# 1/
# HRMS: IS examples

# MODEL

- Very used in dependability analysis of complex systems
- Simplified version and description here
- System is composed of components belonging to $C$ classes
- Components are either up or down
- The whole system is also either up or down
- Failures and repairs (of any component) are exponentially distributed
- CTMC $Y = (Y_1,\ldots,Y_C)$ where $Y_c$ is the number of up components in class $c$

- There is a (structure) function saying for any value of $Y$ if the system's state is up or down
- Let $\Delta$ be the set of down states
- Let **1** be the state where all components are up, the initial state of $Y$
- Goal: evaluate $\mu = \Pr($ hitting $\Delta$ before $\mathbf{1}$ $)$
- This is an important metric in dependability analysis (and also in performance issues)

- Usual situation: at some time scale, failure rates are in $O(\varepsilon)$ and repairs in $O(1)$
- To estimate $\mu$, we can
  - move to the canonically embedded DTMC
  - and use the standard Monte Carlo approach
- In the DTMC, failure probabilities are in $O(\varepsilon)$ (except for state **1**) and repair ones in $O(1)$
- Idea: use Importance Sampling

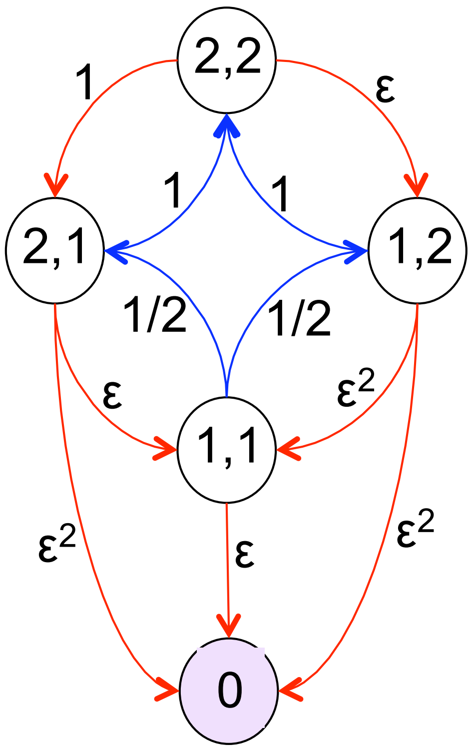
# Failure Biaising (FB)

- Also sometimes called Simple Failure Biaising
- Probably the simplest idea
- Idea: just push the system towards $\Delta$ by changing the (small) failure probabilities into something “not small”
- For any state $x \neq \mathbf{1}$ in the DTMC,
  - we change the total (the sum) of the corresponding failure probabilities to some constant $q$ (typically $0.5 < q < 0.8$)
  - and we distribute it proportionally to the original values

# FB example

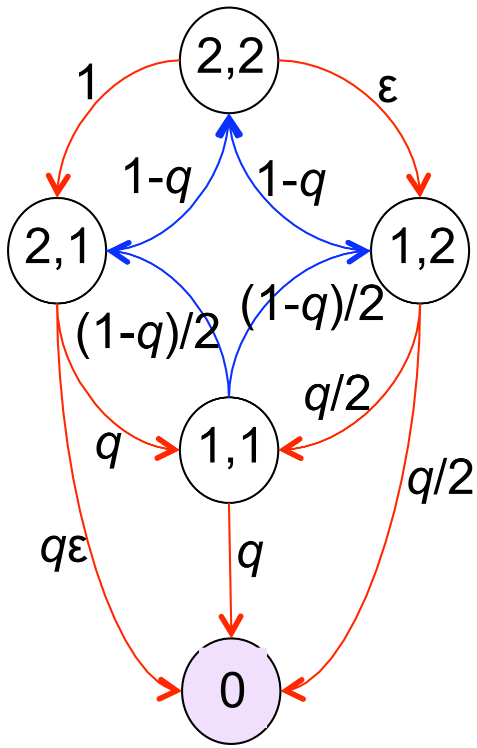
first order (in ε) expressions

before




**1** = (2,2)

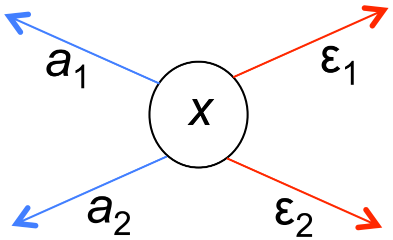
after



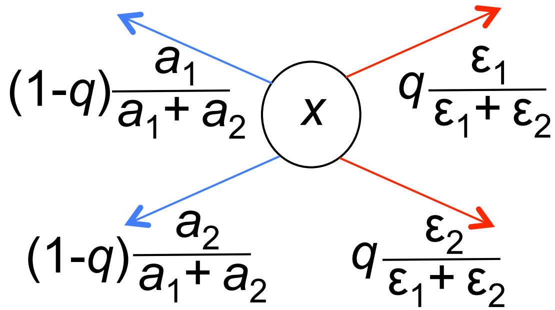
# FB

symbolically, for an operational state $x \neq \mathbf{1}$

before

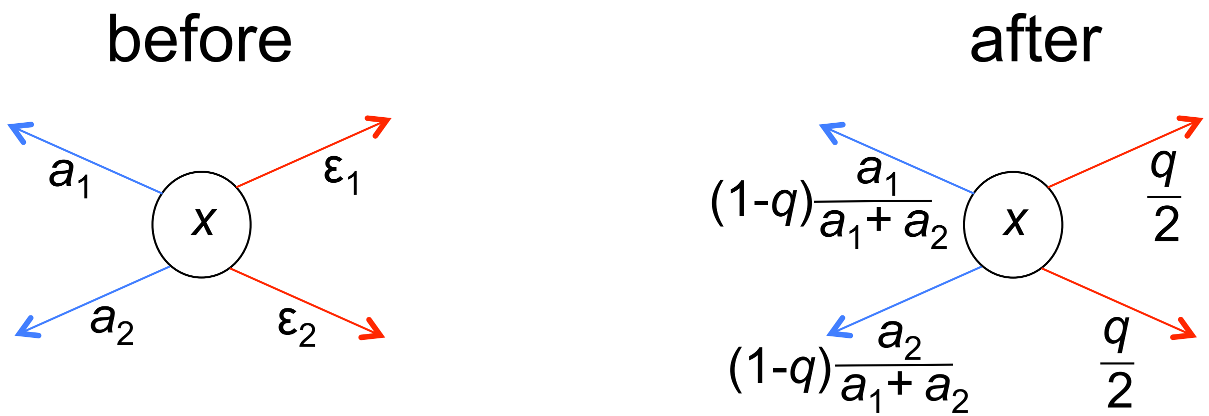



after




This technique works but
it does not have the BRE property

# Balanced FB



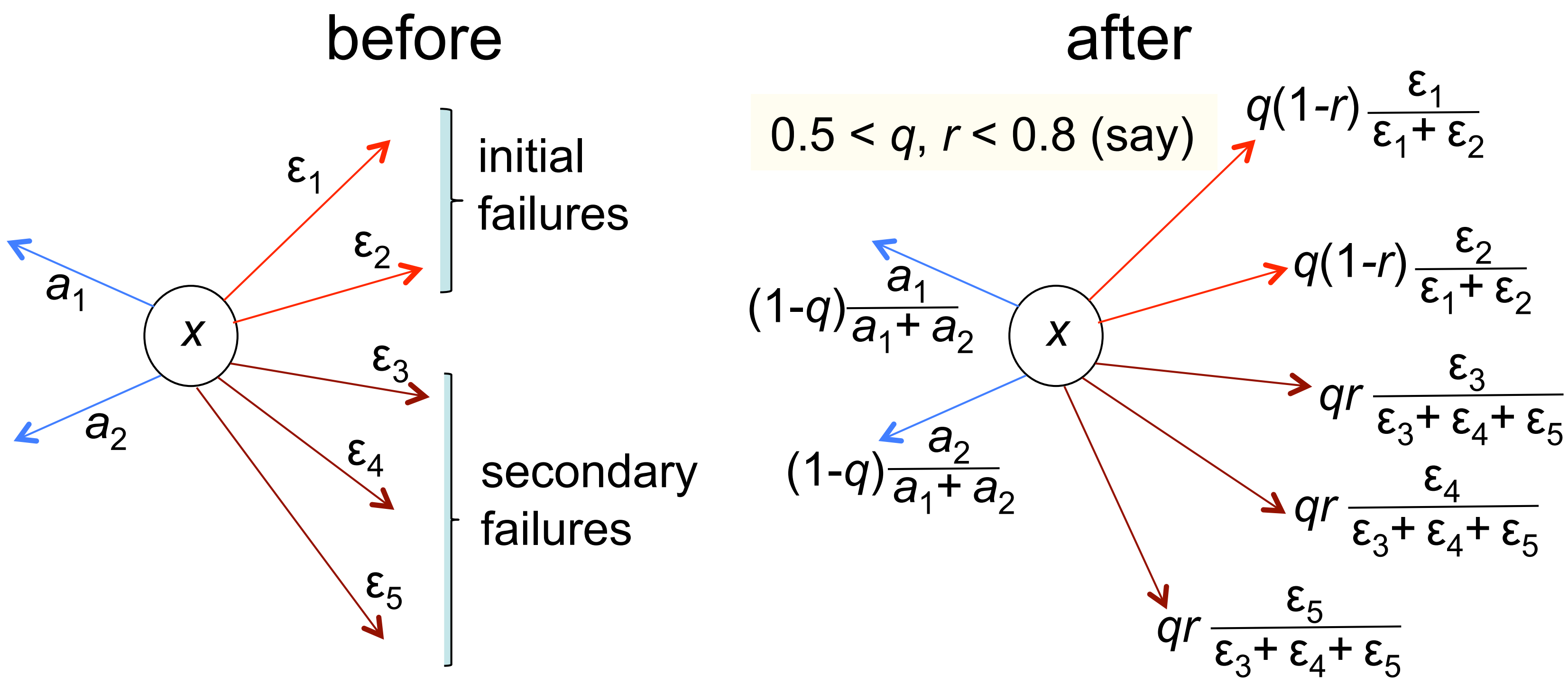

This version does have the BRE property

# Selective FB (SFB)

- 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 set $\Delta$ quicker this way.
- This leads to the Selective Biasing scheme shown in next slide.

