Dependability analysis through Monte Carlo methods: the case of rare events

G. Rubino

Inria Rennes, France

Agenda

- 1 Introduction
- 2 Dependability analysis
- 3 Importance sampling ideas
- 4 The splitting approach
- 5 Bounding techniques
- 6 Summary
- Questions

1 Introduction

- Speaker:
  - Senior researcher at INRIA, France
  - Head of the *Dionysos* team
  - Area: dependability and performance of (complex) systems
- The topic of this presentation:
  - Dependability analysis of **critical** systems
  - Technically, how to deal with rare events (events having very small probabilities) from the quantitative point of view?
- Critical systems: those where catastrophic failures can lead to human losses.
- Definition extended to include the cases of events leading to huge financial losses or to huge environmental harms.

1 Introduction

- When there are possible losses in human lives, the system is seen as non-repairable
  - this refers to its critical aspects, or its critical services, defined as previously stated
  - typical examples: transportation (aircrafts, …), nuclear production plants, some medical systems, …
- If criticality comes from the risk of huge financial losses, for example, or from the environment, systems can be non-repairable as before, or repairable
  - for a repairable system, think of some communication networks and information systems, …
- This classification refers to the relevant metrics to analyze critical systems; see below.

Critical systems

- More requirements in modern industry lead to new needs, in particular for understanding complex systems' behavior.
- One of the main points of view in this understanding process refers to the dependability properties, in particular for critical systems.
- Very recently we have realized that criticality also appears because of the emerging interconnection of very different huge networks (**a topic in itself**).
- The usual case is that in a critical system, the "bad" situation, the catastrophic failure, is rare (some exceptions exist, e.g. the case of the American Space Shuttle).
- It must be also added that in industry many assessments are expert-based, instead of model-based (**a topic in itself**). Here, we will be concerned only with the latter.

Main stochastic tool: Markov

- Markov models are powerful: a solid theory, a large body of evaluation methods (analytical, numerical, plus simulation); some extensions are possible
- Price for accuracy: large (huge) state spaces
- Critical system --> rare events --> stiffness --> serious numerical problems (if numerical analysis is possible)

However, for this rare event situation

1. **Efficient Monte Carlo techniques exist**; they allow the evaluation of very complex models
2. **Efficient numerical techniques also exist** for computing **bounds** of metrics of interest

2 Dependability analysis

- Quantitative analysis of the behavior of (complex) systems with respect to failures and repairs of their components
- A reference for standard associated concepts:
  - "*Basic Concepts and Taxonomy of Dependable and Secure Computing*", by A. Avizienis, J.-C. Laprie, B. Randell and C. Landwehr, IEEE TRAN. on DEPENDABLE AND SECURE COMPUTING, VOL. 1, NO. 1, JAN-MAR 2004

Basic metrics: systems with no repairs

- Assumption: at time *t*, systems and their components are
  - either up, = working (perfectly)
  - or down, = failed (completely)

  (extensions exist for the case of more system's states)
- **MTTF**: Mean Time To Failure
  - mean time from beginning of operation until first failure
- ***R*(*t*)**: reliability at time *t* = Pr(system is up from 0 to *t*)
- Let *T* be r.v. "time until first failure" = "life-time"
  - MTTF = E(*T*)
  - *R*(*t*) = Pr(*T* > *t*)
  - We have

$$\mathrm{MTTF} = \int_0^\infty R(t)\,\mathrm{d}t$$

Basic metrics: systems with repairs

All presented before, plus

- Point availability at *t*:
  - ***A*(*t*)** = Pr(system is up at time *t*)
- Asymptotic availability: ***A*(∞)**
- Interval availability on [0, *t*]:
  - the **r.v. *IA*(*t*)** = "fraction of [0,*t*] during which system is up"
  - minimal info. about this **random variable**:
    the expected interval availability on [0, *t*]: **E(*IA*(*t*))**
  - we have: $$\mathrm{E}(IA(t)) = \frac{1}{t}\int_0^t A(u)\,\mathrm{d}u$$
- MTTR: Mean Time To Repair
- MTBF: Mean Time Between Failures = MTTF + MTTR

Rare events

- When the system is not repairable,
  - we typically deal with constraints such as

    $$\text{MTTF} > a >> 1,$$

    or

    $$R(t) > 1 - \varepsilon,$$

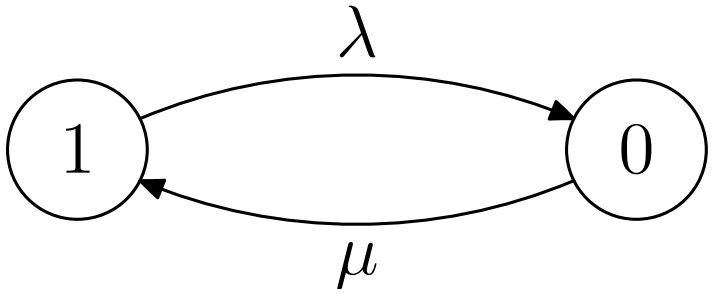
    where t is the mission time.
  - in practice, we can find values of ε such as 10^{-8}, 10^{-9}, …
- If the system is repairable,
  - we often focus on the asymptotic unavailability $\boldsymbol{U} = 1 - A(\infty)$
  - we can find cases where $U \approx 10^{-9}$ but also systems with higher tolerable unavailability values (10^{-5}, 10^{-6}, …)

On the exact metric evaluation

- Most used tools: Markov models
- Continuous time, homogeneous, in general (but not always) finite state spaces
- Remarks:
  - Computing MTTF: a **linear** problem
  - Computing $R(t)$, $A(t)$: a **linear differential** problem
  - Computing $A(\infty)$: a **linear** problem as well
  - Analyzing $IA(t)$: a **much more complex differential** problem
  - …
- Many algorithms available, a solid theory and **also many techniques to apply Markov tools to more general models**

Elementary example: basic metrics

- One component, failure rate λ, repair rate μ:



- MTTF = 1/λ, MTTR = 1/μ
- $R(t) = \exp(-\lambda t)$
- $A(t)$: we must solve the balance linear differential system, to obtain

$$A(t) = \frac{\mu}{\lambda+\mu} + \frac{\lambda}{\lambda+\mu} e^{-(\lambda+\mu)t}$$

- For $A(\infty)$, we must solve the equilibrium linear balance equations, or, here, take the limit above, to get $A(\infty) = \mu/(\lambda+\mu)$. For the unavailability, $U = \lambda/(\lambda+\mu)$

Elementary example: *IA*(*t*)

- If we look now at E(*IA*(*t*)), we also have a linear differential problem; we obtain

$$\mathrm{E}(IA(t)) = \frac{\mu}{\lambda+\mu} + \frac{\lambda}{(\lambda+\mu)^2 t}\,\mathrm{e}^{-(\lambda+\mu)t}$$

- The distribution of *IA*(*t*) is also extremely useful. In many systems, we want (or we should want) to have Pr(*IA*(*t*) > 1 – δ) > 1 – ε ("system is available most of the mission time with high probability").
- But if we need the distribution of *IA*(*t*), then things are much more complex: first, observe that

$$Pr(IA(t) = 1) = \mathrm{e}^{-\lambda t}$$

meaning that there is a mass at 1 (a defective r.v.).

- for $x < 1$, distribution of $IA(t)$, first form:

$$\Pr(IA(t) \le x) = 1 - e^{-\lambda x t}\left(1 + \sqrt{\lambda\mu x t}\int_0^{(1-x)t} \frac{e^{-\mu y}}{\sqrt{y}} I_1\left(2\sqrt{\lambda\mu x t y}\right) dy\right)$$

$$I_1(z) = \sum_{j\ge 0}\left(\frac{z}{2}\right)^{2j+1} \frac{1}{j!(1+j)!} \qquad \text{(Bessel function, first kind)}$$

- for $x < 1$, distribution of $IA(t)$, second form:

$$\Pr(IA(t) \le x) = \sum_{n\ge 0} e^{-\mu(1-x)t} \frac{\left(\mu(1-x)t\right)^n}{n!} \sum_{k\ge n+1} e^{-\lambda x t} \frac{\left(\lambda x t\right)^k}{k!}$$

- IA(t) distribution, third form: for $x < 1$,

$$\Pr(IA(t) \le x) = \sum_{n \ge 1} e^{-\nu t} \frac{(\nu t)^n}{n!} \sum_{k=1}^{n} C_{n,k-1} p^{n-k+1} q^{k-1} \sum_{i=k}^{n} C_{n,i}\, x^i (1-x)^{n-1}$$

$$C_{m,j} = \frac{m!}{j!(m-j)!}, \quad \nu = \lambda + \mu, \quad p = 1 - q = \frac{\lambda}{\nu}$$

