Aug 25 2010
ERC Workshop

Random Sampling Algorithms with Applications

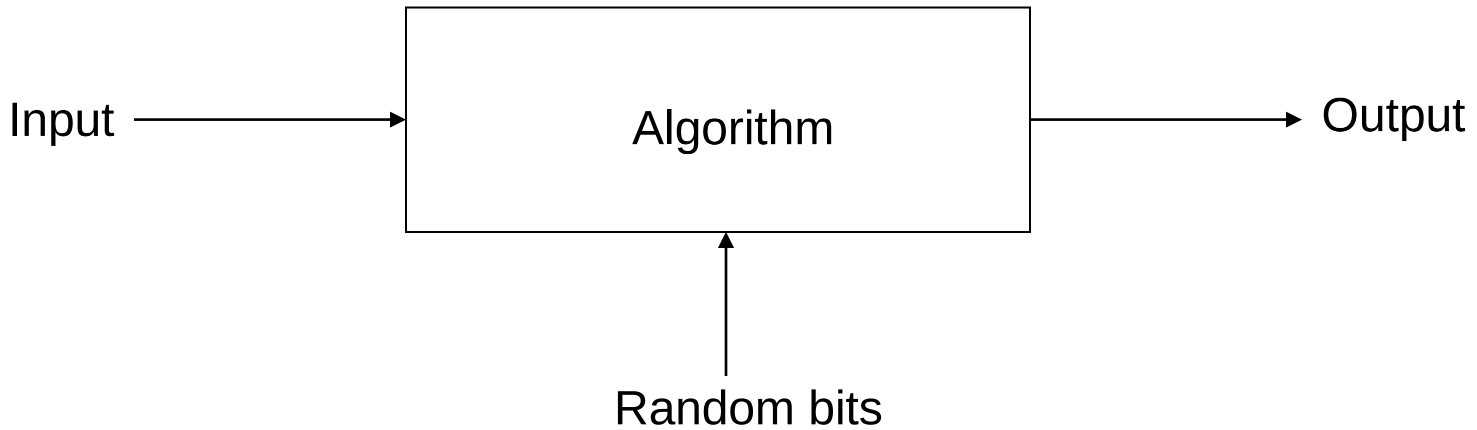
Kyomin Jung
KAIST

Contents

Randomized Algorithm

A randomized algorithm is defined as an algorithm that is allowed to access a source of independent, unbiased random bits, and it is then allowed to use these random bits to influence its computation.

Ex) Computer games, randomized quick sort…



Why randomness can be helpful?

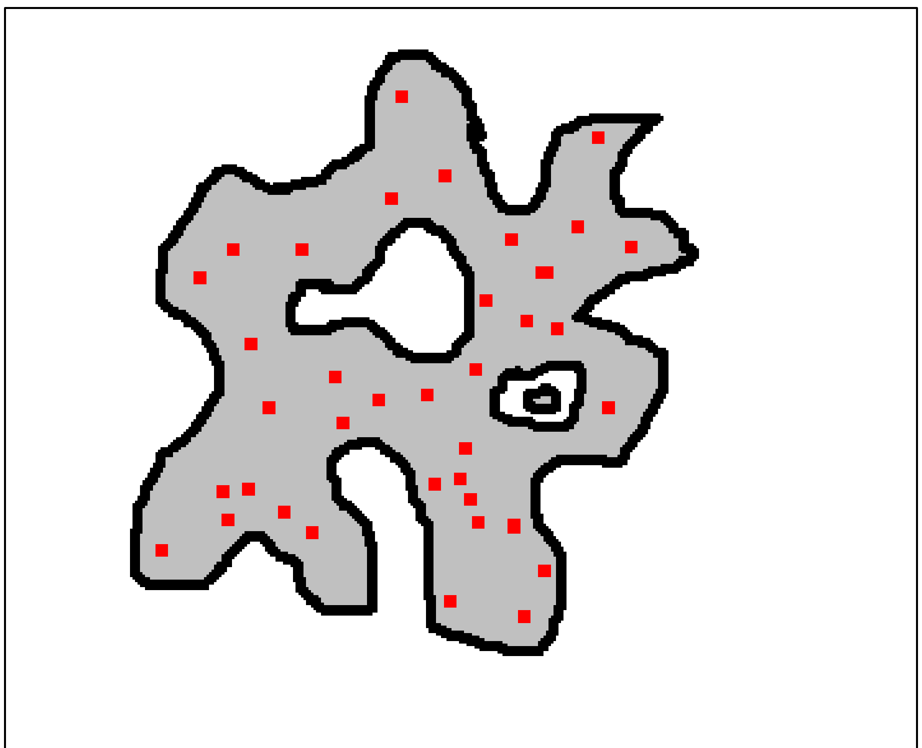
- A Simple example
  - Suppose we want to check whether an integer set $A = \{a_1, a_2, a_3 ..., a_n\}$ has an even number or not.
  - Even when A has n/2 many even numbers, if we run a Deterministic Algorithm, it may check n/2 +1 many elements in the worst case.
  - A Randomized Algorithm: At each time, choose an elements (to check) at random.
    - Smooths the “worst case input distribution” into “randomness of the algorithm”