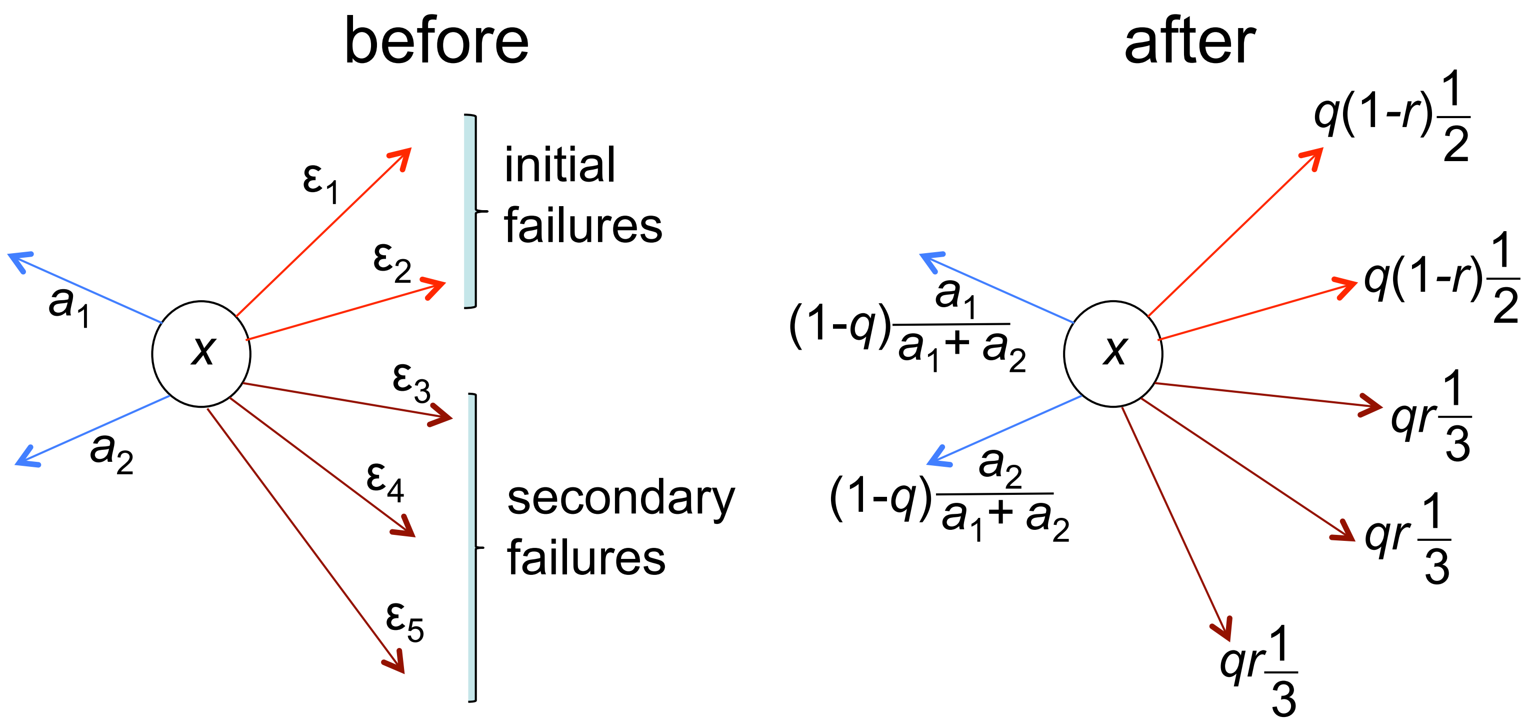
# SFB

symbolically, for an operational state $x \neq \mathbf{1}$




This technique works but
it does not have the BRE property

# Balanced SFB




This technique does have the BRE property

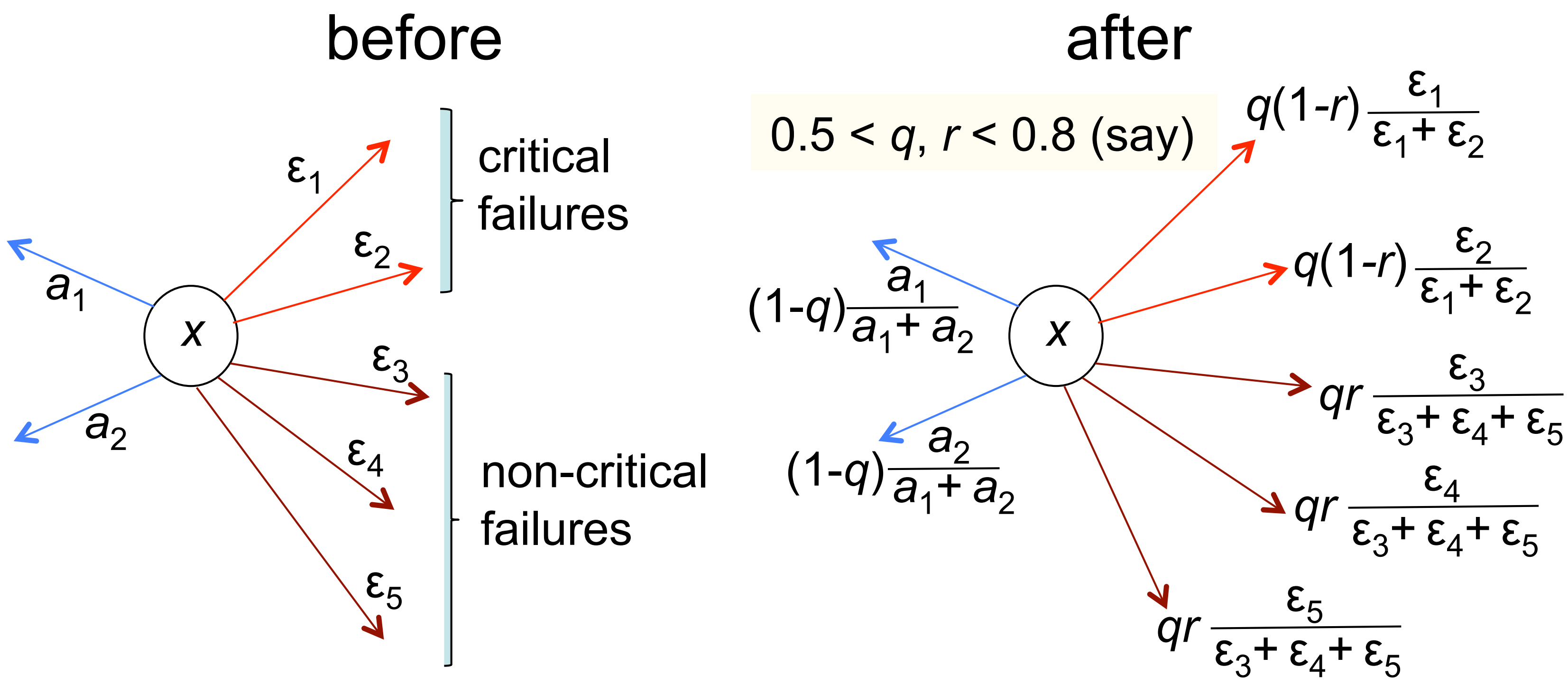
# SFBS: SFB for Series-like sys

- Always the same idea: take advantage of available information, if possible.
- Here, we assume that not only we know if failures are initial or not, but also, we know that the system's structure is "series-like".
- The typical example is a series of $k$-out-of-$n$ modules. Denote the parameters of module $i$ as $k_i$, $n_i$.
- We define as critical the failure of a component in a module $i$ such that after it, we have

$$Y_i - k_i = \min_j (Y_j - k_j).$$

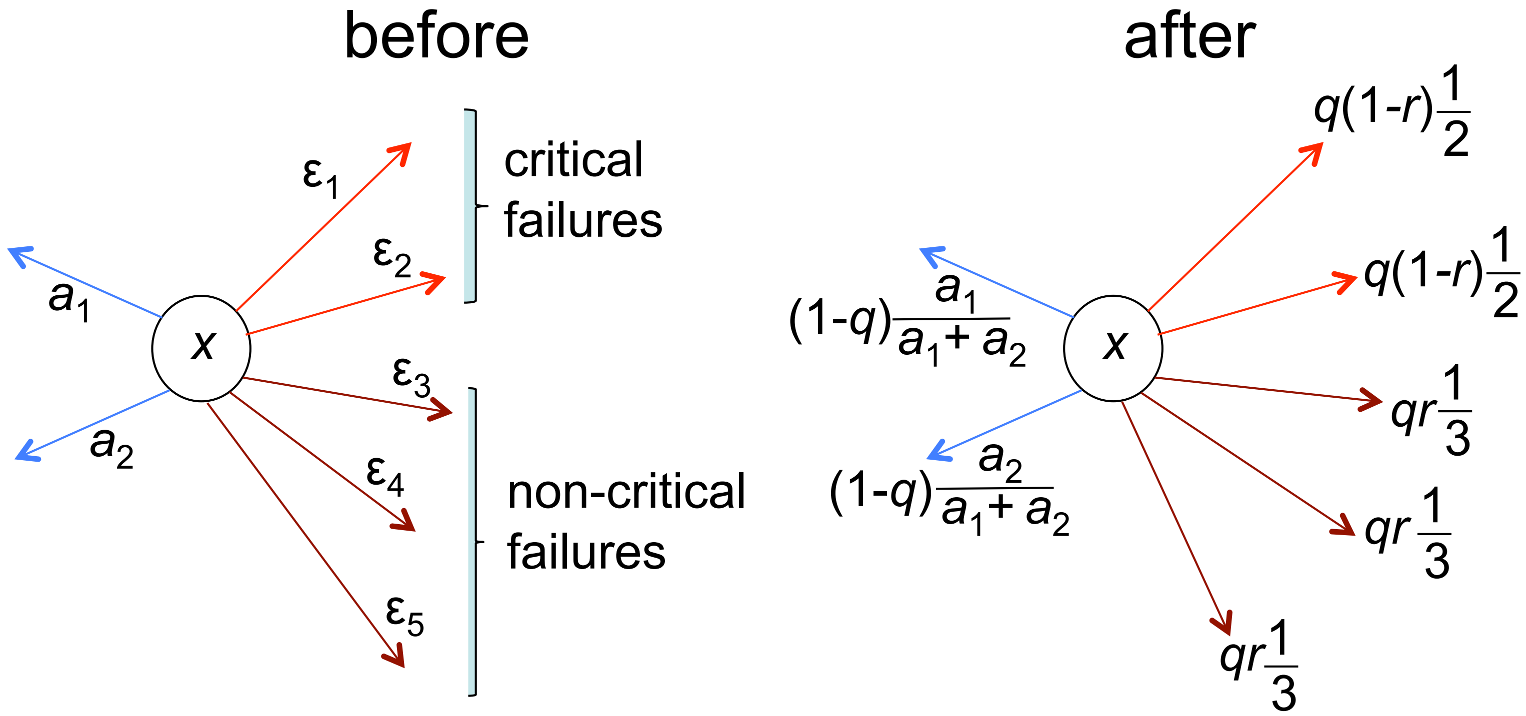
# SFBS

symbolically, for an operational state $x \neq \mathbf{1}$




This technique works but
it does not have the BRE property

# Balanced SFBS




This technique does have the BRE property

# Other ideas

- Other ideas have been published and shown to be effective (names are not "standardized"):
  - SFBP: SFB for Parallel-like systems
    - similar to SFBS but for systems composed of a set of modules working in parallel
  - DSFB: Distance-based SFB
    - for systems where it is possible to evaluate with almost no cost the distance from any up state to $\Delta$
  - IDSFB: Inverse-Distance-based SFB
    - an improvement of DSFB
  - IFB: Inverse SFB
    - a method based on the optimal IS c.o.m. for the M/M/1 queuing model
  - and others…

# Zero variance

- The zero variance idea in IS leads to very efficient methods for this HRMS family of models.
- Recall that in this approach, we consider the probability $\mu_x$ of hitting $\Delta$ before $\mathbf{1}$, starting from any state $x$ of the chain.
- Some existing results:
  - methods with BRE
  - even "vanishing relative error" obtained, when the approximation of $\mu_x$ can use all paths from $x$ to $\Delta$ with smallest degree in $\varepsilon$.

# Zero variance

- Numerical results so far are impressive with these techniques (in terms of efficiency and also of stability).
- For instance, in a model with 20 classes of components, with 4 components per class, and the working criterion "system is up iff at least 7 components are up", a typical result is

| METHOD | BFB | 0 var |
|---|---|---|
| μ | $3.1\ .10^{-11}$ | $3.0\ 10^{-11}$ |
| Var | $8.5\ 10^{-18}$ | $1.3\ 10^{-24}$ |
| CPU time | 11 sec | 97 sec |

# 2a/
# A RECURSIVE VARIANCE REDUCTION TECHNIQUE FOR STATIC PROBLEMS

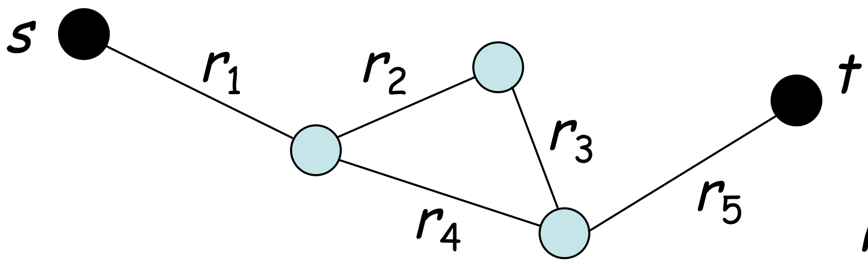
# ABSTRACT

- A variance-reduction procedure for network reliability estimation
- Main ideas:
  - exploit the specificities of the problem (graph theory concepts, binary coherent structures)
  - improve efficiency by including exact computations into the Monte Carlo procedure
- More specifically:
  - a recursive decomposition-based approach
  - using two basic concepts in the considered family of systems: paths and cuts

