On Valued Negation Normal Form Formulas*

Hélène Fargier
IRIT-CNRS, Toulouse
email: fargier@irit.fr

Pierre Marquis
CRIL-CNRS, Université d'Artois, Lens
email: marquis@cril.univ-artois.fr

Abstract

Subsets of the Negation Normal Form formulas (NNFs) of propositional logic have received much attention in AI and proved as valuable representation languages for Boolean functions. In this paper, we present a new framework, called VNNF, for the representation of a much more general class of functions than just Boolean ones. This framework supports a larger family of queries and transformations than in the NNF case, including optimization ones. As such, it encompasses a number of existing settings, e.g. NNFs, semiring CSPs, mixed CSPs, SLDDs, ADD, AADDs. We show how the properties imposed on NNFs to define more "tractable" fragments (decomposability, determinism, decision, read-once) can be extended to VNNFs, giving rise to subsets for which a number of queries and transformations can be achieved in polynomial time.

1 Introduction

For the past few years, several frameworks specializing the standard propositional one have been developed. Such frameworks are centered on fragments which are proper subsets of a full propositional language. Among them is the influential DNNF fragment [Darwiche, 2001], and its subsets d-DNNF, FBDD and OBDD, which have been successfully applied to a number of AI tasks, including diagnosis, reasoning under uncertainty, and planning. The success of such languages comes from the fact that they support a number of queries and transformations in polynomial time and are quite spatially succinct. The sources of their efficiency have been identified as formal properties on the corresponding structures, e.g. decomposability for DNNF, while d-DNNF asks also for determinism [Darwiche and Marquis, 2002].

On the other hand, in the recent years, many algebraic frameworks generalizing the standard CSP one have been pointed out; among them are VCSP [Schiex *et al.*, 1995], valuation algebra [Shenoy and Shafer, 1988; Kohlas and Shenoy, 2000] and semiring CSP [Bistarelli *et al.*, 1995]; PFU [Pralet *et al.*, 2006]. In such frameworks, satisfaction is a more gradual notion than in the standard CSP one; valuations range over some ordered scale (which can e.g. be interpreted as a utility/uncertainty scale) and can be aggregated using some operators. These frameworks consider a number of queries which extend the ones considered in the standard framework (mainly, the consistency query); especially, optimization is an important query in such settings.

At the intersection of those two research lines are a few approaches, where DAGs are used to represent functions ranging over an ordered scale. Let us mention SLDD [Wilson, 2005], AADD [Sanner and McAllester, 2005], ADD [Bahar *et al.*, 1993], and arithmetic circuits [Darwiche, 2002].

This work can be viewed as a further, yet more systematic attempt to bridge the gap between both research streams. We present a general framework, called VNNF (for "Valued NNFs"), suited for the representation of a much more general class of functions than just Boolean ones. This framework supports a larger family of queries and transformations than in the NNF case, including variable elimination, suited to optimization issues and encompasses a number of existing representation settings, e.g. NNF, semiring CSP, mixed CSP, SLDD, ADD, and AADD. We show how the properties imposed on NNFs to define "tractable" fragments (decomposability, determinism, decision, read-once) can be extended to give rise to subsets of VNNF, for which a number of queries and transformations can be achieved in polynomial time.

2 The VNNF Framework

The VNNF framework gathers the family of VNNF languages, and the queries and transformations they support. Each VNNF language allows the representation of some functions ranging over an ordered scale; such a language is fully characterized by a *representation context* $\langle \mathcal{E}, X, F \rangle$ consisting of a valuation structure $\mathcal{E}$, a set X of variables and a set F of primitive or "local" functions (the word is taken from [Pralet *et al.*, 2006] where local functions represent preferences or plausibility degrees over assignments). Let us first make precise the notion of valuation structure:

Definition 1 (valuation structure) *A* valuation structure *is a triple* $\mathcal{E} = \langle E, \succeq, OP \rangle$ *where:*

- $(E, \succeq)$ *is a set ordered by a relation* $\succeq$ *(which is thus*

*The authors would like to thank Sylvie Coste-Marquis for her help. The second author has been partly supported by the Région Nord/Pas-de-Calais, the IRCICA Consortium and by the European Community FEDER Program.

reflexive, antisymmetric and transitive) and such that E has a greatest element $\top$ and a least element $\bot$;

- *OP is a subset of $OP_{\mathcal{E}}$, the set of all binary operators $\otimes$ on E such that each $\otimes$ is associative, commutative, monotonic ($\forall a, b, c \in E$, if $a \succeq b$ then $a \otimes c \succeq b \otimes c$) and has a neutral element $e_{\otimes}$.*

When E is totally ordered by $\succeq$, elements of $OP_{\mathcal{E}}$ are generally called *uninorms*. When $\bot$ (resp. $\top$) is the neutral element of $\otimes$, $\otimes$ is usually called a *t-conorm* (resp. a *t-norm*) on E. We shall use the same terminology here, neglecting the fact that E is not necessarily totally ordered. For instance, t-conorms on $E = [0, 1]$ include the operations $\max(a, b)$, $a + b - a.b$, $\min(a + b, 1)$. T-norms include the operations $\min(a, b)$, $a.b$. Max and $\sum$ are also t-conorms on $E = \mathbb{R}^+ \cup \{+\infty\}$. $\sum$ is a uninorm on $E = \mathbb{R} \cup \{+\infty, -\infty\}$.

