Compiling Solution Configurations in Semiring Valuation Systems

Marc Pouly[1], Rolf Haenni[2,3], and Michael Wachter[3]

[1] University of Fribourg, Switzerland
marc.pouly@unifr.ch
[2] Bern University of Applied Sciences, Switzerland
rolf.haenni@bfh.ch
[3] University of Bern, Switzerland
{wachter,haenni}@iam.unibe.ch

Abstract. This paper describes a new method for solving optimization queries in semiring valuation systems. In contrast to existing techniques which focus essentially on the identification of solution configurations, we propose foremost the construction of an implicit representation of the solution configuration set in the shape of a Boolean function. This intermediate compilation step allows then to efficiently execute many further relevant queries that go far beyond the traditional task of enumerating solution configurations.

1 Introduction

A valuation algebra [1–3] is an abstract system that unifies various formalisms of knowledge representation which are all based on the two principal operations of combination and variable elimination. Based on this generic framework, a collection of efficient inference algorithms was developed which deal successfully with the exponential nature behind the operations. These computational architectures are referred to as local computation algorithms. Thereupon, [4] remarked that the general inference problem turns into an optimization task when dealing with valuation algebras whose variable elimination operator corresponds in some way to either minimization or maximization. Consequently, an extended local computation scheme was proposed that allows furthermore to identify the configurations that map to the computed optimum value. Under this perspective, local computation methods turn into *non-serial dynamic programming* [5]. However, practical applications in diagnosis often make further demands that go far beyond the identification of solution configurations. Possible scenarios range from the simple requirement of counting solution configurations without their explicit enumeration to the question about the probability of single configurations or the equivalence of objective functions. To allow for these requirements, we propose the construction of a Boolean function whose models collapse with the solution configurations we are looking for. By choosing then a suitable representation of the latter, we obtain a very compact graphical structure that makes the fast evaluation of such queries possible. Because this compilation process is by itself based on local computation, we inherit also its efficiency. With this proposition, we follow recent ambitions [6, 7] in combining local computation with knowledge compilation techniques [8, 9].

A. Gelbukh and A.F. Kuri Morales (Eds.): MICAI 2007, LNAI 4827, pp. 248-259, 2007.

This paper is organized in the following way. We first give a quick introduction to valuation algebras, the generic inference problem and the fusion algorithm for its efficient solution. Next, we introduce a particular class of valuation algebras that map configurations to semiring values. On the according semiring, we can identify sufficient conditions such that the inference problem turns into an optimization task. This rounds off the needed background and allows us to introduce this new technique of solving optimization tasks in semiring valuation systems.

2 Valuation Algebras

The basic elements of a valuation algebra are so-called *valuations*. Intuitively, a valuation can be regarded as a representation of knowledge about the possible values of a set of variables r. It can be said that each valuation ϕ refers to a finite set of variables $d(\phi) \subseteq r$, called its *domain*. For an arbitrary set $s \subseteq r$ of variables, Φ_s denotes the set of valuations ϕ with $d(\phi) = s$. With this notation, the set of all possible valuations over r can be defined as

$$\Phi = \bigcup_{s \subseteq r} \Phi_s.$$

Let 2^r be the power set[4] of r and Φ a set of valuations with their domains in 2^r. We assume the following operations defined in $(\Phi, 2^r)$:

- *Labeling:* $\Phi \to 2^r$; $\phi \mapsto d(\phi)$,
- *Combination:* $\Phi \times \Phi \to \Phi$; $(\phi, \psi) \mapsto \phi \otimes \psi$,
- *Variable Elimination:* $\Phi \times r \to \Phi$; $(\phi, X) \mapsto \phi^{-X}$.