- This is the right form for numerical evaluation, accuracy and even probabilistic insight. We say that it is “Uniformization-based”

Extensions to the Markov setting

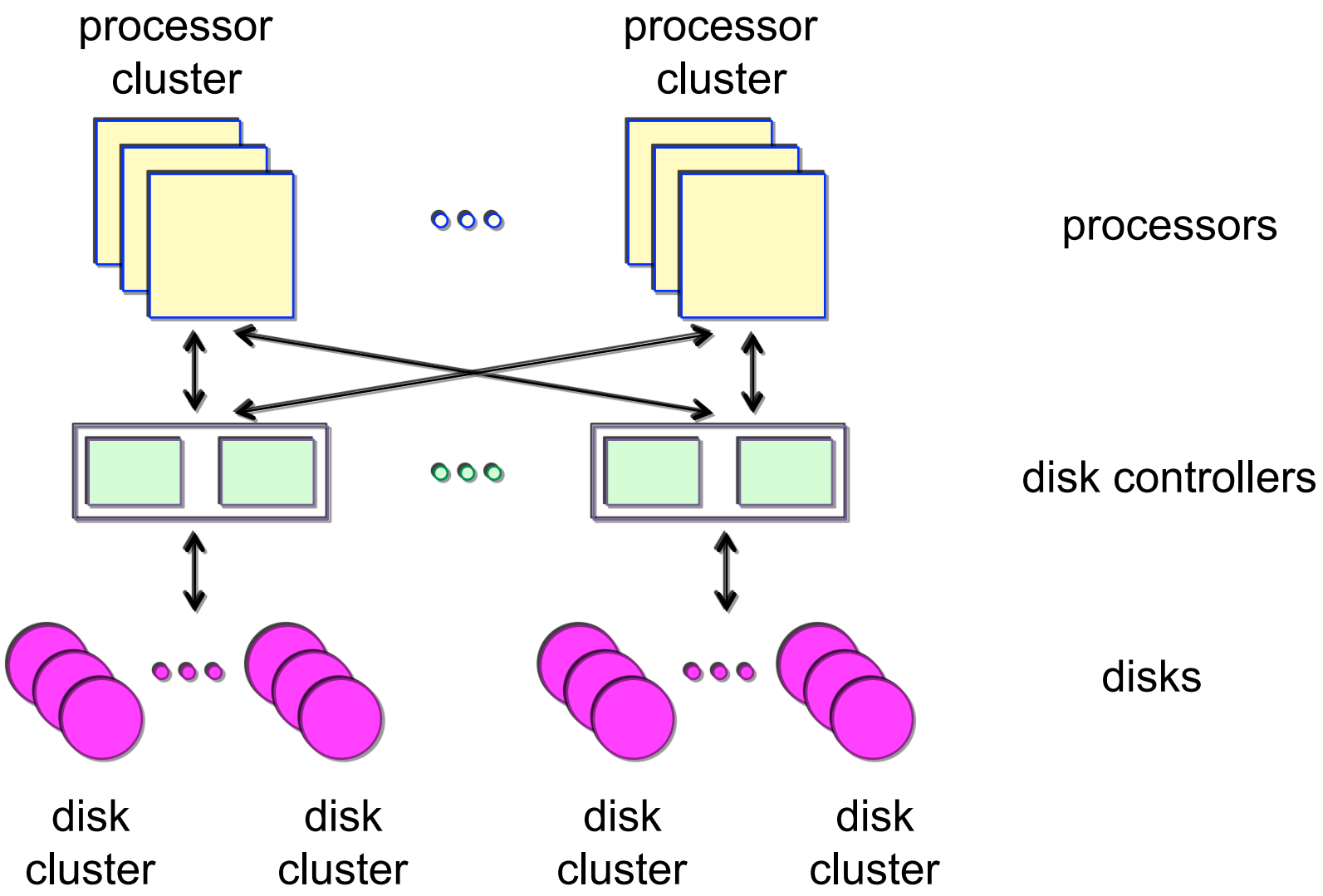
- Life-times and/or repair-times with non-exponential distributions (Coxian, series-parallel, Phase-type…)
- Semi-Markov models
- Non-homogeneous models
- Hybrid models with both discrete and continuous state components
- Inclusion of deterministic times in the stochastic models

- In these, **part** of the good sides of Markov models are conserved
- In the talk, we will stay Markovian

A typical example

- Case study from literature: a multi-processor fault-tolerant computer system, where
  - processors were organized in clusters
  - data was stored in disk arrays, where redundancy made that any array supported the failure of any (but only one) of its disks (that is, without loosing data)
  - disk controllers were organized in pairs, each pair controlling several disk arrays
  - each cluster of processors was connected to all the pairs of disk controllers
  - system worked (all data could be accessed) with
    - at least one processor per cluster,
    - at least one controller per pair,
    - at most one failed disk in each array

A realistic example: graphical scheme



A realistic example: some details

- There was one repairman, with FIFO scheduling
- Assume exponentially distributed components' life-times and repair times
- Typical parameters' values:
  - 2 clusters, 4 pairs of controllers, 8 disk arrays, each with 4 disks
  - processor failure rate = controller failure rate = 1/2000 f/day (≈ 1 fail. every 5.5 y, on ave.)
  - disk failure rate: disks come in pairs with failure rates 1/4000, 1/5000, 1/8000 and 1/1000 f/day (highest ≈ 1 fail. every 2.7 y, on ave., lowest ≈ 1 fail. every 22 y, on ave.)
- When a processor failed, with probability 0.1 another processor in the other cluster was contaminated and failed as well; there was also a set of failure modes, and the repair rate depended on the mode (not detailed here)
- The resulting model was a continuous time irreducible and homogeneous Markov chain; number of states was > 7.4×10^{14}

A realistic example: analysis?

- Assume we want to evaluate the MTTF and the asymptotic availability $A(\infty)$ of such a model.
- There is no analytical expression of these metrics (no formula) and model's size precludes the use of numerical methods, even if problems are just linear systems to solve.
- Simulation? Yes, no problem with size, but then, rarity strikes:
  - the MTTF is too high; we have to wait too much to observe failures;
  - the asymptotic availability is too close to 1, with the same consequences, we have to wait too much to be able to see failures and repairs.

The rest of the presentation

- We will introduce some of the available tools to deal with these problems, mainly through examples.
- One of the points we want to make is that many of these techniques are, as far as we have seen, unknown to most software used in the area, and also to many analysts.
- The corresponding research areas are very active today.

Crude Monte Carlo for MTTF

- Group all down states into a single state *d*.
- Call 0 the initial state (where everything is perfectly working): $X(0) = 0$.
- Denote

$$\tau_0 = \inf\left\{t > 0 : X(t) = 0, X(t^-) \neq 0\right\}$$

$$\tau_{\mathrm{d}} = \inf\left\{t > 0 : X(t) = \mathrm{d}\right\}$$

- It can then be proved that

$$\mathrm{MTTF} = \frac{\mathrm{E}\left(\inf\left\{\tau_0, \tau_{\mathrm{d}}\right\}\right)}{\Pr\left(\tau_{\mathrm{d}} < \tau_0\right)}$$