We assume that OP contains the following operators $\wedge$ and $\vee$, which can be viewed as generalizations of the well-known Boolean connectives:

$a \wedge \top = \top \wedge a = a$, $\quad a \wedge b = \bot$ if $a, b \neq \top$
$a \vee \bot = \bot \vee a = a$, $\quad a \vee b = \top$ if $a, b \neq \bot$.

When $\succeq$ is a total order, $\min$ and $\max$ are alternative generalizations of the Boolean connectives. Interestingly, $\wedge$ and $\vee$ are admissible in any valuation structure (hence our assumption is harmless), while $\min$ and $\max$ are admissible in any valuation structure with a totally ordered domain:

Proposition 1 [1] *Let $\mathcal{E}$ be any valuation structure. $OP_{\mathcal{E}}$ contains $\wedge$ (resp. $\vee$) which is a t-norm (resp. a t-conorm) on E. Furthermore, if $\succeq$ is a total order, $OP_{\mathcal{E}}$ contains* $\min$ *(resp.* $\max$*) which is a t-norm (resp. a t-conorm) on E.*

The last two elements of a representation context are much simpler notions:

- $X = \{x_1, \ldots, x_n\}$ is a finite set of variables ranging on finite domains. $dom(x)$ denotes the domain of variable $x \in X$. When $Y \subseteq X$, we note $dom(Y) = \Pi_{x \in Y} dom(x)$ and call Y-assignments the elements $\vec{y} \in dom(Y)$. If Y and Z are disjoint subsets of X, then $\vec{y}+\vec{z}$ is the $Y \cup Z$-assignment obtained by ordering the values given in the two tuples in an increasing way w.r.t. the indexes of the associated variables.
- F is a set of functions f ranging over E (the "local functions"). When f is a function from $Y \subseteq X$ to E, $scope(f) = Y$ is called the *scope* of f. Constant functions are identified with elements of E without loss of generality. We assume that F contains all functions f such that $card(scope(f)) \leq 1$, i.e. F contains all "literals" (cf. Section 5.3) and constants from E.

Let $f \in F$ such that $scope(f) = Y$. Let Z s.t. $Y \subseteq Z \subseteq X$ and $\vec{z}$ be any Z-assignment. We consider that $f(\vec{z})$ is equal to $f(\vec{y})$ where $\vec{y}$ is the Y-assignment which coincides with $\vec{z}$ for every variable from Y.

Now, let f, $f' \in F$; let $\otimes \in OP$ and $x \in X$; let $\vec{z}$ be any Z-assignment s.t. $Z \subseteq scope(f)$. We shall use the following notations:

- $f_{\vec{z}}$ denotes the *restriction* (or *conditioning*) of f by $\vec{z}$, i.e. the function given by $scope(f_{\vec{z}}) = scope(f) \setminus Z$ and for any $scope(f_{\vec{z}})$-assignment $\vec{t}$, $f_{\vec{z}}(\vec{t}) = f(\vec{z} + \vec{t})$. Clearly enough, the conditioning of f by $\vec{z}$ where Z is any subset of X (and not necessarily of $scope(f)$) can also be defined, and considered equal to $f_{\vec{y}}$ where $\vec{y}$ is the $Z \cap scope(f)$-assignment which coincides with $\vec{z}$ for every variable from $Z \cap scope(f)$.
- $f \otimes f'$ is the $\otimes$-*combination* of f and f', i.e. the function given by $scope(f \otimes f') = scope(f) \cup scope(f')$ and for any $scope(f \otimes f')$-assignment $\vec{t}$, $f \otimes f'(\vec{t}) = f(\vec{t}) \otimes f'(\vec{t})$.
- $f_{[x,\otimes]}$ is the $[x, \otimes]$-*projection* of f (or $\otimes$-elimination of variable x), i.e. the function given by $scope(f_{[x,\otimes]}) = scope(f) \setminus \{x\}$ and for any $scope(f_{[x,\otimes]})$-assignment $\vec{t}$, $f_{[x,\otimes]}(\vec{t}) = \bigotimes_{\vec{x} \in dom(x)} f_{\vec{x}}(\vec{t})$.