These are the three basic operations of a valuation algebra. We impose now the following set of axioms:

1. *Commutative Semigroup:* Φ is associative and commutative under $\otimes$.
2. *Labeling:* For $\phi, \psi \in \Phi$, $d(\phi \otimes \psi) = d(\phi) \cup d(\psi)$.
3. *Variable Elimination:* For $\phi \in \Phi$ and $X \in d(\phi)$,
$$d(\phi^{-X}) = d(\phi) - \{X\}.$$
4. *Commutativity of Elimination:* For $\phi \in \Phi_s$ and $X, Y \in s$,
$$(\phi^{-X})^{-Y} = (\phi^{-Y})^{-X}.$$
5. *Combination:* For $\phi, \psi \in \Phi$ with $X \notin d(\phi)$, $X \in d(\psi)$,
$$(\phi \otimes \psi)^{-X} = \phi \otimes \psi^{-X}.$$

Instances of valuation algebras are large in number and occur in very different contexts. Among them are probability mass functions used in Bayesian networks, Dempster-Shafer belief functions, relations from database theory or more general constraint systems, various kinds of logics and many more. We refer to [3] for an extensive listing of instances.

[4] The more general definition given in [3] considers arbitrary distributive lattices. In this case, we must replace the operation of variable elimination by marginalization.

3 Inference Problem & Local Computation

The computational interest in valuation algebras is generally resumed as *inference problem*. Given a set of valuations $\{\phi_1, \ldots, \phi_n\}$, we need to eliminate all variables from the joint valuation $\phi = \phi_1 \otimes \cdots \otimes \phi_n$ that do not belong to some query $x \subseteq r$. More formally, this consists in the computation of

$$\phi^{-\{X_1,\ldots,X_k\}} = (\phi_1 \otimes \cdots \otimes \phi_n)^{-\{X_1,\ldots,X_k\}} \tag{1}$$

for $\{X_1, \ldots, X_k\} = d(\phi) - x$. Note that we are allowed to eliminate sets of variables because their elimination order is not decisive (Axiom 4). It is however well-known that an explicit computation of the joint valuation is out of question because in most cases, the complexity of valuation algebra operations tends to increase exponentially with the size of the involved factor domains. Local computation methods counteract this problem by organizing the computations in such a way that factors do not grow significantly [3]. In the following, we restrict our attention to one such algorithm called *fusion algorithm* [10] and refer to [11] for a broad discussion of further local computation schemes.

In order to describe the fusion algorithm, we consider first the elimination of a single variable Y from a set of valuations. This operation can be performed as follows:

$$\mathrm{Fus}_Y(\{\phi_1, \ldots, \phi_n\}) = \{\psi^{-Y}\} \cup \{\phi_i : Y \notin d(\phi_i)\},$$

where

$$\psi = \bigotimes_{i:Y \in d(\phi_i)} \phi_i.$$

The fusion algorithm follows then by a repeated application of this operation:

$$\phi^{-\{X_1,\ldots,X_k\}} = \bigotimes \mathrm{Fus}_{X_k}(\ldots(\mathrm{Fus}_{X_1}(\{\phi_1, \ldots, \phi_n\})).$$

We refer to [3] for a proof and further considerations regarding the complexity of this generic inference algorithm.

4 Semirings

Semirings are algebraic structures with two binary operations $+$ and $\times$ over a set of values A. We call a tuple $\mathcal{A} = \langle A, +, \times \rangle$ a *semiring* if both operations $+$ and $\times$ are associative and commutative and if $\times$ distributes over $+$. Elsewhere, this is also called a *commutative semiring*. If there is an element $\mathbf{0} \in A$ such that $\mathbf{0} + a = a + \mathbf{0} = a$ and $\mathbf{0} \times a = a \times \mathbf{0} = \mathbf{0}$ for all $a \in A$, then $\mathcal{A}$ is called a semiring with *zero element*. A zero element can be adjoined to any semiring such that we can assume its existence without loss of generality. If addition is idempotent, i.e. if $a + a = a$ for all $a \in A$, we call $\mathcal{A}$ an *idempotent semiring*. Finally, a semiring element $\mathbf{1} \in A$ is said to be a *unit element* if $\mathbf{1} \times a = a \times \mathbf{1} = a$ for all $a \in A$. We can extend any idempotent semiring to include a unit element and therefore, we assume subsequently that all idempotent semirings possess a unit element. The following table lists some of the most famous semiring examples which will later be reconsidered in the discussion of semiring valuation algebras.

