# Lecture 5 — Markov Chains & Random Walks → Monte Carlo Sampling

> Course **2-INF-135/15 Pravdepodobnostné algoritmy**, LS 2025/26. Source slides: `05_MC.pdf` (46 pages). Štátnicové syllabus topics covered here: *Markov chains, transition matrix and stationary distribution; connection to eigenvalues; mixing (variation distance), strong uniform stopping times; sampling via random walks (random graphs with prescribed degrees, random knapsack solutions); reversible chains and the Metropolis–Hastings algorithm (how to build a chain with a given stationary distribution).*

---

## The one-paragraph map of the whole lecture

A **Markov chain** is random motion with *no memory*: where you go next depends only on where you are now, not on how you got there. The deep fact is that a "nice" chain **forgets its starting point** — run it long enough and the probability of being in state $i$ converges to a single number $\pi_i$ that does **not** depend on where you began. That number is the **stationary distribution**.

This turns into a powerful algorithmic machine. Suppose you want to **sample** an object from some weird, exponentially large space $S$ according to a prescribed distribution $\sigma$ (a uniform independent set, a random knapsack solution, a random shuffle of a deck). You cannot list $S$. But if you can design a chain that *walks around* $S$ and whose stationary distribution is exactly $\sigma$, then you just run the walk and read off where you land. Two questions then dominate the entire lecture:

1. **How do I BUILD a chain with a prescribed stationary distribution $\sigma$?** → the idea of **reversibility / detailed balance**, culminating in the **Metropolis–Hastings** algorithm.
2. **How LONG must I run it?** → **mixing time**, measured in **variation distance**, and bounded three ways: **coupling**, **strong uniform stopping times**, and **eigenvalues** (expanders mix fast).

The closing payoff: **sampling lets you count** (estimate the size of huge sets — FPRAS, #DNF), and as a bonus, **expander walks recycle random bits** to amplify BPP using $O(k)$ bits instead of $O(kn)$.

---

## 1. What a Markov chain is

A **stochastic process** is just a family of random variables indexed by time, $X = \{X_t \mid t \in T\}$, where $X_t$ is "the state at time $t$." We assume **discrete time** ($t = 0, 1, 2, \ldots$) and a **discrete state space** $\{0, 1, 2, \ldots\}$ (finite or countable).

The process is a **Markov chain** if it has the **memorylessness (Markov) property**:

$$\Pr[X_t = a_t \mid X_{t-1} = a_{t-1}, X_{t-2} = a_{t-2}, \ldots, X_0 = a_0] \;=\; \Pr[X_t = a_t \mid X_{t-1} = a_{t-1}].$$

> **Intuition.** The *present screens off the past from the future.* The only thing the chain "knows" is its current state; the entire history before that is irrelevant to what happens next. Equivalently, for times $u < t < v$, the past ($X_u$) and the future ($X_v$) are **conditionally independent given the present** ($X_t$): $\Pr[X_u = i, X_v = k \mid X_t = j] = \Pr[X_u = i \mid X_t = j] \cdot \Pr[X_v = k \mid X_t = j]$.

### The transition matrix

All the dynamics live in a single matrix. Let

$$p_{i,j} = \Pr[X_t = j \mid X_{t-1} = i]$$

be the probability of stepping from $i$ to $j$ in one move (assumed **time-homogeneous** — the same at every $t$). Collect them into the **transition matrix** $P = (p_{i,j})$. Each **row sums to 1** (from $i$ you must go *somewhere*):

$$\sum_{j \geq 0} p_{i,j} = 1 \quad \text{for every } i.$$

A matrix with non-negative entries and rows summing to 1 is called **stochastic** — and *every* such matrix is the transition matrix of some chain.

### How distributions evolve — it's just matrix multiplication

Let $p_i(t) = \Pr[\text{system is in state } i \text{ at time } t]$ and collect these into a **row vector** $\bar{p}(t) = (p_0(t), p_1(t), \ldots)$, the distribution over states at time $t$. To be in state $i$ at time $t$, you were in *some* state $j$ at time $t-1$ and stepped $j \to i$:

$$p_i(t) = \sum_{j \geq 0} p_j(t-1)\, p_{j,i} \qquad \Longleftrightarrow \qquad \boxed{\bar{p}(t) = \bar{p}(t-1)\, P}$$

Iterating, the **$m$-step transition matrix** is just the matrix power $P^{(m)} = P^m$, and

$$\bar{p}(t+m) = \bar{p}(t)\, P^m.$$

> **Punchline.** A Markov chain *is* a stochastic matrix. Running the chain = repeatedly multiplying a distribution vector by $P$. Everything that follows — convergence, stationary distributions, mixing rates — is secretly a statement about the powers $P^m$ and the **eigenvalues** of $P$. Hold that thought; it returns in §11 (expanders).

---

## 2. First examples — random walks that *solve* problems (2SAT, 3SAT)

Before the theory, two examples show *why we care*: a random walk on assignments solves satisfiability, and the chain analysis gives the running time.

### 2SAT — the symmetric walk, $O(n^2)$

**Algorithm (Papadimitriou).** Start from any truth assignment $\alpha_0$. While the formula is unsatisfied, pick an unsatisfied clause and **flip a uniformly random one of its two literals**. Repeat.

Why does this make progress? Fix a satisfying assignment $\alpha^*$ (assume one exists). An unsatisfied 2-clause has both its literals wrong *under the current assignment*, but $\alpha^*$ satisfies the clause, so $\alpha^*$ disagrees with us on **at least one** of those two variables. Flipping a random one of the two therefore moves us *closer* to $\alpha^*$ with probability $\geq \frac{1}{2}$.

Track $j$ = the number of variables on which we currently **agree** with $\alpha^*$ (Hamming distance is $n - j$). This is (a lower bound on) a chain on $\{0, 1, \ldots, n\}$:

- at $j = n$ we have found a satisfying assignment: $h_n = 0$;
- at $j = 0$ any flip increases agreement: $h_0 = 1 + h_1$;
- in between, with prob $\frac{1}{2}$ we go to $j+1$, with prob $\frac{1}{2}$ to $j-1$: $h_j = 1 + \frac{1}{2}h_{j-1} + \frac{1}{2}h_{j+1}$.

Here $h_j =$ expected number of steps to reach $\alpha^*$ starting from agreement $j$. Solving the recurrence (it telescopes to $h_j = h_{j+1} + (2j+1)$) gives

$$\boxed{h_0 = n^2}$$

so the symmetric walk finds a satisfying assignment in **expected $O(n^2)$ steps**. (This is the classic gambler's-ruin / drift argument: a fair walk on a line of length $n$ takes $\Theta(n^2)$ time to cross it.)

### 3SAT — the biased walk, and why restarts win

For 3SAT an unsatisfied 3-clause has *three* literals; $\alpha^*$ disagrees on at least one, so a random flip moves us toward $\alpha^*$ with probability only $\frac{1}{3}$ and **away** with probability $\frac{2}{3}$. The recurrence becomes biased:

$$h_j = 1 + \tfrac{2}{3}h_{j-1} + \tfrac{1}{3}h_{j+1}, \qquad h_j = h_{j+1} + (2^{\,j+2} - 3).$$

A walk biased *away* from the target takes **exponential** time: $h_0 = O(2^n)$. Running one long walk is hopeless.