We shall typically consider tractable representation contexts:

Definition 2 (tractable representation context) *A representation context $\langle \mathcal{E}, X, F \rangle$ where $\mathcal{E} = \langle E, \succeq, OP \rangle$ is* tractable *iff each element of $F \cup OP$ is in linear time,*[2] *and this is also the case for the characteristic function of $\succeq$.*

If a function f is in linear time, then every conditioning of f and every $\otimes$-combination of f with another linear time function are in linear time as well, provided that $\otimes$ is in linear time. and any $\vec{z}$, $f_{\vec{z}} \in F$). We are now ready to define in a formal way the family of VNNF languages:

Definition 3 (VNNF) *Given a representation context $\langle \mathcal{E}, X, F \rangle$ where $\mathcal{E} = \langle E, \succeq, OP \rangle$, VNNF is the set of all finite, rooted directed acyclic graphs (DAGs) where each internal node is labeled by the name of an operator of OP and can have many arbitrarily children and where each leaf node is labeled by the name of an element of F and by a Z-assignment where $Z \subseteq X$.*

It is important to observe that such DAGs are not concerned by the representation of local functions (or operators): they are just given by their names, and can be represented as data structures or algorithms, but outside the DAG.

Each leaf node N of a VNNF ϕ labeled by f and $\vec{z}$ actually represents $f_{\vec{z}}$: $\vec{z}$ grounds some of the variables of $scope(f)$. Let $vars(N) = scope(f) \setminus Z$ denote the set of free variables of the function associated to N. For any internal node M, let $op(M)$ denote the label of M, and $Children(M)$ the set of its children. The set of free variables occurring in the VNNF ϕ rooted at node M is thus $vars(\phi) = vars(M) = \bigcup_{N \in Children(M)} vars(N)$. ϕ is said to be *grounded* when $vars(\phi) = \emptyset$.

Definition 4 (semantics of a VNNF) *Let ϕ be a VNNF w.r.t. $\langle \mathcal{E}, X, F \rangle$. The* semantics $val(\phi)$ *of ϕ is the function from $scope(val(\phi)) = vars(\phi)$ to E recursively defined by:*

- *If ϕ is a leaf node labeled by $f \in F$ and a Z-assignment $\vec{z}$, then $val(\phi) = f_{\vec{z}}$;*
- *Otherwise $\phi = \otimes(\phi_1, \ldots, \phi_n)$, and $val(\phi) = val(\phi_1) \otimes \ldots \otimes val(\phi_n)$.*

[1] Due to space reasons, proofs are omitted.

[2] A function is said to be polytime (resp. in linear time) when there exists a polynomial (resp. linear) time algorithm computing it.

A VNNF is thus simply the structured representation of a function ($val(\phi)$) that does not necessarily belong to the set of primitives F. Importantly if $\langle \mathcal{E}, X, F \rangle$ is tractable, then $val(\phi)$ is a polytime function.

Obviously, a given function over a valuation structure can be represented by many distinct but equivalent VNNFs. Since VNNFs are defined as DAGs, subformulas that would have several occurrences in tree-like representations do not need to be duplicated. But they can. It is always possible to simplify a VNNF by merging identical subgraphs. This reduction does not change the semantics and can be achieved by an algorithm similar to the one used for OBDDs. However, it does not lead to a canonical form in the general case (i.e., we do not necessarily have $val(\phi) = val(\psi)$ only if the reduced form of ϕ coincides with the reducd form of ψ).

Let us now "quantify" VNNFs. In classical logic, two quantifiers are used: $\exists$ and $\forall$. In the VNNF framework, every operator of OP can be used as a quantifier:

Definition 5 (Q-VNNF) *A (prenex)* quantified VNNF *(Q-VNNF) is a sentence of the form* $\Phi = \otimes_1 x_{i_1}, \ldots \otimes_j x_{i_j} \phi$ *where* ϕ *is a VNNF, each variable* x_{i_j} *belongs to* X,[3] *and each* $\otimes_i$ *belongs to* OP *(*$i \in 1 \ldots j$*).* $\otimes_1 x_{i_1}, \ldots \otimes_j x_{i_j}$ *is called the prefix of* Φ *and* ϕ *its matrix.*
The semantics of Φ*, denoted* $val(\Phi)$ *is the function from* $scope(val(\Phi)) = vars(\phi) \setminus \{x_{i_1}, \ldots, x_{i_j}\}$ *to* E *recursively defined as follows:*

- *If the prefix of* Φ *is empty, then* $val(\Phi) = val(\phi)$.
- *Otherwise,* Φ *is of the form* $\otimes y \Psi$ *and* $val(\Phi) = \bigotimes_{\vec{y} \in dom(y)} val(\Psi)_{\vec{y}}$.

