Probabilistic Model Checking

Marta Kwiatkowska
Dave Parker

Oxford University Computing Laboratory

ESSLLI'10 Summer School, Copenhagen, August 2010

Part 5

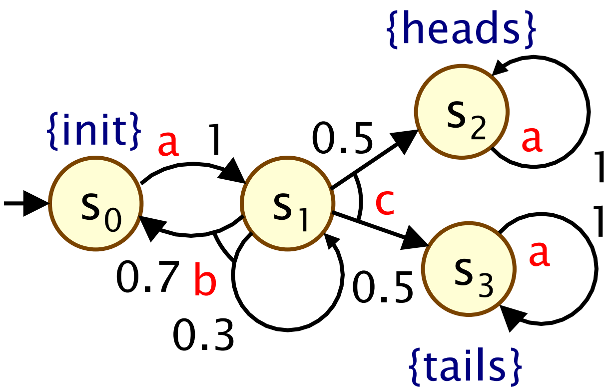
Probabilistic timed automata

Probabilistic models

	Fully probabilistic	Nondeterministic
Discrete time	Discrete-time Markov chains (DTMCs)	Markov decision processes (MDPs) (probabilistic automata)
Continuous time	Continuous-time Markov chains (CTMCs)	CTMDPs/IMCs
		Probabilistic timed automata (PTAs)

Recall – MDPs

- Markov decision processes (MDPs)
  - mix probability and nondeterminism
  - in a state, there is a nondeterministic choice between multiple probability distributions over successor states



- Adversaries
  - resolve nondeterministic choices based on history so far
  - properties quantify over all possible adversaries
  - e.g. $P_{<0.1}[\lozenge \text{err}]$ – maximum probability of an error is < 0.1

Real-world protocol examples

- Systems with probability, nondeterminism and real-time
  - e.g. communication protocols, randomised security protocols

- Randomised back-off schemes
  - Ethernet, WiFi (802.11), Zigbee (802.15.4)
- Random choice of waiting time
  - Bluetooth device discovery phase
  - Root contention in IEEE 1394 FireWire
- Random choice over a set of possible addresses
  - IPv4 dynamic configuration (link-local addressing)
- Random choice of a destination
  - Crowds anonymity, gossip-based routing

Overview (Part 5)

- Time, clocks and zones
- Probabilistic timed automata (PTAs)
  - definition, examples, semantics, time divergence
- PTCTL: A temporal logic for for PTAs
  - syntax, examples, semantics
- Model checking for PTAs
  - the region graph
  - digital clocks
  - zone-based approaches:
  - (i) forwards reachability
  - (ii) backwards reachability
  - (iii) game-based abstraction refinement
- Costs and rewards

Time, clocks and clock valuations

- Dense time domain: non-negative reals $\mathbb{R}_{\geq 0}$
  - from this point on, we will abbreviate $\mathbb{R}_{\geq 0}$ to $\mathbb{R}$

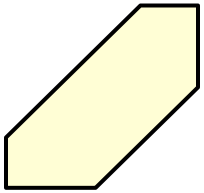
- Finite set of clocks $x \in X$
  - variables taking values from time domain $\mathbb{R}$
  - increase at the same rate as real time

- A clock valuation is a tuple $v \in \mathbb{R}^X$. Some notation:
  - $v(x)$: value of clock x in v
  - $v+t$: time increment of t for v
    - $(v+t)(x) = v(x)+t \quad \forall x \in X$
  - $v[Y:=0]$: clock reset of clocks $Y \subseteq X$ in v
    - $v[Y:=0](x) = 0$ if $x \in Y$ and $v(x)$ otherwise

Zones (clock constraints)

- Zones (clock constraints) over clocks X, denoted Zones(X):

$$\zeta ::= x \leq d \;\mid\; c \leq x \;\mid\; x{+}c \leq y{+}d \;\mid\; \neg\zeta \;\mid\; \zeta \vee \zeta$$



  - where $x, y \in X$ and $c, d \in \mathbb{N}$
  - used for both syntax of PTAs/properties and algorithms

- Can derive:
  - logical connectives, e.g. $\zeta_1 \wedge \zeta_2 \equiv \neg(\neg\zeta_1 \vee \neg\zeta_2)$
  - strict inequalities, through negation, e.g. $x{>}5 \equiv \neg(x{\leq}5)$...

- Some useful classes of zones:
  - closed: no strict inequalities (e.g. $x{>}5$)
  - diagonal-free: no comparisons between clocks (e.g. $x{\leq}y$)
  - convex: define a convex set, efficient to manipulate

Zones and clock valuations

- A clock valuation v satisfies a zone ζ, written $v \triangleright \zeta$ if
  - ζ resolves to true after substituting each clock x with $v(x)$

- The semantics of a zone $\zeta \in \text{Zones}(X)$ is the set of clock valuations which satisfy it (i.e. a subset of $\mathbb{R}^X$)
  - NB: multiple zones may have the same semantics
  - e.g. $(x\leq 2)\wedge(y\leq 1)\wedge(x\leq y+2)$ and $(x\leq 2)\wedge(y\leq 1)\wedge(x\leq y+3)$

- We consider only canonical zones
  - i.e. zones for which the constraints are as 'tight' as possible
  - $O(|X|^3)$ algorithm to compute (unique) canonical zone [Dil89]
  - allows us to use syntax for zones interchangeably with semantic, set-theoretic operations

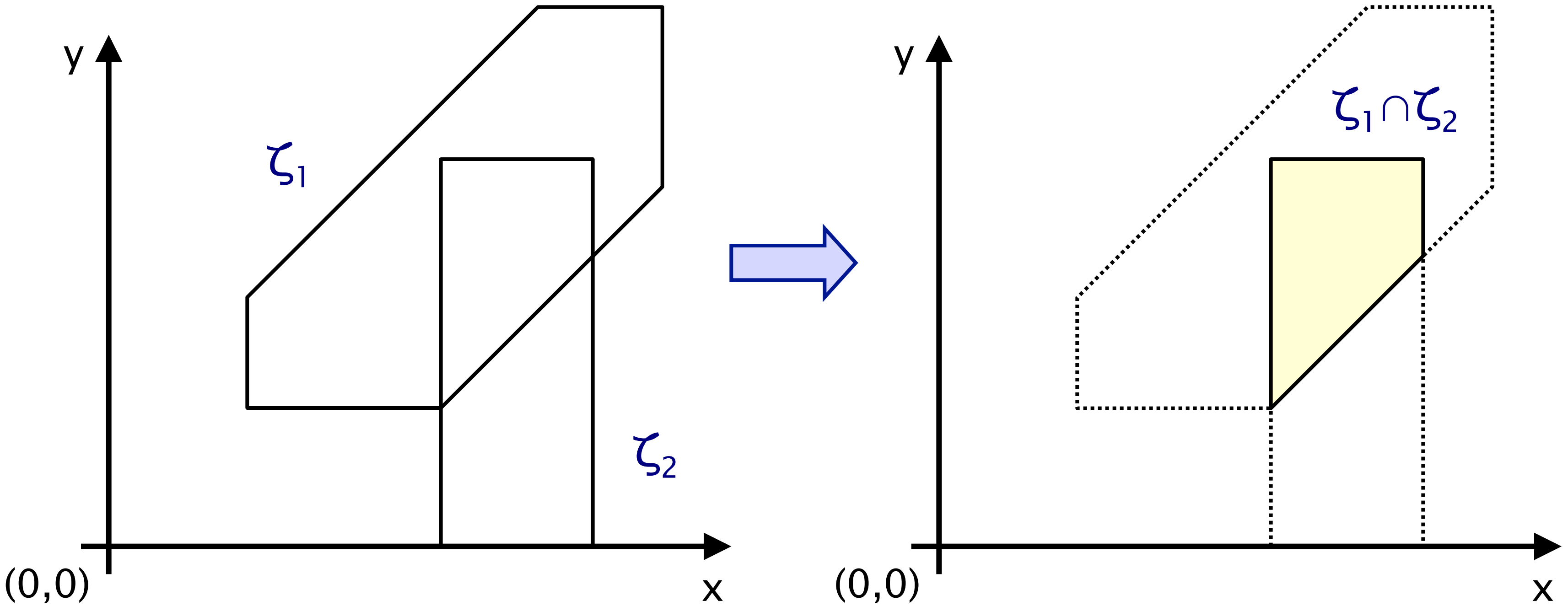
c-equivalence and c-closure

- Clock valuations v and v' are c-equivalent if for any $x,y \in X$
  - either $v(x) = v'(x)$, or $v(x) > c$ and $v'(x) > c$
  - either $v(x)-v(y) = v'(x)-v'(y)$ or $v(x)-v(y) > c$ and $v'(x)-v'(y) > c$

- The c-closure of the zone ζ, denoted close(ζ,c), equals
  - the greatest zone $\zeta' \supseteq \zeta$ such that, for any $v' \in \zeta'$, there exists $v \in \zeta$ and v and v' are c-equivalent
  - c-closure ignores all constraints which are greater than c
  - for a given c, there are only a finite number of c-closed zones