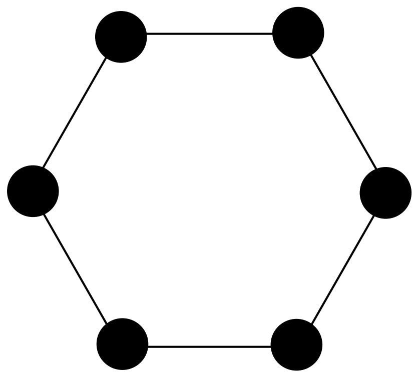
# OUTLINE

- introduction
- some graph concepts
- the method
- some results

# 1 NETWORK RELIABILITY




$$R_{st} = r_1 (r_2 r_3 + r_4 - r_2 r_3 r_4) r_5$$

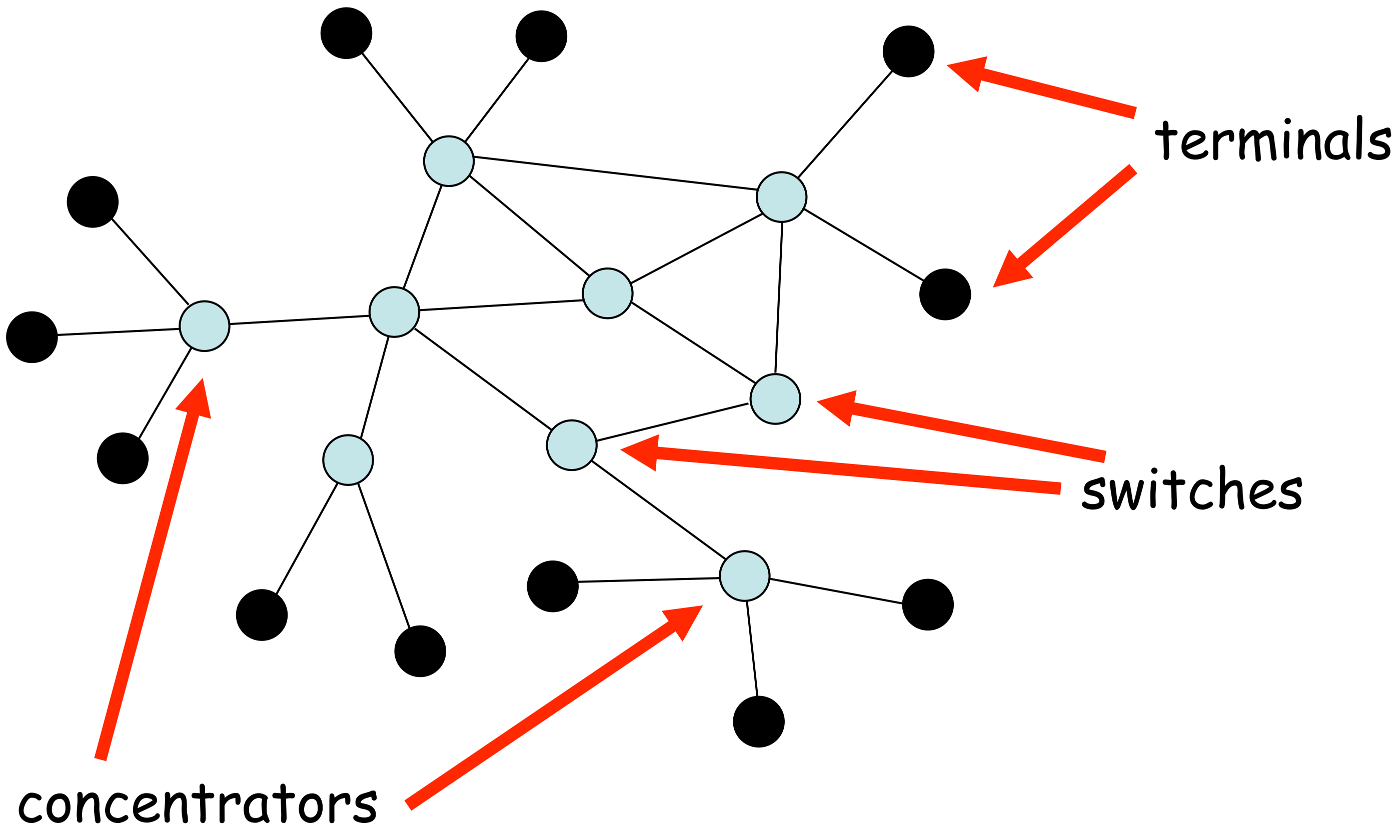


$$r_i = r$$

$$R_{all} = r^5 (6 - 5r)$$

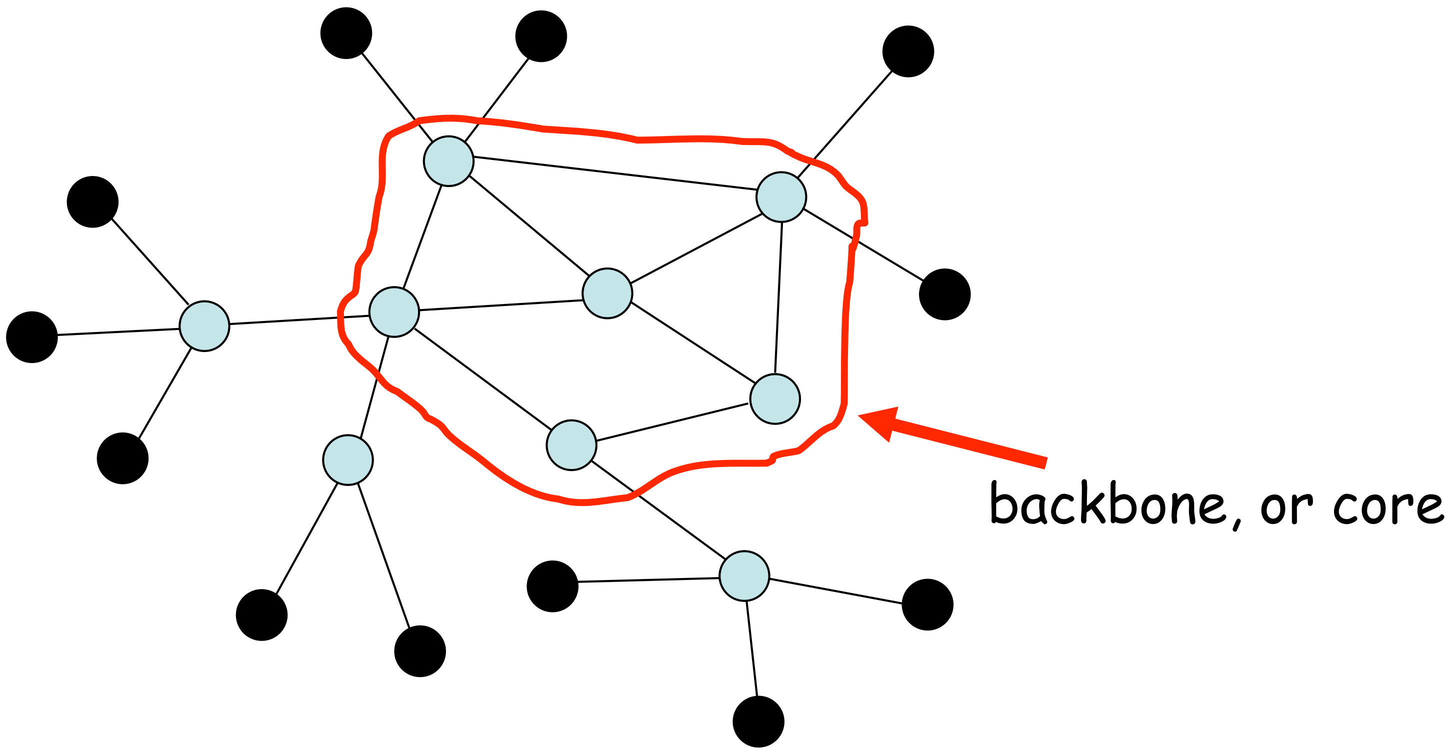
# REFERENCE MODEL

## A COMMUNICATION NETWORK

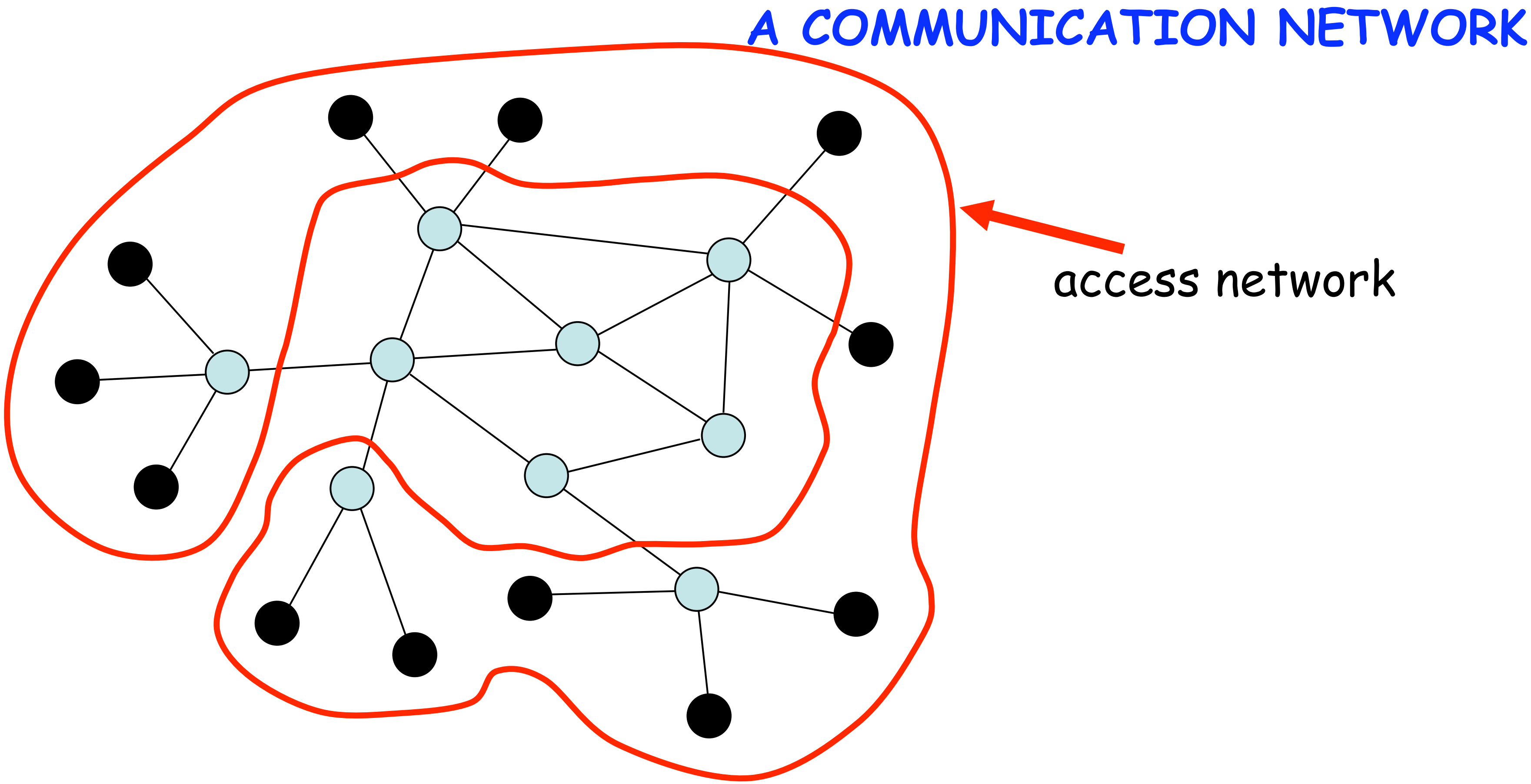


# REFERENCE MODEL

## A COMMUNICATION NETWORK

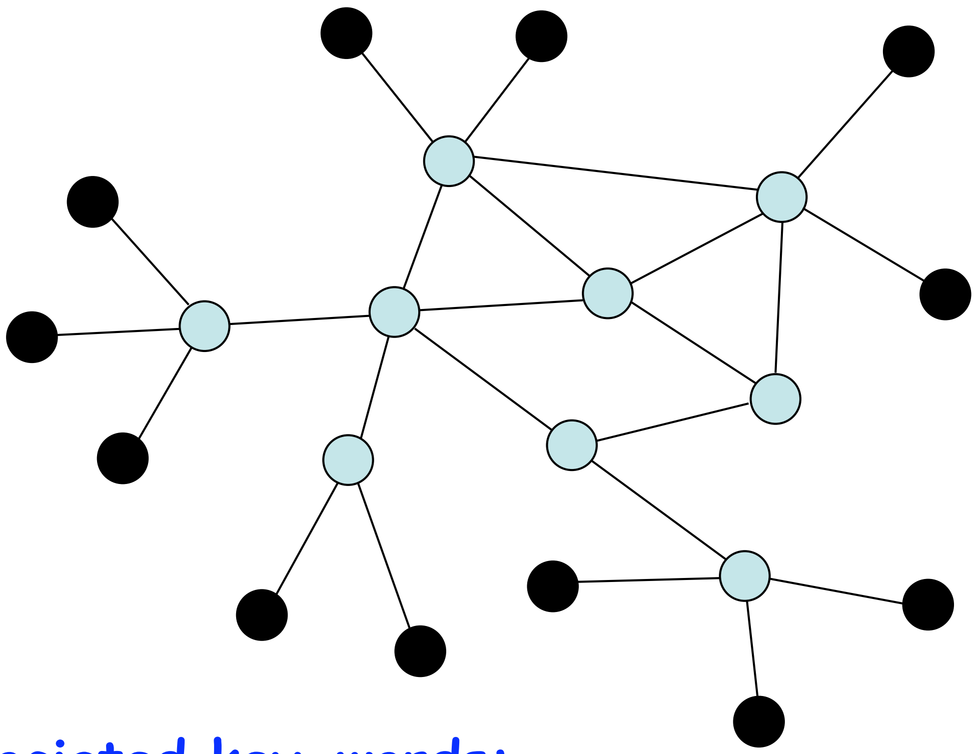


# REFERENCE MODEL



# REFERENCE MODEL

### A COMMUNICATION NETWORK



- nodes are perfect
- lines behave independently
- lines are up or down
- for each line $i$,
  $r_i$ = Pr(line $i$ is up)