Random Sampling

- What is a random sampling?
  - Given a probability distribution π, pick a point according to π.
  - e.g. Monte Carlo method for integration
    - Choose numbers uniformly at random from the integration domain, and compute the average value of f at those points

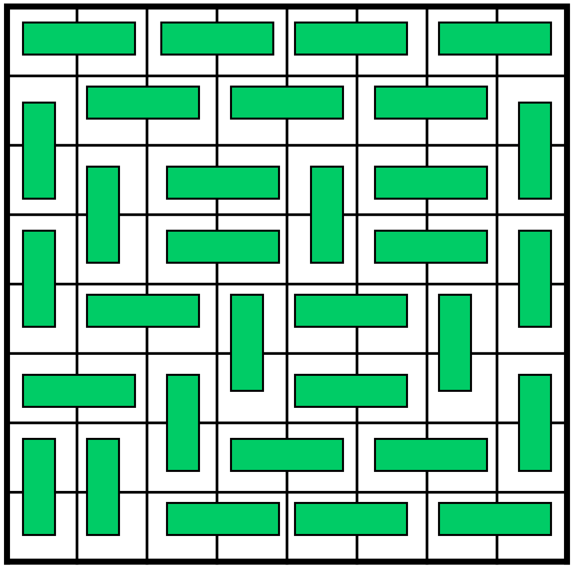
How to use Random Sampling?

- Volume computation in Euclidean space.
- Can be used to approximately count discrete objects. Ex) # of matchings in a graph



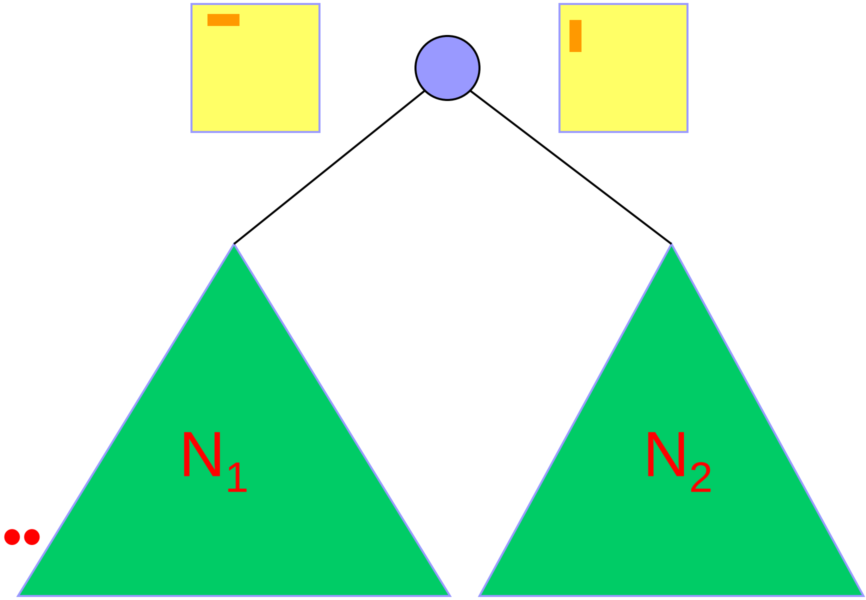
Application : Counting

- *How many ways can we tile with dominos?*



Application : Counting

- Sample tilings uniformly at random.
- Let *P_1 = proportion of sample of type 1*.
- N^* : estimation of N.
- $N^* = N_1^* / P_1 = N_{11}^* / (P_1 P_{11})$...

