# Sum-Product-Quotient Networks

**Or Sharir**
The Hebrew University of Jerusalem
or.sharir@cs.huji.ac.il

**Amnon Shashua**
The Hebrew University of Jerusalem
shashua@cs.huji.ac.il

## Abstract

We present a novel tractable generative model that extends Sum-Product Networks (SPNs) and significantly boosts their power. We call it *Sum-Product-Quotient Networks* (SPQNs), whose core concept is to incorporate conditional distributions into the model by direct computation using quotient nodes, e.g. $P(A|B) = \frac{P(A,B)}{P(B)}$. We provide sufficient conditions for the tractability of SPQNs that generalize and relax the decomposable and complete tractability conditions of SPNs. These relaxed conditions give rise to an exponential boost to the expressive efficiency of our model, i.e. we prove that there are distributions which SPQNs can compute efficiently but require SPNs to be of exponential size. Thus, we narrow the gap in expressivity between tractable graphical models and other Neural Network-based generative models.

## 1 Introduction

Sum-Product Networks (SPNs)(Poon and Domingos, 2011) are a class of generative models capable of exact and tractable inference, where the probability function is directly modelled as a simple computational graph composed of just *weighted sum* and *product* nodes, known also as Arithmetic Circuits (Shpilka and Yehudayoff, 2010), following a strict set of constraints on its connectivity. SPNs have been applied to solve a wide range of tasks, e.g. image classification (Gens and Domingos, 2012), activity recognition (Amer and Todorovic, 2012, 2016), and missing data (Sharir et al., 2016). While SPNs have certain advantages in some areas, in cases where expressiveness is a limiting factor, they fall far behind contemporary generative models such as those leveraging neural networks as their inference engine (Uria et al., 2016; van den Oord et al., 2016b; Dinh et al., 2017).

Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS) 2018, Lanzarote, Spain. PMLR: Volume 84. Copyright 2018 by the author(s).

When SPNs were first introduced, it was hypothesized that perhaps all tractable distributions could be represented efficiently by SPNs. However, this hypothesis was later proven to be false by Martens and Medabalimi (2014). More specifically, they have shown that the uniform distribution on the spanning trees of a complete graph on $n$ vertices, which is known to be tractable by other methods, cannot be realized by SPNs, unless their size is exponential in $n$. The reason behind this limitation is not due to the simple operations on which they are built on, as any efficiently computable function could be approximated arbitrarily well by a polynomially-sized arithmetic circuits (Hoover, 1990), but rather its the strict structural constraints of SPNs that are required for tractability.

In this paper, we introduce an extension to SPNs, which we call *Sum-Product-Quotient Networks* (SPQNs for short), that addresses the limited expressivity of SPNs. The underlying concept behind our extension is to incorporate conditional probabilities into the model through direct computation, i.e. by repeatedly applying the formula $P(A|B) = \frac{P(A,B)}{P(B)}$. Specifically, we show that by adding a quotient node, i.e. a node with two inputs that computes their division, we can relax the structural constraints of SPNs and still have a model capable of tractable inference, where each internal node represents a conditional probability over its input variables. Moreover, we prove that while SPQNs can represent any distribution SPNs can, by virtue of being their extension, there exists distributions that are efficient for SPQNs but require SPNs to be of exponential size, proving SPQNs are exponentially more expressively efficient.

The rest of the article is organized as follows. In sec. 2 we briefly describe the SPNs model and its basic concepts. This is followed by sec. 3 in which we present our SPQNs extension, and prove that the resulting model is indeed tractable. In sec. 4 we analyze the expressive efficiency of SPQNs with respect to SPNs. Finally, we discuss the implications of our model on prior work and our plans for future research in sec. 5.

## 2 Preliminaries

In this section we give a brief description of Sum-Product Networks (SPNs). For simplicity, we limit our description to probability models over binary variables, where the extension to higher-dimensional or continuous variables is quite straightforward.

An SPN over binary random variables $X_1, \ldots, X_N$ is a rooted computational directed acyclic graph, which computes the unnormalized probability function of the evidence $x_1, \ldots, x_N \in \{0, 1, *\}$, denoted by $\Psi(x_1, \ldots, x_N)$, where $*$ denotes missing variables under which the SPN computes just the unnormalized marginal of the visible variables. The leaves of an SPN are univariate indicators of the binary variables, i.e. $\mathbb{I}[x_i = 0]$ and $\mathbb{I}[x_i = 1]$, with the special property that for $x_i = *$ all respective indicators of $x_i$ equal 1. The internal nodes of the SPN compute either a positive weighted sum or a product, i.e. an SPN is an Arithmetic Circuit over the indicator variables defined above. We denote by $S$ the set of sum nodes, by $P$ the set of product nodes, by $I$ the set of indicator nodes, and by $V = S \cup P \cup I$ the set of all nodes in the SPN. For all $v \in V$, we denote by $\text{ch}(v)$ the set of children nodes pointing to $v$, and define the scope of $v$, denoted by $\text{sc}(v)$, as the index set of all variables, such that there exists a path starting at an indicator of a variable, which ends at the node $v$. Formally, we define $\text{sc}(v) \equiv \{i\}$ for leaf nodes of the $i$-th variables, and otherwise $\text{sc}(v) \equiv \cup_{c \in \text{ch}(v)} \text{sc}(c)$. We denote the function induced by the sub-graph rooted at $v$ over the variables in $\text{sc}(v)$ by $\Psi_v(\cdot)$. Last, we define the following structural properties for SPNs:

**Definition 1.** An SPN is complete if for every sum node $v \in S$ and for every $c_1, c_2 \in \text{ch}(v)$ it holds that $\text{sc}(c_1) = \text{sc}(c_2)$.