**Associated key-words:**

- **reliability diagrams, fault-trees…**
- **graph theory, coherent binary structure theory**

# MODEL

- $V$: the nodes
  $K$: the *terminals*, or target-set, $K \subseteq V$
  $E$: the *lines* or edges
  $\{r_i\}_{i \text{ in } E}$: the *elementary reliabilities*
- $N = (V, E)$: (the underlying) undirected graph (also called the *network* when we include the probabilities associated with the edges)

# RANDOM STRUCTURE

- $\Omega$: set of all partial sub-graphs of $N$ (same nodes, part of the edges)
- $G = (V, F)$: a random graph on $\Omega$; probabilistic structure:

  for any $H \subseteq E$,

$$\Pr(G = (V, H)) = \prod_{i \in H} r_i \prod_{j \notin H} (1 - r_j)$$

# METRIC

- goal: $R$ = $K$-network reliability,
  = Pr(the nodes in $K$ are connected)
  (or equivalently $Q = 1 - R$)
- $U$: set of all partial sub-graphs of $N$ where all nodes in $K$ are connected; thus,
  $R = \Pr(G \text{ in } U)$
- usual situation: $R \sim 1$, that is, $Q \sim 0$
- only MC can handle medium/large models; but possible problem:
  the rare event situation

# 2 STANDARD MC

- #failed = 0
- **for** m = 1, 2, ..., M
  - $g = \text{sample}(G)$
  - **if** $g \notin U$ **then** #failed += 1
- $Q^{std}$ = #failed/M
- $V^{std} = Q^{std}(1 - Q^{std})/(M - 1)$

→ The rare event problem when $Q \ll 1$

# COMPUTATIONAL COMPLEXITY

- Internal loop: sampling a graph state (state of each edge), and verifying if it belongs or not to set $U$ (DFS search); total complexity is $O(|E|)$.
- $M$ iterations; initialization time and final computations in $O(1)$.
- Total computation time in $O(M|E|)$, linear in # of edges and # of replications.

# ACCURACY

- the correct answer given by the MC method is not "unreliability is $Q$" but "unreliability belongs with high probability to $(Q1, Q2)$"
- more precisely, here is a possible output "routine":
  - $\Pr( Q \in (Q1, Q2) ) \approx 0.95$
  - $Q1 = Q^{std} - 1.96\, V^{std}, \qquad Q2 = Q^{std} + 1.96\, V^{std}$
  - $V^{std} = \text{StandardEstimator}(\text{Variance}(Q^{std}))$

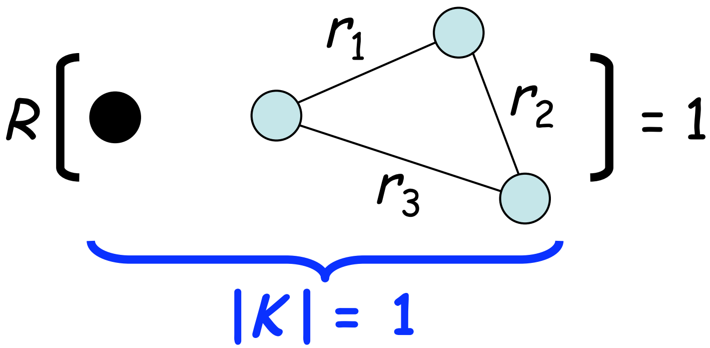
    $= [Q^{std}(1 - Q^{std})/(M - 1)]^{1/2}$

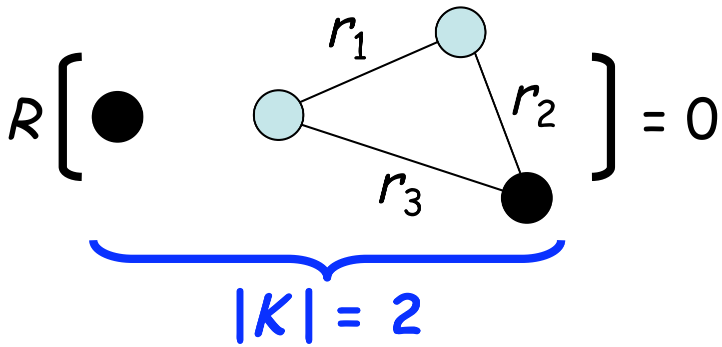
- relative error in the answer:
  $\text{RelErr} = 1.96\, V^{std} / Q^{std}$
  $= [(1 - Q^{std})/((M-1)\, Q^{std}]^{1/2}$
  $\approx 1/(MQ^{std})^{1/2}$
  which is problematic when $Q^{std} \ll 1$
- Variance Reduction Techniques: estimation methods such that the variance of the estimators (and their own estimators) are smaller than the variance of the crude estimator.

# "SOME TRIVIAL CASES"

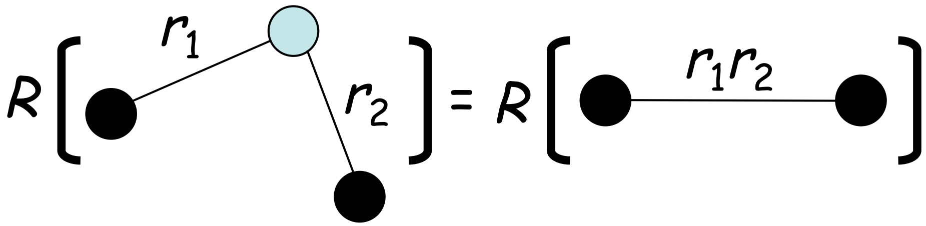
$$\underbrace{R[\bullet]}_{|K|=1} = 1$$

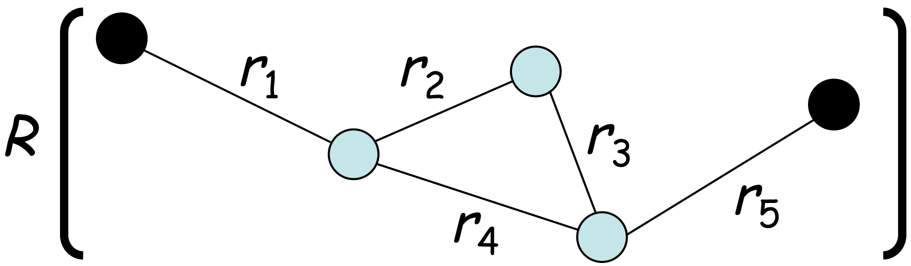
$$\underbrace{R[\bullet\!-\!\!-\!\!-\!\bullet]}_{|K|=2} = r \times R[\bullet] = r$$

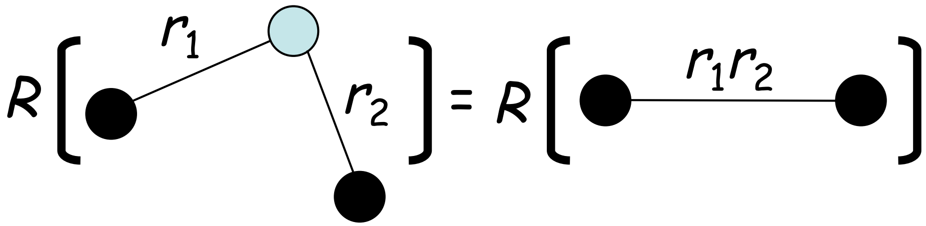


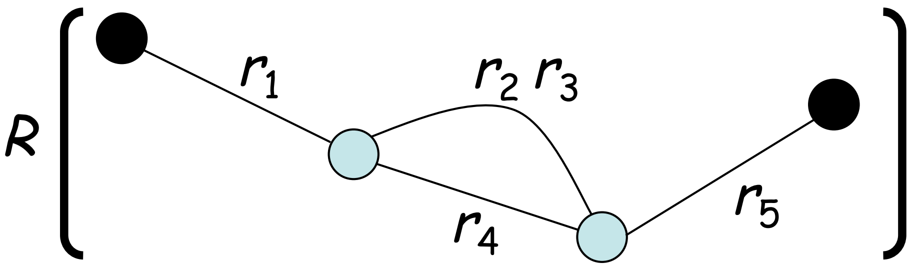



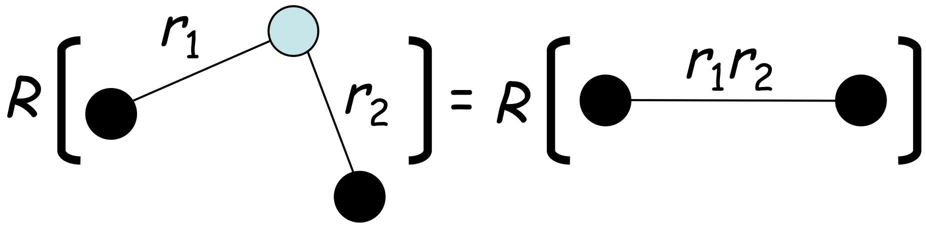
# 3 SERIES-PARALLEL REDUCTIONS

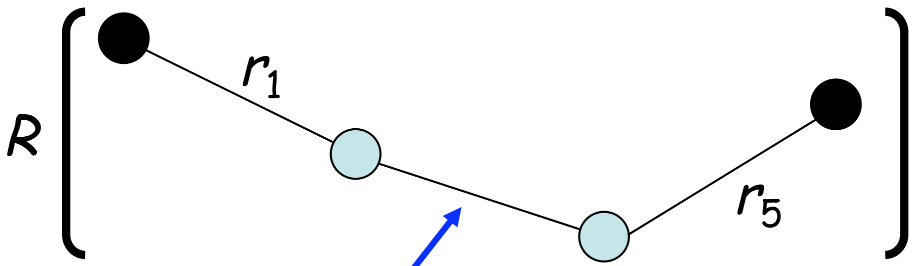



R
r1
r2
= R
r1r2

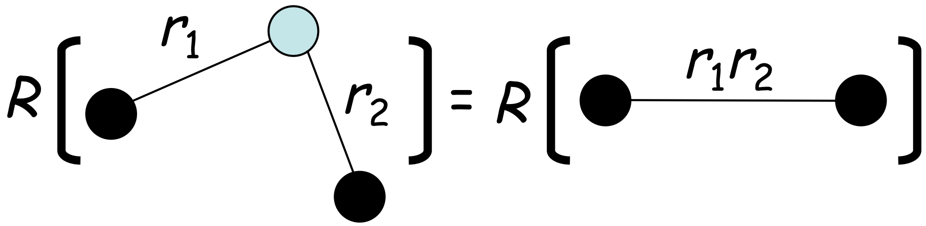

R
r1
r2 r3
r4
r5

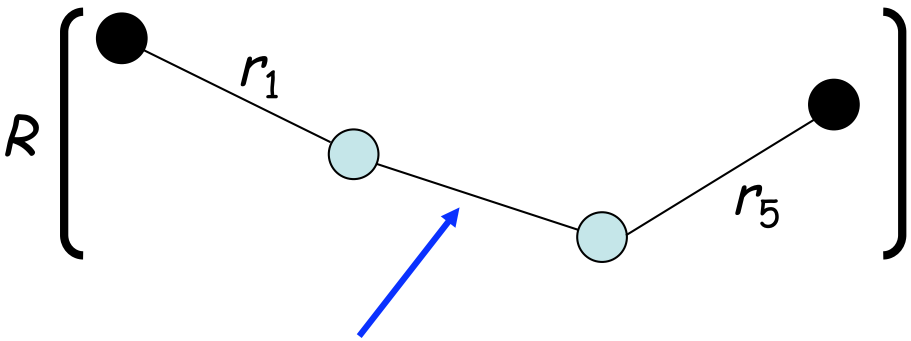



etc.

