Motivation
Example: Simulation of a die by coins

Knuth & Yao die
Example: Simulation of a die by coins

Knuth & Yao die

![Diagram of a Knuth & Yao die simulation]

Question:
- What is the probability of obtaining 2?
Definition:
A discrete-time Markov chain (DTMC) is a tuple \((S, P, \pi_0)\) where

- \(S\) is the set of states,
- \(P : S \times S \rightarrow [0, 1]\) with \(\sum_{s' \in S} P(s, s') = 1\) is the transitions matrix, and
- \(\pi_0 \in [0, 1]^{\vert S \vert}\) with \(\sum_{s \in S} \pi_0(s) = 1\) is the initial distribution.
Example: Craps

Two dice game:

- First: $\sum \in \{7, 11\} \Rightarrow \text{win}, \sum \in \{2, 3, 12\} \Rightarrow \text{lose}$, else $s = \sum$
- Next rolls: $\sum = s \Rightarrow \text{win}, \sum = 7 \Rightarrow \text{lose}$, else iterate
Example: Craps

Two dice game:

- First: $\sum \in \{7, 11\} \Rightarrow \text{win}$, $\sum \in \{2, 3, 12\} \Rightarrow \text{lose}$, else $s = \sum$
- Next rolls: $\sum = s \Rightarrow \text{win}$, $\sum = 7 \Rightarrow \text{lose}$, else iterate
Example: Craps

Two dice game:

- First: $\sum \in \{7, 11\} \Rightarrow \text{win}$, $\sum \in \{2, 3, 12\} \Rightarrow \text{lose}$, else $s = \sum$
- Next rolls: $\sum = s \Rightarrow \text{win}$, $\sum = 7 \Rightarrow \text{lose}$, else iterate
Example: Zero Configuration Networking (Zeroconf)

- Previously: Manual assignment of IP addresses
- Zeroconf: Dynamic configuration of local IPv4 addresses
- Advantage: Simple devices able to communicate automatically

Automatic Private IP Addressing (APIPA) – RFC 3927

- Used when DHCP is configured but unavailable
- Pick randomly an address from 169.254.1.0 – 169.254.254.255
- Find out whether anybody else uses this address (by sending several ARP requests)

Model:

- Randomly pick an address among the \( K \) (65024) addresses.
- With \( m \) hosts in the network, collision probability is \( q = \frac{m}{K} \).
- Send 4 ARP requests.
- In case of collision, the probability of no answer to the ARP request is \( p \) (due to the lossy channel)
Example: Zero Configuration Networking (Zeroconf)

For 100 hosts and $p = 0.001$, the probability of error is $\approx 1.55 \cdot 10^{-15}$. 
What is probabilistic model checking?

- **Probabilistic** specifications, e.g. probability of reaching bad states shall be smaller than 0.01.
- Probabilistic model checking is an automatic verification technique for this purpose.

Why quantities?

- **Randomized** algorithms
- **Faults** e.g. due to the environment, lossy channels
- **Performance** analysis, e.g. reliability, availability
Basics of Probability Theory
(Recap)
What are probabilities? - Intuition

Throwing a fair coin:

- The outcome **head** has a probability of 0.5.
- The outcome **tail** has a probability of 0.5.
What are probabilities? - Intuition

Throwing a fair coin:

- The outcome **head** has a probability of 0.5.
- The outcome **tail** has a probability of 0.5.

**But … [Bertrand’s Paradox]**

Draw a random chord on the unit circle. What is the probability that its length exceeds the length of a side of the equilateral triangle in the circle?

![Diagram](image.png)
What are probabilities? - Intuition

Throwing a fair coin:

- The outcome **head** has a probability of 0.5.
- The outcome **tail** has a probability of 0.5.

But ... [Bertrand’s Paradox]

Draw a random chord on the unit circle. What is the probability that its length exceeds the length of a side of the equilateral triangle in the circle?
What are probabilities? - Intuition

Throwing a fair coin:

- The outcome head has a probability of 0.5.
- The outcome tail has a probability of 0.5.

But … [Bertrand’s Paradox]

Draw a random chord on the unit circle. What is the probability that its length exceeds the length of a side of the equilateral triangle in the circle?
**Probability Theory - Probability Space**

**Definition: Probability Function**
Given sample space $\Omega$ and $\sigma$-algebra $\mathcal{F}$, a probability function $P : \mathcal{F} \to [0, 1]$ satisfies:

- $P(A) \geq 0$ for $A \in \mathcal{F}$,
- $P(\Omega) = 1$, and
- $P(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i)$ for pairwise disjoint $A_i \in \mathcal{F}$

**Definition: Probability Space**

A probability space is a tuple $(\Omega, \mathcal{F}, P)$ with a sample space $\Omega$, $\sigma$-algebra $\mathcal{F} \subseteq 2^\Omega$ and probability function $P$.

**Example**

A random real number taken uniformly from the interval $[0, 1]$.
- Sample space: $\Omega = [0, 1]$. 
Definition: Probability Function

Given sample space $\Omega$ and $\sigma$-algebra $\mathcal{F}$, a probability function $P : \mathcal{F} \rightarrow [0, 1]$ satisfies:

- $P(A) \geq 0$ for $A \in \mathcal{F}$,
- $P(\Omega) = 1$, and
- $P(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i)$ for pairwise disjoint $A_i \in \mathcal{F}$

Definition: Probability Space

A probability space is a tuple $(\Omega, \mathcal{F}, P)$ with a sample space $\Omega$, $\sigma$-algebra $\mathcal{F} \subseteq 2^\Omega$ and probability function $P$.

Example

A random real number taken uniformly from the interval $[0, 1]$.

- Sample space: $\Omega = [0, 1]$.
- $\sigma$-algebra: $\mathcal{F}$ is the minimal superset of $\{[a, b] \mid 0 \leq a \leq b \leq 1\}$ closed under complementation and countable union.
- Probability function: $P([a, b]) = (b - a)$, by Carathéodory’s extension theorem there is a unique way how to extend it to all elements of $\mathcal{F}$. 


Random Variables

```c
int getRandNumber()
{
    return 4;  // chosen by fair dice roll.
    // guaranteed to be random.
}
```
**Definition: Random Variable**

A random variable $X$ is a measurable function $X : \Omega \rightarrow I$ to some $I$. Elements of $I$ are called random elements. Often $I = \mathbb{R}$:

![Diagram showing a random variable $X$ mapping from a set $\Omega$ to the real numbers $\mathbb{R}$]

**Example (Bernoulli Trials)**

Throwing a coin 3 times: $\Omega_3 = \{hhh, hht, hth, htt, thh, tht, tth, ttt\}$. We define 3 random variables $X_i : \Omega \rightarrow \{h, t\}$. For all $x, y, z \in \{h, t\}$,

- $X_1(xyz) = x$,
- $X_2(xyz) = y$,
- $X_3(xyz) = z$. 
Stochastic Processes and Markov Chains
Definition:
Given a probability space \((\Omega, \mathcal{F}, P)\), a **stochastic process** is a family of random variables
\[
\{X_t \mid t \in T\}
\]
defined on \((\Omega, \mathcal{F}, P)\). For each \(X_t\) we assume
\[
X_t : \Omega \to S
\]
where \(S = \{s_1, s_2, \ldots\}\) is a finite or countable set called **state space**.

A stochastic process \(\{X_t \mid t \in T\}\) is called
- **discrete-time** if \(T = \mathbb{N}\) or
- **continuous-time** if \(T = \mathbb{R}_{\geq 0}\).

For the following lectures we focus on discrete time.
Example: Weather Forecast

- $S = \{\text{sun}, \text{rain}\}$,
- we model time as discrete – a random variable for each day:
  - $X_0$ is the weather today,
  - $X_i$ is the weather in $i$ days.
- how can we set up the probability space to measure e.g. $P(X_i = \text{sun})$?
Let us fix a state space $S$. How can we construct the probability space $(\Omega, \mathcal{F}, P)$?

**Definition: Sample Space $\Omega$**

We define $\Omega = S^\infty$. Then, each $X_n$ maps a sample $\omega = \omega_0 \omega_1 \ldots$ onto the respective state at time $n$, i.e.,

$$(X_n)(\omega) = \omega_n \in S.$$
**Definition: Cylinder Set**

For \( s_0 \ldots s_n \in S^{n+1} \), we set the **cylinder** \( C(s_0 \ldots s_n) = \{s_0 \ldots s_n \omega \in \Omega \} \).