> **The fix (Schöning).** Don't run one long walk — run **many short walks from fresh random starts**. The key insight on the slide: "*if a satisfying $\alpha$ exists, then there are many*" assignments from which a short walk reaches it. A random start lands within a favourable distance often enough that capping

each walk at $O(n)$ steps and restarting gives $h_0 = O\big(n^{3/2}\,(4/3)^n\big)$, dramatically better than $2^n$. (This is the random-walk k-SAT algorithm previewed in the derandomization lecture — derandomized there by sweeping a covering code instead of guessing the start.)

**Professor-pleaser.** The *same* random-flip rule is polynomial for 2SAT and exponential for 3SAT, and the **only** thing that changed is the **drift** of the induced walk on Hamming distance: fair coin ($\frac{1}{2}/\frac{1}{2}$) → $O(n^2)$; biased coin ($\frac{1}{3}/\frac{2}{3}$) → $2^n$. The behaviour of the algorithm is *entirely* the behaviour of the underlying Markov chain. This is the whole reason to study chains.

---

## 3. Classifying states — which chains "settle down"?

For a chain to forget its start and converge, it must be structurally well-behaved. The vocabulary (this is the part to be able to *recite* in the oral):

- **Reachable** ($j$ reachable from $i$): $p_{i,j}^{(t)} > 0$ for some $t$. **Communicating** states: each reachable from the other.
- **Irreducible** chain: every state communicates with every other — i.e. the directed graph of the chain is a **single strongly connected component**. (You can get from anywhere to anywhere.)
- **Recurrent / persistent** state $i$: you return to it with certainty. Let $r_{i,j}^{(t)} = \Pr[\text{first visit to } j \text{ from } i \text{ is at time } t]$. Then $i$ is recurrent iff $f_{i,i} = \sum_{t\ge 1} r_{i,i}^{(t)} = 1$.
- **Transient (prechodový)** state: $f_{i,i} = \sum_t r_{i,i}^{(t)} < 1$ — there is positive probability you *never* come back.
- **Absorbing** state: once entered, never left ($p_{i,i} = 1$).
- A **recurrent chain** has all states recurrent.

Among recurrent states there is a subtle but exam-relevant split. The **mean return time** is $h_{i,i} = \sum_{t\ge 1} t \cdot r_{i,i}^{(t)}$.

- **Positive recurrent**: $h_{i,i} < \infty$ (you come back, and in finite *expected* time).
- **Null recurrent**: $h_{i,i} = \infty$ (you come back with probability 1, but the expected wait is infinite!).

**The null-recurrent example (worth knowing — it's a classic trap).** Consider a chain on $\{1, 2, 3, \ldots\}$ where from $i$ you go *up* to $i+1$ with probability $\frac{i}{i+1}$ and *home* to $1$ with probability $\frac{1}{i+1}$. The probability of first returning to $1$ at exactly time $t$ is $r_{1,1}^{(t)} = \underbrace{\tfrac{1}{2}\cdot\tfrac{2}{3}\cdots\tfrac{t-1}{t}}_{\text{climb } 1\to t}\cdot\underbrace{\tfrac{1}{t+1}}_{t\to 1} = \tfrac{1}{t}\cdot\tfrac{1}{t+1} = \tfrac{1}{t(t+1)}$. Then $f_{1,1} = \sum_t \frac{1}{t(t+1)} = 1$ (telescopes), so state 1 **is** recurrent — but $h_{1,1} = \sum_t t\cdot\frac{1}{t(t+1)} = \sum_t \frac{1}{t+1} = \infty$. **Null recurrent.** You always come home, but the average homecoming time is infinite. *This cannot happen in a finite chain* (next lemma).

### Two equivalent tests for recurrence

A clean characterisation via the $m$-step return probabilities:
$i$ **is recurrent** $\iff \sum_{t=1}^{\infty} p_{i,i}^{(t)} = \infty$, $\quad$ $i$ **is transient** $\iff \sum_{t=1}^{\infty} p_{i,i}^{(t)} < \infty$. *Why:* if $i$ is transient with return probability $q < 1$, the number of returns is geometric, and $\sum_t p_{i,i}^{(t)} = \sum_n q^n = \frac{q}{1-q} < \infty$. If recurrent, each return is certain so the expected number of returns — which equals this sum — is infinite.

### Periodicity and ergodicity

- **Period** of state $i$: $\gcd\{t : p_{i,i}^{(t)} > 0\}$ — the gcd of all times at which return is possible. **Aperiodic** = period 1.

  > *Quick check the slide asks:* a random walk on an **undirected cycle** has period **2 if the length is even** (you can only come back after an even number of steps) and period **1 if the length is odd** (odd cycles let you "turn around"). Bipartiteness = period 2. This is exactly why we add self-loops to kill periodicity later.

- **Ergodic state**: aperiodic **and** positive recurrent — "you come back, in finite expected time, but at *irregular* times." An **ergodic chain** has all states ergodic.

**Lemma (finite chains are well-behaved).** In a *finite* Markov chain:

1. there is at least one recurrent state;
2. all recurrent states are **positive** recurrent (no null recurrence in finite chains).

### Stationary distribution

A distribution $\bar{\pi}$ is **stationary** if it is a fixed point of the dynamics: $\boxed{\bar{\pi} = \bar{\pi}P}$ i.e. if you start in $\bar{\pi}$, after one step you are *still* in $\bar{\pi}$ (and hence forever). It is a **left eigenvector of $P$ with eigenvalue 1**.

---

## 4. The Fundamental Theorem of Markov Chains

This is *the* theorem of the lecture — the precise statement that "nice chains forget their start."

> **Theorem (Fundamental / hlavná veta).** In an **irreducible, finite, aperiodic** Markov chain:
>
> 1. every state is **ergodic**;
> 2. there exists a **unique** stationary distribution $\pi$;
> 3. $f_{i,i} = 1$ and the mean return time is $h_{i,i} = \dfrac{1}{\pi_i}$;
> 4. the long-run **fraction of time** spent in $i$ converges to $\pi_i$: $\lim\limits_{t\to\infty} \dfrac{N(i,t)}{t} = \pi_i$, where $N(i,t)$ counts visits to $i$ in $t$ steps.

> Equivalently (the "convergence" form): 3′. $\lim_{t\to\infty} P^t_{j,i}$ **exists and is independent of the starting state** $j$; 4′. $\pi_i = \lim_{t\to\infty} P^t_{j,i} = \dfrac{1}{h_{i,i}}$.

Read parts 3′–4′ slowly — they *are* the magic:

> **Intuition.** $P^t_{j,i}$ is the probability of being at $i$ after $t$ steps **given you started at** $j$. The theorem says this stops depending on $j$. The chain **erases the memory of its origin.** Where you end up is governed only by the chain's structure, summarised in $\pi$. And $\pi_i = 1/h_{i,i}$ has a lovely reading: *a state you return to quickly (small mean return time) is a state you spend a lot of time in* (large stationary probability). Frequently visited ⇔ short return time.