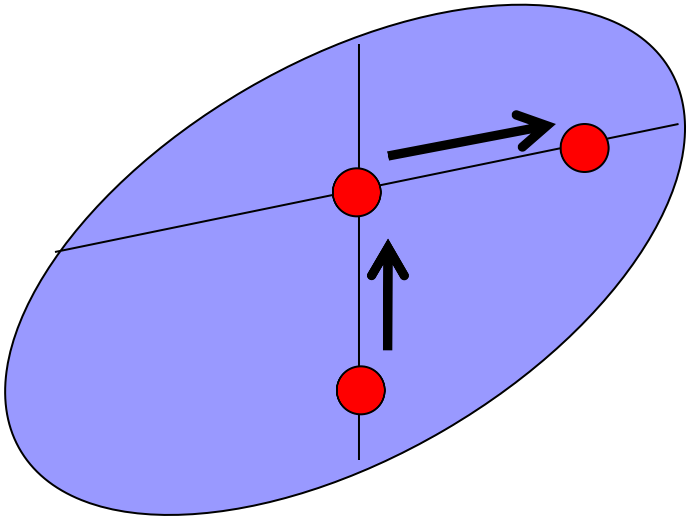


$N = N_1 + N_2$

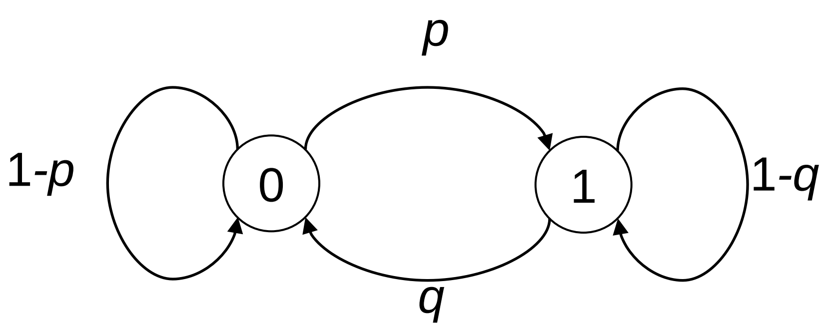
$N_1 = N_{11} + N_{12}$

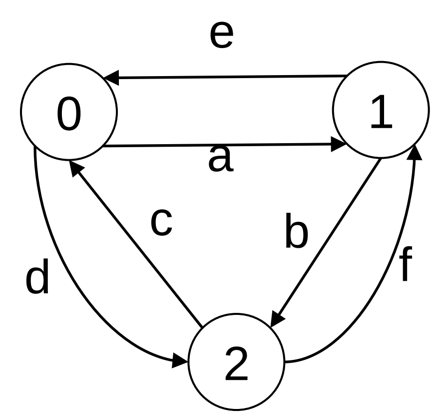
How to Sample? Ex: Hit and Run

- Hit and Run algorithm is used to sample from a convex set in an n-dimensional Euclidean space.
- It converges in $O(n^3)$ time. (n: dimension)



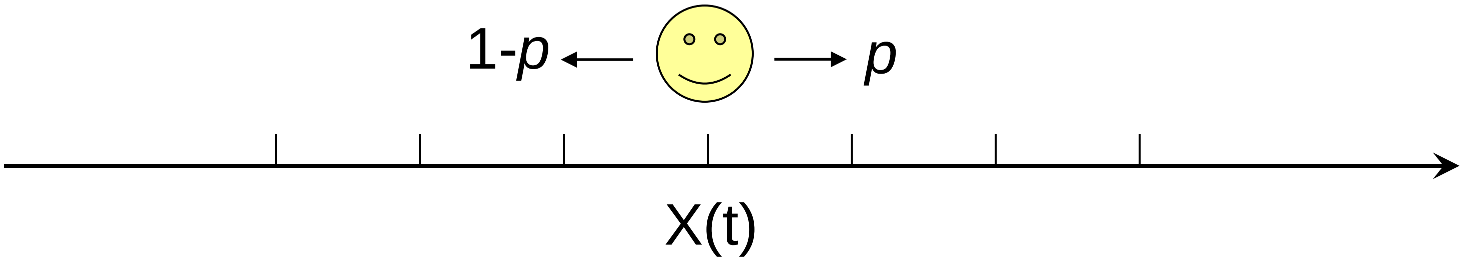
How to Sample? : Markov Chain (MC)