**Definition 2.** An SPN is decomposable if for every product node $v \in P$ and for every $c_1, c_2 \in \text{ch}(v)$, such that $c_1 \neq c_2$, it holds that $\text{sc}(c_1) \cap \text{sc}(c_2) = \emptyset$.

Generally, for an SPN that is not decomposable and complete, $\Psi(\cdot)$ only represents an unnormalized distribution over $X_1, \ldots, X_N$, due to the positive constraints on its weights, while computing its normalization term is not typically tractable. A generative model is said to possess *tractable inference* if computing its normalized probability function is tractable. Though general SPNs do not posses tractable inference, limiting them to be decomposable and complete (D&C) is a sufficient condition for tractability, underwhich computing the normalization term, i.e. computing $\sum_{x_1, \ldots, x_N \in \{0,1\}} \Psi(x_1, \ldots, x_N)$ is equivalent to evaluating $\Psi(*, \ldots, *)$, and thus the normalized probability is given by $P(X_1{=}x_1, \ldots, X_N{=}x_N){=}\frac{\Psi(x_1, \ldots, x_N)}{\Psi(*, \ldots, *)}$. Also, not only is $\Psi(\cdot)$ a valid probability function, but for any $v \in V$, $\Psi_v(\cdot)$ defines a valid distribution over $\text{sc}(v)$. As shown by Peharz et al. (2015), simply normalizing the weights of each sum node to sum to one ensures that $\Psi(x_1, \ldots, x_N)$ is already a normalized probability function, with no need to compute a normalization factor, and furthermore, this restriction does not affect the expressiveness of the model, namely any SPN with unnormalized sum nodes could be converted to an SPN of same size but with normalized sum nodes. Hence, for the remainder of the article we will simply assume sum nodes have normalized weights.

It is important to understand why D&C leads to tractability. The decomposability condition ensures that the children of a product node do not have shared variables, and because the product of distributions over different sets of variables is also a normalized distribution, then a product node of a decomposable SPN represents a normalized distribution as long as its children represent normalized distributions. Similarly, the completeness condition ensures that the children of sum nodes have the exact same scope, and because a weighted average of distributions over the same set of variables, with normalized sum weights, is also a normalized distribution over these variables, then a sum node represents a normalized distribution if its children do as well. Employing an induction argument, both conditions combined together guarantee that every node in an SPN will represent a valid distribution.

An additional positive outcome of the D&C condition is that not only is it tractable to compute $P(X_1, \ldots, X_N)$, it is also tractable to compute any of its marginals, e.g. $P(X_1, \ldots, X_K)$ for $K < N$, by simply replacing the values of a marginalized variable with the special value $*$, e.g. $P(X_1{=}x_1, \ldots, X_K{=}x_K) = \Psi(x_1, \ldots, x_K, *, \ldots, *)$. We call this last property *tractable marginalization*, which is distinct from the weaker property of tractable inference.

Lastly, learning an SPN model of a given structure is typically carried out simply according to the Maximum Likelihood Principle, for which several methods have been proposed, ranging from specialized Expectation Maximization algorithms to gradient based methods, e.g. simply performing Stochastic Gradient Ascent.

## 3 Sum-Product-Quotient Networks

As discussed in sec. 1, not all tractable distributions can be represented by an SPN of a reasonable size, a limitation which stems from the D&C connectivity constraints imposed on the computational graphs of SPNs to achieve tractable inference. In this section we describe an extension of SPNs, under which we can relax these constraints and thus dramatically increase its capacity to efficiently represent tractable distributions. At the heart of our model is the introduction of a quotient node, i.e. a node with two inputs, a numerator and a denominator, that outputs their division. Quotient nodes can have a natural interpretation as a conditional probability, i.e. $P(A|B) = \frac{P(A\cap B)}{P(B)}$. Hence, we call our model Sum-Product-Quotient Networks, or SPQNs for short.

As with SPNs, not any computational graph made of sum, product and quotient nodes results in a model possessing tractable inference. To ensure the tractability of SPQNs, we introduce a set of restrictions generalizing the D&C conditions defined in sec. 2. Formally, and in accordance with the notations of sec. 2, we denote by $Q$ the set of quotient nodes, where $V \equiv S \cup P \cup Q \cup I$ is the set of all nodes, and for all $v \in Q$ we denote its numerator and denominator nodes by $\mathrm{nu}(v)$ and $\mathrm{de}(v)$, respectively. As we will shortly show, each node $v \in V$ of an SPQN essentially represents a conditional distribution over the variables in its scope, which give rise to a natural partition of the scope $\mathrm{sc}(v)$ into two disjoint sets: (i) *conditioning scope*, denoted by $\mathrm{cond}(v)$, and (ii) *effective scope*, denoted by $\mathrm{eff}(v)$ – under this partition, for tractable SPQNs, each node computes the conditional probability $P_v(\mathrm{eff}(v)|\mathrm{cond}(v))$. Formally, the conditioning scope is defined as the complement of the effective scope, i.e. $\mathrm{cond}(v) \equiv \mathrm{sc}(v) \setminus \mathrm{eff}(v)$, while the effective scope is defined the same as the general scope for all nodes except for quotient nodes, namely, $\mathrm{eff}(v) \equiv \{i\}$ for leaf nodes and $\mathrm{eff}(v) \equiv \cup_{c\in\mathrm{ch}(v)}\mathrm{eff}(c)$ for sum and product nodes. For quotient nodes we define $\mathrm{eff}(v) \equiv \mathrm{eff}(\mathrm{nu}(v)) \setminus \mathrm{eff}(\mathrm{de}(v))$ following our intuition of quotient nodes as conditional probabilities, e.g. for $P(X_1|X_2,X_3) = \frac{P(X_1,X_2|X_3)}{P(X_2|X_3)}$ it holds that $\mathrm{eff}(v) = \{1\}$ and $\mathrm{cond}(v) = \{2,3\}$, because we started with the effective variables of the numerator $\mathrm{eff}(\mathrm{nu}(v)) = \{1,2\}$, from which we subtracted the effective variables of the denominator $\mathrm{eff}(\mathrm{de}(v)) = \{2\}$.

