Knowledge Compilation: Theory, Practice, and Applications

Jean-Marie Lagniez[1] Johannes Fichte[2]

[1] CRIL, Univ. Artois & CNRS, France
[2] Linköping University, Sweden

Outline

How to model using SAT

Outline

Outline

The SAT problem

$$\neg a \vee b \qquad \begin{array}{l} a \vee \neg b \\ \neg b \vee c \vee d \\ e \vee f \end{array} \qquad \begin{array}{l} \neg a \vee c \\ \neg b \vee c \vee \neg d \\ e \vee \neg f \end{array} \qquad \neg b \vee c$$

- Propositional variables: a, b, c, d, e, f
- Literals: $a, \neg a$
- Clauses: $a \vee b$
- A `CNF` formula F: $(a \vee \neg b) \wedge (\neg a \vee c) \wedge \ldots$

The SAT problem

$$\neg a \vee b \qquad \begin{array}{l} a \vee \neg b \\ \neg b \vee c \vee d \\ e \vee f \end{array} \qquad \begin{array}{l} \neg a \vee c \\ \neg b \vee c \vee \neg d \\ e \vee \neg f \end{array} \qquad \neg b \vee c$$

a	b	c	d	e	f

- Propositional variables: a, b, c, d, e, f
- Literals: $a, \neg a$
- Clauses: $a \vee b$
- A `CNF` formula F: $(a \vee \neg b) \wedge (\neg a \vee c) \wedge \ldots$
- An assignment is a mapping $\mathscr{I} : var(F) \rightarrow \{\bot, \top\}$ and

  $\mathscr{I}$ is satisfying if results in F being $\top$ (true)

The SAT problem

$$\neg a \vee b \qquad \begin{array}{l} a \vee \neg b \\ \neg b \vee c \vee d \\ e \vee f \end{array} \qquad \begin{array}{l} \neg a \vee c \\ \neg b \vee c \vee \neg d \\ e \vee \neg f \end{array} \qquad \neg b \vee c$$

a	b	c	d	e	f

- Propositional variables: a, b, c, d, e, f
- Literals: $a, \neg a$
- Clauses: $a \vee b$
- A `CNF` formula F: $(a \vee \neg b) \wedge (\neg a \vee c) \wedge \ldots$
- An assignment is a mapping $\mathscr{I} : var(F) \to \{\bot, \top\}$ and
  $\mathscr{I}$ is satisfying if results in F being $\top$ (true)
- SAT problem: does there exist a satisfying assignment for F?

The SAT problem

$\neg a \vee b$ $\quad$ $a \vee \neg b$ $\quad$ $\neg b \vee c \vee d$ $\quad$ $e \vee f$ $\quad$ $\neg a \vee c$ $\quad$ $\neg b \vee c \vee \neg d$ $\quad$ $e \vee \neg f$ $\quad$ $\neg b \vee c$

a	b	c	d	e	f
$\bot$	$\top$	$\top$	$\top$	$\top$	$\bot$

- Propositional variables: a, b, c, d, e, f
- Literals: $a, \neg a$
- Clauses: $a \vee b$
- A `CNF` formula F: $(a \vee \neg b) \wedge (\neg a \vee c) \wedge \ldots$
- An assignment is a mapping $\mathscr{I} : var(F) \to \{\bot, \top\}$ and
  $\mathscr{I}$ is satisfying if results in F being $\top$ (true)
- SAT problem: does there exist a satisfying assignment for F?

The SAT problem

$$\neg a \vee b \qquad a \vee \neg b \quad \neg b \vee c \vee d \quad e \vee f \qquad \neg a \vee c \quad \neg b \vee c \vee \neg d \quad e \vee \neg f \qquad \neg b \vee c$$

a	b	c	d	e	f
$\bot$	$\bot$	$\top$	$\top$	$\top$	$\bot$

- Propositional variables: a, b, c, d, e, f
- Literals: $a, \neg a$
- Clauses: $a \vee b$
- A `CNF` formula F: $(a \vee \neg b) \wedge (\neg a \vee c) \wedge \ldots$
- An assignment is a mapping $\mathscr{I} : var(F) \to \{\bot, \top\}$ and
  $\mathscr{I}$ is satisfying if results in F being $\top$ (true)
- SAT problem: does there exist a satisfying assignment for F?

Outline

A Brief Detour into Complexity Theory

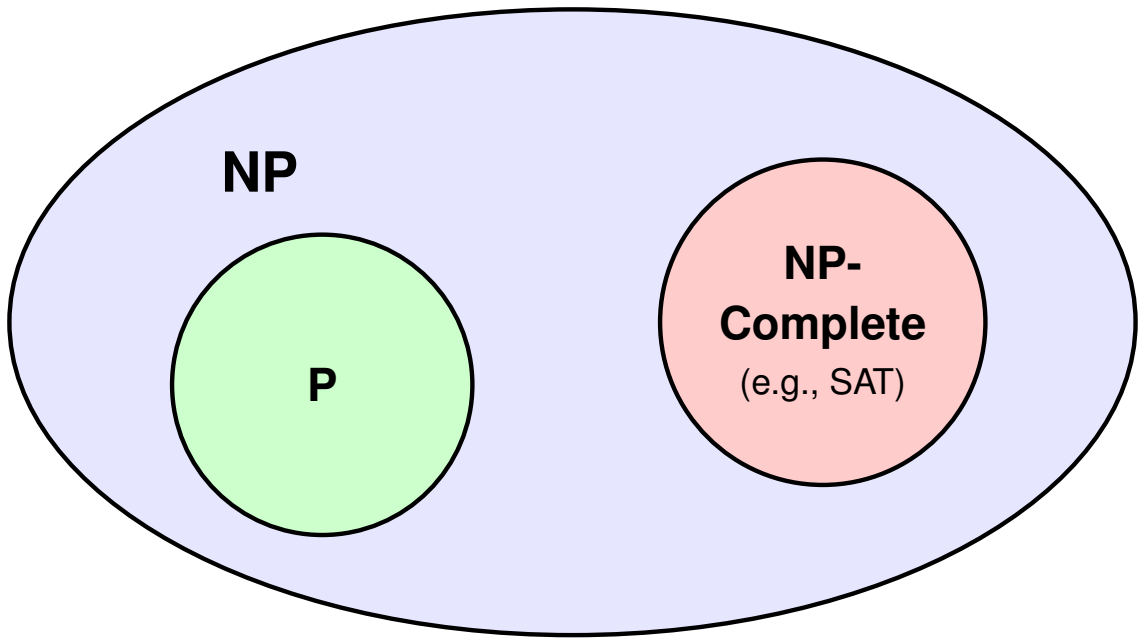
- To understand why using SAT, we must understand how hard SAT is to solve.
- **Class P (Polynomial Time):**
  - Problems that can be solved efficiently by a deterministic algorithm.
  - Examples: Sorting an array, finding the shortest path in a graph.
- **Class NP (Nondeterministic Polynomial Time):**
  - Problems where a proposed solution can be verified efficiently.
  - Example: Sudoku. Hard to solve from scratch, but easy to check if a completed grid is correct.

Does P = NP? (2nd Millennium Prize Problem of Mathematics)

SAT and NP-Completeness

- **Cook-Levin Theorem (Cook, 1971):** SAT was the first problem proven to be NP-Complete.
- What does NP-Complete mean?
  - It is in NP (a satisfying assignment is easy to check).
  - It is as hard as the hardest problems in NP.
  - *Every* problem in NP can be translated (reduced) into a SAT problem efficiently.

- **ETH (Exponential-Time Hypothesis):** Worst-case SAT cannot be solved significantly better than exponential time $\mathcal{O}(2^n)$ (more precisely: there is no subexponential-time algorithm for 3-SAT).



The SAT Solving Paradox

Theory vs. Practice

- **Theory:** SAT is NP-Complete ⇒ intractable in the worst case.
- **Practice:** Modern SAT solvers can routinely solve industrial instances with millions of variables and clauses.

- **How is this possible?**
  - Real-world problems (like hardware verification or XAI) are highly structured, unlike random formulas.
  - Modern solvers exploit this hidden structure.

- **The CDCL (Conflict-Driven Clause Learning) Revolution (Silva and Sakallah, 1996):**
  - When the solver makes a mistake, it analyzes the logical conflict.
  - It *learns* a new clause to prevent making the exact same mistake again.
  - It then backtracks non-chronologically to jump out of dead search spaces.

Outline

Graph Coloring Problem

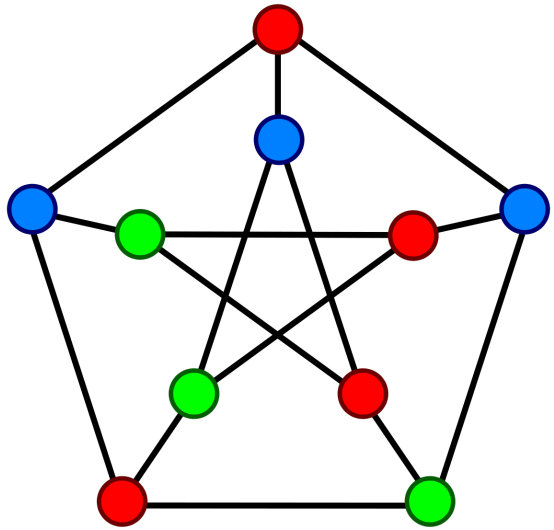
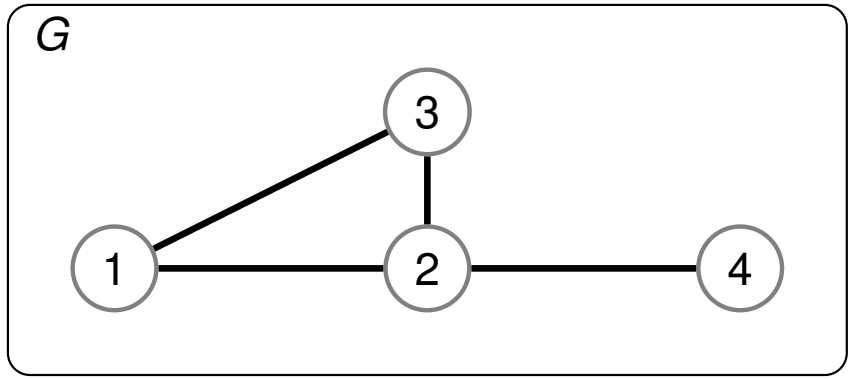


Figure: $\chi(G) = 3$

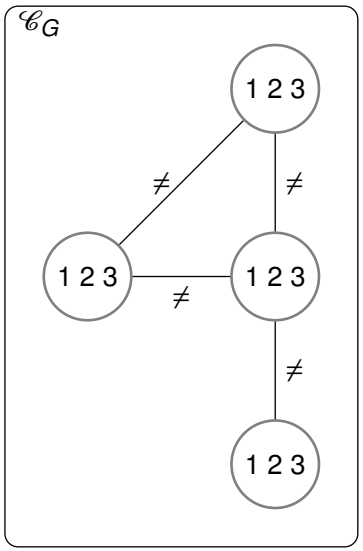
- To assign a minimum number of colors to all vertices s.t. no adjacent vertices have the same color
- The smallest number of colors: *chromatic number* ($\chi(G)$)
- Determining the chromatic number of a graph is an NP-hard task

Classical Graph Coloring Encoding

- *Decision Problem*: Can we color $G = (V, E)$ with 3 colors?



- We can express the problem with a set of constraints

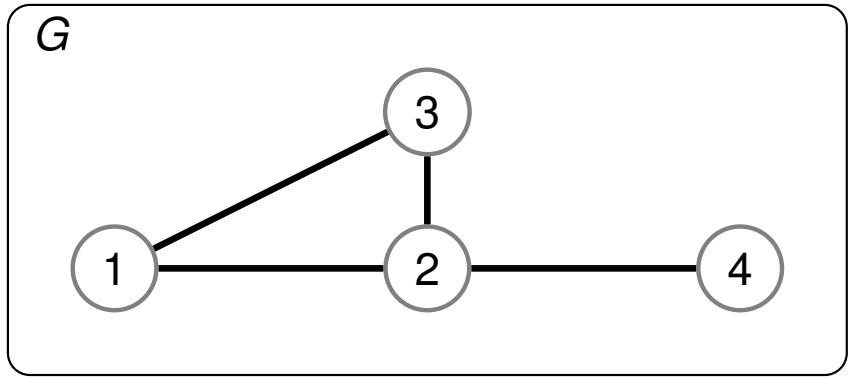


1. x_{ij} is true if the vertex i is colored with the color j
2. At least one color by vertex: $\bigwedge_{i=1}^{4}(\bigvee_{j=1}^{3} x_{ij})$
3. At most one color by vertex: $\bigwedge_{i=1}^{4}\bigwedge_{j=1}^{3}\bigvee_{k=j+1}^{3}(\neg x_{ij} \vee \neg x_{ik})$
4. Two adjency vertices cannot share the same color: $\forall \{i,j\} \in E, \bigwedge_{k=1}^{3} \neg x_{ik} \vee \neg x_{jk}$

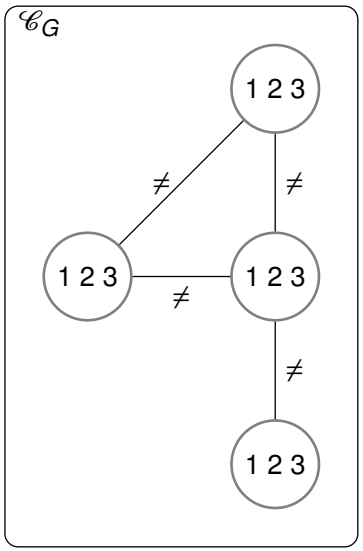
Live example?

Classical Graph Coloring Encoding

- *Decision Problem*: Can we color $G = (V, E)$ with 3 colors?



- We can express the problem with a set of constraints

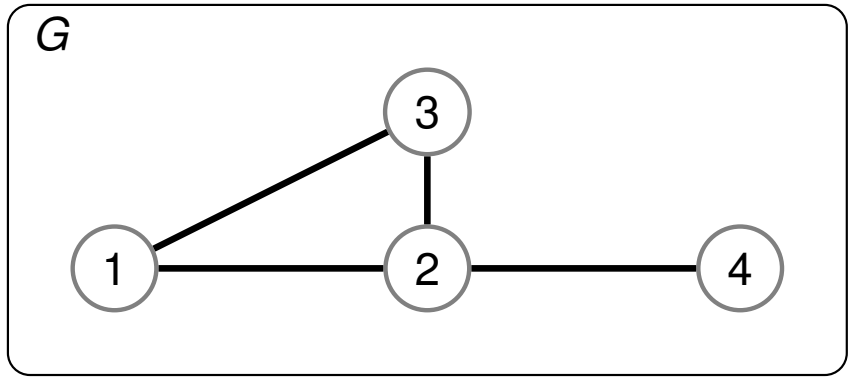


1. x_{ij} is true if the vertex i is colored with the color j
2. At least one color by vertex: $\bigwedge_{i=1}^{4}(\bigvee_{j=1}^{3} x_{ij})$
3. At most one color by vertex: $\bigwedge_{i=1}^{4}\bigwedge_{j=1}^{3}\bigvee_{k=j+1}^{3}(\neg x_{ij} \vee \neg x_{ik})$
4. Two adjency vertices cannot share the same color: $\forall \{i,j\} \in E, \bigwedge_{k=1}^{3} \neg x_{ik} \vee \neg x_{jk}$

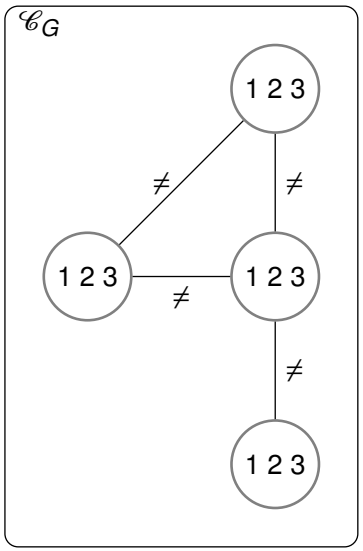
Live example?