$r_2\, r_3 + r_4 - r_2\, r_3\, r_4$



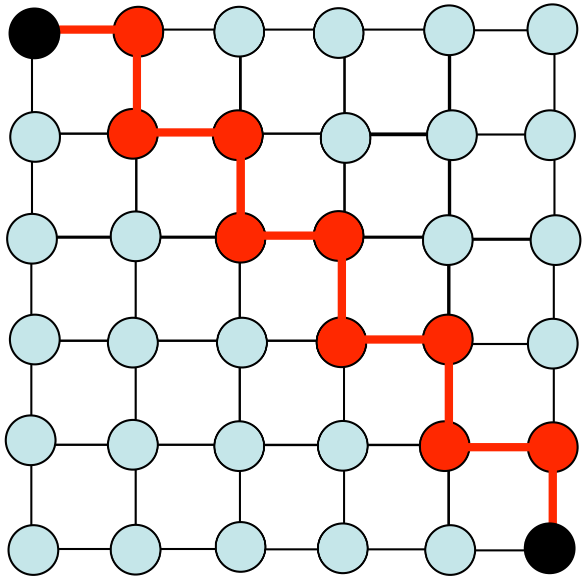




etc.

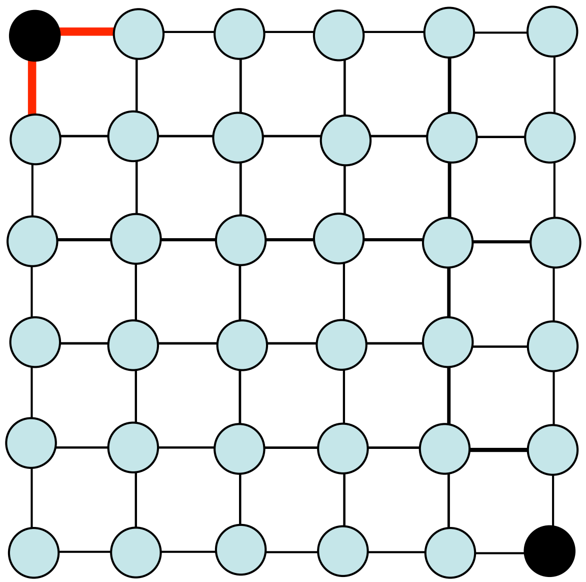
$r_2 r_3 + r_4 - r_2 r_3 r_4$

**Series-parallel reductions have polynomial cost.**

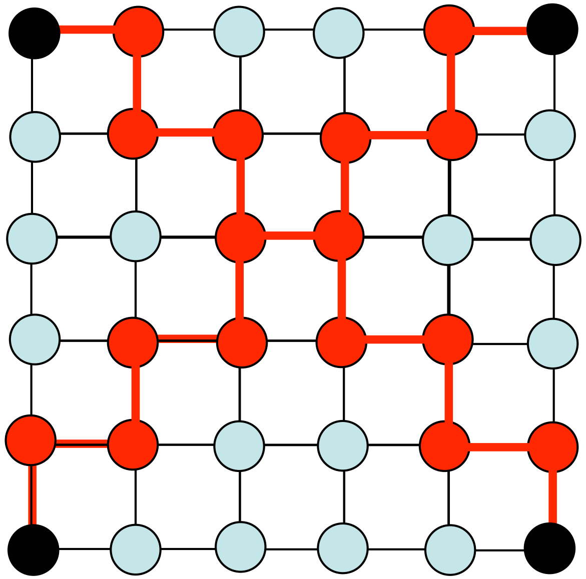
# 4 PATHS AND CUTS



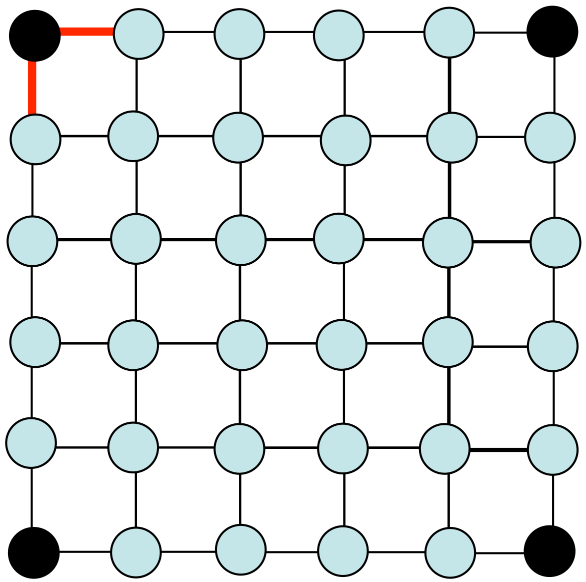
A PATH ($|K| = 2$)



A CUT ($|K| = 2$)



A PATH (|K| = 4)



A CUT ($|K| = 4$)

- Let $P$ be a path.
- Let $P$-up denote the event

  $P$-up = "all links in $P$ are up",

  $\Pr(P\text{-up}) = \prod_{\text{link } i \text{ is in } P} r_i$
- Since $P$-up $\Rightarrow$ system is up,

  $\Pr(P\text{-up}) \leq R$

- Let $C$ be a cut.
- Let $C$-down denote the event

  $C$-down = "all links in $C$ are down",

  $\Pr(C\text{-down}) = \prod_{\text{link } i \text{ is in } C} (1 - r_i)$
- Since $C$-down $\Rightarrow$ system is down,

  $\Pr(C\text{-down}) \leq Q = 1 - R$

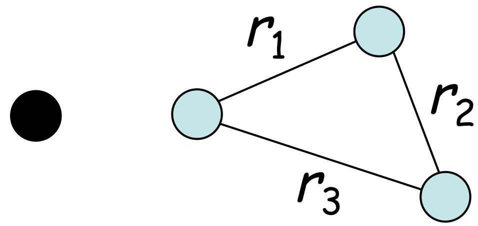
# 5 THE GENERAL IDEA

- Again, with $P$ be a path and $P$-up the event "all links in $P$ are up", we can write

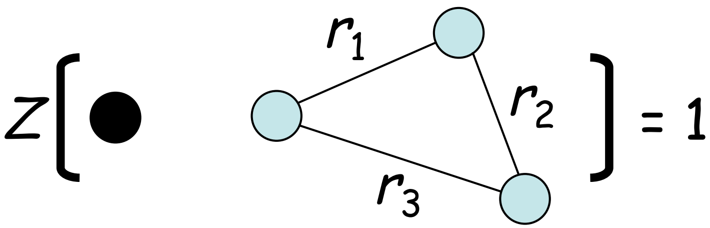
$$R = \Pr(P\text{-up}) + [1 - \Pr(P\text{-up})] \Pr(\text{sys. up} \mid \text{"at least one link in } P \text{ is down"})$$

- This suggest to sample on a conditional system, to estimate the last conditional probability (idea used in previous works by the authors).

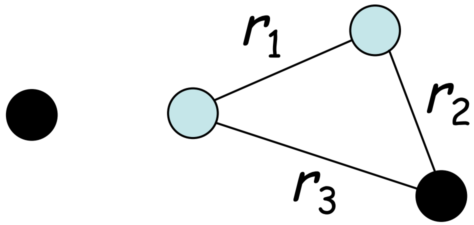
- Here, we follow another conditioning-based idea coming also from previous works (the RVR estimator).
- We define an estimator $Z$ associated with our MC method which is illustrated here through examples:
  - if the network is



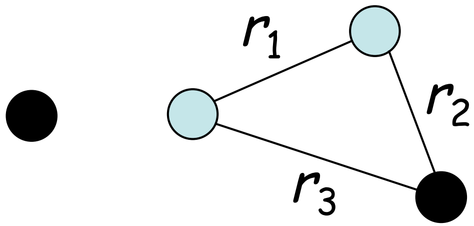

  - then



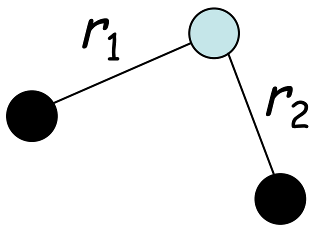
- if the network is



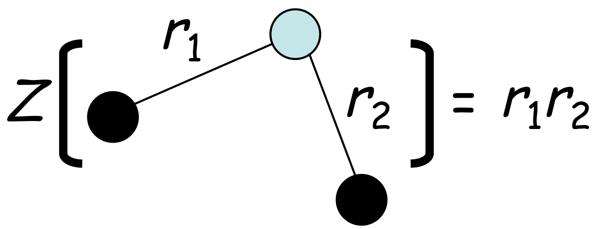

- then

$Z\Big[$  $\Big] = 0$

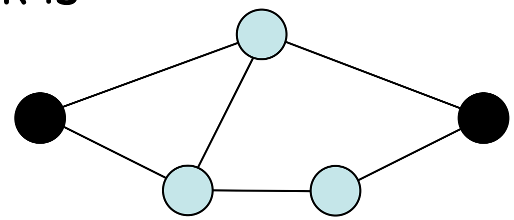
- if the network is



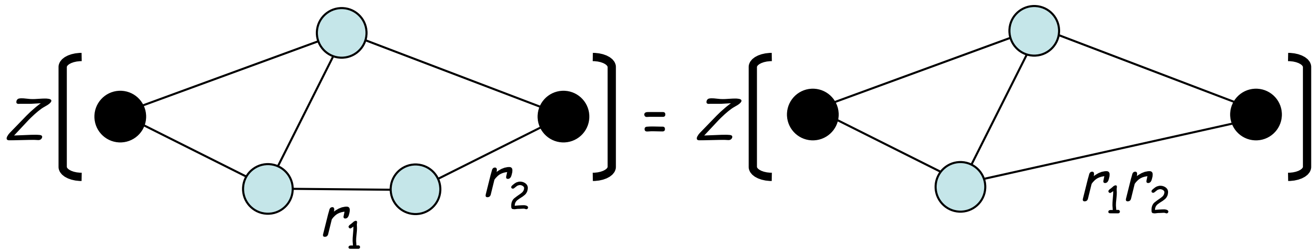

- then



- then if the network is

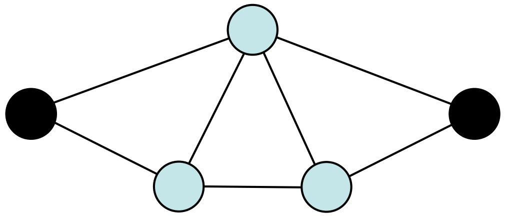


- then

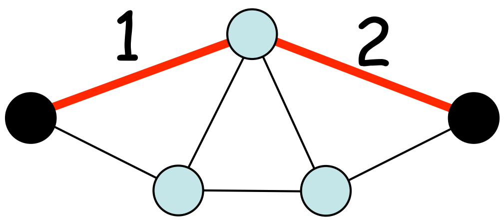



- The remaining case is a $K$-connected network where there is no possible series-parallel reduction and $|K| \geq 2$.

Let $N$ be



We first select a path*:




Let $L_i$ be the event "line $i$ is up",
and let $\underline{L}_i$ be the event "line $i$ is down", $i = 1{,}2$.

---

* The RVR method used a cut, but the idea is the same.