With the above definitions in place, we are now ready to present our generalization of the D&C conditions for SPQNs. As discussed in sec. 2, the intuition behind the D&C conditions is that they allow for a rather basic way to combine the distributions defined by the children of a given node, each over their respective scope, to form a valid distribution over the scope of their parent node. In broad terms, we simply carry over the same idea to SPQNs, but apply it on conditional distributions instead. For sum and product nodes, this translates in essence to applying the D&C conditions with respect to the effective scope of a node instead of its general scope, which we formalize as:

**Definition 3.** An SPQN is *conditionally complete* if it is complete with respect to the effective scope, i.e. for every sum node $v \in S$ and for every $c_1, c_2 \in \mathrm{ch}(v)$, it holds that $\mathrm{eff}(c_1) = \mathrm{eff}(c_2)$.

**Definition 4.** An SPQN is *conditionally decomposable* if for every product node $v \in P$:

1. It is decomposable with respect to the effective scope, i.e. for every $c_1, c_2 \in \mathrm{ch}(v)$, such that $c_1 \neq c_2$, it holds that $\mathrm{eff}(c_1) \cap \mathrm{eff}(c_2) = \emptyset$.
2. Its induced dependency graph over its children does not contain a cycle, where the directed graph is defined by the vertices $\mathrm{ch}(v)$ and edges $\{c' \to c''|c', c'' \in \mathrm{ch}(v), \mathrm{eff}(c') \cap \mathrm{cond}(c'') \neq \emptyset\}$.

Under the conditional completeness condition, for every sum node $v \in S$, and for any fixed values to the variables in its conditional scope $\mathrm{cond}(v)$, we can treat the conditional distributions of its children simply as distributions over the variables in the effective scope. Because $v$ is complete with respect to the effective scope, then following the same arguments as in sec. 2, $v$ represents a distribution as long as its children do as well. The above logic can also be applied to product nodes under a more restrictive form of conditional decomposability, where for every child $c \in \mathrm{ch}(v)$ it holds that $\mathrm{cond}(c) \subset \mathrm{cond}(v)$, under which the variables in the conditional scope of each child node are fixed. However, under the more general setting of conditional decomposability, there could be shared variables between the conditional scope of one child $c_1 \in \mathrm{ch}(v)$ and the effective scope of another child $c_2 \in \mathrm{ch}(v)$ – in which case we say that $c_1$ depends on $c_2$, as the probability of the effective scope of $c_1$ is conditioned on the variables in the effective scope of $c_2$. By representing all the dependencies between the children of $v$ as a directed graph, then if each child represents a valid conditional distribution over its scope and the graph is *acyclic*, then it effectively defines a Bayesian Network factorization to the conditional probability over the scope of $v$, hence $v$ too is a valid conditional distribution.

At this point it is important to note how conditional D&C are actually relaxed versions of their "unconditional" counterparts. First, notice that when the conditional scope is empty, i.e. when the sub-graph rooted at $v$ contains only sum and product nodes, or in other words this sub-graph is an SPN, then conditional D&C are equivalent to D&C. Second, and more importantly, notice that when the conditional scope is nonempty, conditional decomposability allows taking the product of nodes with overlapping scopes, which is forbidden under the stricter decomposability constraint. This entails that conditional D&C SPQNs allow for a richer set of structures than D&C SPNs.

At last, to ensure the tractability of SPQNs we must also introduce a condition on its quotient nodes, to which there is no equivalent in classical SPNs. The following condition captures our motivation of a quotient node as a way to compute conditional distributions by direct representation of their definition, i.e. that the denominator is a strictly positive marginal distribution of the numerator:

**Definition 5.** An SPQN is *conditionally sound* if for every quotient node $v \in Q$, it holds that $\Psi_{\text{de}(v)}(\cdot)$ is strictly positive, as well as a *marginal* of $\Psi_{\text{nu}(v)}(\cdot)$, i.e. that $\text{cond}(\text{de}(v)) \subset \text{cond}(\text{nu}(v))$, $\text{eff}(\text{de}(v)) \subset \text{eff}(\text{nu}(v))$, and for all $\mathbf{a} \in \{0,1,*\}^N$ it holds that:

$$\Psi_{\text{de}(v)}(\mathbf{a}) = \sum_{\substack{\mathbf{z}\in\{0,1,*\}^N \\ \forall i, i\notin \text{eff}(v) \to z_i = a_i \\ \forall i, i \in \text{eff}(v) \to z_i \in \{0,1\}}} \Psi_{\text{nu}(v)}(\mathbf{z})$$

An SPQN is *strongly* conditionally sound if in addition to the above, for $\mathbf{z}\in\{0,1,*\}^N$ such that $z_i{=}*$ if $i{\in}\text{eff}(v)$ and otherwise $z_i{=}a_i$, it holds that $\Psi_{\text{de}(v)}(\mathbf{a}){=}\Psi_{\text{nu}(v)}(\mathbf{z})$.

The definition of strong conditional soundness above is not required for tractability – only the weaker conditional soundness – but does ensure efficient sampling as discussed in sec. 3.2. We conclude by formally proving that an SPQN that meets the above conditions, which will henceforth be referred to as a *tractable SPQN*, results in a tractable generative model, as described by the following theorem (see app. A.1 for proof):

**Theorem 1.** *For any conditionally decomposable, conditionally complete, and conditionally sound SPQN over the random binary variables $X_1, \ldots, X_N$, for all $v \in V$, and any values of the variables found in* $\text{cond}(v)$*, it holds that $\Psi_v(\cdot)$ is a normalized probability function over* $\text{eff}(v)$ *conditioned on* $\text{cond}(v)$.

