Randomness in Computing

CS 537

Lecture 23

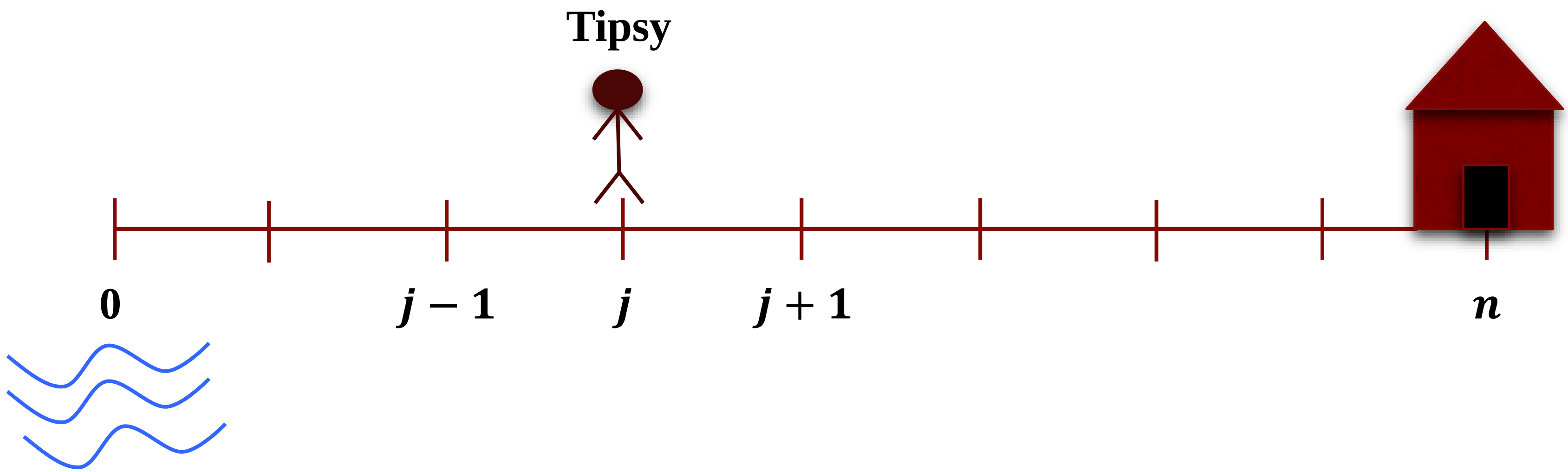
Last time

- Probabilistic method
  - Algorithmic LLL
  - Applications of LLL

Today

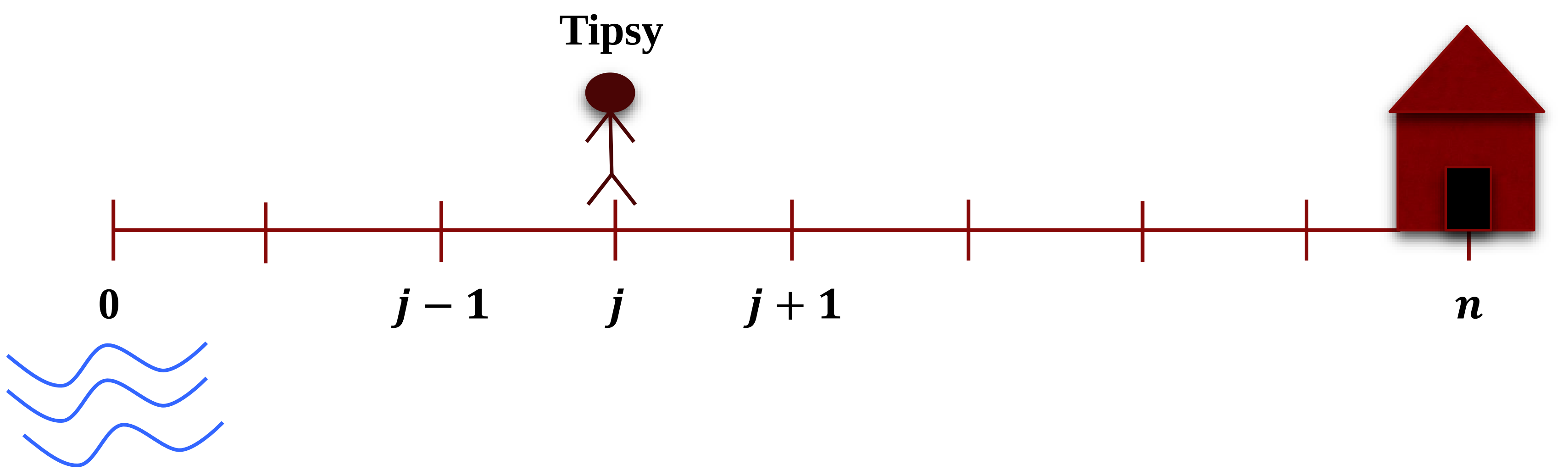
- Drunkard's walk
- Markov chains
- Randomized algorithm for 2SAT

Drunkard's walk problem



$$p_j = \Pr[\text{Tipsy goes home } \mid \text{he started at position } j]$$
$$p_n = 1$$
$$p_0 = 0$$

