Automated Verification of Cyber-Physical Systems Notes on Probabilistic Model Checking

Igor Melatti

- Slides from `https://www.prismmodelchecker.org/lectures/pmc/`, collected in `all_slides.pdf`
  - in the following, slides numbering refers to such file
- All to be read, here we will comment the most important ones
- Slide 4: "validation" used in a broad sense
- Slide 17: there are protocols containing some like `if (rand() < 0.5) do_something; else do_something_else;`
  - using standard model checking techniques, we may only use non-determinism
  - thus verifying if there is a path leading to an error (if we are checking a safety property)
  - but having a path going to the error may be straightforward
  - instead, we may want to verify that an error has a low probability
  - with probabilistic model checking, probabilities are embedded in the model
- Compare slides 13 and 20...
  - counterexamples not as important as in standard model checking
- Slides 21–26: sketch of a widely used leader election protocol
- Slides 27–30: results of verifying the above protocol using PRISM (PRobabilistIc Symbolic Model checker)
  - state-of-the-art probabilistic model checker
  - all figures are obtained by performing many verifications, each time varying some parameters
  - T or the bias of a coin used in the protocol itself

- Slide 32: standard model checking only accepts a Kripke Structure-like input for the model
  - in PRISM, 3 different mathematical models may be used
  - it is the modeler task to understand which one to use
  - some logic is for some input only (e.g., CSL is only for CTMCs)
- Slide 46:
  - "termination": arrive at one of the rightmost states
  - "number of coin tosses": number of transitions to "terminate"
- Slide 51: all rows sum to 1
- Slide 52: in a stochastic matrix, from any state we must go to some (possibly the same) state
- Slide 54:
  - a more correct formula is $\mathbf{P}(x(n+k) = s \mid x(k) = s') = \mathbf{P}(x(n) = s \mid x(0) = s')$ for all n, k, s, s' in respective domains
  - the idea is that it is not important how you arrived in state s: the process "re-starts over" from s, regardless of the past
  - using only the property of slide 53 (i.e., $\mathbf{P}(x(k+1) = s \mid x(0) = s_0, \ldots, x(k) = s_k) = \mathbf{P}(x(k+1) = s \mid x(k) = s_k)$), we may still have a non-homogeneous (also called non-stationary) Markov Chain: the transition relation depends on s, s_k and k
  - that is, the property of slide 53 only looks at paths of some fixed size k; for paths of a different size (where some more or less time has passed...), probabilities may be different (thus, it is not truly "memoryless")
  - here, we will only consider stationary Markov Chains; thus, for any path (of any length) leading to s, we only consider the last step to define the probability
  - this allows us to define transition probabilities to only depend on the starting and ending states
- Slide 55: of course, suitable APs may be used to label also the other "final" states
  - only "interesting" labels are being shown
- Slide 57 (and 56): this is what each node entering the protocol does
  - $s_0 \to s_1$ corresponds to the three items in slide 56: new node picks address U at random, broadcasts probe message: "Who is using U?" and a node already using U replies to the probe (the last step happens with probability $q = \frac{\#addrs\ used}{\#all\ addrs}$)

  - $s_1 \to s_2$ given that the picked address is not good, it may be the case that the probe address was lost, so send it again with probability p (this is inside the protocol!)
  - if the probe got lost but the address is ok, it does not matter
  - probability p is typically low
  - message loss is only considered when answering the probe, not for the initial probe itself
  - after error, needs manual restart or perhaps too many devices are using the network
  - the "waiting after each one" part is not directly modeled, i.e., we are after each wait
  - protocols already with an IP must only answer to probes, if they reach one

- Slide 58:
  - first two properties are close to what it may be done in standard model checking
  - of course, dropping the probabilistic part
  - last two are in probabilistic model checking only

- Slide 60: if ω is a path of length 0, we have that all paths starting from s are in the cylinder
  - in probabilistic model checking, we only consider "basic events"
  - thus an event is any subset of paths, but a basic event is a "well-formed" (*measurable*) subset of path
  - well-formedness is defined through the concept of σ-algebra