Clearly enough, quantified VNNFs are convenient for representing at the syntactic level projections of functions. When $x_{i_1}, \ldots, x_{i_j}$ are pairwise distinct, Φ is said to be polite. When $vars(\phi) \subseteq \{x_{i_1}, \ldots, x_{i_j}\}$, the Q-VNNF Φ is said to be closed. Its semantics is an element of E. Computing it is what is called the *evaluation* of a (closed) Q-VNNF.

3 Generality of the VNNF Framework

Let us now show how the VNNF framework encompasses various representation settings pointed out so far.

Negation Normal Forms (NNFs) The propositional language NNF (Negation Normal Form formulas) is a well-known fragment for representing Boolean functions. For recovering it as a VNNF language, it is enough to set $E = \{0, 1\}$, $\succeq$ is such that $1 \succeq 0$, $dom(x) = E$ for each $x \in X$, F is the set of all Boolean functions of arity at most 1 (such functions can be represented by literals and Boolean constants), $OP = \{\min, \max\}$ ($\min$ nodes correspond to conjunctions of formulas and $\max$ nodes to disjunctions). Then $val(\phi)(\vec{x})$ is simply the truth value taken by the boolean function which is the semantics of ϕ when applied to $\vec{x}$. Quantified Boolean formulas in negation normal form are also easily recovered as Q-VNNFs: $\min x$ (resp. $\max x$) stands for the universal (resp. existential quantification) on variable x.

[3] If $Y = \{y_1, \ldots, y_k\} \subseteq X$ and $\otimes \in OP$, then we abbreviate $\otimes y_1 \ldots \otimes y_k$ by $\otimes Y$; this is harmless since each $\otimes \in OP$ is associative and commutative.

CSPs Recovering the standard CSP framework is also easy, considering the same valuation structure as for the NNF framework. No restriction is put on X (it can be any set of discrete variables), but for getting a CSP as a VNNF language we must add some restrictions on VNNF formulas. First of all, the root node has to be a $\min$ node. In general CSPs, each child of the root is a local function $f : scope(f) \rightarrow \{0, 1\}$. In table-defined CSPs, each child of the root is a $\max$ node, whose children are themselves $\min$ nodes connecting functions of the form $x == a$, that take value 1 if satisfied and value 0 otherwise. Quantified CSPs can also be viewed as specific Q-VNNFs, where $\min x$ (resp. $\max x$) stands for the universal (resp. existential quantification) on variable x.

Semiring CSPs and valuation algebras E is equipped with two operators: $OP = \{\vee, \otimes\}$ and $\succeq$ is defined by $a \succeq b$ iff $a \vee b = a$. It is moreover assumed that $\langle E, \succeq, \vee, \otimes \rangle$ is a commutative semiring and $\vee$ is idempotent. In a semiring CSP viewed as a VNNF, the root node of the DAG is a $\otimes$ node and each of its children is a local function. Interestingly, the definition of quantified semiring CSPs follows from our framework. In particular, if $\succeq$ is complete, then computing the best value of a semiring CSP ϕ is equivalent to compute $\max X \phi$ (note that $\min$ can also be added to OP and used as a quantifier). Valuation algebras [Shenoy and Shafer, 1988; Kohlas and Shenoy, 2000] can be more general than VNNF since valuations can bear on subsets of assignment, e.g. for representing commonality functions. When restricted to the distributional case they can be recovered in way similar to the one used for semiring CSPs.

Other frameworks for decision making under uncertainty The VNNF framework subsumes many of the constraint-based frameworks for decision making under uncertainty, e.g. mixed CSPs (they can be represented by disjunctions of a conjunction of constraints C_i, $i \in 1 \ldots n$ representing what is satisfying and, with a disjunction of constraints K_i, $i \in 1 \ldots m$ representing the knowledge about the state variables) and possibilistic mixed CSPs (each such CSP can be represented by a VNNF whose root is labeled by $\max$ and has two children: a $\min$-rooted VNNF and a $\max$-rooted one). Stochastic or more generally expected utility CSP networks can be represented as well by products between a VNNF ϕ_p representing the probability density over the state variables (e.g. a bayesian net) and a semiring CSP ϕ_u over $E = [0, 1]$ synthesizing the utility function. Denoting X the set of decision variables and Y the set of state variables, maximizing expected utility amounts to compute $\max X \sum Y \phi_p \ . \ \phi_u$. In the qualitative cases (mixed and possibilistic) a standard quantifying sequence is rather $\max X \min Y$. PFU generic networks [Pralet *et al.*, 2006]) are not stricto sensu subsumed by the VNNF framework, since they may involve a non-commutative combination of utility degrees and plausibility degrees. Nevertheless, non-commutativity is not a compulsory condition and most of the practical instances of PFU consider a commutative aggregator. Arithmetic circuits, as considered in [Darwiche, 2002], for the factored representation of belief networks can also be recovered as VNNFs.