Drunkard's walk: probability

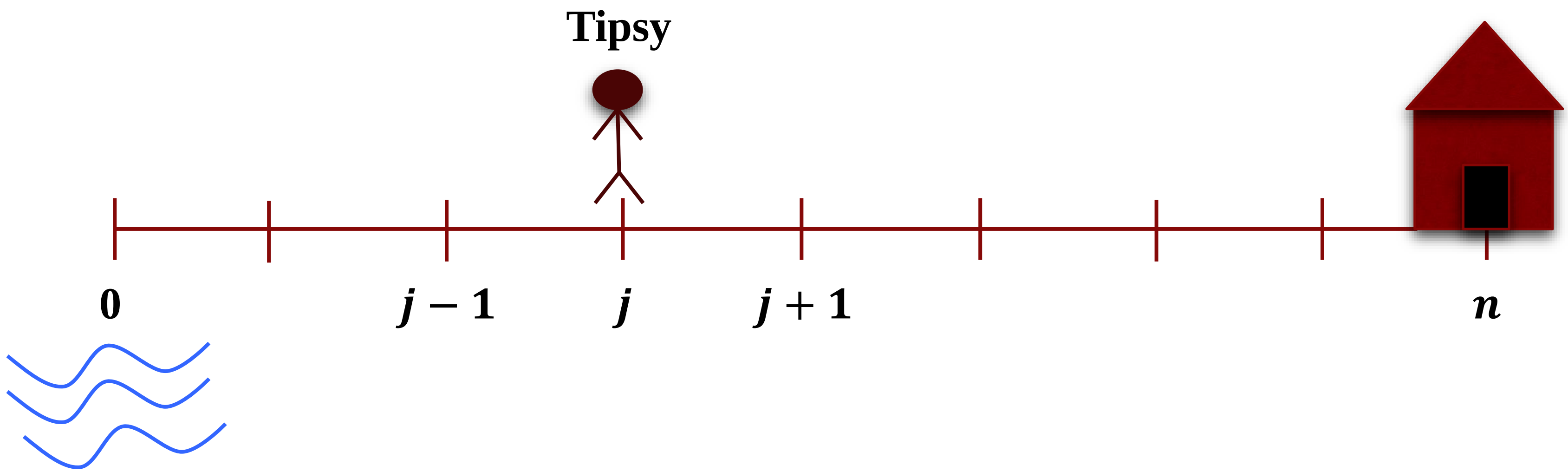


$$p_j = \Pr[\text{Tipsy goes home} \mid \text{he started at position } j]$$

$$p_n = 1$$

$$p_0 = 0$$

Drunkard's walk: probability



$$p_j = \Pr[\text{Tipsy goes home } | \text{ he started at position } j]$$

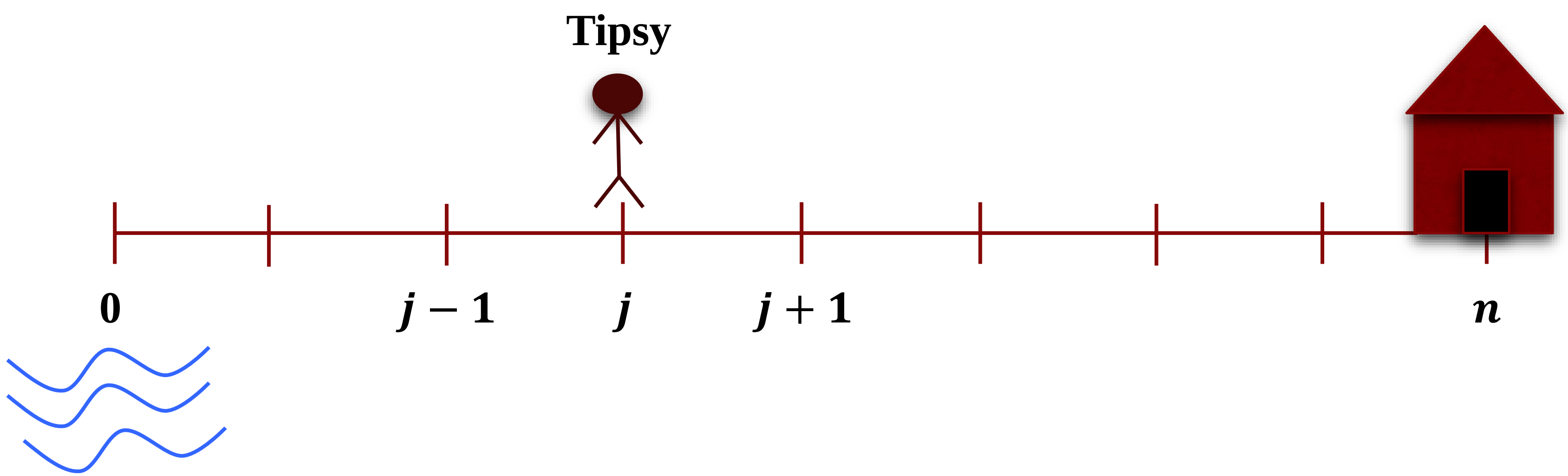
$$p_n = 1$$

$$p_0 = 0$$

$$\text{for all } j \in [1, n-1]: \quad p_j = \frac{p_{j-1}}{2} + \frac{p_{j+1}}{2}$$