Given an SPQN with a fixed structure that meets the tractability conditions of theorem 1, then its output is a differential probability function of the data, and so we can learn its parameters simply by maximizing the likelihood of the data through gradient ascent methods, as commonly employed by both SPNs and other deep learning methods. Adjusting other methods typically used to learn SPNs, e.g. EM-type algorithms for parameter learning and the various suggested structure learning algorithms, is deferred to future works.

Though theorem 1 provides sufficient conditions for SPQNs to be tractable, it is not prescriptive as to how exactly these models must be structured. Specifically, while the conditionally decomposable and conditionally complete conditions are quite simple to follow, it is generally not clear how to adhere to the conditionally sound condition. We address this in the next section.

## 3.1 Conditional Mixing Operator

As discussed in the previous section, tractable SPQNs must comply with the conditionally sound condition, and verifying that a given model adheres to it is nontrivial. In this section, we suggest instead to follow a stricter restriction that leads to a concrete construction of a tractable SPQN. Specifically, we define a building block operator composed of sum, product, and quotient nodes that guarantees the resulting model to be tractable, which we call the *Conditional Mixing Operator*:

**Definition 6.** The *Conditional Mixing Operator* (CMO) over non-negative matrices $A \in \mathbb{R}_+^{\gamma,\alpha}$ and $B \in \mathbb{R}_+^{\gamma,\beta}$, where $\alpha, \beta, \gamma \in \mathbb{N}$, $\beta > 0$, and parametrized by strictly positive weights $\mathbf{w} \in \mathbb{R}_+^{\gamma}$ such that $\sum_{i=1}^{\gamma} w_i = 1$, is defined as follows:

$$\text{CMO}(A, B; \mathbf{w}) = \frac{\sum_{i=1}^{\gamma} w_i \left(\prod_{j=1}^{\alpha} A_{ij}\right) \cdot \left(\prod_{j=1}^{\beta} B_{ij}\right)}{\sum_{i=1}^{\gamma} w_i \prod_{j=1}^{\alpha} A_{ij}} \tag{1}$$

In the context of SPQNs, a CMO node with children $a_{11}, \ldots, a_{\gamma\alpha}, b_{11}, \ldots, b_{\gamma\beta} \in V$ outputs $\text{CMO}(A, B; \mathbf{w})$, where $A_{ij} = \Psi_{a_{ij}}(\cdot), B_{ij} = \Psi_{b_{ij}}(\cdot)$.

The motivation behind this construction is its connection to the conditional probability of a mixture model. Notice that the numerator of eq. 1 essentially represents a mixture model with decomposable mixing components divided into two sets according to $A$ and $B$, while the denominator represents the marginalization over the variables relating to $B$.

The tractability of SPQNs composed of CMOs is ensured by the definition a *valid CMO node* as follows:

**Definition 7.** A CMO node with children $a_{11}, \ldots, a_{\gamma\alpha}, b_{11}, \ldots, b_{\gamma\beta} \in V$ is said to be valid if the following conditions are met:

1. The children of a CMO node are either valid CMO nodes themselves, or it holds that $\alpha = 0, \beta = 1, \gamma = 2$, and its children are exactly $\mathbb{I}[x_i = 0]$ and $\mathbb{I}[x_i = 1]$ for some $i \in [N]$.
2. The internal sum nodes of the CMO are conditionally complete.
3. The internal product nodes of the CMO, i.e. the ones computing $\prod_{j=1}^{\alpha} A_{ij}$, $\prod_{j=1}^{\beta} B_{ij}$, and $\left(\prod_{j=1}^{\alpha} A_{ij}\right) \cdot \left(\prod_{j=1}^{\beta} B_{ij}\right)$, are conditionally decomposable, and in the dependency graph of the top product node there are no arrows pointing from $B$ to $A$.
4. $\forall i_1, i_2 \in [\gamma],\ \text{eff}\left(\prod_{j=1}^{\beta} B_{i_1 j}\right) = \text{eff}\left(\prod_{j=1}^{\beta} B_{i_2 j}\right)$.

We proceed to formalize our claim as follows:

**Proposition 1.** *Any SPQN that is composed of valid CMO nodes is tractable. Moreover, it is strongly conditionally sound.*

*Proof Sketch.* Since the internal sum and product nodes of a valid CMO are already conditionally D&C,

it is only left to show that it is also conditionally sound. This is achieved by an induction argument on the depth of an SPQN composed of valid CMOs, where we assume all nodes up to a given depth $d$ are strictly positive, conditionally sound, and hence also represent valid distributions according to theorem 1. By the assumption, the internal sum and product nodes of a depth $d+1$ valid CMO node also represent valid distributions, as they are already conditionally D&C. Hence we can directly compute its marginalization over the variables in the effective scope of the B-type children, to conclude our proof of conditional soundness. Strong conditional soundness follows from conditional soundness and the definition of the CMO, since placing $*$ in all variables of the effective scope of the B-type children is equivalent to substituting their values with 1's. See our complete proof in app. A.2. □

Unlike conditional soundness, it is practical to validate that all CMO nodes in a given SPQN are valid. Simply start at the root and recursively validate that each of the children of a given node are valid, with the base case of CMO nodes connected to one of the indicator nodes, as govern by the first condition in def. 7. We then proceed to verifying that the internal product and sum nodes follow the conditional D&C constraints, by simply testing their effective and conditional scopes according to def. 3 and def. 4.