Finally, we shall see in Section 5.3 that VNNFs also encompass decision diagrams like AADDs (and thus ADDs)

and SLDDs.

4 Queries and Transformations

The VNNF framework includes a number of algorithms for answering queries about functions represented as VNNFs, and related functions which can be characterized via the application of transformations on functions represented by VNNFs. A fundamental difference between the NNF setting and the VNNF one is that in the VNNF framework, the underlying valuation structure does not necessarily reduce to the Boolean one. Especially, since E may contain more than two elements, the equivalences $a \succ \bot$ iff $a = \top$ and $a \prec \top$ iff $a = \bot$ do not hold anymore. In the NNF framework, the key queries include consistency (CO), validity (VA), and model counting (CT), while the key transformations are conditioning (CD), conjunction ($\wedge$C), disjunction ($\vee$C) and forgetting (FO). Let us explain how to state related queries and transformations in the VNNF framework, and point out some additional ones which make sense in this more general setting.

Full consistency/validity If we consider $\top$ as the norm, we get very strong notions of consistency/validity: ϕ is fully consistent iff there exists $\vec{x}$ s.t. $val(\phi)(\vec{x}) = \top$ and ϕ is fully valid iff for each $\vec{x}$, $val(\phi)(\vec{x}) = \top$.

Partial consistency/validity On the contrary, one may consider every value of E except $\bot$ as the norm, and this leads to the following notions of partial consistency and partial validity: ϕ is partially consistent iff there exists $\vec{x}$ s.t. $val(\phi)(\vec{x}) \succ \bot$ and ϕ is partially valid iff for each $\vec{x}$, $val(\phi)(\vec{x}) \succ \bot$.

Optimal satisfaction More interesting in practice is the notion of optimal satisfaction, that looks for best assignments; this query is typically useful when VNNFs encode preferences or plausibility degrees and calls for *optimization*: given a VNNF ϕ, find $\vec{x}$ and $a \in E$ such that $val(\phi)(\vec{x}) = a$ and such that there is no $\vec{y}$ such that $val(\phi)(\vec{y}) \succ a$. We thus search for a non-dominated solution. When $\succeq$ is a total order, this is equivalent to classical maximization. The associated decision problem (deciding whether a value is optimal) consists in determining whether, given $a \in E$ and a VNNF ϕ, a is the value taken by ϕ for some non-dominated solution.

Counting extends easily from NNF to VNNF: Given $a \in E$ and a VNNF ϕ, how many $\vec{x}$ are such that $val(\phi)(\vec{x}) = a$?

Evaluation of a quantified form The extended notion of quantification naturally leads to the problem of evaluating a (closed and polite) Q-VNNF. As a decision problem, this can be written as determining whether $val(\Phi) \succeq a$. This query is of great importance in many settings: in the NNF one, this query generalizes the famous PSPACE-complete problem called QBF. Quantification also makes sense when $E \neq \{0, 1\}$. When $\succeq$ is a total order, evaluating a Q-VNNF is a way to achieve optimization (find the best value of ϕ amounts to evaluating $\max X\phi$). The relationship between optimization and quantified forms is less obvious when partial orders are considered (the identification of an operator encoding non-domination is not an easy problem and may have no solution; for instance, computing a non-dominated value for a semiring CSP represented by a VNNF ϕ is not equivalent to evaluate the quantified one $\vee X\phi$).

We have obtained the following complexity results for VNNF assuming that the underlying representation context is tractable (which is a reasonable assumption that we make from now on up to the end of the paper):

Proposition 2

- *Partial and full consistency are* NP*-complete.*
- *Partial and full validity are* coNP*-complete.*
- *Optimization is* NP*-hard*
- *Optimal value is* DP*-complete.*
- *Counting is* #P*-hard.*
- *Evaluation of a closed Q-VNNF is* PSPACE*-complete.*

Observe that, though VNNF is a much more general framework than the NNF one, the generalization does not lead to a complexity shift w.r.t. the basic queries under consideration in [Darwiche and Marquis, 2002]. Note that we could also define notions of full entailment and full equivalence and show the corresponding decision problems coNP-complete.

Let us now focus on transformations. Three transformations are mainly to be considered; conditioning (compute a VNNF representing $val(\phi)_{\vec{z}}$), $\otimes$-combination (compute a VNNF representing $val(\phi_1) \otimes val(\phi_2)$) and $\otimes$-variable elimination (compute a VNNF representing $val(\phi)_{[y,\otimes]} = val(\otimes y\phi)$, or more generally $val(\otimes Y\phi)$).