$$\boxed{\boldsymbol{p_j = \frac{j}{n}}}$$

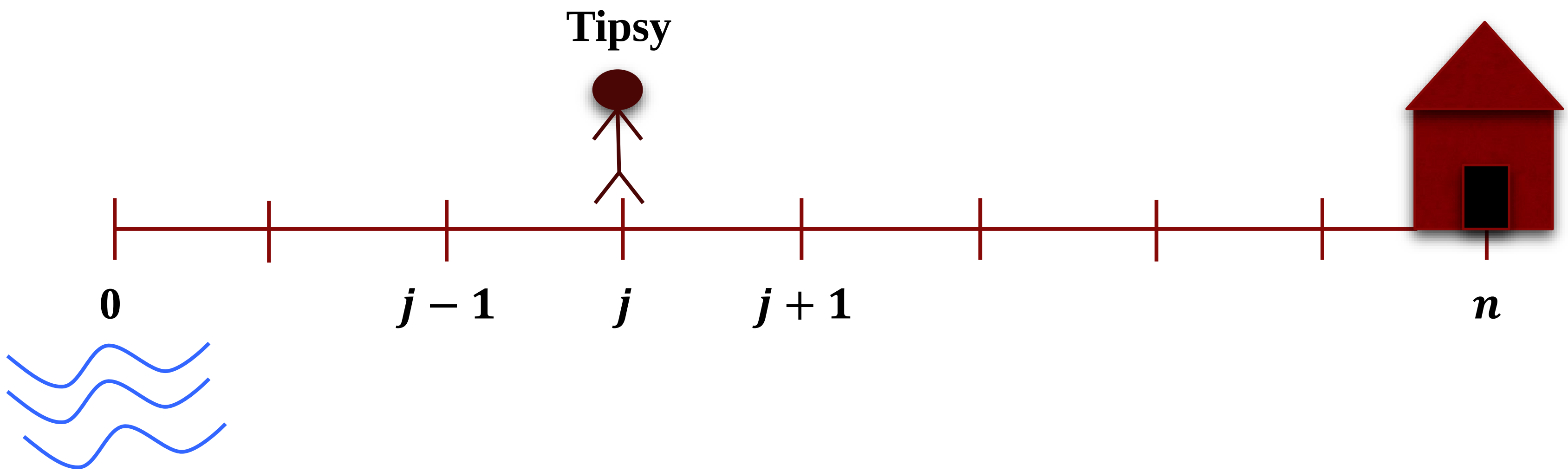
Drunkard's walk: probability



$$\Pr[\text{Tipsy goes home } \mid \text{he started at position } j] = \frac{j}{n}$$

$$\Pr[\text{Tipsy falls into the river } \mid \text{he started at position } j] = \frac{n-j}{n}$$

Drunkard's walk: expected time



$s_j =$ expected number of steps to finish the walk, starting at postion j

$$s_0 = 0$$

$$s_n = 0$$

for $j \in [1, n-1]$: $\quad s_j = 1 + \frac{s_{j-1}}{2} + \frac{s_{j+1}}{2}$

$$\boxed{\boldsymbol{s_j = j(n-j)}}$$

CS 537 Markov Chains

- A (discrete time) stochastic process is a (finite or countably infinite) collection of random variables $X_0, X_1, X_2, \ldots$

represent evolution of some random process over time

- A discrete time stochastic process is a Markov chain if $\forall t \geq 1$ and $\forall$values $a_0, a_1, \ldots, a_t$,

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

$$= \Pr[X_t = a_t | X_{t-1} = a_{t-1}]$$

Markov property or memoryless property

$$= P_{a_{t-1}, a_t}$$

Time-homogeneous property

Terminology

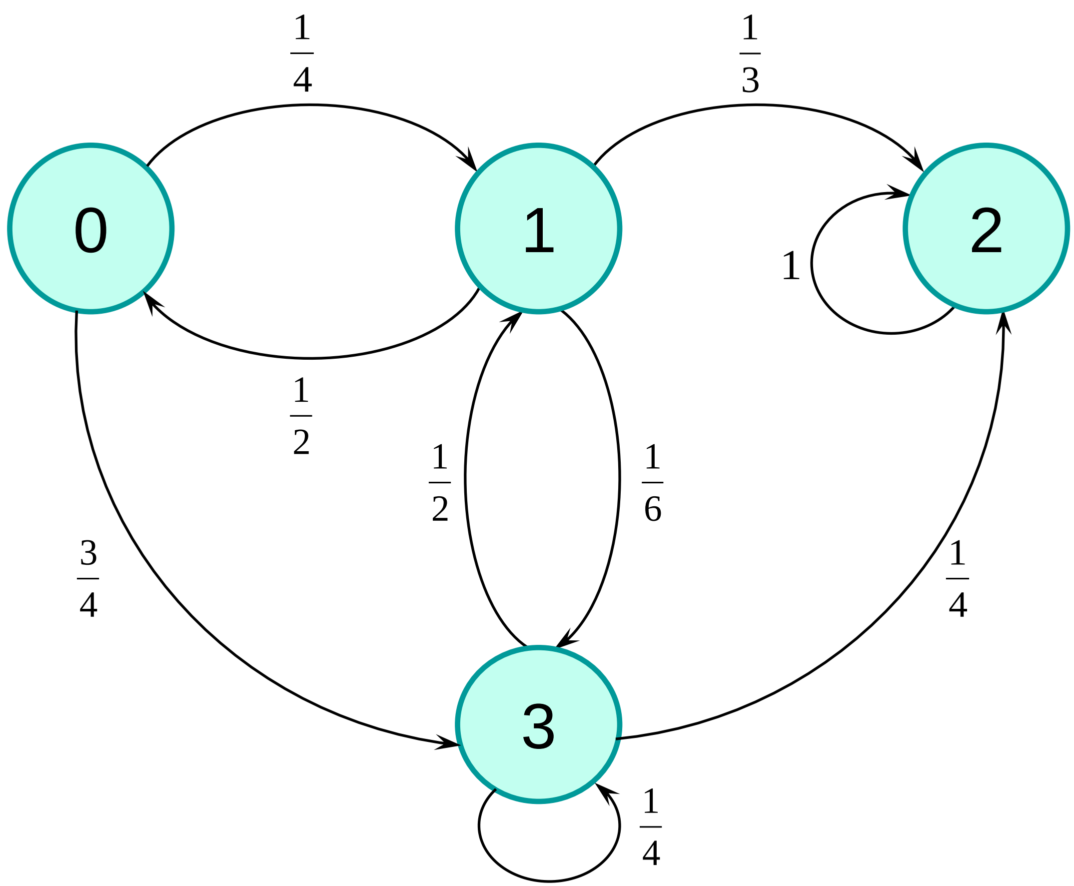
state space	the set of values the RVs can take, e.g. 0,1,2, ...
states visited by the chain	$\boldsymbol{X_0, X_1, \ldots}$
transition probability from $\boldsymbol{a_{t-1}}$ *to* $\boldsymbol{a_t}$	$\boldsymbol{P_{a_{t-1},a_t}}$