- We partition $\Omega$ in the following way:

$$\Omega = \{L_1L_2\ ,\ \underline{L_1}\ ,\ L_1\underline{L_2}\}$$

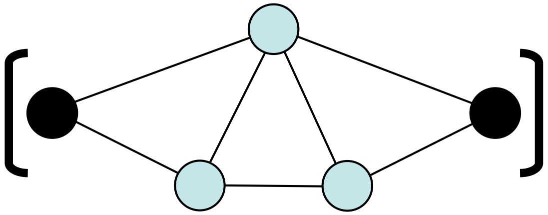
Prob. = $r_1r_2$

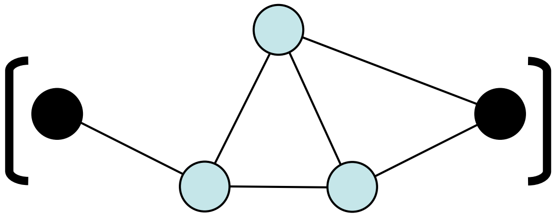
Prob. = $1 - r_1$

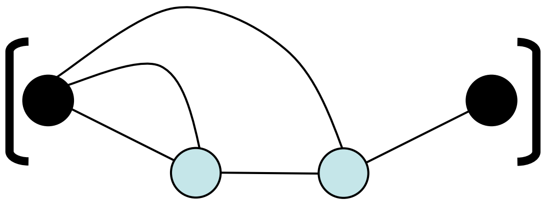
Prob. = $r_1(1 - r_2)$

- Let $X$ be the r.v. "first line down in the path", $i = 1,2$, with $X = 0$ if all lines in the path are up.
- Let $Y = X \mid X > 0$ ($Y$ lives in $\{1,2\}$)
- We have $\Pr(Y = 1) = (1 - r_1) / (1 - r_1r_2)$ and $\Pr(Y = 2) = r_1(1 - r_2)/(1 - r_1r_2)$.

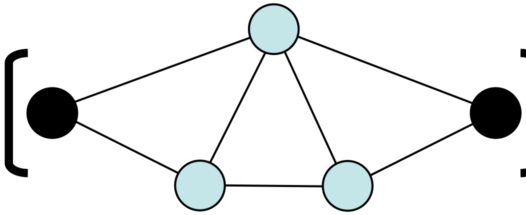
$\Omega = \{L_1L_2\ ,\ \underline{L_1}\ ,\ L_1\underline{L_2}\}$

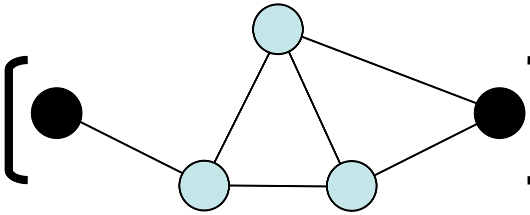
$Z$  $= \quad r_1r_2$

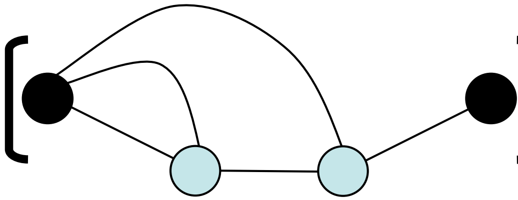
$+ (1 - r_1r_2)\ 1\{Y = 1\}$ $Z$ 

$+ (1 - r_1r_2)\ 1\{Y = 2\}$ $Z$ 

$\Omega = \{L_1L_2\ ,\ \underline{L_1}\ ,\ L_1\underline{L_2}\}$

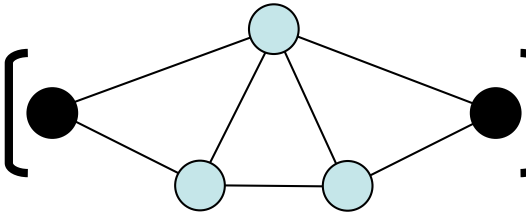
$Z\Big[$  $\Big] = r_1r_2$

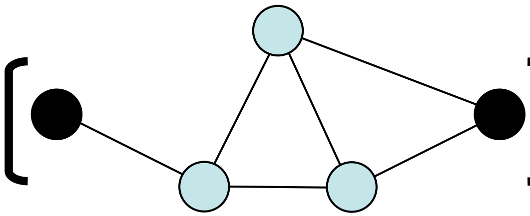
$+\ (1 - r_1r_2)\ 1\{Y = 1\}\ Z\Big[$  $\Big]$

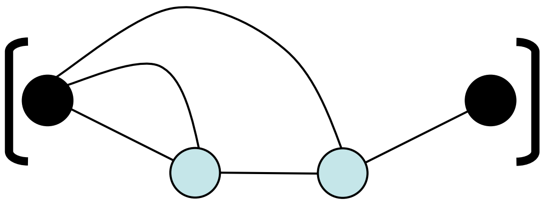
$+\ (1 - r_1r_2)\ 1\{Y = 2\}\ Z\Big[$  $\Big]$

to evaluate $Z$ here,

$\Omega = \{L_1L_2\,,\ \underline{L_1}\,,\ L_1\underline{L_2}\}$

$Z\Big[$  $\Big] = \quad r_1 r_2$

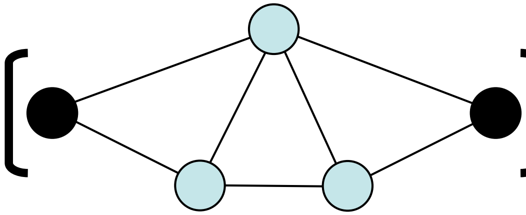
$+ \ (1 - r_1 r_2)\ 1\{Y = 1\}\ \ Z\Big[$  $\Big]$

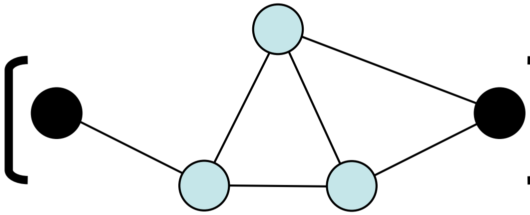
$+ \ (1 - r_1 r_2)\ 1\{Y = 2\}\ \ Z\Big[$  $\Big]$

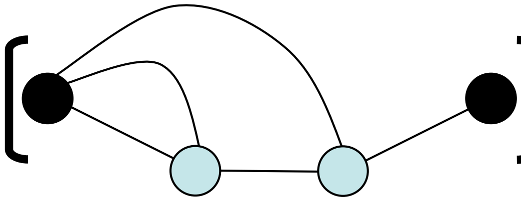
to evaluate $Z$ here,

we sample $Y$

$\Omega = \{L_1L_2\,,\ \underline{L_1}\,,\ L_1\underline{L_2}\}$

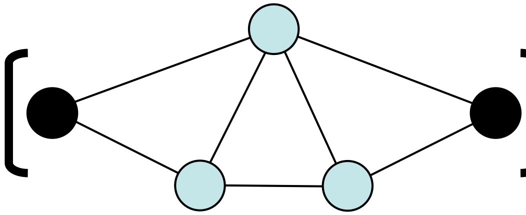
$Z\Big[$  $\Big] = \; r_1r_2$

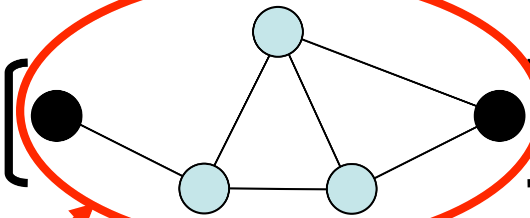
$+ \; (1 - r_1r_2)\; 1\{Y = 1\} \;\; Z\Big[$  $\Big]$

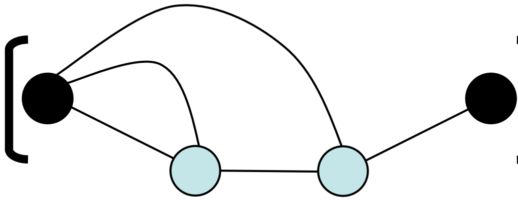
$+ \; (1 - r_1r_2)\; 1\{Y = 2\} \;\; Z\Big[$  $\Big]$

If we get for $Y$ value 2, we have a
series-parallel reducible network,
so, we do an exact and fast computation.

$\Omega = \{L_1L_2\,,\ \underline{L_1}\,,\ L_1\underline{L_2}\}$

$Z\Big[$  $\Big] = r_1 r_2$

$+\ (1 - r_1 r_2)\, 1\{Y = 1\}\ Z\Big[$  $\Big]$

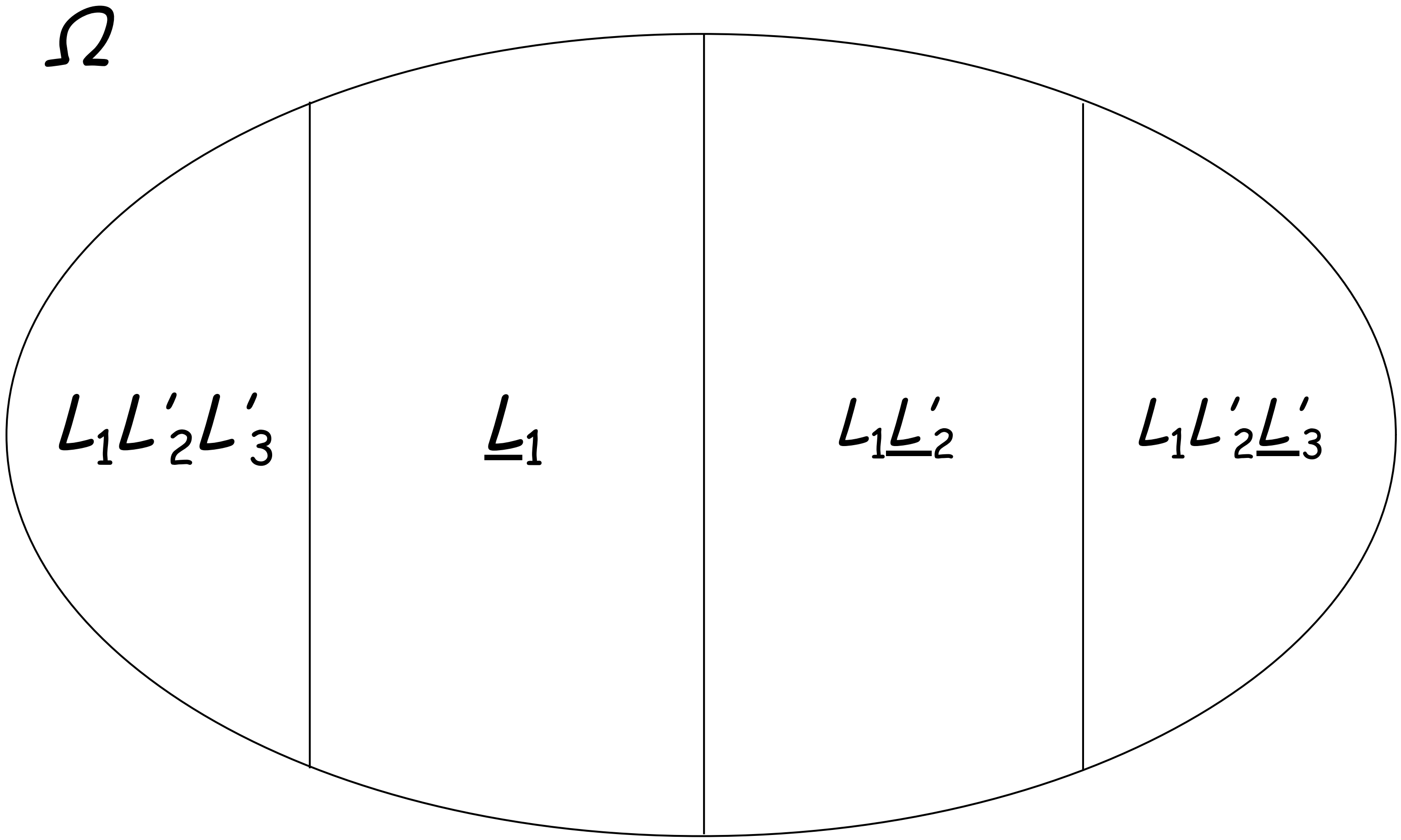
$+\ (1 - r_1 r_2)\, 1\{Y = 2\}\ Z\Big[$  $\Big]$