The case of $\otimes$-combination is obvious since $\phi_1 \otimes \phi_2$ is a VNNF and represents $val(\phi_1) \otimes val(\phi_2)$. If the representation context is tractable, then each $\otimes$-combination is in linear time. If ϕ is a VNNF, then a VNNF representing $val(\phi)_{\vec{z}}$ can also be obtained in linear time in the size of ϕ. It is enough to revise the assignment associated to ϕ:

Definition 6 (conditioning a VNNF) *Given a VNNF ϕ, we denote $\phi_{\vec{z}}$ the VNNF obtained by replacing in the label of each leaf, the Y-assignment $\vec{y}$ by $\vec{y} \circ \vec{z}$, the $Y \cup Z$-assignment which coincides with $\vec{y}$ on each variable from Y and with $\vec{z}$ on each variable from $Z \setminus Y$.*

It can be easily checked that $\phi_{\vec{z}}$ represents the restriction of $val(\phi)$ by $\vec{z}$, i.e. that $val(\phi_{\vec{z}}) = val(\phi)_{\vec{z}}$. That is why we call $\phi_{\vec{z}}$ the conditioning of ϕ by $\vec{z}$.

$\otimes$-variable elimination generalizes the transformation known as forgetting in the NNF case. In contrast to the previous transformations, it can be very expensive: at each elimination step, the size of the current VNNF ϕ may increase in a non-constant way (the size of $\bigotimes_{\vec{x} \in dom(x)} \phi_{\vec{x}}$ can be $|dom(x)|$ larger than the size of ϕ). Thus, applying the definition directly would lead to an exponentially larger VNNF, unless a bounded number of variables has to be eliminated.

5 Determinism, Decomposability, Decision and Read-Once

The previous sections have shown the VNNF framework quite general. However, every query under consideration, if it is no more difficult than in the Boolean case, is intractable under the standard assumptions of complexity theory. An important issue is thus to define restrictions on the VNNF languages allowing efficient queries and transformations.

In the NNF framework, a few properties, namely decomposability, determinism, decision and read-once are sufficient to achieve many of them in polynomial time. We shall see that a third one is valuable in the VNNF framework, namely distributivity. The key point is that it enables efficient $\otimes$-variable elimination, which is an important transformation for several issues, including the optimization one.

5.1 Distributivity and decomposability

The space explosion inherent to a direct application of $\otimes$-variable elimination in the general case does not necessarily occur; especially, it can be limited when some of the children of ϕ do not depend on the variable x to be eliminated. Thus, in the DNNF case, at most one child of each $\wedge$ node may depend on the variable to be forgotten. Actually, in addition to independence, a further property is implicitly used in the NNF framework for ensuring polytime forgetting, and it has to be made explicit in order to be extended to VNNF: the distributivity of $\wedge$ over $\vee$. The impact of distributivity on variable elimination in valuation algebras is known for a while (see e.g. [Shenoy and Shafer, 1988]) and requiring it is not so demanding in many representation contexts.

Definition 7 ($\otimes$-distributivity) *A VNNF ϕ ensures* $\otimes$-distributivity *iff for any $\odot$ labelling an internal node in ϕ, $\odot$ distributes over $\otimes$: $\forall a, b, c \in E : a \odot (b \otimes c) = (a \odot b) \otimes (a \odot c)$.*

Proposition 3

- *Any VNNF w.r.t. any representation context ensures $\otimes$-distributivity for $\otimes = \vee$ and for $\otimes = \wedge$.*
- *Any VNNF w.r.t. any representation context where $\succeq$ is a total order ensures $\otimes$-distributivity for $\otimes = max$ and for $\otimes = min$.*

We are now ready to define the fragment of $\otimes$-*decomposable* VNNFs:

Definition 8 ($\otimes$-decomposability) *Let ϕ be a VNNF:*

- *A node N of ϕ is* simply decomposable *iff for each $N_i, N_j \in Children(N)$, $vars(N_i) \cap vars(N_j) = \emptyset$ when $i \neq j$.*
- *A node N of ϕ is* $\otimes$-decomposable *iff it is simply decomposable and $op(N)$ is distributive over $\otimes$.*
- *ϕ is* $\otimes$-decomposable *iff each of its (internal) nodes N is $\otimes$-decomposable when $op(N) \neq \otimes$. $\otimes$-DVNNF is the class of all the $\otimes$-decomposable VNNFs.*

As to $\otimes$-decomposability, the main result is:

Proposition 4 *$\otimes$-DVNNF is linearly closed for $\otimes$-variable elimination, i.e., there exists a linear time algorithm for computing a $\otimes$-DVNNF equivalent to $\otimes Y \phi$ when $\phi \in \otimes$-DVNNF and $Y \subseteq X$.*

This generalizes the result of tractability of DNNF for the forgetting operation. It moreover implies that the optimization of any VNNF the nodes of which are either $\max$ nodes or decomposable nodes is tractable.

Proposition 5 *The restrictions of full consistency, partial consistency and optimisation on* $\max$*-DVNNF are in* P.