Memoryless property:

- X_t depends on X_{t-1}, but not on how the process arrived at state X_{t-1}.
- It does not imply that X_t is independent of $X_0, \ldots, X_{t-2}$ (only that this dependency is captured by X_{t-1})

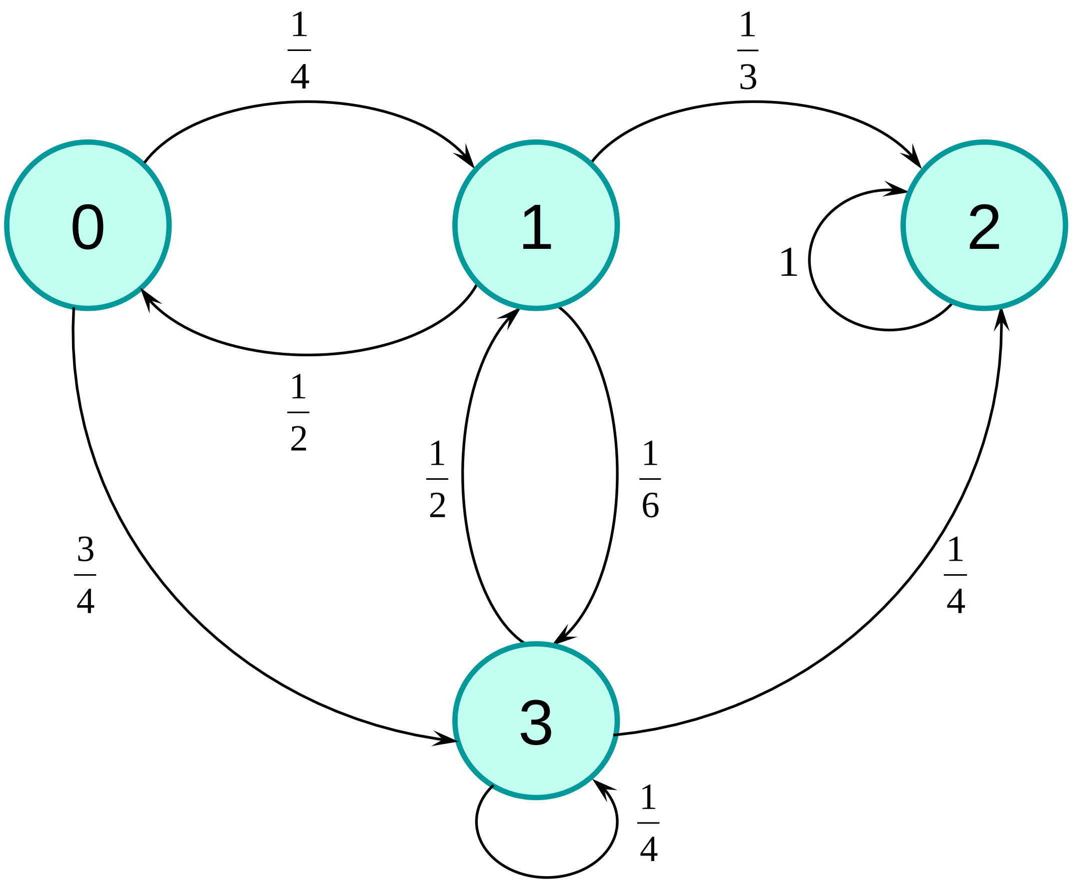
Representation: directed weighted graph

- Set of vertices = state space
- Directed edge (i, j) iff $P_{i,j} > 0$; the edge weight is $P_{i,j}$



Representation: Transition Matrix

- Entry $P_{i,j}$ in matrix $\boldsymbol{P}$ is the transition probability from i to j
- For all rows i, the sum $\sum_{j \geq 0} P_{i,j} = 1$



$$P = \begin{pmatrix} 0 & 1/4 & 0 & 3/4 \\ 1/2 & 0 & 1/3 & 1/6 \\ 0 & 0 & 1 & 0 \\ 0 & 1/2 & 1/4 & 1/4 \end{pmatrix}$$

Distribution of states

- Let $p_j(t)$ be the probability that the process is at state j at time t. By the Law of Total Probability,

$$p_j(t) = \sum_{i \geq 0} p_i(t-1) \cdot P_{i,j}$$

$$\bar{p}(t-1) \cdot (\boldsymbol{j^{th}\ column\ of\ P})$$

- Let $\bar{p}(t) = (p_0(t), p_1(t), \dots)$ be the (row) vector giving the distribution of the chain at time t.

$$\bar{p}(t) = \bar{p}(t-1)\, \boldsymbol{P}$$

- For all $m \geq 0$, we define the m-step transition probability

$$P_{i,j}^m = \Pr[X_{t+m} = j \mid X_t = i]$$

- Conditioning on the first transition from i, by the Law of Total Probability,

$$P_{i,j}^m = \sum_{k \geq 0} P_{i,k} P_{k,j}^{m-1}$$

Distribution of states at time m

$$P_{i,j}^m = \sum_{k \geq 0} P_{i,k} P_{k,j}^{m-1}$$

- Let $\boldsymbol{P}^{(\boldsymbol{m})}$ be the matrix whose entries (i, j) are the $\boldsymbol{m}$-step transitional probabilities $P_{i,j}^m$.

$$\boldsymbol{P}^{(\boldsymbol{m})} = \boldsymbol{P} \cdot \boldsymbol{P}^{(m-1)}$$

