Logic for Knowledge Representation, Learning, and Inference

Luciano Serafini

serafini@fbk.eu

Version August 23, 2023

Contents

CHAPTER 1

Model counting

1. Introduction

A propositional formula ϕ partition the set of interpretations into two disjoint subsets. those that satisfy ϕ and those that do not.

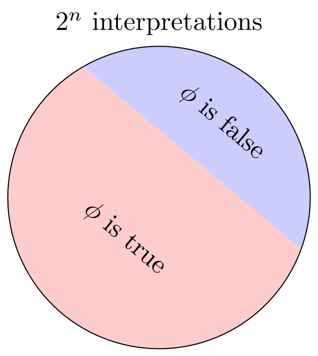


Figure 1. Visualisation of Model counting problem. The circle contains all the interpretations of the propositional variables of ϕ, which are 2^n. Model counting has to count the size of the red area.

Propositional model counting or #sat Gomes, Sabharwal, and Selman 2009 is the problem of computing the number of models of a given propositional formula, i.e., the number of distinct truth assignments to propositional variables of a formula ϕ which satisfy the formula. Let us define the problem more precisely.

Definition 1.1 (Model counting problem). *For a propositional formula ϕ let $\mathcal{P}(\phi)$ be the set of propositional variables occurring in ϕ. the* model counting of ϕ *#sat(ϕ) is the problem of finding the number of truth assignments to the propositional variables of ϕ that satisfies ϕ.*

$$\#\textsc{sat}(\phi) = |\{\mathcal{I} : \mathcal{P}(\phi) \rightarrow \{0,1\} \mid \mathcal{I} \models \phi\}|$$

Example 1.1. *A naive method to solve the model counting problem is via truth tables. For instance let us compute the model counting of the following formulas:* $p \wedge q$, $p \vee q$, $p \rightarrow q$, $p \equiv q$, $\neg p$ *and* $p \wedge \neg p$

p	q	$p \wedge q$	$p \vee q$	$p \rightarrow q$	$p \leftrightarrow q$	$\neg\ p$	$p \wedge \neg\ p$	$p \vee \neg\ p$
1	1	1 1 1	1 1 1	1 1 1	1 1 1	0 1	1 0 0 1	1 1 0 1
1	0	1 0 0	1 1 0	1 0 0	1 0 0	0 1	1 0 0 1	1 1 0 1
0	1	0 0 1	0 1 1	0 1 1	0 0 1	1 0	0 0 1 0	0 1 1 0
0	0	0 0 0	0 0 0	0 1 0	0 1 0	1 0	0 0 1 0	0 1 1 0
#SAT		1	3	3	2	2	0	4

Notice that in the above truth table the first four formulas contains all the propositional variables of the truth table, p and q in this case. while the last three formulas contains only one variables of the two. If we had computed model counting of the last three formulas separately, we would have obtained a different result:

p	$\neg\ p$	$p \wedge \neg\ p$	$p \vee \neg\ p$
1	0 1	1 0 0 1	1 1 0 1
0	1 0	0 0 1 0	0 1 1 0
	1	0	2

The difference is due to the fact that, in the first truth table, we compute model counting in the context *of the language of the propositional variables p and q, even if the formulas contain only one propositional variable (p in this case). This implies that we consider four possible truth assignments. In the second truth table, instead, we computee model counting in the context of the langauge that contains only one single variable, i.e., the one that appears in the formulas; in this case the number of interpretations are only two.*

The previous example highlights the fact that, to be precise, #SAT should also take also an extra parameter that is the language (set of propositional variables) of the interpretations. For this reason, when necessary we make explicit the langaue in which we compute the model counting, by writing $\#\text{SAT}(\phi, \mathcal{P})$, where $\mathcal{P}$ is a set of propositional variables such that $\mathcal{P}(\phi) \subseteq \mathcal{P}$. When we write the simpler notation $\#\text{SAT}(\phi)$ we actually mean $\#\text{SAT}(\phi, \mathcal{P}(\phi))$.

An alternative and equivalent formulation of the model counting problem, which will be useful when we want to extend the problem to *weighted model counting* is the following:

$$\#\text{SAT}(\phi, \mathcal{P}) = \sum_{\mathcal{I}:\mathcal{P}\rightarrow\{0,1\}} \mathcal{I}(\phi) \tag{1}$$

where $\mathcal{I}(\phi) = 1$ if $\mathcal{I} \models \phi$ and 0 otherwise.

The[1] model counting problem presents fascinating challenges for practitioners and poses several new research questions. Effcient algorithms for this problem will have a significant impact on many application areas that are inherently beyond SAT ('beyond' under standard complexity theoretic assumptions), such as probabilistic reasoning Chavira and Darwiche 2008; Holtzen, Van den Broeck, and Millstein 2020 For example, various probabilistic inference problems, such as Bayesian net reasoning, can be effectively translated into model counting problems.Another application is in the study of hard combinatorial problems, such as combinatorial designs, where the number of solutions provides further insights into the problem. Even finding a single solution can be a challenge for such problems; counting the number of solutions is much harder. Not surprisingly, the largest formulas we can solve for the

[1]This paragraph is an excerpt of the introductory section of Gomes, Sabharwal, and Selman 2009

model counting problem with state-of-the-art model counters are orders of magnitude smaller than the formulas we can solve with the best SAT solvers. Generally speaking, current exact counting methods can tackle problems with a couple of hundred variables, while approximate counting methods push this to around 1,000 variables.

A rough intuition on the relation among model counting and probabilistic reasoning is as follows. If we consider an assignment to a set of variables as the outcome of an experiment and the formula ϕ a measure on this outcome, which is 1 if the outocme satisfy ϕ and 0 otherwise, then the higher $\#\textsc{sat}(\phi)$ the higher the probability of observing a 1 in the measure.

As mentioned before the solution of the model counting problem is a very complex task and therefore one could also consider more efficient algirhtms that that does not guarante an exact solution, but provides an approximated one. Therefore, we will divide practical model counting techniques into two main categories: *exact counting* and *approximate counting*. Within exact counting, we will distinguish between methods based on DPLL-style exhaustive search, and those based on *knowledge compilation* or conversion of the formula into certain normal forms for which model counting is efficient (polinomial) Within approximate counting, we will distinguish between methods that provide fast estimates without any guarantees and methods that provide lower or upper bounds with a correctness guarantee.

2. Basic properties of model counting

As shown by the introductory example, the model counting of the same formula depends from the set of propositional variables that we consider (which should include those of the formula itself), Such a dependency is clarified in the following property:

Proposition 1.1. *If ϕ is a propositinal formula that contains propositional variables in $\mathcal{P}$, i.e., $\mathcal{P}(\phi) \subseteq \mathcal{P}$, then $\#\textsc{sat}(\phi, \mathcal{P}) = \#\textsc{sat}(\phi) \cdot 2^{|\mathcal{P} \setminus \mathcal{P}(\phi)|}$.*

Proof. Every interpretation $\mathcal{I}$ in $\mathcal{P}(\phi)$ that satisfies ϕ can be extended to an interpretation $\mathcal{I}'$ in the language of $\mathcal{P}$ by assigning any value in $\{0, 1\}$ to the variables in $\mathcal{P} \setminus \mathcal{P}(\phi)$. This implies that there are $2^{|\mathcal{P} \setminus \mathcal{P}(\phi)|}$ estensions. Since the truth value of ϕ is independent from the assignment to the variables not in $\mathcal{P}(\phi)$, we have that $\mathcal{I} \models \phi$ if and only if $\mathcal{I}' \models \phi$. Therefore for every models of ϕ in $\mathcal{P}(\phi)$ we have $2^{|\mathcal{P} \setminus \mathcal{P}(\phi)|}$ distinct models of ϕ in $\mathcal{P}$. Furthermore, every interpretation $\mathcal{I}'$ in $\mathcal{P}$ that satisfies ϕ, can be restricted to an interpretation $\mathcal{I}$ in $\mathcal{P}(\phi)$ by dropping the assignments to $\mathcal{P} \setminus \mathcal{P}(\phi)$. This guarantees that $\#\textsc{sat}(\phi) \cdot 2^{|\mathcal{P} \setminus \mathcal{P}(\phi)|} \leq \#\textsc{sat}(\phi, \mathcal{P})$. To show that $\#\textsc{sat}(\phi) \cdot 2^{|\mathcal{P} \setminus \mathcal{P}(\phi)|} \geq \#\textsc{sat}(\phi, \mathcal{P})$, notice that any pair of distinct interpretations $\mathcal{I}$ and $\mathcal{J}$ in $\mathcal{P}(\phi)$ which are extended into $\mathcal{I}'$ and $\mathcal{J}'$ interpretations of $\mathcal{P}$, will be such that $\mathcal{I}'$ is different from $\mathcal{J}'$. $\square$

The previous proposition states that the model counting of a fomrula ϕ can be obtained multiplying the model counting of the formula w.r.t., the set of variables it contains times the number of assignments to the variables not containing in ϕ.

Other important properties of $\#\textsc{sat}$ are the following:

Proposition 1.2. *(1) If ϕ is valid, $\#\textsc{sat}(\phi)$ is equal to $2^{|\mathcal{P}(\phi)|}$*
(2) If ϕ is unsatisfiable $\#\textsc{sat}(\phi)$ is equal to 0
(3) $\#\textsc{sat}(\neg\phi) = 2^{|\mathcal{P}(\phi)|} - \#\textsc{sat}(\phi)$

(4) If $\phi \models \psi$ then $\#\textsc{sat}(\phi, \mathcal{P}) \leq \#\textsc{sat}(\psi, \mathcal{P})$, where $\mathcal{P}$ is the set of propositional variables of ϕ and ψ.
(5) if ϕ is equivalent to ψ, then $\#\textsc{sat}(\phi, \mathcal{P}) = \#\textsc{sat}(\psi, \mathcal{P})$ and (by Property 1.1) $\#\textsc{sat}(\phi) \cdot 2^{\mathcal{P} \setminus \mathcal{P}(\psi)} = \#\textsc{sat}(\phi) \cdot 2^{\mathcal{P} \setminus \mathcal{P}(\phi)}$

Proof. (1) If ϕ is valied then every truth assignment of the propositional variables in ϕ will satisty ϕ. Since there are $2^{\mathcal{P}(\phi)}$ interpretation then $\#\textsc{sat}(\phi) = 2^{\mathcal{P}(\phi)}$.
(2) If ϕ is unsatisfiable then ϕ is not satisfied by any interpretation and therefore $\#\textsc{sat}(\phi) = 0$;
(3) Notice that $\mathcal{I} \models \neg\phi$ is and only if $\mathcal{I} \not\models \phi$. Since the total number of interpretations of the language of ϕ is $2^{\mathcal{P}(\phi)}$, we have that $\#\textsc{sat}(\neg\phi) = 2^{\mathcal{P}(\phi)} - \#\textsc{sat}(\phi)$.
(4) If $\phi \models \phi$ then every interpretation in the language of ϕ and ψ that satisfies ϕ also satisfies ψ.

□

A second set of properties concerns the relationship between the model counting of a formula and the model counting of its direct sub-formulas. We concentrate on the propositional connectives $\neg$, $\wedge$ and $\vee$ since, all the other connectives can be rewritten in ters of these three connectives.

2.1. #sat of negation. Since we have the low of excluded middle, i.e. every assignment $\mathcal{I}$ is such that either $\mathcal{I} \models \phi$ or $\mathcal{I} \models \neg\phi$, to count the models of $\neg\phi$, we can subtract the number of models of ϕ from the total set of assignments to the variables of ϕ which is $2^{|\mathcal{P}(\phi)|}$.