	A	$+$	$\times$	**0**	**1**	**idempotent**
1	$\mathbb{R}_{\geq 0}$	$+$	$\cdot$	0	1	$\times$
2	$\mathbb{R} \cup \{\pm\infty\}$	max	min	$-\infty$	∞	$\checkmark$
3	$\mathbb{B} = \{0, 1\}$	$\vee$	$\wedge$	0	1	$\checkmark$
4	$\mathbb{N} \cup \{0, \infty\}$	min	$+$	∞	0	$\checkmark$
5	$\mathbb{R} \cup \{-\infty\}$	max	$+$	$-\infty$	0	$\checkmark$
6	$[0, 1]$	max	t-norm	0	1	$\checkmark$
7	$\{f : \mathbb{B}^v \to \mathbb{B}\}$	max	min	f_0	f_1	$\checkmark$

The first semiring is called *Arithmetic Semiring* and consists of the non-negative reals together with the usual operations of addition and multiplication. The second example is named *Bottleneck Semiring* and often used in the context of optimization tasks. Next follows the famous *Boolean Semiring* with disjunction and conjunction for addition and multiplication respectively. Examples 4 and 5 are known as *Tropical Semirings* and Example 6 shows that maximization together with any t-norm induces again a semiring. T-norms are binary operations on the unit interval which are commutative, associative, have the number 1 as unit element and are non-decreasing in both arguments [12]. Finally, Example 7 shows the semiring of Boolean functions over a set of propositional variables v. Here, f_0 (f_1) denominates the constant Boolean function that always maps to 0 (1).

4.1 Ordered Semirings

On every idempotent semiring $\mathcal{A}$, we can introduce a relation $\leq_{id}$ by:

$$a \leq_{id} b \text{ if and only if } a + b = b.$$

This relation is a partial order and both semiring operations $+$ and $\times$ are monotone with respect to it. Furthermore, we have for all elements $a, b \in A$ that $a + b = \sup\{a, b\}$. However, in many interesting semirings the relation $\leq_{id}$ is actually a total order which leads to the following refinement of this result:

Lemma 1. *If $\leq_{id}$ is total, we have $a + b = \max\{a, b\}$.*

Finally, we require another property to guarantee that all solution configurations are captured during the compilation process. Indeed, it was shown in [13] that only a non-empty subset of the solution configurations are found in case of its absence. Such an example is given by the Bottleneck semiring in the above table.

Definition 1. *An idempotent semiring is called strictly monotonic over $\times$ if for $c \neq \mathbf{0}$, $a <_{id} b$ implies that $a \times c <_{id} b \times c$.*

5 Semiring Valuation Algebras

Let r be a set of propositional variables[5]. Without abandoning the usual interpretation of frame elements as truth values, we denote the frame of variable X as $\Omega_X = \{0_X, 1_X\}$.

[5] The theory of semiring induced valuation algebras can be developed with arbitrary finite variables. Here, we restrict ourselves to propositional variables for simplicity reasons.

Correspondingly, the frame Ω_s of a non-empty variable set $s \subseteq r$ is built by the Cartesian product

$$\Omega_s = \prod_{X \in s} \Omega_X.$$

The Boolean vectors $\mathbf{x} \in \Omega_s$ are called configurations of s, and by convention, we define the frame of the empty variable set as $\Omega_\emptyset = \{\diamond\}$. Furthermore, we write $\mathbf{x}^{\downarrow t}$ for the projection of some configuration $\mathbf{x} \in \Omega_s$ to a subset t of s. In particular, we have $\mathbf{x}^{\downarrow \emptyset} = \diamond$.

A *semiring valuation* ϕ with domain $s \subseteq r$ is defined to be a function that associates a value from a given semiring $\mathcal{A} = \langle A, +, \times \rangle$ with each configuration $\mathbf{x} \in \Omega_s$, i.e.

$$\phi : \Omega_s \ \rightarrow \ A.$$

Again, we refer to the set of all semiring valuations over variables in s as Φ_s and define Φ to be their union over all subsets $s \subseteq r$. Thus, the operation of labeling for semiring valuations can be defined as follows:

1. *Labeling:* $d(\phi) = s$ if $\phi \in \Phi_s$.