Classical Graph Coloring Encoding

- *Decision Problem*: Can we color $G = (V, E)$ with 3 colors?



- We can express the problem with a set of constraints



1. x_{ij} is true if the vertex i is colored with the color j
2. At least one color by vertex: $\bigwedge_{i=1}^{4}(\bigvee_{j=1}^{3} x_{ij})$
3. At most one color by vertex: $\bigwedge_{i=1}^{4}\bigwedge_{j=1}^{3}\bigvee_{k=j+1}^{3}(\neg x_{ij} \vee \neg x_{ik})$
4. Two adjency vertices cannot share the same color: $\forall \{i,j\} \in E, \bigwedge_{k=1}^{3} \neg x_{ik} \vee \neg x_{jk}$

Live example?

The Counting Challenge

Finding one coloring is “easy” (SAT). But how many colorings exist?

Outline

Outline

The Model Counting Problem (#SAT)

Definition

Model Counting is the problem of computing the number of total satisfying assignments (models) for a given propositional formula Σ.

$$\Sigma \mapsto \|\Sigma\| = |\{\mathscr{I} \mid \mathscr{I} \models \Sigma\}|$$

Example: Let $\Sigma = (\neg a \vee \neg b \vee \neg c) \wedge (a \vee c) \wedge (a \vee b) \wedge (\neg b \vee \neg c)$

The models of Σ over variables $\{a, b, c\}$ are:

a	b	c
$\top$	$\bot$	$\bot$
$\top$	$\bot$	$\top$
$\top$	$\top$	$\bot$

Answer: $\|\Sigma\| = 3$

Outline

The Complexity of Model Counting

- **A Central Problem in AI**
  Model counting is crucial for various applications:
  - Probabilistic reasoning (Bayesian inference)
  - Automated verification
  - Planning under uncertainty

- **Computational Hardness**
  #SAT is **#P-complete** (Valiant, 1979).
  - It is significantly harder than SAT (NP-complete).

The Power of Counting: Toda's Theorem 1991

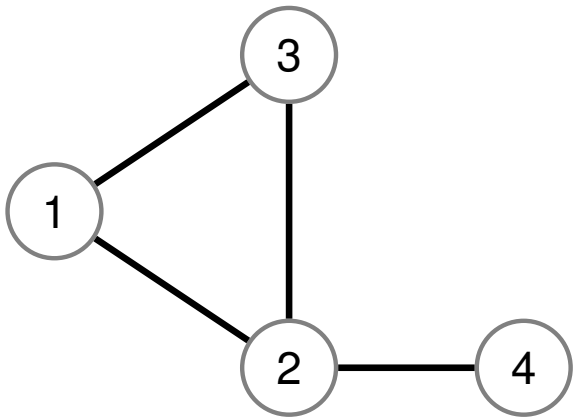
Seinosuke Toda (Gödel Prize 1998)

The entire Polynomial Hierarchy is contained in $P^{\#P}$:

$$PH \subseteq P^{\#P}$$

Outline

The SAT Encoding in Practice



$G = (V, E)$

CNF Formula Σ_G ($n = 4, k = 3$) where $X = \text{var}(\Sigma_G)$

1. At least one color per vertex:

$$(x_{1,1} \vee x_{1,2} \vee x_{1,3}) \wedge (x_{2,1} \vee x_{2,2} \vee x_{2,3}) \wedge (x_{3,1} \vee x_{3,2} \vee x_{3,3}) \wedge (x_{4,1} \vee x_{4,2} \vee x_{4,3})$$

2. At most one color per vertex (Mutual Exclusion):

$$(\neg x_{1,1} \vee \neg x_{1,2}) \wedge (\neg x_{1,1} \vee \neg x_{1,3}) \wedge (\neg x_{1,2} \vee \neg x_{1,3})$$
$$(\neg x_{2,1} \vee \neg x_{2,2}) \wedge (\neg x_{2,1} \vee \neg x_{2,3}) \wedge (\neg x_{2,2} \vee \neg x_{2,3})$$
$$(\neg x_{3,1} \vee \neg x_{3,2}) \wedge (\neg x_{3,1} \vee \neg x_{3,3}) \wedge (\neg x_{3,2} \vee \neg x_{3,3})$$
$$(\neg x_{4,1} \vee \neg x_{4,2}) \wedge (\neg x_{4,1} \vee \neg x_{4,3}) \wedge (\neg x_{4,2} \vee \neg x_{4,3})$$

3. Adjacency (Edge Constraints):

$$(\neg x_{1,1} \vee \neg x_{2,1}) \wedge (\neg x_{1,2} \vee \neg x_{2,2}) \wedge (\neg x_{1,3} \vee \neg x_{2,3}) \quad \textit{(Edge 1–2)}$$
$$(\neg x_{1,1} \vee \neg x_{3,1}) \wedge (\neg x_{1,2} \vee \neg x_{3,2}) \wedge (\neg x_{1,3} \vee \neg x_{3,3}) \quad \textit{(Edge 1–3)}$$
$$(\neg x_{2,1} \vee \neg x_{3,1}) \wedge (\neg x_{2,2} \vee \neg x_{3,2}) \wedge (\neg x_{2,3} \vee \neg x_{3,3}) \quad \textit{(Edge 2–3)}$$
$$(\neg x_{2,1} \vee \neg x_{4,1}) \wedge (\neg x_{2,2} \vee \neg x_{4,2}) \wedge (\neg x_{2,3} \vee \neg x_{4,3}) \quad \textit{(Edge 2–4)}$$

- **Variables:** $4 \times 3 = 12$ Boolean variables.
- **Clauses:** $4 + (4 \times 3) + (4 \times 3) = 28$ clauses.
- **The Goal:** Compute $\|\Sigma_G\| = |\{\mathscr{I} \in \{0,1\}^X \mid \mathscr{I} \models \Sigma_G\}|$.

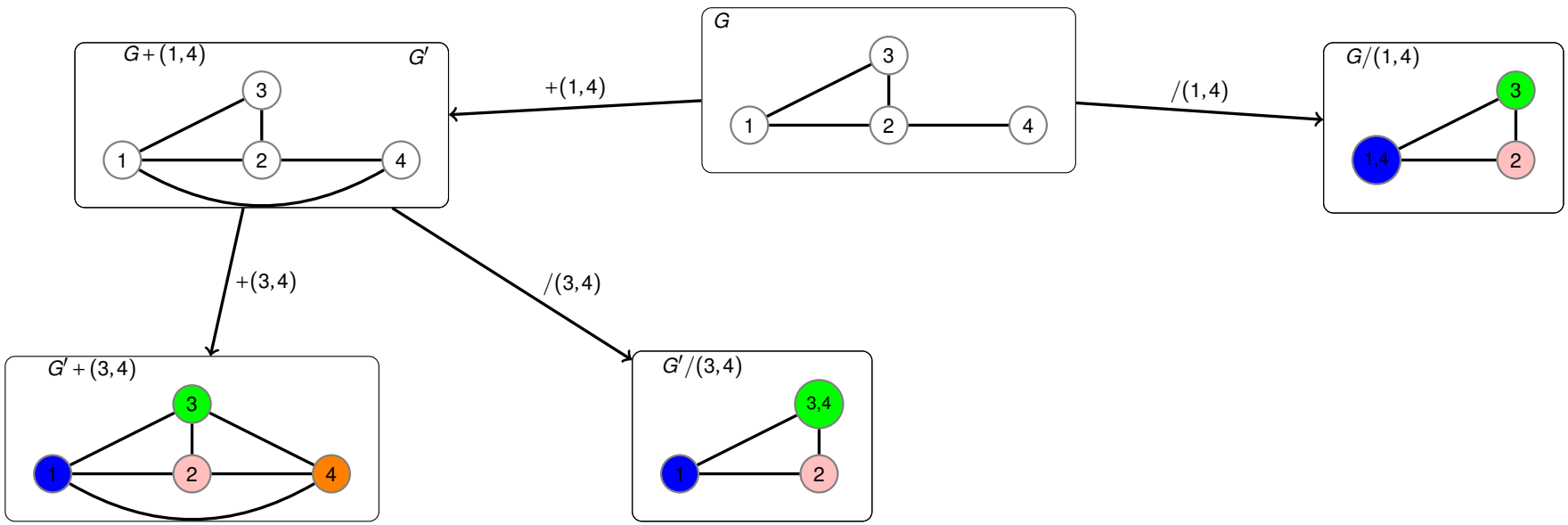
Even for this small graph, a manual count is tedious. Knowledge Compilation automates this by transforming Σ_G into a tractable form.

Zykov's deletion-contraction recurrence

Operation on the graph:

- We branch on the missing edges;
- G+(i,j) means G with the edge (i,j);
- G/(i,j) means we merge the vertices i and j in G.

Zykov's recurrence: $\chi(G) = \min\{\, \chi(G/(ij)),\ \chi(G+(ij)) \,\}\ \forall (i,j) \notin E$



Encoding Zykov's Graph Coloring in CNF (Glorian et al., 2019)

1. The Equivalence Relation (s_{ij})

$s_{ij} \equiv$ vertices i and j share the same color.

- **Edge Constraints:** $\neg s_{ij} \quad \forall (i,j) \in E$
- **Transitivity:** $(\neg s_{ij} \vee \neg s_{jk} \vee s_{ik}) \quad \forall i,j,k \in V$

2. The k-Color Constraint

Identify "color leaders" c_j who start a new color:

$$c_j \iff \bigwedge_{i<j} \neg s_{ij} \quad \text{and ensure} \quad \sum_{j=1}^{n} c_j \leq k$$

The Compilation Challenge

Encoding $\sum c_j \leq k$ into CNF requires **auxiliary variables** (e.g., via Totalizers or Adder circuits).

- We only care about the models over the original s_{ij} and c_j variables.
- **Problem:** Standard counting over the compiled circuit will be biased by these extra bits.

Outline

Outline

Projected Model Counting (∃#SAT)

The Problem: Auxiliary Variables

Modern SAT encodings Σ often use a set of variables $\mathbf{X} = \mathbf{P} \cup \mathbf{A}$:

- $\mathbf{P}$: **Primary variables** (e.g., the actual colorings x_{ik}).
- $\mathbf{A}$: **Auxiliary variables** (e.g., Tseytin variables, cardinality counters).

Definition (Projected Model Count)

Given a formula Σ over $\mathbf{P} \cup \mathbf{A}$, the projected model count over $\mathbf{P}$ is the number of assignments to $\mathbf{P}$ that can be extended to a model of Σ:

$$\|\Sigma\|_{|\mathbf{P}} = \left|\{\mathscr{I}_{\mathbf{P}} \mid \exists \mathscr{I}_{\mathbf{A}} : (\mathscr{I}_{\mathbf{P}} \cup \mathscr{I}_{\mathbf{A}}) \models \Sigma\}\right|$$

Standard #SAT
Counts every bit-string
$|\text{Models}| = 2^{|\mathbf{A}|} \times |\text{Solutions}|$
(if $\mathbf{A}$ is free/unconstrained)

$\Rightarrow$

Projected #SAT
Counts unique solutions
Ignores the "internal logic" of the encoding.

*Crucial for Knowledge Compilation: Can we compile a circuit that supports **Existential Quantification** efficiently?*

Outline

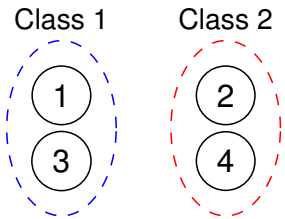
Revisiting Zykov: Partitions vs. Colorings

What are we actually counting?

Recall the Zykov variables s_{ij} (true if i, j share a color). A satisfying assignment of Σ_{Zykov} represents a **partition** of V into $m \leq k$ non-empty sets (equivalence classes).

- **The Discrepancy:** One partition into m color classes corresponds to multiple ways to assign k actual colors.

One Partition ($m = 2$)



Ways to color with $k = 3$:
Pick 2 colors out of 3, then arrange:

$$P(3,2) = \frac{3!}{(3-2)!} = \textbf{6} \text{ colorings}$$

The Problem

If a partition uses m color classes, it represents $k!/(k-m)!$ colorings. A standard #SAT count on Σ_{Zykov} treats every partition as 1, regardless of m.

Solution: Assign a **weight** to each model based on the number of color classes m.

Outline

Outline

Weighted Model Counting

Weighted CNF Formula

- Let $V = \{v_1, \ldots, v_n\}$ be a set of Boolean variables.
- Let $L = \{v_i, \overline{v}_i \mid v_i \in V\}$ be the set of literals over V.
- A wCNF formula over V is a pair (F, w) such that:
  - F is a conjunction of clauses,
  - w is a map from L to $\mathbb{Q}$ (i.e., a weighting of literals)

Weighted Model Counting

- The weight of a variable assignment $\boldsymbol{v} \in \{0,1\}^n$ is the product of weights of literals which are true in $\boldsymbol{v}$:

$$W(\boldsymbol{v}) = \prod_{\ell \in L, \boldsymbol{v} \models \ell} w(\ell)$$

- The weighted model count of a wCNF formula is the sum of weights of the models of F:

$$W(F) = \sum_{\boldsymbol{v} \in \{0,1\}^n, \boldsymbol{v} \models F} W(\boldsymbol{v})$$

Outline

Questions

Qualitative (Decision or Optimization)

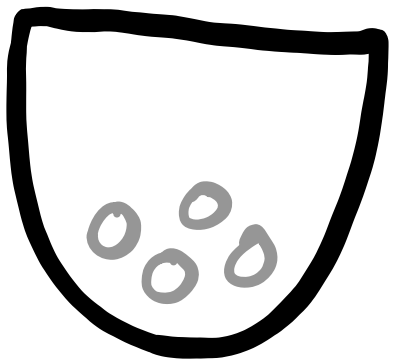
- Does the problem have a (optimal/preferred) solution?
- Output a reason for a decision.

Quantitative Questions

- How many solutions does the problem have?
- How likely is an occurrence?