Proposition 1.3. *$\#\textsc{sat}(\neg\phi) = 2^n - \#\textsc{sat}(\phi)$ where n is the number of propositinal variables that appears in ϕ.*

2.2. #sat of conjunction. In this section we consider how we can count the models of $\phi \wedge \phi$ by separately counting the models of ϕ and ψ or some derived (and simpler) formulas. If ϕ and ψ do not share propositional variables, then then the assignment to the propositional variables of ϕ does not interfere with the assignment to the propositional variables of ϕ. Therefore an assignment that satisfies $\phi \wedge \psi$ can be obtained by selecting any pair composed of a models of ϕ and a model of ψ. And therefore the number of models n of $\phi \wedge \psi$ is the product of the models of ϕ and the models ψ.

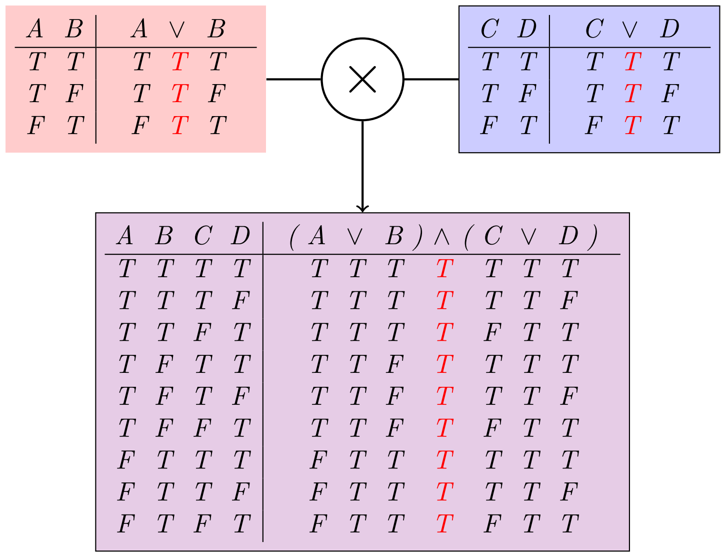
Proposition 1.4. *If ϕ and ψ do not share propositional variables, then $\#\textsc{sat}(\phi \wedge \psi) = \#\textsc{sat}(\phi) \cdot \#\textsc{sat}(\psi)$.*

Proof. Let $n = \#\textsc{sat}(\phi)$ and $m = \#\textsc{sat}(\psi)$. For every pair $\mathcal{I}$ and $\mathcal{J}$ of models of ϕ and ψ respectively we can define the assignment $\mathcal{I} \otimes \mathcal{J}$ on the propositinal variables of $\phi \wedge \psi$ defined as

$$\mathcal{I} \otimes \mathcal{J}(p) = \begin{cases} \mathcal{I}(p) & \text{if } p \in \mathcal{P}(\phi) \\ \mathcal{J}(p) & \text{if } p \in \mathcal{P}(\psi) \end{cases}$$

Since there is no overlap between the variables of ϕ and ψ the definition of $\mathcal{I} \otimes \mathcal{J}$ is well founded, Furthermore we have that $\mathcal{I} \otimes \mathcal{J} \models \phi \wedge \psi$ if and only if $\mathcal{I} \models \phi$ and $\mathcal{J} \models \psi$. Viceversa any model $\mathcal{I}$ of $\phi \wedge \psi$ can be decomposed of a model of ϕ and a model of ψ by restricting it to the propositional variables that appear in the two formulas. Therefore the number of models of $\phi \wedge \psi$ is equal to $n \times m$. $\square$

EXAMPLE 1.2. *Let us consider the formula $(A \vee B) \wedge (C \vee D)$ In the following picture we show how the assignments that satisfy this formula can be obtained by the composition of the of any models of $(A \vee B)$ and a model of $(C \vee D)$.*



A	B	A	$\vee$	B
T	T	T	T	T
T	F	T	T	F
F	T	F	T	T

$$\times$$

C	D	C	$\vee$	D
T	T	T	T	T
T	F	T	T	F
F	T	F	T	T

A	B	C	D	$($ A	$\vee$	B $)$	$\wedge$	$($ C	$\vee$	D $)$
T	T	T	T	T	T	T	T	T	T	T
T	T	T	F	T	T	T	T	T	T	F
T	T	F	T	T	T	T	T	F	T	T
T	F	T	T	T	T	F	T	T	T	T
T	F	T	F	T	T	F	T	T	T	F
T	F	F	T	T	T	F	T	F	T	T
F	T	T	T	F	T	T	T	T	T	T
F	T	T	F	F	T	T	T	T	T	F
F	T	F	T	F	T	T	T	F	T	T

What happens if ϕ and ψ contains common variables? We can combine a model of ϕ with a model of ψ only if the two models agree on the assignment of the shared propositional variables. Therefore for every assignment of the propositional variables we can combine the models that of ϕ and ψ that agree with this assignment. This is stated in the following proposition.

PROPOSITION 1.5. *If $\mathcal{Q} = \mathcal{P}(\phi) \cap \mathcal{P}(\psi)$ is the set of propositional variables that appears in both ϕ and ψ, then $\#\text{SAT}(\phi \wedge \psi) = \sum_{\mathcal{I}:\mathcal{Q} \to \{0,1\}} \#\text{SAT}(\phi|_{\mathcal{I}}) \cdot \#\text{SAT}(\psi|_{\mathcal{I}})$ where $\phi|_{\mathcal{I}}$ is the formula obtained by replacing p with $\top$ in ϕ if $\mathcal{I}(p) = 1$ and with $\bot$ if $\mathcal{I}(p) = 0$.*

Clearly Proposition 1.4 is a special case of Proposition 1.5.

EXAMPLE 1.3. *Let us compute the number of models of $(p \vee q) \wedge (\neg q \vee r)$.*

$$\begin{aligned}\#\text{SAT}((p \vee q) \wedge (\neg q \vee r)) &= \#\text{SAT}((\top \vee q) \wedge (\neg\top \vee r)) + \#\text{SAT}((\bot \vee q) \wedge (\neg\bot \vee r)) \\ &= \#\text{SAT}(\top \vee q) \cdot \#\text{SAT}(\neg\top \vee r) + \#\text{SAT}(\bot \vee q) \cdot \#\text{SAT}(\neg\bot \vee r)\end{aligned}$$

Notice that if ϕ and ψ share k propositional variables, the summations over all the possible interpretations of the k shared variable will contains 2^k addends.

2.3. #sat of disjunction. Let us now see how we can count the models of $\phi \vee \psi$. We know that a $\mathcal{I}$ is a model of $\phi \vee \psi$ if it is a model of ϕ or it is a model of ψ, So a first attempt would be to sum the models of ϕ and the models of ψ. However, in this way we counting twice the interpretations that satisfy both ϕ and ψ Therefore to fix this we have to subtract the models of $\phi \wedge \psi$.

PROPOSITION 1.6. *Let $\mathcal{P}$ be the set of propositional variables that occour in $\phi \vee \psi$. The following properties holds:*

(1) $\#\text{SAT}(\phi \vee \psi) = \#\text{SAT}(\phi, \mathcal{P}) + \#\text{SAT}(\psi, \mathcal{P}) - \#\text{SAT}(\phi \wedge \psi)$*;*
(2) If $\phi \wedge \psi$ *is unsatisfiable then* $\#\text{SAT}(\phi \vee \psi) = \#\text{SAT}(\phi, \mathcal{P}) + \#\text{SAT}(\psi, \mathcal{P})$*.*
(3) If ϕ *and* ψ *contain the same set of propositional variables then* $\#\text{SAT}(\phi \vee \psi) = \#\text{SAT}(\phi) + \#\text{SAT}(\psi) - \#\text{SAT}(\phi \wedge \psi)$*.*
(4) If ϕ *and* ψ *contain the same set of propositional variables and* $\phi \wedge \psi$ *is unsatisfiable then* $\#\text{SAT}(\phi \vee \psi) = \#\text{SAT}(\phi) + \#\text{SAT}(\psi)$*.*

PROOF. We prove only property (1) since all the other properties are corollaries. The set models$(\phi \vee \psi)$ of the models of $\phi \vee \psi$ can be partitioned in three disjoint subsets models$(\phi \wedge \neg\psi)$, models$(\neg\phi \wedge \psi)$, and models$(\phi \wedge \psi)$ Which implies that

$$\#\text{SAT}(\phi \vee \psi) = |\text{models}(\phi \wedge \neg\psi)| + |\text{models}(\neg\phi \wedge \psi)| + |\text{models}(\phi \wedge \psi)| \tag{2}$$

The set of assignments to $\mathcal{P}$ that satisfy ϕ can be partitioned in the two subsets models$(\phi \wedge \neg\psi)$ and models$(\phi \wedge \psi)$. From which we have that

$$\#\text{SAT}(\phi, \mathcal{P}) = |\text{models}(\phi \wedge \neg\psi)| + |\text{models}(\phi \wedge \psi)|$$

Similarly, we have that

$$\#\text{SAT}(\psi, \mathcal{P}) = |\text{models}(\neg\phi \wedge \psi)| + |\text{models}(\phi \wedge \psi)| \tag{3}$$

And therefore,

$$\begin{aligned}&\#\text{SAT}(\phi, \mathcal{P}) + \#\text{SAT}(\psi, \mathcal{P}) \\ &\qquad = |\text{models}(\phi \wedge \neg\psi)| + |\text{models}(\neg\phi \wedge \psi)| + 2 \cdot |\text{models}(\phi \wedge \psi)|\end{aligned}$$

From which we have that

$$\begin{aligned}&\#\text{SAT}(\phi, \mathcal{P}) + \#\text{SAT}(\psi, \mathcal{P}) - \text{models}(\phi \wedge \psi)| \\ &\qquad = |\text{models}(\phi \wedge \neg\psi)| + |\text{models}(\neg\phi \wedge \psi)| + |\text{models}(\phi \wedge \psi)|\end{aligned} \tag{4}$$

By combining (2) and (4), we obtain property (1). □

PROPOSITION 1.7. *If* p *is a propositional variable of* ϕ*,, then* $\#\text{SAT}(\phi) = \#\text{SAT}(\phi|_p) + \#\text{SAT}(\phi|_{\neg p})$

PROOF. The proposition is a direct consequence of property (4) of Proposition 1.6. Indeed $\phi|_p$ and $\phi|_{\neg p}$ contain the same set of propositional variables; furthermore, since $\phi|_p$ and $\phi|_{\neg p}$ are equivalent to $\phi \wedge p$ and $\phi \wedge \neg p$ respectively, there is no interpretation that satisfies both formulas. This implies that property (4) of Proposition± 1.6 is applicable. Notice that the fact that p occurs in ϕ is an essential assumption, otherwise $\phi|_p$ and $\phi_{\neg p}$ would be the same formula □

We have seen that a naïve method for counting the models of a propositional formula is by computing the whole truth table. Since the truth table for a formula ϕ with n propositional variables contains 2^n lines, this method is very costly as it takes exponential time on the number of propositional variables. There are two possible direction in construct more efficient model counting algorithms.