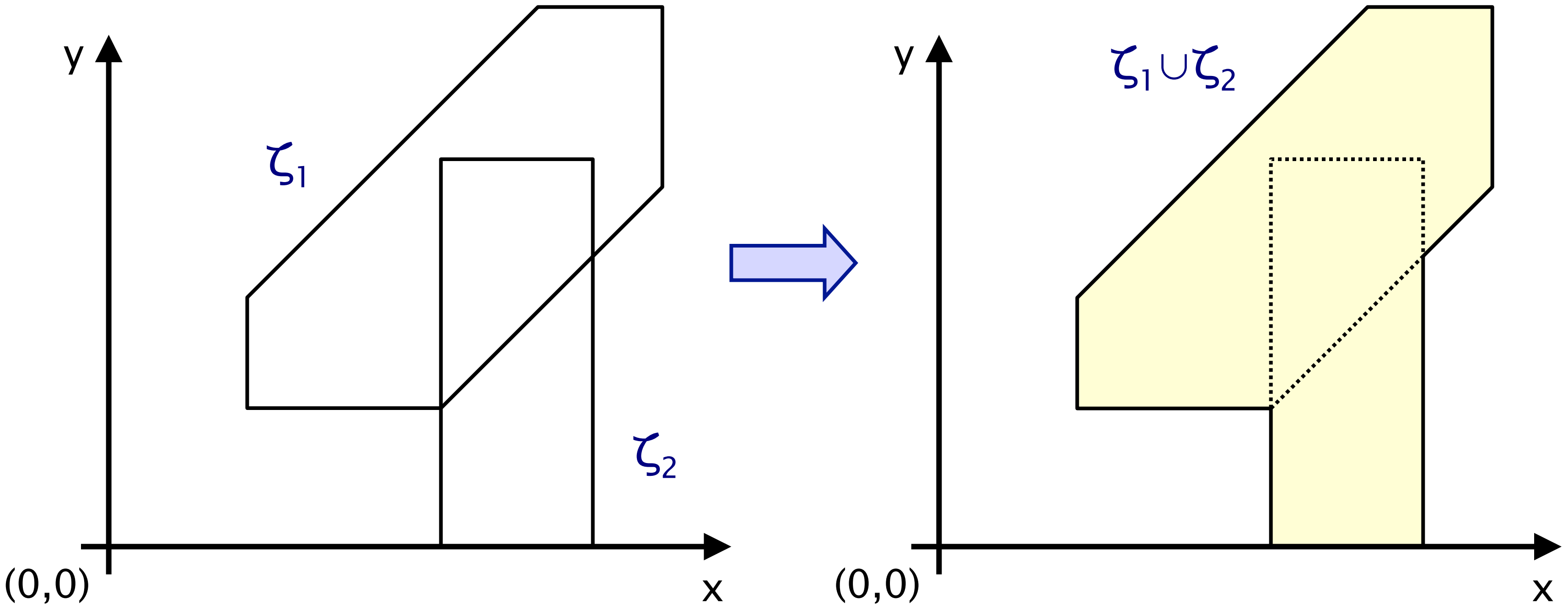
Operations on zones – Set theoretic

- Intersection of two zones: $\zeta_1 \cap \zeta_2$



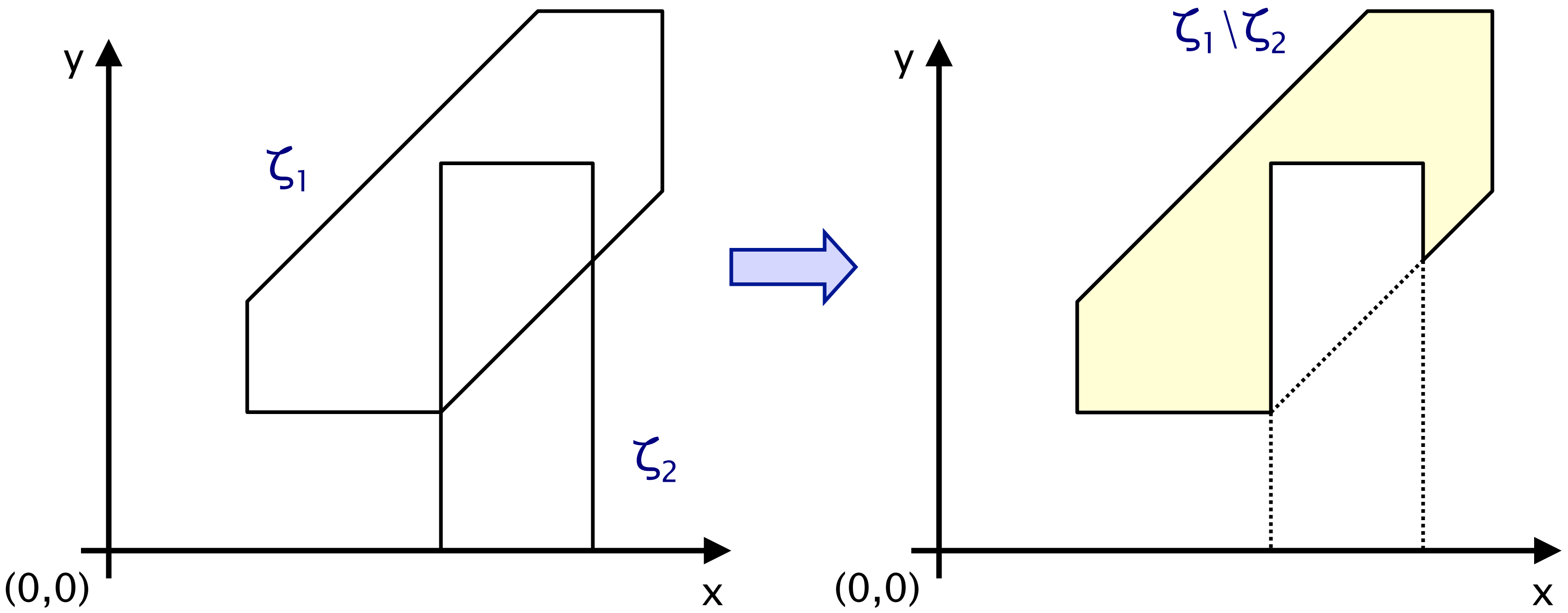
Operations on zones – Set theoretic

- Union of two zones: $\zeta_1 \cup \zeta_2$



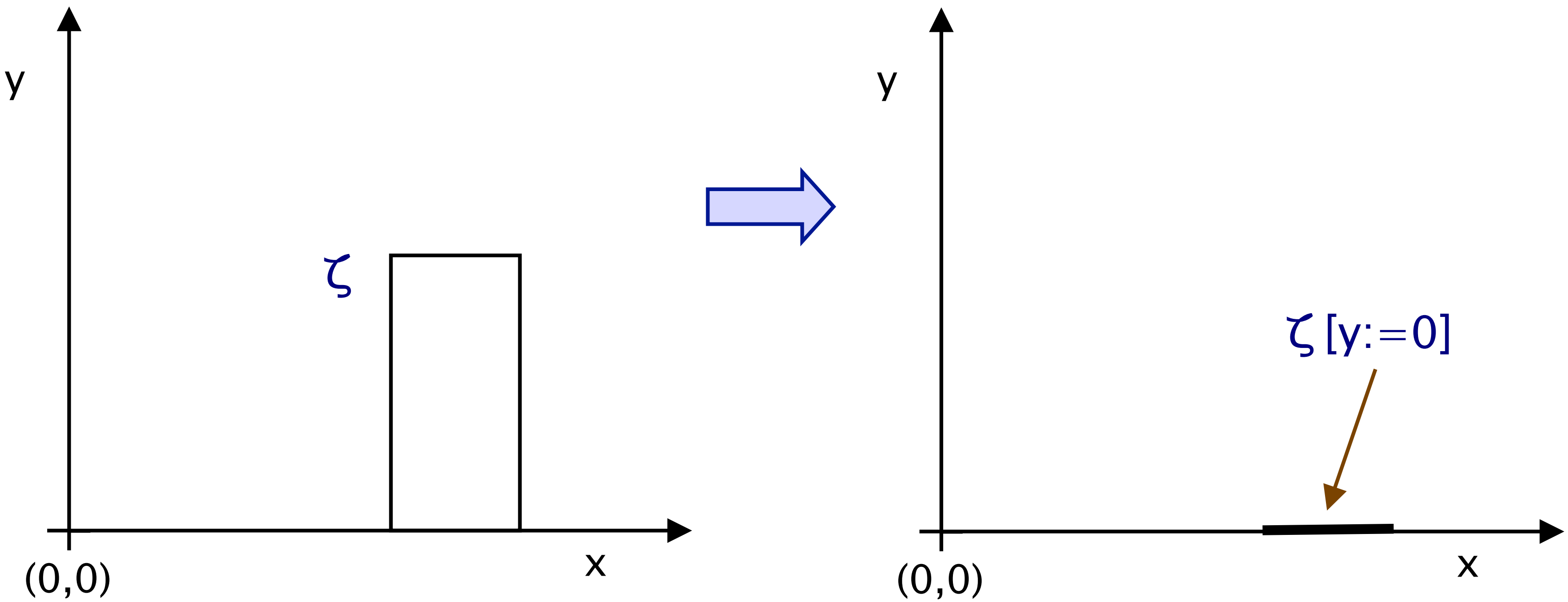
Operations on zones – Set theoretic

- Difference of two zones: $\zeta_1 \setminus \zeta_2$



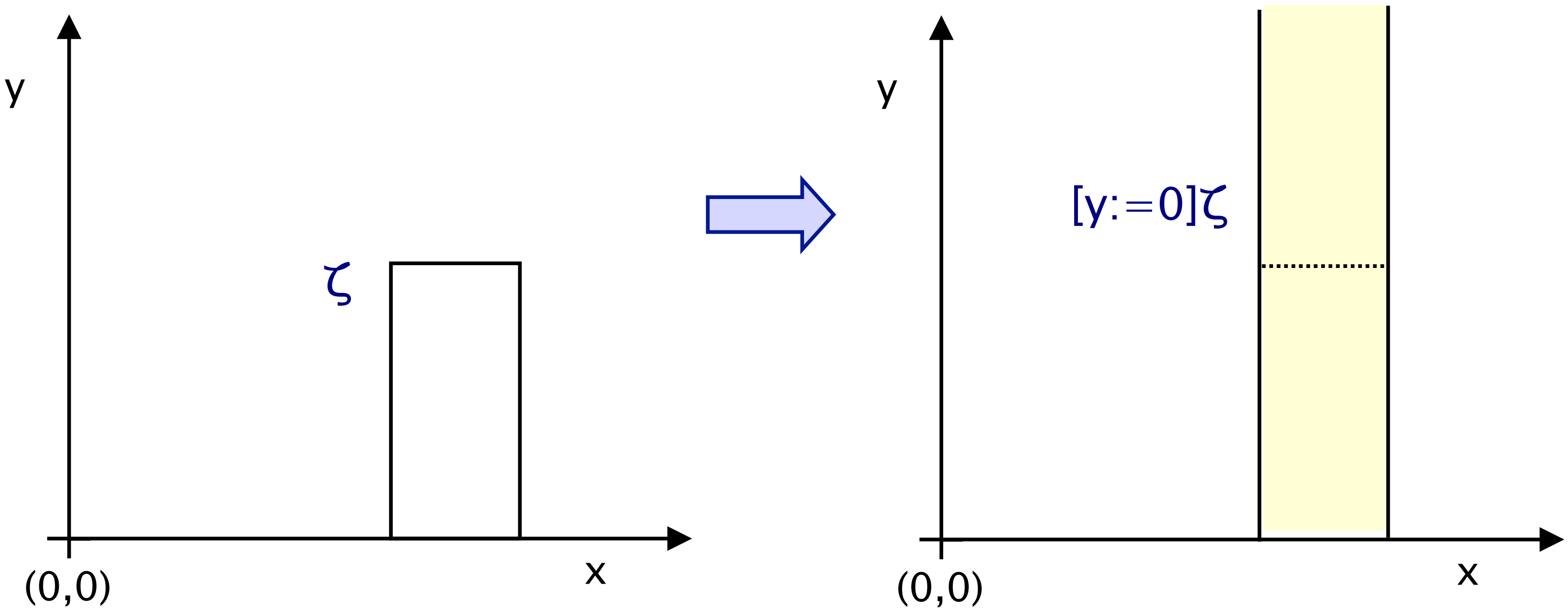
Operations on zones – Clock resets

- $\zeta[Y{:=}0] = \{\, v[Y{:=}0] \mid v \rhd \zeta \,\}$
  - clock valuations obtained from ζ by resetting the clocks in Y



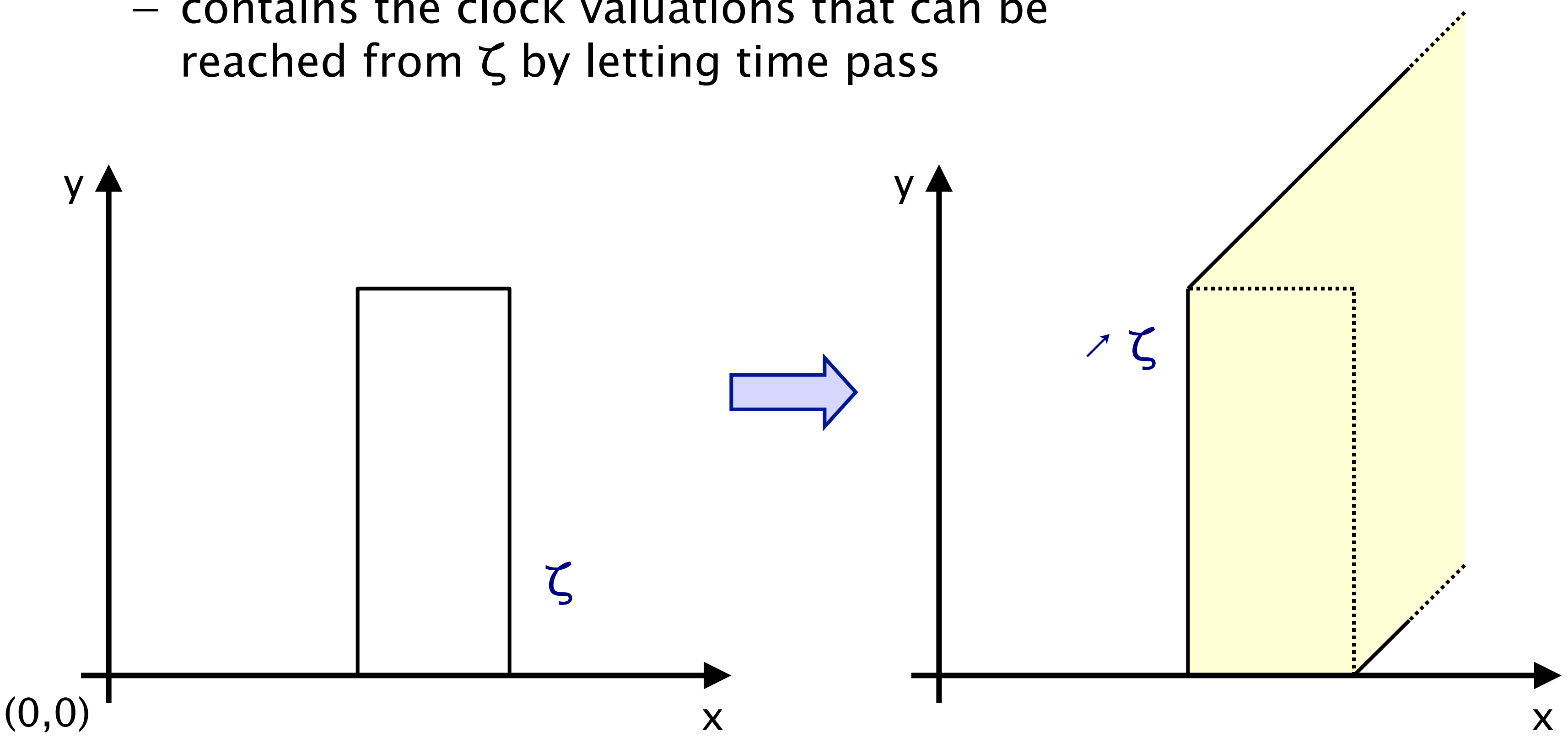
Operations on zones – Clock resets

- $[Y{:=}0]\zeta = \{ v \mid v[Y{:=}0] \triangleright \zeta \}$
  - clock valuations which are in ζ if the clocks in Y are reset