Though valid CMOs pave the way to tractable SPQNs, they raise the question of what we have lost in the process. Indeed, conditional soundness allows for a richer set of valid structures than valid CMOs, e.g. they allow for the distribution at the denominator and numerator of a quotient node to be defined by completely different sub-graphs, unlike with CMOs that share children. While we have yet to determine if there is a significant expressivity gap between these two cases, an important property of an SPQN composed of valid CMOs is that any D&C SPN can be effectively represented by such a model[1], hence this restriction is at least as expressive as any D&C SPN. In sec. 4 we show that they are in fact significantly more expressive than SPNs.

## 3.2 The Generative Process of SPQNs

In prior sections we have presented our SPQN model, and showed that it can be tractable under simple conditions, and more importantly that any of its internal nodes represent a conditional distribution over its scope. In this section we leverage these relations to describe the generative process of SPQNs, showing sampling from an SPQN is just as efficient as inference, under the strongly conditional soundness constraint. The ability to efficiently draw samples from a probabilistic model is a highly desirable trait with many applications, e.g. completing missing values, and introspection of the learned models.

Sampling from a tractable SPQN model follow the same general steps as sampling from a D&C SPN. We begin at the root node of the graph, and then stochastically traverse the nodes according to parameters of the model, until we reach the indicator nodes, each representing the sampled value for its respective random variable. In SPNs, traversal follow two simple rules: (i) if we encounter a product node, then because it is decomposable, each child is a distribution over separate sets of variables, hence we can recursively sample from each child separately; (ii) if we encounter a sum node, then we sample one of its children according to the categorical distribution defined by their respective weights. Given that SPQNs are extensions of SPNs, their generative process can be seen as simply a generalization of the traversal rules of the SPNs. However, their distinctive property of having nodes which represent conditional distributions, calls for some adjustments. Namely, it is not only required to traverse the graph, but also to keep track of the values that have already been sampled so far in the process, and then pass it along to nodes which depend on it.

**Algorithm 1** Sampling procedure for SPQNs. Accepts as input a node $v \in V$, and a partial sample $\mathbf{s} \in \{0,1,*\}^N$, where $*$ denotes missing values.

```
1:  function SampleSPQN($v, \mathbf{s}$)
2:      if $v \in Q$ then
3:          $\mathbf{s} \leftarrow$ SampleSPQN(nu($v$), $\mathbf{s}$)
4:      else if $v \in P$ then
5:          $children \leftarrow$ TopologicalSort(ch($v$))
6:          for all $c \in children$ do
7:              if $\forall i \in \text{eff}(c), s_i \neq *$ then skip iteration
8:              $\mathbf{s} \leftarrow$ SampleSPQN($c, \mathbf{s}$)
9:      else if $v \in S$ then
10:         $\mathbf{w} \leftarrow$ GetWeights($v$)
11:         for all $c \in \text{ch}(v)$ do
12:             $w_c \leftarrow w_c \cdot \Psi_c(\mathbf{s})$
13:         $\mathbf{w} \leftarrow \mathbf{w}/\sum_c w_c$
14:         $c \sim Cat(\text{ch}(v), \mathbf{w})$
15:         $\mathbf{s} \leftarrow$ SampleSPQN($c, \mathbf{s}$)
16:     else if $\exists i \in \{1,\ldots,N\}, v \equiv \mathbb{I}[x_i = a]$ then
17:         $s_i \leftarrow a$
18:     return $\mathbf{s}$
```

The above reasoning brings us to algo. 1, which receives as input a starting root node $v \in V$, and a partial sample $\mathbf{s} \in \{0,1,*\}^N$, where $s_i = *$ denotes values which have yet to be sampled. Typi-

[1] An edge case of SPNs which demands a unique treatment is when there exists a sum node which is connected to just one of $\mathbb{I}[x_i = 0]$ or $\mathbb{I}[X_i = 1]$, but not both, while a valid CMO must have positive weights for both indicator leaves. In this scenario we can instead arbitrarily approximate the SPN, by approaching the zero weight $\epsilon \to 0$.

cally, the first call to algo. 1 will be with the root $v \in V$ and $\mathbf{s} = (*, \ldots, *)$, i.e. sampling a complete instance $X_1, \ldots, X_N \sim P(X_1, \ldots, X_N)$, but often times it is useful to also be able to sample from the conditional distribution[2], e.g. $X_1, \ldots, X_K \sim P(X_1, \ldots, X_K | X_{K+1}{=}s_{k+1}, \ldots, X_N{=}s_N)$ by calling with $\mathbf{s} = (*, \ldots, *, s_{k+1}, \ldots, s_N)$. The inner-workings of algo. 1 follow the traversal workflow of SPNs as described above, with the following adjustments: (i) For quotient nodes, we directly traverse to its numerator child, as the denominator only serve as a normalization factor. (ii) For product nodes, though the effective scopes of the children are disjoint sets and could be processed separately as with SPNs, the dependencies induced by the conditional scopes of each child require sampling according to the topological order of the dependencies graph. Additionally, there is the possibility that the effective scope of some child nodes have already been sampled, in which case we simply skip it. (iii) For sum nodes, the probability of sampling each child is no longer given just by its weights, but also by the marginal probability of the already sampled variables given by $\mathbf{s}$, namely if $Q \equiv \{i \in [N] | s_i {\neq} *\}$ denotes the set of sampled variables then we can factor the conditional distribution of the sum node $v \in S$, i.e. $P_v(\text{eff}(v) \backslash Q | Q)$, as the following expression:

$$\sum_{c \in \text{ch}(v)} P_c(\text{eff}(c) \backslash Q | Q) \frac{w_c \cdot P_c(\text{eff}(c) {\cap} Q | \text{cond}(c))}{\sum_{c' \in \text{ch}(v)} w_{c'} \cdot P_{c'}(\text{eff}(c') {\cap} Q | \text{cond}(c'))}$$

where $P_c(\text{eff}(c) {\cap} Q | \text{cond}(c))$ can be computed by $\Psi_c(\mathbf{s})$ according to strong conditional soundness, and thus the probability of sampling the child $c$ is $\propto w_c \cdot \Psi_c(\mathbf{s})$.