The first one, consists in exploiting the properties seen in the previous section in order to take shortcuts, parallelise, and decompose the #SAT problem. Following this direction, for instance the fact that $\#\text{SAT}(\phi \wedge \psi) = \#\text{SAT}(\phi) \cdot \#\text{SAT}(\psi)$ when ϕ and ψ do not share propositinal variables, can be used to reduce the complexity, form 2^n to $2^{\max(n_1,n_2)}$ where n_1 and n_2 is the number of propositional variables occurring in ϕ and ψ respectively. This direction falls under the name of "exact model counting" as the algorithms are guaranteed to return the correct value for $\#\textsc{sat}(\phi)$. Since, as we will see in the following, exact algorithms are anyway very complex, the second direction aims to develop efficient algorithm that return an approximation of $\#\textsc{sat}(\phi)$. In this section we describe exact algorithms while an example of approximate algorithms is described in the next section.

3. DPLL-Based Model Counting

In counting models of a propositional formula ϕ, one can see that [2] the models of ϕ can be split in two disjoint subsets by selecting a propositional variable p and counting the models of $\phi \wedge p$ and $\phi \wedge \neg p$. Counting the models of $\phi \wedge p$, is the same as counting the model of $\phi|_p$ which is the formula obtained by replacing p with $\top$ in ϕ. Similarly for $\phi|_{\neg p}$, which is obtained from ϕ by replacing p with $\bot$. With this simple rule we reduce $\#\textsc{sat}(\phi)$ to $\#\textsc{sat}(\phi|_p) + \#\textsc{sat}(\phi|_{\neg p})$. Furthermore it is possible that ϕ_p and $\phi_{\neg p}$ are equivalent to formulas in which many other propositional letters have been removed. For instance if $\phi = (p \vee q) \wedge (r \vee s)$ then $\phi|_p$ is equivalent to $r \wedge s$. Therefore, by property 5 of Proposition 1.2 we can compute $\#\textsc{sat}(\phi)$ by summing $\#\textsc{sat}(\text{simplify}(\phi|'_p)) \cdot 2^n$ and $\#\textsc{sat}(\text{simplify}(\phi|_{\neg p})) \cdot 2^m$ where n and m are the number of propositional variables that has been eliminated by simplifying $\phi|_p$ and $\phi|_{\neg p}$ respectively.

The property just described constitute the base for algorithm CDP (Counting Decision Procedure), shown in Algorithm 1. In particular the CPD algorithms exploits the property that for every ϕ in CNF

$$\#\textsc{sat}(\phi, \mathcal{P}) = \#\textsc{sat}(\phi|_p, \mathcal{P} \setminus \{p\}) + \#\textsc{sat}(\phi|_{\neg p}, \mathcal{P} \setminus \{p\})$$

Notice that $\#\textsc{sat}(\phi, \mathcal{P}) = \#\textsc{sat}(\phi) \cdot 2^m$ where $m = |\mathcal{P} \setminus \mathcal{P}(\phi)|$. We could therefore replace $\mathcal{P}$ with n where n is the cardinality of $\mathcal{P}$, Since, $\mathcal{P}(\phi) \subseteq \mathcal{P}$ whe have that $m = n \setminus |\mathcal{P}(\phi)|$. The above property therefore can be rephrased in

$$\#\textsc{sat}(\phi, n) = \#\textsc{sat}(\phi|_p, n-1\}) + \#\textsc{sat}(\phi|_{\neg p}, n-1)$$

CPD(ϕ, n) computes $\#\textsc{sat}(\phi, \mathcal{P})$ for a formula ϕ in conjunctive normal form w.r.t. a set $\mathcal{P}$ of n propositional variables.

Example 1.4. *Let $\phi = \{\{p, q\}, \{\neg r, \neg p\}\}$. To compute $\#\textsc{sat}(\phi)$ we call CDP$(\phi, 3)$ since ϕ contains 3 propositional variables. The execution of $CP(\phi, 3)$ is shown in Figure 2.*

3.1. Literal selection. As for the case of DPLL the choice of the propositional variable to expand (line 8 of algorithm 1) has great influence on the efficiency of the algorithm. According to the above analisys the choice of p will generates two subproblems of expected complexity of $T(m_1, n-1)$ and $T(m_2, n-1)$ where m_1 is the set of clauses in $\phi|_p$ and m_2 the number of clauses in
$phi|_{\neg p}$. TO maximize this reduction we should choose the split that minimize $\max(m_1, m_2)$ where

3.2. Caching. If a set of clauses ϕ is encountered more than one time in the CPD algorithm, it would clearly be beneficial to cash the result of the first computation of $\#\textsc{sat}(\phi)$ and to be able to efficiently recognize it in the next steps and reuse previous results.

[2] See Birnbaum and Lozinskii 1999.

Algorithm 1 CPD(ϕ, n)

```
Require: $\phi$ a propositional formula in CNF
Require: $n$ an integer larger han $|\mathcal{P}(\phi)|$
 1: if $\phi = \{\}$ then                          ▷ $\phi$ is the empty set of clauses
 2:     return $2^n$
 3: end if
 4: if $\{\} \in \phi$ then                        ▷ $\phi$ contains an empty clause
 5:     return 0
 6: end if
 7: if $\{l\} \in \phi$ then                       ▷ Unit propagation
 8:     return CDP($\phi|_l, n-1$)
 9: else
10:     $p \leftarrow$ select a propositinal variable of $\phi$
11:     return CDP($\phi|_p, n-1$) + CDP($\phi|_{\neg p}, n-1$)
12: end if
```

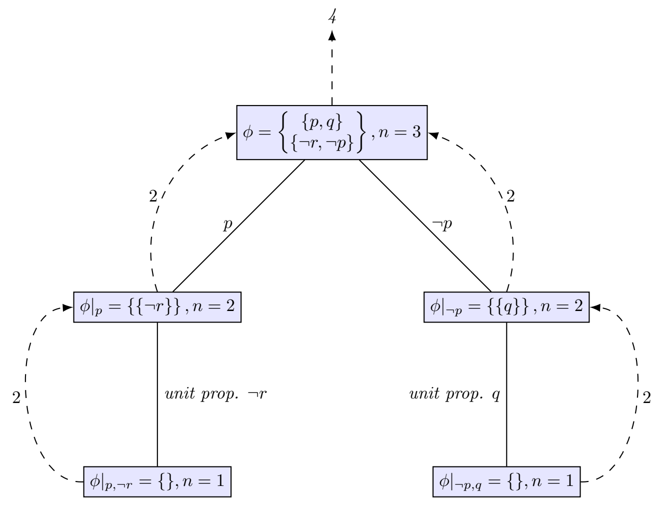

4
$\phi = \left\{ \begin{matrix} \{p, q\} \\ \{\neg r, \neg p\} \end{matrix} \right\}, n = 3$
2
2
p
$\neg p$
$\phi|_p = \{\{\neg r\}\}, n = 2$
$\phi|_{\neg p} = \{\{q\}\}, n = 2$
2
unit prop. $\neg r$
unit prop. q
2
$\phi|_{p,\neg r} = \{\}, n = 1$
$\phi|_{\neg p,q} = \{\}, n = 1$

FIGURE 2. The execution tree of the function $CPD(\phi, 3)$. The total number of models returned by this procedure is 4, which is the sum of the two recursive calls.

3.2.1. *Complexity of CPD.* In the worse case the CDP decision procedure will generate all the possible assignments and therefore runs for 2^n steps. We way that CPD is *worse case exponential.* However we can provide a probabilistic estimation of the complexity of the CDP procedure. Birnbaum and Lozinskii 1999 provide such a result, and we report it in the following.

Assume that ϕ contains m clauses on n propositional variable. Assume also that the literals have the same probability p to appear in each clause ϕ. Let $T(m, n)$ denote the average running time of $CDP(\phi, n)$ We have the following theorem:

THEOREM 1.1. *$T(m, n) = O(m^d \cdot n)$ where* $d = \lceil \frac{-1}{\log_2(1-p)} \rceil$

The assumption of $p = \frac{1}{3}$ is commonly adopted in probabilistic analysis of algorithms handling CNF or DNF formulas Franco and Paull 1983 which means that for each variable, its occurrence in a clause with or without negation or non-occurrence, all have the same probability. Under this assumption we have that $T(m, n) = O(m^2 \cdot n)$.

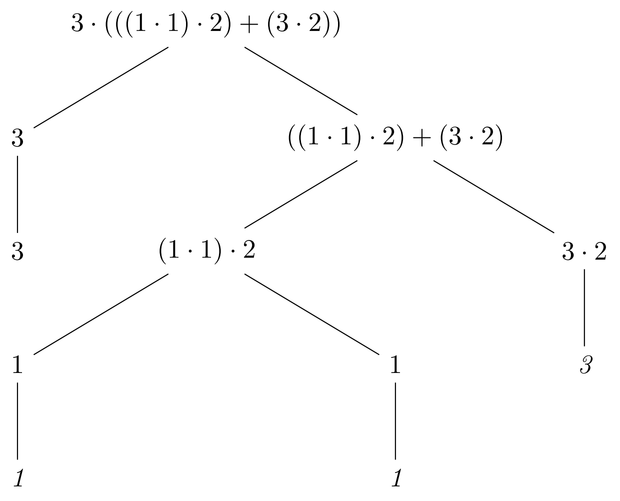
4. Model counting via Knowledge Compilation

The second method for exact model counting of a propositional formula is based on the transformation of ϕ is an equivalent formula for which model counting is decomposable using the properties introduced at the beginning of this chapter. Consider the following example:

EXAMPLE 1.5. *In computing $\#\text{SAT}((A \vee B) \wedge ((C \wedge D) \vee (\neg D \wedge E)))$ we can recursively apply the properties of model counting. This allow to decompose the problem of counting the models of a complex formula in the problem of counting the models of its subformulas and aggregating the results properly. The following tree shows how the $\#\text{SAT}$ ofr such a formula can be decomposed.*

$$\begin{tikzcd}
 & & \#\text{SAT}((A \vee B) \wedge ((C \wedge D) \vee (\neg D \wedge E))) \arrow[dl, no head] \arrow[dr, no head] & & \\
 & \#\text{SAT}(A \vee B) \arrow[d, no head] & & \#\text{SAT}((C \wedge D) \vee (\neg D \vee E)) \arrow[dl, no head] \arrow[dr, no head] & \\
 & 3 & \#\text{SAT}(C \wedge D) \cdot 2 \arrow[dl, no head] \arrow[dr, no head] & & \#\text{SAT}(\neg D \vee E) \cdot 2 \arrow[d, no head] \\
 & \#\text{SAT}(C) \arrow[d, no head] & & \#\text{SAT}(D) \arrow[d, no head] & 3 \\
 & 1 & & 1 &
\end{tikzcd}$$

The above tree show that to compute $\#\text{SAT}((A \vee B) \wedge ((C \wedge D) \vee (\neg D \wedge E)))$ we compute $\#\text{SAT}(A \vee B) = 3$, $\#\text{SAT}(C) = 1$, $\#\text{SAT}(D) = 1$, and $\#\text{SAT}(\neg D \vee E = 3$. We then aggregates these results by seeing the decomposition tree as a algebraic expression:

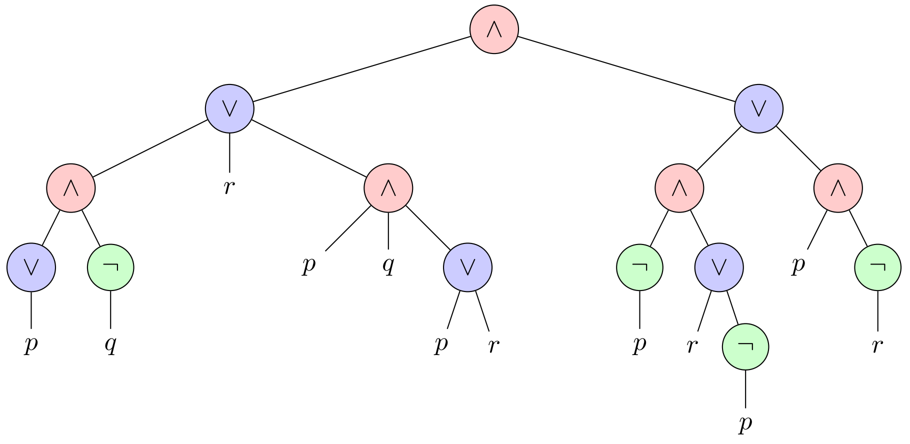

$3 \cdot (((1 \cdot 1) \cdot 2) + (3 \cdot 2))$
3
$((1 \cdot 1) \cdot 2) + (3 \cdot 2)$
3
$(1 \cdot 1) \cdot 2$
$3 \cdot 2$
1
1
3
1
1

Knowledge compilation is the transformation of a formula ϕ in a form such that the decompositions shown in the previous example are possible. To this purpose we define a new normal form of propositional formulas and the rewriting rules that are necessary to transform every formula in such a form.