By induction on m,

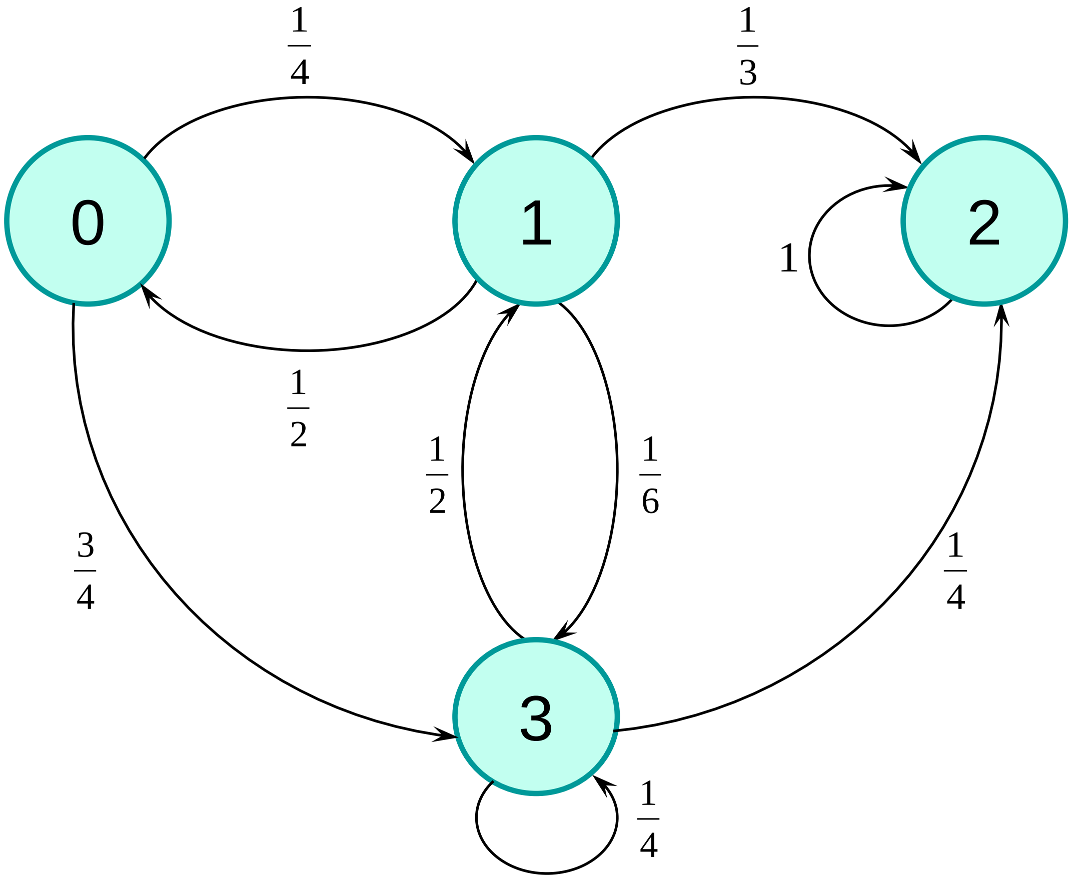
$$\boldsymbol{P}^{(\boldsymbol{m})} = \boldsymbol{P}^{\boldsymbol{m}}$$

- For all $t \geq 0$ and $m \geq 1$,

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

Example

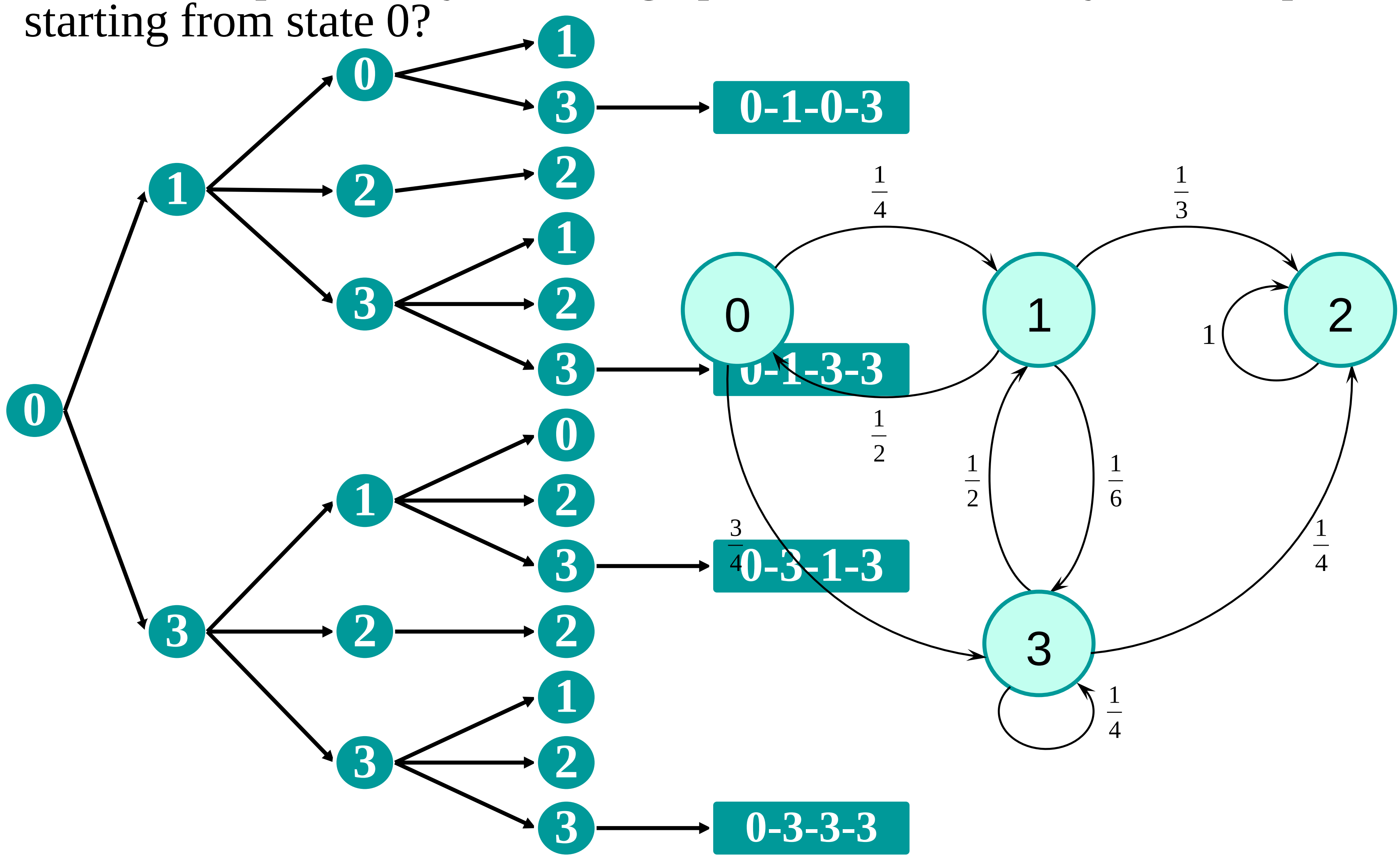
What is the probability of ending up in state 3 in exactly three steps, starting from state 0?