Next, we define the operations of combination and variable elimination for semiring valuations in terms of the semiring operations $+$ and $\times$:

2. *Combination:* $\phi \in \Phi_s$, $\psi \in \Phi_t$ and $\mathbf{x} \in \Omega_{s \cup t}$

$$\phi \otimes \psi(\mathbf{x}) = \phi(\mathbf{x}^{\downarrow d(\phi)}) \times \psi(\mathbf{x}^{\downarrow d(\psi)}). \tag{2}$$

3. *Variable Elimination:* $\phi \in \Phi_s$, $X \in s$ and $\mathbf{x} \in \Omega_{s-\{X\}}$

$$\phi^{-X}(\mathbf{x}) = \phi(\mathbf{x}, 0_X) + \phi(\mathbf{x}, 1_X). \tag{3}$$

Theorem 1. *A system of semiring valuations with labeling, combination and variable elimination as defined above, satisfies the axioms of a valuation algebra.*

A proof of this important theorem can be found in [14]. The insight that every semiring induces a valuation algebra foreshadows the richness of formalisms that are covered by this theory. If we take for example the Arithmetic Semiring restricted to the unit interval, we obtain the valuation algebra of probability potentials [1] which represent the conditional probability functions in Bayesian networks. In the same way, the Boolean semiring induces the valuation algebra of indicator functions that is used in various constraint-based applications [15] and, as another example, the t-norm semiring leads to the valuation algebra of possibility potentials [16].

6 Optimization Problems

The inference problem, as stated in Equation (1), adopts a very special meaning in the case of valuation algebras that are induced by totally ordered idempotent semirings. According to Lemma 1 we obtain by elimination of a subset $t \subseteq d(\phi) = s$ of variables

$$\phi^{-t}(\mathbf{x}) = \sum_{\mathbf{y} \in \Omega_t} \phi(\mathbf{x}, \mathbf{y}) = \max\{\phi(\mathbf{x}, \mathbf{y}),\ \mathbf{y} \in \Omega_t\}.$$

In particular, we obtain after eliminating all variables in s

$$\phi^{-s}(\diamond) = \max\{\phi(\mathbf{x}),\ \mathbf{x} \in \Omega_s\}. \tag{4}$$

Thus, the inference problem consists in the computation of the maximum value of ϕ, and we refer to configurations that adopt this value as *solution configurations*.

Definition 2. *Let (Φ, D) be a valuation algebra induced by a totally ordered idempotent semiring. For $\phi \in \Phi_s$, we call $\mathbf{x} \in \Omega_s$ a solution configuration if $\phi(\mathbf{x}) = \phi^{-s}(\diamond)$.*

Example 1. Let ϕ be an semiring valuation defined over propositional variables A, B, C, and induced by the Tropical Semiring with maximization. We observe the result of full variable elimination:

A	B	C	ϕ
0	0	0	5
0	0	1	10
0	1	0	10
0	1	1	3
1	0	0	9
1	0	1	5
1	1	0	7
1	1	1	3

$\rightsquigarrow$

A	B	$\phi^{-\{C\}}$
0	0	10
0	1	10
1	0	9
1	1	7

$\rightsquigarrow$

A	$\phi^{-\{B,C\}}$
0	10
1	9

$\rightsquigarrow$

	$\phi^{-\{A,B,C\}}$
$\diamond$	10

We see that $\phi^{-\{A,B,C\}}(\diamond)$ is indeed the maximum value of ϕ and that $(0_A, 0_B, 1_C)$ and $(0_A, 1_B, 0_C)$ are its solution configurations.

7 Compiling Solution Configurations

This section is dedicated to the task of identifying solution configurations of some joint valuation ϕ that is given by a factorization according to Equation (1). For this purpose, [4] proposed an extension of the fusion algorithm which allows to identify single solution configurations. Whenever a variable is eliminated during the fusion process, this algorithm stores the frame value of the eliminated variable that leads to the maximum value. In this way, solution configurations are obtained at the end of fusion by combining the retained frame values, and from this perspective, the extended fusion algorithm corresponds to *non-serial dynamic programming*. A generalization of this algorithm to valuation algebras induced by idempotent semirings can be found in [13]. This paper embarks on another strategy. Instead of storing partial solution configurations (or frame values) explicitly, we rather represent them in an implicit way by constructing a Boolean function whose set of models corresponds to the solution configurations of ϕ. In short, we *compile* solution configurations into a Boolean function. This idea is sketched in the following example.