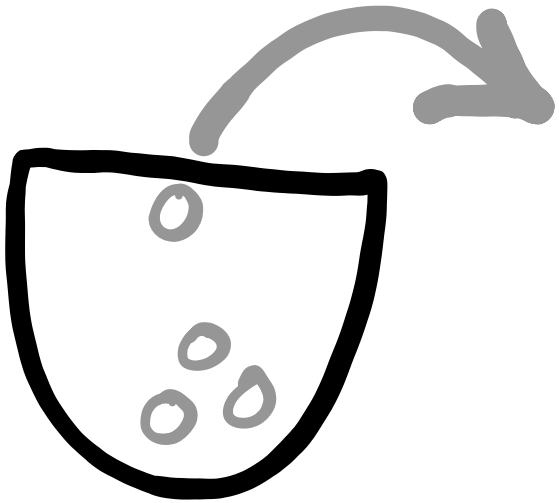
Bayesian Inference as Counting Problem

Example



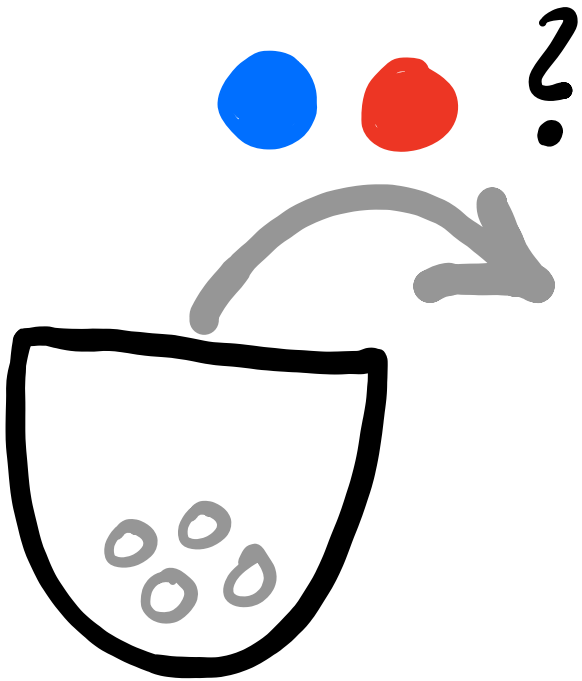
Bayesian Inference as Counting Problem

Example



Bayesian Inference as Counting Problem

Example

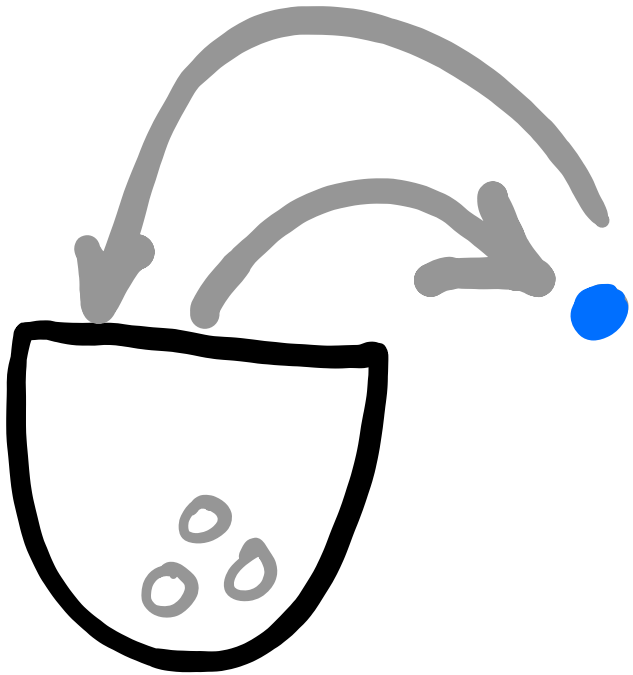


Bayesian Inference as Counting Problem

Example

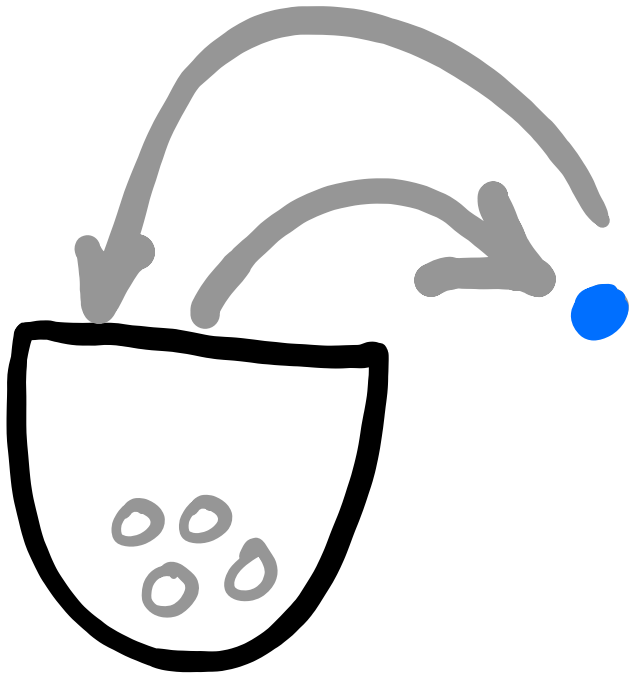
Bayesian Inference as Counting Problem

Example



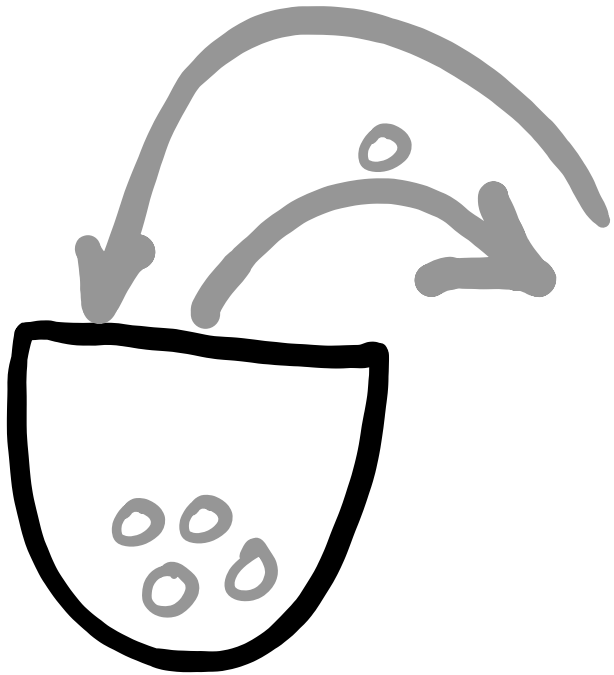
Bayesian Inference as Counting Problem

Example



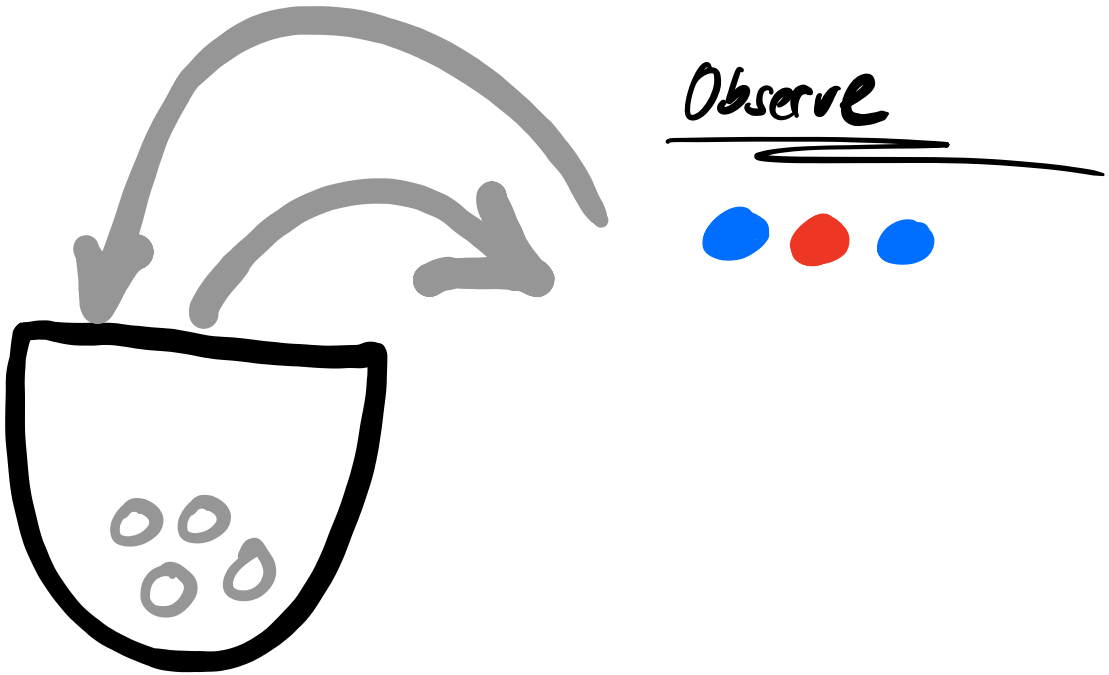
Bayesian Inference as Counting Problem

Example



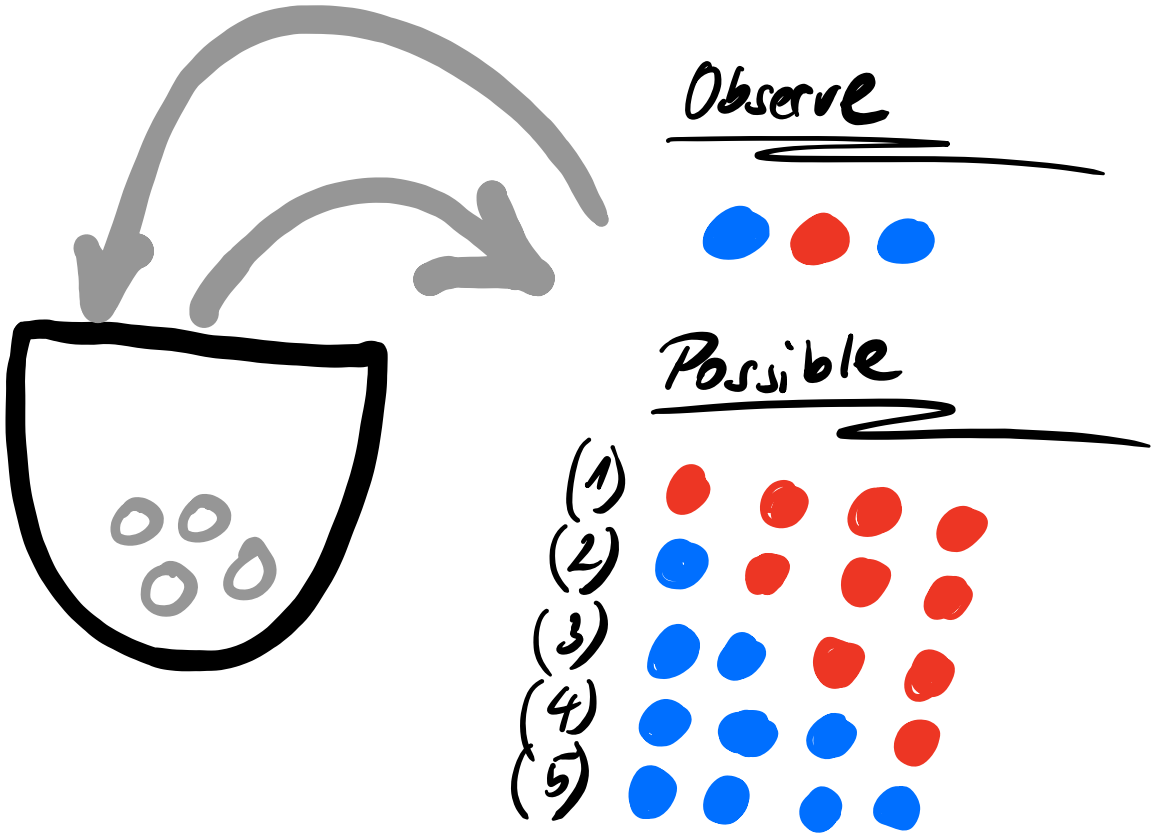
Bayesian Inference as Counting Problem

Example



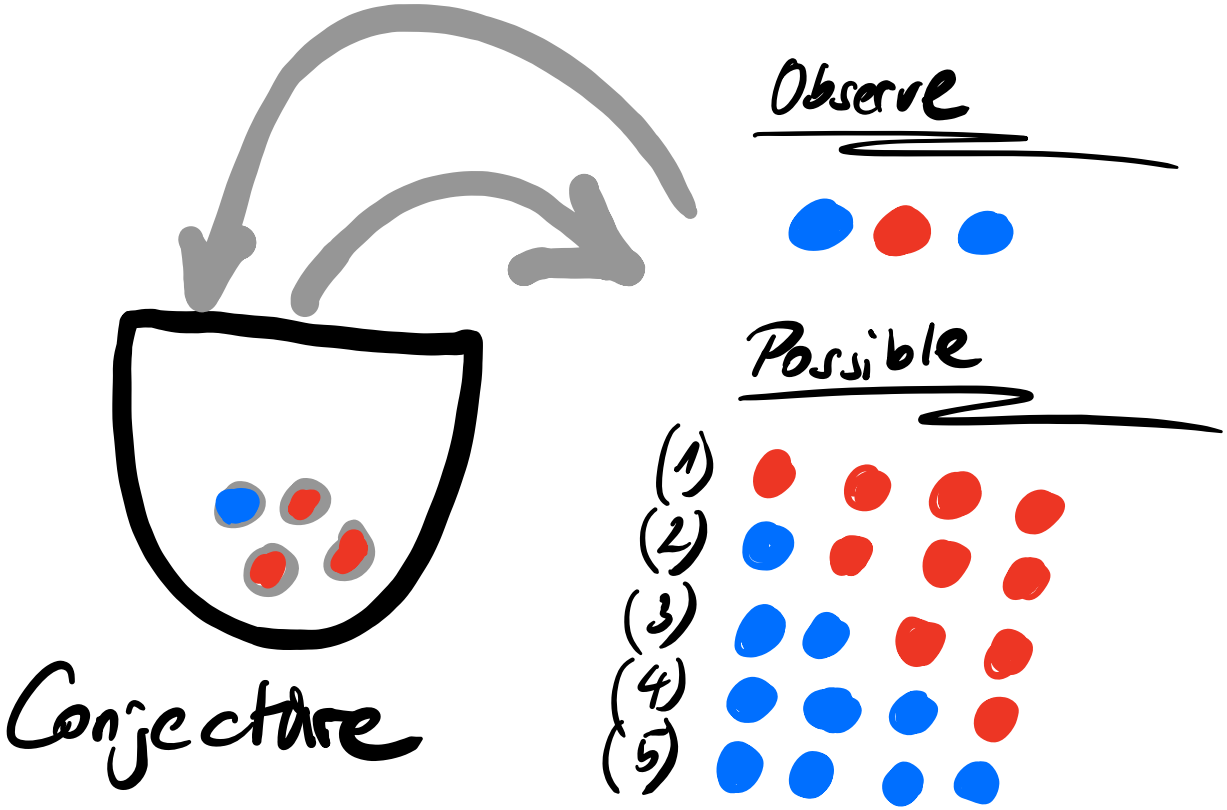
Bayesian Inference as Counting Problem

Example



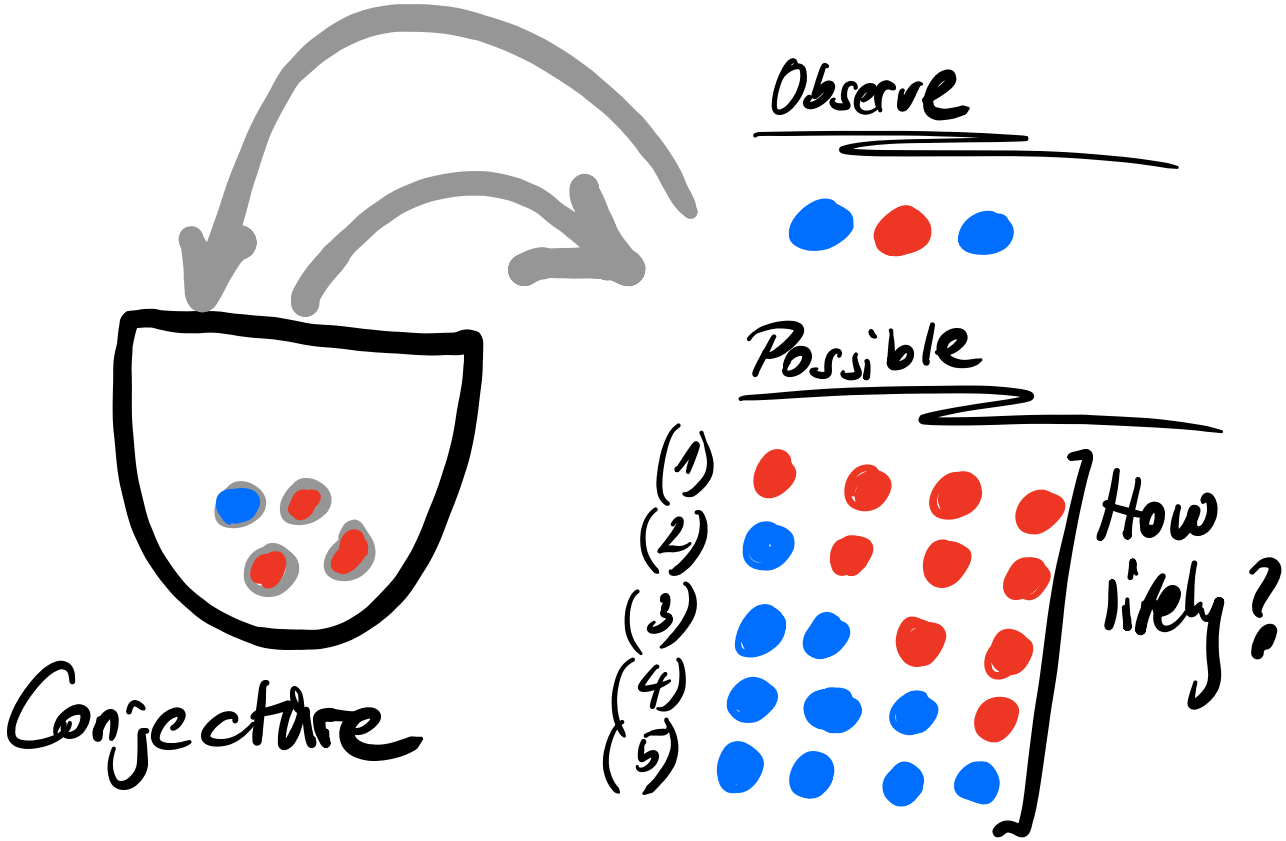