$$P = \begin{pmatrix} 0 & \frac{1}{4} & 0 & \frac{3}{4} \\ \frac{1}{2} & 0 & \frac{1}{3} & \frac{1}{6} \\ 0 & 0 & 1 & 0 \\ 0 & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \end{pmatrix}$$

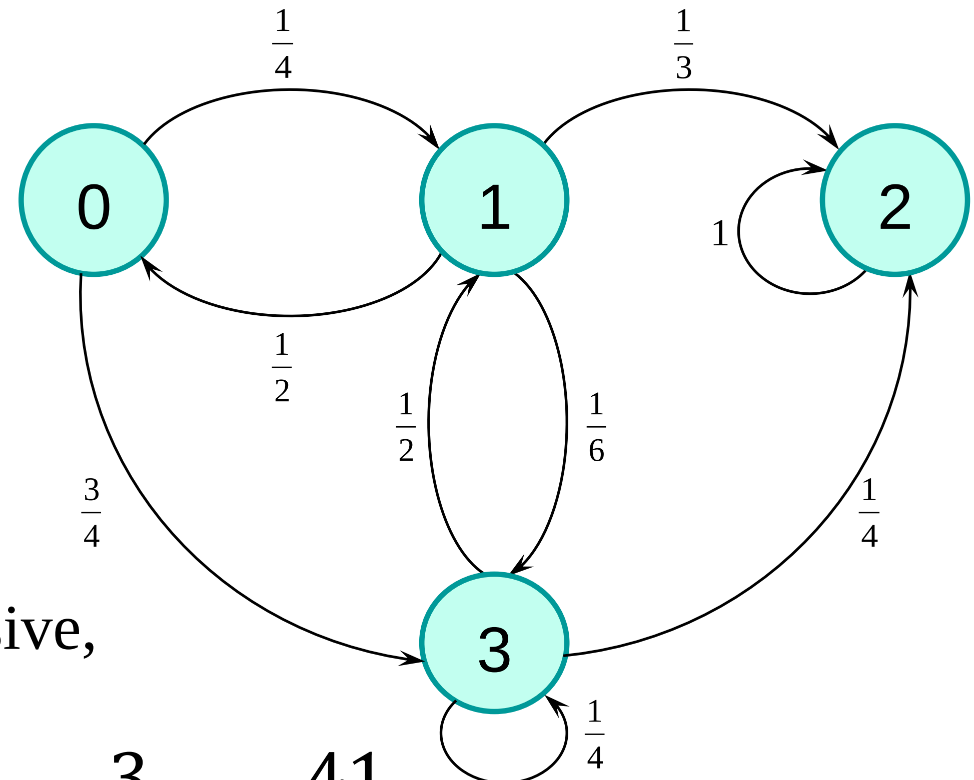
Example

What is the probability of ending up in state 3 in exactly three steps, starting from state 0?



Example

- We calculate the probability of the four events:
  - $0 - 1 - 0 - 3 \mid \Pr = 3/32$
  - $0 - 1 - 3 - 3 \mid \Pr = 1/96$
  - $0 - 3 - 1 - 3 \mid \Pr = 1/16$
  - $0 - 3 - 3 - 3 \mid \Pr = 3/64$



- Since they are mutually exclusive, the total probability is

$$P_{0,3}^{3} = \frac{3}{32} + \frac{1}{96} + \frac{1}{16} + \frac{3}{64} = \frac{41}{192}$$

CS 537

Example

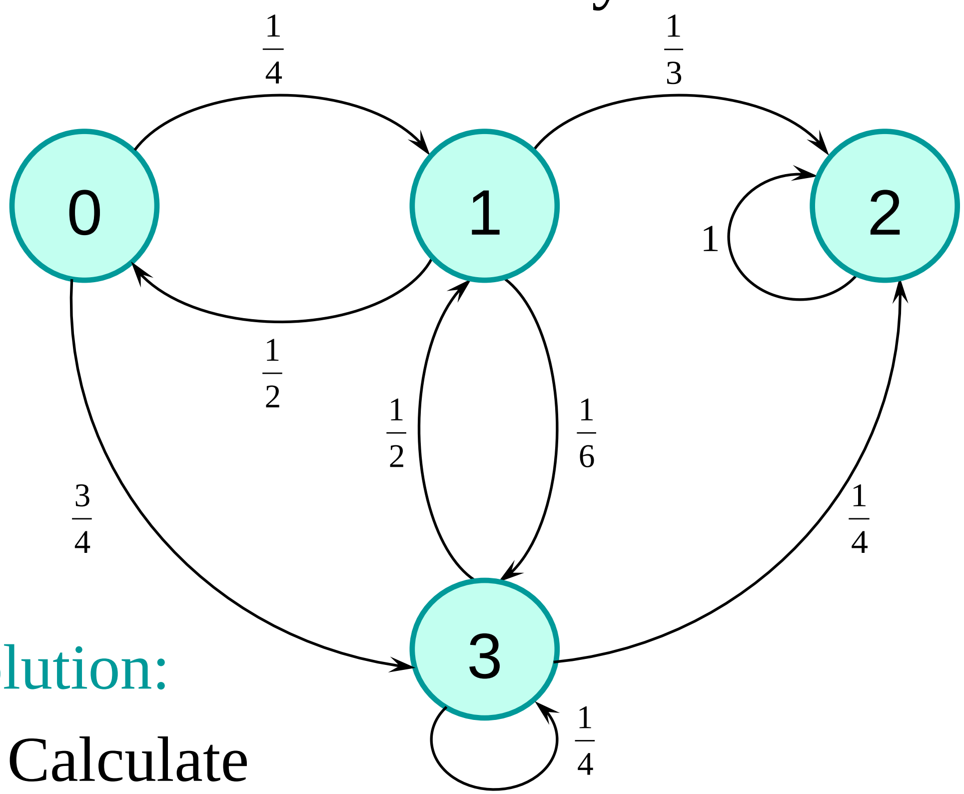
Alternatively, we can calculate $\boldsymbol{P}^3$