- Slide 61: if a family $\mathcal{F}$ does not fulfill the σ-algebra properties, simply add (the minimal number of) elements in order to fulfill them
  - σ-algebra may also be called *Borel field* (requires countably infinite unions)
  - note that "family of subsets of Ω" means a set $\Sigma \subseteq 2^\Omega$
  - since a subset of Ω is an "event", Σ is a set of events
  - of course, the first and the last property imply that $\Omega \in \Sigma$
  - example: $\mathcal{F} = 2^\Omega$ is a σ-algebra for all Ω
  - example: $\mathcal{F} = \{\varnothing, \Omega\}$ is a σ-algebra for all Ω
  - example: for $\Omega = \{a, b\}$, $\mathcal{F} = \{\varnothing, \{a\}, \{a, b\}\}$ is not a σ-algebra since $\Omega \setminus \{a\} = \{b\} \notin \mathcal{F}$

- Slide 62: typically, $\Sigma = 2^\Omega$

- however, we could be interested in understanding the "minimal" $\Sigma \subseteq 2^{\Omega}$ we may use without disrupting probability definition
- thus, we take "good" subsets of $\Sigma \subseteq 2^{\Omega}$, namely σ-algebras
- we will never ask which is probability of a "bad" subset of Ω, i.e., of an element not in Σ

- Slide 65:
  - the "experiment" consists in selecting a path in the DTMC
  - however, for a given path π, the subset $\{\pi\}$ may not be an event (see example below)
  - note that there are $|S|$ probability spaces in a DTMC...
  - informally: in probabilistic model checking, we consider sets of paths (that is, subsets of Path), but not all of them: an event must consider *all and only* paths having some common prefix
    - an event with two paths without a common prefix (not even in the very first state) is not an event
    - an event not including a path having some common prefix to all other paths in the event is not an event
  - more formally, w.r.t. σ-algebras, sets in Σ must have the following property: taken some finite prefix ω, *all* infinite paths having ω as a prefix must be in the family
  - i.e., $\text{Cyl}(\omega) \in \Sigma$
  - we can see a cylinder $\text{Cyl}(\omega)$ as the sub-tree of paths starting from the last state of ω
  - suppose we have only three paths starting from s, i.e., $\Omega = \text{Path}(s) = \{\pi_1, \pi_2, \pi_3\}$ and that π_1, π_2 share a common prefix $|\omega| > 0$
  - then $\Sigma = \{\emptyset, \{\pi_2\}, \{\pi_1, \pi_3\}, \{\pi_1, \pi_2, \pi_3\}\}$ is a σ-algebra but does not fulfill the above property because $\{\pi_1, \pi_2\} \notin \Sigma$
  - simply adding $\{\pi_1, \pi_2\}$ we have $\Sigma' = \{\emptyset, \{\pi_2\}, \{\pi_1, \pi_3\}, \{\pi_1, \pi_2, \pi_3\}, \{\pi_1, \pi_2\}\}$ which is not a σ-algebra
  - we have to take the least σ-algebra containing $\{\pi_1, \pi_2\}$, i.e., $\Sigma^* = \{\emptyset, \{\pi_1, \pi_2\}, \{\pi_3\}, \{\pi_1, \pi_2, \pi_3\}\}$
  - we will never ask the probability of, e.g., $\{\pi_1, \pi_3\}$: the only finite path they have in common is s, which is also in common with π_2...
    - note that all three paths share the common prefix consisting in the state s alone; thus, $\{\pi_1, \pi_2, \pi_3\}$ must be in Σ^*, which is true
    - $\{\pi_1\}$ and $\{\pi_2\}$ are not events, despite being possible experiment outcomes

- Slide 66: in the bottom part, note that all such cylinders (set of paths) have null intersection, thus we may sum their probabilities