easy to estimate

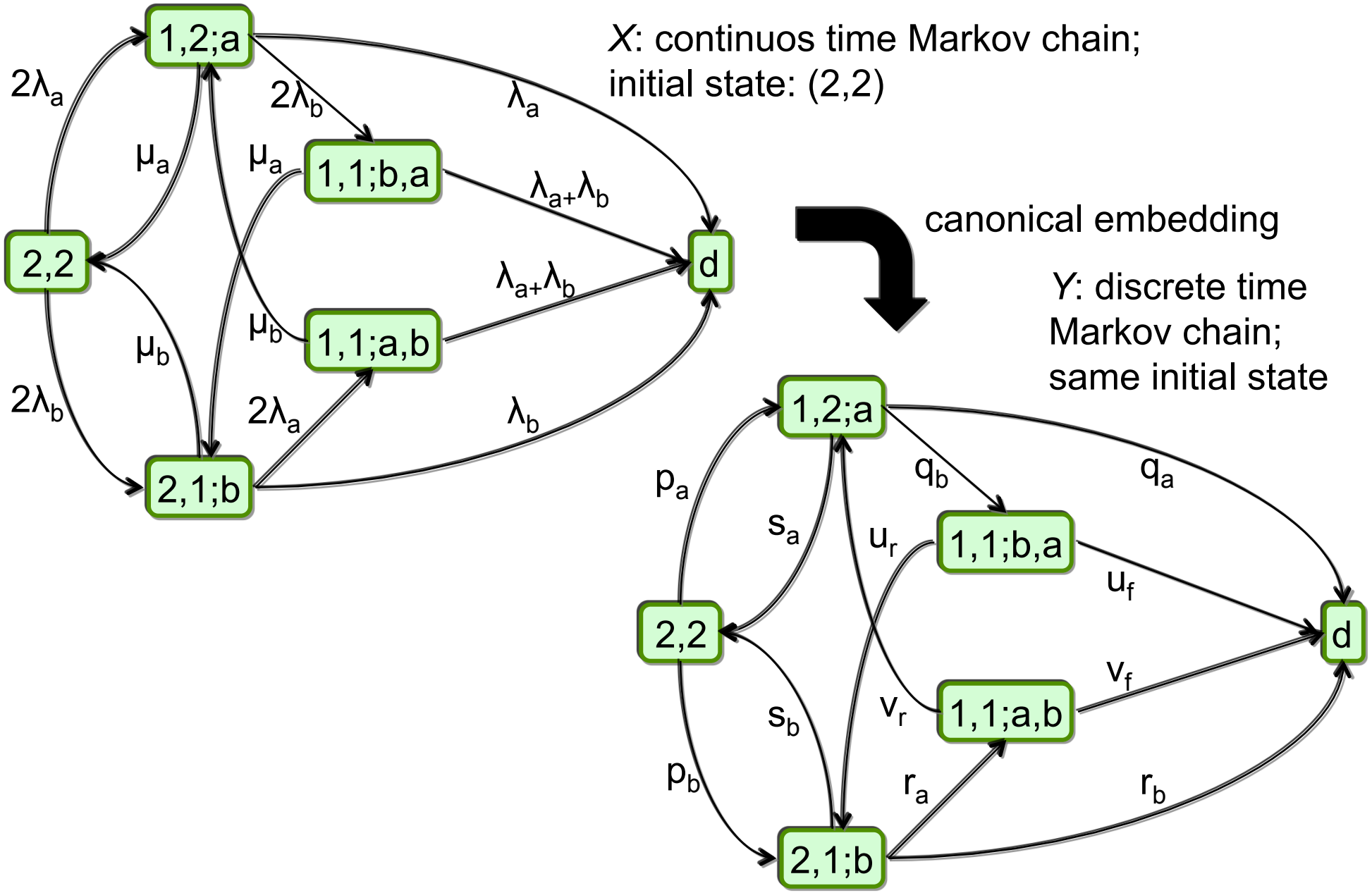
hard to estimate

- Denote $$\gamma = \Pr\left(\tau_{\mathrm{d}} < \tau_0\right)$$
- The problem reduces to find an efficient way of estimating the probability γ (in a critical system, a very small number).
- The interest is that now we only need to work in **discrete time**.
- Computing γ is a linear problem; same size as for computing MTTF.
- Let us look at an example:

An example

- Suppose there are 2 types of components: a, b, and that initially, we have 2 units of each type.
- System works if there is at least one component working per class. There is a single repairman working in FIFO order.
- States:
  - initial state: (2, 2)
  - other states: (n_a, n_b; state of the repair queue), where n_a (resp. n_b) is the number of type a (resp. type b) working units (n_a, n_b in {2, 1, 0}).
  - examples:
    - (1, 2; a): class a unit at repair
    - (1, 1; a, b): class b unit at repair, class a unit failed and waiting
    - (1, 1; b, a): class a unit at repair, class b unit failed and waiting

Model


1,2;a
2λa
2λb
λa
μa
μa
1,1;b,a
λa+λb
2,2
d
λa+λb
μb
μb
1,1;a,b
2λb
2λa
λb
2,1;b
X: continuos time Markov chain;
initial state: (2,2)
canonical embedding
Y: discrete time
Markov chain;
same initial state
1,2;a
pa
qb
qa
sa
ur
1,1;b,a
uf
2,2
d
vf
sb
vr
1,1;a,b
pb
ra
rb
2,1;b

$$p_a = \frac{\lambda_a}{\lambda_a + \lambda_b}, \qquad p_b = \frac{\lambda_b}{\lambda_a + \lambda_b}$$

$$q_a = \frac{\lambda_a}{\lambda_a + 2\lambda_b + \mu_a}, \quad q_b = \frac{2\lambda_b}{\lambda_a + 2\lambda_b + \mu_a}, \quad s_a = \frac{\mu_a}{\lambda_a + 2\lambda_b + \mu_a}$$

$$r_a = \frac{2\lambda_a}{2\lambda_a + \lambda_b + \mu_b}, \quad r_b = \frac{\lambda_b}{2\lambda_a + \lambda_b + \mu_b}, \quad t_a = \frac{\mu_b}{2\lambda_a + \lambda_b + \mu_b}$$

$$u_f = \frac{\lambda_a + \lambda_b}{\lambda_a + \lambda_b + \mu_a}, \qquad u_r = \frac{\mu_a}{\lambda_a + \lambda_b + \mu_a}$$

$$v_f = \frac{\lambda_a + \lambda_b}{\lambda_a + \lambda_b + \mu_b}, \qquad v_r = \frac{\mu_b}{\lambda_a + \lambda_b + \mu_b}$$

Exact computation of γ

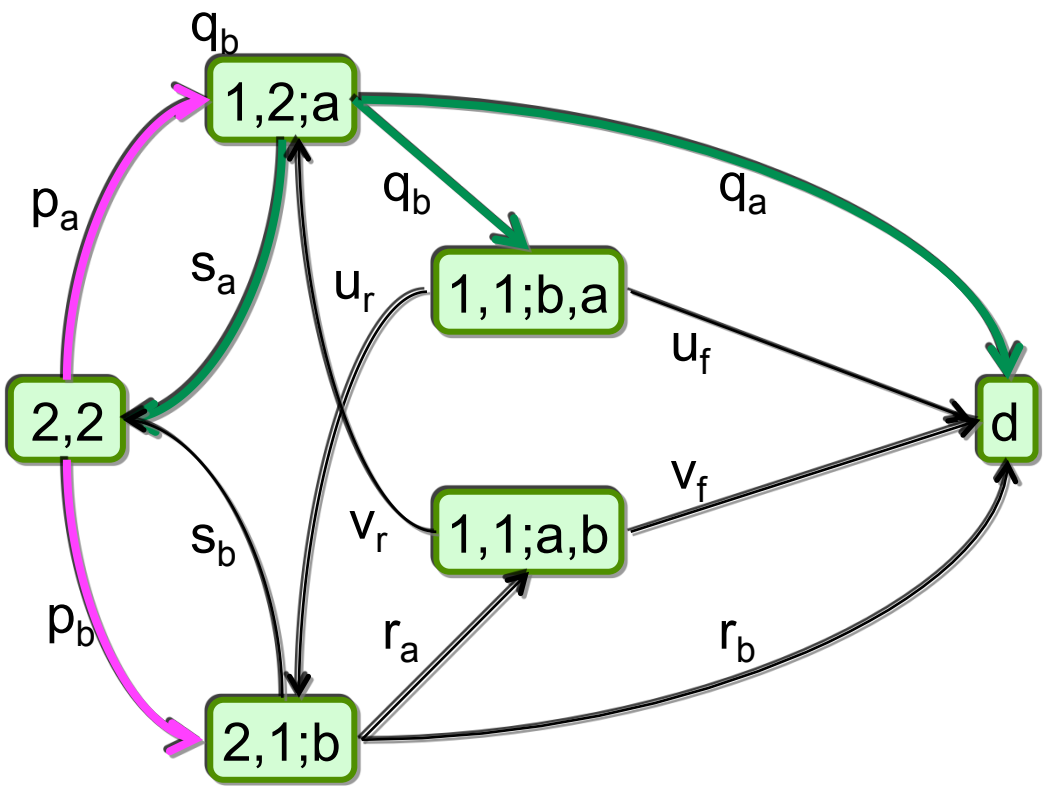
Using

$$\gamma_x = \Pr(\tau_d < \tau_0 \mid X(0) = x),$$

$$\gamma_{2,2} = p_a \gamma_{1,2;a} + p_b \gamma_{2,1;b}$$

$$\gamma_{1,2;a} = q_a + q_b \gamma_{1,1;b,a}$$

...



