Model Counting for Logical Theories

Monday

Dmitry Chistikov Rayna Dimitrova

Department of Computer Science
University of Oxford, UK

Max Planck Institute for Software Systems (MPI-SWS)
Kaiserslautern and Saarbrücken, Germany

ESSLLI 2016

How do we count?

How do we count the elements of a set given as a linked list?

How do we count?

How do we count the elements of a set given as a linked list?

What if the set is not given explicitly, but we can only check if an element from a given universe is in this set or not?

How do we count?

How do we count the elements of a set given as a linked list?

What if the set is not given explicitly, but we can only check if an element from a given universe is in this set or not?

What can we say about the size of a set if we are able to check this and other implicitly given sets for emptiness?

How do we count?

How do we count the elements of a set given as a linked list?

What if the set is not given explicitly, but we can only check if an element from a given universe is in this set or not?

What can we say about the size of a set if we are able to check this and other implicitly given sets for emptiness?

Counting the number of arrangements

A summer school offers 6 courses, each with one lecture per day.

Day 1	Day 2	...
Course 1	Course 1	...
Course 2	Course 2	...
...	...	...
Course 6	Course 6	...

Ada wants to attend some subset of lectures per day so that:

(1) she takes at least one lecture per day,

(2) she rests between each two lectures, and

(3) she takes at most 3 lectures per day.

For how many days should the school last, so that Ada can try out all arrangements that meet her constraints?

Counting the number of arrangements

We focus on the constraints for an arbitrary day of the school.
We can model Ada's choice with Boolean variables $x_1, x_2, \ldots, x_6$

$$\begin{aligned} x_i = \mathsf{T} &\quad : \quad \text{attend lecture } i \\ x_i = \mathsf{F} &\quad : \quad \text{do not attend lecture } i \end{aligned}$$

and express her constraints in a logical form.

(1) She takes at least one lecture per day.

$$x_1 \text{ or } x_2 \text{ or } x_3 \text{ or } x_4 \text{ or } x_5 \text{ or } x_6$$

(2) She rests between each two lectures.

$$(\neg x_1 \text{ or } \neg x_2) \text{ and } (\neg x_2 \text{ or } \neg x_3) \text{ and } \ldots$$

(3) She takes at most 3 lectures per day.

$$\big((x_1 \text{ and } x_2 \text{ and } x_3) \to (\neg x_4 \text{ and } \neg x_5 \text{ and } \neg x_6)\big) \text{ and } \ldots$$

Counting the number of arrangements

We focus on the constraints for an arbitrary day of the school.
We can model Ada's choice with Boolean variables $x_1, x_2, \ldots, x_6$

$$\begin{aligned} x_i = \mathsf{T} \quad &: \quad \text{attend lecture } i \\ x_i = \mathsf{F} \quad &: \quad \text{do not attend lecture } i \end{aligned}$$

and express her constraints in a logical form.

(1) She takes at least one lecture per day.

$$x_1 \vee x_2 \vee x_3 \vee x_4 \vee x_5 \vee x_6$$

(2) She rests between each two lectures.

$$(\neg x_1 \vee \neg x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge \ldots$$

(3) She takes at most 3 lectures per day.

$$\big((x_1 \wedge x_2 \wedge x_3) \rightarrow (\neg x_4 \wedge \neg x_5 \wedge \neg x_6)\big) \wedge \ldots$$

Counting the number of arrangements

We focus on the constraints for an arbitrary day of the school.
We can model Ada's choice with Boolean variables $x_1, x_2, \ldots, x_6$

$$\begin{aligned} x_i = \mathsf{T} &\quad : \quad \text{attend lecture } i \\ x_i = \mathsf{F} &\quad : \quad \text{do not attend lecture } i \end{aligned}$$

and express her constraints in a logical form.

The number of days the school should last is equal to the number of truth assignments to $x_1, x_2, \ldots, x_6$ that satisfy the constraints.

We can compute this number by counting the satisfying assignments.

Counting the number of arrangements

We focus on the constraints for an arbitrary day of the school.
We can model Ada's choice with Boolean variables $x_1, x_2, \ldots, x_6$

$$\begin{aligned} x_i = \mathsf{T} \quad &: \quad \text{attend lecture } i \\ x_i = \mathsf{F} \quad &: \quad \text{do not attend lecture } i \end{aligned}$$

and express her constraints in a logical form.