Bayesian Inference as Counting Problem

Example



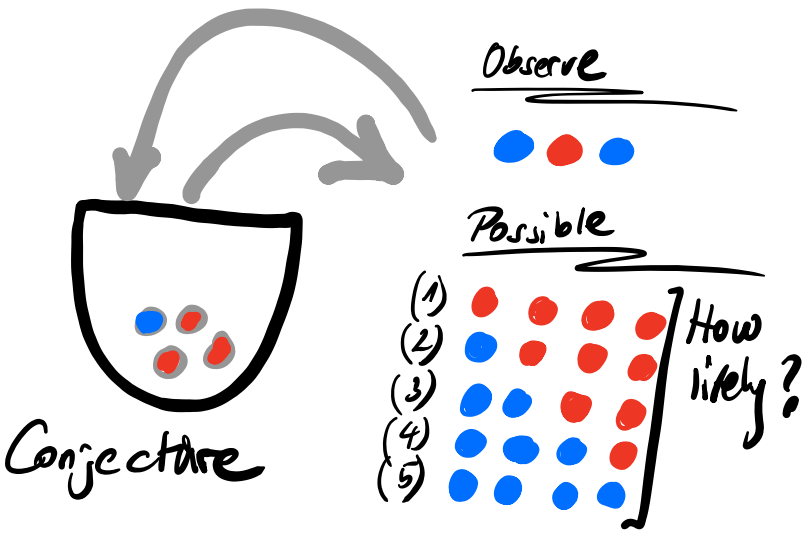
Bayesian Inference as Counting Problem

Example



Bayesian Inference as Counting Problem

Example



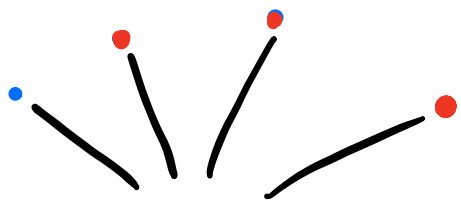
- How “likely”

⇒ Notion of Probability

- How does this relate to counting?

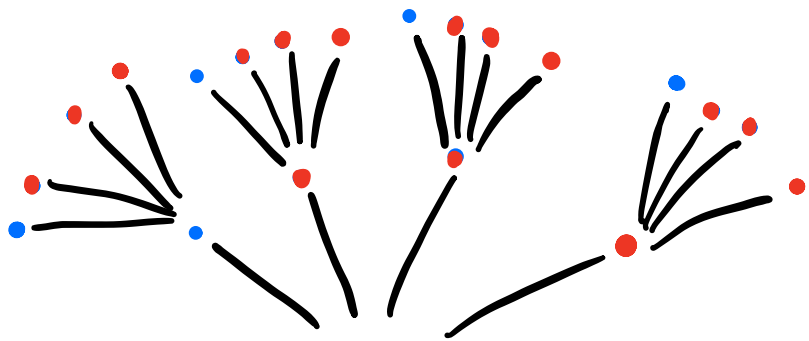
Bayesian Inference as Counting Problem

What possible combinations can we obtain?



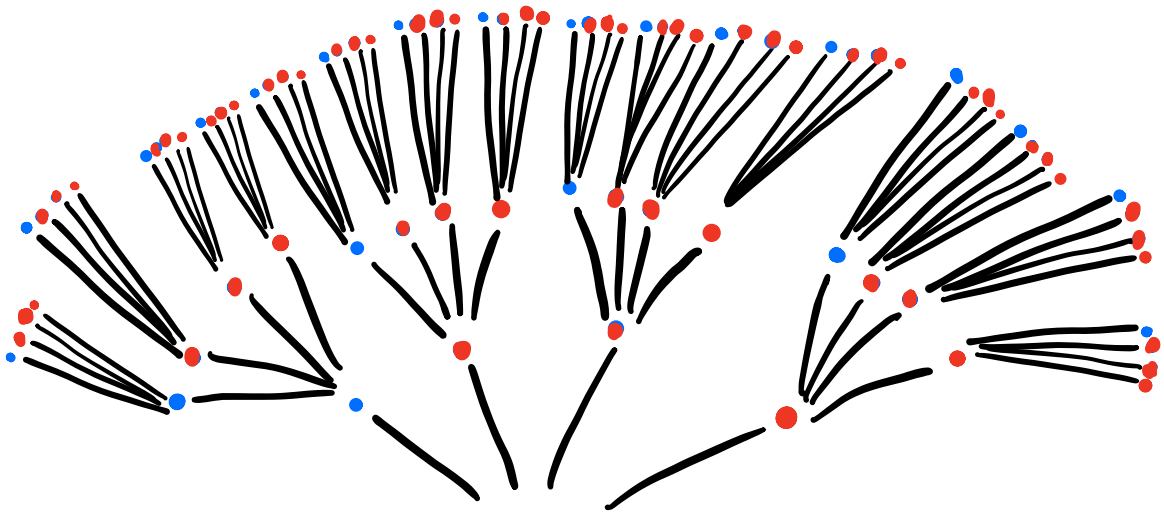
Bayesian Inference as Counting Problem

What possible combinations can we obtain?



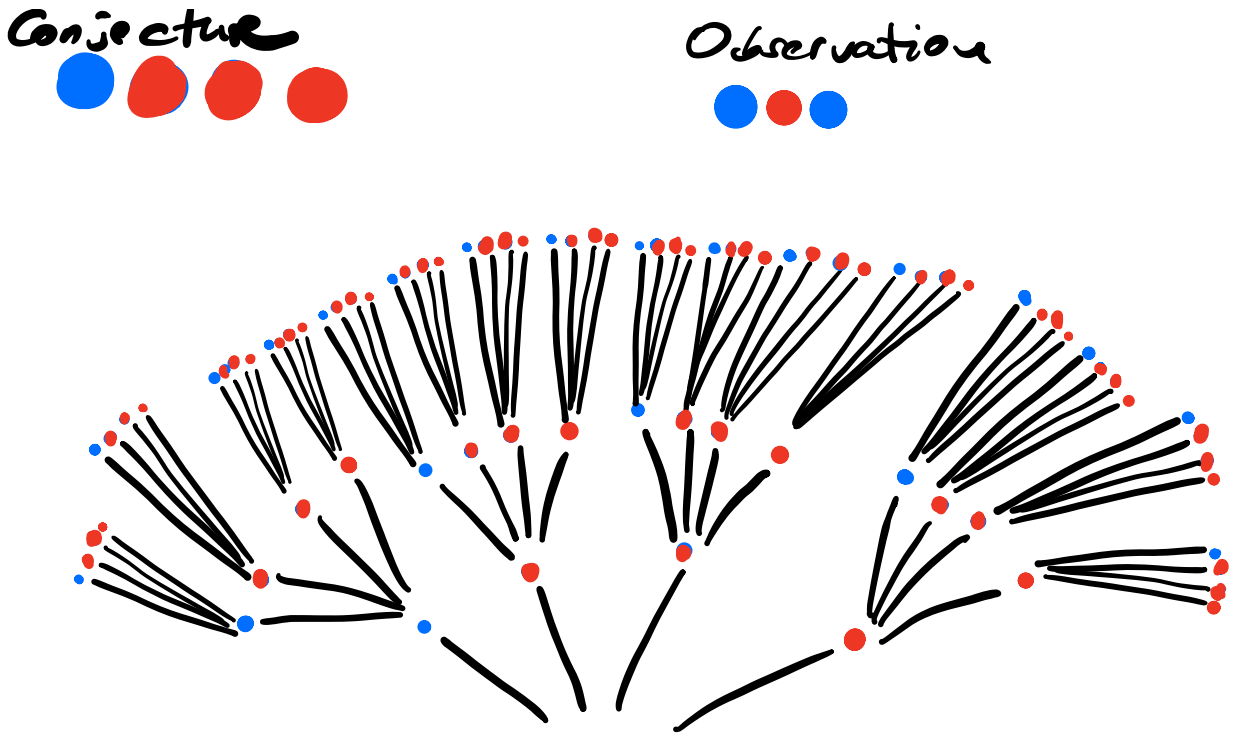
Bayesian Inference as Counting Problem

What possible combinations can we obtain?



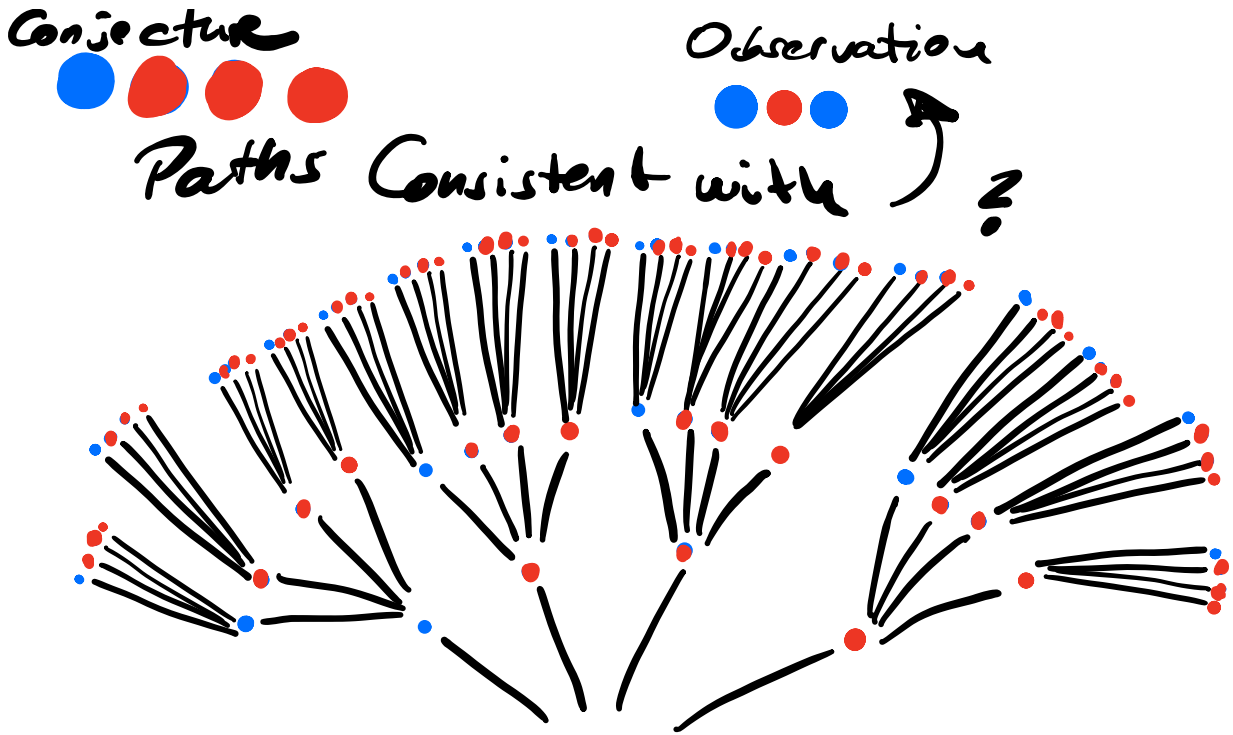
Bayesian Inference as Counting Problem

What possible combinations can we obtain?