**Example:**

\( S = \{s_1, s_2, s_3\} \) and \( C(s_1s_3) \)

**Definition: \( \sigma \)-algebra \( \mathcal{F} \)**

We define \( \mathcal{F} \) to be the **smallest** \( \sigma \)-Algebra that contains all cylinder sets, i.e.,

\[
\{C(s_0 \ldots s_n) \mid n \in \mathbb{N}, s_i \in S\} \subseteq \mathcal{F}.
\]

**Check:** Is each \( X_i \) measurable?

(on the discrete set \( S \) we assume the full \( \sigma \)-algebra \( 2^S \)).
How to specify the probability Function $P$?
We only needs to specify for each $s_0 \cdots s_n \in S^n$

$$P(C(s_0 \cdots s_n)).$$

This amounts to specifying

1. $P(C(s_0))$ for each $s_0 \in S$, and
2. $P(C(s_0 \cdots s_i) \mid C(s_0 \cdots s_{i-1}))$ for each $s_0 \cdots s_i \in S^i$

since

$$P(C(s_0 \cdots s_n)) = P(C(s_0 \cdots s_n) \mid C(s_0 \cdots s_{n-1})) \cdot P(C(s_0 \cdots s_{n-1}))$$

$$= P(C(s_0)) \cdot \prod_{i=1}^{n} P(C(s_0 \cdots s_i) \mid C(s_0 \cdots s_{i-1}))$$
How to specify the probability Function $P$?

We only need to specify for each $s_0 \cdots s_n \in S^n$

$$P(C(s_0 \cdots s_n)).$$

This amounts to specifying

1. $P(C(s_0))$ for each $s_0 \in S$, and
2. $P(C(s_0 \cdots s_i) \mid C(s_0 \cdots s_{i-1}))$ for each $s_0 \cdots s_i \in S^i$

since

$$P(C(s_0 \cdots s_n)) = P(C(s_0 \cdots s_n) \mid C(s_0 \cdots s_{n-1})) \cdot P(C(s_0 \cdots s_{n-1}))$$

$$= P(C(s_0)) \cdot \prod_{i=1}^{n} P(C(s_0 \cdots s_i) \mid C(s_0 \cdots s_{i-1}))$$

Still, lots of possibilities...
Weather Example: Option 1 - statistics of days of a year

- the forecast starts on Jan 01,
- a distribution $p_j$ over \{sun, rain\} for each $1 \leq j \leq 365$,
- for each $i \in \mathbb{N}$ and $s_0 \cdots s_i \in S^{i+1}$

$$P(C(s_0 \cdots s_i) \mid C(s_0 \cdots s_{i-1})) = p_i \% 365(s_i)$$

Weather Example: Option 2 - two past days

- a distribution $p_{s's''}$ over \{sun, rain\} for each $s', s'' \in S$,
- for each $i \geq 2$ and $s_0 \cdots s_i \in S^{i+1}$

$$P(C(s_0 \cdots s_i) \mid C(s_0 \cdots s_{i-1})) = p_{s_{i-2}s_{i-1}}(s_i)$$
Weather Example: Option 1 - statistics of days of a year

- the forecast starts on Jan 01,
- a distribution \( p_j \) over \{sun, rain\} for each \( 1 \leq j \leq 365 \),
- for each \( i \in \mathbb{N} \) and \( s_0 \ldots s_i \in S^{i+1} \)

\[
P(C(s_0 \ldots s_i) \mid C(s_0 \ldots s_{i-1})) = p_i \mod 365(s_i)
\]

Weather Example: Option 2 - two past days

- a distribution \( p_{s's''} \) over \{sun, rain\} for each \( s', s'' \in S \),
- for each \( i \geq 2 \) and \( s_0 \ldots s_i \in S^{i+1} \)

\[
P(C(s_0 \ldots s_i) \mid C(s_0 \ldots s_{i-1})) = p_{s_{i-2}s_{i-1}}(s_i)
\]

Here: time-homogeneous Markovian stochastic processes

Not time-homogeneous.

Not Markovian.
Definition: Markov
A discrete-time stochastic process \( \{X_n \mid n \in \mathbb{N}\} \) is Markov if
\[
P(X_n = s_n \mid X_{n-1} = s_{n-1}, \ldots, X_0 = s_0) = P(X_n = s_n \mid X_{n-1} = s_{n-1})
\]
for all \( n > 1 \) and \( s_0, \ldots, s_n \in S \) with \( P(X_{n-1} = s_{n-1}) > 0 \).

Definition: Time-homogeneous
A discrete-time Markov process \( \{X_n \mid n \in \mathbb{N}\} \) is time-homogeneous if
\[
P(X_{n+1} = s' \mid X_n = s) = P(X_1 = s' \mid X_0 = s)
\]
for all \( n > 1 \) and \( s, s' \in S \) with \( P(X_0 = s) > 0 \).
Stochastic Processes - Restrictions

**Definition: Markov**
A discrete-time stochastic process \( \{X_n \mid n \in \mathbb{N}\} \) is Markov if

\[
P(X_n = s_n \mid X_{n-1} = s_{n-1}, \ldots, X_0 = s_0) = P(X_n = s_n \mid X_{n-1} = s_{n-1})
\]

for all \( n > 1 \) and \( s_0, \ldots, s_n \in S \) with \( P(X_{n-1} = s_{n-1}) > 0 \).

**Definition: Time-homogeneous**
A discrete-time Markov process \( \{X_n \mid n \in \mathbb{N}\} \) is time-homogeneous if

\[
P(X_{n+1} = s' \mid X_n = s) = P(X_1 = s' \mid X_0 = s)
\]

for all \( n > 1 \) and \( s, s' \in S \) with \( P(X_0 = s) > 0 \).
Weather Example: Option 3 - one past day

- a distribution $p_{s'}$ over \{sun, rain\} for each $s' \in S$,
- for each $i \geq 1$ and $s_0 \ldots s_i \in S^{i+1}$
  \[
P(C(s_0 \ldots s_i) \mid C(s_0 \ldots s_{i-1})) = p_{s_{i-1}}(s_i)
  \]
- a distribution $\pi$ over \{sun, rain\} such that $P(C(s_0)) = \pi(s_0)$. 
Weather Example: Option 3 - one past day

- a distribution $p_{s'}$ over \{\textit{sun, rain}\} for each $s' \in S$,
- for each $i \geq 1$ and $s_0 \cdots s_i \in S^{i+1}$

$$P(C(s_0 \cdots s_i) \mid C(s_0 \cdots s_{i-1})) = p_{s_{i-1}}(s_i)$$

- a distribution $\pi$ over \{\textit{sun, rain}\} such that $P(C(s_0)) = \pi(s_0)$.

Overly restrictive, isn’t it?
Weather Example: Option 3 - one past day

- a distribution $p_{s'}$ over \{\textit{sun}, \textit{rain}\} for each $s' \in S$,
- for each $i \geq 1$ and $s_0 \cdots s_i \in S^{i+1}$
  \[
P(C(s_0 \cdots s_i) \mid C(s_0 \cdots s_{i-1})) = p_{s_{i-1}}(s_i)
\]
- a distribution $\pi$ over \{\textit{sun}, \textit{rain}\} such that $P(C(s_0)) = \pi(s_0)$.

Overly restrictive, isn’t it?

Not really – one only needs to extend the state space

- $S = \{1, \ldots, 365\} \times \{\textit{sun}, \textit{rain}\} \times \{\textit{sun}, \textit{rain}\}$,
- now each state encodes current day of the year, current weather, and weather yesterday,
- we can define over $S$ a time–homogeneous Markov process based on both Options 1 & 2 given earlier.
Discrete-time Markov Chains
DTMC
Stochastic process → Graph based
Given a discrete-time homogeneous Markov process \( \{ X(n) \mid n \in \mathbb{N} \} \)
- with state space \( S \),
- defined on a probability space \( (\Omega, \mathcal{F}, P) \)

we take over the state space \( S \) and define
- \( P(s, s') = P(X_n = s' \mid X_{n-1} = s) \) for an arbitrary \( n \in \mathbb{N} \) and
- \( \pi_0(s) = P(X_0 = s) \).

Graph based → stochastic process
Given a DTMC \((S, P, \pi_0)\), we set \( \Omega \) to \( S^\infty \), \( \mathcal{F} \) to the smallest \( \sigma \)-Algebra containing all cylinder sets and

\[
P(C(s_0 \ldots s_n)) = \pi_0(s_0) \cdot \prod_{1 \leq i \leq n} P(s_{i-1}, s_i)
\]

which uniquely defines the probability function \( P \) on \( \mathcal{F} \).
Let \((S, P, \pi_0)\) be a DTMC. We denote by

- \(P_s\) the probability function of DTMC \((S, P, \delta_s)\) where

\[
\delta_s(s') = \begin{cases} 
1 & \text{if } s' = s \\
0 & \text{otherwise}
\end{cases}
\]

- \(E_s\) the expectation with respect to \(P_s\)
Analysis questions

- Transient analysis
- Steady-state analysis
- Rewards
- Reachability
- Probabilistic logics
DTMC - Transient Analysis
Example: Gambling with a Limit

What is the probability of being in state 0 after 3 steps?
Definition:
Given a DTMC \((S, P, \pi_0)\), we assume w.l.o.g. \(S = \{0, 1, \ldots\}\) and write \(p_{ij} = P(i, j)\). Further, we have

- \(P^{(1)} = P = (p_{ij})\) is the 1-step transition matrix
- \(P^{(n)} = (p_{ij}^{(n)})\) denotes the \(n\)-step transition matrix with

\[
p_{ij}^{(n)} = P(X_n = j \mid X_0 = i) \quad (= P(X_{k+n} = j \mid X_k = i)).
\]

How can we compute these probabilities?
Definition: Chapman-Kolmogorov Equation

Application of the law of total probability to the $n$-step transition probabilities $p_{ij}^{(n)}$ results in the Chapman-Kolmogorov Equation

$$p_{ij}^{(n)} = \sum_{h \in S} p_{ih}^{(m)} p_{hj}^{(n-m)} \quad \forall 0 < m < n.$$ 

Consequently, we have $P^{(n)} = PP^{(n-1)} = \ldots = P^n$. 