Now, suppose that the school lasts the computed number of days. We want to know how many evenings can Ada go out partying, if course i makes her tired for the evening with probability p_i.

$$\begin{aligned} \text{weight for } x_i = \mathsf{T} \quad &: \quad w_i(\mathsf{T}) = 1 - p_i \\ \text{weight for } x_i = \mathsf{F} \quad &: \quad w_i(\mathsf{F}) = 1 \end{aligned}$$

The weight of a truth assignment $(a_1, \ldots, a_6)$ is $\prod_{i=1}^{6} w_i(a_i)$.

Counting the number of arrangements

We focus on the constraints for an arbitrary day of the school.
We can model Ada's choice with Boolean variables $x_1, x_2, \ldots, x_6$

$$\begin{array}{lcl} x_i = \mathsf{T} & : & \text{attend lecture } i \\ x_i = \mathsf{F} & : & \text{do not attend lecture } i \end{array}$$

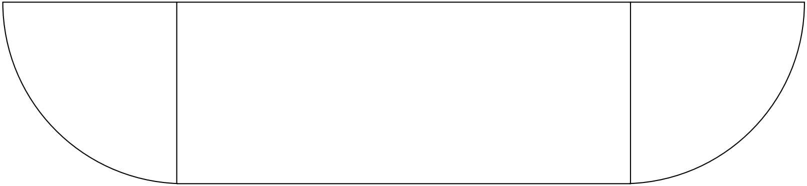
and express her constraints in a logical form.

Now, suppose that the school lasts the computed number of days.
We want to know how many evenings can Ada to go out partying, if course i makes her tired for the evening with probability p_i.

The expected number of nights that Ada can party is the weighted count of the assignments satisfying the constraints.

Counting integral points

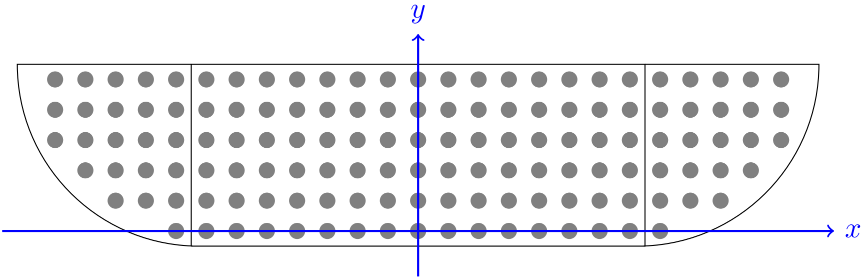
A summer school lecture hall has the following shape.



If the chairs are arranged in a grid-like fashion at a given distance, what is the number of students that can attend a lecture?

Counting integral points

A summer school lecture hall has the following shape.

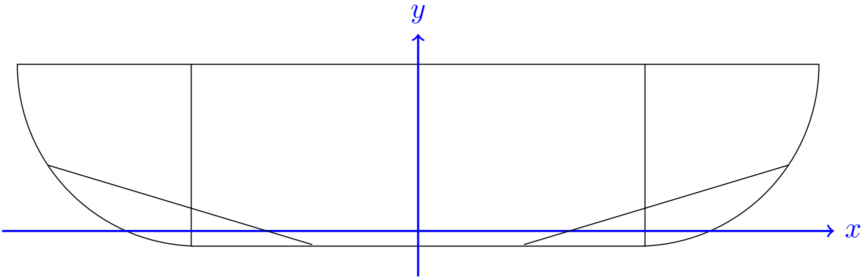


We write down the definition of the shape as a set of constraints and count the "integral points" that satisfy these constraints.

$$y \geq -0.5 \wedge y \leq 5.5 \quad \wedge \quad \begin{aligned}\big((x \geq -7.5 \wedge x \leq 7.5) \vee \\ (x-7.5)^2 + (y-5.5)^2 \leq 5.5^2 \vee \\ (x+7.5)^2 + (y-5.5)^2 \leq 5.5^2\big)\end{aligned}$$

Computing the area of a shape

A summer school lecture hall has the following shape.



For estimating the costs of maintenance, we might be interested in computing the area of the frequently used parts of the lecture hall.