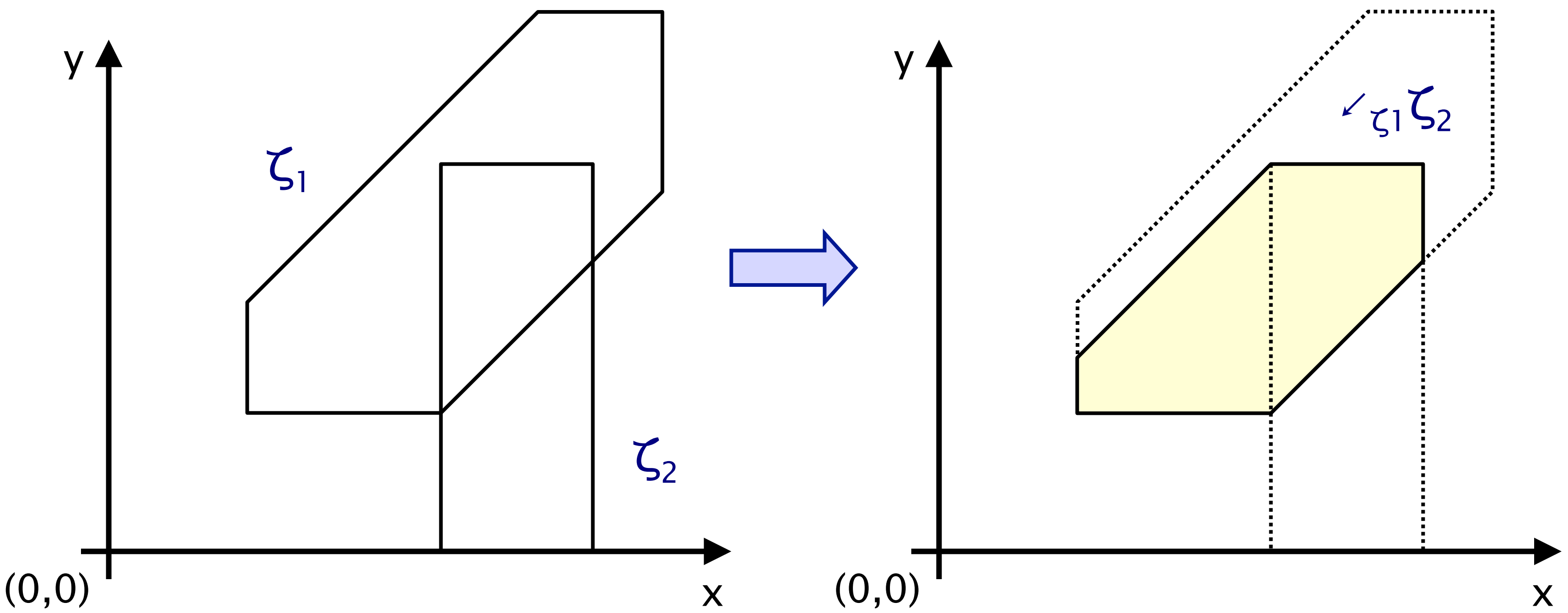
Operations on zones: Projections

- Forwards diagonal projection
- $\nearrow\zeta = \{\ v \mid \exists t\geq 0\ .\ (v-t)\triangleright\zeta\ \}$
  - contains the clock valuations that can be reached from ζ by letting time pass



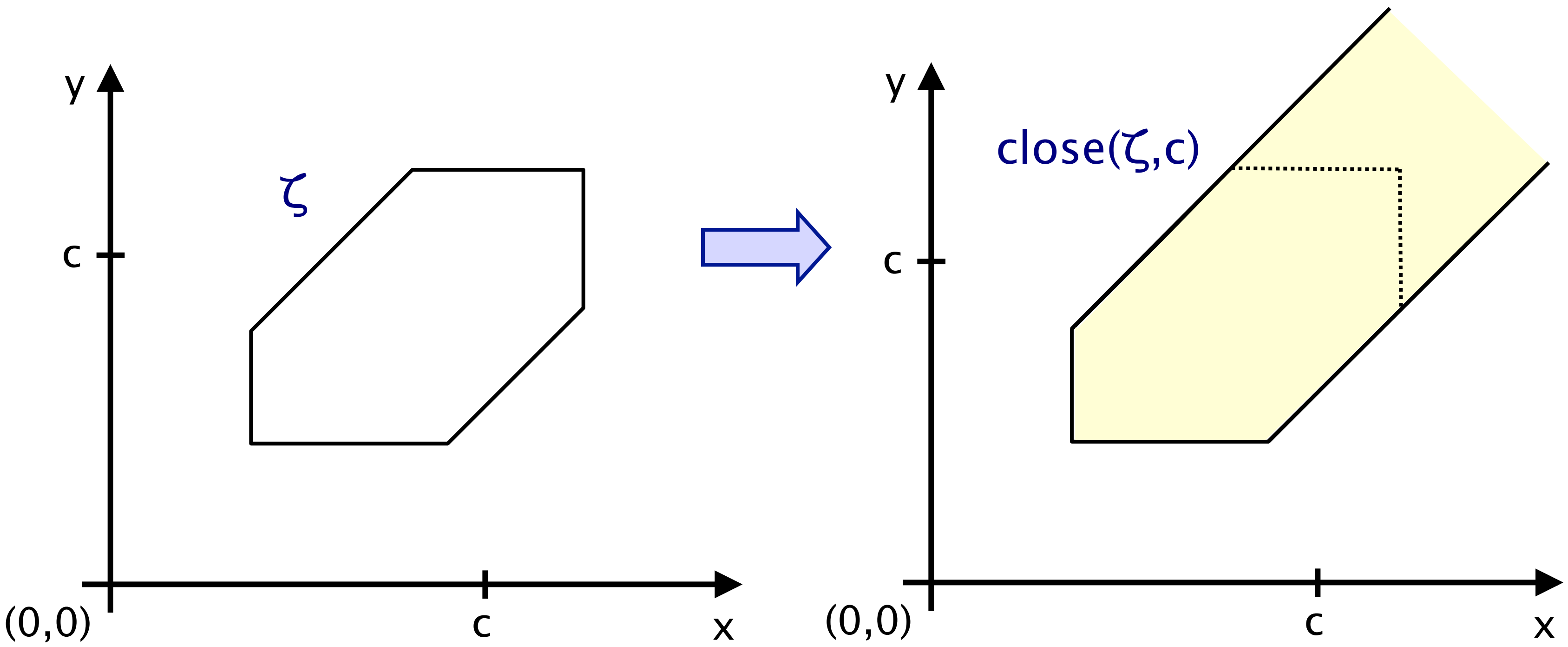
Operations on zones: Projections

- Backwards diagonal projection
- $\swarrow_{\zeta'} \zeta = \{ v \mid \exists t \geq 0 \,.\, ((v+t) \triangleright \zeta \wedge \forall t' < t \,.\, ((v+t') \triangleright \zeta')) \}$
  - contains the clock valuations that, by letting time pass, reach a clock valuation in ζ and remain in ζ' until ζ is reached



Operations on zones: c-closure

- c-closure: close(ζ,c)
  - ignores all constraints which are greater than c



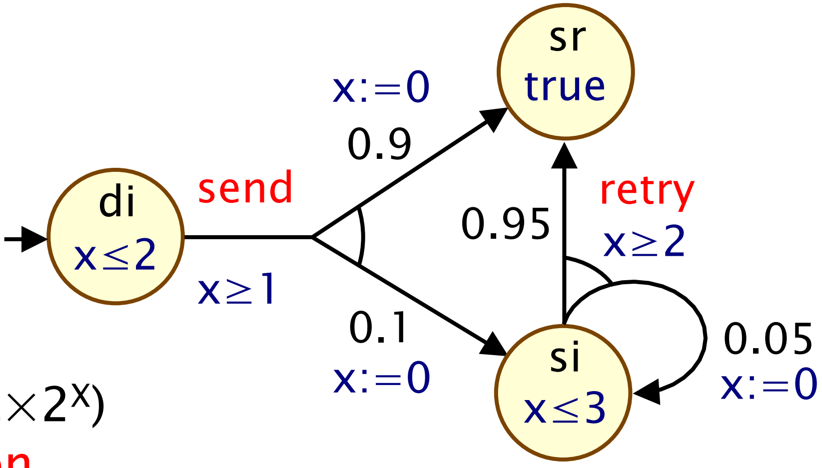
Overview (Part 5)

- Time, clocks and zones
- Probabilistic timed automata (PTAs)
  - definition, examples, semantics, time divergence
- PTCTL: A temporal logic for for PTAs
  - syntax, examples, semantics
- Model checking for PTAs
  - the region graph
  - digital clocks
  - zone-based approaches:
  - (i) forwards reachability
  - (ii) backwards reachability
  - (iii) game-based abstraction refinement
- Costs and rewards

Probabilistic timed automata (PTAs)

- Probabilistic timed automata (PTAs)
  - Markov decision processes (MDPs) + real-valued clocks
  - or: timed automata + discrete probabilistic choice
  - model probabilistic, nondeterministic and timed behaviour

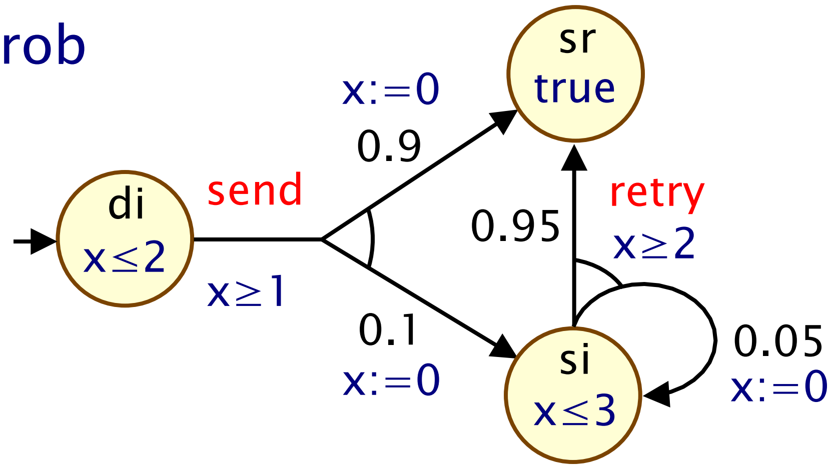
- Syntax: A PTA is a tuple (Loc, l_{init}, Act, X, inv, prob, L)
  - Loc is a finite set of locations
  - $l_{init} \in$ Loc is the initial location
  - Act is a finite set of actions
  - X is a finite set of clocks
  - inv : Loc → Zones(X) is the invariant condition
  - prob ⊆ Loc×Zones(X)×Dist(Loc×2^X) is the probabilistic edge relation
  - L : Loc → AP is a labelling function



Probabilistic edge relation

- Probabilistic edge relation
  - prob $\subseteq$ Loc$\times$Zones(X)$\times$Act$\times$Dist(Loc$\times 2^X$)