**Definition: Chapman-Kolmogorov Equation**

Application of the law of total probability to the $n$-step transition probabilities $p_{ij}^{(n)}$ results in the Chapman-Kolmogorov Equation

$$p_{ij}^{(n)} = \sum_{h \in S} p_{ih}^{(m)} p_{hj}^{(n-m)} \quad \forall 0 < m < n.$$  

Consequently, we have $P^{(n)} = PP^{(n-1)} = \ldots = P^n$.

**Definition: Transient Probability Distribution**

The transient probability distribution at time $n > 0$ is defined by

$$\pi_n = \pi_0 P^n = \pi_{n-1} P.$$
Example:

\[
P = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0.5 & 0 & 0.5 & 0 & 0 \\
0 & 0.5 & 0 & 0.5 & 0 \\
0 & 0 & 0.5 & 0 & 0.5 \\
0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
P^2 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0.5 & 0.25 & 0 & 0.25 & 0 \\
0.25 & 0 & 0.5 & 0 & 0.25 \\
0 & 0.25 & 0 & 0.25 & 0.5 \\
0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

For \( \pi_0 = [0, 0, 1, 0, 0] \), \( \pi_2 = \pi_0 P^2 = [0.25, 0, 0.5, 0, 0.25] \).

For, \( \pi_0 = [0.4, 0, 0, 0, 0.6] \), \( \pi_2 = \pi_0 P^2 = [0.4, 0, 0, 0, 0.6] \).

Actually, \( \pi_n = [0.4, 0, 0, 0, 0.6] \) for all \( n \in \mathbb{N} \)!
Example:

\[
P = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0.5 & 0 & 0.5 & 0 & 0 \\
0 & 0.5 & 0 & 0.5 & 0 \\
0 & 0 & 0.5 & 0 & 0.5 \\
0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
P^2 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0.5 & 0.25 & 0 & 0.25 & 0 \\
0.25 & 0 & 0.5 & 0 & 0.25 \\
0 & 0.25 & 0 & 0.25 & 0.5 \\
0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

- For \( \pi_0 = [0 \ 0 \ 1 \ 0 \ 0] \), \( \pi_2 = \pi_0 P^2 = [0.25 \ 0 \ 0.5 \ 0 \ 0.25] \).
- For \( \pi_0 = [0.4 \ 0 \ 0 \ 0 \ 0.6] \), \( \pi_2 = \pi_0 P^2 = [0.4 \ 0 \ 0 \ 0 \ 0.6] \).

Actually, \( \pi_n = [0.4 \ 0 \ 0 \ 0 \ 0.6] \) for all \( n \in \mathbb{N} \!\).
DTMC - Steady State Analysis
Definition: Stationary Distribution

A distribution \( \pi \) is **stationary** if

\[
\pi = \pi P.
\]

Stationary distribution is generally **not unique**.
Definition: Stationary Distribution
A distribution $\pi$ is stationary if

$$\pi = \pi P.$$

Stationary distribution is generally not unique.

Definition: Limiting Distribution

$$\pi^* := \lim_{n \to \infty} \pi_n = \lim_{n \to \infty} \pi_0 P^n = \pi_0 \lim_{n \to \infty} P^n = \pi_0 P^*.$$ 

The limit can depend on $\pi_0$ and does not need to exist.
Definition: Stationary Distribution
A distribution $\pi$ is stationary if

$$\pi = \pi P.$$ 

Stationary distribution is generally not unique.

Definition: Limiting Distribution

$$\pi^* := \lim_{n \to \infty} \pi_n = \lim_{n \to \infty} \pi_0 P^n = \pi_0 \lim_{n \to \infty} P^n = \pi_0 P^*.$$ 

The limit can depend on $\pi_0$ and does not need to exist.

Connection between stationary and limiting?
Example: Gambling with Social Guarantees

What are the stationary and limiting distributions?
Example: Gambling with Social Guarantees

What are the stationary and limiting distributions?

**Definition: Periodicity**

The *period* of a state $i$ is defined as

$$d_i = \gcd\{n \mid p^n_{ii} > 0\}.$$ 

A state $i$ is called *aperiodic* if $d_i = 1$ and *periodic* with period $d_i$ otherwise. A Markov chain is *aperiodic* if all states are aperiodic.

**Lemma**

In a finite *aperiodic* Markov chain, the limiting distribution exists.
Example

![Diagram showing a DTMC example](image-url)
Definition:
A DTMC is called **irreducible** if for all states $i, j \in S$ we have $p_{ij}^n > 0$ for some $n \geq 1$.

Lemma

*In an aperiodic and irreducible Markov chain, the limiting distribution exists and does not depend on $\pi_0$.***
What is the stationary / limiting distribution?
What is the stationary / limiting distribution?

Lemma

In a finite aperiodic and irreducible Markov chain, the limiting distribution exists, does not depend on $\pi_0$, and equals the unique stationary distribution.
What is the stationary / limiting distribution?

Lemma

In a finite aperiodic and irreducible Markov chain, the limiting distribution exists, does not depend on $\pi_0$, and equals the unique stationary distribution.
Definition:
Let $f_{ij}^{(n)} = P(X_n = j \land \forall 1 \leq k < n : X_k \neq j \mid X_0 = i)$ for $n \geq 1$ be the $n$-step hitting probability. The hitting probability is defined as

$$f_{ij} = \sum_{n=1}^{\infty} f_{ij}^{(n)}$$

and a state $i$ is called

- transient if $f_{ii} < 1$ and
- recurrent if $f_{ii} = 1$. 
Definition:
Denoting expectation \( m_{ij} = \sum_{n=1}^{\infty} n \cdot f_{ij}^{(n)} \), a recurrent state \( i \) is called

- positive recurrent or recurrent non-null if \( m_{ii} < \infty \) and
- recurrent null if \( m_{ii} = \infty \).

Lemma
The states of an irreducible DTMC are all of the same type, i.e.,

- all periodic or
- all aperiodic and transient or
- all aperiodic and recurrent null or
- all aperiodic and recurrent non-null.
Definition: Ergodicity
A DTMC is **ergodic** if all its states are **irreducible**, **aperiodic** and recurrent non-null.

Theorem
*In an ergodic Markov chain, the limiting distribution exists, does not depend on $\pi_0$, and equals the unique stationary distribution.*

As a consequence, the steady-state distribution can be computed by solving the equation system

$$\pi = \pi P, \sum_{x \in S} \pi_s = 1.$$ 

Note: The Lemma for finite DTMC follows from the theorem as every irreducible finite DTMC is positive recurrent.
Example: Unbounded Gambling with House Edge

The DTMC is only \textit{ergodic} for \( p \in [0, 0.5) \).
DTMC – Rewards
Definition
A reward Markov chain is a tuple \((S, P, \pi_0, r)\) where \((S, P, \pi_0)\) is a Markov chain and \(r : S \rightarrow \mathbb{Z}\) is a reward function.
Definition
A reward Markov chain is a tuple \((S, P, \pi_0, r)\) where \((S, P, \pi_0)\) is a Markov chain and \(r : S \rightarrow \mathbb{Z}\) is a reward function.

Every run \(\rho = s_0, s_1, \ldots\) induces a sequence of values \(r(s_0), r(s_1), \ldots\)

Value of the whole run can be defined as
Definition
A reward Markov chain is a tuple \((S, P, \pi_0, r)\) where \((S, P, \pi_0)\) is a Markov chain and \(r : S \rightarrow \mathbb{Z}\) is a reward function.

Every run \(\rho = s_0, s_1, \ldots\) induces a sequence of values \(r(s_0), r(s_1), \ldots\)

Value of the whole run can be defined as

total reward
\[
\sum_{i=1}^{T} r(s_i)
\]
Definition
A reward Markov chain is a tuple \((S, P, \pi_0, r)\) where \((S, P, \pi_0)\) is a Markov chain and \(r : S \to \mathbb{Z}\) is a reward function.

Every run \(\rho = s_0, s_1, \ldots\) induces a sequence of values \(r(s_0), r(s_1), \ldots\)

Value of the whole run can be defined as

- total reward
  \[\sum_{i=1}^{T} r(s_i)\]
  But what if \(T = \infty\)?
A reward Markov chain is a tuple \((S, P, \pi_0, r)\) where \((S, P, \pi_0)\) is a Markov chain and \(r : S \to \mathbb{Z}\) is a reward function.