Add constraints $y \geq ax - b$ and $y \geq -ax - b$. Area?

Model counting

counting discrete objects

$$(x_1 \vee x_2 \vee x_3 \vee x_4 \vee x_5 \vee x_6) \wedge \dots$$

$x_1, \dots, x_6$ are Boolean

counting integral points

$$y \geq -0.5 \wedge y \leq 5.5 \wedge \big((x \geq -7.5 \wedge x \leq 7.5) \vee \dots\big)$$

x, y are integer

computing the volume of a body

$$y \geq -0.5 \wedge y \leq 5.5 \wedge \big((x \geq -7.5 \wedge x \leq 7.5) \vee \dots\big)$$

x, y are real

How do we count?

How do we count the elements of a set given as a linked list?

What if the set is not given explicitly, but you can only check if an element from a given universe is in this set or not?

What can we say about the size of a set if we are able to check this and other implicitly given sets for emptiness?

Model counting

counting discrete objects

$$(x_1 \vee x_2 \vee x_3 \vee x_4 \vee x_5 \vee x_6) \wedge \dots$$

$x_1, \dots, x_6$ are Boolean

counting integral points

$$y \geq -0.5 \wedge y \leq 5.5 \wedge \big((x \geq -7.5 \wedge x \leq 7.5) \vee \dots\big)$$

x, y are integer

computing the volume of a body

$$y \geq -0.5 \wedge y \leq 5.5 \wedge \big((x \geq -7.5 \wedge x \leq 7.5) \vee \dots\big)$$

x, y are real

$$\begin{tikzcd}[column sep=huge, row sep=huge]
\begin{array}{c}\mathrm{SAT}\\ \textsf{satisfiability}\end{array} \arrow[r] \arrow[d] & \begin{array}{c}\mathrm{SMT}\\ \textsf{satisfiability}\\ \textsf{modulo theories}\end{array} \arrow[d] \\
\begin{array}{c}\#\mathrm{SAT}\\ \textsf{model counting}\end{array} \arrow[r] & \begin{array}{c}\#\mathrm{SMT}\\ \textsf{model counting}\\ \textsf{modulo theories}\end{array}
\end{tikzcd}$$

SAT satisfiability $\longrightarrow$ SMT satisfiability modulo theories

SAT: Propositional logic

Propositional logic: a language of propositional formulas

Syntax

- $x_1, x_2, x_3 \ldots$ Boolean variables
- $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ logical connectives
- a set of rules for constructing formulas

Semantics: gives meaning to formulas

negation

x	$\neg x$
T	F
F	T

conjunction

x	y	$x \wedge y$
T	T	T
T	F	F
F	T	F
F	F	F

disjunction

x	y	$x \vee y$
T	T	T
T	F	T
F	T	T
F	F	F

SAT: Boolean satisfiability

Model: a variable assignment for which the formula evaluates to T

Formula $\varphi \equiv (x \vee y) \wedge (\neg x \vee \neg y)$
Models of φ: $[\![\varphi]\!] = \{(\mathsf{T}, \mathsf{F}), (\mathsf{F}, \mathsf{T})\}$

Satisfiability (SAT): Given a formula φ, does φ have a model?

Determining satisfiability via truth tables requires examining 2^n assignments, where n is the number of propositional variables.

SAT is an $\mathbf{NP}$-complete problem: all problems in the class $\mathbf{NP}$ can be solved by translating them to SAT (in polynomial time).

SAT solving in practice

Modern SAT solvers are very fast most of the time

- for details, check the annual SAT competitions

and have an enormous number of applications:

- scheduling, creation of time-tables
- chip design, hardware verification
- program synthesis, network design
- ...

SMT

Problems are often modelled at a higher level than Boolean logic. Translation to SAT is expensive and loses modelling insights.

Satisfiability Modulo Theories (SMT): reason about satisfiability at the higher level of abstraction provided by first-order logic.

SMT: First-order logic

Logical symbols

- $x_1, x_2, x_3 \ldots$ variables
- $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ logical connectives
- $\exists, \forall$ quantifiers

Non-logical symbols

- constant symbols: 42, Ada, $\frac{1}{4}$
- predicate symbols: $x > y$, $attends(x, y)$
- function symbols: $attendees(x)$

Example formula

$$\forall x \big(student(x) \rightarrow \exists y.\ course(y) \wedge attends(x, y) \wedge attendees(y) > 2 \big)$$

SMT: First-order logic

To define the semantics of a first-order formula, we need to give meaning to the constant, predicate and function symbols.