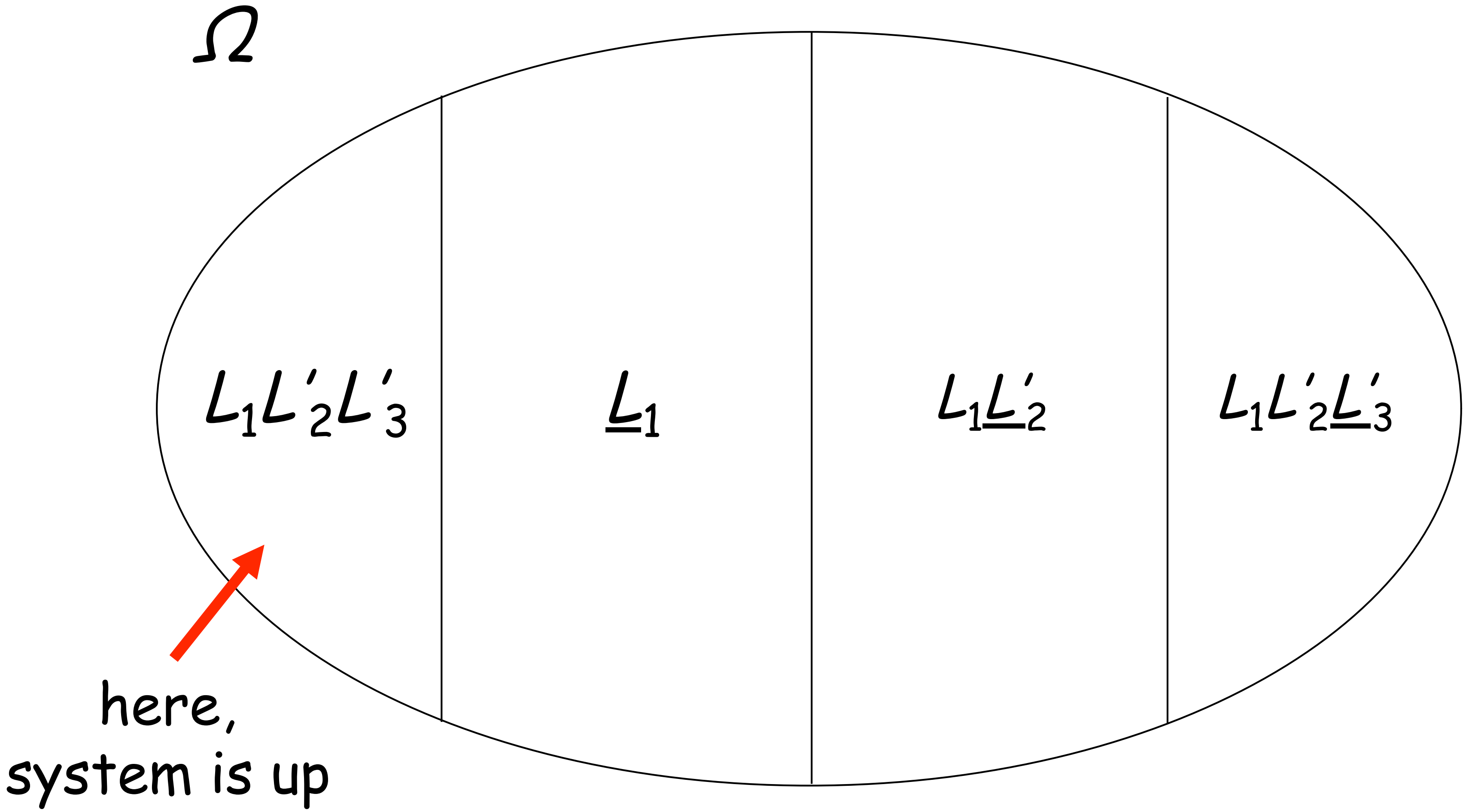
and this is series-parallel reducible

**If we get value 1, the procedure continues recursively from this network.**

# 6 THE METHOD (SKETCH)

- It consists of generalizing this idea to work with a path and a cut simultaneously.
- Assume we select a cut $C = (l_1, l_2, l_3)$ and a path $P = (l_1, l'_2, l'_3)$ (paper's notation here).
- Denote by $L_i$ (resp. by $L'_i$) the event "link $l_i$ is up" (resp. "link $l'_i$ is up"), and by $\underline{L}_i$ (resp. by $\underline{L}'_i$) the event "link $l_i$ is down" (resp. "link $l'_i$ is down").
- Consider we partition first $\Omega$ into the events $\{L_1L'_2L'_3,\ \underline{L}_1,\ L_1\underline{L}'_2,\ L_1L'_2\underline{L}'_3\}$, that is, the events
  - "all 3 links in $P$ work",
  - "link $l_1$ of $P$ is down"
  - "in $P$, link $l_1$ is up and link $l'_2$ is down"
  - "in $P$, links $l_1$ and $l'_2$ are up and link $l'_3$ is down"


$\Omega$
$L_1L'_2L'_3$
$\underline{L}_1$
$L_1\underline{L'}_2$
$L_1L'_2\underline{L'}_3$


$\Omega$
$L_1L'_2L'_3$
$\underline{L}_1$
$L_1\underline{L}'_2$
$L_1L'_2\underline{L}'_3$
here,
system is up

- Now, we refine the partition crossing the previous decomposition with the partition $\{\underline{L}_1\underline{L}_2\underline{L}_3,\ L_1,\ \underline{L}_1L_2,\ \underline{L}_1\underline{L}_2L_3\}$, using now cut $C$.

| $L_1L'_2L'_3$ | $\underline{L}_1$ | $L_1\underline{L}'_2$ | $L_1L'_2\underline{L}'_3$ |
|---|---|---|---|
| $\underline{L}_1\underline{L}_2\underline{L}_3$ | | | |
| $L_1$ | | | |
| $\underline{L}_1L_2$ | | | |
| $\underline{L}_1\underline{L}_2L_3$ | | | |

| | $L_1L'_2L'_3$ | $\underline{L}_1$ | $L_1\underline{L}'_2$ | $L_1L'_2\underline{L}'_3$ |
|---|---|---|---|---|
| $\underline{L}_1\underline{L}_2\underline{L}_3$ | $\varnothing$ | | | |
| $L_1$ | | $\varnothing$ | | |
| $\underline{L}_1L_2$ | $\varnothing$ | | $\varnothing$ | $\varnothing$ |
| $\underline{L}_1\underline{L}_2L_3$ | $\varnothing$ | | $\varnothing$ | $\varnothing$ |

| | $L_1L'_2L'_3$ | $\underline{L}_1$ | $L_1\underline{L}'_2$ | $L_1L'_2\underline{L}'_3$ |
|---|---|---|---|---|
| $\underline{L}_1\underline{L}_2\underline{L}_3$ | ∅ | – | – | – |
| $L_1$ | + | ∅ | | |
| $\underline{L}_1L_2$ | ∅ | | ∅ | ∅ |
| $\underline{L}_1\underline{L}_2L_3$ | ∅ | | ∅ | ∅ |

| | $L_1L'_2L'_3$ | $\underline{L}_1$ | $L_1\underline{L}'_2$ | $L_1L'_2\underline{L}'_3$ |
|---|---|---|---|---|
| $\underline{L}_1\underline{L}_2\underline{L}_3$ | $\varnothing$ | – | – | – |
| $L_1$ | + | $\varnothing$ | $L_1\underline{L}'_2$ | $L_1L'_2\underline{L}'_3$ |
| $\underline{L}_1L_2$ | $\varnothing$ | $\underline{L}_1L_2$ | $\varnothing$ | $\varnothing$ |
| $\underline{L}_1\underline{L}_2L_3$ | $\varnothing$ | $\underline{L}_1\underline{L}_2L_3$ | $\varnothing$ | $\varnothing$ |

| | $L_1L'_2L'_3$ | $\underline{L}_1$ | $L_1\underline{L}'_2$ | $L_1L'_2\underline{L}'_3$ |
|---|---|---|---|---|
| $\underline{L}_1\underline{L}_2\underline{L}_3$ | $\varnothing$ | $-$ | $-$ | $-$ |
| $L_1$ | $+$ | $\varnothing$ | $L_1\underline{L}'_2$ | $L_1L'_2\underline{L}'_3$ |
| $\underline{L}_1\underline{L}_2$ | $\varnothing$ | $\underline{L}_1\underline{L}_2$ | $\varnothing$ | $\varnothing$ |
| $\underline{L}_1\underline{L}_2L_3$ | $\varnothing$ | $\underline{L}_1\underline{L}_2L_3$ | $\varnothing$ | $\varnothing$ |

$B_2$ $B_3$ $B'_2$ $B'_3$

| | $L_1L'_2L'_3$ | $\underline{L}_1$ | $L_1\underline{L}'_2$ | $L_1L'_2\underline{L}'_3$ |
|---|---|---|---|---|
| $\underline{L}_1\underline{L}_2\underline{L}_3$ | $\varnothing$ | − | − | − |
| $L_1$ | + | $\varnothing$ | $L_1\underline{L}'_2$ | $L_1L'_2\underline{L}'_3$ |
| $\underline{L}_1L_2$ | $\varnothing$ | $\underline{L}_1L_2$ | $\varnothing$ | $\varnothing$ |
| $\underline{L}_1\underline{L}_2L_3$ | $\varnothing$ | $\underline{L}_1\underline{L}_2L_3$ | $\varnothing$ | $\varnothing$ |

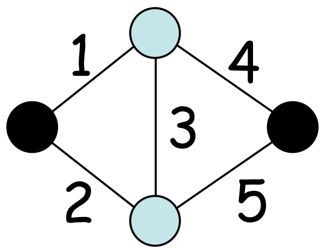
$q_2$ $q_3$ $p_2$ $p_3$

# ALGORITHM

- sum = 0.0
- **for** m = 1, 2, ..., M
  - z = sample(Z)
  - sum += z
- $Q^{re}$ = sum/M

- Remark: the only sampled variables are the $Y_k$ (the auxiliary variable at the $k$th irreducible and not trivial network found in the recursive process)
- THEOREM
  - $E(Z) = Q$ (so, unbiaised estimator)
  - $Var(Z) \leq [R - Pr(P\text{-up})][Q - Pr(C\text{-down})] \leq RQ$ (so, variance reduction)

# 7 EXAMPLE: the bridge




$M = 10^6$ samples

- $r_1 = 1 - \exp(-1/0.3) \sim 0.964326$
- $r_2 = 1 - \exp(-1/0.1) \sim 0.999955$
- $r_3 = 1 - \exp(-1/0.8) \sim 0.713495$
- $r_4 = 1 - \exp(-1/0.1) \sim 0.999955$
- $r_5 = 1 - \exp(-1/0.2) \sim 0.993262$

($R \sim 0.999929$)

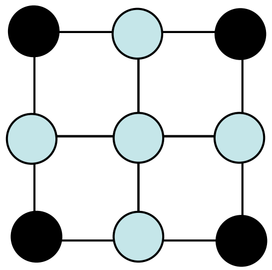
**$Var(Q^{std})/Var'(Q^{re}) \sim 1.95 \times 10^6$**

**$Var'(Q^{ce})/Var'(Q^{re}) \sim 187$**

($Var'(Q^x)$ is the variance of the considered estimator $x$ of $Q$)

a recent cross-entropy-based estimator

# 7 EXAMPLE: the 3-grid

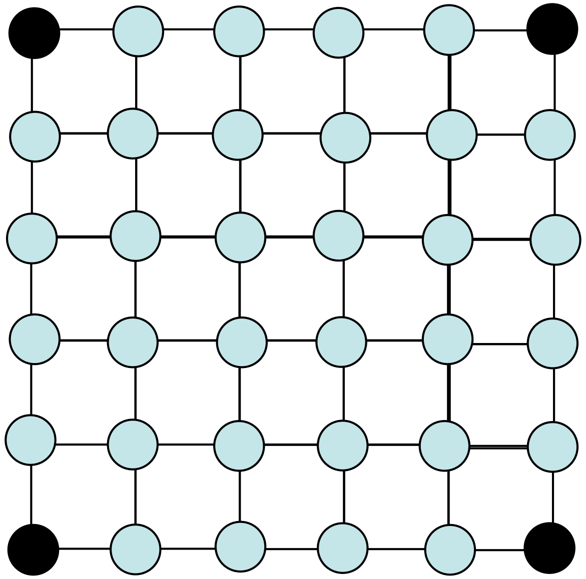


$|V| = 9$,
$|K| = 4$,
$|E| = 12$

$M = 10^6$ samples

| $q_i = 1 - r_i$ | $Q$ | $Var(Q^{std})/Var'(Q^{re})$ | $Var'(Q^{ce})/Var'(Q^{re})$ |
|---|---|---|---|
| $10^{-3}$ | $\sim 4.00\times10^{-12}$ | $\sim 4.49\times10^{5}$ | $\sim 2.07$ |
| $10^{-6}$ | $\sim 4.00\times10^{-18}$ | $\sim 4.44\times10^{11}$ | $\sim 2.06$ |