Every run \(\rho = s_0, s_1, \ldots\) induces a sequence of values \(r(s_0), r(s_1), \ldots\)

Value of the whole run can be defined as

- total reward: \(\sum_{i=1}^{T} r(s_i)\)
- discounted reward: \(\sum_{i=1}^{\infty} \lambda^i r(s_i)\) for some \(0 < \lambda < 1\)

But what if \(T = \infty\)?

Definition

The expected average reward is \(\text{EAR} := \lim_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n} E[r(X_i)]\)
Definition

A **reward Markov chain** is a tuple \((S, P, \pi_0, r)\) where \((S, P, \pi_0)\) is a Markov chain and \(r : S \rightarrow \mathbb{Z}\) is a **reward function**.

Every run \(\rho = s_0, s_1, \ldots\) induces a sequence of values \(r(s_0), r(s_1), \ldots\)

Value of the whole run can be defined as
- **total reward**
  \[\sum_{i=1}^{T} r(s_i)\]
  But what if \(T = \infty\)?
- **discounted reward**
  \[\sum_{i=1}^{\infty} \lambda^i r(s_i)\]
  for some \(0 < \lambda < 1\)
- **average reward**
  \[\lim_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n} r(s_i)\]
  also called long-run average or mean payoff
Definition
A reward Markov chain is a tuple \((S, P, \pi_0, r)\) where \((S, P, \pi_0)\) is a Markov chain and \(r : S \rightarrow \mathbb{Z}\) is a reward function.

Every run \(\rho = s_0, s_1, \ldots\) induces a sequence of values \(r(s_0), r(s_1), \ldots\)

Value of the whole run can be defined as
- total reward
  \[\sum_{i=1}^{T} r(s_i)\]
  But what if \(T = \infty\)?
- discounted reward
  \[\sum_{i=1}^{\infty} \lambda^i r(s_i)\] for some \(0 < \lambda < 1\)
- average reward
  \[\lim_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n} r(s_i)\] also called long-run average or mean payoff

Definition
The expected average reward is

\[EAR := \lim_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n} \mathbb{E}[r(X_i)]\]
Definition: Time-average Distribution

\[ \hat{\pi} = \lim_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n} \pi_i. \]

\[ \hat{\pi}(s) \] expresses the ratio of time spent in \( s \) on the long run.

\[ ^1 \text{More details later for Markov decision processes.} \]
Definition: Time-average Distribution

\[ \hat{\pi} = \lim_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n} \pi_i. \]

\( \hat{\pi}(s) \) expresses the ratio of time spent in \( s \) on the long run.

Lemma

1. \( \mathbb{E}[r(X_i)] = \sum_{s \in S} \pi_i(s) \cdot r(s). \)
2. If \( \hat{\pi} \) exists then \( EAR = \sum_{s \in S} \hat{\pi}(s) \cdot r(s). \)
3. If limiting distribution exists, it coincides with \( \hat{\pi}. \)

\(^1\)More details later for Markov decision processes.
Definition: Time-average Distribution

\[ \hat{\pi} = \lim_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n} \pi_i. \]

\( \hat{\pi}(s) \) expresses the ratio of time spent in \( s \) on the long run.

Lemma

1. \( \mathbb{E}[r(X_i)] = \sum_{s \in S} \pi_i(s) \cdot r(s). \)
2. If \( \hat{\pi} \) exists then \( EAR = \sum_{s \in S} \hat{\pi}(s) \cdot r(s). \)
3. If limiting distribution exists, it coincides with \( \hat{\pi} \).

Algorithm

1. Compute \( \hat{\pi} \) (or limiting distribution if possible).\(^1\)
2. Return \( \sum_{s \in S} \hat{\pi}(s) \cdot r(s). \)

\(^1\)More details later for Markov decision processes.
Definition: Reachability

Given a DTMC \((S, P, \pi_0)\), what is the probability of eventually reaching a set of goal states \(B \subseteq S\)?

Let \(x(s)\) denote \(P_s(\Diamond B)\) where \(\Diamond B = \{s_0s_1 \cdots | \exists i : s_i \in B\}\). Then

- \(s \in B\): \(x(s) = \)
- \(s \in S \setminus B\): \(x(s) = \)
Definition: Reachability

Given a DTMC $(S, P, \pi_0)$, what is the probability of eventually reaching a set of goal states $B \subseteq S$?

Let $x(s)$ denote $P_s(\diamond B)$ where $\diamond B = \{s_0s_1 \cdots \mid \exists i : s_i \in B\}$. Then

- $s \in B$: $x(s) = 1$
- $s \in S \setminus B$: $x(s) = \sum_{t \in S \setminus B} P(s, t)x(t) + \sum_{u \in B} P(s, u)$. 

![Diagram](image-url)
Lemma (Reachability Matrix Form)

Given a DTMC $(S, P, \pi_0)$, the column vector $x = (x(s))_{s \in S \setminus B}$ of probabilities $x(s) = P_s(\diamond B)$ satisfies the constraint

$$x = Ax + b,$$

where matrix $A$ is the submatrix of $P$ for states $S \setminus B$ and $b = (b(s))_{s \in S \setminus B}$ is the column vector with $b(s) = \sum_{u \in B} P(s, u)$. 
DTMC - Reachability

Example:

\[ P = \begin{bmatrix} 0 & 0.5 & 0.5 & 0 \\ 0 & 0.5 & 0.25 & 0.25 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad A = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}, \quad b = 0 \]

\[ \mathbf{B} = \{s_3\} \]

The vector \( \mathbf{x} = [x_0 \ x_1 \ x_2]^T = [0.25 \ 0.5 \ 0]^T \) satisfies the equation system \( \mathbf{x} = A \mathbf{x} + \mathbf{b} \).
DTMC - Reachability

Example:

The vector $x = [x_0 \ x_1 \ x_2]^T = [0.25 \ 0.5 \ 0]^T$ satisfies the equation system $x = Ax + b$.

Is it the only solution?
**Example:**

![Transition Diagram]

\[ P = \begin{bmatrix}
0 & 0.5 & 0.5 & 0 \\
0 & 0.5 & 0.25 & 0.25 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}, \quad B = \{s_3\} \]

The vector \( \mathbf{x} = [x_0 \ x_1 \ x_2]^T = [0.25 \ 0.5 \ 0]^T \) satisfies the equation system \( \mathbf{x} = \mathbf{A}\mathbf{x} + \mathbf{b} \).

**Is it the only solution?**

- **No!** Consider, e.g., \( [0.55 \ 0.7 \ 0.4]^T \) or \( [1 \ 1 \ 1]^T \).
- While reaching the goal, such **bad** states need to be avoided.
- We first generalise such “avoiding”.
Definition:
Let $B, C \subseteq S$. The (unbounded) probability of reaching $B$ from state $s$ under the condition that $C$ is not left before is defined as $P_s(C \cup B)$ where

$$C \cup B = \{s_0s_1\cdots | \exists i : s_i \in B \land \forall j < i : s_j \in C\}.$$
Definition:
Let \( B, C \subseteq S \). The (unbounded) probability of reaching \( B \) from state \( s \) under the condition that \( C \) is not left before is defined as \( P_s(C \cup B) \) where

\[
C \cup B = \{s_0s_1 \cdots | \exists i : s_i \in B \land \forall j < i : s_j \in C\}.
\]

The probability of reaching \( B \) from state \( s \) within \( n \) steps under the condition that \( C \) is not left before is defined as \( P_s(C \cup \leq^n B) \) where

\[
C \cup \leq^n B = \{s_0s_1 \cdots | \exists i \leq n : s_i \in B \land \forall j < i : s_j \in C\}.
\]

What is the equation system for these probabilities?
Let \( S_0 = \{ s \mid P_s(C \cup B) = 0 \} \) and \( S^? = S \setminus (S_0 \cup B) \).
Let \( S_{=0} = \{ s \mid P_s(C \cup B) = 0 \} \) and \( S? = S \setminus (S_{=0} \cup B) \).

**Theorem:**
The column vector \( x = (x(s))_{s \in S?} \) of probabilities \( x(s) = P_s(C \cup B) \)
is the unique solution of the equation system

\[
x = Ax + b,
\]

where \( A = (P(s, t))_{s, t \in S?} \), \( b = (b(s))_{s \in S?} \) with \( b(s) = \sum_{u \in B} P(s, u) \).

Furthermore, for \( x_0 = (0)_{s \in S?} \) and \( x_i = Ax_{i-1} + b \) for any \( i \geq 1 \),

1. \( x_n(s) = P_s(C \cup B \leq^n B) \) for \( s \in S? \),
2. \( x_i \) is increasing, and
3. \( x = \lim_{n \to \infty} x_n \).
Proof Sketch:

- $(x_s)_{x \in S}$ is a solution: by inserting into definition.
- Unique solution: By contradiction. Assume $y$ is another solution, then $x - y = A(x - y)$. One can show that $A - I$ is invertible, thus $(A - I)(x - y) = 0$ yields $x - y = (A - I)^{-1}0 = 0$ and finally $x = y$.  