- "States" can be labeled 0,1,2,3,…
- At every time slot a "jump" decision is made randomly based on **current state**
- $\sum_j p_{ij} = 1$

Ex of MC: 1-D Random Walk



- Time is slotted
- The walker flips a coin every time slot to decide which way to go
- $$X(t+1) = \begin{cases} X(t)+1 & \text{w.p. } p \\ X(t)-1 & \text{w.p. } 1-p \end{cases}$$

Markov Property

- "Future" is independent of "Past" and depend only on "Present"
- In other words: Memoryless
- Useful for modeling and analyzing real systems

Stationary Distribution

Define $\pi_k(i) = \Pr\{X_k = i\}$

Then $\pi_{k+1} = \pi_k P$ (π_k is a row vector)

Stationary Distribution: $\pi = \lim_{k\to\infty} \pi_k$

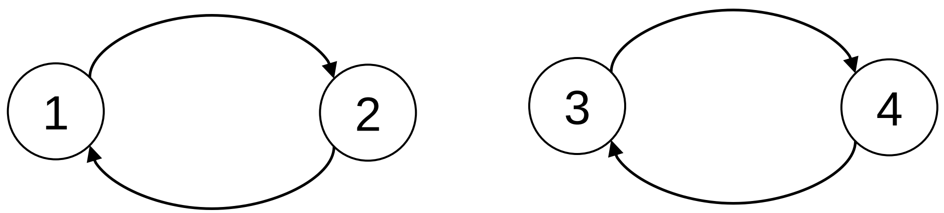
if the limit exists.

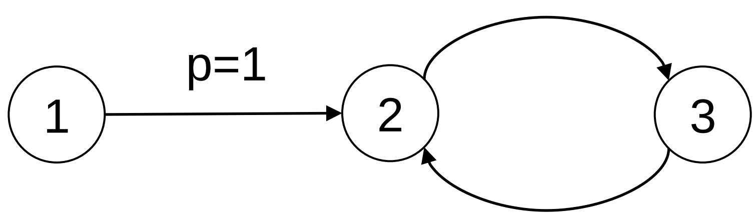
If π exists, it satisfies that

$$\sum_i \pi_i \mathrm{P}_{ij} = \pi_j \text{ for all } j, \quad \sum_i \pi(i) = 1$$

Conditions for π to Exist (I)

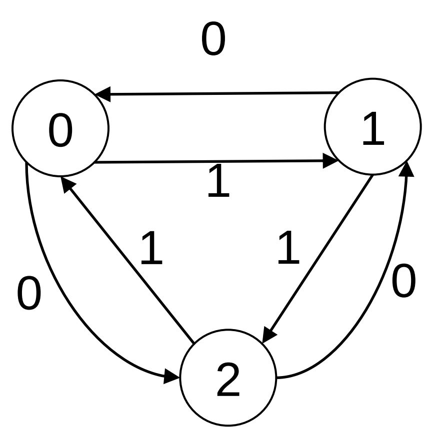
- The Markov chain is **irreducible.**
- Counter-examples:





Conditions for π to Exist (II)

- The Markov chain is **aperiodic**.
  - A MC is aperiodic if all the states are aperiodic.
- Counter-example:



Special case

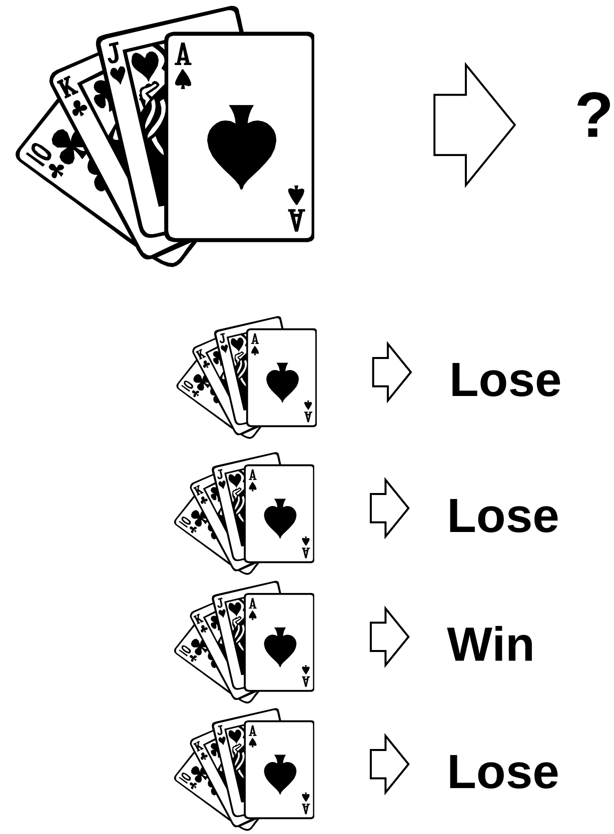
- It is known that a Markov Chain has stationary distribution π if the detailed balance condition holds:

$$\pi_{\mathrm{i}} \mathrm{P}_{\mathrm{ij}} = \pi_{j} \mathrm{P}_{\mathrm{ji}}$$

Monte Carlo principle

- Consider a card game: what’s the chance of winning with a properly shuffled deck?
- Hard to compute analytically

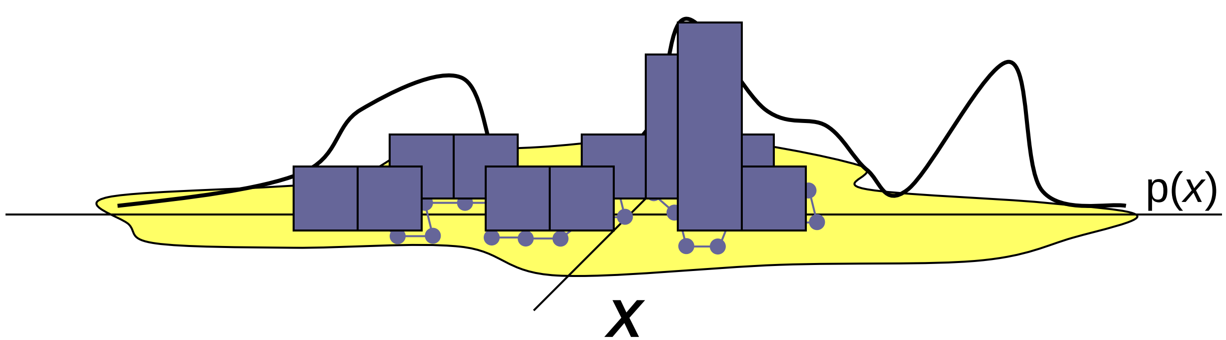
- Insight: why not just *play a few games*, and see empirically how many times win?
- More generally, can approximate a probability density function using samples from that density?



Chance of winning is 1 in 4!

Markov chain Monte Carlo (MCMC)

- Recall again the set *X* and the distribution p(*x*) we wish to sample from
- Suppose that it is hard to sample p(*x*) but that it is possible to "walk around" in *X* using only local state transitions
- Insight: we can use a "random walk" to help us draw random samples from p(*x*)



Markov chain Monte Carlo (MCMC)

- In order for a Markov chain to useful for sampling p(x), we require that for any starting state $x^{(1)}$

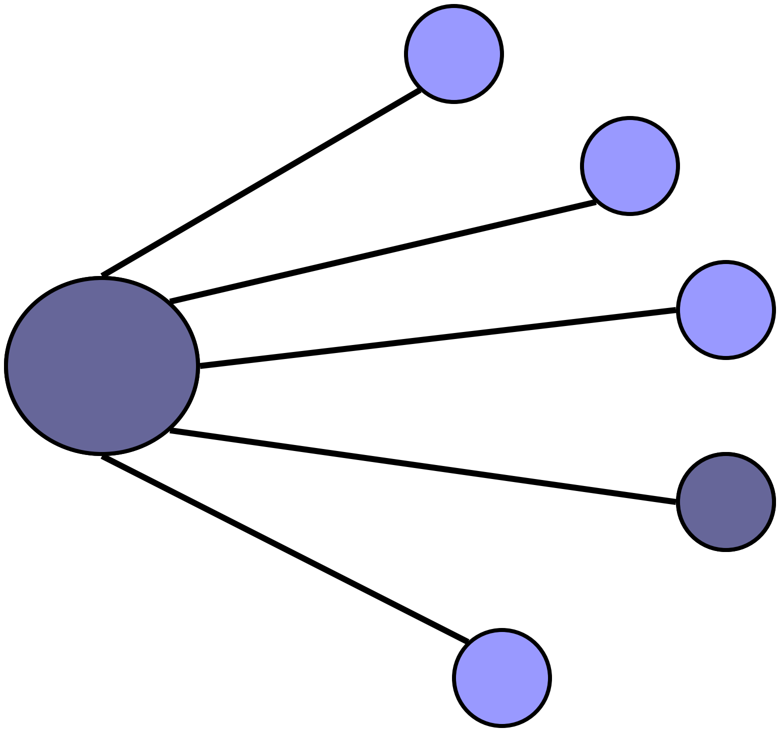
$$\mathrm{p}^{(t)}_{x^{(1)}}(x) \underset{t\to\infty}{\longrightarrow} \mathrm{p}(x)$$

- Equivalently, the stationary distribution of the Markov chain must be p(x).