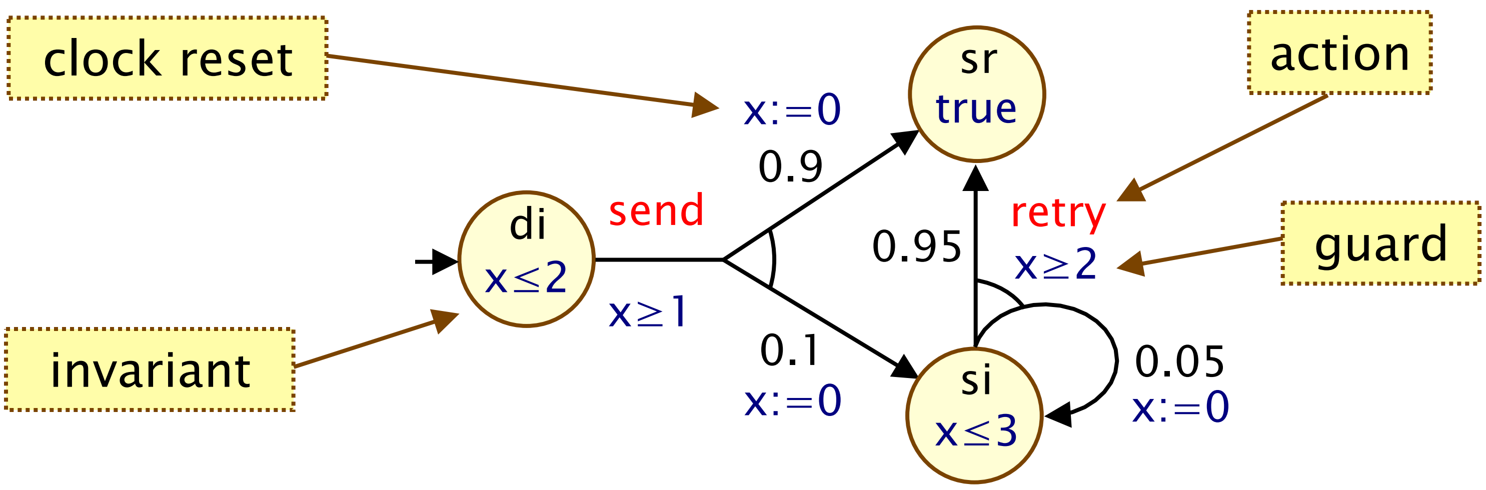
- Probabilistic edge (l,g,a,p) $\in$ prob
  - l is the source location
  - g is the guard
  - a is the action
  - p target distribution



- Edge (l,g,a,p,l',Y)
  - from probabilistic edge (l,g,a,p) where p(l',Y)>0
  - l' is the target location
  - Y is the set of clocks to be reset

PTA – Example

- Models a simple probabilistic communication protocol
  - starts in location di; after between 1 and 2 time units, the protocol attempts to send the data:
    - with probability 0.9 data is sent correctly, move to location sr
    - with probability 0.1 data is lost, move to location si
  - in location si, after 2 to 3 time units, attempts to resend
    - correctly sent with probability 0.95 and lost with probability 0.05

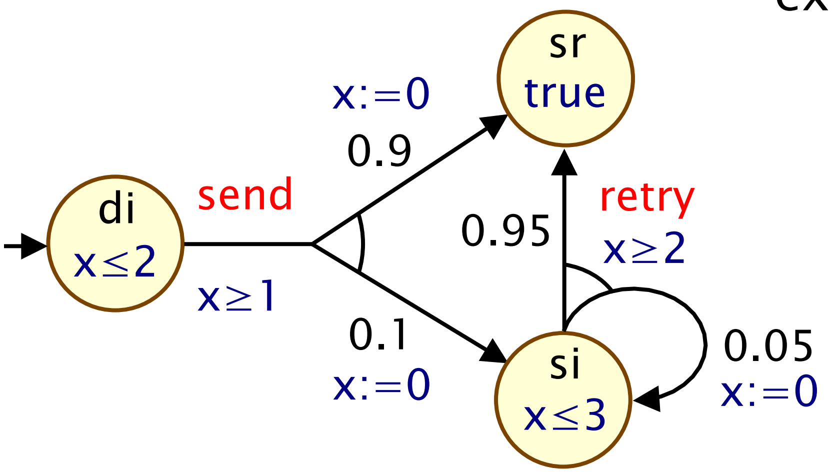


PTAs – Behaviour

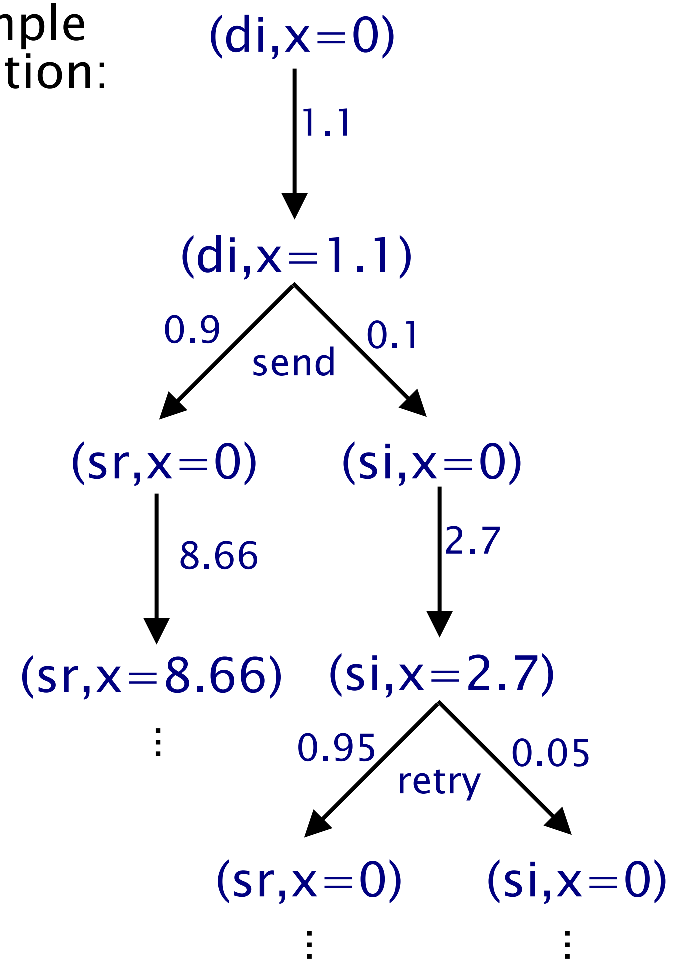
- A state of a PTA is a pair $(l,v) \in Loc \times \mathbb{R}^X$ such that $v \rhd inv(l)$
- A PTAs start in the initial location with all clocks set to zero
  - let $\underline{0}$ denote the clock valuation where all clocks have value 0
- For any state (l,v), there is nondeterministic choice between making a discrete transition and letting time pass
  - discrete transition (l,g,a,p) enabled if $v \rhd g$ and probability of moving to location l' and resetting the clocks Y equals $p(l',Y)$
  - time transition available only if invariant $inv(l)$ is continuously satisfied while time elapses

PTA – Example

PTA:



Example execution:



PTA semantics

- Formally, the semantics of a PTA P is an infinite-state MDP $M_P = (S_P, s_{init}, \mathbf{Steps}, L_P)$ with:
- States: $S_P = \{ (l,v) \in Loc \times \mathbb{R}^X$ such that $v \triangleright inv(l) \}$
- Initial state: $s_{init} = (l_{init}, \underline{0})$

actions of MDP M_P are the actions of PTA P or real time delays

- **Steps**: $S_P \rightarrow 2^{(Act \cup \mathbb{R}) \times Dist(S)}$ such that $(\alpha, \mu) \in \mathbf{Steps}(l,v)$ iff:
  - (time transition) $\alpha = t \in \mathbb{R}$, $\mu(l,v+t)=1$ and $v+t' \triangleright inv(l)$ for all $t' \leq t$
  - (discrete transition) $\alpha = a \in Act$ and there exists $(l,g,a,p) \in prob$

    such that $v \triangleright g$ and, for any $(l',v') \in S_P$: $\mu(l',v') = \sum_{Y \subseteq X \wedge v[Y:=0]=v'} p(l',Y)$

- Labelling: $L_P(l,v) = L(l)$

multiple resets may give same clock valuation

Time divergence

- We restrict our attention to time divergent behaviour
  - a common restriction imposed in real-time systems
  - unrealisable behaviour (i.e. corresponding to time not advancing beyond a time bound) is disregarded
  - also called non-zeno behaviour

- For a path $\omega = s_0(\alpha_0,\mu_0)s_1(\alpha_1,\mu_1)s_2(\alpha_2,\mu_2)\ldots$ in the MDP M_P
  - $D_\omega(n)$ denotes the duration up to state s_n
  - i.e. $D_\omega(n) = \Sigma\ \{|\ \alpha_i \mid 0 \leq i < n \wedge \alpha_i \in \mathbb{R}\ |\}$

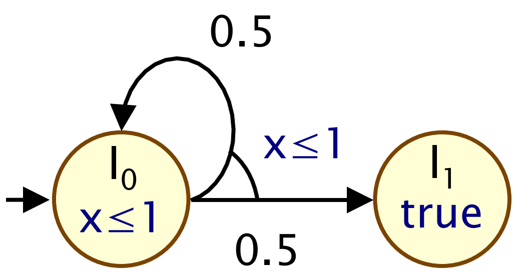
- A path ω is time divergent if, for any $t \in \mathbb{R}_{\geq 0}$:
  - there exists $j \in \mathbb{N}$ such that $D_\omega(j) > t$

- Example of non-divergent path:
  - $s_0(1,\mu_0)s_0(0.5,\mu_1)s_0(0.25,\mu_2)s_0(0.125,\mu_2)s_0\ldots$

Time divergence

- An adversary of M_P is divergent if, for each state $s \in S_P$:
  - the probability of divergent paths under A is 1
  - i.e $Pr^A_s\{ \omega \in Path^A(s) \mid \omega \text{ is divergent} \} = 1$

- Motivation for probabilistic definition of divergence:



  - in this PTA, any adversary has one non-divergent path:
    - takes the loop in l_0 infinitely often, without 1 time unit passing
  - but the probability of such behaviour is 0
  - a stronger notion of divergence would mean no divergent adversaries exist for this PTA

Overview (Part 5)

- Time, clocks and zones
- Probabilistic timed automata (PTAs)
  - definition, examples, semantics, time divergence
- PTCTL: A temporal logic for for PTAs
  - syntax, examples, semantics
- Model checking for PTAs
  - the region graph
  - digital clocks
  - zone-based approaches:
  - (i) forwards reachability
  - (ii) backwards reachability
  - (iii) game-based abstraction refinement
- Costs and rewards

PTCTL – Syntax

- PTCTL: Probabilistic timed computation tree logic
  - derived from PCTL [BdA95] and TCTL [AD94]

- Syntax:

ϕ U ϕ is true with probability ~p

  - $\phi ::= \text{true} \mid a \mid \zeta \mid z.\ \phi \mid \phi \wedge \phi \mid \neg\phi \mid P_{\sim p}\ [\ \phi\ U\ \phi\ ]$

"zone over X∪Z"

"freeze quantifier"

- where:
  - where Z is a set of formula clocks, $\zeta \in$ Zones(X∪Z), $z \in Z$,
  - a is an atomic proposition, $p \in [0,1]$ and $\sim \in \{<,>,\leq,\geq\}$

PTCTL – Examples

- $z \,.\, P_{>0.99}$ [packet2unsent U packet1delivered $\wedge$ $(z<5)$]
  - "with probability greater than 0.99, the system delivers packet 1 within 5 time units and does not try to send packet 2 in the meantime"

- $z \,.\, P_{>0.95}$[$(x\leq3)$ U $(z=8)$]
  - "with probability at least 0.95, the system clock x does not exceed 3 before 8 time units elapse"

- $z \,.\, P_{\leq0.1}$[G (failure $\vee$ $(z\leq60)$)]
  - "the system fails after the first 60 time units have elapsed with probability at most 0.01"