and find the entry (0,3)

$$\begin{pmatrix} 3/16 & 7/48 & 29/64 & 41/192 \\ 5/48 & 5/24 & 79/144 & 5/36 \\ 0 & 0 & 1 & 0 \\ 1/16 & 13/96 & 107/192 & 47/192 \end{pmatrix}$$

Example 2

What is the probability of ending up in state 3 after three steps if we start in a uniformly random state?



$$P = \begin{pmatrix} 0 & 1/4 & 0 & 3/4 \\ 1/2 & 0 & 1/3 & 1/6 \\ 0 & 0 & 1 & 0 \\ 0 & 1/2 & 1/4 & 1/4 \end{pmatrix}$$

Solution:

- Calculate

$$\left(\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4}\right)P^3 = \left(\frac{17}{192},\frac{47}{384},\frac{737}{1152},\frac{43}{288}\right)$$

Answer

Application: Algorithm for 2SAT

Recall: A 2CNF formula is an AND of clauses

- Each clause is an OR of literals.
- Each literal is a Boolean variable or its negation.
- E.g. $(x_1 \vee \overline{x_2}) \wedge (x_2 \vee \overline{x_3}) \wedge (x_3 \vee \overline{x_4}) \wedge (x_1 \vee x_4) \wedge (\overline{x_2} \vee \overline{x_4})$

2SAT Problem (search version): Given a 2CNF formula, find a satisfying assignment if it is satisfiable.

CS 537

Randomized Algorithm for 2SAT

Input: a 2CNF formula ϕ on n variables

parameter

1. Start with an arbitrary truth assignment, e.g., all 0's.
2. Repeat R times, terminating if ϕ is satisfied:
   a) Choose an arbitrary clause C that is not satisfied.
   b) Pick a uniformly random literal in C and flip its assignment.
3. If a satisfying assignment is found, return it.
4. Otherwise, return ``unsatisfiable''.

Example: $\phi = (x_1 \vee \overline{x_2}) \wedge (x_2 \vee \overline{x_3}) \wedge (x_3 \vee \overline{x_4}) \wedge (x_1 \vee x_4) \wedge (\overline{x_2} \vee \overline{x_4})$

- Initial assignment: $x_1 = 0, x_2 = 0, x_3 = 0, x_4 = 0$
- Unsatisfied clause: $C = (x_1 \vee x_4)$
- Pick x_1 or x_4 and flip its value: $x_1 = 0, x_2 = 0, x_3 = 0, x_4 = 1$
- New unsatisfied clause: $C = (x_3 \vee \overline{x_4})$
- Pick x_3 or $\overline{x_4}$ and flip its value: $x_1 = 0, x_2 = 0, x_3 = 1, x_4 = 1$

When can the algorithm fail?

- Only if ϕ is satisfiable, but we did not find a satisfying assignment in R iterations (steps).

- We will analyze the number of steps necessary.

- Each step can be implemented to run in $O(n^2)$ time, since there are $O(n^2)$ clauses.

Analysis of the number of steps

- Let $S =$ a satisfying assignment of ϕ.
- $A_i =$ an assignment to ϕ after i steps
- $X_i =$ number of variables that have the same value in A_i and S

When $X_i = n$, the algorithm terminates with a satisfying assignment.

(It could do it before $X_i = n$ if it finds another satisfying assignment.)

- If $X_i = 0$ then $X_{i+1} = 1$

$$\Pr[X_{i+1} = 1 \mid X_i = 0] = 1$$

- If $X_i \in [1, n-1]$ then A_i disagrees with S on 1 or 2 literals of C

1/2 or 1

$$\Pr[X_{i+1} = j + 1 \mid X_i = j] \geq 1/2$$

$$\Pr[X_{i+1} = j - 1 \mid X_i = j] \leq 1/2$$

$X_0, X_1, X_2, \ldots$ is not necessarily a Markov chain, since the probability of $X_{i+1} > X_i$ depends on whether A_i and S disagree on 1 or 2 literals of C (which could depend on previous choices, not just X_i)