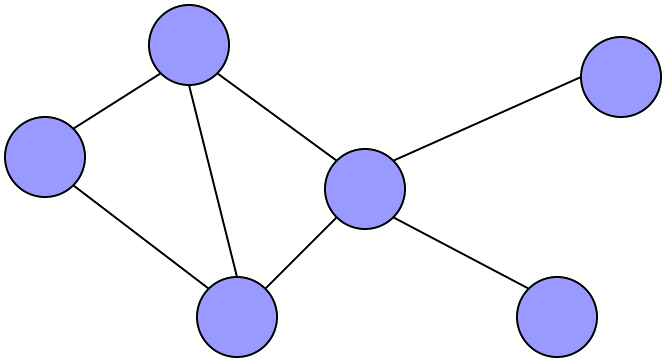
- Then we can start in an arbitrary state, use the Markov chain to do a random walk for a while, and stop and output the current state $x^{(t)}$.
- The resulting state will be sampled from p(x)!

Random Walk on Undirected Graphs

At each node, choose a neighbor u.a.r and jump to it



Random Walk on Undirected Graph G=(V,E)



Ω=V

$$P(x,y)=\begin{cases} 1/d(x) & (x,y)\in E \\ 0 & otherwise \end{cases}$$

- Irreducible $\Leftrightarrow$ G is connected
- Aperiodic $\Leftrightarrow$ G is not bipartite

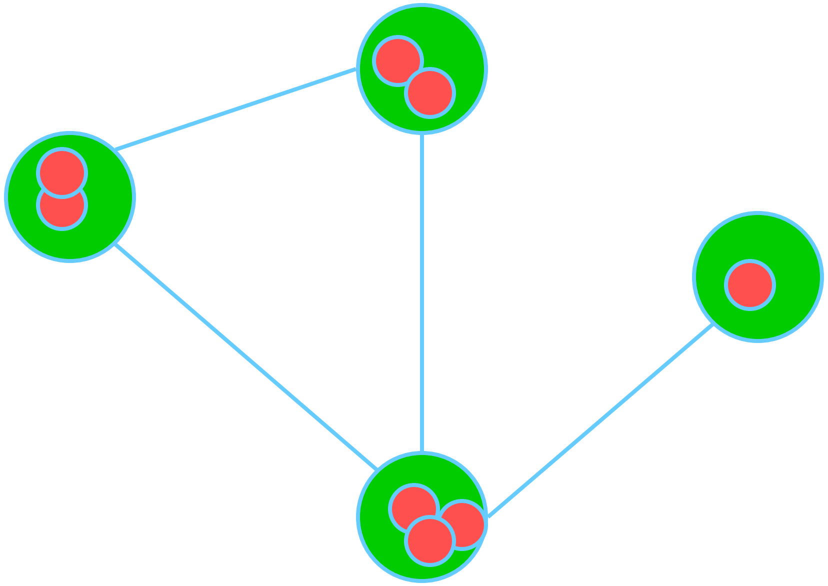
The Stationary Distribution

Claim: If G is connected and not bipartite, then the probability distribution induced by the random walk on it converges to