Example 2. Here, we use propositional formulae as an appropriate representation of Boolean functions. Doing so, the solution configurations found in Example 1 are the models of the Boolean function that corresponds to the following propositional formula:

$$\neg A \wedge ((\neg B \wedge C) \vee (B \wedge \neg C)).$$

We already discussed the semiring that consists of all Boolean functions over a finite set of propositional variables r with minimization for $\times$ and maximization for $+$ respectively. Such a Boolean function f can likewise be viewed as a set of Boolean vectors $\mathbf{x} \in \Omega_r$ for which f evaluates to 1. We refer to these vectors as the *models* of the Boolean function f. Besides the already introduced constant function f_0 and f_1, it is very convenient for our purposes to identify the Boolean function that reflects the value of some variable X. Thus, f_X represents the Boolean function that evaluates to 1 if and only if variable $X \in r$ adopts value 1. Accordingly, we write $f_{\overline{X}}$ to denote its inverse. This notation allows to introduce the concept of *memorizing semiring valuations*.

7.1 Memorizing Semiring Valuations

Let $\mathcal{A} = \langle A, +, \times \rangle$ be a totally ordered idempotent semiring, and r a finite set of propositional variables. A memorizing semiring valuation ϕ with domain $s \subseteq r$ is defined to be a function that associates a two-dimensional vector with each configuration $\mathbf{x} \in \Omega_s$. The first value of this vector $\phi_A(\mathbf{x})$ corresponds again to a semiring value, whereas the second $\phi_F(\mathbf{x})$ constitutes a Boolean function defined over r. More formally, we have $\phi : \Omega_s \to A \times F_r$ and $\mathbf{x} \mapsto (\phi_A(\mathbf{x}), \phi_F(\mathbf{x}))$ where F_r denotes the set of Boolean functions over r. We will again denote the set of all memorizing semiring valuations with domain $s \subseteq r$ by Φ_s and use Φ for all memorizing semiring valuations defined over subsets of r. This definition extends usual semiring valuations by attaching a Boolean function to every configuration $\mathbf{x} \in \Omega_s$ which will be used during the fusion process to memorize the frame values of eliminated variables that are part of the solution configurations. This is reflected in the following definitions of operations for memorizing semiring valuations:

1. *Labeling:* $\Phi \to D$: $d(\phi) = s$ if $\phi \in \Phi_s$.
2. *Combination:* $\phi \in \Phi_s$, $\psi \in \Phi_t$ and $\mathbf{x} \in \Omega_{s \cup t}$

$$\phi \otimes \psi(\mathbf{x}) = (\phi_A(\mathbf{x}^{\downarrow s}) \times \psi_A(\mathbf{x}^{\downarrow t}), \min\{\phi_F(\mathbf{x}^{\downarrow s}), \psi_F(\mathbf{x}^{\downarrow t})\}).$$

Combination of memorizing semiring valuations is defined for both vector components independently. The two values from semiring $\mathcal{A}$ are again combined by the corresponding generic semiring operation $\times$, and the same holds for the Boolean functions because in this semiring $\times$ corresponds to minimization. The semantics of this definition is that in case of combination, we combine conjunctively the memories of both factors ϕ and ψ.

3. *Variable Elimination:* $\phi \in \Phi_s$, $Y \in s$ and $\mathbf{x} \in \Omega_{s-\{Y\}}$

$$\phi^{-Y}(\mathbf{x}) = (\phi_A^{-Y}(\mathbf{x}),\ \phi_F^{-Y}(\mathbf{x}))$$

where

$$\phi_A^{-Y}(\mathbf{x}) \ = \ \phi_A(\mathbf{x}, 0_Y) + \phi_A(\mathbf{x}, 1_Y)$$

and

$$\phi_F^{-Y}(\mathbf{x}) = \begin{cases} \min\{f_Y,\ \phi_F(\mathbf{x}, 1_Y)\}, \text{ if } \phi_A(\mathbf{x}, 1_Y) >_{id} \phi_A(\mathbf{x}, 0_Y), \\ \min\{f_{\overline{Y}},\ \phi_F(\mathbf{x}, 0_Y)\}, \text{ if } \phi_A(\mathbf{x}, 1_Y) <_{id} \phi_A(\mathbf{x}, 0_Y), \\ \max\{\min\{f_Y, \phi_F(\mathbf{x}, 1_Y)\}, \min\{f_{\overline{Y}}, \phi_F(\mathbf{x}, 0_Y)\}\}, \text{otherwise.} \end{cases}$$