Crude or Standard Monte Carlo

- Crude Monte Carlo for estimating γ:
  - set counter *C* to 0
  - perform *N* times (*N* large):
    - build a path of *X* from state 0, stopping when the path reaches either state *d* or state 0 back
    - increase *C* by 1 if the path ends in *d*
  - estimate γ by $\gamma^* = C/N$
  - a confidence interval for γ^* with confidence level, say, 95%:

    $$(\gamma^* - 1.96\,\Delta,\ \gamma^* + 1.96\,\Delta)$$

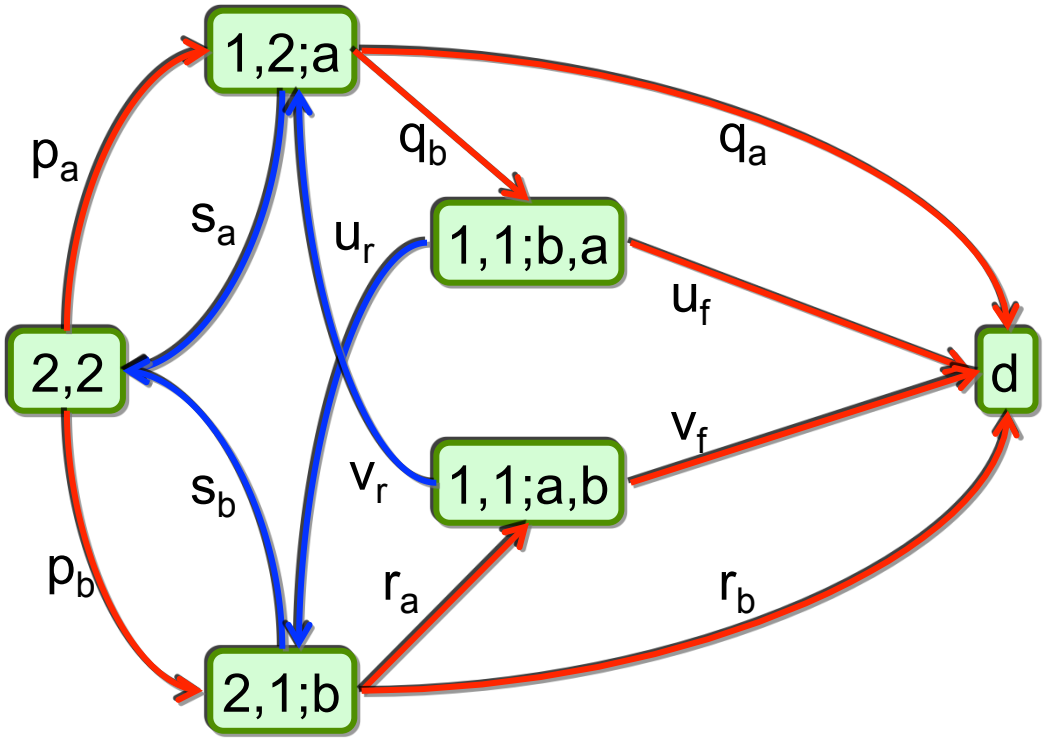
    where $\Delta = (\gamma^*(1 - \gamma^*)/N)^{1/2}$
  - the ratio $\Delta/\gamma^* = ((1 - \gamma^*)/(N\gamma^*))^{1/2}$ is seen as the relative error associated with the estimator, $RE(\gamma^*)$
  - when γ^* is very small, we need *N* huge in order to keep $RE(\gamma^*)$ reasonable (small enough); often impossible to do

3 Importance sampling (IS) ideas

- **Instead of simulating model *Y*, simulate a different model *Z* having the same topology than *Y* but different dynamics**
- Visiting state *d* before coming back to 0 is rare in *Y*. The transition probabilities associated with failures are usually very small. If *Z* is "built correctly", the event becomes frequent in *Z*.
- This "change of dynamics" can be done by replacing the failure rates (in *X*) by new values of the order of magnitude of the repair rates.
- **Instead of estimating γ, a different quantity γ' is estimated in *Z***, such that estimating it allows to derive an estimation of the real target γ.

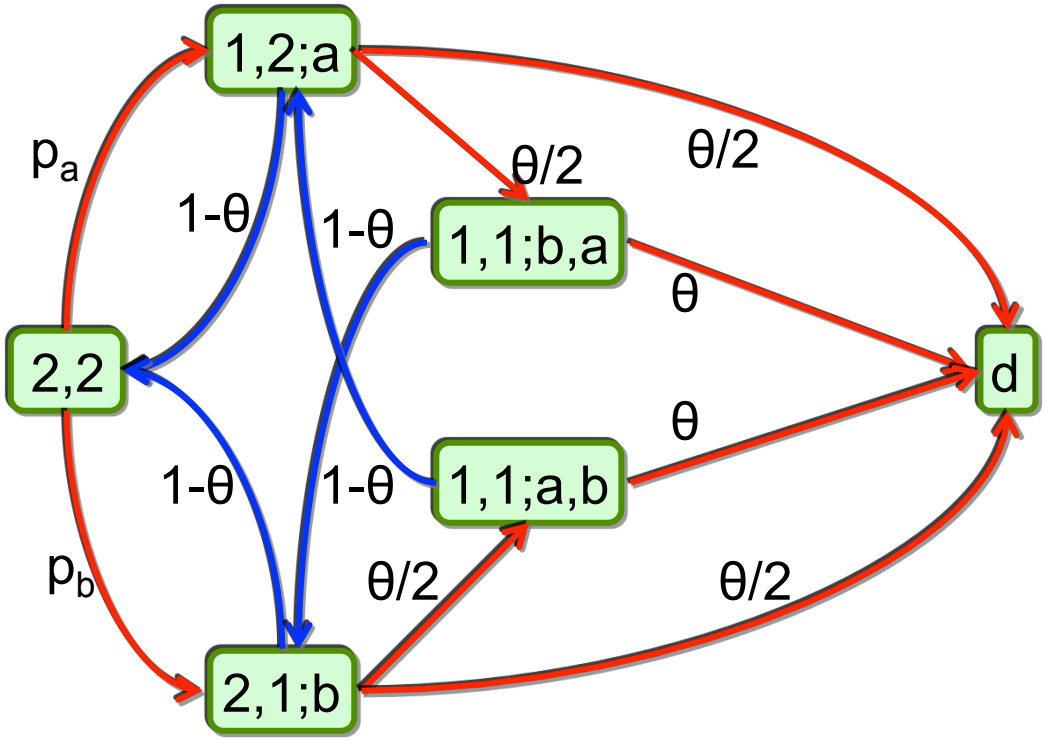
Simplest IS scheme: FB

- FB: Failure Biasing
  - replace each transition probability (in *Y*) corresponding to a failure (except if they start at state 0) by a constant θ and then scale all those starting from a given state
  - take $0.5 < \theta < 0.8$ (say)
  - for instance, from (1,2;a) to *d*: change q_a to θ/2, change q_b to θ/2



Simplest IS scheme: FB

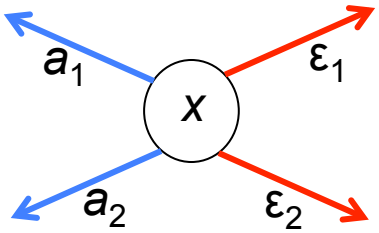
- FB: Failure Biasing:
  - replace each transition probability (in *Y*) corresponding to a failure (except if they start at 0) by a constant θ and then scale all those starting from a given state
  - take $0.5 < \theta < 0.8$ (say)
  - for instance, from (1,2;a) to *d*: change q_a to θ/2, change q_b to θ/2
  - result in this case:



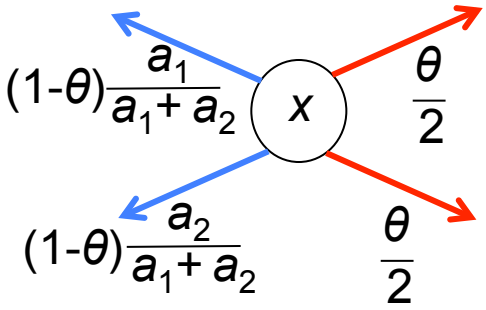
symbolically, for an operational state $x \neq 0$,

$0.5 < \theta < 0.8$ (say)

before



after



this is actually Balanced Failure Biasing,
a (better) variant of the original FB

Why it works

- Call Π the set of paths π of the form $\pi = (0,x_1,x_2,\ldots,x_m)$ where $x_1, x_2, \ldots, x_{m-1} \neq 0$ nor d, and x_m is 0 or d.
- Call Π_d the set of paths of Π ending at state d.
- γ = Pr(a generic path W belongs to Π_d).
- It is useful to write $$\gamma = \sum_{\pi \in \Pi} \Pr(\pi)\, 1(\pi \in \Pi_d),$$ which can also be written $$\gamma = \mathrm{E}\left[1\left(W \in \Pi_d\right)\right],$$
- For path $\pi = (0,x_1,x_2,\ldots,x_m)$, we have $$\Pr(\pi) = \Pr(W = \pi) = P(0,x_1)P(x_1,x_2)\ldots P(x_{m-1},x_m),$$ where P = t.p.m. of X.