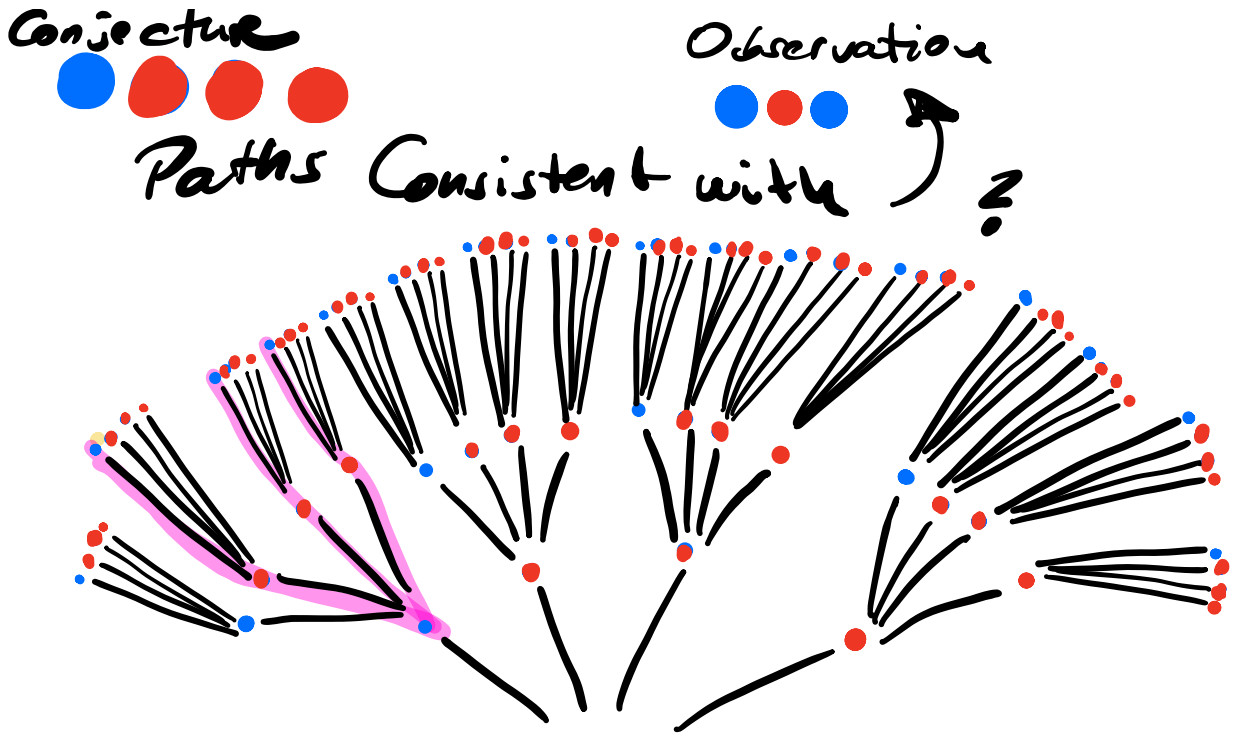
Bayesian Inference as Counting Problem

What possible combinations can we obtain?



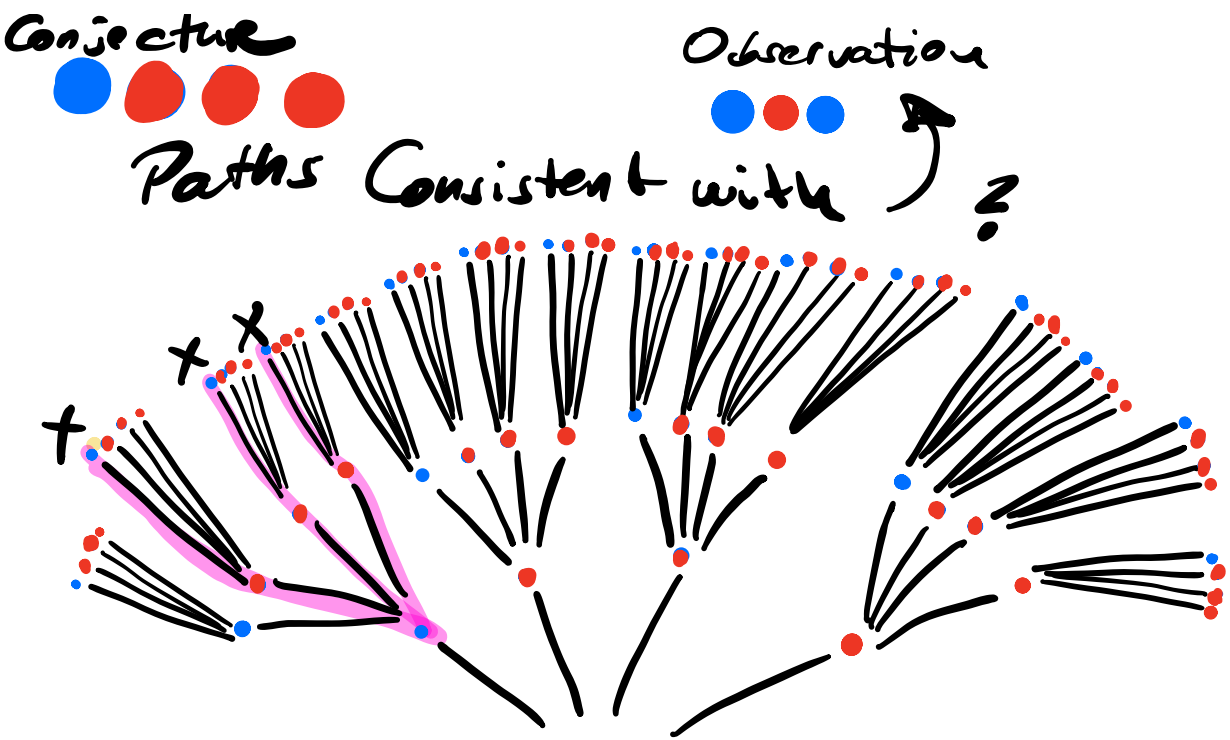
Bayesian Inference as Counting Problem

What possible combinations can we obtain?



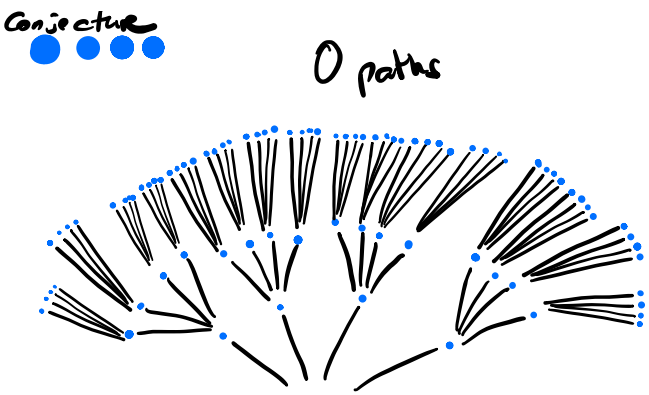
Bayesian Inference as Counting Problem

What possible combinations can we obtain?



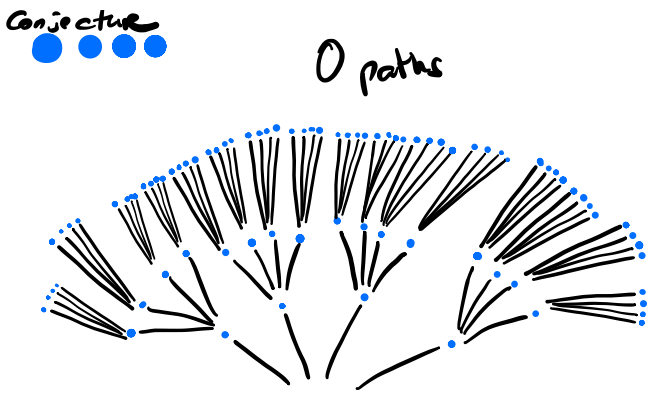
Bayesian Inference as Counting Problem

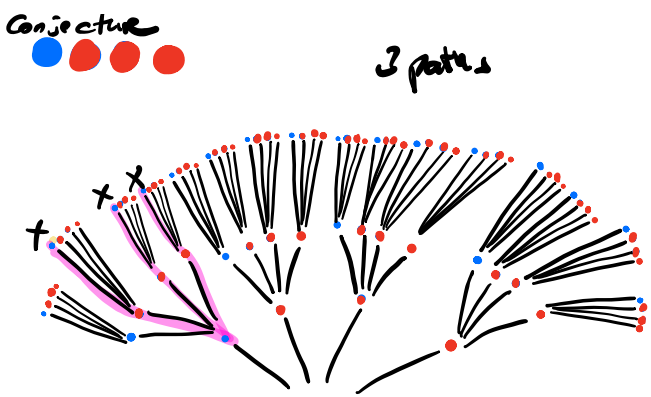
What possible combinations can we obtain?



Bayesian Inference as Counting Problem

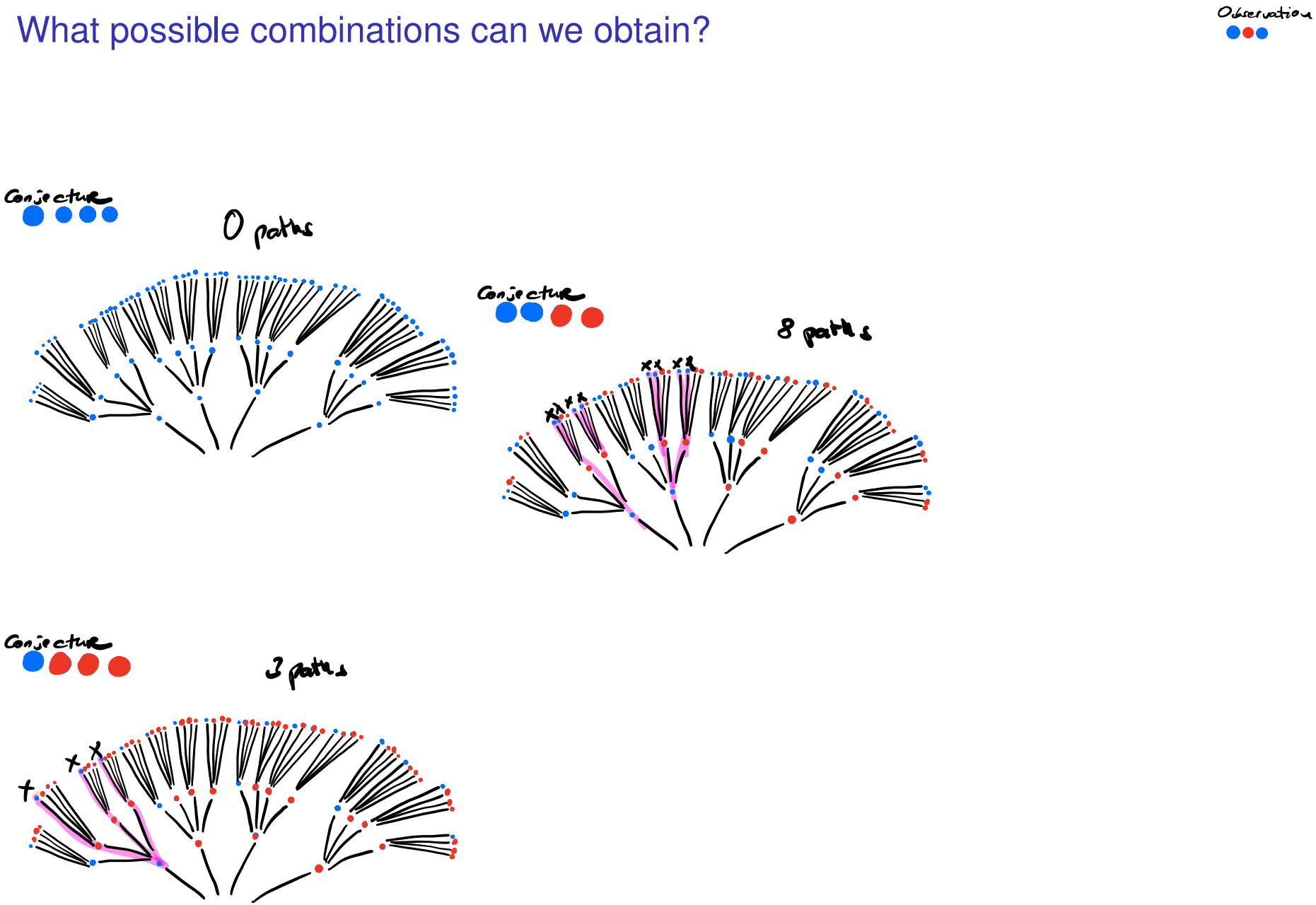
What possible combinations can we obtain?





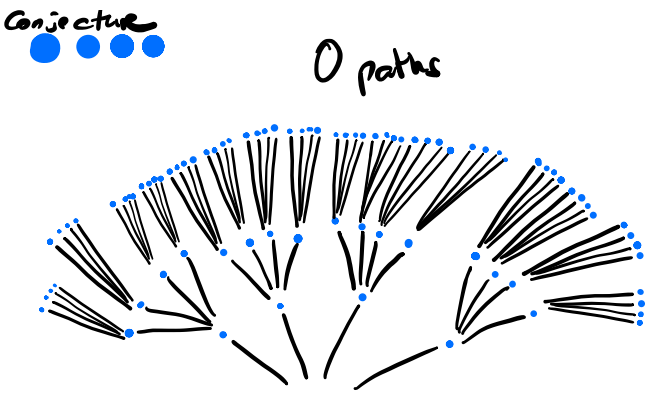
Bayesian Inference as Counting Problem

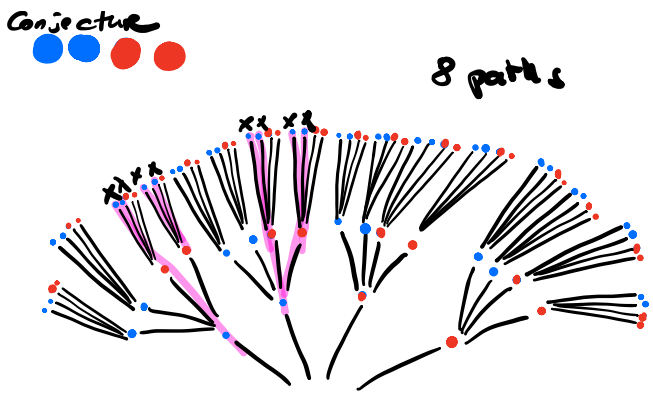
What possible combinations can we obtain?