Finally, regarding the complexity of the sampling algorithm, traversing the computational graph is linear in the number of nodes, and while computing $\Psi_v(\cdot)$ when sampling from sum nodes could result in an $O(|V|^2)$ runtime, in practice we could reuse prior computations to reduce it to just $O(|V|)$. In this analysis, we do not take into account the topological sort applied to the children of the product nodes, as this is a one time operation that is not required for every sampling. In conclusion, sampling from a tractable SPQN that is also strongly conditionally sound, e.g. by composition of valid CMOs, is just as efficient as with SPNs.

## 4 Analysis of Expressive Efficiency

In sec. 3, we have shown that tractable SPQNs extend D&C SPNs, and can thus efficiently replicate any tractable distribution that D&C SPNs can realize. In this section, we will show a simple tractable distribution which SPQNs can realize, but D&C SPNs cannot, unless their size is exponential in the length of their input, where the size of an SPQN (or SPN) is defined as the number of its internal nodes. More specifically, we show that tractable SPQNs can represent a strictly positive distribution of sampling an undirected triangle-free graph on $M$ vertices, where each edge is represented by a random binary number, while D&C SPNs of polynomial size cannot represent, or even approximate, such distributions.

First, let us formally define a strictly positive distribution over triangle-free undirected graphs on $M$ vertices. We define the binary random variables $\mathbf{E} \equiv \{E_{ij} | 1 \leq i < j \leq M\}$, such that if $E_{ij} = 1$, then the edge $\{i, j\}$ is part of the graph, and not otherwise, and denote by $N \equiv |\mathbf{E}| = \binom{M}{2}$ the number of variables. For a given graph, we say it contains a triangle if and only if there are three vertices in the graph such that between any two of them there is an edge, i.e. there exists $i_1 < i_2 < i_3$ such that $(E_{i_1 i_2}{=}1) \wedge (E_{i_2 i_3}{=}1) \wedge (E_{i_1 i_3}{=}1)$. Finally, we say that a probability function $d(\mathbf{E})$ on the edges $\mathbf{E}$ is a *strictly positive distribution on triangle-free graphs* if it holds that $d(\mathbf{E}) > 0$ if and only if $\forall i_1 < i_2 < i_3, (E_{i_1 i_2}{=}0) \vee (E_{i_2 i_3}{=}0) \vee (E_{i_1 i_3}{=}0)$.

The above definition falsely appears to lead to an efficient realization through SPNs of a strictly positive distribution on triangle-free graphs: simply define a node for each potential triangle, such that it is positive only if it is legal, i.e. at least one of its edges is not part of the graph, and then take the product of all such nodes to guarantee all triangles are legal. More specifically, we can define a sum node for each triplet $(E_{i_1 i_2}, E_{i_2 i_3}, E_{i_1 i_3})$, for which there are $\binom{M}{3}$ combinations, such that each sum node is equal to $(\mathbb{I}[E_{i_1 i_2}{=}0] + \mathbb{I}[E_{i_2 i_3}{=}0] + \mathbb{I}[E_{i_1 i_3}{=}0])$, and then take the product of all of these sum nodes and modify their weights such that they output a normalized probability function. However, this SPN is not D&C, because each sum node does not meet the completeness condition as its children have different scopes, e.g. $\text{sc}(\mathbb{I}[E_{i_1 i_2}{=}0]) \neq \text{sc}(\mathbb{I}[E_{i_2 i_3}{=}0])$, and because the product node over all sum nodes does not meet the decomposability condition, as each edge $E_{i_1 i_2}$ is present in multiple triplets, i.e. multiple child nodes, resulting in non-disjoint scopes. Because it is not D&C, computing its normalization factor in practice is intractable. More generally, we can show that any D&C SPN approximating a strictly positive distribution on triangle-free graphs must be exponentially large:

**Theorem 2.** *Let $d(\mathbf{E})$ be a strictly positive distribution on triangle-free graphs of $M$ vertices. Suppose that $d(\mathbf{E})$ can be approximated arbitrarily well by D&C SPNs of size $\leq s$. Then $s \geq 2^{\Omega(M)}$.*

[2]Exactly sampling from a conditional distribution is possible only if it respects the dependencies induced by the model on the input variables.

*Proof Sketch.* We have modified the proof of a similar theorem by Martens and Medabalimi (2014), which showed that a D&C SPN that can approximate arbitrarily well the probability function of the uniform distribution on the spanning trees of the complete graph, must be of size $\geq 2^{\Omega(M)}$. See app. A.3 for the complete modification of that proof to our case. □

In contrast to D&C SPNs, tractable SPQNs can efficiently realize at least some strictly positive distributions on triangle-free graphs, with size at most polynomial in $M$. In the case of SPQNs built on CMOs, exact realization is replaced by arbitrarily good approximation, without any size increase. This is formalized by the following theorem:

**Theorem 3.** *There exists a tractable SPQN exactly realizing a probability function $d(\mathbf{E})$, such that $d(\mathbf{E})$ is a strictly positive distribution on triangle-free graphs of $M$ vertices, where the size of the SPQN is $O(M^4)$. In the case of SPQNs composed strictly of CMOs, instead of exact realization, they can approximate said distribution arbitrarily well with size $O(M^4)$.*

*Proof Sketch.* Taking inspiration from the failed attempt to realize such a distribution via D&C SPNs, let us now construct a tractable SPQN which does realize such a distribution efficiently. As before, we begin by examining all potential triangles, but instead of directly modelling the constraints individually, we group them by their largest edge (according to lexical ordering). For each edge and its respective group of triangles, we can define the conditional probability of that edge conditioned on all other edges participating in these triangles, such that the conditional probability is non-zero only if triangles which include this edge are not all part of that graph. For edges that are not part of any triangle for which they are the largest edge, we simply define a sum node which represent an equal probability for including the edge or not. Finally, we can simply take the product of all conditional distributions of each edge, giving rise to a normalized probability function over all edges $\mathbf{E}$, which is non-zero if and only if the edges in $\mathbf{E}$ represent a triangle free graph. See app. A.4 for our complete proof. □

To conclude, we have shown that tractable SPQNs, as well as ones composed of valid CMOs, are exponentially efficient with respect to D&C SPNs.