- Crude Monte Carlo for the estimation of γ means building N independent copies $W_1,\ldots,W_N$ of W, and computing $\gamma^* = (1(W_1 \text{ in } \Pi_d) + \ldots + 1(W_N \text{ in } \Pi_d))/N$, which reduces to counting the good paths, as we have seen.
- Suppose we change the P(x,y) into a new $P'(x,y)$. For a path π, we write

$$\Pr'(\pi) = P'(0,x_1)\, P'(x_1,x_2) \ldots P'(x_{m-1},x_m).$$

- Write now

$$\gamma = \sum_{\pi\in\Pi} \Pr(\pi)\, 1(\pi \in \Pi_d) = \sum_{\pi\in\Pi} L(\pi) \Pr'(\pi)\, 1(\pi \in \Pi_d),$$

where L(π) = Pr(π) / Pr'(π) (called the likelihood ratio).

CTRQ 2017

- This leads to

$$\gamma = \mathrm{E}'\left[L(W)\,1\left(W \in \Pi_d\right)\right],$$

  where E' means that we use Pr' (that is, *P*').
- To derive a new estimator, we must build *N* independent copies $W_1,\ldots, W_N$ of *W*, **but using *P*' now**, and compute
  $\gamma^{IS} = (L(W_1)1(W_1 \text{ in } \Pi_d) + \ldots + L(W_N)1(W_N \text{ in } \Pi_d))/N.$
- This shows why γ^{IS} is another estimator of γ.
- If P' is chosen such that the rare event is now a frequent one, the new estimator is more efficient than the crude one (more efficient here means basically with a smaller variance). See the references.

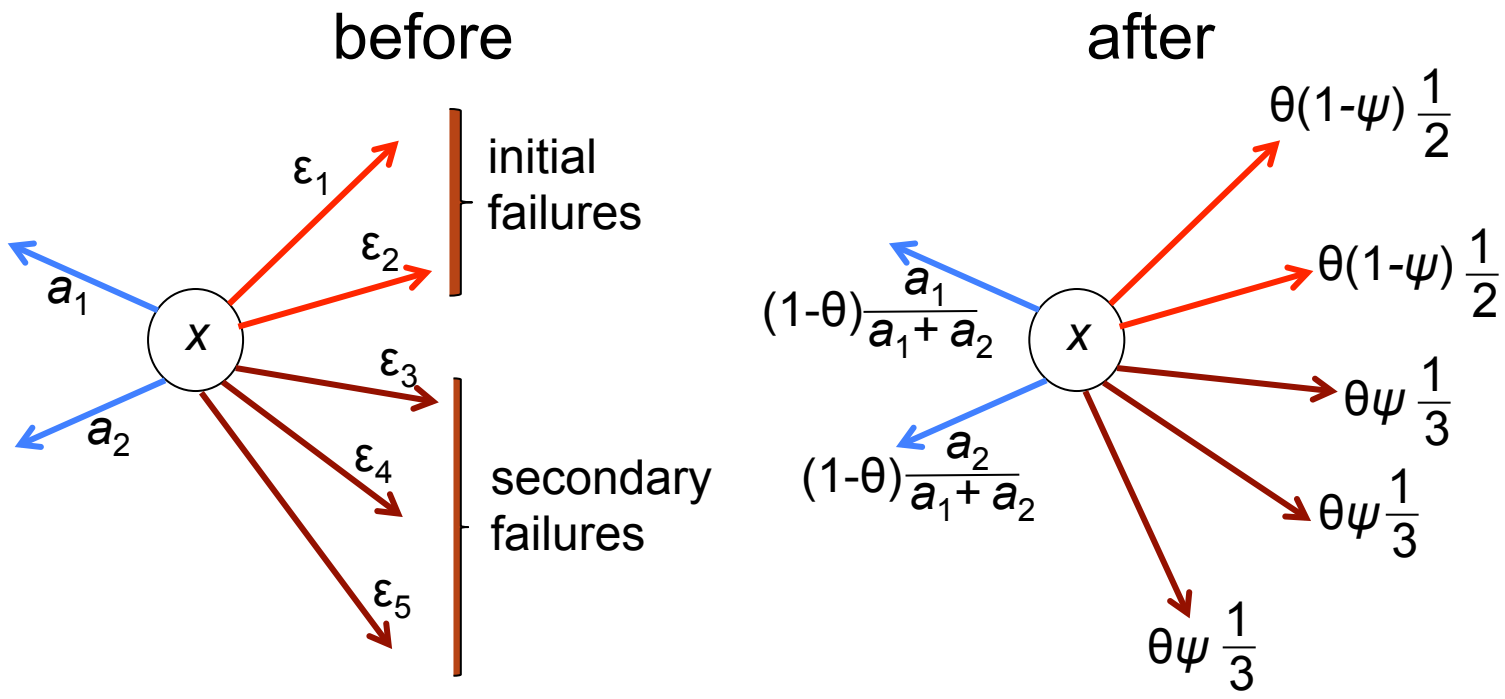
Improvement on FB

- Let us call **initial** a failure event consisting in the first failure in some class of components.
- Accordingly, a **secondary** failure is any other failure event.
- Intuitively, it seems a good idea to give more “weight” to secondary failures, expecting to reach *d* more quickly this way.
- This leads to the Selective Biasing scheme shown in next slide.

BSFB

symbolically, for an operational state $x \neq 0$,

$0.5 < \theta, \psi < 0.8$ (say)



this is actually Balanced Selective Failure Biasing, a good variant of SFB

More variants

- Other ideas have been published and shown to be effective (names are not "standardized"):
  - SFBS: SFB for Series-like systems
    - for systems composed of a set of modules working in series, or being "close enough" to that behavior
  - SFBP: SFB for Parallel-like systems
    - similar to SFBS but for systems composed of a set of modules working in parallel (or being "close enough" to that behavior)
  - DSFB: Distance-based SFB
    - for systems where it is possible to evaluate with almost no cost the distance from any up state to *d*
  - IDSFB: Inverse-Distance-based SFB
    - an improvement of DSFB
  - ISFB: Inverse SFB
    - a method based on IS designed for queuing overflow estimation
  - ...

Drawbacks, limits

- First, the change of transition probability matrix (more generally, the change of dynamics, also called “change of measure”) is not always easy to find.
- Second, if we move too much the process toward the rare area of the state space, we can do even worst than Crude Monte Carlo.

Current trends in IS

- **Adaptive IS**: here, the change of the dynamics is done on-the-fly, during the simulation process. The changes adapt to what is happening with the estimation in progress.
- **Zero variance approaches**: there is a theoretical optimal change of measure: it actually corresponds to exact evaluation, and it needs to know the value of the target! So, it is (apparently) useless…
- …However, we can find recursive expressions of this optimal (and out-of-reach) change of measure, and they can suggest new measures that, in some cases, behave very well. A very active research area today.

4 The splitting approach

- Here, the idea is quite different. I will introduce it using an example.
- Suppose the rare event is a queue overflow, and we want analyze the overflow phenomenon, leading to customer losses.
- (For a single queue we can perform an exact analysis in many cases, but the same ideas hold for networks of queues, where exact computation is out of reach).
- If we simulate trajectories of the queue occupation, for instance starting from an empty queue, the rare event situation will make that overflow is hard to observe; in many cases, we must wait too much time to see the first one.

- In IS, we will change the dynamics of the queue (i.e., the arrival and the service processes), in order to make overflow a frequent phenomenon.
- In Splitting, we define a set of M intermediate levels between level 0 (empty queue) to, say, level H (the queue storage capacity): $0 = l_0 < l_1 < l_2 < \ldots < l_M = H$.
- We start from level 0 and simulate N_0 trajectories, or paths. Since l_1 is much closer to 0 than $H = l_M$, some of these trajectories, say R_1, will (hopefully) reach next level l_1.
- Each of these R_1 paths will give birth to N_1 new continuations, of which R_2 will reach l_2.

- The paths that starting from 0 come back to 0 before reaching level l_2, and those that starting from l_1 come also back to 0 before reaching l_2, are killed.
- The ratio $p^*_1 = R_1 / N_0$ is an estimator of p_1, the probability for a path, starting at 0, to reach l_1 before coming back to level 0.
- Pursuing the same way, we come up with a set of estimators $p^*_1, p^*_2, \ldots, p^*_M$, whose product is an (unbiased) estimator of the overflow probability γ.

Some details

- It should be clear that, due to the splitting procedure, we have much more chances of observing an overflow, and even many of them.
- On the other side, we are (perhaps) paying too much CPU time for this, because of all those killed paths.
- Question: is the balance positive for us, that is, more efficient than Crude Monte Carlo?
- Answer: many times yes, it is, even thousands of times more efficient.
- The details and the proofs come once we have specified the procedure we follow to choose the parameters of the algorithm: how many levels, how many splits, static or dynamic, etc.