We again define this operation per component. For the two values of semiring $\mathcal{A}$ we proceed as normal and apply the corresponding semiring addition. In contrast, the computation of the Boolean function varies with respect to the order of the two semiring values $\phi_A(\mathbf{x}, 0_Y)$ and $\phi_A(\mathbf{x}, 1_Y)$, and this is where we memorize the frame value of Y that is contained in a solution configuration. For example, if $\phi_A(\mathbf{x}, 1_Y) >_{id} \phi_A(\mathbf{x}, 0_Y)$, then $Y = 1$ must hold in every solution configuration of ϕ. This is guaranteed by the particular Boolean function f_Y that is conjunctively added to the memory $\phi_F(\mathbf{x}, 1_Y)$, which contains the constraints from former eliminations. A similar reasoning applies in the case where $\phi_A(\mathbf{x}, 0_Y) >_{id} \phi_A(\mathbf{x}, 1_Y)$. Finally, if the two values $\phi_A(\mathbf{x}, 0_Y)$ and $\phi_A(\mathbf{x}, 1_Y)$ are equal, both constructions are combined disjunctively, because the solution configurations do not depend on the value Y.

Theorem 2. *A system of memorizing semiring valuations with labeling, combination and marginalization as defined above, satisfies the axioms of a valuation algebra.*

This theorem is proved in [13] and its statement allows to apply local computation, or more concretely the fusion algorithm on memorizing semiring valuations. Thus, for a given factorization $\phi = \phi_1 \otimes \ldots \otimes \phi_n$ of semiring valuations over a totally ordered, strictly monotonic, idempotent semiring $\mathcal{A}$, we embed the factors ϕ_i into a set of memorizing semiring valuations as follows: For $i = 1, \ldots, n$ we define

$$\widehat{\phi_i} : \ \mathbf{x} \in \Omega_{d(\phi_i)} \mapsto (\phi_i(\mathbf{x}), f_1) \, .$$

After this initialization step, we execute the fusion algorithm and eliminate all variables. We obtain

$$\widehat{\phi}^{-s}(\diamond) \ = \ \left(\widehat{\phi_1} \otimes \ldots \otimes \widehat{\phi_n}\right)^{-s}(\diamond).$$

Clearly, the semiring component $\widehat{\phi}_A^{\downarrow\emptyset}(\diamond)$ contains again the maximum value of ϕ over all its configurations, i.e.

$$\widehat{\phi}_A^{-s}(\diamond) \ = \ \phi^{-s}(\diamond) \ = \ \max\{\phi(\mathbf{x}), \ \mathbf{x} \in \Omega_s\}.$$

Let us now focus on the Boolean function $\widehat{\phi}_F^{-s}(\diamond)$ that has been built simultaneously. The following theorem confirms what we have foreshadowed all along, namely that the set of models of this function corresponds exactly to the solution configurations we are looking for.

Theorem 3. *For $\mathbf{x} \in \Omega_s$ and $s = d(\phi)$ we have*

$$\left(\widehat{\phi}_F^{-s}(\diamond)\right)(\mathbf{x}) = 1 \ \textit{ if and only if } \ \phi(\mathbf{x}) = \phi^{-s}(\diamond).$$

Thus, every solution configuration of ϕ evaluates the constructed Boolean function to 1 and is therefore a model. Conversely, every model is also a solution configuration of ϕ. The proof of this result is given in [13].

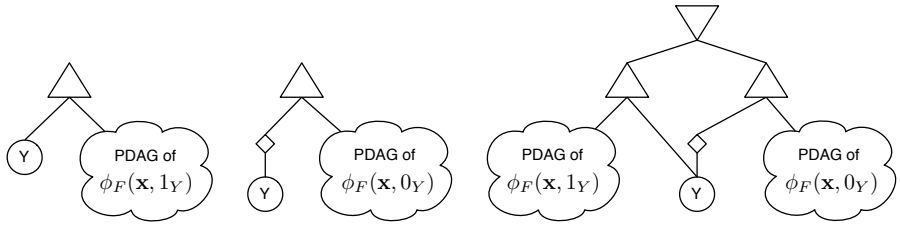

Fig. 1. Variable elimination rules of memorizing semiring valuations as PDAGs.