## 5 Discussion and Related Works

In this work we address the limited expressive efficiency of SPNs, which Martens and Medabalimi (2014) have proven to be incapable of approximating even simple tractable distributions, unless their size is exponential in the number of variables. To mitigate this limitation of SPNs, we have presented a novel extension to SPNs which we call Sum-Product-Quotient Networks, or SPQNs for short. SPQNs introduce a new node type that computes the quotient of its two inputs, which in part enabled us to relax the strict structural conditions that are commonly used to ensure the tractability of SPNs. By requiring less strict conditions for tractability, we have proven that SPQNs are a strict superset of SPNs, and moreover that SPQNs are exponentially more expressive efficient than SPQNs.

There is a vast literature on analyzing the expressivity of arithmetic circuits (ACs) (Shpilka and Yehudayoff, 2010; Cohen et al., 2016; Cohen and Shashua, 2017; Cohen et al., 2017; Levine et al., 2017), and more particularly of SPNs (Delalleau and Bengio, 2011; Martens and Medabalimi, 2014). Notable amongst those is the work of Sharir and Shashua (2017), where they compared the expressive efficiency of Convolutional ACs (ConvACs) having no overlapping receptive fields, which are equivalent to a sub-class of SPNs following a tree-structure partitioning of scopes, against a ConvAC with overlaps, which have no equivalent D&C SPN. They have found that simply introducing overlaps, i.e. breaking the decomposability condition, had the effect of exponentially increasing the expressive efficiency of the model. A closer examination of their overlapping ConvAC reveals that it shares the same construct as the numerator of our CMOs nodes, but without the denominator, and thus their results could be trivially adapted to SPQNs following a similar architecture. This entails that not only are there some distributions which SPQNs can represent efficiently that SPNs cannot, as we have showed in sec. 4, but that almost all distributions realized by SPQNs cannot be realized by tree-like SPNs[3], known also as Latent Tree Models (Mourad et al., 2013), unless they are of exponential size. Nevertheless, it is important to stress the importance of our own results, which separate between SPQNs and D&C SPNs of any conceivable structure, and not just a small sub-class of SPNs.

Recently, Telgarsky (2017) has examined the relations between neural networks and rational functions, i.e. quotient of two polynomials, as well as a model he called *rational networks*, which is a neural network with activation functions limited to only rational functions. He found that a new neural network with ReLU activations could be approximated arbitrarily well by a similarly size rational network, and that the reverse is true as well. Though this might seem to suggest that SPQNs could be on par with neural networks, Hoover

[3]Not to be confused with Sum-Product Trees (Peharz et al., 2015) that are a far more restricted sub-class of SPNs, in which every sum and product nodes have just a single parent, as opposed to limiting just the sum nodes to have a single parent as in non-overlapping ConvACs.

(1990) proved that any computable function can be realized by ACs – hence the power of SPQNs is not due to quotient nodes, but rather their richer structure.

In the broader literature on ACs (Shpilka and Yehudayoff, 2010), the proposal of introducing a quotient node has been previously considered and deemed unnecessary. Their argument is based on the proof that in circuits which compute polynomial functions all quotient nodes could be replaced by just a single negation node, or in other words, that a quotient node does not add any power to ACs. Despite this negative outcome, it does not apply to our case on two accounts: (i) It assumes the output of the circuit is identically a polynomial function instead of a rational function, and since the proof itself relies on the structural properties of the polynomial, namely its degree and homogenous decomposition, it cannot be adapted to our case. (ii) It does not apply to monotone ACs, where the weights are restricted to be non-negative, as is the case of SPNs, where negation is not allowed. In this context, it was proven that even a single negation gate leads to exponential separation from monotone circuits, and while quotient nodes could be replaced by negation, the reverse is not generally true, hence this last result does not trivialize our own. Overall, given our results, it might suggest that the role of quotient nodes should be reexamined for ACs.

While we prove that our SPQN model is exponentially more expressive than D&C SPNs, this increase in expressive efficiency does not come without a cost. One of the great advantages of SPNs is that they not only possess tractable inference, but also tractable marginalization (see sec. 2). This uncommon ability amongst generative models has many uses, e.g. for missing data (Rubin, 1976; Sharir et al., 2016). However, once we relax decomposability to conditional decomposability, it means that SPQNs effectively induce a partial ordering on the input variables, which limit tractable marginalization only to the subsets of the variables that agree with the ordering. While there appear to be fewer tasks which benefit significantly from tractable marginalization compared to just tractable inference, in the cases in which it is required, SPNs even with their limited expressivity still have an advantage over SPQNs. This is a limitation that we aim to address in future works, as we detailed below. Additionally, Martens and Medabalimi (2014) have shown that under mild assumptions D&C is not only sufficient but also necessary for tractable marginalization, which entails that any possible relaxation to D&C would result in losing general tractable marginalization, hence it is not specific to the case of SPQNs.