PTCTL – Semantics

- Let $(l,v) \in S_p$ and $\mathcal{E} \in \mathbb{R}^Z$ be a formula clock valuation

combined clock valuation of v and $\mathcal{E}$ satisfies ζ

after resetting z, ϕ is satisfied

  - $(l,v),\mathcal{E} \models a \quad \Leftrightarrow \quad a \in L(l,v)$
  - $(l,v),\mathcal{E} \models \zeta \quad \Leftrightarrow \quad v,\mathcal{E} \triangleright \zeta$
  - $(l,v),\mathcal{E} \models z.\phi \quad \Leftrightarrow \quad (l,v),\mathcal{E}[z:=0] \models \phi$
  - $(l,v),\mathcal{E} \models \phi_1 \wedge \phi_2 \quad \Leftrightarrow \quad (l,v),\mathcal{E} \models \phi_1$ and $(l,v),\mathcal{E} \models \phi_2$
  - $(l,v),\mathcal{E} \models \neg\phi \quad \Leftrightarrow \quad (l,v),\mathcal{E} \models \phi$ is false
  - $(l,v),\mathcal{E} \models P_{\sim p}[\psi] \quad \Leftrightarrow \quad Pr^A_{(l,v)}\{ \omega \in Path^A(l,v) \mid \omega,\mathcal{E} \models \psi \} \sim p$ for all adversaries $A \in Adv_{M_P}$

the probability of a path satisfying ψ meets ~p for all divergent adversaries

PTCTL – Semantics of until

- Let ω be a path in M_P and $\mathcal{E}$ be a formula clock valuation
  - $\omega,\mathcal{E} \models \psi$ satisfaction of ψ by ω, assuming $\mathcal{E}$ initially

- $\omega,\mathcal{E} \models \phi_1 \ \mathrm{U} \ \phi_2$ if and only if
  there exists $i \in \mathbb{N}$ and $t \in D_\omega(i+1)-D_\omega(i)$ such that
  - $\omega(i)+t,\mathcal{E}+(D_\omega(i)+t) \models \phi_2$
  - $\forall \ t' \leq t \ . \ \omega(i)+t',\mathcal{E}+(D_\omega(i)+t') \models \phi_1 \vee \phi_2$
  - $\forall \ j<i \ . \ \forall \ t' \leq D_\omega(j+1)-D_\omega(j) \ . \ \omega(j)+t',\mathcal{E}+(D_\omega(j)+t') \models \phi_1 \vee \phi_2$

- Condition "$\phi_1 \vee \phi_2$" different from PCTL and CSL
  - usually ϕ_2 becomes true and ϕ_1 is true until this point
  - difference due to the density of the time domain
  - to allow for open intervals use disjunction $\phi_1 \vee \phi_2$
  - for example consider $x \leq 5 \ \mathrm{U} \ x > 5$ and $x < 5 \ \mathrm{U} \ x \geq 5$

Probabilistic reachability in PTAs

- For simplicity, in some cases, we just consider probabilistic reachability, rather than full PTCTL model checking
  - i.e. min/max probability of reaching a set of target locations
  - can also encode time-bounded reachability (with extra clock)

- Still captures a wide range of properties
  - probabilistic reachability: "with probability at least 0.999, a data packet is correctly delivered"
  - probabilistic invariance: "with probability 0.875 or greater, the system never aborts"
  - probabilistic time-bounded reachability: "with probability 0.01 or less, a data packet is lost within 5 time units"
  - bounded response: "with probability 0.99 or greater, a data packet will always be delivered within 5 time units"

Overview (Part 5)

- Time, clocks and zones
- Probabilistic timed automata (PTAs)
  - definition, examples, semantics, time divergence
- PTCTL: A temporal logic for for PTAs
  - syntax, examples, semantics
- Model checking for PTAs
  - the region graph
  - digital clocks
  - zone-based approaches:
  - (i) forwards reachability
  - (ii) backwards reachability
  - (iii) game-based abstraction refinement
- Costs and rewards

PTA model checking – Summary

- Several different approaches developed
  - basic idea: reduce to the analysis of a finite-state model
  - in most cases, this is a Markov decision process (MDP)

- Region graph construction [KNSS02]
  - shows decidability, but gives exponential complexity
- Digital clocks approach [KNPS06]
  - (slightly) restricted classes of PTAs
  - works well in practice, still some scalability limitations
- Zone-based approaches:
  - (preferred approach for non-probabilistic timed automata)
  - forwards reachability [KNSS02]
  - backwards reachability [KNSW07]
  - game-based abstraction refinement [KNP09c]

The region graph

- Region graph construction for PTAs [KNSS02]
  - adapts region graph construction for timed automata [ACD93]
  - partitions PTA states into a finite set of regions
  - based on notion of clock equivalence
  - construction is also dependent on PTCTL formula

- For a PTA P and PTCTL formula ϕ
  - construct a time-abstract, finite-state MDP $R(\phi)$
  - translate PTCTL formula ϕ to PCTL formula ϕ'
  - ϕ is preserved by region quivalence
  - i.e. ϕ holds in a state of M_P if and only if ϕ' holds in the corresponding state of $R(\phi)$
  - model check $R(\phi)$ using standard methods for MDPs

The region graph – Clock equivalence

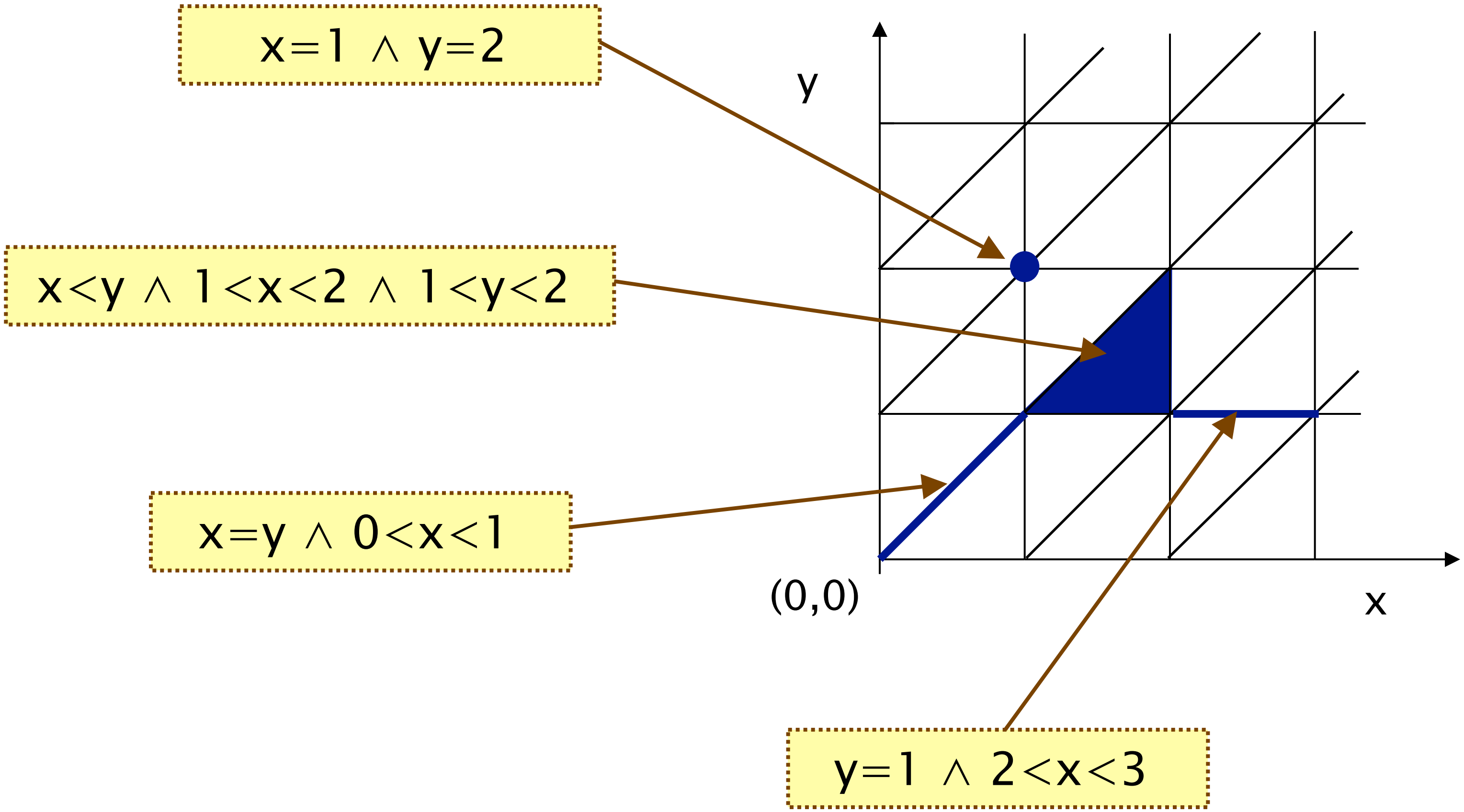
- Regions are sets of clock equivalent clock valuations

- Some notation:
  - let c be largest constant appearing in PTA or PTCTL formula
  - let $\lfloor t \rfloor$ denotes the integral part of t
  - t and t' agree on their integral parts if and only if
    (1) $\lfloor t \rfloor = \lfloor t' \rfloor$
    (2) t and t' are both integers or neither is an integer

- The clock valuations v and v' are clock equivalent ($v \cong v'$) if:
  - for all clocks $x \in X$, either:
    - $v(x)$ and $v'(x)$ agree on their integral parts
    - $v(x)>c$ and $v'(x)>c$
  - for all clock pairs $x,y \in X$, either:
    - $v(x) - v(x')$ and $v'(x) - v'(x')$ agree on their integral parts
    - $v(x) - v(x') > c$ and $v'(x) - v'(x') > c$

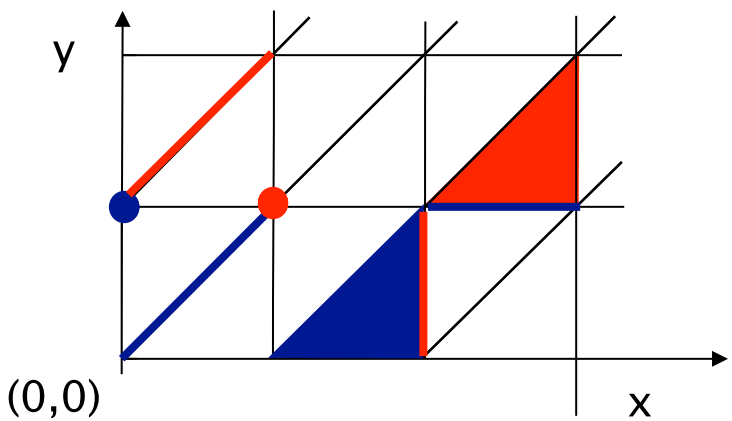
Region graph – Clock equivalence

- Example regions (for 2 clocks x and y)



Region graph – Clock equivalence

- Fundamental result: if $v \cong v'$, then $v \rhd \zeta \Leftrightarrow v' \rhd \zeta$
  - it follows that $r \rhd \zeta$ is well defined for a region r
- r' is the successor region of r, written succ(r) = r', if
  - for each $v \in r$, there exists $t > 0$ such that $v + t \in r'$ and $v + t' \in r \cup r'$ for all $t' < t$