The three hypotheses each pull their weight:

- **irreducible** → the limit can't depend on $j$ (otherwise disconnected pieces keep separate fates);
- **aperiodic** → the limit *exists* (a period-2 chain oscillates forever and never settles — recall the even cycle);
- **finite** → positive recurrence is automatic (no null-recurrent escape to infinity).

---

## 5. Computing $\pi$ — cuts and reversibility (the load-bearing trick)

How do you actually *find* $\pi$? Solve $\pi P = \pi$ with $\sum_i \pi_i = 1$. Written out component-wise, the equation $\pi_i = \sum_j \pi_j P_{j,i}$ is a **balance / cut** condition:

$$\sum_j \pi_j P_{j,i} \;=\; \pi_i \;=\; \pi_i \sum_j P_{i,j}.$$

The left side is **flow into** $i$; the right side is **flow out of** $i$. Stationarity = *global balance*: total probability flowing into each state equals total flowing out. Solving the full linear system is doable but painful. There is a much stronger, much easier-to-check sufficient condition.

> **Theorem (detailed balance ⇒ stationary).** Let the chain be finite, irreducible, ergodic with matrix $P$, and let $\pi = (\pi_1, \dots, \pi_n)$ be non-negative with
>
> 1. $\sum_i \pi_i = 1$, and
> 2. $\pi_i P_{i,j} = \pi_j P_{j,i}$ **for every pair** $i, j$ (the *detailed balance* / **reversibility** condition).
>
> Then $\pi$ is the (unique) stationary distribution, and the chain is **time-reversible**.

*Proof — one line.* Sum detailed balance over $i$: $\sum_i \pi_i P_{i,j} = \sum_i \pi_j P_{j,i} = \pi_j \sum_i P_{j,i} = \pi_j$, which is exactly the stationarity equation $(\pi P)_j = \pi_j$. ■

> **Why this is the whole game.** Global balance (the true stationarity equation) couples *all* states together — a big linear system. **Detailed balance** is a *local, pairwise* condition: just check each edge $i \leftrightarrow j$ in isolation. It is strictly stronger (not every chain is reversible), but when it holds it is *trivial* to verify, and — crucially — it is the lever we use to **engineer** a chain with a *prescribed* $\pi$. "Reversible" means the chain looks statistically the same run forwards or backwards in time. **Remember this condition; the entire Metropolis construction in §8–§9 is just "rig the transition probabilities so detailed balance holds for the $\pi$ I want."**

---

## 6. Random walks on undirected graphs

The most important concrete chain. Given a connected undirected graph $G = (V, E)$, the **simple random walk** steps from $v$ to a uniformly random neighbour: $P_{i,j} = \frac{1}{\deg(i)} \quad \text{if } (i,j) \in E, \text{ else } 0.$

(Aperiodic $\iff$ $G$ is **not bipartite** — same parity-of-cycles fact as before.)

> **Theorem.** The simple random walk on a connected non-bipartite graph converges to the stationary distribution $\boxed{\pi_v = \frac{\deg(v)}{2|E|}}$ — proportional to degree. High-degree vertices are visited more.

*Check.* It sums to 1: $\sum_v \frac{\deg(v)}{2|E|} = \frac{2|E|}{2|E|} = 1$ (handshake lemma). And it satisfies **detailed balance** — the cleanest possible verification: $\pi_u P_{u,v} = \frac{\deg(u)}{2|E|} \cdot \frac{1}{\deg(u)} = \frac{1}{2|E|} = \frac{\deg(v)}{2|E|} \cdot \frac{1}{\deg(v)} = \pi_v P_{v,u}$. Both directions of an edge carry the *same* flow $\frac{1}{2|E|}$. (This is exactly why §5's reversibility machinery is so handy.)

By the Fundamental Theorem, the **mean return time** to $u$ is $h_{u,u} = \frac{1}{\pi_u} = \frac{2|E|}{\deg(u)}$.

### Hitting times and cover time — why a walk explores fast

**Hitting time on an edge.** If $(u,v) \in E$, then the expected time to go from $v$ to $u$ is bounded: $h_{v,u} < 2|E|$. *Proof.* Condition the return-to-$u$ time on the first step out of $u$: $\frac{2|E|}{\deg(u)} = h_{u,u} = \frac{1}{\deg(u)} \sum_{w \in N(u)} (1 + h_{w,u})$. Multiply by $\deg(u)$: $2|E| = \sum_{w \in N(u)} (1 + h_{w,u})$. Every term is positive, so each single $1 + h_{w,u} < 2|E|$; in particular $h_{v,u} < 2|E|$. ∎

**Cover time** = expected time to visit **all** vertices, from the worst start.

> **Theorem.** $\text{cover time} < 4|V|\,|E|$.

*Proof idea (elegant).* Fix a **spanning tree**. Walk the tree in a fixed closed tour (e.g. a DFS traversal, around the tree) that crosses each of its $|V|-1$ edges once in each direction — $2(|V|-1)$ edge-traversals. The expected time for each traversal of an edge $(v_i, v_{i+1})$ is $h_{v_i,v_{i+1}} < 2|E|$. Summing,
$\text{cover time} \le \sum_i h_{v_i,v_{i+1}} < 2(|V|-1)\cdot 2|E| < 4|V|\,|E|.$ ■

> **Why this matters algorithmically.** A random walk explores a connected graph in polynomial time *using essentially no memory* — you only remember your current vertex. The headline application: **undirected $s$–$t$ connectivity (USTCON) in randomized log-space**. To decide whether $t$ is reachable from $s$, just walk from $s$ for $O(|V||E|) = \text{poly}$ steps; if you hit $t$, accept. The walk needs only $O(\log n)$ bits (the current vertex + a counter). This is the basis of the result $\mathbf{USTCON} \in \mathbf{RL}$ — and famously was later *derandomized* to $\mathbf{L}$ by Reingold. The cover-time bound is exactly the running-time guarantee.

---

## 7. From walks to sampling — the Monte Carlo Markov Chain (MCMC) idea

Here is the conceptual pivot of the lecture.

> **Problem.** Sample an object from a huge space $S$ according to a prescribed distribution $\sigma$. ($S$ is exponential — you cannot enumerate it.)
>
> **Solution (MCMC).** Design an **ergodic Markov chain** such that
>
> - its **state space is $S$** (the objects you want to sample), and
> - its **stationary distribution is the desired $\sigma$**.
>
> Then *run the chain*. By the Fundamental Theorem it converges to $\sigma$ from **any** starting state, so after enough steps the current state is an (almost) $\sigma$-sample.

To get *several near-independent* samples, run a long walk $x_0, x_1, x_2, \ldots$ and take states spaced $r$ apart: $x_0 \xrightarrow{r} x_r \xrightarrow{r} x_{2r} \xrightarrow{r} x_{3r} \cdots$ If $r$ is at least the mixing time, $x_r, x_{2r}, x_{3r}, \ldots$ are nearly independent $\sigma$-samples.