Drawbacks, limits

- In some cases, finding an **appropriate** number of levels and **a procedure for deciding how many paths must be created**, is very hard.
- The theory behind this approach is just starting, and it is not easy in general to analyze the variance reduction obtained from a specific splitting procedure (that is, the efficiency of the procedure).

Some references

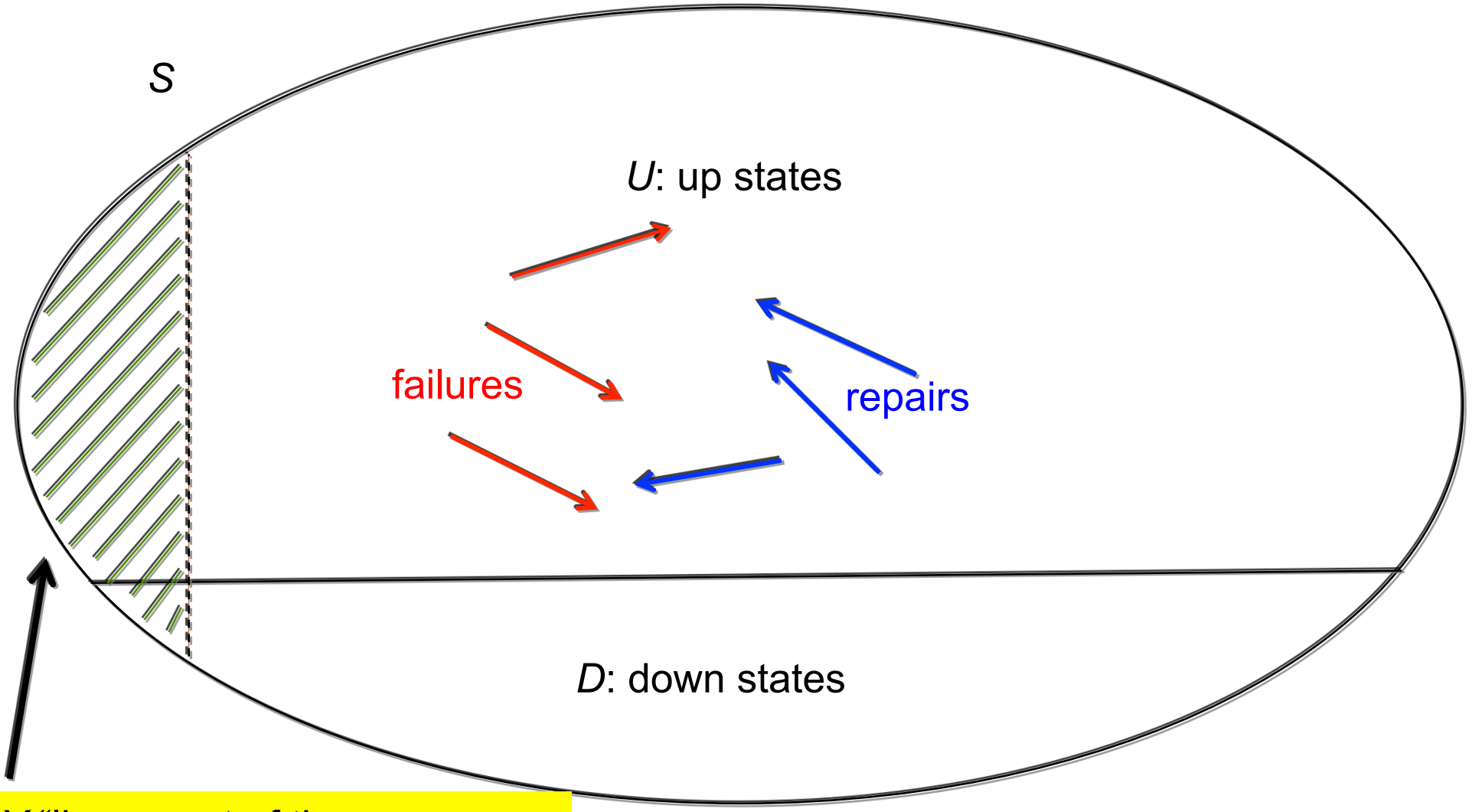
- Main references for author's work on rare event analysis through simulation:
  - "*On the application of accelerating simulation methods in network analysis*", by J. Incera and G. Rubino, in PDPTA'00: Parallel and Distributed Techniques and Applications, Las Vegas, June 2000
  - "*MTTF Estimation using Importance Sampling on Markov Models*", by H. Cancela, G. Rubino and B. Tuffin, in Monte Carlo Methods and Applications, 8(4): pp. 312-341, 2002
  - "*Markovian Models for Dependability Analysis*", by G. Rubino and B. Tuffin, Chapter 6 in *Rare Event Simulation using Monte Carlo*, edited by G. Rubino and B. Tuffin, WILEY, pp. 125-144, MARCH 2009

- "*Approximate Zero-Variance Importance Sampling for Static Network Reliability Estimation*", by P. L'Ecuyer, G. Rubino, S. Saggadi, B. Tuffin, IEEE Transactions on Reliability 60(3): 590-604, 2011
- "*A new simulation method based on the RVR principle for the rare event network reliability problem*", by H. Cancela, M. El Khadiri and G. Rubino, Annals of Operations Research, Vol. 196, pages 111-136, 2012
- "*A splitting algorithm for network reliability estimation*", by H. Cancela, L. Murray and G. Rubino, in IIE Transactions, Vol. 45, No. 2, pages 177–189, 2013
- "*Balanced and Approximate Zero-Variance Recursive Estimators for the Static Communication Network Reliability Problem*", H. Cancela, M. El Khadiri, G. Rubino and B. Tuffin, in ACM Transactions on Modeling and Computer Simulation, vol. 25, no. 1, 19 pages, December 2014

5 Bounding techniques

- This a complement with respect to the main topic, Monte Carlo methods for rare events.
- Here, we don't estimate but **bound** the metrics of interest.
- Let us illustrate the ideas with the evaluation of the asymptotic availability metric now, $A(\infty)$.
- We have an irreducible and homogeneous continuous time Markov chain X modeling system's evolution.
- The (very large) state space S is partitioned into
  - U, the states where system is up,
  - D, the states where system is down.
- $A(\infty) = \Pr(\ X(\infty)$ belongs to $U\ )$