Before doing this let us recall what is Negated Normal Form (NNF).

Definition 1.2 (NNF). *A formula is in NNF (Negated Normal Form) if it contains only $\wedge$, $\vee$ and $\neg$ connectives, and the $\neg$ connective occours only in front of propositional variables*

Example 1.6. *In the following we show the parse tree of a formula in NNF. Notice that every branch contains an alternation of $\wedge$ and $\vee$ and ends with either an atom or the negation of the atom. aAA*



A formula is in NNF can be written as

$$\bigwedge_{i_1=1}^{n} \bigvee_{i_2=1}^{n_{i_1}} \bigwedge_{i_3=1}^{n_{i_1 i_2}} \cdots \bigvee_{i_k=1}^{n_{i_1 \dots i_k}} l_{i_1,\dots,i_k} \tag{5}$$

where $l_{i_1,\dots,i_k}$ are literals and k is the depth (= maximum branch length) of the tree. CNF is a special kind of NNF with $k = 2$.

DEFINITION 1.3 (DNNF). *A propositional formula ϕ is in* Decomposable negation normal form (DNNF) *it is in Negated Normal Form (NNF) and for each conjunction $\phi_1 \wedge \phi_2 \wedge \cdots \wedge \phi_n$ $\mathcal{P}(\phi_i) \cap \mathcal{P}(\phi_j) = \emptyset$.*

Notice that in a DNNF formula in order to compute the model counting of a sub-formula that is a disjunction $\phi \wedge \psi$ we can apply Proposition 1.4 by computing the model counting of ϕ and ψ separately and then multiply the results.

Let us define a form that guarantee a similar property for disjunction.

DEFINITION 1.4 (d-DNNF). *A formula is in d-DNNF (deterministic DNNF) if it is in DNNF and for each disjunction $\phi_1 \vee \phi_2 \vee \cdots \vee \phi_n$ occurring in the formula, there is at most one $\mathcal{I}$ such that $\mathcal{I} \models \phi_i$ for every ϕ_i.*

As in the case of CNF we can define a set of rules that allows to tranform a formula ϕ in an equivalent formula in d-DNNF. The key tranformation to is called *Shannon's expansion.* The Shannon's expansion is a transformation that at the same time remove shared variables in a conjunction $\phi \wedge \psi$ and introduces a deterministic disjunction. The Shannon's expansion of ϕ is equal to

$$(p \wedge \phi|_p) \vee (\neg p \wedge \phi|_{\neg p}) \tag{6}$$

- Notice that, if ϕ is a conjunction $\phi_1 \wedge \phi_2$, then $p \wedge \phi|_p = p \wedge \phi_1|_p \wedge \phi_2|_p$, which is a conjunction of three formulas that do not share the variable p since p has been removed in $\phi|_p$ and $\phi_{\neg p}$. Similar observation holds for $\neg p \wedge \phi|_{\neg p}$. Furthermore, the disjunction introduced by the Shannon expansion is deterministic, since it is not possible that $p \wedge \phi|_p$ and $\neg p \wedge \phi|_{\neg p}$ are both true in an interpretation.
- If instead ϕ is the disjunction $\phi_1 \vee \phi_2$ and p occours either in ϕ_1 or in ϕ_2 or in both, then $p \wedge \phi|_p \vee \neg p \wedge \phi|_{\neg p}$ is equal to
$$(p \wedge (\phi_1|_p \vee \phi_2|_p)) \vee (\neg p \wedge (\phi_1|_{\neg p} \vee \phi_2|_{\neg p}))$$
is a deterministic disjunction, and the internal disjunctions $\phi_1|_p \vee \phi_2|_p$ and $\phi_1|_{\neg p} \vee \phi_2|_{\neg p}$ are less non-deterministic since they are interpreted on a smaller set of propositional variables.

DEFINITION 1.5 (Circuit for a d-DNNF formula). *Given a d-DNNF formula ϕ, the circuit for ϕ is the aritmetic expression circuit(ϕ) that computes $\#\text{SAT}(\phi)$ recursively defined as follows:*

- *if ϕ is a literal p or $\neg p$ then circuit(l) is* 1
- *if ϕ is $\phi_1 \wedge \phi_2$, then circuit(ϕ) = circuit(ϕ_1) $\cdot$ circuit(ϕ_2)*
- *if ϕ is $\phi_1 \vee \phi_2$, then circuit(ϕ) = circuit(ϕ_1) $\cdot 2^{n_2}$ + circuit(ϕ_2) $\cdot 2^{n_1}$, where n_1 (resp. n_2) is the number of propositional variables that occour in ϕ_1 (resp. ϕ_2) but not in ϕ_2 (resp. phi_1),*

EXAMPLE 1.7. *Let us transform $\phi = (A \vee B) \wedge (C \vee D) \wedge (\neg D \vee E)$ in d-DNNF. First notice that ϕ is the conjunction of two formulas ϕ_1 and ϕ_2 that do not share common variables. So the main conjunciton does not require any transformation, and we need to transform in d-DNNF the two sub-formula ϕ_1 and ϕ_2.*

$$\phi_1 = A \vee B$$
$$\phi_1 = (C \vee D) \wedge (\neg D \vee E)$$

ϕ_2 is not deterministic, since there is an interpretation that satisfies both disjunct A and B (i.e., the interpretation that makes both A and B true) so we have to apply

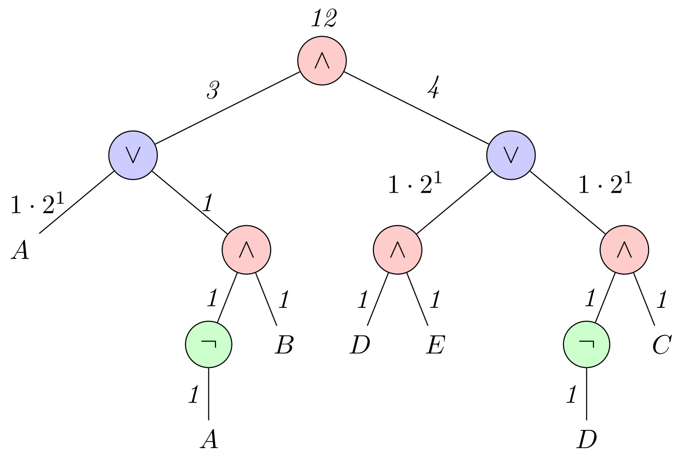


FIGURE 3. The formula tree of the d-DNNF formula $(A \vee (\neg A \wedge B)) \wedge ((D \wedge E) \vee (\neg D \wedge C))$. The numbers on the arcs represent the model counting of the corresponding subformula. To compute the model counting of the entire formula is is sufficient to start from the bottom, assigning a 1 to every proposition, and propagate up $\wedge$ nodes by multiplying the model counting of the subformulas (using property (1.4)) and propagating up $\vee$ nodes by summing the mc of the subformulas multiplied by 2^m where m is the number of variable occouring in the other subformula composing the or (property (2) of Proposition 1.6)

Shannon's expansion on some proposition of ϕ_1. Let us consider A. By Shannon's expansion we obtain

$$(A \wedge (\top \vee B)) \vee (\neg A \wedge (\bot \vee B))$$

and simplifying we obtain:

$$\phi'_1 = A \vee (\neg A \wedge B)$$

Notice that ϕ'_1 is deterministic, since every interpretation either falsify A or $\neg A$, with implies that there is no interpretation that simultaneously satisfies both the disjuncts. Furthermore every conjunction in ϕ'_1 is such that the two conjuncts ($A \wedge \top$ and $\neg A \wedge B$) do not share propositional variables. So ϕ'_1 is in d-DNNF.

Let us not tranform ϕ_2 in d-DNNF. Notice that ϕ_2 is the conjunciton of two formulas that share the propositonal variable D. Therefore we have to apply Shannon expansion on ϕ_2 w.r.t. the propositional variable D. We obvtain:

$$(D \wedge (C \vee \top) \wedge (\bot \vee E) \vee (\neg D \wedge (C \vee \bot) \wedge (\top \vee E))$$

which is equivalent to

$$\phi'_2 = (D \wedge E) \vee (\neg D \wedge C)$$

Therefore the original formula ϕ is equivalent to $\phi' = \phi'_1 \wedge \phi'_2$ which is in d-DNNF. The formula tree of ϕ' and how model counting can be obtained from the associated circuit is shown in Figure 3.

5. Approximate algorithm for model counting

In this section we describe one of the most basic algorithm for approximate model counting. The algorithm called ApproxCount has beed introduced in Wei and Selman 2005. The algorithm is based on a method called SampleSat for uniformly sampling models of a formula ϕ. Let us first introduce the SampleSat algorithm.

5.1. SampleSat.

SampleSat algorithm is based on random walk strategies. It requires in input a formula ϕ in CNF, i.e., a set of clauses $\{C_1, \dots, C_n\}$. Random walk (RW) strategy to search for a truth assignment that satisfies a formula ϕ starts from a random truth assignment. At every iteration the if the current assignment satisfies a formula then it is returned; otherwise one unsatisfied clause C_i is chosen uniformly at random from ϕ and a variable in the clause is chosen by some heuristic. The value of the variable is flipped. The algorithm repeats these steps until a satisfying assignment is reached.

Algorithm 2 SampleSat

```
1: $\mathcal{I} \leftarrow$ random assignments to the variabels of $\phi$
2: while true do
3:     if $\mathcal{I} \models \phi$ then
4:         return $\mathcal{I}$
5:     else
6:         $C \leftarrow$ random clause $C \in \phi$ such that $\mathcal{I} \not\models C$
7:         $p \leftarrow$ SelectVar($C$);
8:         $\mathcal{I}(p) \leftarrow 1 - \mathcal{I}(p)$                ▷ Flip the truth value of $p$ in $\mathcal{I}$
9:     end if
10: end while
```

The function SelectVar(C) select a literal from the clause C. A random selection of the literal when clauses are longer than 2 does not result in an unbias choice. It has been shown in Selman, Kautz, Cohen, et al. 1993 that using ϵ-greedy strategy provide a better behaviour (but other stratecies are also possible and could lead to better results).

For an assignment $\mathcal{I}$, the *break degree* of a propositional variable p is the number of clauses that are true in $\mathcal{I}$ and become false if we change the truth value of p.

$$\textit{break}(p, \mathcal{I}) = |\{C' \in \phi \mid \mathcal{I} \models C' \text{ and } \mathcal{I}_{flip(p)} \not\models C'\}$$

The SelectVar(C) method proposed by Selman, Kautz, Cohen, et al. 1993 selects a variable with break degree equal to 0 if there exists one, otherwise it selects the variable of C with lower break degree with probability $1 - \epsilon$ and a random literal with probability ϵ. (for some small ϵ). Though this strategy does not guarantee unbiased sampleing it is an improvement w.r.t., random sampling. The algorithm is shown in Algorithm 3. Advanced method are described in Selman, Kautz, Cohen, et al. 1993.