$$\pi(x)=d(x)/\Sigma_x d(x).$$

$\Sigma_x d(x)=2|E|$

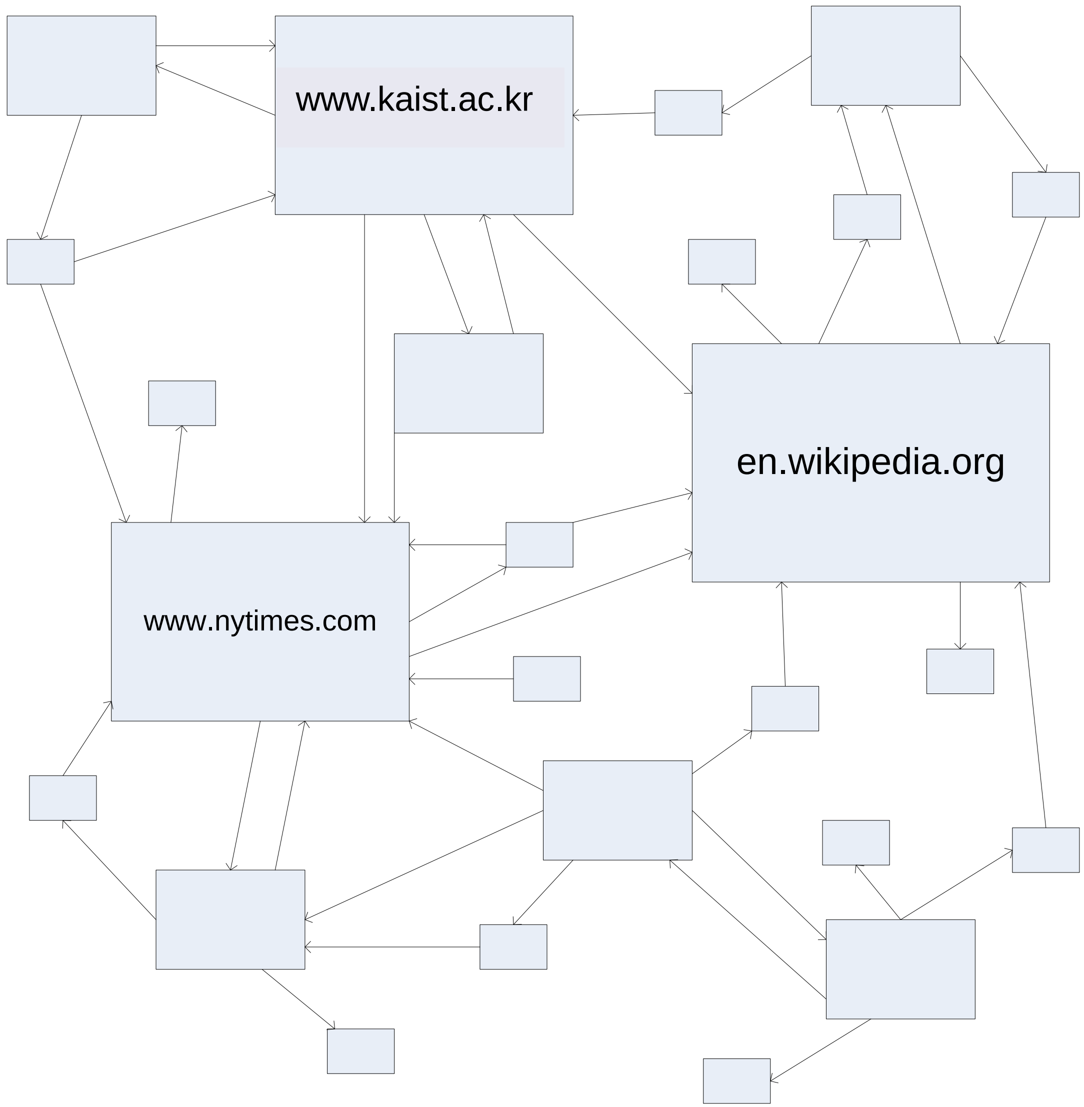
Proof: detailed balance condition holds.



PageRank: Random Walk Over The Web

- If a user starts at a random web page and surfs by clicking links and randomly entering new URLs, what is the probability that s/he will arrive at a given page?
- The PageRank of a page captures this notion
  - More "popular" or "worthwhile" pages get a higher rank
  - This gives a rule for random walk on The Web graph (a directed graph).