Theory: a (possibly infinite) set of logical formulas

Commonly used theories come with a fixed set of symbols and their standard interpretation: integer arithmetic, real arithmetic.

SMT: Integer Arithmetic (IA)

Syntax

- constant symbols 0 and 1
- function symbols $+, -, \cdot$
- predicate symbol $\leq$
- equality

Semantics is defined in the structure $\langle \mathbb{Z}, +, -, \cdot, \leq \rangle$

Example formulas

$$even(x) \quad : \quad \exists y.\, x = y + y$$

$$\forall x \forall y \forall z.\, x^3 + y^3 = z^3 \rightarrow (x = 0 \vee y = 0 \vee z = 0),$$

where x^3 is a shortcut for $x \cdot x \cdot x$

SMT: Integer Arithmetic (IA)

Syntax

- constant symbols 0 and 1
- function symbols $+, -, \cdot$
- predicate symbol $\leq$
- equality

Semantics is defined in the structure $\langle \mathbb{Z}, +, -, \cdot, \leq \rangle$

With multiplication, checking satisfiability is undecidable.

If the variable domains are bounded, then satisfiability is decidable.

SMT: Real Arithmetic (RA)

Syntax

- constant symbols 0 and 1
- function symbols $+, -, \cdot$
- predicate symbol $\leq$
- equality

Semantics is defined in the structure $\langle \mathbb{R}, +, -, \cdot, \leq \rangle$

Example formula

$$\exists x.\ x > 1 \land x \cdot x - x - 1 = 0$$

SMT: Real Arithmetic (RA)

Syntax

- constant symbols 0 and 1
- function symbols $+, -, \cdot$
- predicate symbol $\leq$
- equality

Semantics is defined in the structure $\langle \mathbb{R}, +, -, \cdot, \leq \rangle$

Linear fragment

- extend the set of constant symbols with the computable reals
- restrict $\cdot$ so that at least one argument is a constant

SMT solving in practice

State-of-the-art SMT solvers (Z3, CVC4, ...) are widely used in software verification and synthesis and in test case generation.

Annual competition (SMT-COMP), workshop, summer school.

Model counting for logical theories

counting discrete objects: propositional logic, integer arithmetic

$$(x_1 \vee x_2 \vee x_3 \vee x_4 \vee x_5 \vee x_6) \wedge \dots$$

$x_1, \dots, x_6$ Boolean

counting integral points: real and integer arithmetic

$$y \geq -0.5 \wedge y \leq 5.5 \wedge \big((x \geq -7.5 \wedge x \leq 7.5) \vee \dots\big)$$

x, y are integer

volume computation: real arithmetic

$$y \geq -0.5 \wedge y \leq 5.5 \wedge \big((x \geq -7.5 \wedge x \leq 7.5) \vee \dots\big)$$

x, y are real

From logical theories to measured logical theories

$$\begin{array}{c}\text{SAT}\\ \text{satisfiability}\end{array} \longrightarrow \begin{array}{c}\#\text{SAT}\\ \text{model counting}\end{array}$$

$$\begin{array}{c}\text{SMT}\\ \text{satisfiability}\\ \text{modulo theories}\end{array} \longrightarrow \begin{array}{c}\#\text{SMT}\\ \text{model counting}\\ \text{modulo theories}\end{array}$$

We need a common framework for counting modulo theories.

Such a framework is provided by **measured logical theories.**

σ-algebras

σ-**algebra** $(D, \mathcal{F})$: domain D, set of subsets $\mathcal{F} \subseteq 2^D$ such that

$$
\begin{array}{rcll}
 & & \varnothing \in \mathcal{F} & \text{(the empty set is an element)} \\
A \in \mathcal{F} & \implies & D \setminus A \in \mathcal{F} & \text{(closure under complementation)} \\
A_i \in \mathcal{F} & \implies & \bigcup_i A_i \in \mathcal{F} & \text{(closure under countable union)}
\end{array}
$$

Examples

- finite set D, $\mathcal{F} = 2^D$
- $D = \mathbb{R}$, $\mathcal{F}$ defined starting from the set of all open intervals by adding all complements and countable unions iteratively so that the closure properties are met

Measure: How big is a set?