Let us now describe the ApproxCount algorithm, which is shown ,in Algorithm 4, The algorithm is based on the following intuition. Let S be a set of models of ϕ and S_p the subset of S that assigns p to true, I.e, $S_p = \{\mathcal{I} \in S \mid I(p) = \mathsf{True}\}$.

Algorithm 3 SelectVar($C, \mathcal{I}$)

```
1: if there is a $p$ in $C$ with $break(p, \mathcal{I}) = 0$ then
2:     return $p$
3: else
4:     with probability $1 - \epsilon$ return $\mathrm{argmin}_{p \in C}\, break(p, \mathcal{I})$
5:     with probability $\epsilon$ return a random propositional variable of $C$
6: end if
```

If S is a good sampling of the models of ϕ then we can estimate

$$\frac{\#\mathrm{SAT}(\phi|_p)}{\#\mathrm{SAT}(\phi)} \approx \frac{|S_p|}{|S|}$$

From which we have that

$$\#\mathrm{SAT}(\phi) = \frac{|S|}{|S_p|} \cdot \#\mathrm{SAT}(\phi_p)$$

Which means that we can appriximate the model counting of a formula ϕ by multiplying the model counting of a simpler formula ϕ_p by the factor $\frac{|S|}{|S_p|}$. The same resoning can be done by considering the set $S_{\neg p} = \{I \in S \mid \mathcal{I} \models \neg p\}$, obtaining a second approximation of $\#\mathrm{SAT}(\phi)$:

$$\#\mathrm{SAT}(\phi) = \frac{|S|}{|S_{\neg p}|} \cdot \#\mathrm{SAT}(\phi_{\neg p})$$

For every literal l we call $\frac{|S|}{|S_l|}$ the *muliplying factor* associated to the literal l. Let $l_1, \ldots, l_k$ be a set of non contraddictory literals, then by iterating the above argument we obtain the approximation

$$\#\mathrm{SAT}(\phi) = \#\mathrm{SAT}(\phi|_{l_1,\ldots,l_k}) \prod_{i=1}^{k} \frac{|S^{(i)}|}{|S^{(i)}_{l_i}|} \tag{7}$$

Where $S^{(i)}$ is the assignments that satisfies $\phi|_{l_1 \ldots, L_{i-1}}$. Notice that at eery iteration we sample different models. The ApproxCount algorithm uses this method to approximate $\#\mathrm{SAT}(\phi)$. Finally let us see a small example on how ApproxCount.

Example 1.8. *Let us ppply* ApproxCount *to estimate the number of models of the following formulas and compare the the exact solution.*

$$(p \wedge r) \vee (q \wedge \neg s)$$

Run ApproxCount *with input* $\phi = (p \wedge r) \vee (q \wedge \neg s)$ *and* $k = 2$

(1) *Repeatetly all* SampleSat *to obtain a set S of n models for ϕ. You can decide the number n of models that you want. The larger n the better the approximation.*
(2) *suppose that* $S = \{1010, 1100, 0110, 1110, 0100\}$*, (an interpretation on p, q, r, s is represented with the 4-bit value $\mathcal{I}(p)\mathcal{I}(q)\mathcal{I}(r)\mathcal{I}(s)$);*
(3) *select a propositional variable $x \in \{p, q, r, s\}$ to split in order to minimize $(|S_x| - |S_{\neg x}|)^2$. The proposiitional variables that minimize this difference are p and r. Suppose that we select p*
(4) $S_p = \{1010, 1100, 1110\}$
(5) $S_{\neg p} = \{0110, 0100\}$

Algorithm 4 ApproxCount ϕ in CNF

```
 1: $m \leftarrow 1$                                             ▷ $m$ is called the multiplier factor
 2: while $|props(\phi)| \geq k$ do                  ▷ When $\phi$ is small we apply exact method
 3:     $S \leftarrow$ SampleSat($\phi$) $n$ times   ▷ Select $n$ models for $\phi$ the larger the better
 4:     $p \leftarrow$ Select a propositional variable of $\phi$      ▷ Heuristic: Choose $p$ that
                                                                 maximizes $(|S_p| - |S_{\neg p}|)^2$
 5:     $S_p \leftarrow \{\mathcal{I} \in S \mid \mathcal{I}(p) = True\}$
 6:     $S_{\neg p} \leftarrow \{\mathcal{I} \in S \mid \mathcal{I}(p) = False\}$
 7:     if $|S_p| \geq |S_{\neg p}|$ then
 8:         $m \leftarrow m \cdot \frac{|S|}{|S_p|}$
 9:         $\phi \leftarrow \phi|_p$
10:     else
11:         $m \leftarrow m \cdot \frac{|S|}{|S_{\neg p}|}$
12:         $\phi \leftarrow \phi|_{\neg p}$
13:     end if
14: end while
15: return $m \cdot \#\textsc{sat}(\phi)$          ▷ $\#\textsc{sat}(\phi)$ is computed with an exact method
```

(6) *since* $|S_P| > |S_{\neg p}|$, *we choose to multiplication fuctor* $\frac{|S|}{|S_p|} = \frac{5}{3}$.
(7) $m = \frac{5}{3}$.
(8) $\phi = \phi|_p = (r \vee (q \wedge \neg s)$
(9) *repeatetly call* SampleSat *in order to sample a new set of models for* ϕ
(10) *suppose that* $S = \{100, 010, 110\}$
(11) *the propositional variable that minimies* $(|S_x| - |S_{\neg x}|)^2$ *are* q *and* r. *Suppose that we select* q
(12) $S_q = \{100, 110\}$
(13) $S_{\neg q} = \{010\}$
(14) *since* $|S_q| > |S_{\neg q}|$ *we select the multiplicative factor* $\frac{|S|}{|S_q|} = \frac{3}{2}$
(15) $m = m \cdot \frac{3}{2} = \frac{5}{2}$
(16) $\phi = \phi|_q = (r \vee \neg s)$
(17) *since* ϕ *contains* $2 \leq k$ *variables, we exit the wile and return* $m \cdot \#\textsc{sat}(r \vee \neg s) = \frac{5}{2} \cdot 3 = 7.5$

To understand quality of the result and measure the error let us compare the output of ApproxCount *and the exact value of* #sat *computed via truth table:*

p	q	r	s	$(p \wedge r) \vee (q \wedge \neg s)$
1	1	1	1	1
1	1	1	0	1
1	1	0	1	0
1	1	0	0	1
1	0	1	1	1
1	0	1	0	1
1	0	0	1	0
1	0	0	0	0
0	1	1	1	0
0	1	1	0	1
0	1	0	1	0
0	1	0	0	1
0	0	1	1	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	0
				7

The error is around 7%.

5.2. Graph Neural Networks for Propositional Model Counting. Saveri and Bortolussi 2022 TO DO

6. Exercises

Exercise 1:

Explain the difference between #sat(ϕ) and #sat$(\phi, \mathcal{P})$ when $\mathcal{P}$ is a set of propositional variables larger than the set of propositional variables that appears in ϕ.

Exercise 2:

Find an example in which $\phi \models \psi$ but #sat$(\phi) >$ #sat(ψ).

Exercise 3:

Prove that if ϕ contains p, then $\phi|_p$ is equivalent to $\phi \wedge p$. Provide a countere-sample of this property when p does not occour in ϕ.

Exercise 4:

Given a set of propositional variables $p_1, \ldots, p_n$, describe a method to produce a formula that has exactly k models for every $k \leq 2^n$.

Solution Let $\boldsymbol{b} = b_1 \ldots b_n$ be the bitwise representation of any the integer h between 0 and $2^n - 1$ (included). Let us define

$$\phi_h = \bigwedge_{\substack{i=1 \\ b_i=1}}^{n} p_i \wedge \bigwedge_{\substack{i=1 \\ b_i=0}}^{n} \neg p_i$$

Notice that the formula ϕ_h is satisfied by a single interpretation, i.e., the interpretation $\mathcal{I}$, such that $\mathcal{I}(p_i) = True$ if and only if $b_i = 1$. Furthermore, for every $g \neq h$, we have that there is no model that satisfies both ϕ_g and ϕ_h. We can therefore define the formula

$$\phi_{\leq k} = \bigvee_{0=1}^{k-1} \phi_h$$

which will be satisfied by exaclty k models. $\Box$

Exercise 5:

Use CPD to count the models of the formulas $(A \to C) \wedge (B \to C)$ and the formula $(A \vee B) \to C$. **Solution** The two formulas need to be tranformed in CNF.

They both are transformed in the following set of clauses:

$$(\neg A, B\}, \{\neg B, C\}$$

Let us run CPD on them.

```
CPD($\phi = \{\{\neg A, B\}, \{\neg B, C\}\}, n = 0$)
  unit propagation is not applicable since there are no unit clauses
  select the literal $\neg A$
  $\phi_{\neg A} = \{\{\neg B, C\}\}$
  CPD($\phi = \{\{\neg B, C\}\}, n = 1$)
    unit propagation is not applicable since there are no unit clauses
    select the literal $\neg B$
    $\phi_{\neg B} = \{\}$
    CPD($\phi = \{\}, n = 2$)
      return $2^{|\mathcal{P}|-n} = 2^{3-2} = 2$ ($\mathcal{P} = A, B, C\}$
    select the literal $B$
    $\phi_B = \{\{C\}\}$
    CPD($\phi = \{\{C\}\}, n = 2$)
      apply unit propagation obtaining $\{\}$ and $n = 3$
      return $2^{|\mathcal{P}|-n} = 2^0 = 1$
    return $2 + 1 = 3$
  select the literal $A$
  $\phi_{\neg A} = \{\{B\}, \{\neg B, C\}\}$
  CPD($\phi = \{\{B\}\{\neg B, C\}\}, n = 1$)
    apply unit propagation obtaining $\{\}$ and $n = 3$
    return $2^{|\mathcal{P}|-n} = 2^0 = 1$
  return $3 + 1 = 4$
```

$\Box$

Exercise 6:

Use CPD to count the models of the formula $(A \to B) \wedge (B \to C) \wedge (C \to D)$ **Solution** CPD require the formula in CNF. So we first need to transform $(A \to B) \wedge (B \to C) \wedge (C \to D)$ in CNF, which results in the forllowing clauses:

$$\{\neg A, B\}, \{\neg B, C\}, \{\neg C, D\}$$

```
CPD($\phi = \{\{\neg A, B\}, \{\neg B, C\}, \{\neg C, D\}\}, n = 0$)
  unit propagation is not applicable since there are no unit clauses
  select the literal $\neg A$
  $\phi_{\neg A} = \{\{\neg B, C\}, \{\neg C, D\}\}$
  CPD($\phi = \{\{\neg B, C\}, \{\neg C, D\}\}, n = 1$)
    unit propagation is not applicable since there are no unit clauses
    select the literal $\neg B$
    $\phi_{\neg B} = \{\{\neg C, D\}\}$
    CPD($\phi = \{\{\neg C, D\}\}, n = 2$)
      unit propagation is not applicable since there are no unit clauses
      select the literal $\neg C$
      $\phi_{\neg C} = \{\}$
      CPD($\phi = \{\}, n = 3$)
      return $2^{|\mathcal{P}|-n} = 2^1 = 2$ ($\mathcal{P} = A, B, C, D\}$
      select the literal $C$
      $\phi_C = \{D\}$
      CPD($\phi = \{D\}, n = 3$)
        apply unit propagation which returns $\phi = \{\}$ and $n = 4$
      return $2^{|\mathcal{P}|-n} = 2^0 = 1$
    return $2 + 1 = 3$
    select the literal $B$
    $\phi_B = \{\{C\}, \{\neg C, D\}\}$
    CPD($\phi = \{\{C\}, \{\neg C, D\}\}, n = 2$)
      apply unit propagation obtaining $\{\}$ and $n = 4$
    return $2^{|\mathcal{P}|-n} = 2^0 = 1$
  return $3 + 1 = 4$
  select the literal $A$
  $\phi_A = \{\{B\}\{\neg B, C\}, \{\neg C, D\}\}$
  CPD($\phi = \{\{B\}\{\neg B, C\}, \{\neg C, D\}\}, n = 1$)
    apply unit propagation obtaining $\{\}$ and $n = 4$
  return $2^{|\mathcal{P}|-n} = 2^0 = 1$
return $4 + 1 = 5$
```