PageRank: Example

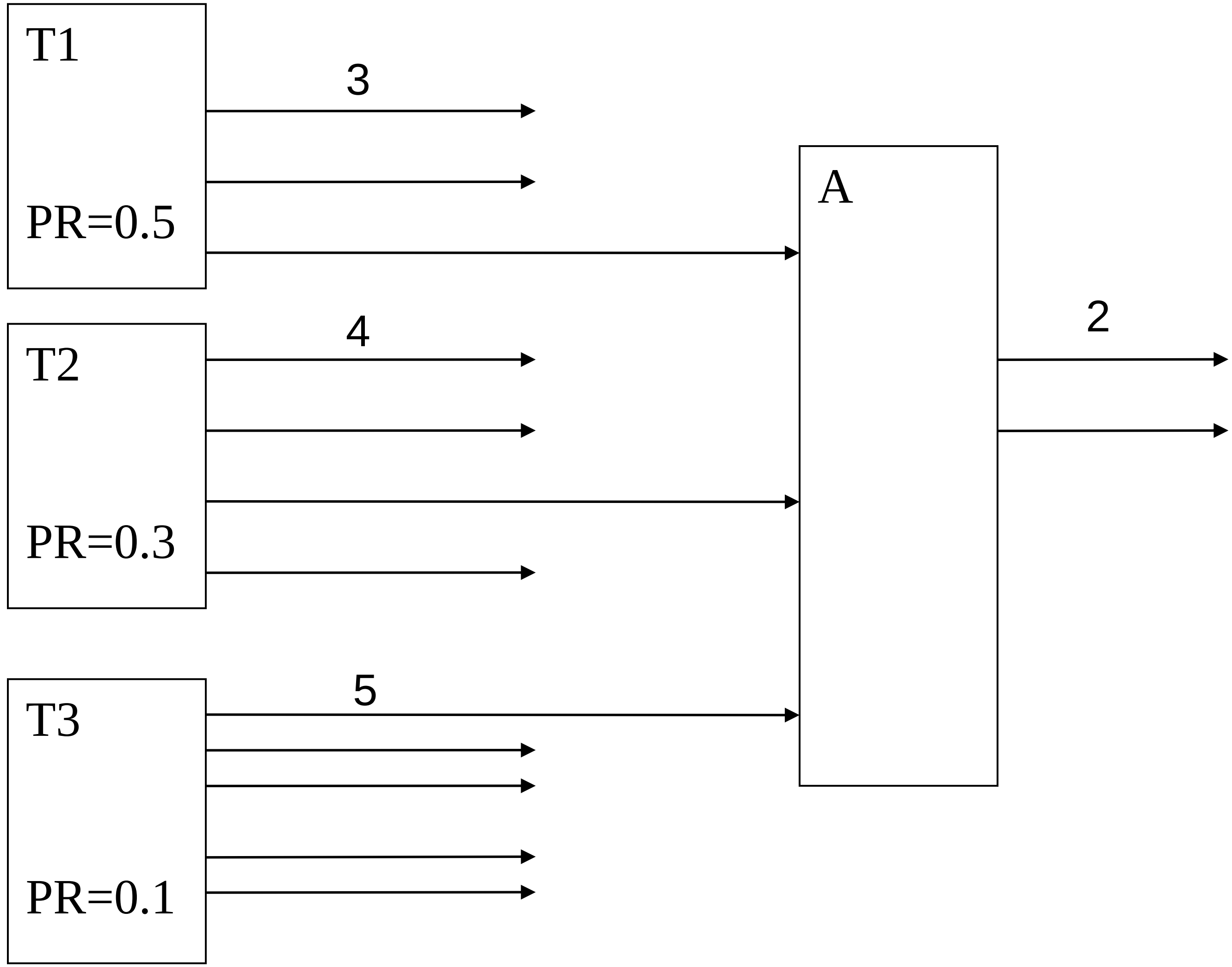


PageRank: Formula

Given page A, and pages T_1 through T_n linking to A, PageRank of A is defined as:

$$PR(A) = (1\text{-}d) + d\,(PR(T_1)/C(T_1) + \ldots + PR(T_n)/C(T_n))$$

- C(P) is the out-degree of page P
- d is the “random URL” factor (≈0.85)
- This is the stationary distribution of the Markov chain for the random walk.



PR(A)=(1-d) + d*(PR(T1)/C(T1) + PR(T2)/C(T2) + PR(T3)/C(T3))
=0.15+0.85*(0.5/3 + 0.3/4+ 0.1/5)

PageRank: Intuition & Computation

- Each page distributes its PR_i to all pages it links to. Linkees add up their awarded rank fragments to find their PR_{i+1}.
- *d* is the "random jump factor"

- Can be calculated iteratively : **PR_{i+1} is computed based on PR_i.**

$$PR_{i+1}(A) = (1-d) + d\,(PR_i(T_1)/C(T_1) + \ldots + PR_i(T_n)/C(T_n))$$