Furthermore,

1. From the definitions, by straightforward induction.
2. From 1. since $C \cup B \subseteq C \cup B \subseteq n+1 B$.
3. Since $C \cup B = \bigcup_{n \in \mathbb{N}} C \cup B \subseteq n B$.

---

2cf. page 766 of Principles of Model Checking
Algorithmic aspects
Algorithmic Aspects - Summary of Equation Systems

Equation Systems

- Transient analysis: $\pi_n = \pi_0 P^n = \pi_{n-1} P$
- Steady-state analysis: $\pi P = \pi$, $\pi \cdot 1 = \sum_{s \in S} \pi(s) = 1$ (ergodic)
- Reachability: $x = Ax + b$ (with $(x(s))_{s \in S}$)

Solution Techniques

1. Analytic solution, e.g. by Gaussian elimination
2. Iterative power method ($\pi_n \to \pi$ and $x_n \to x$ for $n \to \infty$)
3. Iterative methods for solving large systems of linear equations, e.g. Jacobi, Gauss-Seidel

Missing pieces

a. finding out whether a DTMC is ergodic,
b. computing $S_\emptyset = S \setminus \{s \mid P_s(\diamond B) = 0\}$,
c. efficient representation of $P$. 
Ergodicity = Irreducibility + Aperiodicity + P. Recurrence

- A DTMC is called **irreducible** if for all states \( i, j \in S \) we have \( p^n_{ij} > 0 \) for some \( n \geq 1 \).
- A state \( i \) is called **aperiodic** if \( \gcd\{n \mid p^n_{ii} > 0\} = 1 \).
- A state \( i \) is called **positive recurrent** if \( f_{ii} = 1 \) and \( m_{ii} < \infty \).

How do we tell that a finite DTMC is ergodic?
Ergodicity = Irreducibility + Aperiodicity + P. Recurrence

- A DTMC is called irreducible if for all states $i, j \in S$ we have $p_{ij}^n > 0$ for some $n \geq 1$.
- A state $i$ is called aperiodic if $\gcd\{n \mid p_{ii}^n > 0\} = 1$.
- A state $i$ is called positive recurrent if $f_{ii} = 1$ and $m_{ii} < \infty$.

How do we tell that a finite DTMC is ergodic?

By analysis of the induced graph!

For a DTMC $(S, P, \pi(0))$ we define the induced directed graph $(S, E)$ with $E = \{(s, s') \mid P(s, s') > 0\}$.

Recall:

- A directed graph is called strongly connected if there is a path from each vertex to every other vertex.
- Strongly connected components (SCC) are its maximal strongly connected subgraphs.
- A SCC $T$ is bottom (BSCC) if no $s \not\in T$ is reachable from $T$. 
Algorithmic Aspects: a. Ergodicity of finite DTMC (2)

Ergodicity = Irreducibility + Aperiodicity + P. Recurrence

- A DTMC is called **irreducible** if for all states $i, j \in S$ we have $p^n_{ij} > 0$ for some $n \geq 1$.
- A state $i$ is called **aperiodic** if $\gcd\{n \mid p^n_{ii} > 0\} = 1$.
- A state $i$ is called **positive recurrent** if $f_{ii} = 1$ and $m_{ii} < \infty$.

**Theorem:**
For **finite** DTMCs, it holds that:
Ergodicity = Irreducibility + Aperiodicity + P. Recurrence

- A DTMC is called irreducible if for all states $i, j \in S$ we have $p_{ij}^n > 0$ for some $n \geq 1$.
- A state $i$ is called aperiodic if $\gcd\{n \mid p_{ii}^n > 0\} = 1$.
- A state $i$ is called positive recurrent if $f_{ii} = 1$ and $m_{ii} < \infty$.

Theorem:
For finite DTMCs, it holds that:
- The DTMC is irreducible iff the induced graph is strongly connected.
Ergodicity = Irreducibility + Aperiodicity + P. Recurrence

- A DTMC is called irreducible if for all states $i, j \in S$ we have $p_{ij}^n > 0$ for some $n \geq 1$.
- A state $i$ is called aperiodic if $\gcd\{n \mid p_{ii}^n > 0\} = 1$.
- A state $i$ is called positive recurrent if $f_{ii} = 1$ and $m_{ii} < \infty$.

**Theorem:**
For finite DTMCs, it holds that:

- The DTMC is irreducible iff the induced graph is strongly connected.
- A state in a BSCC is aperiodic iff the BSCC is aperiodic, i.e. the greatest common divisor of the lengths of all its cycles is 1.
Ergodicity = Irreducibility + Aperiodicity + P. Recurrence

- A DTMC is called **irreducible** if for all states $i, j \in S$ we have $p^n_{ij} > 0$ for some $n \geq 1$.
- A state $i$ is called **aperiodic** if $\gcd\{n \mid p^n_{ii} > 0\} = 1$.
- A state $i$ is called **positive recurrent** if $f_{ii} = 1$ and $m_{ii} < \infty$.

**Theorem:**
For finite DTMCs, it holds that:

- The DTMC is **irreducible** iff the induced graph is strongly connected.
- A state in a BSCC is **aperiodic** iff the BSCC is aperiodic, i.e. the greatest common divisor of the lengths of all its cycles is 1.
- A state is **positive recurrent** iff it belongs to a BSCC otherwise it is **transient**.
Algorithmic Aspects: a. Ergodicity of finite DTMC (3)

How to check: is gcd of the lengths of all cycles of a strongly connected graph 1?
How to check: is gcd of the lengths of all cycles of a strongly connected graph 1?

\[ \text{gcd}\{n \geq 1 \mid \exists s : P^n(s, s) > 0\} = 1 \]
Algorithmic Aspects: a. Ergodicity of finite DTMC (3)

How to check: is gcd of the lengths of all cycles of a strongly connected graph 1?

- $\gcd\{n \geq 1 \mid \exists s : P^n(s,s) > 0\} = 1$
- in time $O(n + m)$?
Algorithmic Aspects: a. Ergodicity of finite DTMC (3)

How to check: is gcd of the lengths of all cycles of a strongly connected graph 1?

- $\gcd\{n \geq 1 \mid \exists s : P^n(s, s) > 0\} = 1$
- in time $\mathcal{O}(n + m)$? By the following DFS-based procedure:

Algorithm: PERIOD(vertex $v$, unsigned level : init 0)

1. global period : init 0;
2. if period = 1 then
3.     return
4. end
5. if $v$ is unmarked then
6.     mark $v$;
7.     $v_{\text{level}} = \text{level}$;
8.     for $v' \in \text{out}(v)$ do
9.         PERIOD($v'$, level + 1)
10.     end
11. else
12.     period = gcd(period, level − $v_{\text{level}}$);
13. end
Algorithmic Aspects: b. Computing the set $S_?$

We have $S_? = S \setminus (B \cup S_{=0})$ where $S_{=0} = \{s \mid P_s(\Diamond B) = 0\}$. Hence,

$$s \in S_{=0} \iff p^n_{ss'} = 0 \text{ for all } n \geq 1 \text{ and } s' \in B.$$
Algorithmic Aspects: b. Computing the set \( S_? \)

We have \( S_? = S \setminus (B \cup S_{=0}) \) where \( S_{=0} = \{s \mid P_s(\Diamond B) = 0\} \).

Hence,

\[
s \in S_{=0} \text{ iff } p^{n}_{ss'} = 0 \text{ for all } n \geq 1 \text{ and } s' \in B.
\]

This can be again easily checked from the induced graph:

**Lemma**

We have \( s \in S_{=0} \) iff there is no path from \( s \) to any state from \( B \).

**Proof.**

Easy from the fact that \( p^{n}_{ss'} > 0 \) iff there is a path of length \( n \) to \( s' \).
1. There are many entries in the transition matrix.
2. There are many similar entries in the transition matrix.

Sparse matrices offer a more concise storage.

Multi-terminal binary decision diagrams offer a more concise storage, using automata theory.
1. There are many 0 entries in the transition matrix. 
Sparse matrices offer a more concise storage.
1. There are many 0 entries in the transition matrix. Sparse matrices offer a more concise storage.

2. There are many similar entries in the transition matrix. Multi-terminal binary decision diagrams offer a more concise storage, using automata theory.
DTMC - Probabilistic Temporal Logics for Specifying Complex Properties
Definition:
A labeled DTMC is a tuple $\mathcal{D} = (S, P, \pi_0, L)$ with $L : S \rightarrow 2^{AP}$, where
- $AP$ is a set of atomic propositions and
- $L$ is a labeling function, where $L(s)$ specifies which properties hold in state $s \in S$. 
States and transitions

state = configuration of the game;
transition = rolling the dice and acting (randomly) based on the result.

State labels

- init, rwin, bwin, rkicked, bkicked, ...
- r30, r21, ...
- b30, b21, ...