Other recent works on tractable generative models have mainly focused on the family of autoregressive models that are based on neural networks, most notable amongst are NADE (Uria et al., 2016), PixelRNN (van den Oord et al., 2016b), and PixelCNN (van den Oord et al., 2016a). Despite the significant differences between the underlying operations of SPQNs and these models, there are also some similarities and shared concepts. Specifically, both our model and theirs are based on inducing a partial ordering on the input variables, and modelling the conditional probabilities between subsets of them, with the main difference as to how these probabilities are represented. While they employ neural networks as a black box to model them, we leverage interpretable SPNs to compose conditional distributions in a hierarchy. We conjecture that the embedded hierarchy of conditional distributions used in our model leads to an advantage in terms of its expressive capacity, while, in addition, the interpretable nature of the inner-workings of our model has many real-world applications.

Lastly, we conducted preliminary experiments demonstrating the practical advantages of SPQNs over SPNs in app. B. Nevertheless, it still remains to be verified that their superior expressive power translates to real-world applications – a task we aim to tackle in future works. Beyond that, SPQNs give rise to many straightforward extensions:

1. **Generative Classifier:** SPQNs could be naturally extended to represent a distribution conditioned on a given class, i.e. $P(X|Y)$, especially suitable for semi-supervised learning, and for classification under missing data.
2. **Tractable Marginalization:** despite that marginalization is not naturally supported by SPQNs, they do induce normalized distributions over any subset of its input variables, which are not generally consistent with each other. Joint training of SPQNs on random subsets of its variables, could be sufficient for ensuring the consistency of the induced marginal distributions.
3. **Convolutional SPQNs:** our model has a natural formulation as a ConvNet-like generative model following the theoretical architecture of ConvACs with overlaps (Sharir and Shashua, 2017). The unparalleled success of ConvNets with the theoretical advantages of SPQNs has potential for rivalling neural tractable generative models.

This paper has demonstrated the theoretical viability of Sum-Product-Quotient Networks, suggesting a promising outlook for the above research directions.

## Acknowledgments

This work is supported by Intel grant ICRI-CI #9-2012-6133, by ISF Center grant 1790/12 and by the European Research Council (TheoryDL project).

## References

Mohamed R Amer and Sinisa Todorovic. Sum-product networks for modeling activities with stochastic structure. *Computer Vision and Pattern Recognition*, 2012.

Mohamed R Amer and Sinisa Todorovic. Sum Product Networks for Activity Recognition. *IEEE transactions on pattern analysis and machine intelligence*, 38(4):800–813, 2016.

Nadav Cohen and Amnon Shashua. Inductive Bias of Deep Convolutional Networks through Pooling Geometry. In *International Conference on Learning Representations ICLR*, April 2017.

Nadav Cohen, Or Sharir, and Amnon Shashua. On the Expressive Power of Deep Learning: A Tensor Analysis. In *Conference on Learning Theory COLT*, May 2016.

Nadav Cohen, Ronen Tamari, and Amnon Shashua. Boosting Dilated Convolutional Networks with Mixed Tensor Decompositions. *arXiv.org*, 2017.

Olivier Delalleau and Yoshua Bengio. Shallow vs. Deep Sum-Product Networks. *Advances in Neural Information Processing Systems*, pages 666–674, 2011.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. In *International Conference on Learning Representations ICLR*, April 2017.

Robert Gens and Pedro M Domingos. Discriminative Learning of Sum-Product Networks. *Advances in Neural Information Processing Systems*, 2012.

Robert Gens and Pedro M Domingos. Learning the Structure of Sum-Product Networks. *ICML*, 2013.

H James Hoover. Feasible Real Functions and Arithmetic Circuits. *SIAM Journal on Computing*, 19(1):182–204, February 1990.

Diederik Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*, 2015.

Yoav Levine, David Yakira, Nadav Cohen, and Amnon Shashua. Deep Learning and Quantum Entanglement: Fundamental Connections with Implications to Network Design. *arXiv.org*, April 2017.

James Martens and Venkatesh Medabalimi. On the Expressive Efficiency of Sum Product Networks. *CoRR abs/1202.2745*, cs.LG, 2014.

Raphaël Mourad, Christine Sinoquet, Nevin Lianwen Zhang, Tengfei Liu, and Philippe Leray. A Survey on Latent Tree Models and Applications. *Journal of Artificial Intelligence Research*, 47(1):157–203, May 2013.

Robert Peharz, Sebastian Tschiatschek, Franz Pernkopf, and Pedro M Domingos. On Theoretical Properties of Sum-Product Networks. *International Conference on Artificial Intelligence and Statistics*, pages 744–752, 2015.

Hoifung Poon and Pedro Domingos. Sum-Product Networks: A New Deep Architecture. In *Uncertainty in Artificail Intelligence*, 2011.

Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, December 1976.

Or Sharir and Amnon Shashua. On the Expressive Power of Overlapping Operations of Deep Networks. *arXiv preprint arXiv:1703.02065*, 2017.

Or Sharir, Ronen Tamari, Nadav Cohen, and Amnon Shashua. Tensorial Mixture Models. *arXiv.org*, October 2016.

Amir Shpilka and Amir Yehudayoff. Arithmetic Circuits: A survey of recent results and open questions. *Foundations and Trends® in Theoretical Computer Science*, 5(3–4): 207–388, March 2010.

Matus Telgarsky. Neural networks and rational functions. In *International Conference on Machine Learning ICML*, August 2017.

Benigno Uria, Marc-Alexandre Cote, Karol Gregor, Iain Murray, and Hugo Larochelle. Neural Autoregressive Distribution Estimation. *Journal of Machine Learning Research ()*, 17(205):1–37, 2016.

Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Koray Kavukcuoglu, Oriol Vinyals, and Alex Graves. Conditional Image Generation with PixelCNN Decoders. *Advances in Neural Information Processing Systems*, 2016a.

Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel Recurrent Neural Networks. In *International Conference on Machine Learning*, 2016b.