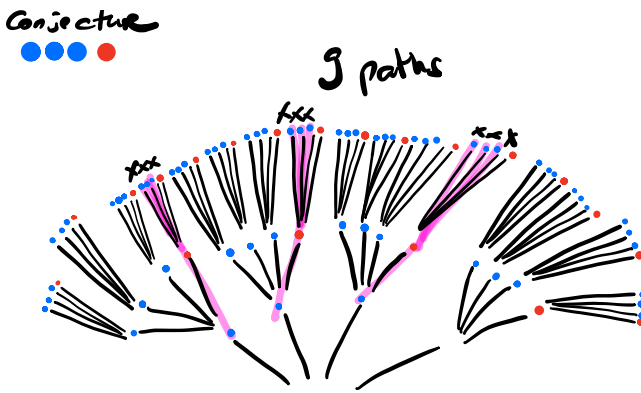


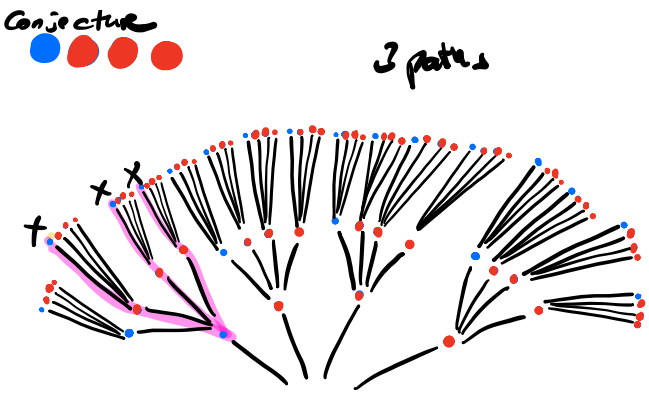
Bayesian Inference as Counting Problem

What possible combinations can we obtain?



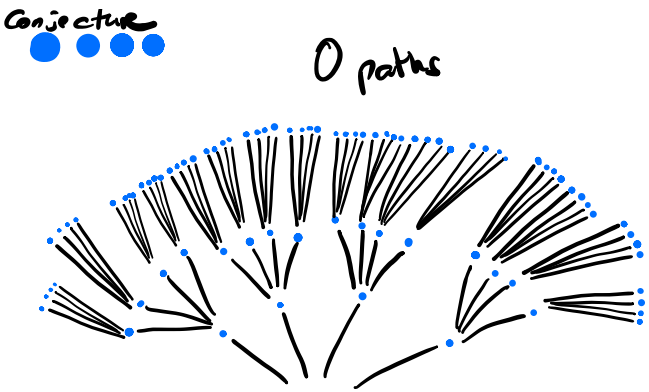


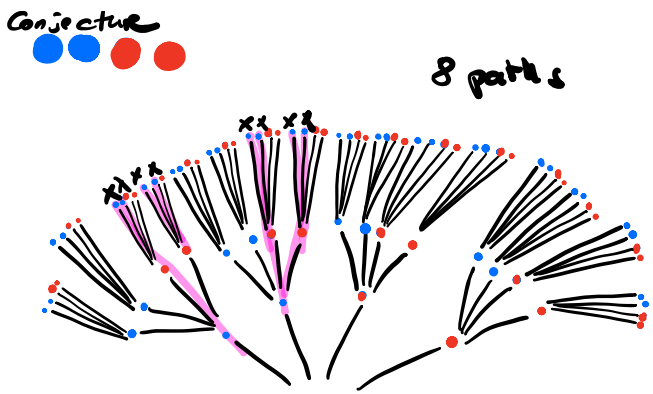


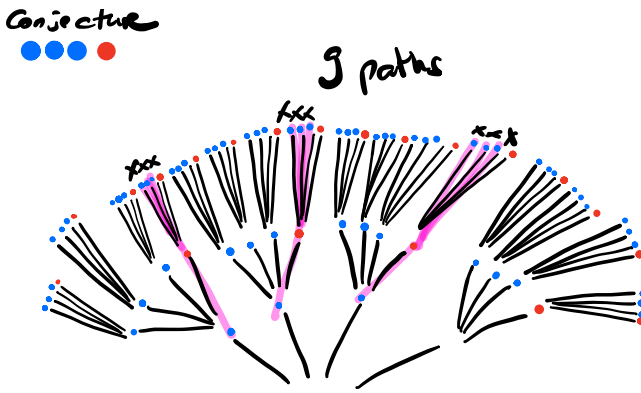


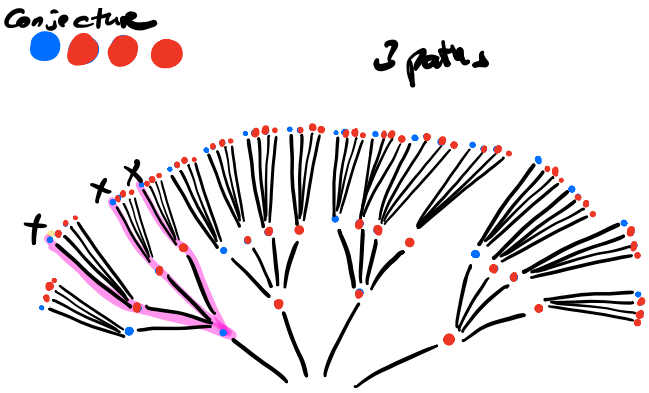
Bayesian Inference as Counting Problem

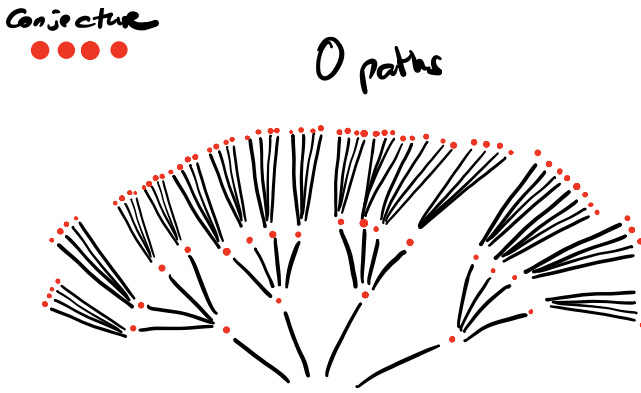
What possible combinations can we obtain?











Bayesian Inference as Counting Problem

What possible combinations can we obtain?

Conjecture	Possible Paths for ⟨●●●⟩
[●●●●]	0
[●●●●]	9
[●●●●]	8
[●●●●]	3
[●●●●]	0
Total:	20

Bayesian Inference as Counting Problem

What possible combinations can we obtain?

Conjecture	Possible Paths for ⟨●●●⟩
[●●●●]	0
[●●●●]	9
[●●●●]	8
[●●●●]	3
[●●●●]	0
Total:	20

Probability: relationship between actual and possible occurrence

Bayesian Inference as Counting Problem

What possible combinations can we obtain?

Conjecture	Possible Paths for ⟨●●●⟩
[●●●●]	0
[●●●○]	9
[●●○○]	8
[●○○○]	3
[○○○○]	0
Total:	20

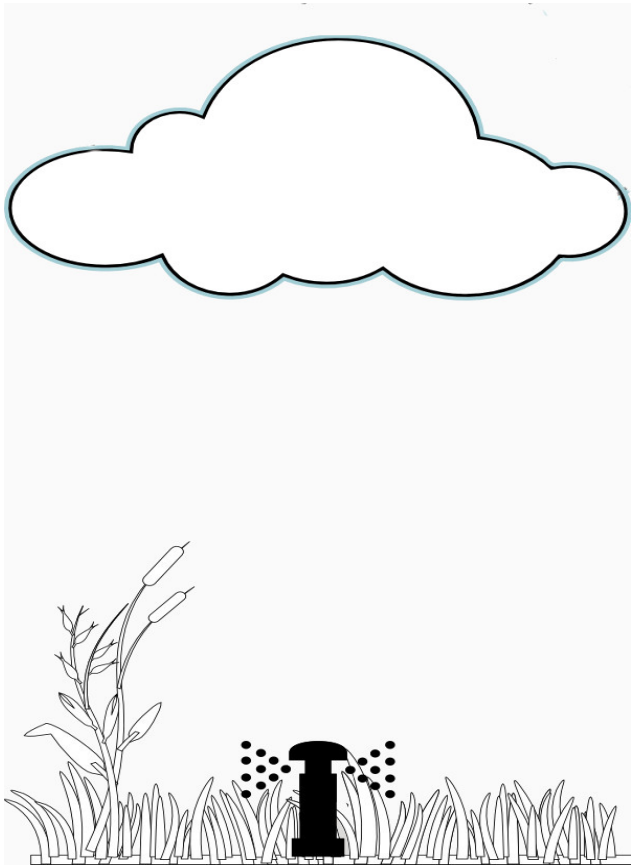
Probability [●○○○] "=" $\frac{3}{0+9+8+3+0} = \frac{3}{20}$

Bayesian Inference as Counting Problem

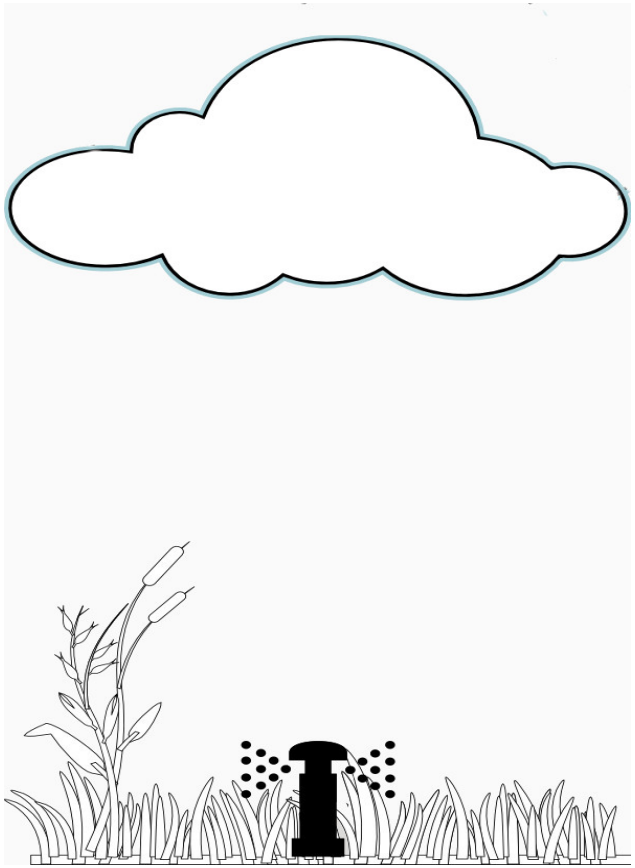
What possible combinations can we obtain?

Conjecture	Possible Paths for ⟨●●●⟩	Probability
[●●●●]	0	$0/20 = 0$
[●●●●]	9	$9/20 = 0.45$
[●●●●]	8	$8/20 = 0.40$
[●●●●]	3	$3/20 = 0.15$
[●●●●]	0	$0/20 = 0$
Total:	20	1

Probabilistic Inference

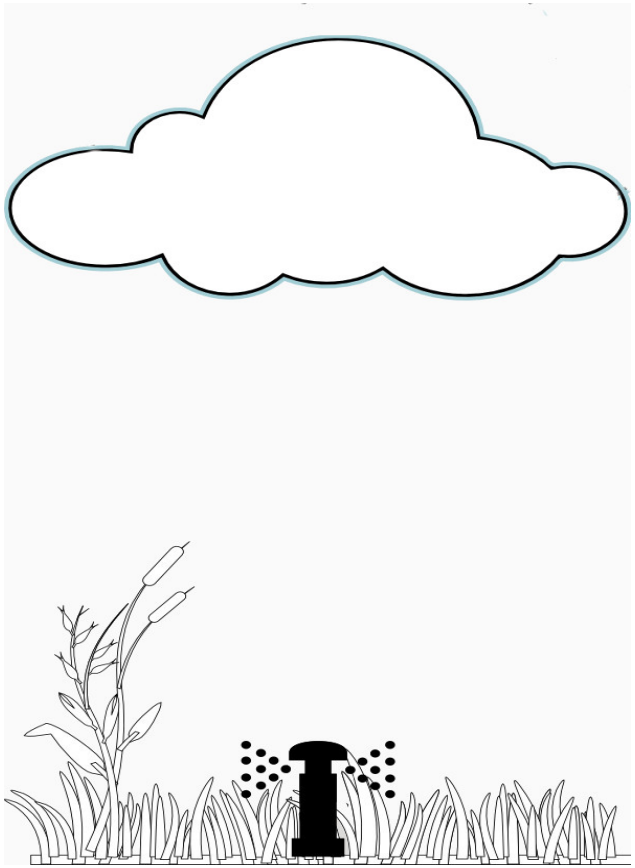


Probabilistic Inference



- Observations:
  - It is cloudy
  - The grass is wet
  - ...

Probabilistic Inference

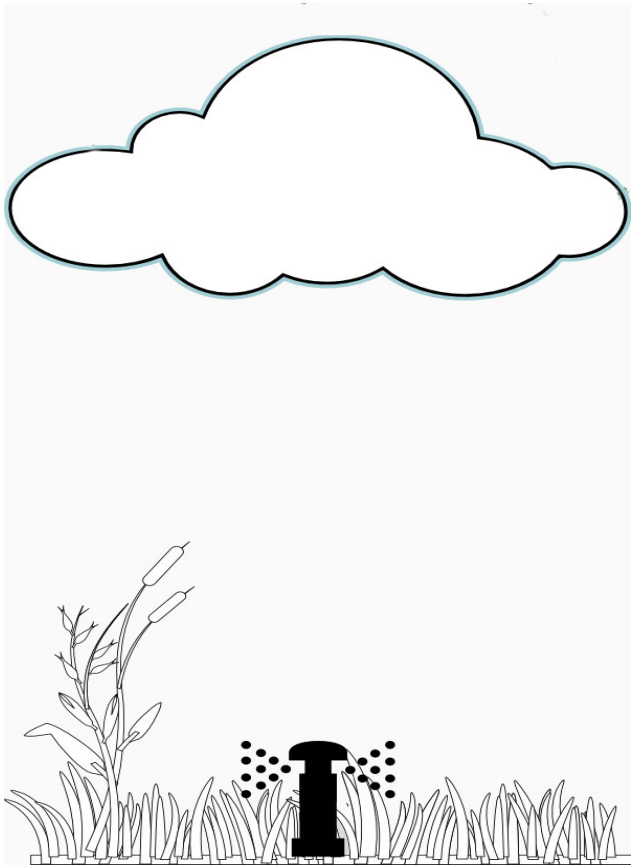


- Observations:
  - It is cloudy
  - The grass is wet
  - ...

day	cloudy	wet grass
1	1	1
2	1	1
3	1	0
4	0	0

$$\mathbb{P}(\text{wet grass}|\text{cloudy}) = \frac{2}{3}$$

Probabilistic Inference



- Observations:
  - It is cloudy
  - The grass is wet
  - ...