To check the correctness of the result let us compute the truth table explicitly:

A	B	C	D	(A	→	B)	∧	((B	→	C)	∧	(C	→	D))
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	0	1	1	1	0	1	1	1	0	1	0	0
1	1	0	1	1	1	1	0	1	0	0	0	0	1	1
1	1	0	0	1	1	1	0	1	0	0	0	0	1	0
1	0	1	1	1	0	0	0	0	1	1	1	1	1	1
1	0	1	0	1	0	0	0	0	1	1	0	1	0	0
1	0	0	1	1	0	0	0	0	1	0	1	0	1	1
1	0	0	0	1	0	0	0	0	1	0	1	0	1	0
0	1	1	1	0	1	1	1	1	1	1	1	1	1	1
0	1	1	0	0	1	1	0	1	1	1	0	1	0	0
0	1	0	1	0	1	1	0	1	0	0	0	0	1	1
0	1	0	0	0	1	1	0	1	0	0	0	0	1	0
0	0	1	1	0	1	0	1	0	1	1	1	1	1	1
0	0	1	0	0	1	0	0	0	1	1	0	1	0	0
0	0	0	1	0	1	0	1	0	1	0	1	0	1	1
0	0	0	0	0	1	0	1	0	1	0	1	0	1	0
8	8	8	8	8	12	8	**5**	8	12	8	8	8	12	8

□

Exercise 7:
Count the models that satisfy the formula $(A \to B \vee C) \wedge (C \to D \vee E)$

Solution The formula is a conjunction of two formulas that share the propositional C variable. We can therefore we can apply the Shannon reduction on C obtaining the formula:

$$\begin{aligned} C \wedge & (A \to B \vee \top) \wedge (\top \to D \vee E) \vee \\ \neg C \wedge & (A \to B \vee \bot) \wedge (\bot \to D \vee E) \end{aligned}$$

which is equivalent to

$$\begin{aligned} C \wedge & (A \to B \vee \top) \wedge (D \vee E) \vee \\ \neg C \wedge & (A \to B) \wedge (\bot \to D \vee E) \end{aligned}$$

Since the disjunction is deterministic (since only one of the disjunct holds), we can compute separately the number of models of the formulas and then sum up.

The number of models of $C \wedge (A \to B \vee \top) \wedge (D \vee E)$ is $1 \cdot 4 \cdot 3 = 12$ The number of models of $\neg C \wedge (A \to B) \wedge (\bot \to D \vee E)$ is $1 \cdot 3 \cdot 4 = 12$ In total we have $12 + 12 = 24$ models.

Alternatively one can fill the truth table

A	B	C	D	E	(A	→	(B	∨	C))	∧	(C	→	(D	∨	E))
T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
T	T	T	T	F	T	T	T	T	T	T	T	T	T	T	F
T	T	T	F	T	T	T	T	T	T	T	T	T	F	T	T
T	T	T	F	F	T	T	T	T	T	F	T	F	F	F	F
T	T	F	T	T	T	T	T	T	F	T	F	T	T	T	T
T	T	F	T	F	T	T	T	T	F	T	F	T	T	T	F
T	T	F	F	T	T	T	T	T	F	T	F	T	F	T	T
T	T	F	F	F	T	T	T	T	F	T	F	T	F	F	F
T	F	T	T	T	T	T	F	T	T	T	T	T	T	T	T
T	F	T	T	F	T	T	F	T	T	T	T	T	T	T	F
T	F	T	F	T	T	T	F	T	T	T	T	T	F	T	T
T	F	T	F	F	T	T	F	T	T	F	T	F	F	F	F
T	F	F	T	T	T	F	F	F	F	F	F	T	T	T	T
T	F	F	T	F	T	F	F	F	F	F	F	T	T	T	F
T	F	F	F	T	T	F	F	F	F	F	F	T	F	T	T
T	F	F	F	F	T	F	F	F	F	F	F	T	F	F	F
F	T	T	T	T	F	T	T	T	T	T	T	T	T	T	T
F	T	T	T	F	F	T	T	T	T	T	T	T	T	T	F
F	T	T	F	T	F	T	T	T	T	T	T	T	F	T	T
F	T	T	F	F	F	T	T	T	T	F	T	F	F	F	F
F	T	F	T	T	F	T	T	T	F	T	F	T	T	T	T
F	T	F	T	F	F	T	T	T	F	T	F	T	T	T	F
F	T	F	F	T	F	T	T	T	F	T	F	T	F	T	T
F	T	F	F	F	F	T	T	T	F	T	F	T	F	F	F
F	F	T	T	T	F	T	F	T	T	T	T	T	T	T	T
F	F	T	T	F	F	T	F	T	T	T	T	T	T	T	F
F	F	T	F	T	F	T	F	T	T	T	T	T	F	T	T
F	F	T	F	F	F	T	F	T	T	F	T	F	F	F	F
F	F	F	T	T	F	T	F	F	F	T	F	T	T	T	T
F	F	F	T	F	F	T	F	F	F	T	F	T	T	T	F
F	F	F	F	T	F	T	F	F	F	T	F	T	F	T	T
F	F	F	F	F	F	T	F	F	F	T	F	T	F	F	F

and counth the number of assignments that satisfy the formula. They are 24. □

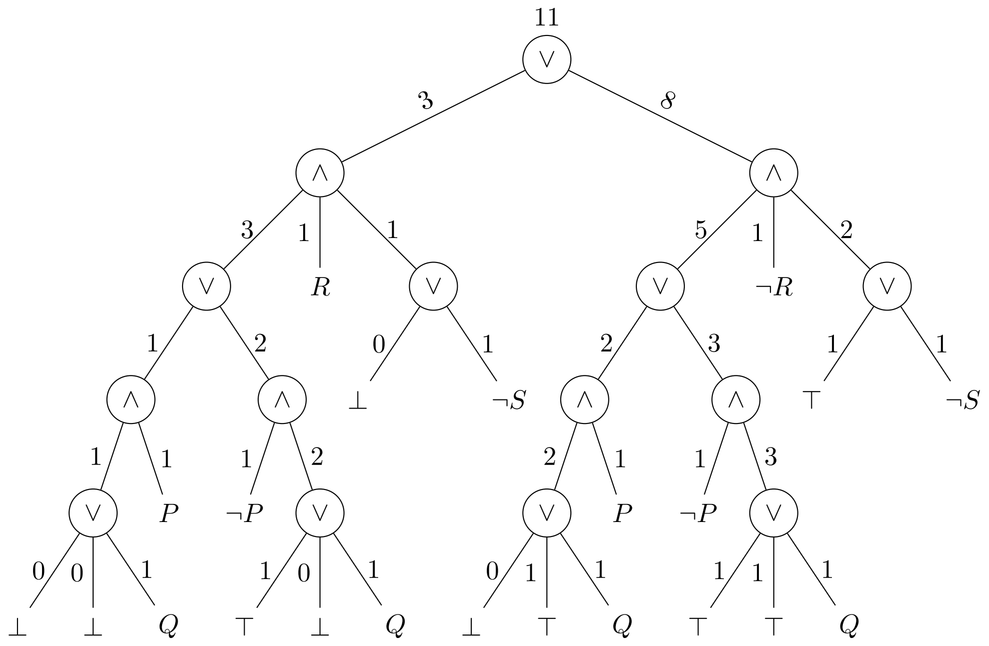
Exercise 8:
Transform the following formula in d-DNNF.

$$(P \vee Q) \wedge (R \vee S) \wedge (\neg S \vee T)$$

Exercise 9:
Compute $\#SAT((P \wedge R \rightarrow Q) \wedge (\neg R \vee \neg S))$ using knowledge compilation method.

Solution We first have to transform the formula in deterministic decomposable negated normal form (d-DNNF).

$$
\begin{aligned}
(P \wedge R \to Q) \wedge (\neg R \vee \neg S) &\equiv \\
(\neg P \vee \neg R \vee Q) \wedge (\neg R \vee \neg S) &\equiv \\
((\neg P \vee \bot \vee Q) \wedge (\bot \vee \neg S) \wedge R) \vee ((\neg P \vee \bot R \vee Q) \wedge (\top \vee \neg S) \wedge \neg R) &\equiv \\
(((\bot \vee \bot \vee Q) \wedge P) \vee ((\top \vee \bot \vee Q) \wedge \neg P) \wedge (\bot \vee \neg S) \wedge R) &\vee \\
(((\bot \vee \top \vee Q) \wedge P) \vee ((\top \vee \top \vee Q) \wedge \neg P) \wedge (\top \vee \neg S) \wedge \neg R)&
\end{aligned}
$$



Let us verify the correctenss of the result by compiling the truth table

P	Q	R	S	((P	∧	R)	→	Q)	∧	(¬	R	∨	¬	S)
T	T	T	T	T	T	T	T	T	F	F	T	F	F	T
T	T	T	F	T	T	T	T	T	T	F	T	T	T	F
T	T	F	T	T	F	F	T	T	T	T	F	T	F	T
T	T	F	F	T	F	F	T	T	T	T	F	T	T	F
T	F	T	T	T	T	T	F	F	F	F	T	F	F	T
T	F	T	F	T	T	T	F	F	F	F	T	T	T	F
T	F	F	T	T	F	F	T	F	T	T	F	T	F	T
T	F	F	F	T	F	F	T	F	T	T	F	T	T	F
F	T	T	T	F	F	T	T	T	F	F	T	F	F	T
F	T	T	F	F	F	T	T	T	T	F	T	T	T	F
F	T	F	T	F	F	F	T	T	T	T	F	T	F	T
F	T	F	F	F	F	F	T	T	T	T	F	T	T	F
F	F	T	T	F	F	T	T	F	F	F	T	F	F	T
F	F	T	F	F	F	T	T	F	T	F	T	T	T	F
F	F	F	T	F	F	F	T	F	T	T	F	T	F	T
F	F	F	F	F	F	F	T	F	T	T	F	T	T	F

□

Exercise 10:

Codify in $\#SAT$ the problem of counting how many different k-tuples $(i_1, \dots, i_5)$ of integers ≤ 8 such that $i_1 \leq \imath_2 \leq \cdots \leq i_5$.