Two quantities control the cost, and they organize the rest of the lecture:

- **$r$ = the mixing time** — how long until one step "looks like" $\sigma$. (The hard part; §10–§11.)
- the **cost of one step** — want a *small neighbourhood* $N(x)$ so each move is cheap.

> **The design freedom and the two sub-problems.** We get to *invent* the chain. That raises exactly the two questions from the map: (1) how do we force the stationary distribution to be the $\sigma$ we want

> (uniform? weighted?) — §8–§9; and (2) how do we bound $r$ so we know when to stop — §10–§11.

We assume a **symmetric neighbourhood structure** $\{N(x)\}$ on $S$: $y \in N(x) \iff x \in N(y)$ (an undirected "move graph" on the objects), and we want $N(x)$ small.

---

## 8. Building a chain with **uniform** stationary distribution

First the easy target: $\sigma =$ **uniform** on $S$. Naïvely walking the move-graph gives $\pi_v \propto \deg(v)$ (§6) — **not** uniform, because different objects have different numbers of neighbours. The fix is to *equalize* the transition probabilities by **padding with self-loops**.

> **Lemma (uniform sampler).** Let $\Omega$ be a finite state space with neighbourhood structure $\{N(x)\}$, $N = \max_x |N(x)|$. Pick any $M > N$ and define the chain
> $$P(x,y) = \begin{cases} 1/M & x \neq y,\ y \in N(x), \\ 0 & x \neq y,\ y \notin N(x), \\ 1 - |N(x)|/M & x = y \quad \text{(self-loop)}. \end{cases}$$
> If this chain is irreducible and aperiodic, its stationary distribution is **uniform**, $\pi_x = 1/|\Omega|$.

*Proof.* Every off-diagonal transition probability is the *same constant* $1/M$, regardless of $x$ or $y$ (the self-loop soaks up the difference in degrees). So $P(x,y) = P(y,x) = 1/M$ — the matrix is **symmetric**. Detailed balance with the uniform $\pi_x = \pi_y = 1/|\Omega|$ reads $\pi_x P(x,y) = \pi_y P(y,x)$, which holds because both $\pi$'s and both $P$'s are equal. By §5, uniform is stationary. ■

> **The idea in one phrase.** *Self-loops are a thermostat for degree.* Every vertex "spends" $|N(x)|/M$ of its probability stepping out and parks the remaining $1 - |N(x)|/M$ on itself, so **outgoing edges all carry the same weight** $1/M$. Equal edge weights ⇒ symmetric matrix ⇒ uniform stationary. We choose $M > N$ strictly so every state has a positive self-loop, which also **kills periodicity** for free.

### Example — sampling a uniform Independent Set

State space $\Omega =$ all independent sets of $G$. Neighbourhood: $N(x) =$ independent sets differing from $x$ in **one vertex**. The chain:

> **Uniform-IS walk.**
>
> 1. $X_0 \leftarrow$ any independent set.
> 2. To compute $X_{i+1}$: pick a uniform random vertex $v \in V$.
>    - if $v \in X_i$: set $X_{i+1} \leftarrow X_i \setminus \{v\}$ (remove);
>    - if $v \notin X_i$ and $X_i \cup \{v\}$ is independent: $X_{i+1} \leftarrow X_i \cup \{v\}$ (add);

- otherwise $X_{i+1} \leftarrow X_i$ (stay — this is the self-loop).

Picking $v$ among all $|V|$ vertices, and staying when the move is illegal, is exactly the "$M > N$ with self-loops" construction (here $M = |V|$). As long as $G$ has at least one edge there is always some illegal move from *some* state, guaranteeing a self-loop, hence aperiodicity. Stationary distribution: **uniform over all independent sets**.

---

## 9. Building a chain with a **prescribed non-uniform $\pi$ — Metropolis**

Now the general target: we want each object $x$ sampled with a *prescribed* probability $\pi_x$. Often $\pi$ is given only **up to normalization**: $\pi_x = \frac{b(x)}{B}, \qquad b(x) > 0, \quad B = \sum_{x \in \Omega} b(x),$ and the normalizer $B$ (a sum over the entire exponential space!) is **unknown and hopeless to compute**. The triumph of Metropolis is that *we never need* $B$.

**Lemma (Metropolis).** Given $\{N(x)\}$, $N = \max_x |N(x)|$, $M > N$, and any desired $\pi_x > 0$, define

$$P(x,y) = \begin{cases} \dfrac{1}{M} \cdot \min\left\{1, \dfrac{\pi_y}{\pi_x}\right\} & x \neq y,\ y \in N(x), \\ 0 & x \neq y,\ y \notin N(x), \\ 1 - \displaystyle\sum_{y \neq x} P(x,y) & x = y. \end{cases}$$

If irreducible and aperiodic, its stationary distribution is exactly $\pi$.

*Proof — detailed balance again.* Take $x \neq y$ neighbours with (WLOG) $\pi_x \leq \pi_y$. Then $\min\{1, \pi_y/\pi_x\} = 1$, so $P(x,y) = 1/M$; and $\min\{1, \pi_x/\pi_y\} = \pi_x/\pi_y$, so $P(y,x) = \frac{1}{M} \cdot \frac{\pi_x}{\pi_y}$. Check: $\pi_x P(x,y) = \frac{\pi_x}{M}, \qquad \pi_y P(y,x) = \pi_y \cdot \frac{1}{M} \cdot \frac{\pi_x}{\pi_y} = \frac{\pi_x}{M}.$ Equal — detailed balance holds, so $\pi$ is stationary. ■

**The two beautiful features.**

1. **Only ratios $\pi_y/\pi_x$ appear.** The unknown normalizer $B$ cancels: $\pi_y/\pi_x = b(y)/b(x)$. *We can sample from $\pi$ without ever knowing the total weight $B$* — exactly the situation in every real counting/physics application.
2. **The acceptance rule is a thermostat toward $\pi$.** Propose a uniform neighbour move ($1/M$). *Always accept a move to a more-probable state* ($\pi_y \geq \pi_x$); accept a move to a less-probable state only **with probability $\pi_y/\pi_x$**. Detailed balance is satisfied by construction, so the walk pools where $\pi$ is large.

### Metropolis for weighted Independent Sets

We want $\mathbf{Pr}[\textbf{independent set } I] \propto \lambda^{|I|}$ for a parameter $\lambda > 0$:

$$\pi_x = \tfrac{\lambda^{|I_x|}}{B}, \quad \text{so} \quad \begin{cases} \lambda = 1: & \textbf{uniform,} \\ \lambda < 1: & \textbf{smaller sets favoured,} \\ \lambda > 1: & \textbf{larger sets favoured (toward max IS).} \end{cases}$$

The ratio for adding a vertex is $\pi_y/\pi_x = \lambda^{|I|+1}/\lambda^{|I|} = \lambda$; for removing, $1/\lambda$. So the algorithm needs only $\lambda$:

> **Metropolis-IS.**
>
> 1. $X_0 \leftarrow$ some independent set.
> 2. To compute $X_{i+1}$: pick $v \in V$ uniformly (prob $1/M$, $M = |V|$).
>    - if $v \in X_i$: **remove** it with probability $\min\{1, 1/\lambda\}$;
>    - if $v \notin X_i$ and $X_i \cup \{v\}$ independent: **add** it with probability $\min\{1, \lambda\}$;
>    - otherwise stay.