# 7 EXAMPLE: the 6-grid



$|V| = 36$,
$|K| = 4$,
$|E| = 60$

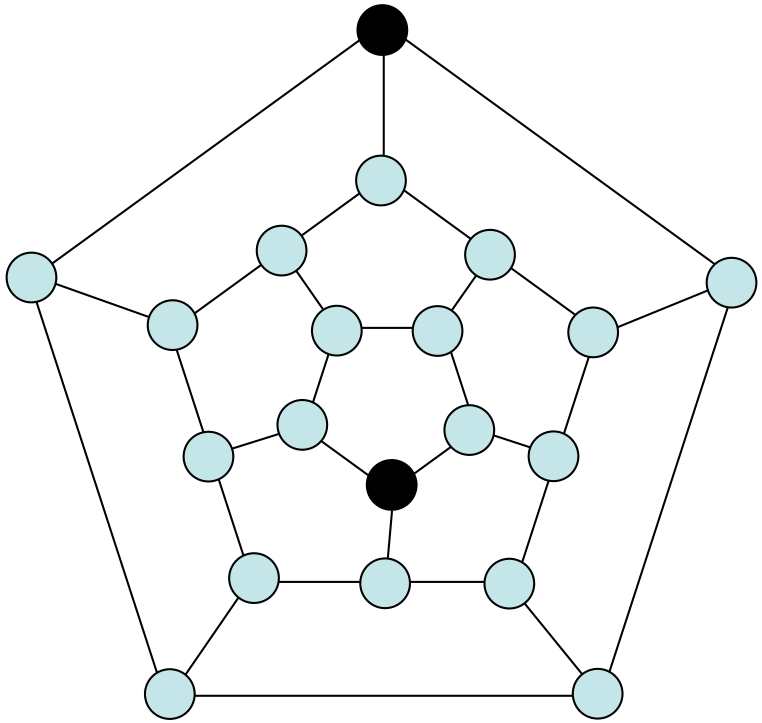
$M = 10^6$ samples

| $q_i$ | $Q$ | $Var(Q^{std})/Var'(Q^{re})$ | $Var'(Q^{ce})/Var'(Q^{re})$ |
|---|---|---|---|
| $10^{-3}$ | $\sim 4.01 \times 10^{-6}$ | $\sim 1.13 \times 10^5$ | $\sim 1.06$ |
| $10^{-6}$ | $\sim 4.00 \times 10^{-12}$ | $\sim 1.11 \times 10^{11}$ | $\sim 1.04$ |

# 7 EXAMPLE: the 10-complete graph

- We varied $K$ ($|K| = 2$, $|K| = 5$, $|K| = 10$) and we considered always the case $r_i = r$ varying $r$ from 0.99 to 0.1.
- Main goal of the experiments: evaluate the improvement of working with both a cut and a path, compared to the original technique that used only a cut.
- For instance, the ratio between the variance of the estimator using only a cut and the variance of the new one goes from ~ 2 when $|K| = 10$ and $r = 0.1$ to ~ $10^{27}$ when $|K| = 2$ and $r = 0.99$.

# 7 EXAMPLE: the dodecahedron



Here, we took always the case of $|K| = 2$ but varied the distance between the two terminals (in the picture, that distance is 5).

- We also varied $r_i = r$ from 0.99 to 0.1.
- When the distance is 1, the ratio between the variance of the estimator using only a cut and the variance of the new one goes from ~ $3.8 \times 10^3$ if $r = 0.99$ to ~ $9.2 \times 10^7$ if $r = 0.1$.
- At the other extreme, when the distance was 5, the method using only a cut was slightly better in the case of $r = 0.99$; for the remaining values of $r$ the ratio ranged in the interval ~ (2.4 , 4.6).

# 8 CONCLUSIONS

- The method is simple and efficient (so far it compares well with previous proposals, while this must be explored more in deep).
- It can be extended to more powerful decompositions.
- More importantly, the choice of the cut and the path is not evident (because of the transformations made to the graphs when simplifying them inside the recursive procedure).

# 2b/
# IMPROVING EFFICIENCY BY TIME REDUCTION (INSTEAD OF VARIANCE REDUCTION)

# ABSTRACT

- A time-reduction procedure for network reliability estimation
- Main idea:
  - again, exploit the specificities of this static problem
  - a version of conditional Monte Carlo
- For simplicity, and because we are probably close to out of time at this point, we consider only the source-to-terminal case

# 2 STANDARD MC

- #failed = 0
- **for** m = 1, 2, …, M
  - g = sample(G)
  - **if** $g \notin U$ **then** #failed += 1
- $Q^{std}$ = #failed/M
- $V^{std} = Q^{std}(1 - Q^{std})/(M - 1)$

→ The rare event problem when $Q \ll 1$

# COMPUTATIONAL COMPLEXITY

- Internal loop: sampling a graph state (state of each edge), and verifying if it belongs or not to set $U$ (DFS search); total complexity is $O(|E|)$.
- $M$ iterations; initialization time and final computations in $O(1)$.
- Total computation time in $O(M|E|)$, linear in # of edges and # of replications.

# MAIN IDEA

- Imagine we implement the crude Monte Carlo procedure in the following way:
  - first, we build a (huge) table with $M$ rows (e.g. $M = 10^9$) and $|E| + 1$ columns
  - each row corresponds to a replication
  - row $m$:
    - column $i$: 1 if edge $i$ works at repl. $m$, 0 otherwise
    - last column ($|E| + 1$): 1 if sys. ok, 0 otherwise
  - then, we count the # of 0 in last column, we divide by $M$ and we have our estimator $Q^{std}$

- We consider the case of case $r_i \approx 1$
- Look at column $i$ now: full of '1', from time to time a '0'
- Call $F_i$ the r.v. "first row in the table with a '0' in column $i$"
- Assume the table is infinite: in that case, $F_i$ is geometric on {1, 2, …} with

$$\Pr(F_i = f) = r_i^{f-1}(1 - r_i)$$

and

$$E(F_i) = r_i/(1 - r_i)$$

- This suggests the following procedure:
  - sample the geometric r.v. $F_1$, $F_2$, … and compute $W = \min\{F_1, F_2, \ldots\}$
  - consider that in replications 1, 2, …, $W$-1, the system was operational
  - for replication $W$, perform the DFS test
  - then, start again for each column (edge), sampling the next ‘0’ value, then looking for the next row with at least one ‘0’…
- Formalizing this procedure we can prove that it can be seen as an implementation of the standard estimator

# COMPLEXITY ISSUE

- For a table with $M$ rows,
  - each $F_i$ is sampled, on the average, $M/E(F_i)$ times, thus, the total cost is

$$M\sum_i \frac{1-r_i}{r_i}$$

  - in the rare event case, the usual situation is $(1 - r_i)/r_i \ll 1$ and even

$$\sum_i \frac{1-r_i}{r_i} \ll 1$$

- # of calls to the DFS procedure?
  - on the average, $M/E(W)$ times
- Since $W$ is also geometric with parameter $r = r_1 r_2 \ldots r_{|E|}$, the mean cost of the method is

$$M\left(\sum_i \frac{1-r_i}{r_i} + |E|\frac{1-r}{r}\right)$$

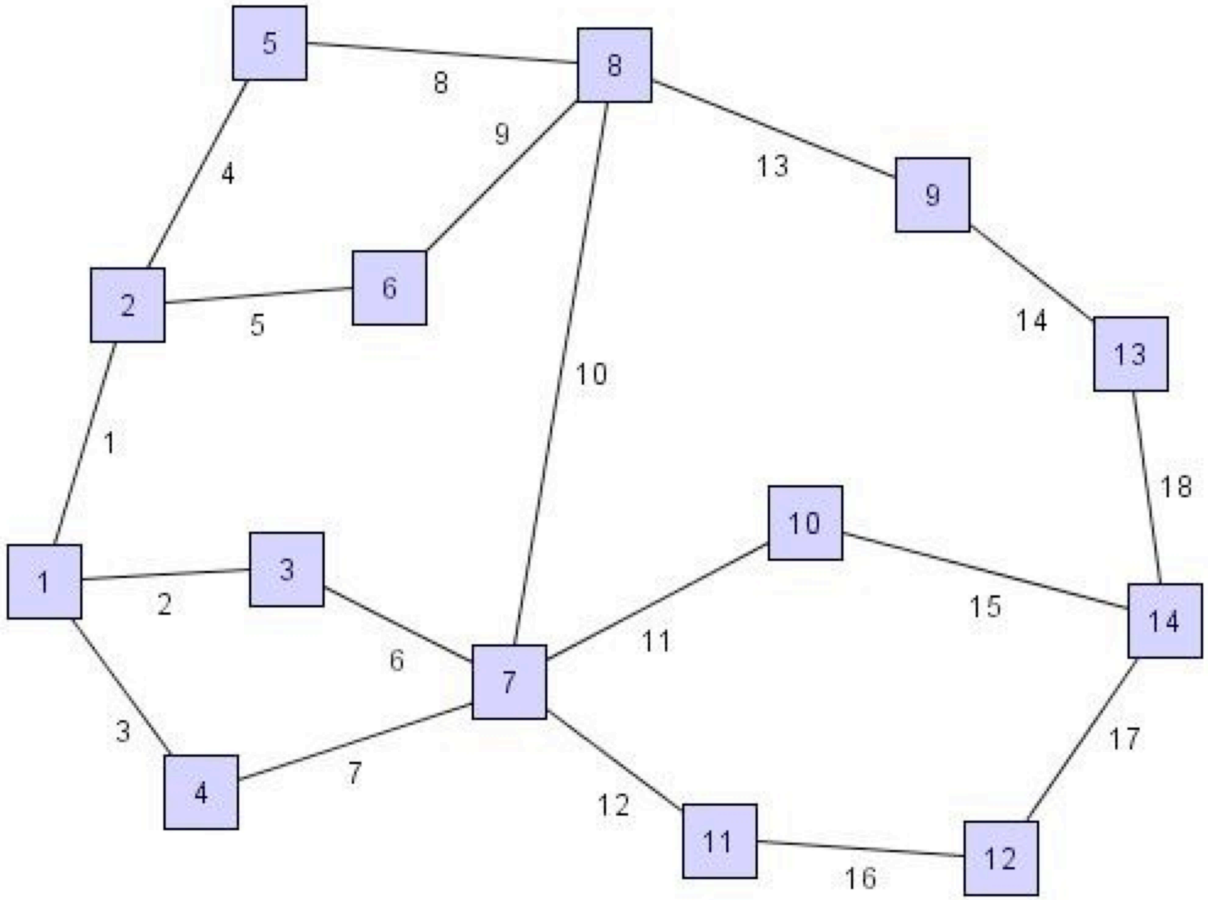
- For instance, assume $r_i = 1 - \varepsilon$
- The total mean cost is $\sim \varepsilon M |E|^2$

- Dividing the mean cost of the standard approach $M\,|E|$ by the mean cost of this more efficient "implementation", we get

$$1/(\varepsilon|E|)$$

# ILLUSTRATIONS

- Consider the following example:




source = 1
terminal = 14
$r_i = 0.9999$

the new method runs ~ 500 times faster than crude (for the same accuracy)

# FINAL COMMENTS

- The procedure can be improved further
- For instance, the DFS can be called only if the number of '0' in the row is at least equal to the breadth of the graph
- In the previous example, this leads to an improvement factor of ~ 600

# TUTORIAL'S CONCLUSIONS

- Many other techniques, applications and problems, not even mentioned here.
- Some randomly chosen examples:
  - use of Quasi-Monte Carlo techniques
  - using other types of information
  - splitting with static models
  - applications in physics (where "everything was invented…")
  - …
- Slides will be on-line with some added bibliography

- Reference:
  G. Rubino, B. Tuffin (Editors),
  "Rare Event Simulation
  Using Monte Carlo Methods"
  John Wiley & Sons,
  278 pages, March 2009.

http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0470772697.html