Solution Let p_{ij}, for $1 \leq i \leq 5$ and $1 \leq j \leq 8$, be a propositional variable that stands for "there is a j in position i". The set of strings that satisfy the conditions of the exercise are those that satisfy the following formulas:

$$\bigwedge_{i=1}^{5} \bigvee_{j=1}^{8} p_{ij} \qquad \text{in every position there is at least one digit}$$

$$\bigwedge_{i=1}^{5} \bigwedge_{j<k=1}^{8} \neg(p_{ij} \wedge p_{ik}) \qquad \text{in every position there is at most one digit}$$

$$\bigwedge_{i=1}^{4} \bigwedge_{j=1}^{8} \left(p_{ij} \rightarrow \bigvee_{k=j}^{8} p_{i+1,k} \right) \qquad \text{In the next position can occour only number larger or equal to the one present in the current position}$$

□

Exercise 11:

Codify in $\#SAT$ the problem of counting how many different strings can be made by reordering the letters of the word "SUCCESS"?

Solution The reordering of the string "SUCCESS" is any string that contains three "S"s, two "C"s, one "U" and one "E". Let us introduce the following propositional variables

S_i	$1 \leq i \leq 7$	there is an "S" in position i
C_i	$1 \leq i \leq 7$	there is an "C" in position i
U_i	$1 \leq i \leq 7$	there is an "U" in position i
E_i	$1 \leq i \leq 7$	there is an "E" in position i

We have the following axioms:

$$\bigwedge_{i=1}^{7} S_i \vee C_i \vee U_i \vee E_i \qquad \text{In each position there is at least one letter}$$

$$\bigwedge_{i \neq j} \neg(U_i \wedge U_j) \qquad \text{At most one "U"}$$

$$\bigwedge_{i \neq j} \neg(E_i \wedge E_j) \qquad \text{At most one "E"}$$

$$\bigwedge_{i \neq j \neq k} \neg(C_i \wedge C_j \wedge C_k) \qquad \text{At most two "C"}$$

$$\bigwedge_{i \neq j \neq k \neq h} \neg(S_i \wedge S_j \wedge S_k \neg S_h) \qquad \text{At most tree "S"}$$

□

Exercise 12:

To buy a computer system, a customer can choose one of 4 monitors, one of 2 keyboards, one of 4 computers and one of 3 printers. But only certain combinations are allowed. Provide some example of how you can express constraints on possible combinations with propositional formulas, and codify the problem of determining the number of possible systems that a customer can choose from as model counting.

Exercise 13:

Suppose you have three coins: the faces of the first coin are black and white, the faces of the second coin are yellow and green, and the faces of the third coin are blue and red. In an experiment you toss the first coin; if you obtain a black you toss the second coin otherwise you toss the third coin. What are the number of possible outcomes? Encode the problem of counting the possible outcomes of this simple experiment in the problem of model counting a set of formulas

Exercise 14:

Transform the formula $(A \vee B) \wedge (\neg B \vee C) \wedge (\neg D \vee E)$ in d-DDNF form, and use it for model counting. **Solution** Split in

(1) $(A \vee B) \wedge (\neg B \vee C)$
(2) $(\neg D \vee E)$

since they don't have common variables, and treat them separately

(1) let us consider $(A \vee B) \wedge (\neg B \vee C)$

$(A \vee B) \wedge (\neg B \vee C)$	Apply Shannon's expansion on B
$(B \wedge (A \vee \top) \wedge (\bot \vee C)) \vee (\neg B \wedge (A \vee \bot) \wedge (\top \vee C))$	in d-DDNF

The last formula has:

$$(1 \cdot (1+1) \cdot (0+1)) + (1 \cdot (1+0) \cdot (1+1) = 4$$

models

(2) Let us now consider $(\neg D \vee E)$

$\neg D \vee E$	Apply Shannon's expansion on D
$(D \wedge (\bot \vee E)) \vee (\neg D \wedge (\top \vee E))$	in d-DDNF

The last formula has:

$$(1 \cdot (0+1)) + (1 \cdot (1+1)) = 3$$

The total number of models is $4 \cdot 3 = 12$. $\square$

Exercise 15:

Transform the following formula in d-DNNF.

$$(P \vee Q) \wedge (R \vee S) \wedge (\neg S \vee T)$$

Solution Notice that the formula is the conjunction of two formulas that do not have common propositional variables. We can therefore proceed to transform each subformula in d-DNNF.

$(P \vee Q)$ is transformed (via Shannon's expansion) in

$$P \wedge (\top \vee Q) \vee \neg P \wedge (\bot \vee Q)$$

$(R \vee S) \wedge (\neg S \vee T)$ instead is the conjunciton of two formulas that have one propositional variable in common (i.e., S). Therefore we have to consider the two cases in which S is true and S is false. We therefore rewrite the formula by using the Shannon's expansion

$$(S \wedge (R \vee \top) \wedge (\bot \vee T)) \vee (\neg S \wedge (R \vee \bot) \wedge (\top \vee T))$$

THe result d-DNNF transofmraiton is the conjunction of the two resutls:, i.e.,

$$\begin{array}{r} P \wedge (\top \vee Q) \vee \neg P \wedge (\bot \vee Q) \wedge \\ (S \wedge (R \vee \top) \wedge (\bot \vee T)) \vee (\neg S \wedge (R \vee \bot) \wedge (\top \vee T)) \end{array}$$

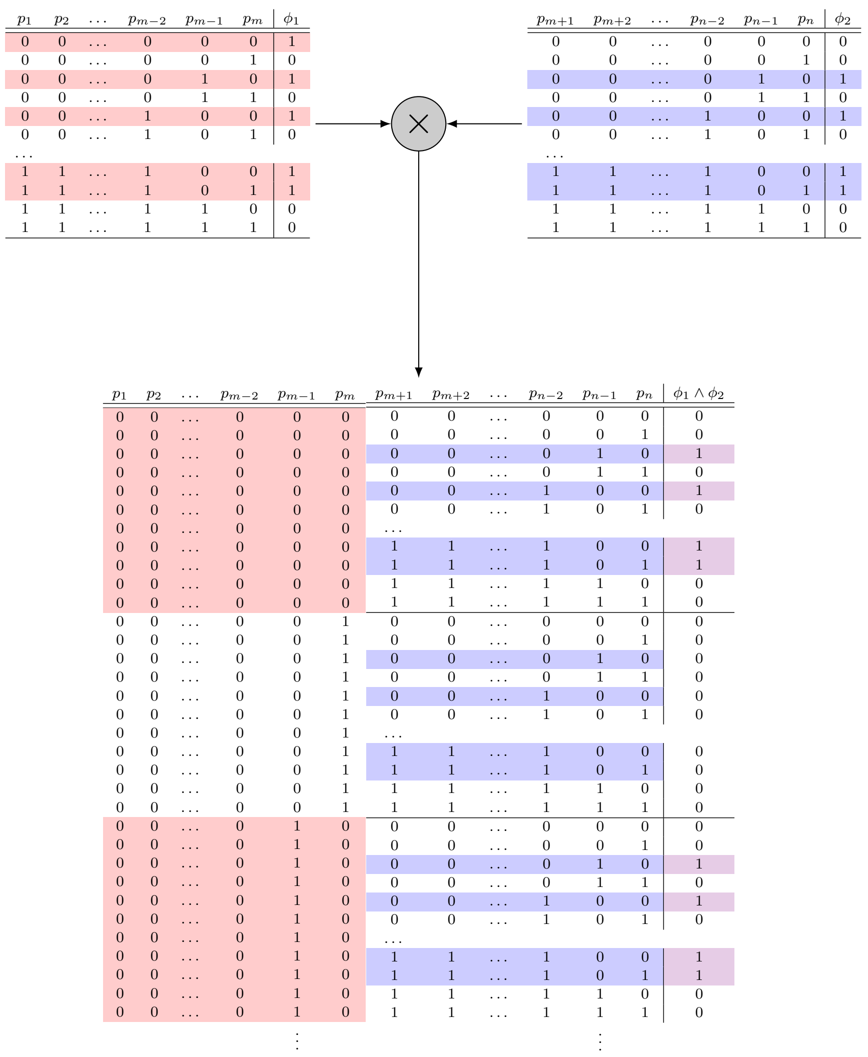
□

Exercise 16:

Explain in at most 10 lines, why if ϕ_1 and ϕ_2 don't share propositional variabels then $\#SAT(\phi_1 \wedge \phi_2) = \#SAT(\phi_1) \cdot \#SAT(\phi_2)$

Solution A model $\mathcal{I}$ of $\phi_1 \wedge \phi_2$ is an assignment of all the variables in ϕ_1 and ϕ_2 that satisfies both ϕ_1 and $\phi_@$. Since ϕ_1 and ϕ_2 do not have variables in common, we can split the assignment $\mathcal{I}$ in two assignments $\mathcal{I}_1$ and $\mathcal{I}_2$ where $\mathcal{I}_1$ assigns the variables in ϕ_1 and $\mathcal{I}_2$ the variables in ϕ_2. Furthermore, we have that $\mathcal{I}_1 \models \phi_1$ and $\mathcal{I}_2 \models \phi_2$. Therefore all the models of $\phi_1 \wedge \phi_2$ corresponds to all the pairs $\mathcal{I}_1$ and $\mathcal{I}_2$ where $\mathcal{I}_i$ is a model of ϕ_i for $i = 1, 2$.

Graphically:



p_1	p_2	$\dots$	p_{m-2}	p_{m-1}	p_m	ϕ_1
0	0	$\dots$	0	0	0	1
0	0	$\dots$	0	0	1	0
0	0	$\dots$	0	1	0	1
0	0	$\dots$	0	1	1	0
0	0	$\dots$	1	0	0	1
0	0	$\dots$	1	0	1	0
$\dots$						
1	1	$\dots$	1	0	0	1
1	1	$\dots$	1	0	1	1
1	1	$\dots$	1	1	0	0
1	1	$\dots$	1	1	1	0

p_{m+1}	p_{m+2}	$\dots$	p_{n-2}	p_{n-1}	p_n	ϕ_2
0	0	$\dots$	0	0	0	0
0	0	$\dots$	0	0	1	0
0	0	$\dots$	0	1	0	1
0	0	$\dots$	0	1	1	0
0	0	$\dots$	1	0	0	1
0	0	$\dots$	1	0	1	0
$\dots$						
1	1	$\dots$	1	0	0	1
1	1	$\dots$	1	0	1	1
1	1	$\dots$	1	1	0	0
1	1	$\dots$	1	1	1	0

