \wedge \ldots \wedge \phi_n(s)=b_n\}$$

  - abstraction function $\alpha : 2^S \rightarrow A$ where for any $S' \subseteq S$

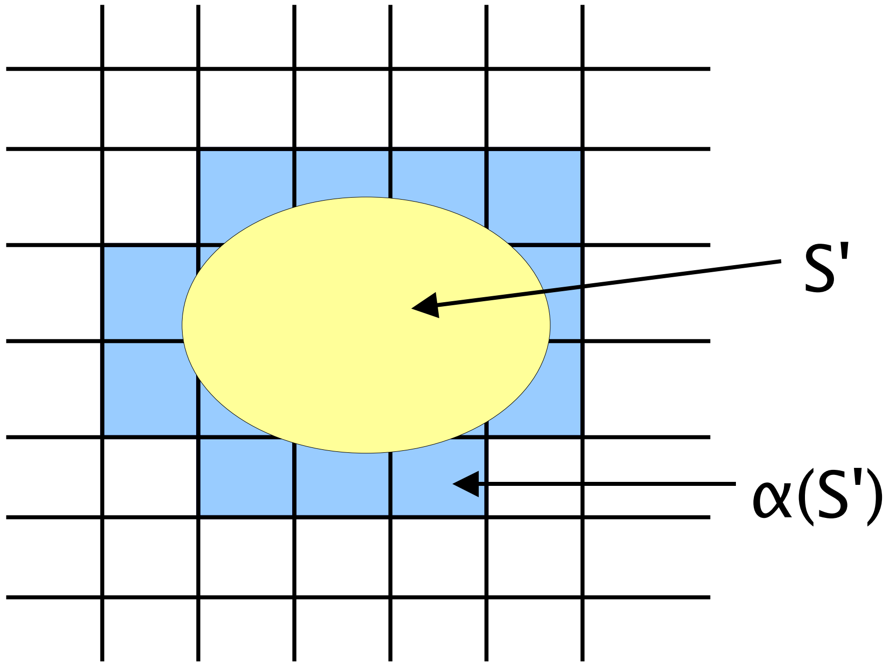
$$\alpha(S') = \{ (b_1,\ldots,b_n) \mid S' \subseteq \gamma(b_1,\ldots,b_n) \}$$

Predicate Abstraction

- abstraction function $\alpha : 2^S \rightarrow A$ where for any $S' \subseteq S$

  $$\alpha(S') = \{ (b_1,\ldots,b_n) \mid S' \subseteq \gamma(b_1,\ldots,b_n) \}$$

  – abstraction function approximates a set of concrete states by a set of predicates



Predicate Abstraction

- Abstract transition relation given by

  $(a,a') \in \mathbf{T}_A$ if and only if $\exists\ s,s' \in S.(\ (s,s') \in \mathbf{T} \wedge \alpha(s)=a \wedge \alpha(s')=a'\ )$
- How to construct the abstract transition relation?
- Original approach based on using theorem proving techniques
- More successful approach based on SAT-solvers
  - search for a solution to the formula

    $$\theta(a,a') = \exists s,s' \in S.(\ (s,s') \in \mathbf{T} \wedge \alpha(s)=a \wedge \alpha(s')=a'\ )$$
  - find solution (b,b')
  - add (b,b') to the abstract transition relation
  - add $(a \neq b) \wedge (a' \neq b')$ to the formula $\theta(a,a')$ and search again
  - repeat until formula is unsatisfiable

Predicate Abstraction – Probabilistic

- PASS tool [Wachter, Zhang & Hermanns 07]
  - Predicate Abstraction for Stochastic Systems
- Combines Rapture approach with predicate abstraction
  - (i.e. aimed an abstracting DMTCs and MDPs)
- Abstract high level model description (PRISM language)
  - map each concrete command to a (set of) abstract command(s)
    - [action] guard → update
  - uses SMT solver (SAT based)
    - SMT = Satisfiability Modulo Theories (extend propositional satisfiability with richer theories e.g linear integer arithmetic)
- Promising preliminary results
  - only one case study (BRP) so far

Predicate Abstraction – AMCs and Games

- Not as straight-forward cannot look at individual commands separately
  - one transition/command of the abstract system cannot be constructed from a single concrete transition/command
- In both cases need to look at how commands "overlap"
  - i.e. when different sets of commands are enabled
  - each combination of enabled commands may lead to different abstract commands
  - over the reachable concrete state space there may be a small number of combinations possible
  - however over the concrete "product state-space" there may be an exponential number of combinations
  - without the concrete state space may get an exponential blow-up in the number of commands

Conclusions

- Need to investigate the difference between the approaches
  - include experimental comparisons
- Exact approaches well studied
  - limited work on MDPs, weak bisimuation and language-level approaches
- Approximate approaches many open questions/problems
  - what is the “best” abstract model?
  - how to refine?
  - what are good counterexamples?
  - extend to language level (both abstraction and refinement)?