The detailed-balance check ($\pi_x P(x,y) = \pi_y P(y,x)$) goes through with the $\lambda$ ratios, and — the punchline on the slide — **"$B$ sme nepotrebovali :-)"**: we never needed the normalizer.

### Metropolis–Hastings — the general recipe (asymmetric proposals)

What if the *proposal* mechanism itself is not symmetric? Let $Q$ be **any** irreducible proposal chain ($q_{ij} =$ probability of proposing $j$ from $i$), and $\pi$ the desired distribution.

> **Metropolis–Hastings.** From $i$, propose $j$ according to $Q$. Accept the move with probability $\alpha = \min\left\{ \frac{\pi_j\, q_{ji}}{\pi_i\, q_{ij}},\ 1 \right\}$ (when $Q$ is **symmetric**, $q_{ij} = q_{ji}$, this collapses to the plain Metropolis $\alpha = \min\{\pi_j/\pi_i,\ 1\}$). Otherwise stay at $i$.

The resulting transition $p_{ij} = q_{ij}\,\alpha$ satisfies detailed balance with $\pi$: the correction factor $q_{ji}/q_{ij}$ exactly cancels the asymmetry of the proposal, so $\pi_i p_{ij} = \pi_j p_{ji}$.

> **Professor-pleaser — the unifying statement of §8–§9.** *Metropolis–Hastings is a universal machine for manufacturing a Markov chain with any prescribed stationary distribution, using only the **ratios** of the target probabilities.* It works by **forcing detailed balance** (§5): start from any convenient proposal walk, then bend the acceptance probabilities until $\pi_i p_{ij} = \pi_j p_{ji}$ holds on every edge. Reversibility is not a curiosity — it is the *design principle* of all of MCMC.

---

## 10. How long? — variation distance, mixing time, and coupling

We can build the chain; now we must know **when its distribution is close enough to $\pi$**. We need (a) a notion of "close," and (b) tools to bound the time to get there.

### Variation distance — the right notion of "close"

For two distributions $D_1, D_2$ on a countable $S$, the **total variation distance** is
$\|D_1 - D_2\| := \frac{1}{2}\sum_{x\in S}|D_1(x) - D_2(x)|$.

> **Lemma (the event form).** $\|D_1 - D_2\| = \max_{A\subseteq S}|D_1(A) - D_2(A)|$ where $D_i(A) = \sum_{x\in A} D_i(x)$.

> **Intuition.** Variation distance is the **largest disagreement on any event**: no matter what yes/no question $A$ you ask, the two distributions assign $A$ probabilities differing by at most $\|D_1 - D_2\|$. The maximizing $A$ is "all $x$ where $D_1 > D_2$." The factor $\frac{1}{2}$ makes the range $[0, 1]$. This is the *operational* meaning: if variation distance is $\le \varepsilon$, **no statistical test can tell the two apart with advantage more than $\varepsilon$.**

### Mixing time

Let $\pi^*$ be stationary, and $p_x^t$ the distribution after $t$ steps starting from $x$. Define
$\Delta_x(t) = \|p_x^t - \pi^*\|, \qquad \Delta(t) = \max_{x\in S}\Delta_x(t),$
$\tau_x(\varepsilon) = \min\{t : \Delta_x(t) \le \varepsilon\}, \qquad \tau(\varepsilon) = \max_x \tau_x(\varepsilon) \quad \textbf{(mixing time).}$ A chain is **rapidly mixing** if $\tau(\varepsilon)$ is **polylog** in $1/\varepsilon$ (and polynomial in the input size) — fast enough to be useful as a sampler. The whole $\varepsilon$-uniform sampling goal is: run until $\|D - U\| \le \varepsilon$.

### Coupling — the workhorse for bounding mixing time

How do you prove a chain mixes fast? **Coupling** is the dominant technique.

> **Definition.** A **coupling** of a chain $M$ (on $S$) is a chain $Z_t = (X_t, Y_t)$ on $S \times S$ such that **each coordinate, viewed alone, is a faithful copy of $M$**:
> $\Pr[X_{t+1} = x' \mid Z_t = (x, y)] = \Pr[M_{t+1} = x' \mid M_t = x]$,
> $\Pr[Y_{t+1} = y' \mid Z_t = (x, y)] = \Pr[M_{t+1} = y' \mid M_t = y]$. The two copies may be *correlated* however we like (that's the design freedom), as long as the marginals are correct.

> **Lemma (Coupling Lemma).** If a coupling $Z_t = (X_t, Y_t)$ satisfies, for some time $T$,
> $\Pr[X_T \ne Y_T \mid X_0 = x, Y_0 = y] \le \varepsilon \quad \textbf{for all } x, y,$ then the mixing time obeys $\tau(\varepsilon) \le T$.

> **Why coupling works — the picture.** Run two copies of the chain at once: one started at the worst state $x$, the other started **in the stationary distribution $\pi$** (so the second copy is *always* exactly $\pi$). Couple their randomness so that **once they meet, they move together forever**. At any time $t$, the variation distance of copy 1 from $\pi$ is at most the probability the two copies have **not yet met**. So: *"design a coupling that makes the two copies collide quickly, and you have bounded the mixing time."* When the copies are close, the chain is close to stationary.

---

## 11. Card shuffling — three couplings, three mixing analyses

Shuffling a deck of $n$ cards *is* a random walk on the symmetric group $S_n$ whose stationary distribution is **uniform** (every permutation equally likely). "How many shuffles to mix" = mixing time. Three classic shuffles, three techniques.

### (a) Random transposition-to-top — coupling ⇒ coupon collector

A first attempt (move a random *position* to the top in both decks) **does not couple well**. The clever coupling:

> Pick a uniform **card value** $C \in \{1, \dots, n\}$ (not a position). In deck $X$, move the card with value $C$ to the top; in deck $Y$, also move the card with value $C$ to the top.

Now the key observation: **once a card has been "named" $C$, it sits in the same relative position in both decks forever after.** So the two decks become identical as soon as *every* value has been chosen at least once — which is exactly the **coupon collector** problem. After $n \ln n + cn$ steps, the probability a *specific* card was never chosen is
$\Pr[\text{specific card not yet on top}] \le \left(1 - \frac{1}{n}\right)^{n \ln n + cn} \le e^{-(\ln n + c)} = \frac{e^{-c}}{n}$, so by a union bound over $n$ cards, $\Pr[\text{some card never chosen}] \le e^{-c}$. By the Coupling Lemma, the mixing time is $n \ln n + O(n)$ — $\Theta(n \log n)$ **shuffles**.