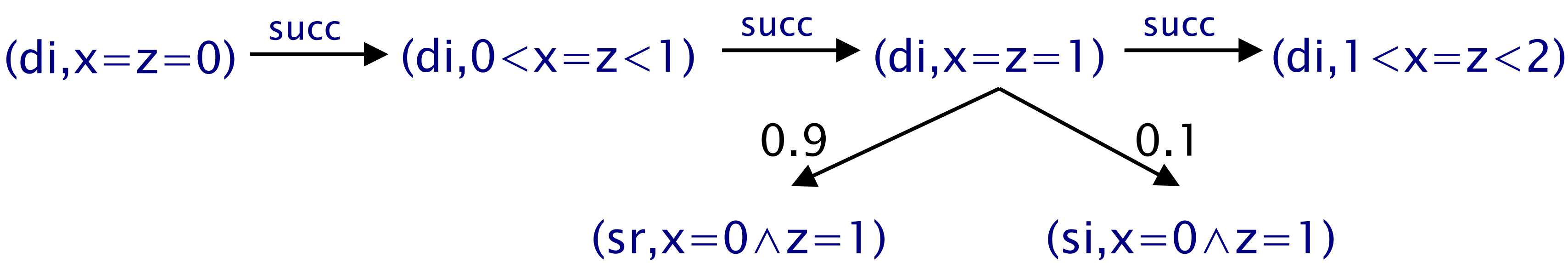
The region graph

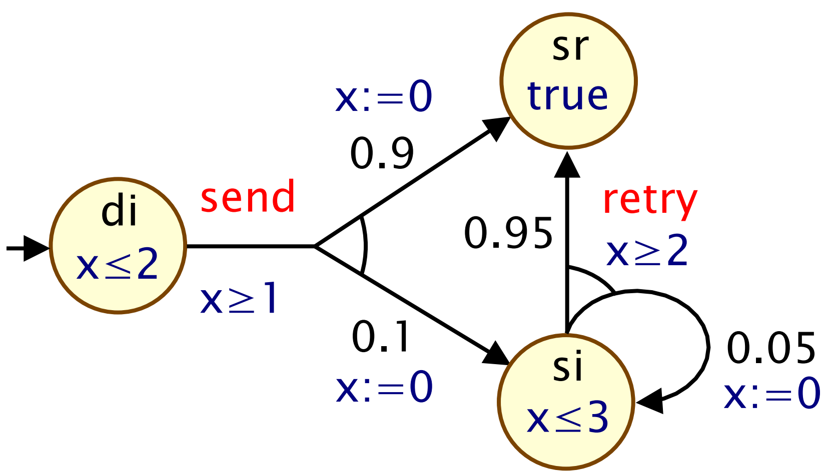
- The region graph MDP is $(S_R, s_{init}, \mathbf{Steps}_R, L_R)$ where…
  - the set of states S_R comprises pairs (l,r) such that l is a location and r is a region over $X \cup Z$
  - the initial state is $(l_{init}, \underline{0})$
  - the set of actions is $\{succ\} \cup Act$
    - succ is a unique action denoting passage of time
  - the probabilistic transition function $\mathbf{Steps}_R$ is defined as:
  - $S_R \times 2^{(\{succ\} \cup Act) \times Dist(S_R)}$
  - $(succ,\mu) \in \mathbf{Steps}_R(l,r)$ iff $\mu(l,succ(r))=1$
  - $(a,\mu) \in \mathbf{Steps}_R(l,r)$ if and only if $\exists\ (l,g,a,p) \in prob$ such that

    $r \triangleright g$ and, for any $(l',r') \in S_R$: $\mu(l',r') = \sum_{Y \subseteq X \wedge r[Y:=0]=r'} p(l',Y)$
  - the labelling is given by: $L_R(l,r) = L(l)$

Region graph – Example

- PTCTL formula: $z.P_{\sim p}$ [true U (sr<4)]





Region graph construction

- Region graph
  - useful for establishing decidability of model checking
  - or proving complexity results for model checking algorithms

- But…
  - the number of regions is exponential in the number of clocks and the size of largest constant
  - so model checking based on this is extremely expensive
  - and so not implemented (even for timed automata)

- Improved approaches based on:
  - digital clocks
  - zones (unions of regions)

Overview (Part 5)

- Time, clocks and zones
- Probabilistic timed automata (PTAs)
  - definition, examples, semantics, time divergence
- PTCTL: A temporal logic for for PTAs
  - syntax, examples, semantics
- Model checking for PTAs
  - the region graph
  - digital clocks
  - zone-based approaches:
  - (i) forwards reachability
  - (ii) backwards reachability
  - (iii) game-based abstraction refinement
- Costs and rewards

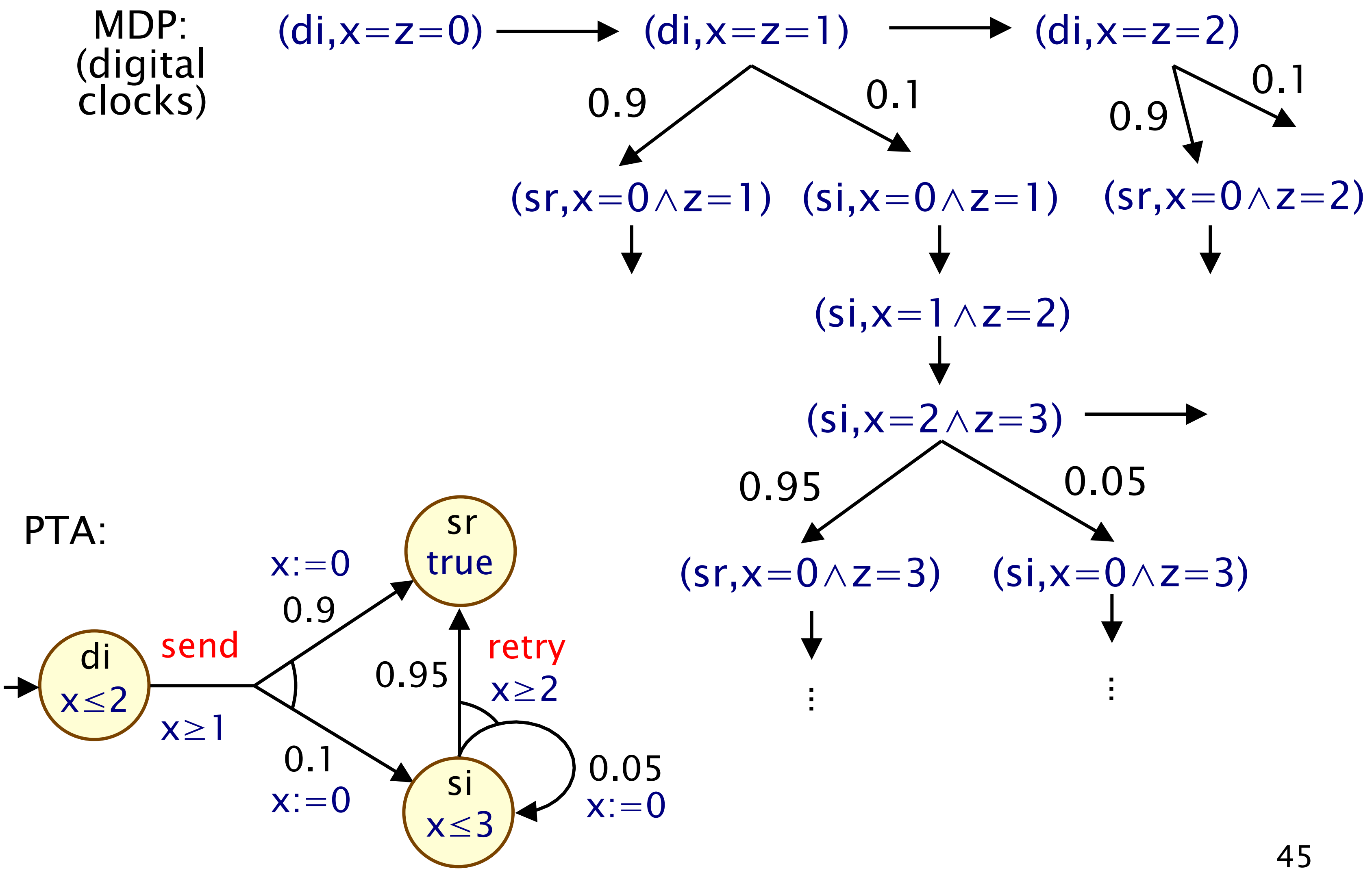
Digital clocks

- Simple idea: Clocks can only take integer (digital) values
  - i.e. time domain is $\mathbb{N}$ as opposed to $\mathbb{R}$
  - based on notion of ϵ-digitisation [HMP92]

- Only applies to arestricted class of PTAs; zones must be:
  - closed – no strict inequalities (e.g. $x>5$)

- Digital clocks semantics yields a finite-state MDP
  - state space is a subset of $\mathrm{Loc} \times \mathbb{N}^X$, rather than $\mathrm{Loc} \times \mathbb{R}^X$
  - clocks bounded by c_{max} (max constant in PTA and formula)
  - then use standard techniques for finite -state MDPs

Example – Digital clocks



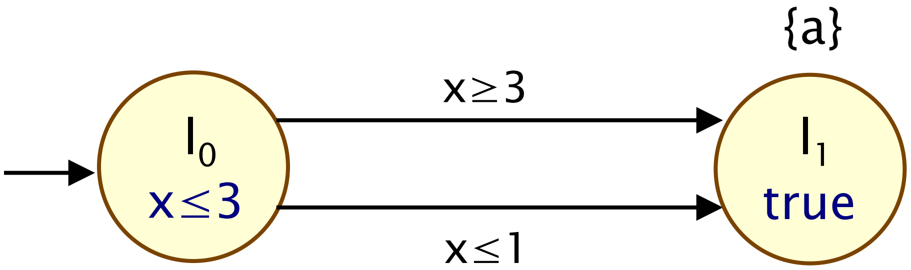
Digital clocks

- Digital clocks approach preserves:
  - minimum/maximum reachability probabilities
  - a subset of PTCTL properties
  - (no nesting, only closed zones in formulae)
  - only works for the initial state of the PTA
  - (but can be extended to any state with integer clock values)

- In practice:
  - translation from PTA to MDP can often be done manually
  - (by encoding the PTA directly into the PRISM language)
  - automated translations exist: mcpta and (soon) PRISM
  - many case studies, despite “closed” restriction

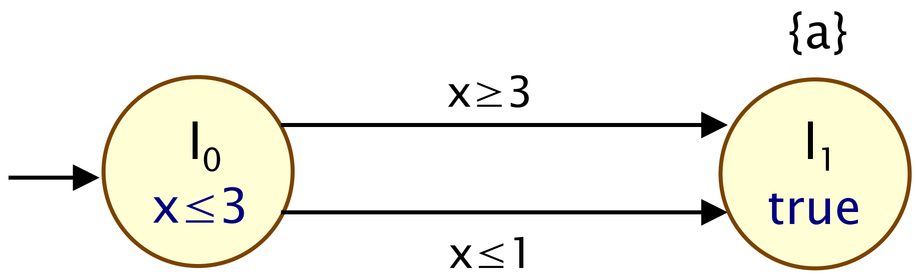
- Problem: can lead to very large MDPs
  - alleviated partially by efficient symbolic model checking

Digital clocks do not preserve PTCTL



- Consider the PTCTL formula $\phi = z.P_{<1}$ [true U (a $\wedge$ $z \leq 1$)]
  - a is an atomic proposition only true in location l_1
- Digital semantics:
  - no state satisfies ϕ since for any state we have $Prob^A(s, \mathcal{E}[z:=0]$, true U $(a \wedge z \leq 1)$) = 1 for some adversary A
  - hence $P_{<1}$ [true U ϕ] is trivially true in all states

Digital clocks do not preserve PTCTL



- Consider the PTCTL formula $\phi = z.P_{<1}$ [true U (a $\wedge$ $z \leq 1$)]
  - a is an atomic proposition only true in location l_1
- Dense time semantics:
  - any state (l_0, v) where $v(x) \in (1,2)$ satisfies ϕ
    more than one time unit must pass before we can reach l_1
  - hence $P_{<1}$ [true U ϕ] is not true in the initial state

Overview (Part 5)

- Time, clocks and zones
- Probabilistic timed automata (PTAs)
  - definition, examples, semantics, time divergence
- PTCTL: A temporal logic for for PTAs
  - syntax, examples, semantics
- Model checking for PTAs
  - the region graph
  - digital clocks
  - zone-based approaches:
  - (i) forwards reachability
  - (ii) backwards reachability
  - (iii) game-based abstraction refinement
- Costs and rewards

Zone-based approaches

- An alternative is to use zones to construct an MDP

- Conventional symbolic model checking relies on computing
  - **post**(S’) the states that can be reached from a state in S’ in a single step
  - **pre**(S’) the states that can reach S’ in a single step