Examples of Properties

- the game cannot return back to start
- at any time, the game eventually ends with prob. 1
- at any time, the game ends within 100 dice rolls with prob. ≥ 0.5
- the probability of winning without ever being kicked out is ≤ 0.3

How to specify them formally?
Linear-time view

- corresponds to our (human) perception of time
- can specify properties of one concrete linear execution of the system

Example: eventually red player is kicked out followed immediately by blue player being kicked out.

Branching-time view

- views future as a set of all possibilities
- can specify properties of all executions from a given state – specifies execution trees

Example: in every computation it is always possible to return to the initial state.
Linear Temporal Logic (LTL)
Syntax for formulae specifying executions:

\[ \psi = true | a | \psi \land \psi | \neg \psi | X \psi | \psi U \psi | F \psi | G \psi \]

Example: eventually red player is kicked out followed immediately by blue player being kicked out: \( F (rkicked \land X bkicked) \)

Question: do all executions satisfy the given LTL formula?

Computation Tree Logic (CTL)
Syntax for specifying states:

\[ \phi = true | a | \phi \land \phi | \neg \phi | A \psi | E \psi \]

Syntax for specifying executions:

\[ \psi = X \phi | \phi U \phi | F \phi | G \phi \]

Example: in all computations it is always possible to return to initial state: \( A G E F init \)

Question: does the given state satisfy the given CTL state formula?
**Logics - LTL**

**Syntax**

\[ \psi = true \mid a \mid \psi \land \psi \mid \neg \psi \mid X \psi \mid \psi U \psi. \]

**Semantics (for a path \( \omega = s_0s_1 \cdots \))**

- \( \omega \models true \) (always),
- \( \omega \models a \) iff \( a \in L(s_0) \),
- \( \omega \models \psi_1 \land \psi_2 \) iff \( \omega \models \psi_1 \) and \( \omega \models \psi_2 \),
- \( \omega \models \neg \psi \) iff \( \omega \not\models \psi \),
- \( \omega \models X \psi \) iff \( s_1s_2 \cdots \models \psi \),

\[
\begin{array}{llllllllllllll}
\psi \\
\end{array}
\]

- \( \omega \models \psi_1 U \psi_2 \) iff \( \exists i \geq 0 : s_is_{i+1} \cdots \models \psi_2 \) and \( \forall j < i : s_js_{j+1} \cdots \models \psi_1 \).

\[
\begin{array}{llllllllllllll}
\psi_1 \cdot \cdot \cdot \psi_1 \psi_2 \\
\end{array}
\]

**Syntactic sugar**

- \( F \psi \equiv \)
- \( G \psi \equiv \)
Syntax \[ \psi = true \mid a \mid \psi \land \psi \mid \neg \psi \mid X \psi \mid \psi U \psi. \]

Semantics (for a path \( \omega = s_0s_1 \cdots \))

- \( \omega \models true \) (always),
- \( \omega \models a \) iff \( a \in L(s_0) \),
- \( \omega \models \psi_1 \land \psi_2 \) iff \( \omega \models \psi_1 \) and \( \omega \models \psi_2 \),
- \( \omega \models \neg \psi \) iff \( \omega \not\models \psi \),
- \( \omega \models X \psi \) iff \( s_1s_2 \cdots \models \psi \),
- \( \omega \models \psi_1 U \psi_2 \) iff \( \exists i \geq 0 : s_is_{i+1} \cdots \models \psi_2 \) and \( \forall j < i : s_js_{j+1} \cdots \models \psi_1 \).

Syntactic sugar

- \( F \psi \equiv true U \psi \)
- \( G \psi \equiv \neg(true U \neg \psi) \quad (\equiv \neg F \neg \psi) \)
**Logics - CTL**

**Syntax**

State formulae:

\[ \phi = \text{true} \mid a \mid \phi \land \phi \mid \neg \phi \mid A \psi \mid E \psi \]

where \( \psi \) is a path formula.

**Semantics**

For a state \( s \):

- \( s \models \text{true} \) (always),
- \( s \models a \) iff \( a \in L(s) \),
- \( s \models \phi_1 \land \phi_2 \) iff \( s \models \phi_1 \) and \( s \models \phi_2 \),
- \( s \models \neg \phi \) iff \( s \not\models \phi \),
- \( s \models A\psi \) iff \( \omega \models \psi \) for all paths \( \omega = s_0s_1 \cdots \) with \( s_0 = s \),
- \( s \models E\psi \) iff \( \omega \models \psi \) for some path \( \omega = s_0s_1 \cdots \) with \( s_0 = s \).

**Path formulae:**

\[ \psi = X \phi \mid \phi \cup \phi \]

where \( \phi \) is a state formula.

For a path \( \omega = s_0s_1 \cdots \):

- \( \omega \models X \phi \) iff \( s_1s_2 \cdots \) satisfies \( \phi \),
- \( \omega \models X \phi \) iff \( \exists i : s_is_{i+1} \cdots \models \phi_2 \) and \( \forall j < i : s_js_{j+1} \cdots \models \phi_1 \).
Linear Temporal Logic (LTL)
Syntax for formulae specifying executions:

\[ \psi = \text{true} \mid a \mid \psi \land \psi \mid \neg\psi \mid X \psi \mid \psi U \psi \mid F \psi \mid G \psi \]

Example: eventually red player is kicked out followed immediately by blue player being kicked out: \( F (rkicked \land X bkicked) \)
Question: do all executions satisfy the given LTL formula?

Computation Tree Logic (CTL)
Syntax for specifying states:

\[ \phi = \text{true} \mid a \mid \phi \land \phi \mid \neg\phi \mid A \psi \mid E \psi \]

Syntax for specifying executions:

\[ \psi = X \phi \mid \phi U \phi \mid F \phi \mid G \phi \]

Example: in all computations it is always possible to return to initial state: \( A G E F \text{ init} \)
Question: does the given state satisfy the given CTL state formula?
Logics - Temporal Logics - probabilistic

Linear Temporal Logic (LTL) + probabilities

Syntax for formulae specifying executions:

\[ \psi = \text{true} | a | \psi \land \psi | \neg \psi | X \psi | \psi U \psi | F \psi | G \psi \]

Example: with prob. \( \geq 0.8 \), eventually red player is kicked out followed immediately by blue player being kicked out:

\[ P(F (rkicked \land X bkicked)) \geq 0.8 \]

Question: is the formula satisfied by executions of given probability?
Linear Temporal Logic (LTL) + probabilities
Syntax for formulae specifying executions:

\[ \psi = \text{true} \mid a \mid \psi \land \psi \mid \neg \psi \mid X \psi \mid \psi U \psi \mid F \psi \mid G \psi \]

Example: with prob. \( \geq 0.8 \), eventually red player is kicked out followed immediately by blue player being kicked out:

\[ P(F (rkicked \land X bkicked)) \geq 0.8 \]

Question: is the formula satisfied by executions of given probability?

Probabilistic Computation Tree Logic (PCTL)
Syntax for specifying states:

\[ \phi = \text{true} \mid a \mid \phi \land \phi \mid \neg \phi \mid P_j \psi \]

Syntax for specifying executions:

\[ \psi = X \phi \mid \phi U \phi \mid \phi U \leq^k \phi \mid F \phi \mid G \phi \]

Example: with prob. at least 0.5 the probability to return to initial state is always at least 0.1:

\[ P_{\geq 0.5} G P_{\geq 0.1} F \text{init} \]

Question: does the given state satisfy the given PCTL state formula?
Logics - PCTL - Examples

Syntactic sugar:

- $\phi_1 \lor \phi_2 \equiv \neg (\neg \phi_1 \land \neg \phi_2)$, $\phi_1 \Rightarrow \phi_2 \equiv \neg \phi_1 \lor \phi_2$, etc.
- $\leq 0.5$ denotes the interval $[0, 0.5]$, $= 1$ denotes $[1, 1]$, etc.