- Slide 67: in KSs, reachability and invariance excludes each other, here they can coexist
- Slide 68: once reached, I'm done, so I don't consider paths going back to T after having already touched T before (see definition of $\mathrm{Reach}_{\mathrm{fin}}$)
  - if a loop is present before going to T, then we have infinite paths, but always countable
- Slides 70–71: from definition to computation
- Slide 72: let's perform the computation
  - $x_{s_1} = x_{s_3} = x_{s_4} = x_{\{1\}} = x_{\{2\}} = x_{\{3\}} = x_{\{5\}} = x_{\{6\}} = 0$
  - $x_{\{4\}} = 1$, $x_{s_5} = \frac{1}{2} x_{\{4\}} = \frac{1}{2}$
  - $x_{s_2} = \frac{1}{2} x_{s_5} + \frac{1}{2} x_{s_5}$, $x_{s_6} = \frac{1}{2} x_{s_2}$, $x_{s_0} = \frac{1}{2} x_{s_2}$, which may be easily solved
- Slides 71 and 73: if the condition "if T is not reachable from s" were omitted in slide 71, then we would have the non-unique solution of slide 73
  - "reachable" here means $\mathrm{Reach}_{\mathrm{fin}}(s, T) \neq \emptyset$
  - to be determined using standard model checking techniques, essentially considering only edges with a strictly positive probability
- Slide 74: A^B is the set of functions $f : B \to A$
  - so, F takes a function from S to $[0, 1]$ and returns another function from S to $[0, 1]$
  - need not to be distribution probabilities, thus for $y \in [0, 1]^S$ we may have $\sum_{s \in S} y(s) \neq 1$
  - note that, for some $y_1, y_2 \in [0, 1]^S$, both $y_1 \leq y_2$ and $y_2 \leq y_1$ may be false, i.e., this is a partial ordering
- Slide 75: no more need of the reachability clause
- Slide 76: "power method" is the one shown in slide 75
- Slide 77: we always have to use infinite paths, as this is our Ω
- Slide 78: in the plot, probability is always closer to 1, without actually being equal to 1
  - only the first probability, which considers infinite paths, is 1
- Slide 84: note that you need two states and a bound to define the transient state probability

  - we have $|S|$ transient state distributions for each value of k, by varying the starting state of the distribution
  - note that, in transient state distributions, the destination varies and the source stays constant
  - of course, being a probability distribution, $\sum_{s' \in S} \pi_{s,k}(s') = 1$
- Slide 90: recall that π_s is a vector where, at position s', we have $\lim_{k \to \infty} \pi_{s,k}(s')$
  - i.e., the probability that, in the long run, you go from s to s'
  - the starting distribution ($k = 0$) is 1 for $s' = s$ and 0 otherwise
  - we have $|S|$ long-run distributions
- Slide 93: you can escape from an SCC, you cannot escape from a BSCC
- Slide 95: a state is aperiodic if $d = 1$, a Markov Chain is aperiodic if all its states are aperiodic
  - if a state as a self loop, then it is aperiodic
  - or if, e.g., has a cycle of length 3 and one of length 4
  - the example in this slide has period 2 for both states, and it is easy to see that the limiting distribution does not exist
  - the other example in slide 92 is not irreducible, thus the limiting distribution depends on the starting distribution
- Slide 96: it is a fix point computation
- Slide 97: written in that way, π is a row vector
  - "balance of leaving and entering": π vs. $\mathbf{P}$
- Slide 98: irreducible and aperiodic, the "original" one (with a loop on s_3) was reducible instead
- Slide 100: period of the small example is 2
  - new limit: we are considering the average of the distributions resulting after $1, \ldots, n$ steps; we then take the limit of such averages
  - the computation is the same, but the interpretation is slightly different
- Slide 101: "compute vector π_s" is of course the final goal...
- Slide 102, let us comment some values
  - in the long run, any SCC which is not BSCC will be left, thus $\pi_t(s_0) = \pi_t(s_1) = 0$ for all t
  - of course, this is a consequence of the algorithm in slide 101

- $\pi_{s_0}(s_2) = \frac{1}{2}(\frac{1}{2}\frac{1}{4} + \frac{1}{2^3}\frac{1}{4} + \frac{1}{2^5}\frac{1}{4} + \ldots) = \frac{1}{2}(\frac{1}{4}\frac{2}{3}) = \frac{1}{12}$