### (b) Top-in-at-random — strong uniform stopping time

Take the top card and insert it into a **uniformly random position** in the deck. A second, equally powerful technique for proving "exactly uniform now":

> **Definition (strong uniform stopping time / silno uniformné pravidlo zastavenia).** A (randomized) stopping rule $T$ such that **for every $K$, conditioned on $T = K$, the permutation is *exactly* uniform.** Stopping at $T$ yields a *perfectly* uniform sample.

> **The stopping rule for top-in-at-random.** Track the card that was **originally on the bottom**. Stop the *first time it gets inserted* (reaches the top and goes in). Why is this strong uniform? Each card inserted *below* the original bottom card lands in a uniformly random position among those below it; by the time the original bottom card itself is finally inserted, the entire deck below it has been built up uniformly at random — so the full permutation is uniform, **regardless of how long that took**.

The time $T$ until the original bottom card rises and inserts is, by the same structure, a **coupon-collector** quantity again (the times $T_i - T_{i-1}$ to accumulate $i$ cards below match the inter-arrival times $V_{n-i+1} - V_{n-i}$ of the collector).

> **Lemma (stopping-time bound on variation distance).** If a shuffle has a strong uniform stopping time $T$, then $\|Q^{*k} - U\| \le \Pr[T > k]$.

*Proof (the clean computation).* For any event $S \subseteq S_n$, split on whether the process has stopped by time $k$:

$$Q^{*k}(S) = \sum_{j \le k} \underbrace{\Pr[X_k \in S \mid T = j]}_{= U(S),\ \text{stopped} \Rightarrow \text{uniform}} \Pr[T = j] \;+\; \Pr[X_k \in S \mid T > k] \Pr[T > k]$$

$$= U(S)\big(1 - \Pr[T > k]\big) + \Pr[X_k \in S \mid T > k] \Pr[T > k] = U(S) + \Pr[T > k] \underbrace{\big(\Pr[X_k \in S \mid T > k] - U(S)\big)}_{\in [-1,1]}.$$

Hence $|Q^{*k}(S) - U(S)| \le \Pr[T > k]$ for every $S$; take the max over $S$. ∎

With $k = \lceil n \ln n + cn \rceil$, the coupon-collector tail gives $\Pr[T > k] \le e^{-c}$, so again $\Theta(n \log n)$ shuffles suffice.

### (c) Riffle shuffle — birthday paradox, $\frac{3}{2}\log_2 n$ shuffles

The realistic shuffle: split the deck into two parts and interleave. The cleanest analysis runs the **inverse** shuffle (assign each card a random bit, pull the 0-cards to the top keeping relative order — equivalent to the forward riffle).

After $k$ inverse shuffles each card carries a $k$-bit label (its sequence of bits). The **strong uniform stopping rule: stop when all $n$ labels are distinct** — at that moment the cards are sorted by a uniformly random ordering, hence uniform. With $K = 2^k$ possible labels, "all $n$ distinct" is exactly the **birthday problem**:

> **Theorem.** After $k$ riffle shuffles of $n$ cards, $\|\mathrm{Rif}^{*k} - U\| \le \Pr[T > k] = 1 - \prod_{i=1}^{n-1} \left(1 - \frac{i}{2^k}\right)$.

By the birthday paradox the labels become distinct once $2^k \gg n^2$, i.e. $k \approx 2\log_2 n$; a sharper analysis gives the famous $\frac{3}{2}\log_2 n \approx 7$ **shuffles for** $n = 52$ threshold.

> **Professor-pleaser — three shuffles, three classics.** The *same* mixing question is answered three different ways, each reducing the analysis to an elementary probability gem: **coupon collector** (random-to-top, via coupling), **coupon collector again** (top-in-at-random, via strong uniform stopping time), and the **birthday paradox** (riffle). The two proof techniques — *coupling* and *strong uniform stopping times* — are the two universal hammers for upper-bounding mixing time, and both reduce "distribution is close to uniform" to "a simple combinatorial event has occurred."

---

## 12. Eigenvalues, expanders, and rapid mixing

Recall $\bar{p}(t) = \bar{p}(0)P^t$ — mixing is governed by the **powers of $P$**, hence by its **eigenvalues**. This is the spectral route to bounding mixing time, and the natural home of **expanders**.

### Spectral facts for graphs

For an undirected (multi)graph let $A(G)$ be the adjacency matrix (symmetric, so real eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ with an orthonormal eigenbasis). Key facts (state these; they recur):

- $G$ **connected** $\Rightarrow \lambda_2 < \lambda_1$ (the top eigenvalue is simple);
- $|\lambda_i| \leq d$ for all $i$, where $d$ is the max degree;
- $d$ is an eigenvalue $\iff$ $G$ is $d$-regular; then $\lambda_1 = d$ with eigenvector $e_1 = \frac{1}{\sqrt{n}}(1, \ldots, 1)$ (the all-ones / uniform direction);
- $G$ **bipartite** $\iff$ the spectrum is symmetric ($\lambda \leftrightarrow -\lambda$); for connected $G$, bipartite $\iff$ $-\lambda_1$ is an eigenvalue. A $d$-regular bipartite graph has $\lambda_n = -d$.

> **The reading.** $\lambda_1 = d$ is the uniform/stationary direction. The **spectral gap** $d - \lambda_2$ measures how fast everything *else* decays under repeated multiplication by $P = A/d$: a big gap ($\lambda_2$ far below $d$) means all non-uniform components die quickly $\Rightarrow$ **fast mixing**. The negative end ($\lambda_n$ near $-d$) signals near-bipartiteness $\Rightarrow$ near-periodicity, which we fix with self-loops.

### Expanders

An $(n, d, c)$**-expander** is a bipartite $d$-regular (multi)graph $G(X, Y, E)$ with $|X| = |Y| = n/2$ that expands every subset: $\forall S \subseteq X: \quad |\Gamma(S)| \geq \left(1 + c\left(1 - \frac{2|S|}{n}\right)\right)|S|$. Every set has many neighbours — the graph has no bottlenecks.

> **Theorem (expansion ⇔ spectral gap — Cheeger-type).**
>
> - If $G$ is an $(n, d, c)$-expander, then $|\lambda_2| \leq d - \dfrac{c^2}{1024 + 2c^2}$.

- Conversely, if $|\lambda_2| \le d - \varepsilon$, then $G$ is an $(n, d, c)$-expander with $c \ge \dfrac{2d\varepsilon - \varepsilon^2}{d^2}$.

The two notions — **combinatorial expansion** (no small cuts) and a **spectral gap** (large $d - \lambda_2$) — are *quantitatively equivalent*. This is the deep structural fact behind all of expander theory.

### Random walk on an expander mixes rapidly

Take the walk $P = A(G)/d$. Kill periodicity by adding self-loops: $Q = (I + P)/2$, whose eigenvalues shift to $\lambda_i' = \frac{1+\lambda_i/d}{2} \in [0, 1]$. If $\lambda_2 = d

- \varepsilon, *then* $\lambda_2' = 1 - \frac{\varepsilon}{2d} < 1$.