- Extend these operators to include time passage
  - **dpost**[e](S’) the states that can be reached from a state in S’ by traversing the edge e
  - **tpost**(S’) the states that can be reached from a state in S’ by letting time elapse
  - **pre**[e](S’) the states that can reach S’ by traversing the edge e
  - **tpre**(S’) the states that can reach S’ by letting time elapse

Zone-based approaches

- Symbolic states (l, ζ) where
  - $l \in Loc$ (location)
  - ζ is a zone over PTA clocks and formula clocks
  - generally fewer zones than regions

- **tpost**$(l,\zeta) = (l, \nearrow\zeta \wedge inv(l))$
  - $\nearrow\zeta$ can be reached from ζ by letting time pass
  - $\nearrow\zeta \wedge inv(l)$ must satisfy the invariant of the location l

- **tpre**$(l,\zeta) = (l, \swarrow\zeta \wedge inv(l))$
  - $\swarrow\zeta$ can reach ζ by letting time pass
  - $\swarrow\zeta \wedge inv(l)$ must satisfy the invariant of the location l

Zone-based approaches

- For an edge $e = (l,g,a,p,l',Y)$ where
  - l is the source
  - g is the guard
  - a is the action
  - l' is the target
  - Y is the clock reset

- **dpost**$[e](l,\zeta) = (l', (\zeta \wedge g)[Y:=0]\ )$
  - $\zeta \wedge g$ satisfy the guard of the edge
  - $(\zeta \wedge g)[Y:=0]$ reset the clocks Y

- **dpre**$[e](l',\zeta') = (l,\ \ [Y:=0]\zeta' \wedge (g \wedge inv(l))\ )$
  - $[Y:=0]\zeta'$ the clocks Y were reset
  - $[Y:=0]\zeta' \wedge (g \wedge inv(l))$ satisfied guard and invariant of l

Forwards reachability

- Based on the operation **post**[e](l,ζ) = **tpost**(**dpost**[e](l,ζ))
  - (l',v') ∈ **post**[e](l,ζ) if there exists (l,v) ∈ (l,ζ) such that after traversing edge e and letting time pass one can reach (l',v')

- Forwards algorithm (part 1)
  - start with initial state S_F={**tpost**((l_{init},$\underline{0}$))} then iterate

    for each symbolic state (l,ζ) ∈ S_F and edge e

    add **post**[e](l,ζ) to S_F
  - until set of symbolic states S_F does not change

- To ensure termination need to take c-closure of each zone encountered (c is the largest constant in the PTA)

Forwards reachability

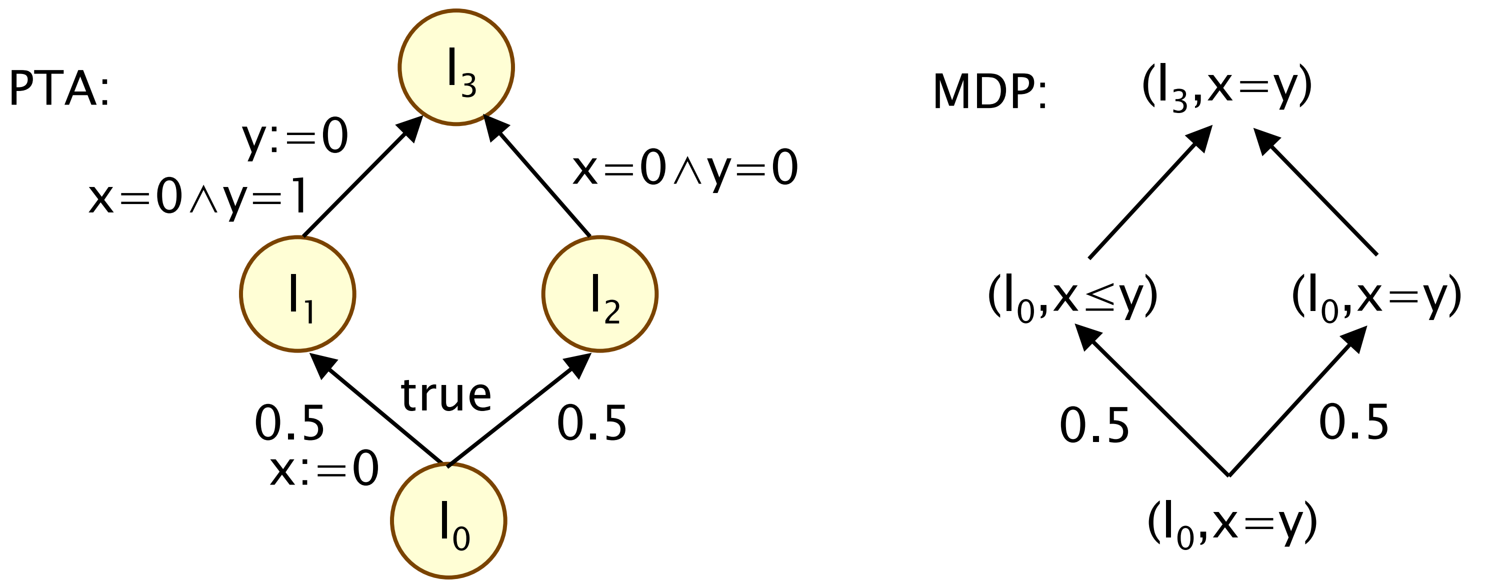
- Forwards algorithm (part 2)
  - construct finite state MDP $(S_F,(l_{init},\underline{0}),\mathbf{Steps}_F,L_F)$
  - states S_F (returned from first part of the algorithm)
  - $L_F(l,\zeta)=L(l)$ for all $(l,\zeta) \in S_F$
  - $\mu \in \mathbf{Steps}_F(l,\zeta)$ if and only if

    there exists a probabilistic edge (l,g,a,p) of PTA such that for any $(l', \zeta') \in Z$:

$$\mu(l',\zeta') = \sum \{\!|\, p(l',X) \mid (l,g,\sigma,p,l',X) \in \mathrm{edges}(p) \wedge \mathrm{post}[e](l,\zeta) = (l',\zeta') \,|\!\}$$

summation over all the edges of (l,g,a,p) such that applying **post** to (l,ζ) leads to the symbolic state (l',ζ')

Forwards reachability – Example



Forwards reachability – Limitations

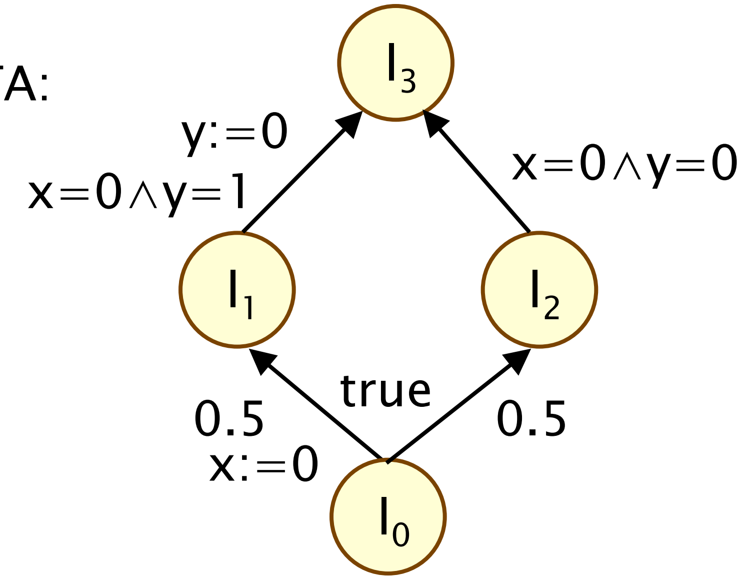
- Only obtain upper bounds on maximum probabilities
  - caused by when edges are combined

- Suppose **post**$[e_1](l,\zeta)=(l_1,\zeta_1)$ and **post**$[e_2](l,\zeta)=(l_2, \zeta_2)$
  - where e_1 and e_2 from the same probabilistic edge
- By definition of **post**
  - there exists $(l,v_i) \in (l,\zeta)$ such that a state in (l_i, ζ_i) can be reached by traversing the edge e_i and letting time pass
- Problem
  - we combine these transitions but are (l,v_1) and (l,v_2) the same?
  - may not exist states in (l,ζ) for which both edges are enabled

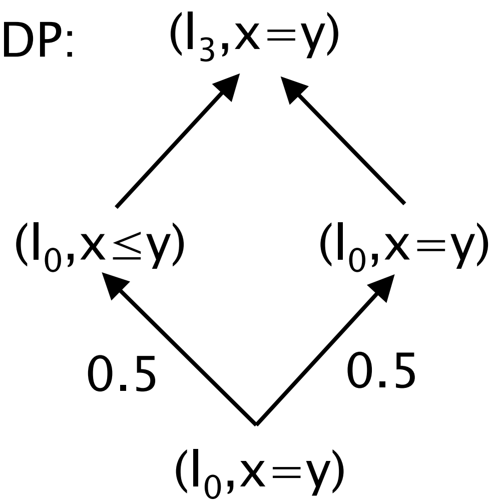
Forwards reachability – Example

- Maximum probability of reaching l_3 is 0.5 in the PTA
  - for the left branch need to take the first transition when $x=1$
  - for the right branch need to take the first transition when $x=0$
- However, in the forwards reachability graph probability is 1
  - can reach l_3 via either branch from $(l_0,x=y)$

PTA:



MDP:



Forwards reachability

- Main result [KNSS02]
  - obtain time-abstract, finite-state MDP over zones
  - bound on maximum reachability probabilities only
  - can model check the MDP using standard methods
  - loss of on-the fly, must construct MDP first

- Implementations
  - KRONOS pre-processor into PRISM input language, outputs time-abstract MDP [DKN02]
  - Explicit, using Difference Bound Matrices (DBMs), to PRISM input language [WK05]
  - Symbolic, using Difference Decision Diagrams (DDDs), via MTBDD-coded PTA syntax directly to PRISM engine [WK05]

Backwards reachability

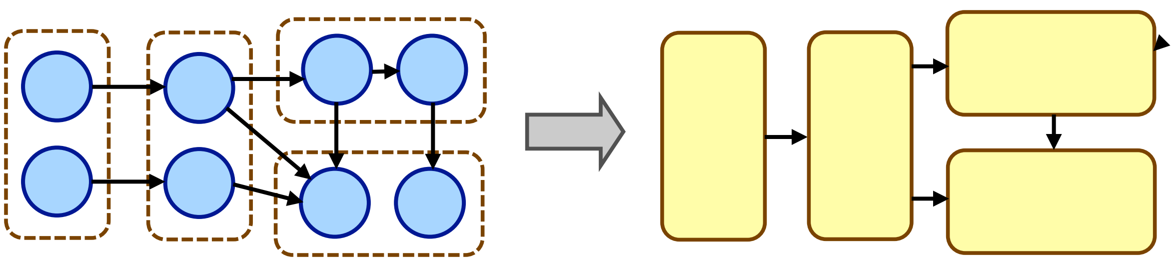
- An alternative zone-based method: backwards reachability
  - state-space exploration in opposite direction, from target to initial states; uses **pre** rather than **post** operator
- Basic ideas: (see [KNSW07] for details)
  - construct a finite-state MDP comprising symbolic states
  - need to keep track of branching structure and take conjunctions of symbolic states if necessary
  - MDP yields maximum reachability probabilities for PTA
  - for min. probs, do graph-based analysis and convert to max.
- Advantages:
  - gives (exact) minimum/maximum reachability probabilities
  - extends to full PTCTL model checking
- Disadvantage:
  - operations to implement are expensive, limits applicability
  - (requires manipulation of non-convex zones)

Overview (Part 5)

- Time, clocks and zones
- Probabilistic timed automata (PTAs)
  - definition, examples, semantics, time divergence
- PTCTL: A temporal logic for for PTAs
  - syntax, examples, semantics
- Model checking for PTAs
  - the region graph
  - digital clocks
  - zone-based approaches:
  - (i) forwards reachability
  - (ii) backwards reachability
  - (iii) game-based abstraction refinement
- Costs and rewards