- Slide 104: note that all BSCCs are reached with probability 1, as in the long run such probabilities do not sum up
  - so reaching a selected BSCC has probability 1...
  - .. and also reached any of the three BSCCs has probability 1!
  - in the computation of π this does not happen only because we have the normalization factor
- Slide 105: both ok and error have probability 1
  - $\frac{1}{2}$ with normalization
  - all other states (including the retry state s_0 mentioned in the slide) have probability 0
- Slide 107: "always eventually" and "infinitely often" = **GF**
- Slide 109: "eventually forever" = **FG**
- Slide 117: some derivable operators, like OR and implication, are omitted; others, like **F** and **G**, are present
- Slide 126: compare with slide 117
  - state formulas with **E** and **A** have disappeared, replaced by the quantitative operator **P**, which allows intermediate results between "at least one" and "for all"
  - the path formulas are actually the same, with the addition of the bounded until
  - as explained in slide 127, there would be no problem in adding it to CTL too
  - of course, $k \geq 1$, and $\Phi_1 \mathbf{U}^{\leq 0} \Phi_2 \equiv \Phi_2$ (see slide 127)
  - **F** and **G**, though absent, are expressible using **U** as shown in slide 123
  - the bounded until also allows bounded **F** and **G** (see slide 130)
- Slide 128: $\text{Prob}(s, \psi)$ to be defined as in slide 66: disjoint sum of cylinders probabilities
  - that is, collect all infinite paths starting from s and satisfying ψ, consider all their common distinct finite prefixes and sum the probabilities of such prefixes
  - note that such prefixes always exist, as we have a finite number of states
- Slides 130-131: explanation

- in LTL, $\mathbf{G}\phi \equiv \neg(\mathbf{F}\phi)$
- in CTL, the same formula cannot be applied, as negations of path formulas are not allowed
- however, since $\mathbf{A}\neg\Psi \equiv \neg\mathbf{E}\Psi$ (the first formula is in CTL*, the second in CTL), we may define $\mathbf{G}$ on $\mathbf{F}$ and ultimately on $\mathbf{U}$
- an analogous trick may be done in PCTL, by negating the comparison: $\mathbf{P}_{<p}[\mathbf{G}\phi] \equiv \mathbf{P}_{\geq p}[\mathbf{F}\neg\phi]$ and similar...

- Slide 132: for the last formula, oper is evaluated on the first state only
  - however, PRISM allows a probability distribution as the initial state...
  - note also that the last property has nested probability operators, as a CTL formula may have nested state formulas
- Slide 134: when the event space is infinite, an event with probability 1 is not sure (and one with probability 0 is not impossible)
- Slide 135:
  - $\mathbf{P}((s_0 s_1)^\omega) = \lim_{k\to\infty} \Pi_{i=0}^{\frac{k}{2}} \frac{1}{2} = \lim_{k\to\infty} \frac{1}{2^k} = 0$
  - actually, it is not even an event! it does not belong to any cylinder, thus it is not in the σ-algebra
  - in fact, any prefix of $(s_0 s_1)^\omega$ with odd length (i.e., ending in s_0) may go on with s_2
  - thus, singling out $(s_0 s_1)^\omega$ only (i.e., considering the singleton event $\{(s_0 s_1)^\omega\}$) is impossible in this example
  - thus, it is correct that the final probability of reaching tails is 1...
- Slide 136: this is outside standard PCTL, but PRISM allows it as it is useful and "easy"; note that it must be the outermost P
- Slide 140: the example provided is in CTL*
- Slide 142: comparing with slide 126
  - state formulas are the same
  - path formulas also allow state formulas, as well as (direct) logical combinations of path formulas
  - note that such logical combinations are NOT redundant, i.e., they cannot be derived from the path formulas
  - the given example is not in PCTL because of $\mathbf{GF}$
- Slide 143: simply LTL + prob does not have a name, you can use PCTL* instead