day	cloudy	wet grass
1	1	1
2	1	1
3	1	0
4	0	0

$$\mathbb{P}(\text{wet grass}|\text{cloudy}) = \frac{2}{3}$$

- Modeling a problem using a joint distribution on a set of random variables

Probabilistic Inference

- Let $\mathbf{X}$ be a set of random variables over finite domains, here mostly $\{0, 1\}$
- $\mathrm{Inst}(\mathbf{X})$ is the set of all complete assignments to $\mathbf{X}$
- An instantiation $\mathbf{x} \in \mathrm{Inst}(\mathbf{X})$ assigns to each variable $X_i \in X$ a value (from its domain)
- A probability mass function $\mathbb{P}$ describes probabilities of instantiations, i.e., a function $\mathbb{P} : \mathrm{Inst}(\mathbf{X}) \to [0, 1]$ and is a joint distribution over $\mathbf{X}$ if $\sum_{\mathbf{x} \in \mathrm{Inst}(\mathbf{X})} \mathbb{P}(\mathbf{x}) = 1$
- Evidence is a partial instantiation, an assignment to only some variables of $\mathbf{X}$

Cloudy	**R**ain	**S**prinkler	**W**et	$\mathbb{P}(C, R, S, W)$
0	0	0	0	0.225
0	0	0	1	0
0	0	1	0	0.045
0	0	1	1	0.18
0	1	0	0	0.0025
0	1	0	1	0.0225
0	1	1	0	0
0	1	1	1	0.025
1	0	0	0	0.09
1	0	0	1	0
1	0	1	0	0.002
1	0	1	1	0.008
1	1	0	0	0.036
1	1	0	1	0.324
1	1	1	0	0
1	1	1	1	0.04

Probabilistic Inference

- Let **X** be a set of random variables over finite domains, here mostly $\{0, 1\}$
- $\mathrm{Inst}(\mathbf{X})$ is the set of all complete assignments to **X**
- An instantiation $\mathbf{x} \in \mathrm{Inst}(\mathbf{X})$ assigns to each variable $X_i \in X$ a value (from its domain)
- A probability mass function $\mathbb{P}$ describes probabilities of instantiations, i.e., a function $\mathbb{P} : \mathrm{Inst}(\mathbf{X}) \to [0, 1]$ and is a joint distribution over **X** if $\sum_{\mathbf{x} \in \mathrm{Inst}(\mathbf{X})} \mathbb{P}(\mathbf{x}) = 1$
- Evidence is a partial instantiation, an assignment to only some variables of **X**

Cloudy	**R**ain	**S**prinkler	**W**et	$\mathbb{P}(C, R, S, W)$
0	0	0	0	0.225
0	0	0	1	0
0	0	1	0	0.045
0	0	1	1	0.18
0	1	0	0	0.0025
0	1	0	1	0.0225
0	1	1	0	0
0	1	1	1	0.025
1	0	0	0	0.09
1	0	0	1	0
1	0	1	0	0.002
1	0	1	1	0.008
1	1	0	0	0.036
1	1	0	1	0.324
1	1	1	0	0
1	1	1	1	0.04

- probability of instantiation **x**

$$\mathbb{P}(C = 0, R = 1, S = 0, W = 1) = 0.0225$$
$$\mathbb{P}(C_0, R_1, S_0, W_1) = 0.0225$$

Probabilistic Inference

- Let **X** be a set of random variables over finite domains, here mostly $\{0,1\}$
- $\mathrm{Inst}(\mathbf{X})$ is the set of all complete assignments to **X**
- An instantiation $\mathbf{x} \in \mathrm{Inst}(\mathbf{X})$ assigns to each variable $X_i \in X$ a value (from its domain)
- A probability mass function $\mathbb{P}$ describes probabilities of instantiations, i.e., a function $\mathbb{P} : \mathrm{Inst}(\mathbf{X}) \to [0,1]$ and is a joint distribution over **X** if $\sum_{\mathbf{x} \in \mathrm{Inst}(\mathbf{X})} \mathbb{P}(\mathbf{x}) = 1$
- Evidence is a partial instantiation, an assignment to only some variables of **X**

Cloudy	**R**ain	**S**prinkler	**W**et	$\mathbb{P}(C,R,S,W)$
0	0	0	0	0.225
0	0	0	1	0
0	0	1	0	0.045
0	0	1	1	0.18
0	1	0	0	0.0025
0	1	0	1	0.0225
0	1	1	0	0
0	1	1	1	0.025
1	0	0	0	0.09
1	0	0	1	0
1	0	1	0	0.002
1	0	1	1	0.008
1	1	0	0	0.036
1	1	0	1	0.324
1	1	1	0	0
1	1	1	1	0.04

- probability of instantiation **x**

$$\mathbb{P}(C=0,R=1,S=0,W=1) = 0.0225$$
$$\mathbb{P}(C_0,R_1,S_0,W_1) = 0.0225$$

- probability of evidence e

$$\mathbb{P}(S_0,W_1) = 0+0.0225+0+0.324$$
$$= 0.3465$$

Probabilistic Inference

- Let $\mathbf{X}$ be a set of random variables over finite domains, here mostly $\{0, 1\}$
- $\mathrm{Inst}(\mathbf{X})$ is the set of all complete assignments to $\mathbf{X}$
- An instantiation $\mathbf{x} \in \mathrm{Inst}(\mathbf{X})$ assigns to each variable $X_i \in X$ a value (from its domain)
- A probability mass function $\mathbb{P}$ describes probabilities of instantiations, i.e., a function $\mathbb{P} : \mathrm{Inst}(\mathbf{X}) \to [0, 1]$ and is a joint distribution over $\mathbf{X}$ if $\sum_{\mathbf{x} \in \mathrm{Inst}(\mathbf{X})} \mathbb{P}(\mathbf{x}) = 1$
- Evidence is a partial instantiation, an assignment to only some variables of $\mathbf{X}$

Cloudy	**R**ain	**S**prinkler	**W**et	$\mathbb{P}(C, R, S, W)$
0	0	0	0	0.225
0	0	0	1	0
0	0	1	0	0.045
0	0	1	1	0.18
0	1	0	0	0.0025
0	1	0	1	0.0225
0	1	1	0	0
0	1	1	1	0.025
1	0	0	0	0.09
1	0	0	1	0
1	0	1	0	0.002
1	0	1	1	0.008
1	1	0	0	0.036
1	1	0	1	0.324
1	1	1	0	0
1	1	1	1	0.04

- probability of instantiation $\mathbf{x}$

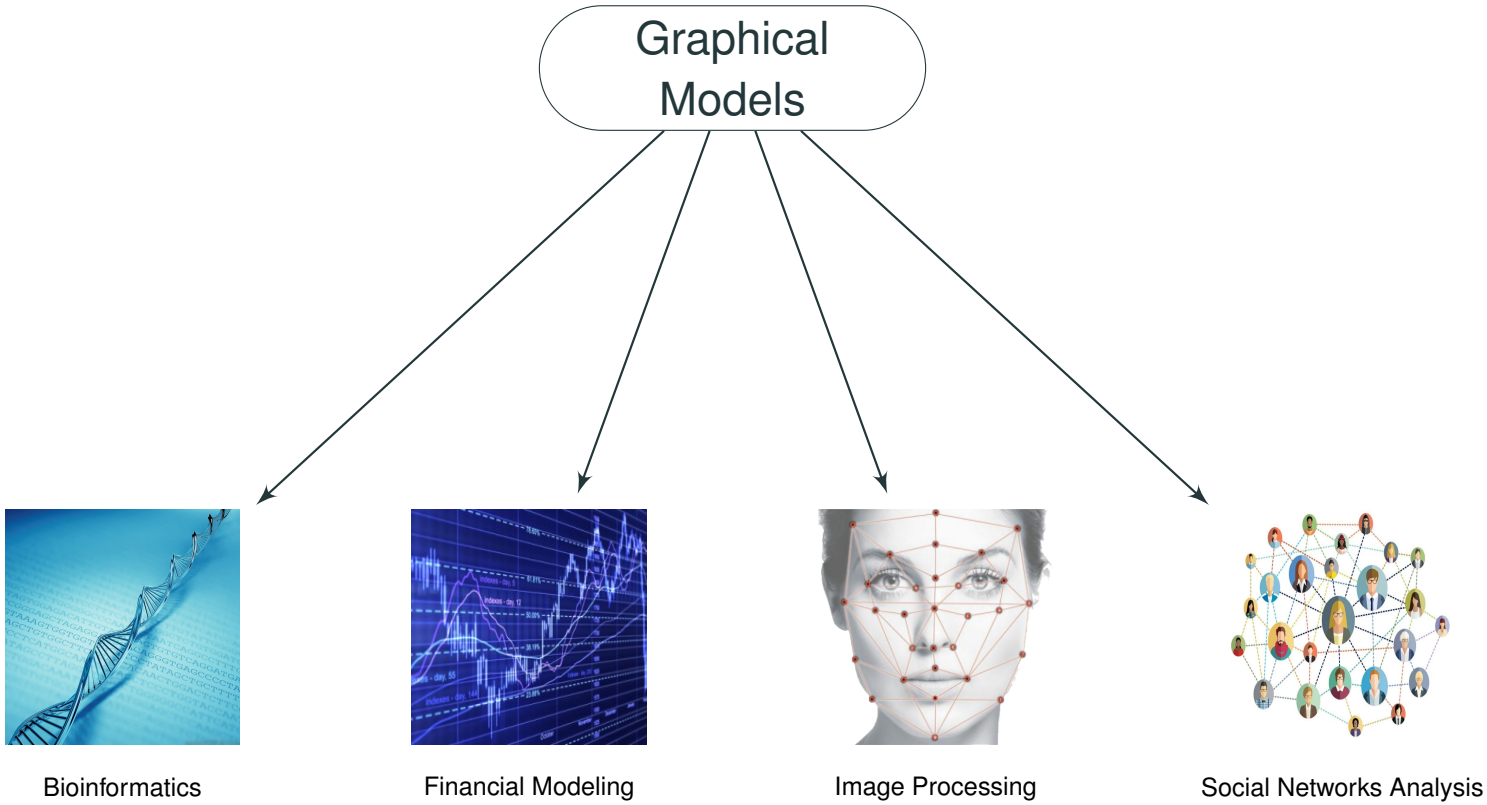
$$\mathbb{P}(C = 0, R = 1, S = 0, W = 1) = 0.0225$$
$$\mathbb{P}(C_0, R_1, S_0, W_1) = 0.0225$$

- probability of evidence e

$$\mathbb{P}(S_0, W_1) = 0 + 0.0225 + 0 + 0.324$$
$$= 0.3465$$

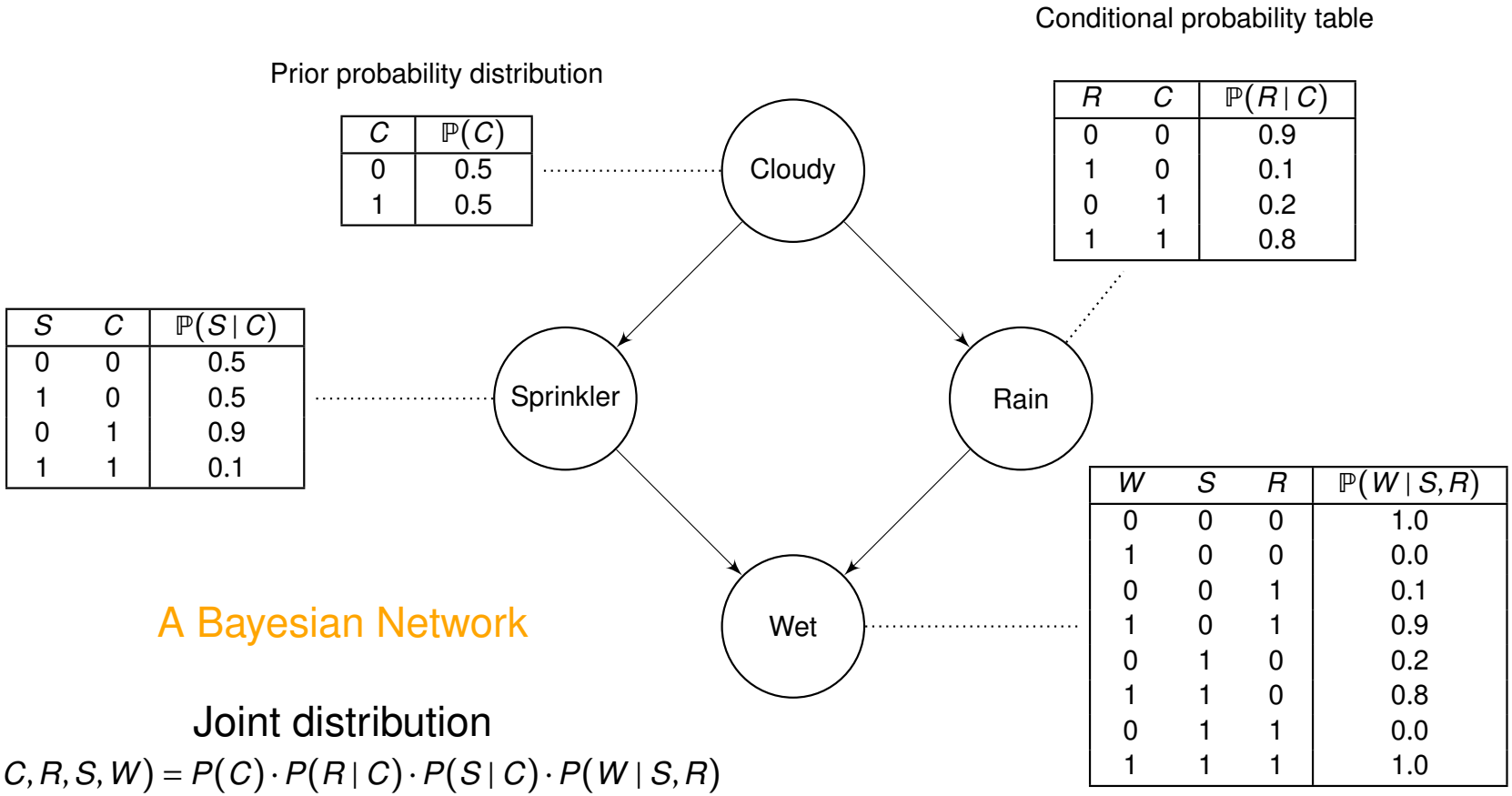
- The size of the joint distribution is exponential in the number of variables!

Graphical Models



Graphical models are a well-studied framework for representing high-dimensional probability distributions, with a wide spectrum of applications.