7.2 Representing Boolean Functions

Naturally, an indispensable requirement for the applicability of this theory is that the models of the final Boolean function can be enumerated efficiently, i.e. in polynomial time. This depends first and foremost on a suitable representation of this particular Boolean function. Here, we propose its representation as *propositional directed acyclic graph* (PDAG) [9].

Definition 3. *A PDAG over the set of propositional variables r is a rooted directed acyclic graph of the following form:*

1. *Leaves are represented by $\bigcirc$ and labeled with $\top$ (true), $\bot$ (false), or $X \in r$.*
2. *Non-leaves are represented by $\triangle$ (logical conjunction), $\triangledown$ (logical disjunction) or $\diamond$ (logical negation).*
3. *$\triangle$- and $\triangledown$-nodes have at least one child and $\diamond$-nodes have exactly one child.*

Figure 1 illustrates the PDAG structures that are created from the three variable elimination rules of memorizing semiring valuations. Combination on the other hand just connects existing PDAGs by a conjunction node to a new PDAG. Thus, because all variables are eliminated, we obtain at the end of the fusion algorithm a single PDAG structure that represents $\widehat{\phi}_F^{-s}(\diamond)$. Some particularities of this graph are summarized in Lemma 2.

Lemma 2. *The PDAG representation of $\widehat{\phi}_F^{-s}(\diamond)$ satisfies the following properties:*

1. *Simple-Negation: Each child of a $\diamond$-node is a leaf.*
2. *Decomposability: The sets of variables that occur in the sub-trees of every $\triangle$-node are disjoint.*
3. *Determinism: The children of every $\triangledown$-node are pairwise logically contradictory., i.e. if α_i and α_j are two children of the same $\triangledown$-node, we have $\alpha_i \wedge \alpha_j \equiv \bot$.*

The first property follows directly from PDAGs 2 and 3 in Figure 1 because these are the only rules that create negation nodes. A variable node is created whenever the corresponding variable is eliminated. Hence, every variable node occurs exactly once in this PDAG which proves Property 2. Finally, we can conclude from PDAG 3 in Figure 1 that in every model of the disjunction node's left child, $Y = 1$ must hold. Similarly,

$Y = 0$ must hold in the right child and therefore, the two model sets are disjoint. This is the statement of Property 3. A PDAG that satisfies all three properties of Lemma 2 is called *cdn-PDAG* [9] or *d-DNNF* [8] and it turns out that model enumeration can indeed be done in polynomial time upon such PDAGs. A corresponding algorithm for this task is given in [17].

It is worth mentioning that during the fusion process, d-DNNFs are built from connecting existing d-DNNFs by either a conjunction or a disjunction node. However, we know from [8] that the d-DNNF language is not closed under these operations. This means concretely that it is in general not possible to reconstruct a d-DNNF structure from the conjunction or disjunction of two d-DNNFs in polynomial time. Fortunately, this does not hold for the case at hand. Since these constructions are performed as valuation algebra operations, we directly obtain the cdn-properties whenever we join two existing d-DNNFs by the rules specified for memorizing semiring valuations. This features our approach in contrast to similar techniques where the needed graph properties have to be reestablished [7].

7.3 Further Efficient Queries

Besides efficient model enumeration, d-DNNFs allow many other *queries* to be performed efficiently, and this fact constitutes the worth of our method compared with classical approaches that essentially content themselves with model identification. [9] give a comprehensive listing of such queries whose complete disquisition would go beyond the scope of this article. Nevertheless, we will suggest some of these possibilities in order to point out the potential of our approach in view of relevant applications in diagnosis for example.