- Slide 148: let us assume it is not a problem to have full graphs in memory
  - as we will see, PRISM uses OBDDs (for sets of states) and a special extension of theirs known as MTBDD for functions $S \to [0,1]$
- Slide 151: it is assumed that $\mathrm{Sat}(\Phi)$ has already been computed
  - formulas has a finite size, so atomic propositions (are logical combinations of atomic propositions) have to be used somewhere
  - we follow the formula syntax tree, starting from the leaves
- Slide 159: two overlapping formulas: $\mathrm{Sat}(b)$ and $\mathrm{Sat}(\mathbf{P}_{\geq 0}[\neg a \mathbf{U} b])$
- Slide 161: for the overlapping formulas, see slide 165
- Slide 169: $A_{i,i} \neq 0$ comes from the definition in slide 164
  - there always is the i-th variable, corresponding to $\mathrm{Prob}(s, \phi_1 \mathbf{U} \phi_2)$
- Slide 186: some limitations in the modelling language
  - probabilities must be *constant*; if something as a function of some value is needed, we have to break it down in multiple states
  - essentially as NuSMV, but with probabilities: only main arithmetic and logical operations are allowed to define next states
  - build the DTMC corresponding to a generic input model
- Slide 187: only one module may move at a time
  - there is an implicit scheduler which decides which module may move
  - NuSMV checks all possible scheduler choices (i.e., it is non-deterministic); in PRISM, instead, if there are n modules, each moves with probability $\frac{1}{n}$
  - then, inside each module, $\frac{1}{n}$ will be further multiplied by the explicitly given probabilities
  - if two or more modules have the same synchronization label (the identifier inside the square brackets `[]`), and of course the guard is true for all such modules, then they move *together* with probability $\frac{1}{n-m+1}$, being m the number of modules with that synchronization label
  - all modules may read all other modules variables, but may modify only their own
  - if inside one module there are two overlapping guards (i.e., which may be true at the same time), a warning is issued
  - if such guards are labeled by different labels, the warning disappears

  - the corresponding DTMC is as follows: if n transitions are triggered, then each has probability $\frac{1}{n}$ (to be multiplied by the probability of the module itself, as above)
  - if there are m guards with the same label (in m modules), then it is necessary that all such guards are true in the same state, in order to trigger the corresponding transition

- Slide 189: "lock-step" means that, at the same time, 2 reads from 1, 3 from 2 etc (as when an army marchs)
  - in the actual protocol this is performed over N "phases": in the first 1 reads from 2, in the second 2 reads from 3, ..., in the N-th N reads from 1
  - in the PRISM model, it is actually lock-step as above, but N rounds are however waited to be closer to reality
  - in order to guarantee that only the (maximum) unique ID survives, all other (equal) IDs are set to 0

```
dtmc
const N = 3;
const K = 2;
module counter
  c : [1..N-1];
  [read] c<N-1 -> (c'=c+1);
  [read] c=N-1 -> (c'=c);
  [done] u1|u2|u3 -> (c'=c);
  [retry] !(u1|u2|u3) -> (c'=1);
  [loop] s1=3 -> (c'=c);
endmodule
module process1
  s1 : [0..3];
  u1 : bool;
  v1 : [0..K-1];
  p1 : [0..K-1];
  [pick] s1=0 -> 1/K : (s1'=1) & (p1'=0) & (v1'=0) & (u1'=true)
              + 1/K : (s1'=1) & (p1'=1) & (v1'=1) & (u1'=true);
  [read] s1=1 &  u1 & c<N-1 -> (u1'=(p1!=v2)) & (v1'=v2);
  [read] s1=1 & !u1 & c<N-1 -> (u1'=false) & (v1'=v2) & (p1'=0);
  [read] s1=1 &  u1 & c=N-1 -> (s1'=2) & (u1'=(p1!=v2)) & (v1'=0) & (p1'=0);
  [read] s1=1 & !u1 & c=N-1 -> (s1'=2) & (u1'=false) & (v1'=0);
  [done] s1=2 -> (s1'=3) & (u1'=false) & (v1'=0) & (p1'=0);
  [retry] s1=2 -> (s1'=0) & (u1'=false) & (v1'=0) & (p1'=0);
  [loop] s1=3 -> (s1'=3);
endmodule

module process2 = process1 [ s1=s2,p1=p2,v1=v2,u1=u2,v2=v3 ] endmodule
```