Starting idea

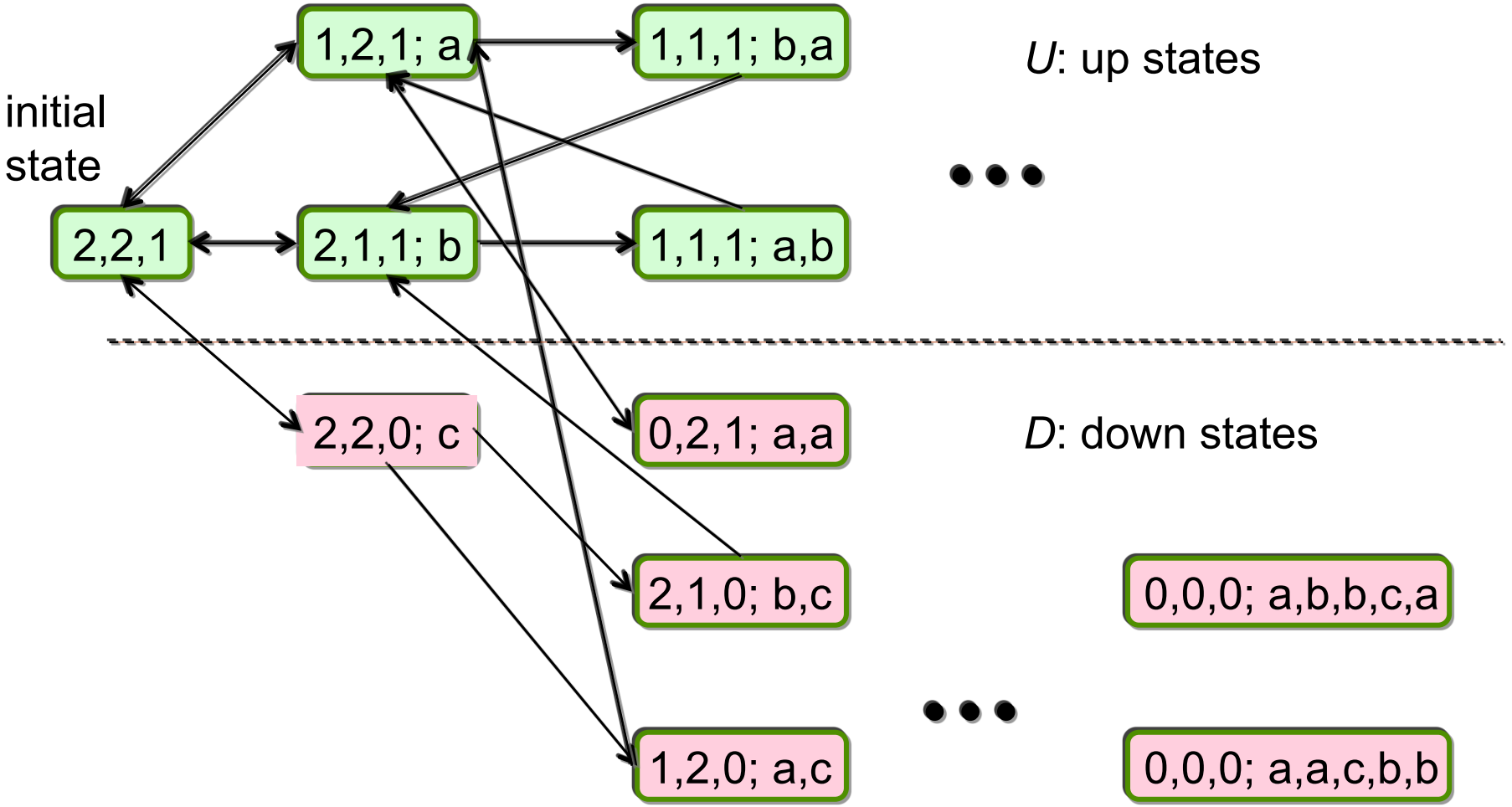


X "lives most of the time" here (states in subset ***G***)

An example

- Suppose there are 3 types of components: a, b, c.
- Initially, we have 2 a-units, 2 b-units, 1 c-unit.
- System works if there is at least one component working per class.
- There is a single repairman working in FIFO order.
- States:
  - initial state: (2, 2, 1)
  - other states: (n_a, n_b, n_c; state of the repair queue), where n_a (resp. n_b, n_c) is the number of type a (resp. type b, type c) working units
  - examples: (1, 2, 1; a), (1, 2, 1; a, a); (1, 0, 0; c, a, b, b)

A part of the associated graph



Main goal

- Build a new model X^* keeping the G-part of X and replacing the rest by “a few states”
- All the problem is: which states, how many, which transitions (and which transition rates) among them and between them and the states in G?
- In some cases, this can be done in such a way that computing specific metrics in model X^* we obtain an upper bound of the asymptotic availability $A(\infty)$ in X.
- Similarly, we can compute a lower bound of $A(\infty)$ using another small model following the same approach.

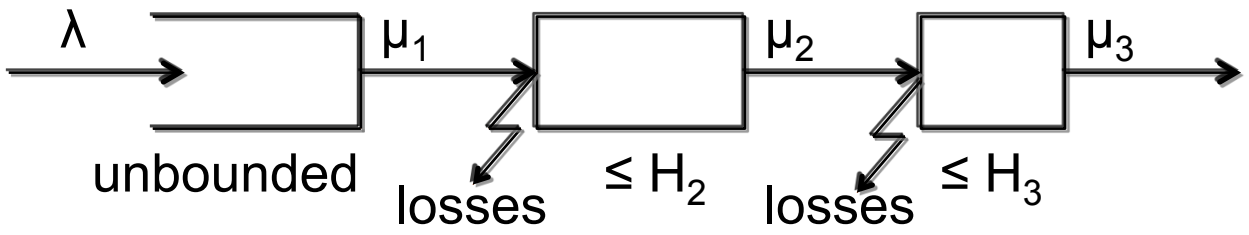
First, a couple of examples

- a communication network
- components are the network's links; two types:
  - primary links (80 units)
  - secondary links (120 units)
- network "supports" the failure of
  - 1 primary link
  - up to 5 secondary links
- 1 repair subsystem; repair times are type-dependent and modeled by Coxian laws
- after repaired, primary links need some time to be put back on operation; for secondary links, this delay can be neglected

- parameters' values:
  - failure rate of primary links: 0.00004
  - failure rate of secondary links: 0.00003
  - mean repair time of primary links: 1.0
    law: Erlang, 6 phases
  - mean repair time of secondary links: 1.0
    law: Erlang, 5 phases
  - post-processing of repaired primary links mean delay: 0.33
- **State space size: 4 344 921**
- Some results for the Asymptotic Availability:
  - keeping only 226 states (≤ 5 comp. down),
    - **0.999 759 7120 ≤ $A(\infty)$ ≤ 0.999 759 7471**
  - keeping 1826 states (≤ 10 comp. down),
    - **0.999 759 7349 ≤ $A(\infty)$ ≤ 0.999 759 7349**

An infinite state example

- an Internet path followed by a specific packet flow:



- λ, μ_1, μ_2, μ_3: rates
- **state space size: ∞**
- values: $\lambda = 0.2$, $\mu_1 = 0.7$, $\mu_2 = 1.5$, $\mu_3 = 0.2$, $H_2 = 18$, $H_3 = 10$
- important performance measures here include loss or blocking probabilities; no analytical solution available
- **after generating only 3839 states, absolute errors in all performance measures < 10^{-9}; for blocking probabilities, the absolute errors were < 10^{-10}**

How it works

- Start by partitioning the state space *S* of Markov process *X* into classes.
- Class *C*(*k*) is, for instance, composed of the states in which there are *k* components down in the system (see slide 48).
- Usually, *C*(*0*) = { 0 }.
- Now, suppose for instance that from each *x* in *C*(*k*) there is at least a failure transition (to some state *y* in *C*(*k+1*)) and a repair transition (to some *z* in *C*(*k-1*)).
- Then, the global transition rate from *C*(*k*) to *C*(*k+1*) is necessarily upper-bounded by max{ total failure rate out of *x* in *C*(*k*) }. The same for lower bounds, and for repair rates.

- We keep the states “on the left” of S unchanged: we set $G = C(0) \cup C(1) \cup \ldots \cup C(k)$, and replace each subset $C(j)$, $j > k$, by a single state c_j.
- Then, using the previous mentioned bounds, we can build new models having sizes of the order of $|G|$, where the computation of the targeted metric provides bounds of the corresponding metric in the original model X.
- As the event of interest (e.g. reaching the set D of down states before coming back to 0) becomes less probable, the mass concentrates more on the left, and for the same G the bounds are tighter (or we need smaller G to achieve the same accuracy).

Drawbacks, limits

- The technique is quite general, but if it is not true that from any state in any class, there is at least a failure transition and a repair one, then computing the intermediate bounds can be difficult, or expensive, or simply impossible.
- In the case everything works, it can happen that *G*'s size is (too) large for achieving some specific accuracy level, and thus the computation of the bounds (too) expensive.
- This approach has, as the other ones, lot of room for improvements. Some have already been published.

References

- Main references for author's work on bounding techniques:
  - "*Bound computation of dependability and performability measures*", by S. Mahevas and G. Rubino, IEEE T. on COMPUTERS, VOL. 148, NO. 3, pp. 188-196, JUNE 2001
  - "*Bounding the Mean Cumulated Reward up to Absorption*", by A. de Couto e Silva and G. Rubino, in *Markov Anniversary Meeting*, edited by A. Langville and W. Stewart, BOSSON BOOKS, pp. 169-188, 2006

  The first one concerns asymptotic availability and extensions to reward-based metrics (irreducible models). The second one concerns the MTTF and extensions to reward-based metrics (absorbing models).

6 Summary

- Rare event analysis is crucial for **critical system**'s design, analysis, management, …
- Accurate representations of complex systems often lead to very large models, so, to the use of simulation.
- Critical systems mean rare event analysis, and rare event analysis through direct simulation is hard, and often impossible.
- Main point of the presentation:
  - sophisticate simulation techniques exist to analyze rare events efficiently, and they are currently being improved
  - bounding techniques also exist for dealing with the same situation

- For simulation, model's size has little impact on the efficiency of the techniques; the bottleneck is the rarity of the events of interest.
- Simulation techniques need specific conditions to be applied, but there are many techniques, and some of them can be applied to wide ranges of systems.
- Bounding techniques have also conditions for their use, more restrictive than those for simulation methods. But some ideas can be used in many cases with extremely high efficiency (the technique can be seen as belonging to the numerical analysis field in these cases).

Missing in the presentation

- Inhomogeneous models
  - components' behavior evolves with time
- … combined with models with local clocks
  - there are "local times" together with the global one
- … using semi-Markov processes, and *k*-order Markovian ones
- … combined with models handling dependency between components
  - the behavior of some components depend on what happens to other parts of the system

- Things are much more complex here and just a few ideas are being developed.