5.2 Determinism

A second important property in the NNF framework is determinism. More than a property of logical exclusion, it is linked to the existence of a neutral element for $\otimes$:

Definition 9 (determinism) *An internal node N in a VNNF ϕ is* deterministic *iff for each $\vec{y} \in dom(vars(N))$, there is at most one $M \in Children(N)$ such that $val(\psi)(\vec{y}) \neq e_{op(N)}$, where ψ is the VNNF rooted at M.*

Definition 10 (d-DVNNF) *d-DVNNF is the class of all VNNFs in which each internal node is either simply decomposable or deterministic.*

As to d-DVNNF, the main result is:

Proposition 6 *The restrictions of full validity and partial validity on d-DVNNF are in* P. *If ϕ is a d-DVNNF, then counting can be achieved in time $\mathcal{O}(|E| + |\phi|)$.*

5.3 Decision diagrams

Let us now focus on other properties that prove useful for defining another interesting fragment of VNNF: the set of all decision diagrams. We first need a number of definitions:

Definition 11 (literals) *A* literal *on $x \in X$ is a function f whose scope is $\{x\}$. L_X is the set of all the literals that can be built on variables from X.*

By extension, we shall also call literals the leaves labeled by literals.

Definition 12 (assignment nodes) *An* assignment node *N on $x \in X$ in a VNNF ϕ is a node of the form $l \otimes \alpha$, where l is a literal on x and $\otimes$ is a t-norm on E.*
Two assignment nodes N_1 and N_2 on $x \in X$ are exclusive *iff their respective literals f_1 and f_2 are exclusive, i.e., $\forall \vec{x} \in dom(x)$, either $f_1(\vec{x}) = \bot$ or $f_2(\vec{x}) = \bot$.*

In any assignment node N of the form $l \otimes \alpha$ where l is a literal on x one may assume without loss of generality that α is not a literal (a literal f can always be replaced by the assignment node $f \wedge \top$). Then $dvar(N)$ can denote the variable x and $tail(N)$ the formula α in a non-ambiguous way.

Definition 13 (decision nodes) *A* decision node *N on x in a VNNF ϕ is a node of the form $N_1 \odot \ldots \odot N_m$ where all the N_i are assignment nodes on the same variable x and $\odot$ is a t-conorm on E. N is* exclusive *iff its assignment nodes are pairwise exclusive.*

Definition 14 (linear node) *A node N is* linear *iff at most one of its children is not grounded.*

Note that leaves are linear nodes. We are now ready to define the language of decision diagrams:

Definition 15 (decision diagrams) *A* decision diagram *is a VNNF in which each internal node is either an exclusive decision node or a linear node.*

Two particular subclasses of decision diagrams are worthwhile noticing, the read-once ones and the ordered ones:

Definition 16 (read-once and ordered decision diagrams)

- *A decision diagram ϕ is* read-once *iff for any assignment node N in ϕ, $dvar(N)$ does not occur in $tail(N)$.*

- *A decision diagram ϕ is* ordered *w.r.t. a strict order $<$ on X iff for every pair of decision nodes M and N in ϕ, if M is an ancestor of N in ϕ, then $dvar(M) < dvar(N)$.*

The second property obviously implies the first one. Interestingly, AADDs and SLDDs are specific decision diagrams:

AADD For recovering algebraic decision diagrams, let us set $E = \mathbb{R} \cup \{-\infty, -\infty\}$, $\succeq = \geq$, $OP = \{min, max, +, .\}$, $\forall x \in X, dom(x) = \{0, 1\}$, $F = L_X$. The decision nodes of such decision diagrams are of the form $N_x = max(min(f_h, c_h + b_h.N_h), min(f_l, c_l + b_l.N_l))$, where:

- f_h (resp. f_l) is the literal on x that valuates to $-\infty$ for $x = 0$ (resp. $x = 1$) and to $+\infty$ for $x = 1$ (resp. $x = 0$);
- c_h, b_h, c_l, b_l are constants of E;
- N_h and N_l are either decision nodes or constant nodes.

The root of an AADD is a linear formula $c+b.N_k$. The reader can check that $min(f_h, c_h + b_h.N_h)$ and $min(f_l, c_l + b_l.N_l)$ are exclusive assignment nodes whose tails are linear ones.

SLDD A SLDD is an ordered decision diagram built on some set E ordered by some $\succeq$ and equipped with two operators $\oplus$ and $\otimes$, where $\otimes$ is a t-norm and $\oplus$ is a t-conorm. It is moreover required that $\otimes$ distributes over $\oplus$: $\langle E, \bot, \top, \oplus, \otimes\rangle$ is a commutative semiring. For recovering SLDD, it is sufficient to set F to the subset of literals of L_X ranging over $\{\bot, \top\}$.

and to consider ordered decision diagrams whose decision nodes are of the form $N = \vee(\wedge(l_1, c_1 \otimes N_1), \ldots, \wedge(l_k, c_k \otimes N_k))$, where the c_i are constants of E.