- Slide 199: "properties" because (see slide 207)
  - the expected value may be compared with some threshold, as for probability-based properties
  - the expected value may be computed for some subset of states
  - defined in two different points:
    - first, define how the reward is computed (model file, e.g., slide 201)
    - then define how to use the reward in a formula (formula fil, slide 207)
  - difference between instantaneous and cumulative is made in the formula (see slide 207)
    - difference between state and/or transition is made when the reward is defined
    - by the fact that [labels] are used (see slide 201)
- Slide 202: what PRISM allows to you to compute
  - given a path chosen at random, which is the expected value for the defined reward
  - if we look at all paths and make an average, the value we get is the expected value
  - that is: for all paths π, the value of the reward in π multiplied by the probability of π
- Slide 203: in a nutshell, rewards allows the modeler to define a *random variable*
  - a random variable is a function $X : \Omega \to \mathbb{R}$
  - it describes some experiment on Ω, by associating to each single outcome $\omega \in \Omega$ a real value
  - example, if I roll a die and I am payed $10n$ if the outcome is $n \leq 3$, and I have to pay $3n$ otherwise, I am defining a random variable $X : \{1, \ldots, 6\} \to [-18, 30]$
  - examples in slide 201 all define a random variable on $\Omega = \text{Path}$: given a path, I can measure each of such rewards definitions
  - typical operation on random variables: expected value $\mathbf{E}[X]$ as defined in this slide
  - other typical operation: asking probabilities for exact values or intervals
  - e.g.: in the die roll example above, which is the probability I win at least 10 EUR $\mathbf{P}(X \geq 10)$?

- Slide 204: how the random variable resulting from a reward definition is computed
  - if T is not specified, we may assume that $T = \varnothing$
  - note that, if there exist a(n infinite) path with non-zero probability (e.g., looping in it last state iwht probability 1) which do not touch T, then the result is ∞
  - for instantaneous, only consider k_τ (see slide 208)
- Slide 206: written as in slide 207, it would be $\mathbf{R}_{=?}[\mathbf{F}s3]$ (assuming s3 is a atomic proposition which is true only in state s_3)
- Slide 233: in CTMCs it is not only important that a transition is taking place, but also *after how much time*
- Slide 235:
  - $f(t)$ is not a probability! If $b - a < 1$, $f(t) > 1$ for $t \in [a, b]$...
  - it may seem confusing, but "probability density function" (PDF) $\neq$ probability
  - it becomes a probability when multiplied by an interval length (also infinitesimal): $f(x)dx$ is the probability that the value of X is inside $[x, x + dx]$
  - the "cumulative distribution function" (CDF) $F(t)$, instead, is a probability
  - integrals go from some lower bound; in the general case, it is $-\infty$ but may be overriden by special cases
  - the X is of course a random variable with values on times, we will define it more precisely in the next slide
- Slide 236:
  - if the PDF f is $f(x) = \lambda e^{-\lambda x}$ then we have this special continuous random variable called exponential
  - despite looking "ugly", many computations are simplified, e.g., we easily derive a closed form for $F(t)$
  - formula for expected value when an $f(x)$ is available: $\mathbb{E}[X] = \int_{-\infty}^{\infty} xf(x)dx$
  - in the case of the exponential distribution: $\int_0^\infty x\lambda e^{-\lambda x} = [-xe^{-\lambda x} - \frac{1}{\lambda}e^{-\lambda x}]|_0^\infty = \frac{1}{\lambda}$
- Slide 238: to clarify examples
  - in all these cases, we have a random variable X with CDF $F(t) = \mathbb{P}(X \leq t) = 1 - e^{-\lambda t}$ (or equivalently $\mathbb{P}(X > t) = e^{-\lambda t}$), and λ is known by some (typically statistical) measures

- t is time (as discussed in the examples in this slides), and λ is the corresponding "rate" or "frequency"
- "time before machine component fails": X =time of machine component failure
- thus, X is random variable where Ω may be the outcomes of the experiment "turn on the machine and use it for T seconds in some random way, recording all activities at any time", and it returns the time at which the first failure happens
- for example, if we know that $\lambda = 1$, then the probability that the component fails after time $t = 1$ is $e^{-1} \approx 36.8\%$; conversely, it fails before $t = 1$ with probability 63.2%
- for example, if we know that $\lambda = 2$, then the probability that the component fails after time $t = 1$ is $e^{-2} \approx 13.5\%$; conversely, it fails before $t = 1$ with probability 86.4%
- in the same assumptions as the previous point, probability of a failure after $t = 3$ is $e^{-6} \approx 2 \times 10^{-3}$; conversely, it fails before $t = 3$ with probability 99.8%
- not surprising: λ is the "rate", meaning the number of failures (in this case) for every time unit
- thus, saying $\lambda = 2$ means there are "typically" 2 failures at each time unit; so, within 3 time units, we should be almost sure that at least one failure has happened...
- of course, "time unit" depends on the problem and on how λ has been estimated; it could be 10 years, in the case of a computer component (so $t = 3$ means 30 years)
- easy to see why the expected value is $\frac{1}{\lambda}$: if the rate is 2, it should happen twice in a time unit, thus we should see the failure in $\frac{1}{2}$ time units...
- so, generally speaking: we know that something happens with some regularity (i.e., λ times every time unit), so which is the probability of the event happening before time t?