$\times$

p_1	p_2	$\dots$	p_{m-2}	p_{m-1}	p_m	p_{m+1}	p_{m+2}	$\dots$	p_{n-2}	p_{n-1}	p_n	$\phi_1 \wedge \phi_2$
0	0	$\dots$	0	0	0	0	0	$\dots$	0	0	0	0
0	0	$\dots$	0	0	0	0	0	$\dots$	0	0	1	0
0	0	$\dots$	0	0	0	0	0	$\dots$	0	1	0	1
0	0	$\dots$	0	0	0	0	0	$\dots$	0	1	1	0
0	0	$\dots$	0	0	0	0	0	$\dots$	1	0	0	1
0	0	$\dots$	0	0	0	0	0	$\dots$	1	0	1	0
0	0	$\dots$	0	0	0	$\dots$						
0	0	$\dots$	0	0	0	1	1	$\dots$	1	0	0	1
0	0	$\dots$	0	0	0	1	1	$\dots$	1	0	1	1
0	0	$\dots$	0	0	0	1	1	$\dots$	1	1	0	0
0	0	$\dots$	0	0	0	1	1	$\dots$	1	1	1	0
0	0	$\dots$	0	0	1	0	0	$\dots$	0	0	0	0
0	0	$\dots$	0	0	1	0	0	$\dots$	0	0	1	0
0	0	$\dots$	0	0	1	0	0	$\dots$	0	1	0	0
0	0	$\dots$	0	0	1	0	0	$\dots$	0	1	1	0
0	0	$\dots$	0	0	1	0	0	$\dots$	1	0	0	0
0	0	$\dots$	0	0	1	0	0	$\dots$	1	0	1	0
0	0	$\dots$	0	0	1	$\dots$						
0	0	$\dots$	0	0	1	1	1	$\dots$	1	0	0	0
0	0	$\dots$	0	0	1	1	1	$\dots$	1	0	1	0
0	0	$\dots$	0	0	1	1	1	$\dots$	1	1	0	0
0	0	$\dots$	0	0	1	1	1	$\dots$	1	1	1	0
0	0	$\dots$	0	1	0	0	0	$\dots$	0	0	0	0
0	0	$\dots$	0	1	0	0	0	$\dots$	0	0	1	0
0	0	$\dots$	0	1	0	0	0	$\dots$	0	1	0	1
0	0	$\dots$	0	1	0	0	0	$\dots$	0	1	1	0
0	0	$\dots$	0	1	0	0	0	$\dots$	1	0	0	1
0	0	$\dots$	0	1	0	0	0	$\dots$	1	0	1	0
0	0	$\dots$	0	1	0	$\dots$						
0	0	$\dots$	0	1	0	1	1	$\dots$	1	0	0	1
0	0	$\dots$	0	1	0	1	1	$\dots$	1	0	1	1
0	0	$\dots$	0	1	0	1	1	$\dots$	1	1	0	0
0	0	$\dots$	0	1	0	1	1	$\dots$	1	1	1	0
				$\vdots$						$\vdots$		

□

Exercise 17:

Explain in at most 10 lines, why if ϕ_1 and ϕ_2 don't share propositional variabels then $\#SAT(\phi_1 \vee \phi_2) = \#SAT(\phi_1) \cdot 2^{|\mathcal{P}_2|} + \#SAT(\phi_2) \cdot 2^{|\mathcal{P}_1|}$, where $\mathcal{P}_1$ ad $\mathcal{P}_2$ are the set of propositional variables that occours in ϕ_1 and ϕ_2 respectively.

Exercise 18:

Find a formula for $\#SAT(\phi_1 \leftrightarrow \phi_2)$ under the ussumption that ϕ_1 and ϕ_2 don't share any propositional variable.

Solution Notice that $\phi_1 \leftrightarrow \phi_2$ is equivalent to $(\phi_1 \wedge \phi_2) \vee (\neg\phi_1 \wedge \neg\phi_2)$. Furthermore the set of models that satisfy $(\phi_1 \wedge \phi_2)$ is disjoint from the set of models that satisfy $(\neg\phi_1 \wedge \neg\phi_2)$. This implies that

$$\#SAT(\phi_1 \veebar \phi_2) = \#SAT(\phi_1 \wedge \phi_2) + \#SAT(\neg\phi_1 \wedge \neg\phi_2)$$

Since ϕ_1 and ϕ_2 don't share any propositional variable, then we have that

$$\#SAT(\phi_1 \veebar \phi_2) = \#SAT(\phi_1) \cdot \#SAT(\phi_2) + \#SAT(\neg\phi_1) \cdot \#SAT(\neg\phi_2)$$

Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be the set of propositional variables occurring in $\phi_!$ and ϕ_2 respectively, then we have that $\#SAT(\neg\phi_i) = 2^{|\mathcal{P}_i)} - \#SAT(\phi_i)$ for $i = 1, 2$. We can therefore conclude that :

$$\begin{aligned}\#SAT(\phi_1 \veebar \phi_2) &= \#SAT(\phi_1) \cdot \#SAT(\phi_2) + (2^{|\mathcal{P}_1|} - \#SAT(\phi_1)) \cdot (2^{|\mathcal{P}_2|} - \#SAT(\phi_2)) \\ &= 2^{|\mathcal{P}_1 \cup \mathcal{P}_2|} - 2^{|\mathcal{P}_1|}\#SAT(\phi_2) - 2^{|\mathcal{P}_2|}\#SAT(\phi_1) \\ &+ 2 \cdot \#SAT(\phi_1) \cdot \#SAT(\phi_2)\end{aligned}$$

□

Solution(alternative) According to the definition of exclusive or, we have that $\phi_1 \veebar \phi_2$ is equivsalent to

$$(\phi_1 \wedge \neg\phi_2) \vee (\neg\phi_1 \wedge \phi_2)$$

The above formula has the following "nice" properties. The disjunciton is deterministic, which means that there is no interpretations that satisfies both the formulas of the dijunction. I.e., the models of $(\phi_1 \wedge \neg\phi_2)$ are not models of $(\neg\phi_1 \wedge \phi_2)$ and viceversa. Furthermore the disjunciton is also smooth, i.e, the set of propositional variables in the two disjuncts are the same. The third "nice"property derives from the fact that ϕ_1 and ϕ_2 do not share any propositional variable, which means that, we can obtain the model counting of the conjunction as the product of the model counting fo the conjuncts. We can therefore conclude that

$$\#\text{SAT}(\phi_1 \veebar \phi_2) = s_1 \cdot (2^{n_2} - s_2) + s_2 \cdot (2^{n_1} - s_1)$$

where $s_i = \#\text{SAT}(\phi_i)$ and n_i is the number of propositional variables that appear in ϕ_i for $i = 1, 2$. □

Exercise 19:

Let $\veebar$ be the connective for exclusive or, i.e., $p \veebar q$ is equivalent to $(p \wedge \neg q) \vee (\neg p \wedge q)$. Express $\#SAT(\phi_1 \veebar \phi_2)$ in terms of $\#SAT(\phi_1)$ and $\#SAT(\phi_2)$ under the hypothesis that ϕ_1 and ϕ_2 don't share any propositional variable. Explain your solution.

Exercise 20:

Show that if $\models \phi_1 \equiv \phi_2$, then

$$\#SAT(\phi_1) = \#SAT(\phi_2) \cdot 2^{|\mathcal{P}_2 \setminus \mathcal{P}_1| - |\mathcal{P}_1 \setminus \mathcal{P}_2|}$$

where $\mathcal{P}_i$ is the set of propositional variables occourring in ϕ_i for $i = 1, 2$.

Solution Let us consider the special case in which $\mathcal{P}_2 \subseteq \mathcal{P}_1$, We prove that

$$\#SAT(\phi_1) = \#SAT(\phi_2) \cdot 2^{|\mathcal{P}_1 \setminus \mathcal{P}_2|}$$

Let $\mathcal{I}_2$ be an assignment to $\mathcal{P}_2$ that satisfies ϕ_2, and let $\mathcal{I}_1$ be an estension of $\mathcal{I}_2$ for the variables in $\mathcal{P}_1 \setminus \mathcal{P}_2$. We have that $\mathcal{I}_1 \models \phi$ and since $\models \phi_1 \equiv \phi_2$ then $\mathcal{I}_2 \models \phi_1$. Since there are $2^{|\mathcal{P}_1 \setminus \mathcal{P}_2|}$ different extensions of $\mathcal{I}_2$, we have that

$$\#SAT(\phi_1) \geq \#SAT(\phi_2) \cdot 2^{|\mathcal{P}_1 \setminus \mathcal{P}_2|}$$

. SInce we also have that $\models \neg\phi_1 \equiv \neg\phi_2$, for the same argument we can show that

$$\#SAT(\neg\phi_1) \geq \#SAT(\neg\phi_2) \cdot 2^{|\mathcal{P}_1 \setminus \mathcal{P}_2|}$$

. Which implies that

$$2^{|\mathcal{P}_1|} - \#SAT(\phi_1) \geq (2^{|\mathcal{P}_2} - \#SAT(\phi_2)) \cdot 2^{|\mathcal{P}_1 \setminus \mathcal{P}_2|}$$

Since $\mathcal{P}_2 \subseteq \mathcal{P}_2$ we can conlcude that

$$\#SAT(\phi_1) \leq \#SAT(\phi_2) \cdot 2^{|\mathcal{P}_1 \setminus \mathcal{P}_2|}$$

And therefore we can have the following lemma:

$$\begin{array}{r} \text{forall } \phi_1 \text{ and } \phi_2 \text{ if } \models \phi_1 \equiv \phi_2 \text{ and } \mathcal{P}_2 \subseteq \mathcal{P}_2 \\ \text{then } \#SAT(\phi_1) = \#SAT(\phi_2) \cdot 2^{|\mathcal{P}_1 \setminus \mathcal{P}_2|} \end{array} \tag{8}$$

We can now use lemma (8) to show the main result. First notice that if $\models \phi_1 \equiv \phi_2$, then $\models \phi_i \equiv \phi_1 \wedge \phi_2$ for $i = 1, 2$. By applying lemma (8) we have that

$$\#SAT(\phi_1 \wedge \phi_2) = \#SAT(\phi_1) \cdot 2^{|\mathcal{P}_1 \cup \mathcal{P}_2 \setminus \mathcal{P}_1|}$$
$$\#SAT(\phi_1 \wedge \phi_2) = \#SAT(\phi_2) \cdot 2^{|\mathcal{P}_1 \cup \mathcal{P}_2 \setminus \mathcal{P}_2|}$$

From which one can derive

$$\begin{aligned} \#SAT(\phi_1) \cdot 2^{|\mathcal{P}_1 \cup \mathcal{P}_2 \setminus \mathcal{P}_1|} &= \#SAT(\phi_2) \cdot 2^{|\mathcal{P}_1 \cup \mathcal{P}_2 \setminus \mathcal{P}_2|} \\ \#SAT(\phi_1) \cdot 2^{|\mathcal{P}_1 \setminus \mathcal{P}_1|} &= \#SAT(\phi_2) \cdot 2^{|\mathcal{P}_1 \setminus \mathcal{P}_2|} \\ \#SAT(\phi_1) &= \#SAT(\phi_2) \cdot 2^{|\mathcal{P}_1 \setminus \mathcal{P}_2| - |\mathcal{P}_1 \setminus \mathcal{P}_1|} \end{aligned}$$

□

Bibliography

Birnbaum, Elazar and Eliezer L Lozinskii (1999). "The good old Davis-Putnam procedure helps counting models". In: *Journal of Artificial Intelligence Research* 10, pp. 457–477.

Chavira, Mark and Adnan Darwiche (2008). "On probabilistic inference by weighted model counting". In: *Artificial Intelligence* 172.6-7, pp. 772–799.

Franco, John and Marvin Paull (1983). "Probabilistic analysis of the Davis Putnam procedure for solving the satisfiability problem". In: *Discrete Applied Mathematics* 5.1, pp. 77–87.

Gomes, Carla P., Ashish Sabharwal, and Bart Selman (2009). "Model Counting". In: *Handbook of Satisfiability*, pp. 633–654.

Holtzen, Steven, Guy Van den Broeck, and Todd Millstein (2020). "Scaling exact inference for discrete probabilistic programs". In: *Proceedings of the ACM on Programming Languages* 4.OOPSLA, pp. 1–31.

Saveri, Gaia and Luca Bortolussi (2022). "Graph Neural Networks for Propositional Model Counting". In: *arXiv preprint arXiv:2205.04423*.

Selman, Bart, Henry A Kautz, Bram Cohen, et al. (1993). "Local search strategies for satisfiability testing." In: *Cliques, coloring, and satisfiability* 26, pp. 521–532.

Wei, Wei and Bart Selman (2005). "A new approach to model counting". In: *International Conference on Theory and Applications of Satisfiability Testing*. Springer, pp. 324–339.