Measure μ for $(D, \mathcal{F})$ maps each $A \in \mathcal{F}$ to a real number $\mu(A) \geq 0$

Examples

- $D = \{1, 2, 3, 4, 5, 6\}$, with $\mu(\{d\}) = 1$ for each $d \in D$
  - $\mu(\varnothing) = 0$,
  - $\mu(\{2, 4, 6\}) = |\{2, 4, 6\}| = 3$
- $D = \mathbb{R}$
  - $\mu((10, 15)) = 5$,
  - $\mu([10, 10]) = 0$,
  - $\mu((10, 15) \cup [20, 30]) = 15$

Measure: How big is a set?

Measure μ for $(D, \mathcal{F})$ maps each $A \in \mathcal{F}$ to a real number $\mu(A) \geq 0$

Examples

- $D = \{1, 2, 3, 4, 5, 6\}$, with

$$\begin{aligned} \mu(\{1\}) = \mu(\{3\}) = \mu(\{5\}) = \tfrac{1}{2} \\ \mu(\{2\}) = \mu(\{4\}) = \mu(\{6\}) = 2 \end{aligned}$$

  $\mu(\{2, 3, 4, 6\}) = 6.5$

- $D = \mathbb{R}$, with
  a continuous function $f : \mathbb{R} \to \mathbb{R}$ such that $f(d) \geq 0$ for all d
  $\mu([10, 15]) = \int_{10}^{15} f(x)\,\mathrm{d}x$

Measure: How big is a set?

Measure μ for $(D, \mathcal{F})$ maps each $A \in \mathcal{F}$ to a real number $\mu(A) \geq 0$

More formally

$$
\begin{array}{lcl}
A \in \mathcal{F} & \Longrightarrow & \mu(A) \geq \mu(\varnothing) = 0 \\
A_i \in \mathcal{F} \text{ disjoint} & \Longrightarrow & \mu(\bigcup_i A_i) = \sum_i \mu(A_i)
\end{array}
$$

Measure: How big is a set?

Measure μ for $(D, \mathcal{F})$ maps each $A \in \mathcal{F}$ to a real number $\mu(A) \geq 0$

More formally

$$\begin{array}{lcl} A \in \mathcal{F} & \Longrightarrow & \mu(A) \geq \mu(\varnothing) = 0 \\ A_i \in \mathcal{F} \text{ disjoint} & \Longrightarrow & \mu(\bigcup_i A_i) = \sum_i \mu(A_i) \end{array}$$

Measure space $(D, \mathcal{F}, \mu)$: σ-algebra $(D, \mathcal{F})$, measure $\mu : \mathcal{F} \to \mathbb{R}$

Product Measure

With each variable x_i, associate a measure space $(D_i, \mathcal{F}_i, \mu_i)$.

Models of $\varphi(x_1, \ldots, x_k)$ are elements of $D_1 \times \ldots \times D_k$.

If each D_i is a countable union of elements of $\mathcal{F}_i$ we can define

$$\mu(A_1 \times \ldots \times A_k) = \mu_1(A_1) \ldots \mu_k(A_k).$$

$$\mu([0, 100] \times [0, 100] \times [0, 100]) = 100^3$$

Measured theories and model count

A logical theory $\mathcal{T}$ is measured if every $[\![\varphi]\!]$ is measurable.

The model count of a formula φ is $\mathrm{mc}(\varphi) = \mu([\![\varphi]\!])$.

Model counting problem: Given a formula φ, compute $\mathrm{mc}(\varphi)$.