The judoka's principle

- Rarity impacts standard simulation procedures "monotonically": the rarer the event of interest, the harder its analysis.
- For Importance Sampling or for Splitting, often the rarer the event, the more efficient the method.
- In bounding techniques, the situation is the same: the procedures work much better when the target has very small probabilities.
- The judoka's principle: "use the strength of the opponent": the rarer the event, the more efficient the techniques described here to analyze it.

CTRQ 2017

A current project

- Bounds and Monte Carlo techniques can be combined. This has been done in some particular cases with great success.
- One project in preparation in Europe, under my direction, consists in developing this line in very general contexts.
- This is being prepared in coordination with some players on the application side: in energy production (for nuclear risks) and in avionics (risks related to crashes, for instance). Door is open for other participants.
- The project includes the exploration of some of the extensions previously mentioned (slides 14 and 59).

So, only positive messages?

- With more time, two topics that deserve specific attention:
  - financial systems and rare events
  - software-based systems and rare events
- In these areas, there are very specific problems to solve, in particular in the second, where some structural problems seem very hard to address. See
  - "*The Infeasibility of Quantifying the Reliability of Life-Critical Real-Time Software*", by Ricky W. Butler and George B. Finelli, IEEE T. on SOFTWARE, VOL. 19, NO. 1, JAN 1993

Gerardo RUBINO

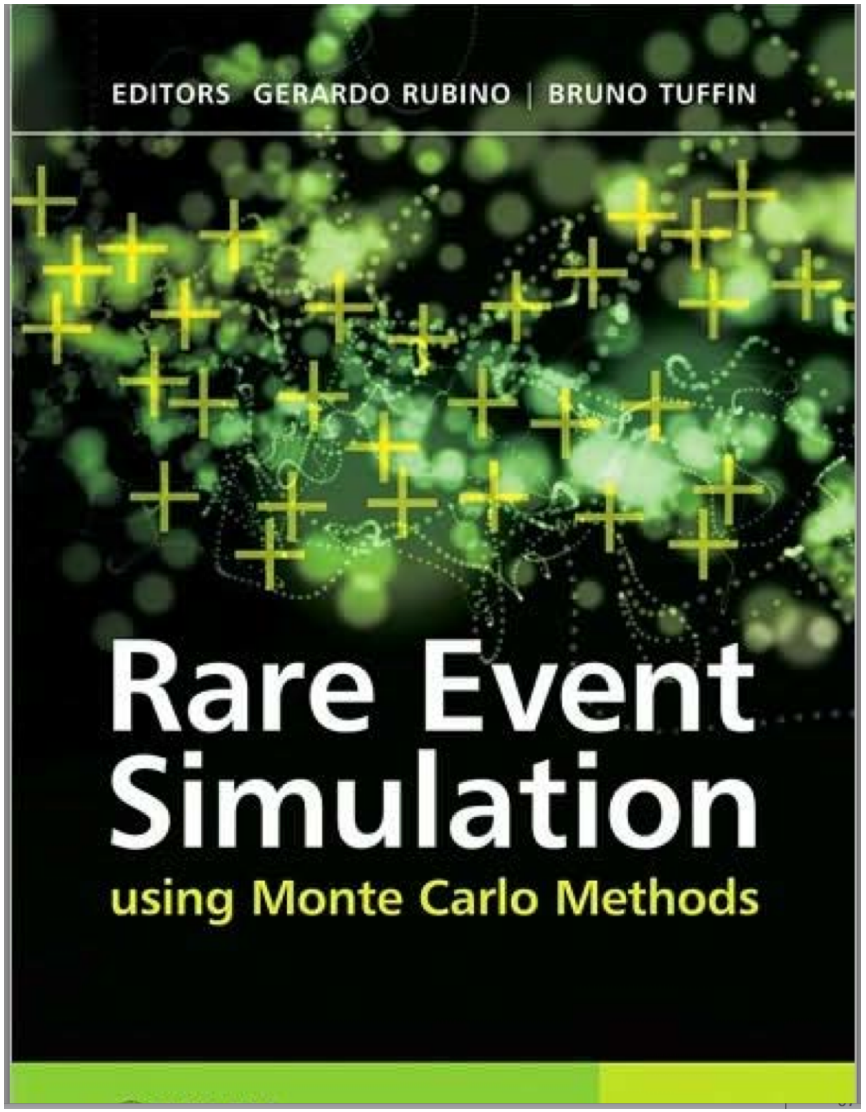
- PhD in Computer Science
- Senior researcher at INRIA, France
  - performance evaluation
  - dependability evaluation
  - quality assessment
- From the scientific foundations viewpoint:
  - stochastic models
  - Markov processes
  - analytical, numerical and simulation techniques for Markov models' analysis
  - main extensions to Markovian models

Where to Get More Information

- Main references: the two following books, the first on simulation, the second on analytical and numerical techniques, including bounding procedures:
  - "*Rare Event Simulation using Monte Carlo Methods*", edited by G. Rubino and B. Tuffin, Wiley, 2009
  - "*Markov Chains and Dependability Theory*", by G. Rubino and B. Sericola, Cambridge U. Press, 2014
- They contain many references on the topics of this presentation, and on related ones (for instance, on Markov chain analysis using numerical procedures, as well as on related theoretical results).

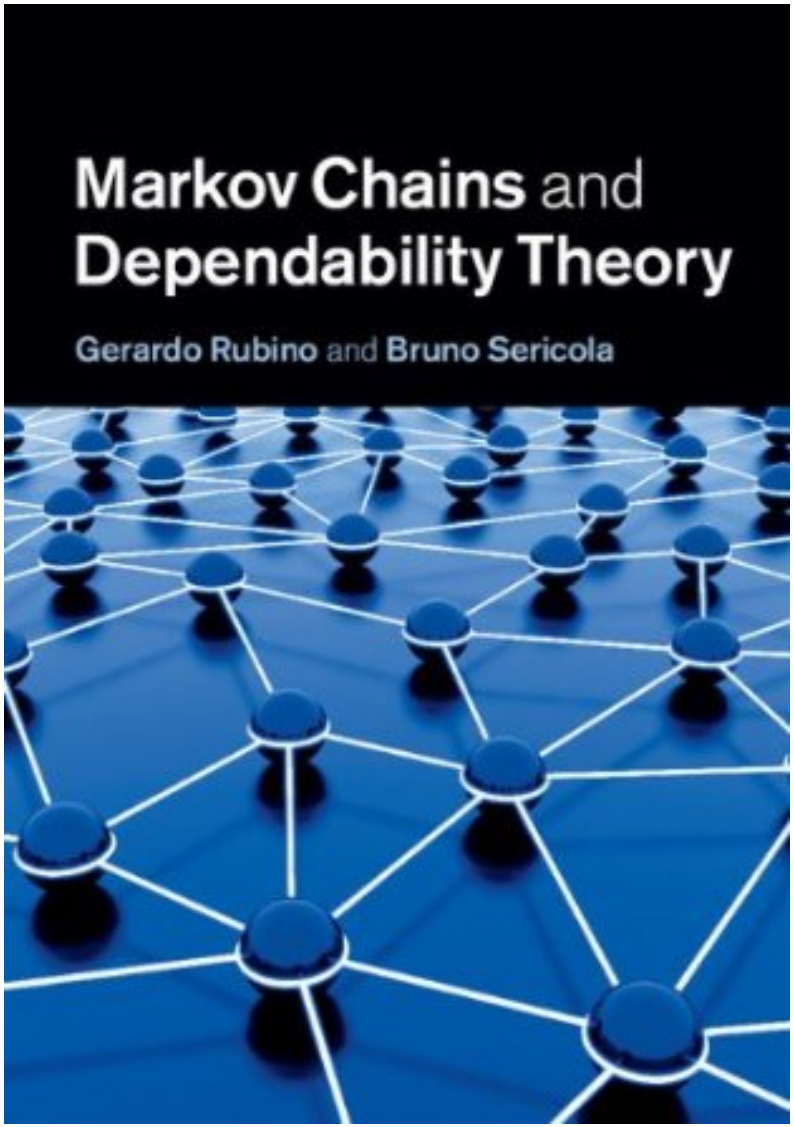
For Monte Carlo issues, see

G. Rubino, B. Tuffin (editors and co-authors of more than half of the chapters),
"Rare Event Simulation Using Monte Carlo Methods"
John Wiley & Sons,
271 pages, 2009.



For bounds and other mathematical aspects, see

G. Rubino, B. Sericola
"Markov Chains and Dependability Theory"
Cambridge University Press,
278 pages, 2014.



CTRQ 2017

Questions

Thank you for your attention.

Do you have any questions?