> **Theorem (rapid mixing).** For the lazy walk $Q$ on an $(n, d, c)$-expander with $\lambda_2 \le d - \varepsilon$, from any start $q^0$, the relative deviation from stationary obeys $\Delta(t) \le n^{1.5}\,(\lambda_2')^t \le n^{1.5}\big(1 - \frac{\varepsilon}{2d}\big)^t$.

Since $\lambda_2' < 1$ is a constant bounded away from 1, $\Delta(t)$ drops below any $\varepsilon$ after only $O(\log n)$ steps — **expanders mix in logarithmically many steps**. This is what "rapidly mixing" means in its purest form.

### ★ The application: amplifying BPP with $O(k)$ random bits

A gorgeous payoff, and a classic exam topic linking *this* lecture to the complexity lecture.

**Setup.** A BPP algorithm $A$ uses $n$ random bits and errs with probability $\le 1/3$. We want to push the error down to $1/2^k$. The textbook way: run $A$ on $k$ **independent** random strings $r_1, \dots, r_k$ and take the **majority**; Chernoff gives $\Pr[\text{majority of } A(x, r_1), \dots, A(x, r_k) \text{ wrong}] \le 2^{-\Omega(k)}$. But that costs $k \cdot n$ random bits. *Random bits are expensive.* **Can we get error $1/2^k$ using far fewer?**

> **Yes — walk on an expander.** Build an expander whose **vertices are the $n$-bit strings** (so $N = 2^n$), $d$-regular with $\lambda_2' \le \frac{1}{10}$ (e.g. degree $d = 7$, a $(N, 7, 2\alpha)$-expander, $Q = (I + A/7)/2$). Take a **single random walk** $X_0, X_1, \dots, X_{7k}$ on it, and use the visited vertices as the random strings:
> $$r_i = X_i, \qquad \text{compute } \mathrm{maj}\big(A(x, r_0), A(x, r_1), \dots, A(x, r_{7k})\big).$$

The cost: the **first** vertex needs $n$ bits, but each **step** of the walk only needs $O(\log d) = O(1)$ bits to choose a neighbour. Total: $n + O(k)$ **random bits** $\quad$ (vs. $kn$ for independent sampling).

> **Theorem.** $\Pr[\text{majority of } A(x, r_0), \dots, A(x, r_{7k}) \text{ wrong}] \le 1/2^k$.

> **Why it works (the deep point).** The walk's steps are **highly correlated** — yet because the expander mixes rapidly, the sequence of visited vertices behaves *almost like independent samples* for the purpose of a majority vote. The spectral gap guarantees the walk doesn't get "stuck" in a bad

region: the fraction of steps landing on bad strings concentrates just as it would for independent draws (an expander Chernoff bound). **Randomness is a costly resource, and expander walks let us recycle it** — getting the error reduction of $k$ independent trials while paying for only $O(k)$ extra bits. This is a cornerstone of the "hardness vs. randomness" / derandomization story.

---

## 13. From sampling to **counting** — Monte Carlo estimation and FPRAS

The final movement: once you can **sample**, you can **count / estimate**. This is the Monte Carlo method proper.

### Warm-up: estimating $\pi$ (the constant)

Throw $m$ uniform random points into the unit square; let $Z_i = 1$ if the $i$-th lands inside the quarter disc ($x^2 + y^2 \le 1$). With $W = \sum_i Z_i$, the fraction inside estimates the area $\pi/4$: $E[W] = \frac{m\pi}{4}$, $\quad$ **estimator** $P = \frac{4W}{m}$. Chernoff ($\Pr[|X-\mu| \ge \delta\mu] \le 2e^{-\mu\delta^2/3}$) on $W$ gives $\Pr\big[\,|P - \pi| \ge \varepsilon\pi\,\big] = \Pr\big[\,|W - E[W]| \ge \varepsilon E[W]\,\big] \le 2\,e^{-m\pi\varepsilon^2/12}$. So $P$ is within $\varepsilon\pi$ of $\pi$ with probability $\ge 1 - 2e^{-m\pi\varepsilon^2/12}$; this exceeds $1-\delta$ once $m \ge \frac{12\ln(2/\delta)}{\pi\,\varepsilon^2}$.

### The $(\varepsilon, \delta)$-approximation and FPRAS

> **Definition.** $X$ is an $(\varepsilon, \delta)$**-approximation** of a value $V$ if $\Pr\big[\,|X - V| \le \varepsilon V\,\big] \ge 1 - \delta$. (Relative error $\le \varepsilon$ with confidence $\ge 1 - \delta$.)

> **Theorem (sample-size for estimation).** Let $X_1, \dots, X_m$ be i.i.d. indicators with mean $\mu = E[X_i]$. If $m \ge \frac{3\ln(2/\delta)}{\varepsilon^2 \mu}$, then $\frac{1}{m}\sum_i X_i$ is an $(\varepsilon, \delta)$-approximation of $\mu$.

> **Definition (FPRAS).** A **fully polynomial randomized approximation scheme** for a quantity $V(X)$ is a randomized algorithm that, given input $X$ and $0 < \varepsilon, \delta < 1$, outputs an $(\varepsilon, \delta)$-approximation of $V(X)$ in time **polynomial in $|X|$, $1/\varepsilon$, and $\ln(1/\delta)$**.

> **The catch that drives everything.** The sample size $\frac{3\ln(2/\delta)}{\varepsilon^2\mu}$ blows up when $\mu$ is **tiny**. If the thing you're estimating is an *exponentially small fraction* of your sample space, naïve uniform sampling needs *exponentially* many samples to ever see a success. **"We need reasonable sampling"** — a sample space where the target is a non-negligible fraction. The art of counting via sampling is *choosing the right space*. The next example shows the fix.

### Case study — counting #DNF (the Karp–Luby coverage trick)

**Problem #DNF.** Given a DNF formula $F = C_1 \vee \cdots \vee C_t$, count $C(F)$ = the number of satisfying assignments. (Note: DNF *satisfiability* is trivial — one clause satisfiable; but *counting* solutions is #P-hard. Also $F$ satisfiable $\iff$ $\#\{\text{satassignments of } \neg F\} < 2^n$.)

**Naïve sampling fails.** Draw $\alpha \in_R \{0,1\}^n$, count fraction satisfying $F$, scale by $2^n$. Estimator $Y = (X/m)2^n$ has $E[Y] = C(F)$, and $X/m$ is an $(\varepsilon, \delta)$-approximation of $C(F)/2^n$ once $m \geq \frac{3 \cdot 2^n}{\varepsilon^2\, C(F)} \ln(2/\delta)$. If $C(F) \geq 2^n/\alpha(n)$ for a polynomial $\alpha$, this $m$ is polynomial — fine. **But if $C(F)$ is exponentially small, $m$ is exponential.** Hopeless in general.