Measured theories: Examples

Theory	**Domain**	**Connectives**	**Quantifiers**	mc(φ)
Boolean satisfiability	$\{\mathsf{T}, \mathsf{F}\}$	$\wedge, \vee, \neg$	None	Number of satisfying assignments
Integer arithmetic	$\mathbb{Z} \cap [a, b]$	$\wedge, \vee, \neg$	$\exists$	Number of models
Linear real arithmetic	$\mathbb{R} \cap [a, b]$	$\wedge, \vee, \neg$	$\exists$	Volume

$$\begin{tikzcd}
\begin{array}{c}\mathrm{SAT}\\ \text{satisfiability}\end{array} \arrow[r] \arrow[d] & \begin{array}{c}\mathrm{SMT}\\ \text{satisfiability}\\ \text{modulo theories}\end{array} \arrow[d] \\
\begin{array}{c}\#\mathrm{SAT}\\ \text{model counting}\end{array} \arrow[r] & \begin{array}{c}\#\mathrm{SMT}\\ \text{model counting}\\ \text{modulo theories}\end{array}
\end{tikzcd}$$

Efficient engines for SAT and SMT are widely used.

SAT
satisfiability

SMT
satisfiability
modulo theories

#SAT
model counting

#SMT
model counting
modulo theories

What about efficient engines for model counting?

Counting by enumeration

- Takes exponential time in the worst case for SAT.
  (2^n possible assignments for n variables)

- Cannot be directly applied to continuous problems, e.g., volume computation. Discretization works in the limit.

  Example: approximate the value of an integral

How much better can we do?

Computational complexity of counting

Some problems are easy and can be solved in polynomial time.

Many counting problems of practical interest are $\#\mathbf{P}$-complete, and cannot be solved in polynomial time unless $\mathbf{P} = \mathbf{NP}$.

The complexity class $\#\mathbf{P}$ consists of the counting problems associated with the decision problems in $\mathbf{NP}$.

More on computational complexity in the lecture on Tuesday.

How about approximation?

What if we ask for a procedure $\mathcal{A}$ that approximates the answer?

We want a procedure $\mathcal{A}$ for a given counting problem such that given an ϵ and an instance I of the problem, $\mathcal{A}(I, \epsilon)$ is such that

$$|\mathcal{A}(I, \epsilon) - count(I)| \leq \epsilon \qquad \text{(additive error)}$$

or

$$\frac{1}{(1+\epsilon)} count(I) \leq \mathcal{A}(I, \epsilon) \leq (1+\epsilon) count(I) \qquad \text{(multiplicative error)}$$

No known efficient deterministic approximation algorithm for any $\#\mathbf{P}$-complete problem.

Randomized approximation algorithms for counting

Use randomization to compute an approximation that is sufficiently close to the actual value with high probability.

There are efficient randomized approximation algorithms for many $\#\mathbf{P}$-complete problems.

More on randomized algorithms on Wednesday and Thursday.

Approaches based on random sampling

Monte Carlo methods use random sampling to estimate a value.

Example

Estimating the value of π.

1. Sample independently at random m points from

$$S = \{(x, y) \in \mathbb{R}^2 : |x| \leq 1, |y| \leq 1\}.$$

2. Let V be the number of samples in

$$C = \{(x, y) \in \mathbb{R}^2 : x^2 + y^2 \leq 1\}.$$

3. Return $\frac{4V}{m}$.

If m is large enough, close approximation with high probability.

Approaches based on random sampling

Markov Chain Monte Carlo methods are random sampling methods that are often used for high dimensional problems (for example, for estimating the volume of a convex body in $\mathbb{R}^n$).

More on Monte Carlo and Markov Chain Monte Carlo on Wednesday.

A naive approach based on choosing a random subset

We want to count the number of elements of a set $C \subseteq S$.

1. Partition S into small disjoint sets $S_1, \ldots, S_m$.
2. Pick a random S_i and count the elements of C in S_i.
3. Return mV_i, where $V_i = |C \cap S_i|$.

Challenge: how to pick a representative subset?

Hashing-based approach

Using **hashing** we can partition the set S by iterative splitting.

Key property: with high probability each split cuts the elements of C in a partition roughly in half. This gives us representative sets!

We will learn about the hashing approach to $\#\mathrm{SAT}$ on Thursday.

This approach can be applied to model counting for real arithmetic by combining hashing and discretization. More on this on Friday.

Summary of today's lecture

- Model counting: What it is and in what contexts it is useful
- Measured logical theories: A common framework for counting problems in different domains
- Randomized approximation algorithms: How they can be useful in the context of model counting

Agenda for the rest of the week

Tuesday	computational complexity, probability theory
Wednesday	randomized algorithms, Monte Carlo methods
Thursday	hashing-based approach to model counting
Friday	from discrete to continuous model counting