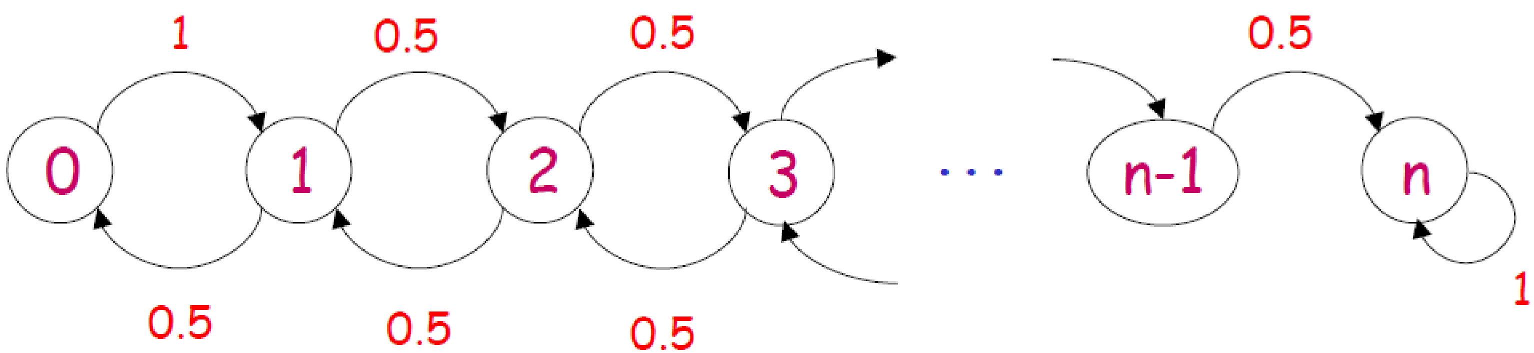
Creating a true Markov chain

- Define a Markov Chain $Y_0, Y_1, Y_2, \ldots$

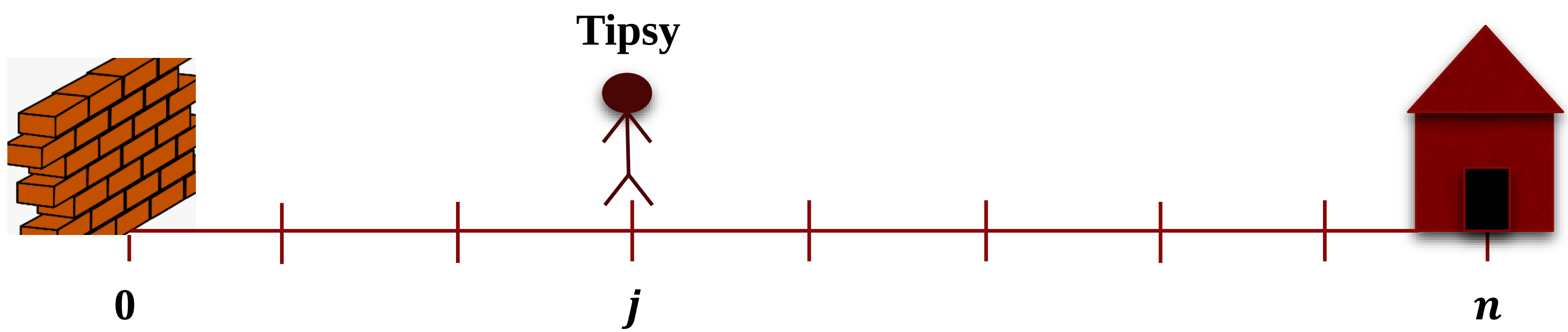
$$Y_0 = X_0$$
$$\Pr[Y_{i+1} = 1 \mid Y_i = 0] = 1$$
$$\Pr[Y_{i+1} = j + 1 \mid Y_i = j] = 1/2$$
$$\Pr[Y_{i+1} = j - 1 \mid Y_i = j] = 1/2$$

- ``Pessimistic version'' of stochastic process $X_0, X_1, X_2, \ldots$

The expected time to reach n is larger for $Y_0, Y_1, Y_2, \ldots$ than for $X_0, X_1, X_2, \ldots$



Expected time to reach n



$$s_j = \text{expected number of steps to reach position } n,$$
$$\text{starting at postion } j$$

$$s_0 = \boldsymbol{s_1 + 1}$$

$$s_n = 0$$

$$\text{for } j \in [1, n-1]: \quad s_j = 1 + \frac{s_{j-1}}{2} + \frac{s_{j+1}}{2}$$

$$\boxed{\boldsymbol{s_j = n^2 - j^2}}$$

$$s_0 = n^2$$

2SAT algorithm: correctness

Theorem

If number of steps $R = 2an^2$ and ϕ is satisfiable, then the algorithm returns a satisfying assignment with probability at least $1 - 2^{-a}$.

Proof:

- The expected number of steps until ALG finds a satisfying assignment is $\leq n^2$, regardless of starting position.
- Brake R into a segments of $2n^2$
- Let $Z =$ # steps ALG takes in segment k without completion.

Application: Algorithm for 3SAT

- First try: the same algorithm as for 2SAT

Input: a 3CNF formula ϕ on n variables

parameter

1. Start with an arbitrary truth assignment, e.g., all 0's.
2. Repeat R times, terminating if ϕ is satisfied:
   a) Choose an arbitrary clause C that is not satisfied.
   b) Pick a uniformly random literal in C and flip its assignment.
3. If a satisfying assignment is found, return it.
4. Otherwise, return ``unsatisfiable''.

- We want to analyze the number of steps (iterations) necessary.

Analysis: What should we change?

- Let $S =$ a satisfying assignment of ϕ.
- $A_i =$ an assignment to ϕ after i steps
- $X_i =$ number of variables that have the same value in A_i and S

When $X_i = n$, the algorithm terminates with a satisfying assignment.

- If $X_i = 0$ then $X_{i+1} = 1$

$$\Pr[X_{i+1} = 1 \mid X_i = 0] = 1$$

- If $X_i \in [1, n-1]$ then A_i disagrees with S on 1 to 3 literals of C

$$\Pr[X_{i+1} = j + 1 \mid X_i = j] \geq 1/3$$
$$\Pr[X_{i+1} = j - 1 \mid X_i = j] \leq 2/3$$

$X_0, X_1, X_2, \ldots$ is not necessarily a Markov chain