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

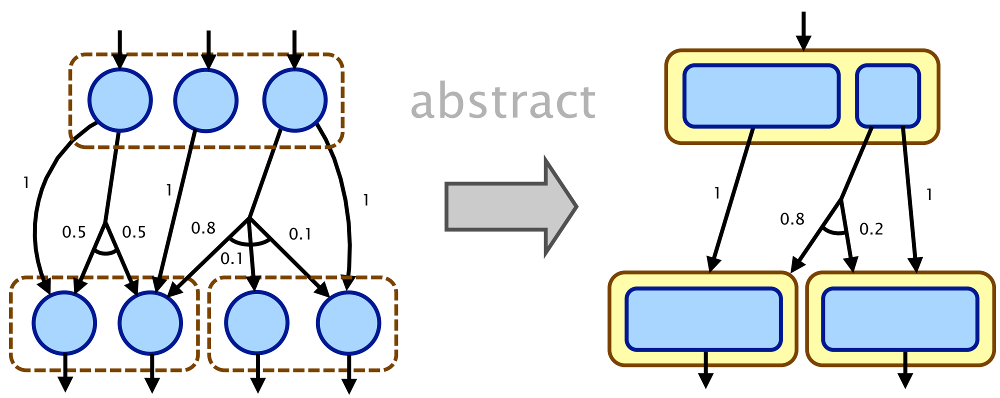


- Automatic generation of abstractions using refinement
  - start with a simple coarse abstraction; iteratively refine

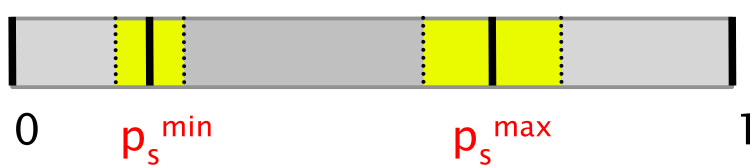
Abstraction of MDPs

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

- We construct abstractions of MDPs using stochastic games

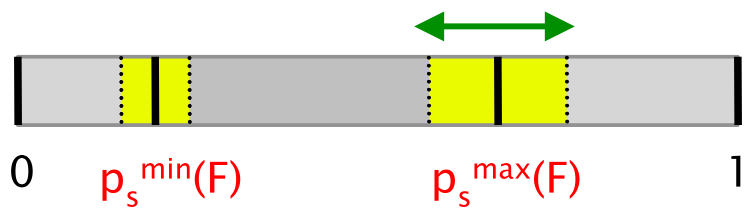


  - yields lower/upper bounds for min/max probabilities



Abstraction refinement

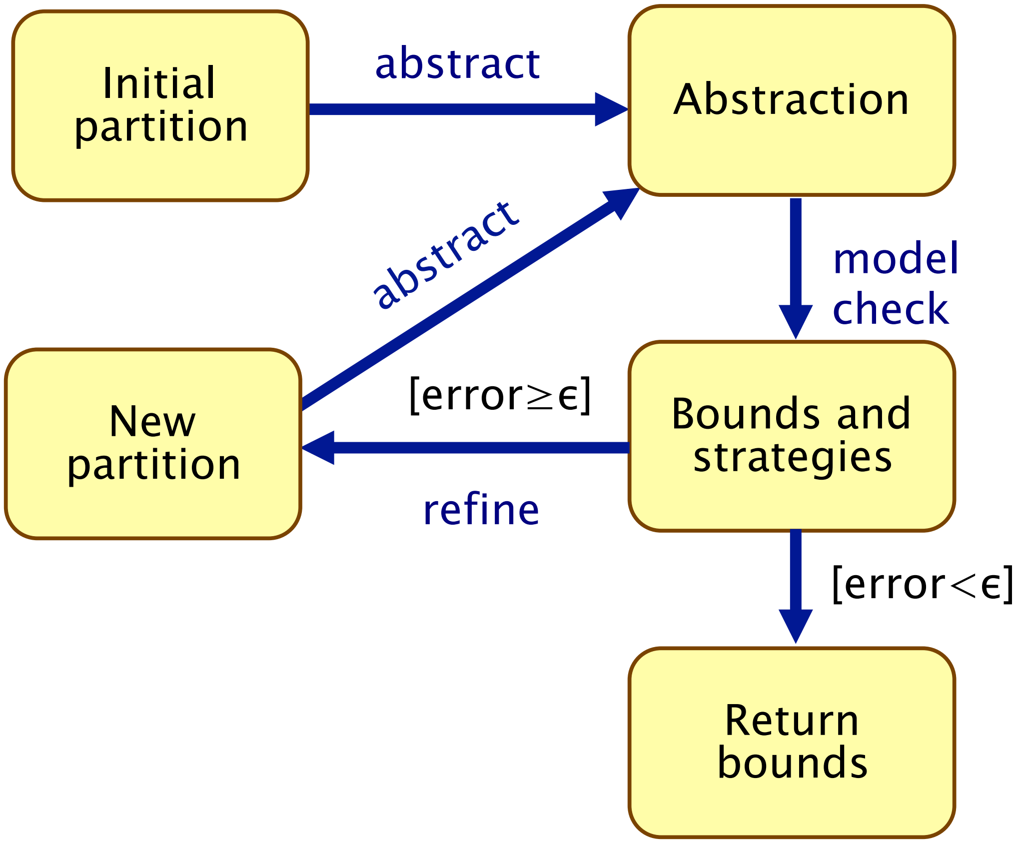
- Consider (max) difference between lower/upper bounds
  - gives a quantitative measure of the abstraction's precision



- 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

Abstraction-refinement loop

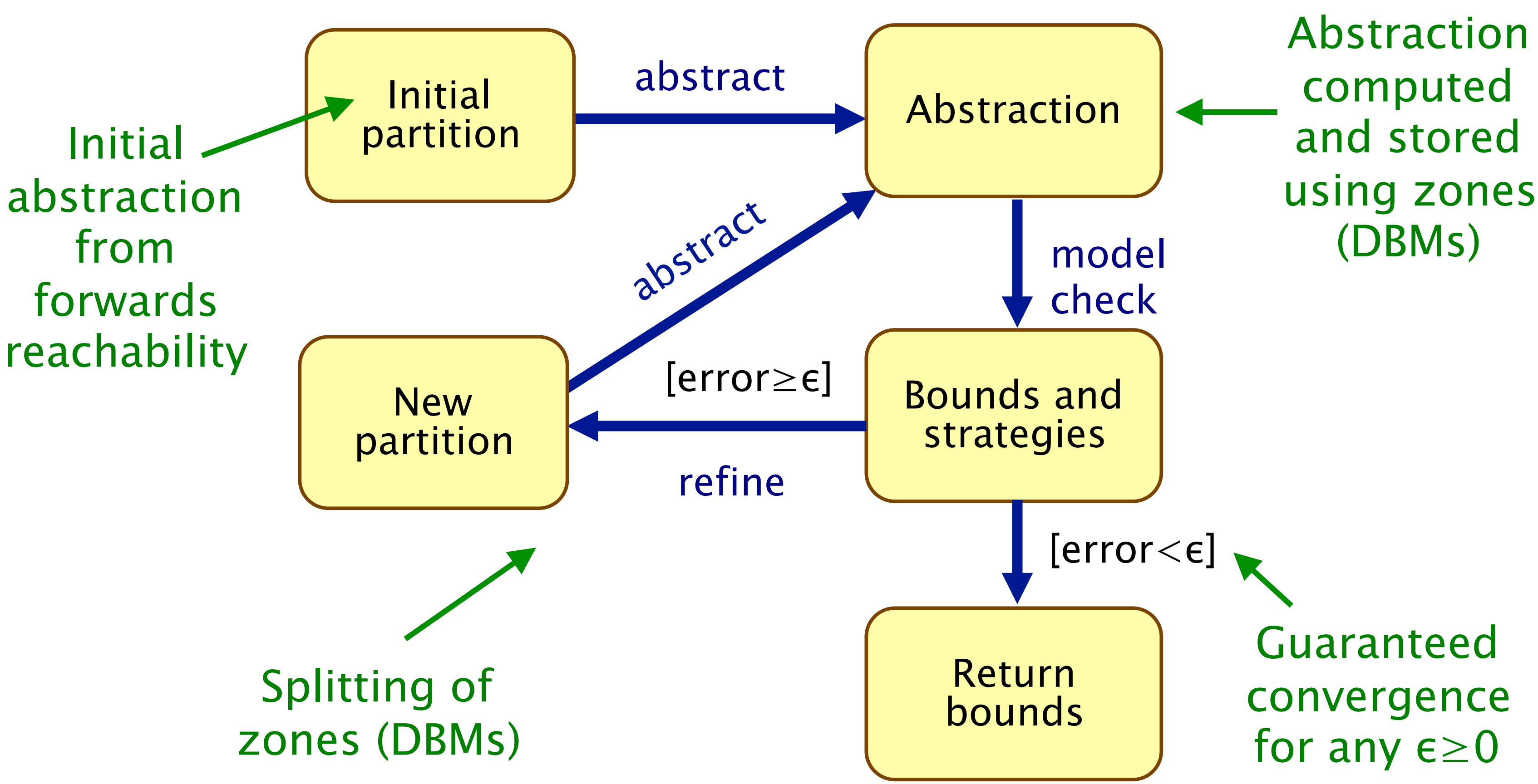
- Quantitative abstraction-refinement loop for MDPs



- Refinements yield strictly finer partition
- Guaranteed to converge for finite models
- Guaranteed to converge for infinite models with finite bisimulation

Abstraction refinement for PTAs

- Model checking for PTAs using abstraction refinement



Abstraction refinement for PTAs

- Computes reachability probabilities in PTAs
  - minimum or maximum, exact values (“error” ϵ=0)
  - also time-bounded reachability, with extra clock
- Integrated in PRISM (next release)
  - PRISM modelling language extended with clocks
  - implemented using DBMs
- In practice performs, performs very well
  - faster than digital clocks or backwards on large example set
  - (sometimes by several orders of magnitude)
  - handles larger PTAs than the digital clocks approach

Overview (Part 5)

- Time, clocks and zones
- Probabilistic timed automata (PTAs)
  - definition, examples, semantics, time divergence
- PTCTL: A temporal logic for for PTAs
  - syntax, examples, semantics
- Model checking for PTAs
  - the region graph
  - digital clocks
  - zone-based approaches:
  - (i) forwards reachability
  - (ii) backwards reachability
  - (iii) game-based abstraction refinement
- Costs and rewards

Costs and rewards

- Like other models, we can define a reward structure $(\underline{\rho},\iota)$ for a probabilistic timed automaton

- $\underline{\rho}$: Loc $\rightarrow \mathbb{R}_{\geq 0}$ location reward function
  - $\underline{\rho}$(l) is the rate at which the reward is accumulated in location l_
- ι : Act $\rightarrow \mathbb{R}_{\geq 0}$ action reward function
  - ι(a) is the reward associated with performing the action a

- Generalises notion for uniformly priced timed automata

- A useful special case is the elapsed time
  - $\underline{\rho}$(l)=1 for all locations l $\in$ Loc
  - ι(a)=0 for all actions a $\in$ Act

Expected reachability

- Expected reachability:
  - min./max. expected cumulated reward until some set of states (locations) is reached
- Example properties
  - “the maximum expected time until a data packet is delivered”
  - “the minimum expected number of retransmissions before the message is correctly delivered”
  - “the maximum expected number of lost messages within the first 200 seconds”
- Model checking
  - digital clocks semantics preserves expected reachability
  - so can use existing MDP reward model checking techniques
  - no zone-based approaches (yet)

Summary

- Probabilistic timed automata (PTAs)
  - combine probability, nondeterminism, real-time
  - well suited for e.g. for randomised communication protocols
  - MDPs + clocks (or timed automata + discrete probability)
  - extension with continuous distributions exists, but model checking only approximate
- PTCTL: Temporal logic for properties of PTAs
  - but many useful properties expressible with just reachability
- PTA model checking
  - region graph: decidability results, exponential complexity
  - digital clocks: simple and effective, some scalability issues
  - forwards reachability: only upper bounds on max. prob.s
  - backwards reachability: exact results but often expensive
  - abstraction refinement using stochastic games: performs well
  - tool support: (PRISM) coming soon, mcpta, UPPAAL-Pro

Thanks for your attention

More info here:
www.prismmodelchecker.org