- Slide 240: there are two possible failures with rates λ_1, λ_2, we want to know when at least one of the two happens
- Slide 241: similar to slide 240
- Slide 243–245: no probabilities, only rates with the clarified meaning
  - this "generates" a probability once also a time t is considered
  - that is: from s_0, at time t there is a probability $1 - e^{-\frac{3}{2}t}$ to go to state s_1, and $e^{-\frac{3}{2}t}$ to stay in s_0
  - this implies that only one rate may be considered from each state: hence the discussion in slide 245

  - see also slide 240
  - note that there are not self loops, as there always is a probability of staying within a given state
  - of course, such a probability of staying in a state decreases as the rate increase ($1 - (1 - e^{-\lambda t}) = e^{-\lambda t}$ is decreasing in λ)...
  - note that, for an absorbing state s, probability of staying in s is 1
  - thus, we have now two types of probability: one of going from one state to another, and one of waiting some time before going
  - this fact will be reflected in the definition of paths (see slide 252)

- Slide 246: of course, if just one rate is available from a given state s to some s', then $R(s, s') = E(s)$...

- Slide 248:

  - so $R = Q$, excluding the diagonal, where R is 0 whilst Q has the information on the probability to stay in s
  - that is, probability of still being in s at time t is $e^{Q(s,s)t}$
  - we can also see that $P^{\text{emb}}(s, s') = \frac{Q(s,s')}{-Q(s,s)}$ if $Q(s, s) \neq 0$ and $P^{\text{emb}}(s, s') = 1$ otherwise
  - rows in P^{emb} sum to 1, rows in Q sum to 0
  - random variables with values in S rather than $\mathbb{R}$; expected value will have a different definition...
  - here, events in $\Omega(t)$ are "make a random walk on the CTMC for at least t time units", where "random walk" must obey the rules seen till now
  - see also the discussion in slide 53: in that case, the family of random variables was infinite but countable, here it is uncountable
  - memorylessness: t_k are any selected time instants

- Slide 250: MTTF and rate, if rate of failure is 2 every day, then MTTF is 12 hours (half a day)...

  - $i\lambda$ in state i: if we have i machines, each failing once every year, then we have i failures in one year...

- Slide 250–251: λ, μ, k_i must be instantiated to some value before going on with verification

- Slide 252:

  - note that a seemingly finite path is instead infinite (but ending in an absorbing state is required)
  - paths in DTMCs only have states, here we have times too

  - no restriction on times, apart from being strictly positive
  - for times growing, probability decreases exponentially, but it is still possible...
  - $\omega@t = s_i$ s.t. $\sum_{j=0}^{i} t_j \geq t$ and i is the minimum
- Slide 254:
  - for DTMCs, cylinders are simply finite prefixes of some path
  - here we also have times, which may be different for the same (sub)sequence of states
  - in order to have cylinders which define sets of infinite paths, we have to somehow abstract on times: that's why we have time ranges on them
- Slide 255: written explicitly, $\mathbb{P}_s(\mathrm{Cyl}(s_0(a_0, b_0]s_1 \ldots (a_n, b_n]s_{n+1})) = \Pi_{i=0}^{n} P^{\mathrm{emb}(C)}(s_i, s_{i+1})(e^{-E(s_i)a_i} - e^{-E(s_i)b_i})$
  - recall that $E(s) = \sum_{s' \neq s} R(s, s')$
  - $P^{\mathrm{emb}(C)}$ is to "disambiguate" race conditions; if only one rate is defined from a state s, then $P^{\mathrm{emb}(C)}(s, s') = 1$ for a single s'...
- Slide 263:
  - Path^C is to emphasize that is about CTMC C
  - easy to define steady state probabilities, compare with slide 96
- slide 265: e^{Qt} denotes the matrix where in position (s, s') we have $e^{tQ(s,s')}$
  - analogously for Q^i (and $\frac{Q}{q}$ in slide 266)
  - remember that in Π_t, t is a time, not a state
  - unstable: the limit exists, but computation may diverge
- Slide 266: rows in P^{unif} sum to 1
- Slide 268: in the Poisson probability, we usually have $\lambda = qt$
  - actually, "probability mass function", as the Poisson process is discrete
  - λ in exponential distribution and in the Poisson probability are different, though related
  - that is: suppose that we have some event which may happen multiple times within a given (fixed) interval of time
  - knowing that the "typical" number of times is λ, which is the probability that we observe k events?
  - of course, it should be high for k close to λ, and low otherwise