Examples:

- A fair die:

\[
\bigwedge_{i \in \{1, \ldots, 6\}} P_{\frac{1}{6}}(\mathcal{F} i).
\]

- The probability of winning “Who wants to be a millionaire” without using any joker should be negligible:

\[
\mathcal{P} < 1e^{-10}(\neg (J_{50\%} \lor J_{audience} \lor J_{telephone}) \cup \text{win}).
\]
Semantics

For a state $s$:

- $s \models true$ (always),
- $s \models a$ iff $a \in L(s)$,
- $s \models \phi_1 \land \phi_2$ iff $s \models \phi_1$ and $s \models \phi_2$,
- $s \models \neg \phi$ iff $s \not\models \phi$,
- $s \models P_J(\psi)$ iff $P_s(Paths(\psi)) \in J$

For a path $\omega = s_0s_1 \cdots$:

- $\omega \models X \phi$ iff $s_1s_2 \cdots$ satisfies $\phi$,
- $\omega \models \phi_1 U \phi_2$ iff $\exists i : s_is_{i+1} \cdots \models \phi_2$ and $\forall j < i : s_js_{j+1} \cdots \models \phi_1$,
- $\omega \models \phi_1 U \leq n \phi_2$ iff $\exists i \leq n : s_is_{i+1} \cdots \models \phi_2$ and $\forall j < i : s_js_{j+1} \cdots \models \phi_1$. 
Logics - Examples of Properties

Examples of Properties

1. the game cannot return back to start
2. at any time, the game eventually ends with prob. \( 1 \)
3. at any time, the game ends within 100 dice rolls with prob. \( \geq 0.5 \)
4. the probability of winning without ever being kicked out is \( \leq 0.3 \)

Formally
Examples of Properties

1. the game cannot return back to start
2. at any time, the game eventually ends with prob. 1
3. at any time, the game ends within 100 dice rolls with prob. \( \geq 0.5 \)
4. the probability of winning without ever being kicked out is \( \leq 0.3 \)

Formally

1. \( P(\mathcal{X} \mathcal{G} \neg \text{init}) = 1 \) (LTL + prob.)
   \[ P_{=1}(\mathcal{X} P_{=0}(\mathcal{G} \neg \text{init})) \) (PCTL)
2. \( P_{=1}(\mathcal{G} P_{=1}(\mathcal{F} (rwin \lor bwin))) \) (PCTL)
3. \( P_{=1}(\mathcal{G} P_{\geq 0.5}(\mathcal{F} \leq_{100} (rwin \lor bwin))) \) (PCTL)
4. \( P((\neg rkicked \land \neg bkicked) U (rwin \lor bwin)) \leq 0.3 \) (LTL + prob.)
PCTL Model Checking Algorithm
**PCTL Model Checking**

**Definition: PCTL Model Checking**
Let $\mathcal{D} = (S, P, \pi_0, L)$ be a DTMC, $\Phi$ a PCTL state formula and $s \in S$. The model checking problem is to decide whether $s \models \Phi$.

**Theorem**
The PCTL model checking problem can be decided in time polynomial in $|\mathcal{D}|$, linear in $|\Phi|$, and linear in the maximum step bound $n$. 
Algorithm:
Consider the bottom-up traversal of the parse tree of $\Phi$:

- The leaves are $a \in AP$ or $true$ and
- the inner nodes are:
  - unary – labelled with the operator $\neg$ or $P_J(X)$;
  - binary – labelled with an operator $\land$, $P_J(U)$, or $P_J(U \leq n)$.

Example: $\neg a \land P_{\leq 0.2}(\neg b \ U \ P_{\geq 0.9}(\Diamond c))$

Compute $Sat(\Psi) = \{ s \in S \mid s \models \Psi \}$ for each node $\Psi$ of the tree in a bottom-up fashion. Then $s \models \Phi$ iff $s \in Sat(\Phi)$. 
“Base” of the algorithm:
We need a procedure to compute $Sat(\psi)$ for $\psi$ of the form $a$ or $true$:
"Base" of the algorithm:
We need a procedure to compute \( Sat(\psi) \) for \( \psi \) of the form \( a \) or \( true \):

**Lemma**

- \( Sat(true) = S \),
- \( Sat(a) = \{ s \mid a \in L(s) \} \)
“Base” of the algorithm:
We need a procedure to compute \( \text{Sat}(\Psi) \) for \( \Psi \) of the form \( a \) or \( \text{true} \):

Lemma
- \( \text{Sat}(\text{true}) = S \),
- \( \text{Sat}(a) = \{ s \mid a \in L(s) \} \)

“Induction” step of the algorithm:
We need a procedure to compute \( \text{Sat}(\Psi) \) for \( \Psi \) given the sets \( \text{Sat}(\Psi') \) for all state sub-formulas \( \Psi' \) of \( \Psi \):

Lemma
- \( \text{Sat}(\Phi_1 \land \Phi_2) = \)
- \( \text{Sat}(\neg \Phi) = \)
“Base” of the algorithm:
We need a procedure to compute $\text{Sat}(\Psi)$ for $\Psi$ of the form $a$ or $\text{true}$:

Lemma

$\text{Sat}(\text{true}) = S$,

$\text{Sat}(a) = \{s \mid a \in L(s)\}$

“Induction” step of the algorithm:
We need a procedure to compute $\text{Sat}(\Psi)$ for $\Psi$ given the sets $\text{Sat}(\Psi')$ for all state sub-formulas $\Psi'$ of $\Psi$:

Lemma

$\text{Sat}(\Phi_1 \land \Phi_2) = \text{Sat}(\Phi_1) \cap \text{Sat}(\Phi_2)$

$\text{Sat}(\neg\Phi) = S \setminus \text{Sat}(\Phi)$

$\text{Sat}(\mathcal{P}_J(\Phi)) = \{s \mid P_s(\text{Paths}(\Phi)) \in J\}$ discussed on the next slide.
Lemma

- **Next:**
  \[ P_s(Paths(\mathcal{X} \Phi)) = \]

- **Bounded Until:**
  \[ P_s(Paths(\Phi_1 \bigcup \leq^n \Phi_2)) = \]

- **Unbounded Until:**
  \[ P_s(Paths(\Phi_1 \bigcup \Phi_2)) = \]
Lemma

- **Next:**
  \[ P_s(\text{Paths}(\mathcal{X} \Phi)) = \sum_{s' \in \text{Sat}(\Phi)} P(s, s') \]

- **Bounded Until:**
  \[ P_s(\text{Paths}(\Phi_1 \cup \leq^n \Phi_2)) = P_s(\text{Sat}(\Phi_1) \cup \leq^n \text{Sat}(\Phi_2)) \]

- **Unbounded Until:**
  \[ P_s(\text{Paths}(\Phi_1 \cup \Phi_2)) = P_s(\text{Sat}(\Phi_1) \cup \text{Sat}(\Phi_2)) \]
Lemma

Next:

\[ P_s(Paths(\chi \ Φ)) = \sum_{s' \in Sat(Φ)} P(s, s') \]

Bounded Until:

\[ P_s(Paths(Φ_1 \cup \lessdot^n Φ_2)) = P_s(Sat(Φ_1) \cup \lessdot^n Sat(Φ_2)) \]

Unbounded Until:

\[ P_s(Paths(Φ_1 \cup Φ_2)) = P_s(Sat(Φ_1) \cup Sat(Φ_2)) \]

As before:
can be reduced to transient analysis and to unbounded reachability.
Precise algorithm

Computation for every node in the parse tree and for every state:

- All node types except for path operator – trivial.
- Next: Trivial.
- Until: Solving equation systems can be done by polynomially many elementary arithmetic operations.
- Bounded until: Matrix vector multiplications can be done by polynomial many elementary arithmetic operations as well.

Overall complexity:
Polynomial in $|D|$, linear in $|\Phi|$ and the maximum step bound $n$.

In practice

The until and bounded until probabilities computed approximatively:

- rounding off probabilities in matrix-vector multiplication,
- using approximative iterative methods without error guarantees.
pLTL Model Checking Algorithm
Definition: LTL Model Checking
Let $\mathcal{D} = (S, P, \pi_0, L)$ be a DTMC, $\Psi$ a LTL formula, $s \in S$, and $p \in [0, 1]$. The model checking problem is to decide whether $s \models P^\mathcal{D}_s(Paths(\Psi)) \geq p$.

Theorem
The LTL model checking can be decided in time $O(|\mathcal{D}| \cdot 2^{|\Psi|})$. 
LTL Model Checking – Overview

Definition: LTL Model Checking
Let $D = (S, P, \pi_0, L)$ be a DTMC, $\Psi$ a LTL formula, $s \in S$, and $p \in [0, 1]$. The model checking problem is to decide whether $s \models P^D_s(Paths(\Psi)) \geq p$.

Theorem
The LTL model checking can be decided in time $O(|D| \cdot 2^{|\Psi|})$.

Algorithm Outline

1. Construct from $\Psi$ a deterministic Rabin automaton $A$ recognizing words satisfying $\Psi$, i.e. $Paths(\Psi) := \{L(\omega) \in (2^Ap)^\infty \mid \omega \models \Psi\}$
Definition: LTL Model Checking
Let $\mathcal{D} = (S, P, \pi_0, L)$ be a DTMC, $\Psi$ a LTL formula, $s \in S$, and $p \in [0, 1]$. The model checking problem is to decide whether $s \models P^\mathcal{D}_s(Paths(\Psi)) \geq p$.

Theorem
The LTL model checking can be decided in time $O(|\mathcal{D}| \cdot 2^{|\Psi|})$.

Algorithm Outline
1. Construct from $\Psi$ a deterministic Rabin automaton $A$ recognizing words satisfying $\Psi$, i.e. $Paths(\Psi) := \{L(\omega) \in (2^A)^\infty \mid \omega \models \Psi\}$
2. Construct a product DTMC $\mathcal{D} \times A$ that “embeds” the deterministic execution of $A$ into the Markov chain.
Definition: LTL Model Checking

Let $\mathcal{D} = (S, P, \pi_0, L)$ be a DTMC, $\Psi$ a LTL formula, $s \in S$, and $p \in [0, 1]$. The model checking problem is to decide whether $s \models P_s^D(Paths(\Psi)) \geq p$.