Graphical Models



C	$\mathbb{P}(C)$
0	0.5
1	0.5

R	C	$\mathbb{P}(R \mid C)$
0	0	0.9
1	0	0.1
0	1	0.2
1	1	0.8

S	C	$\mathbb{P}(S \mid C)$
0	0	0.5
1	0	0.5
0	1	0.9
1	1	0.1

W	S	R	$\mathbb{P}(W \mid S,R)$
0	0	0	1.0
1	0	0	0.0
0	0	1	0.1
1	0	1	0.9
0	1	0	0.2
1	1	0	0.8
0	1	1	0.0
1	1	1	1.0

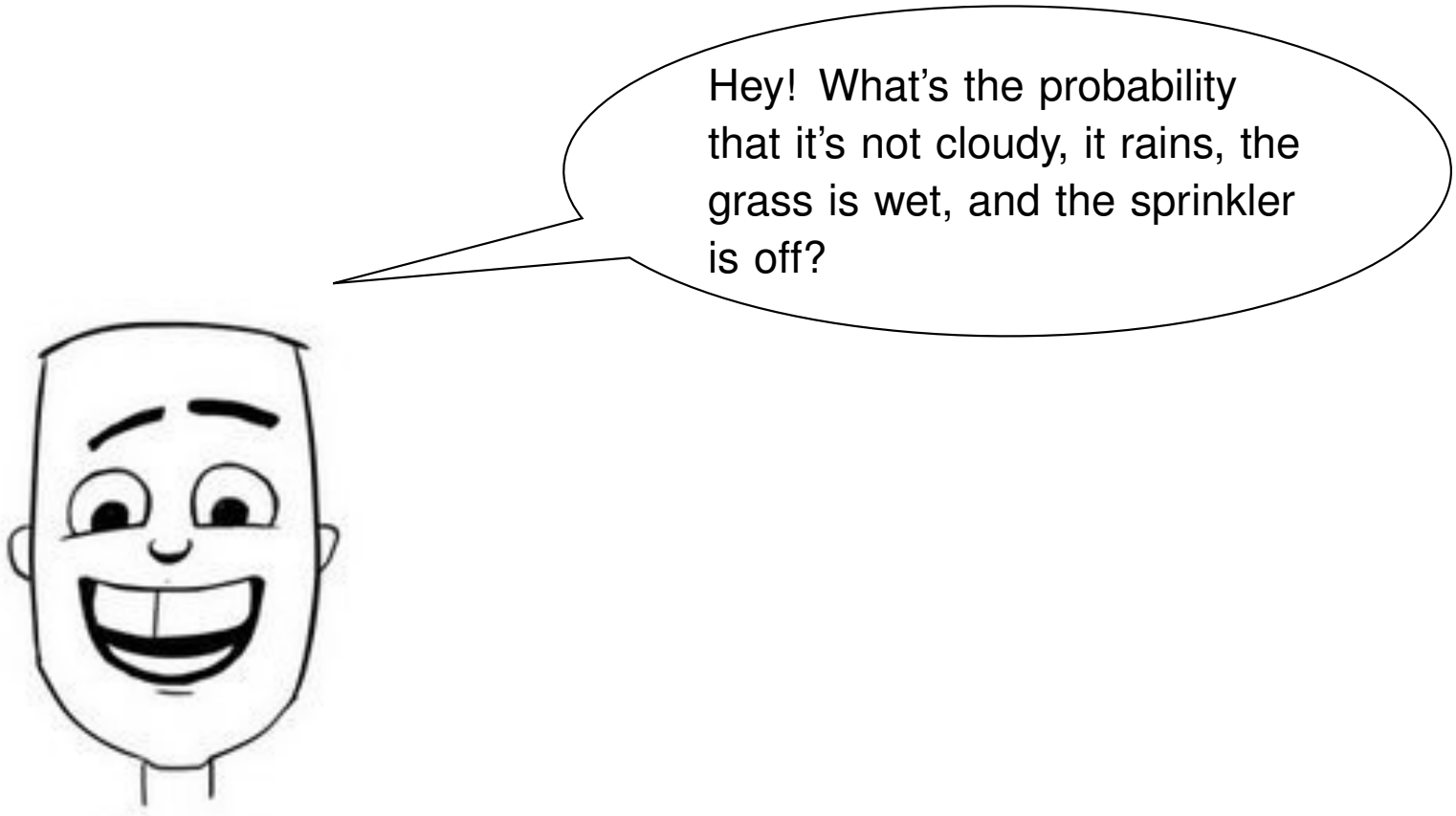
Joint distribution

$$P(C,R,S,W) = P(C) \cdot P(R \mid C) \cdot P(S \mid C) \cdot P(W \mid S,R)$$

Graphical models are commonly separated into:

- directed models (a.k.a Bayesian networks), with conditional probability tables (CPTs) over a directed (acyclic) graph,
- undirected models (a.k.a. Markov networks), with energy functions over the cliques of an undirected graph.

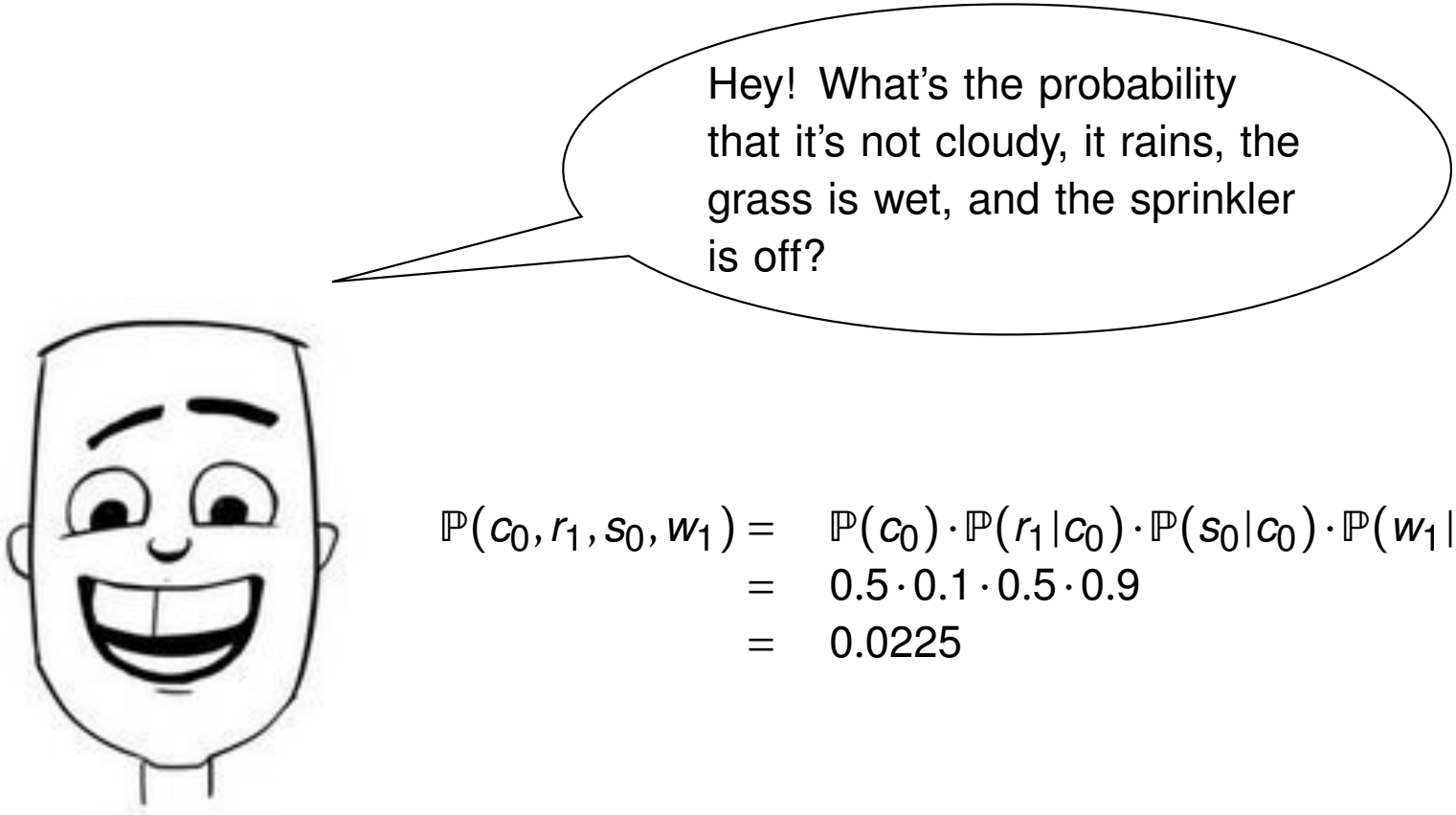
Bayesian network



The semantics of Bayesian networks implies the following joint distribution:

$$\mathbb{P}(x_1, x_2, \ldots, x_n) = \prod_i \mathbb{P}(x_i | u_i)$$

Bayesian network

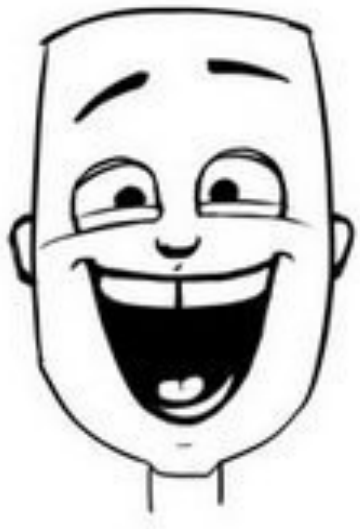


$$\begin{aligned}\mathbb{P}(c_0, r_1, s_0, w_1) &= \mathbb{P}(c_0) \cdot \mathbb{P}(r_1|c_0) \cdot \mathbb{P}(s_0|c_0) \cdot \mathbb{P}(w_1|s_0, r_1) \\ &= 0.5 \cdot 0.1 \cdot 0.5 \cdot 0.9 \\ &= 0.0225\end{aligned}$$

The semantics of Bayesian networks implies the following joint distribution:

$$\mathbb{P}(x_1, x_2, \ldots, x_n) = \prod_i \mathbb{P}(x_i|u_i)$$

Probabilistic Inference



Product rule: $\mathbb{P}(A, X) = \mathbb{P}(A|X) \cdot \mathbb{P}(X)$

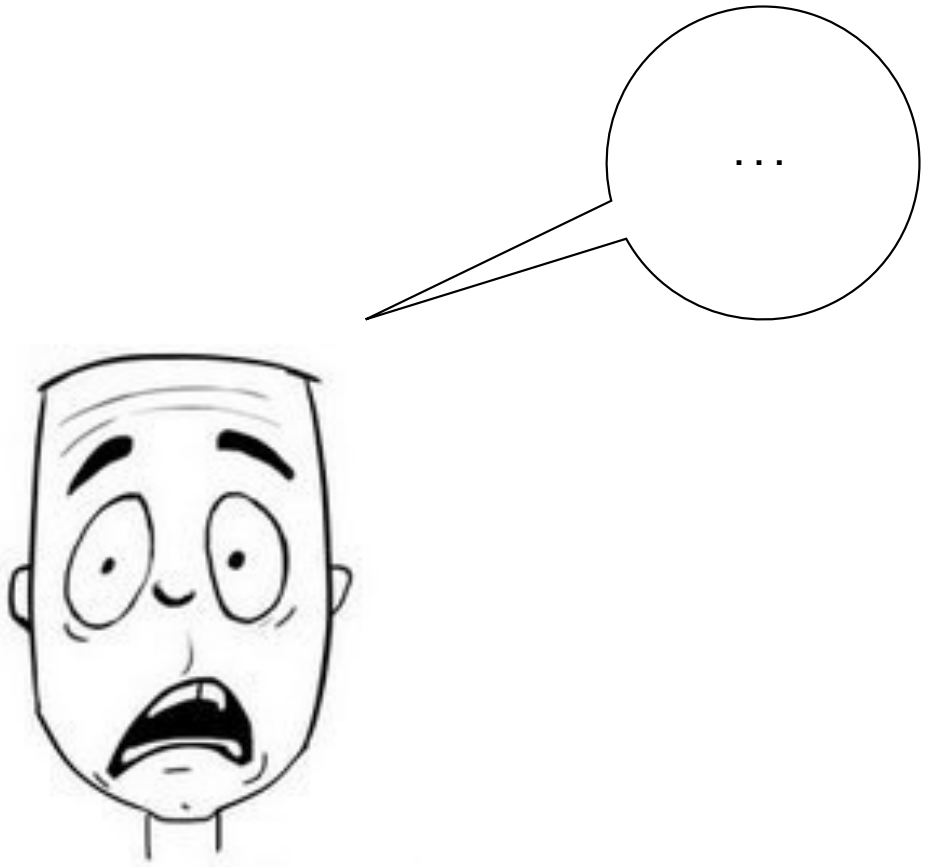
$$\mathbb{P}(w_1 \mid c_1) = \frac{\mathbb{P}(w_1, c_1)}{\mathbb{P}(c_1)}$$

Bayes' rule: $\mathbb{P}(A|X) = \frac{\mathbb{P}(X|A) \cdot \mathbb{P}(A)}{\mathbb{P}(X)}$

$$\mathbb{P}(c_1 \mid s_1) = \frac{\mathbb{P}(s_1|c_1) \cdot \mathbb{P}(c_1)}{\mathbb{P}(s_1)}$$

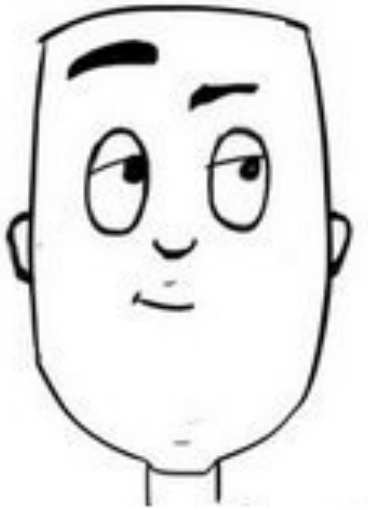
Probabilistic inference is a basic task for handling more complex queries:

- Conditional probabilities: evaluating the probability of an event $\boldsymbol{y}$ given an evidence $\boldsymbol{x}$;
- Most probable explanations: given an evidence $\boldsymbol{x}$, find an assignment $\boldsymbol{y}$ over the complementary variables that maximizes $\mathbb{P}(\boldsymbol{y} \mid \boldsymbol{x})$;
- ...



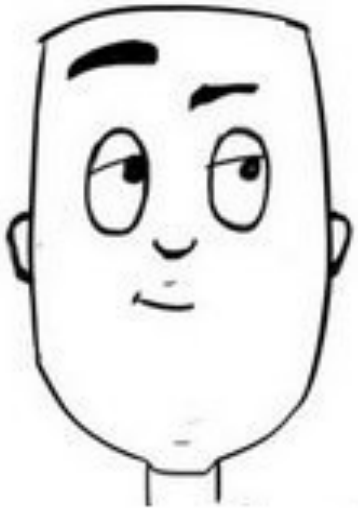
Unfortunately, probabilistic inference is #**P-complete** in graphical models, including both Bayesian networks and Markov networks.

Probabilistic Inference to Weighted Model Counting



So . . . Can we **reduce** the problem of probabilistic inference to Weighted Model Counting?

Probabilistic Inference to Weighted Model Counting



So . . . Can we **reduce** the problem of probabilistic inference to Weighted Model Counting?

Yes! The idea is to find a translation function τ mapping

- any graphical model (Bayes or Markov net) N to a weighted CNF formula $\tau(N)$, and
- any partial assignment $\boldsymbol{x}$ to a term $\tau(\boldsymbol{x})$,

such that

$$\mathbb{P}_N(\boldsymbol{x}) = \frac{W[\tau(N) \wedge \tau(x)]}{W[\tau(N)]}$$

where $W[\tau(N)]$ is the **partition constant** ($= 1$ for BNs).