  - e.g., if there are 2 failures every day, which is probability of having two failures in one day? it is $\mathbb{P}(X = 2) = \frac{2^2 e^{-2}}{2!} \approx 27\%$
  - having 3 failures is $\mathbb{P}(X = 3) = \frac{2^3 e^{-2}}{3!} \approx 18\%$, 1 failure is the same of 2, 0 failures is 13.5%; with 7 failures or above, the probability is below 1%
  - rates are usually shown as r instead of λ, thus $\lambda = rt$ if t is the period length
  - so: exponential distribution is about how much (continuous) time for the first occurrence, Poisson is about how many occurrences we have in a given time

- Slide 279: CSL also have an operator for steady probability
  - may be also be expressed with rewards
- Slide 353: c could have been named a; named c instead because of following examples
- Slide 354: from "transition probability matrix" of DTMCs to "transition probability function" of MDPs
  - Act, if provided, must be finite
  - $\text{Dist}(S) = \{\pi \mid \pi : S \to [0, 1] \wedge \sum_{s \in S} \pi(s) = 1\}$
  - for all $s \in S$, $\text{Steps}(s)$ is a set where each element is a pair (l, π)
- Slide 358: that is not actually a matrix, needs delimiters
  - could be seen as a sequence of matrices $M_1, \ldots, M_{|S|}$ where M_s has $|S|$ columns and $|\text{Steps}(s)|$ rows
  - all piled vertically
- Slides 359–360:
  - blue PRISM input language code: little trick to say "define a new module equal to M1, where all occurrence of variable s are replaced by t"
  - we now have two modules without any synchronizing label, thus we have to make a parallel composition
  - formally, $S = S_1 \times S_2$, where S_i is the set of "local" states of M_i
  - $s_{\text{init}} = (s_0, t_0)$
  - $\text{Steps}(s_i, t_j) = \{((i, j)_1, \lambda x.\ P_1(s_i, x)), ((i, j)_2, \lambda x.\ P_2(t_j, x))\}$
- Slide 363: of course, there are infinitely many adversaries also for this little example
  - note that each adversary must resolve *all* possible finite paths

- in this easy example, σ_1, σ_2 are well defined because there are not other paths, given that choices

- Slide 365: there are no deadlocks, thus there always are infinite paths of finite length

- Slide 369:
  - $\text{Prob}^{\sigma_2}(s_0, \mathbf{F} \text{ tails}) = \text{Prob}(\{s_0 s_1 s_1 p(i_1), s_0 s_1 s_0 s_1 p(i_2) \mid p(k) = s_3^k\}) = 0.3 \cdot 0.5 + 0.7 \cdot 0.5 = 0.5 \cdot (0.3 + 0.7)...$
  - $\text{Prob}^{\sigma_3}(s_0, \mathbf{F} \text{ tails}) = \text{Prob}(\{s_0 s_1 s_1 s_1 p(i_1), \ s_0 s_1 s_0 s_1 s_0 s_1 p(i_2), \ s_0 s_1 s_1 s_0 s_1 p(i_3), \ s_0 s_1 s_0 s_1 s_1 p(i_4) \mid p(k) = s_3^k\}) = 0.3^2 \cdot 0.5 + 0.7^2 \cdot 0.5 + 2 \cdot 0.7 \cdot 0.3 \cdot 0.5 = 0.5 \cdot (0.3^2 + 0.7^2 + 2 \cdot 0.3 \cdot 0.7) = 0.5$; actually,
  - $\text{Prob}^{\sigma_k}(s_0, \mathbf{F} \text{ tails}) = 0.5$
  - so, why the minimum is zero?? because, if we take the limit, then there is always an adversary which traps the MDP in a finite sequence of $s_0 s_1 ...$