**The fix — sample from a smartly chosen space where success is dense.** A clause $C_i$ with $\ell(i)$ literals has *exactly* $2^{n-\ell(i)}$ satisfying assignments; call that set $\mathrm{SC}_i$. Form the **multiset union** $U = \{(i, a) \mid 1 \leq i \leq t,\ a \in \mathrm{SC}_i\}, \qquad |U| = \sum_i |\mathrm{SC}_i| = \sum_i 2^{n-\ell(i)}$, which is **easy to compute and easy to sample from** (pick clause $i$ with probability $|\mathrm{SC}_i|/|U|$, then a uniform satisfying assignment of $C_i$). The true count is the size of the *set* union $C(F) = |\bigcup_i \mathrm{SC}_i|$. The ratio $\frac{C(F)}{|U|} = \frac{|\bigcup_i \mathrm{SC}_i|}{\sum_i |\mathrm{SC}_i|} \geq \frac{1}{t}$ because each satisfying assignment is counted **at most $t$ times** in $U$ (it satisfies at most $t$ clauses). *Success is now a $\geq 1/t$ fraction — polynomially dense!*

> **Coverage estimator (Karp–Luby).** Sample $(i, a)$ uniformly from $U$. Count it as a "success" iff $i$ is the **first** clause that $a$ satisfies (i.e. $a \notin \bigcup_{j<i} \mathrm{SC}_j$) — this picks exactly **one** representative per satisfying assignment, so the success probability is exactly $C(F)/|U|$. Then $Y = (X/m) \cdot |U|$ estimates $C(F)$ .

Because the success probability is $\geq 1/t$, the estimation theorem needs only $m = \left\lceil \frac{3t}{\varepsilon^2} \ln(2/\delta) \right\rceil$ samples — **polynomial**. This yields an **FPRAS for #DNF**.

> **The lesson (professor-pleaser).** You can't estimate the size of a needle-in-a-haystack set by uniform darts. **Karp–Luby's move: don't sample the haystack — sample a slightly larger, easy-to-handle space $U$ in which the target is a $\geq 1/t$ fraction, then correct for over-counting by keeping only the "first clause" representative.** Reducing a counting problem to sampling where the answer is *dense* is the master template of approximate counting.

### FPAUS — when even sampling is only approximate

Sometimes you cannot sample *exactly* from the target, only *almost* uniformly (this is where MCMC re-enters: the chain only reaches $\varepsilon$-close to stationary).

> **Definition.** A sampler $A$ on space $\Omega$ generates an **$\varepsilon$-uniform sample** if for all $S \subseteq \Omega$, $\left|\Pr[w \in S] - |S|/|\Omega|\right| \leq \varepsilon$ (i.e. variation distance from uniform $\leq \varepsilon$). A **FPAUS** (fully polynomial almost-uniform sampler) produces one in time polynomial in $|X|$ and $\ln(1/\varepsilon)$.

> **The grand connection.** FPAUS ⇒ FPRAS for many self-reducible problems (counting matchings, colourings, knapsack solutions, …): *almost-uniform sampling can be bootstrapped into approximate counting.* And the **FPAUS itself is built by MCMC** — design a rapidly mixing chain whose stationary distribution is (almost) uniform on $\Omega$ (§7–§9), run it for the mixing time (§10–§12), read off the state. This closes the entire loop of the lecture: **build the chain (reversibility/Metropolis) → bound the mixing time (coupling/eigenvalues) → sample → count.**

---

## The spine — one line per movement

> **Markov chain = memoryless random motion = a stochastic matrix $P$.** A *nice* (irreducible, finite, aperiodic) chain **forgets its start** and converges to a unique **stationary $\pi$** (Fundamental Theorem; $\pi_i = 1/h_{i,i}$). Stationarity is easiest to verify and to *engineer* via **detailed balance / reversibility** $\pi_i P_{ij} = \pi_j P_{ji}$. That gives a **sampling machine (MCMC)**: design a chain whose stationary distribution is the target $\sigma$ (uniform via self-loops; arbitrary via **Metropolis–Hastings**, using only probability *ratios* so the unknown normalizer cancels), run it past its **mixing time** (variation distance $\leq \varepsilon$), read off a sample. Mixing is bounded by **coupling**, **strong uniform stopping times** (card shuffles: coupon collector ×2, birthday paradox), or the **spectral gap** (expanders mix in $O(\log n)$ — and expander walks recycle random bits to amplify **BPP** with $O(k)$ bits). Finally, **sampling becomes counting**: $(\varepsilon, \delta)$-approximation → **FPRAS**, with #DNF (Karp–Luby coverage) as the model example, and **FPAUS** the bridge from almost-uniform sampling back to approximate counting.

| Theme | The one idea |
|---|---|
| Markov property | present screens off past from future; chain = matrix $P$ |
| Fundamental Theorem | nice chain forgets its start → unique $\pi$; $\pi_i = 1/h_{i,i}$ |
| Reversibility | local pairwise balance $\pi_i P_{ij} = \pi_j P_{ji}$ ⇒ stationary; the design lever |
| Random walk on graph | $\pi_v = \deg(v)/2 |
| MCMC | invent a chain whose $\pi$ = desired $\sigma$; sample by running it |
| Metropolis–Hastings | any prescribed $\pi$ from ratios only — normalizer cancels |
| Mixing / variation distance | $\|D_1-D_2\| = \max_A |
| Coupling | run two copies, bound time-to-meet ⇒ bound mixing |
| Strong uniform stopping | exactly uniform once a combinatorial event fires |
| Card shuffles | coupon collector (×2), birthday paradox ⇒ $\Theta(n \log n)$, $\frac{3}{2}\log_2 n$ |
| Expanders | expansion ⇔ spectral gap ⇒ $O(\log n)$ mixing; BPP error in $O(k)$ bits |
| FPRAS / #DNF | sample where the target is *dense* (Karp–Luby), then scale |

---

## Connections / threads to drill

- **Detailed balance = pairwise independence's cousin in chain-land**: a *local* condition that buys a *global* property, exactly like §5's "reversibility ⇒ stationary."
- **Coupling and strong uniform stopping** are the two universal mixing-time hammers — be ready to *state the Coupling Lemma and the stopping-time lemma* and run the coupon-collector card-shuffle analysis end-to-end.
- **Expander BPP amplification** ($O(k)$ bits, error $2^{-k}$) is the exam bridge between this lecture and the **complexity / derandomization** block — "randomness is expensive; recycle it." Ties to Lecture 3's error-reduction and Session 5's "BPP=P?" thread.
- **Karp–Luby #DNF** is the canonical "*sample where the answer is dense*" trick; the $(\varepsilon, \delta)$/FPRAS framework is the language of approximate counting.
- The **2SAT/3SAT** opener ties the chain's *drift* directly to algorithm running time (fair walk $O(n^2)$ vs. biased $2^n$) — links forward to Schöning's derandomized k-SAT.
- [ ] Be ready to **state the Fundamental Theorem** (all four parts + the two hypotheses each pull their weight) and to **derive $\pi_v = \deg(v)/2|E|$ via detailed balance**.
- [ ] Be ready to **build a Metropolis chain for a given $\pi$** (weighted IS) and explain *why the normalizer $B$ cancels*.
- [ ] Be ready to run the **expander-walk BPP amplification** count of random bits ($n + O(k)$) and say *why correlated walk samples still give a Chernoff-like bound*.