SLDDs and AADDs are obviously read-once. This is important enough, since one can prove that the read-once property on decision diagrams ensure that they are not only decomposable but also deterministic:

Proposition 7 *Read-once decision diagrams are d-DVNNFs.*

As a consequence of Propositions 5, 6 and 7, we get:

Proposition 8 *The restrictions of full and partial consistency, optimization, full and partial validity on read-once (and thus on ordered) decision diagrams are in* P. *Counting can be achieved in time $\mathcal{O}(|E| + |\phi|)$.*

Now, it can be shown that the $\otimes$-elimination of a variable x can be performed in an efficient way on a decision diagram ϕ for which distributivity w.r.t. $\otimes$ is ensured, provided that x is final in ϕ (i.e., for any assignment node N on x in ϕ, $tail(N)$ is grounded).

Definition 17 (distributive Q-VNNF) *A Q-VNNF is said to be* distributive *iff each operator of its matrix distributes over each operator of its prefix.*

For instance, when E is totally ordered, any quantified decision diagram (e.g. a quantified AADD or a quantified ADD) is distributive for a prefix composed of existential ($\max$) and universal ($\min$) quantifiers.

Proposition 9 *Let $<$ be the total ordering on X defined by the prefix of a (closed and polite) Q-VNNF Φ, i.e. $x_{i_1} < \ldots < x_{i_j}$ iff $\Phi = \otimes_1 x_{i_1}, \ldots, \otimes_j x_{i_j} \phi$. The evaluation problem for a distributive, closed and polite Q-VNNF the matrix of which is a decision diagram ordered by $<$ is in* P.

6 Conclusion

In this paper, we have presented a new framework, VNNF, which encompasses many representation settings pointed out so far. We have shown how some properties imposed on NNFs to define more "tractable" fragments (namely, decomposability, determinism, decision, read-once) can be extended to VNNFs.

This work calls for a number of perspectives. The more immediate ones are of algorithmic nature: we need algorithms for compiling VNNFs into $\otimes$-DVNNFs, d-DVNNFs, or (ordered) decision diagrams. From a more theoretical (but not less important) point of view, a number of issues need to be addressed as well, including the succinctness one (how do the VNNF fragments w.r.t. a given representation context relate w.r.t. spatial efficiency?) and the canonicity one (under which requirements can we guarantee that two VNNFs represent the same function iff they are identical?).

References

[Bahar *et al.*, 1993] R.I. Bahar, E.A. Frohm, C.M. Gaona, G.D. Hachtel, E. Macii, A. Pardo, and F. Somenzi. Algebraic Decision Diagrams and Their Applications. In *Proc. ICAD'93*, pages 188–191, 1993.

[Bistarelli *et al.*, 1995] S. Bistarelli, U. Montanari, and F. Rossi. Constraint solving over semirings. In *Proc. IJCAI'95*, pages 624–630, 1995.

[Darwiche and Marquis, 2002] A. Darwiche and P. Marquis. A knowledge compilation map. *Journal of Artificial Intelligence Research*, 17:229–264, 2002.

[Darwiche, 2001] A. Darwiche. Decomposable negation normal form. *JACM*, 48(4):608–647, 2001.

[Darwiche, 2002] A. Darwiche. A logical approach to factoring belief networks. In *Proc. KR'02*, pages 409–420, 2002.

[Kohlas and Shenoy, 2000] J. Kohlas and P.P. Shenoy. Computation in valuation algebras. In J. Kohlas and S. Moral, editors, *Handbook of DRUMS, Volume 5*, pages 5–39. Kluwer, Dordrecht, 2000.

[Pralet *et al.*, 2006] C. Pralet, G. Verfaillie, and T. Schiex. Decision with uncertainties, feasibilities, and utilities: towards a unified algebraic framework. In *Proc. ECAI'06*, 2006.

[Sanner and McAllester, 2005] S. Sanner and D. A. McAllester. Affine algebraic decision diagrams (AADDs) and their application to structured probabilistic inference. In *Proc. IJCAI'05*, pages 1384–1390, 2005.

[Schiex *et al.*, 1995] T. Schiex, H. Fargier, and G. Verfaillie. Valued constraint satisfaction problems: Hard and easy problems. In *Proc. IJCAI'05*, pages 631–639, 1995.

[Shenoy and Shafer, 1988] P. Shenoy and G. Shafer. Axioms for probability and belief-function propagation. In *Proc. UAI'88*, pages 169–198, 1988.

[Wilson, 2005] N. Wilson. Decision diagrams for the computation of semiring valuations. In *Proc. IJCAI'05*, pages 331– 336, 2005.