Theorem

The LTL model checking can be decided in time $O(|\mathcal{D}| \cdot 2^{|\Psi|})$.

Algorithm Outline

1. Construct from $\Psi$ a deterministic Rabin automaton $A$ recognizing words satisfying $\Psi$, i.e. $Paths(\Psi) := \{L(\omega) \in (2^A)^\infty \mid \omega \models \Psi\}$
2. Construct a product DTMC $\mathcal{D} \times A$ that “embeds” the deterministic execution of $A$ into the Markov chain.
3. Compute in $\mathcal{D} \times A$ the probability of paths where $A$ satisfies the acceptance condition.
Deterministic Rabin automaton (DRA): \((Q, \Sigma, \delta, q_0, \text{Acc})\)

- a DFA with a different acceptance condition, 
- \(\text{Acc} = \{(E_i, F_i) \mid 1 \leq i \leq k\}\)
- each accepting infinite path must visit for some \(i\)
  - all states of \(E_i\) at most finitely often and
  - some state of \(F_i\) infinitely often.
Deterministic Rabin automaton (DRA): $(Q, \Sigma, \delta, q_0, \text{Acc})$
- a DFA with a different acceptance condition,
- $\text{Acc} = \{(E_i, F_i) \mid 1 \leq i \leq k\}$
- each accepting infinite path must visit for some $i$
  - all states of $E_i$ at most finitely often and
  - some state of $F_i$ infinitely often.

Example
Give some automata recognizing the language of formulas
- $(a \land X b) \lor aUb$
- $FGa$
- $GFa$
Deterministic Rabin automaton (DRA): \((Q, \Sigma, \delta, q_0, \text{Acc})\)
- a DFA with a different acceptance condition,
- \(\text{Acc} = \{(E_i, F_i) \mid 1 \leq i \leq k\}\)
- each accepting infinite path must visit for some \(i\)
  - all states of \(E_i\) at most finitely often and
  - some state of \(F_i\) infinitely often.

Example
Give some automata recognizing the language of formulas
- \((a \land X b) \lor aUc\)
- \(FGa\)
- \(GFa\)

Lemma (Vardi&Wolper’86, Safra’88)
For any LTL formula \(\Psi\) there is a DRA \(A\) recognizing \(\text{Paths}(\Psi)\) with \(|A| \in 2^{O(|\Psi|)}\).
For a labelled DTMC $\mathcal{D} = (S, P, \pi_0, L)$ and a DRA $A = (Q, 2^A, \delta, q_0, \{(E_i, F_i) \mid 1 \leq i \leq k\})$ we define

1. a DTMC $\mathcal{D} \times A = (S \times Q, P', \pi_0')$:
   - $P'((s, q), (s', q')) = P(s, s')$ if $\delta(q, L(s')) = q'$ and 0, otherwise;
   - $\pi_0'((s, q_s)) = \pi_0(s)$ if $\delta(q_0, L(s)) = q_s$ and 0, otherwise; and
For a labelled DTMC $\mathcal{D} = (S, P, \pi_0, L)$ and a DRA $A = (Q, 2^{Ap}, \delta, q_0, \{(E_i, F_i) \mid 1 \leq i \leq k\})$ we define

1. a DTMC $\mathcal{D} \times A = (S \times Q, P', \pi'_0)$:
   - $P'((s, q), (s', q')) = P(s, s')$ if $\delta(q, L(s')) = q'$ and 0, otherwise;
   - $\pi'_0((s, q_s)) = \pi_0(s)$ if $\delta(q_0, L(s)) = q_s$ and 0, otherwise; and

2. $\{(E'_i, F'_i) \mid 1 \leq i \leq k\}$ where for each $i$:
   - $E'_i = \{(s, q) \mid q \in E_i, s \in S\}$,
   - $F'_i = \{(s, q) \mid q \in F_i, s \in S\}$,
For a labelled DTMC $D = (S, P, \pi_0, L)$ and a DRA $A = (Q, 2^{Ap}, \delta, q_0, \{(E_i, F_i) \mid 1 \leq i \leq k\})$ we define

1. a DTMC $D \times A = (S \times Q, P', \pi'_0)$:
   - $P'((s, q), (s', q')) = P(s, s')$ if $\delta(q, L(s')) = q'$ and 0, otherwise;
   - $\pi'_0((s, q)) = \pi_0(s)$ if $\delta(q_0, L(s)) = q_s$ and 0, otherwise; and

2. $\{(E'_i, F'_i) \mid 1 \leq i \leq k\}$ where for each $i$:
   - $E'_i = \{(s, q) \mid q \in E_i, s \in S\}$,
   - $F'_i = \{(s, q) \mid q \in F_i, s \in S\}$,

**Lemma**

The construction preserves probability of accepting as

$$P_s^D(Lang(A)) = P_{(s, q_s)}^{D \times A}(\{\omega \mid \exists i : \inf(\omega) \cap E'_i = \emptyset, \inf(\omega) \cap F'_i \neq \emptyset\})$$

where $\inf(\omega)$ is the set of states visited in $\omega$ infinitely often.

**Proof sketch.**

We have a one-to-one correspondence between executions of $D$ and $D \times A$ (as $A$ is deterministic), mapping $Lang(A)$ to $\{\cdots\}$, and preserving probabilities.
How to check the probability of accepting in $\mathcal{D} \times \mathcal{A}$?

|Lemma| $P_{\mathcal{D} \times \mathcal{A}}(s, q_s)(\omega) = P_{\mathcal{D} \times \mathcal{A}}(s, q_s)(\diamond \bigcup_j C_j)$.

|Proof sketch.| Note that some BSCC of each finite DTMC is reached with probability 1 (short paths with prob. bounded from below), Rabin acceptance condition does not depend on any finite prefix of the infinite word, every state of a finite irreducible DTMC is visited infinitely often with probability 1 regardless of the choice of initial state.

|Corollary| $P_{\mathcal{D}}(\text{Lang}(\mathcal{A})) = P_{\mathcal{D} \times \mathcal{A}}(s, q_s)(\diamond \bigcup_j C_j)$.}
How to check the probability of accepting in $D \times A$?

Identify the BSCCs $(C_j)_j$ of $D \times A$ that for some $1 \leq i \leq k$,

1. contain no state from $E'_i$ and
2. contain some state from $F'_i$.

Lemma

\[ P_{(s,q_s)}^{D \times A}(\{\omega \mid \exists i: \inf(\omega) \cap E'_i = \emptyset, \inf(\omega) \cap F'_i \neq \emptyset\}) = P_{(s,q_s)}^{D \times A}(\diamond \bigcup_j C_j). \]
How to check the probability of accepting in $D \times A$?

Identify the BSCCs $(C_j)_j$ of $D \times A$ that for some $1 \leq i \leq k$,

1. contain no state from $E'_i$ and
2. contain some state from $F'_i$.

Lemma

$$P_{(s,q_s)}^{D \times A}(\omega | \exists i : \inf(\omega) \cap E'_i = \emptyset, \inf(\omega) \cap F'_i \neq \emptyset) = P_{(s,q_s)}^{D \times A}(\Diamond \bigcup_j C_j).$$

Proof sketch.

- Note that some BSCC of each finite DTMC is reached with probability 1 (short paths with prob. bounded from below),
- Rabin acceptance condition does not depend on any finite prefix of the infinite word,
- every state of a finite irreducible DTMC is visited infinitely often with probability 1 regardless of the choice of initial state.
How to check the probability of accepting in $D \times A$?

Identify the BSCCs $(C_j)_j$ of $D \times A$ that for some $1 \leq i \leq k$,
1. contain no state from $E'_i$ and
2. contain some state from $F'_i$.

Lemma

$$P_{(s,q_s)}^{D \times A}(\{\omega \mid \exists i : \inf(\omega) \cap E'_i = \emptyset, \inf(\omega) \cap F'_i \neq \emptyset\}) = P_{(s,q_s)}^{D \times A}(\Diamond \bigcup_j C_j).$$

Proof sketch.

- Note that some BSCC of each finite DTMC is reached with probability 1 (short paths with prob. bounded from below),
- Rabin acceptance condition does not depend on any finite prefix of the infinite word,
- every state of a finite irreducible DTMC is visited infinitely often with probability 1 regardless of the choice of initial state.

Corollary

$$P_s^D(\text{Lang}(A)) = P_{(s,q_s)}^{D \times A}(\Diamond \bigcup_j C_j).$$
Doubly exponential in $\Psi$ and polynomial in $D$
(for the algorithm presented here):

1. $|A|$ and hence also $|D \times A|$ is of size $2^{O(|\Psi|)}$
2. BSCC computation: Tarjan algorithm - linear in $|D \times A|$
   (number of states + transitions)
3. Unbounded reachability: system of linear equations ($\leq |D \times A|$):
   - exact solution: $\approx$ cubic in the size of the system
   - approximative solution: efficient in practice