- *Counter-Model Enumeration:* The d-DNNF constructed by the fusion algorithm allows also to enumerate all configurations of ϕ that are not solution configurations, i.e. all configurations $\mathbf{x} \in \Omega_s$ with $\phi(\mathbf{x}) <_{id} \phi^{-s}(\diamond)$.
- *Model Counting:* Counting the number of solution configurations can also be done efficiently.
- *Validity:* This answers the query if $\widehat{\phi}_F^{-s}(\diamond) \equiv f_1$, i.e. if all configurations of ϕ adopt the same value.
- *Probability Computation:* If we assume independent marginal probabilities $p(X)$ for all variables $X \in s$, we can efficiently evaluate the probability of the Boolean function $p(\widehat{\phi}_F^{-s}(\diamond))$.
- *Probabilistic Equivalence Test*: If two different factorizations over the same set of variables are given, d-DNNFs allow to test probabilistically if the two joint valuations ϕ_1 and ϕ_2 adopt the same solution configurations, i.e. for all $\mathbf{x} \in \Omega_s$, $\phi_1(\mathbf{x}) = \phi^{-s}(\diamond)$ if and only if $\phi_2(\mathbf{x}) = \phi^{-s}(\diamond)$.

Although this is only a small selection from the extensive list of queries that can be performed efficiently on d-DNNFs and therefore on the fusion algorithm's result, we come to the conclusion that the representation of solution configurations by a Boolean functions offers a lot more possibilities for further evaluations than traditional approaches. Furthermore, because this method is also based on local computation, no loss of efficiency occurs.

8 Conclusion

With the (extended) fusion algorithm, we are in possession of an efficient tool to compute solution configurations of optimization problems that are given as factorizations of semiring valuations. However, the requirements of diagnosis are rarely limited to the identification of solution configurations. Moreover, we often want to perform some further evaluations of these solution configurations without their explicit enumeration. This paper proposes therefore to compile the solution configuration set into a Boolean function and by use of current knowledge compilation techniques, we obtain a very compact, graphical representation of the solution configuration set that allows to carry out efficiently a large collection of new queries, including the enumeration of solution configurations. Additionally, this compilation process does not forfeit efficiency because it is based on the same local computation scheme. This features our new approach which is furthermore a delightful combination of results from different fields of AI research.

References

1. Shenoy, P.P., Shafer, G.: Axioms for probability and belief-function proagation. In Shachter, R.D., Levitt, T.S., Kanal, L.N., Lemmer, J.F., (eds.) Uncertainty in Artificial Intelligence 4. Machine intelligence and pattern recognition 9, 169–198 (1990)
2. Shafer, G.: An axiomatic study of computation in hypertrees. Working Paper 232, School of Business, University of Kansas (1991)
3. Kohlas, J.: Information Algebras: Generic Structures for Inference. Springer, Heidelberg (2003)
4. Shenoy, P.: Axioms for dynamic programming. In Gammerman, A., ed.: Computational Learning and Probabilistic Reasoning. Wiley, Chichester, UK (1996) 259–275
5. Bertele, U., Brioschi, F.: Nonserial Dynamic Programming. Academic Press (1972)
6. Wilson, N.: Decision diagrams for the computation of semiring valuations. In IJCAI'05, 19th International Joint Conference on Artificial Intelligence (2005) 331–336
7. Mateescu, R., Dechter, R.: Compiling constraint networks into and/or multi-valued decision diagrams (aomdds). In: CP. (2006) 329–343
8. Darwiche, A., Marquis, P.: A knowledge compilation map. J. Artif. Intell. Res. (JAIR) **17** (2002) 229–264
9. Wachter, M., Haenni, R.: Propositional DAGs: a new graph-based language for representing Boolean functions. In KR'06, 10th International Conference on Principles of Knowledge Representation and Reasoning (2006) 277–285
10. Shenoy, P.: Valuation-based systems: A framework for managing uncertainty in expert systems. In: Fuzzy Logic for the Management of Uncertainty (1992) 83–104
11. Schneuwly, C., Pouly, M., Kohlas, J.: Local computation in covering join trees. Technical Report 04-16, University of Fribourg (2004)
12. Schweizer, B., Sklar, A.: Statistical metric spaces. Pacific J. Math. **10** (1960) 313–334
13. Pouly, M., Kohlas, J.: Local computation & dynamic programming. Technical Report 07-02, University of Fribourg (2007)
14. Kohlas, J., Wilson, N.: Exact and approximate local computation in semiring induced valuation algebras. Technical Report 06-06, University of Fribourg (2006)
15. Bistarelli, S., Montanari, U., Rossi, F., Schiex, T., Verfaillie, G., Fargier, H.: Semiring-based CSPs and valued CSPs: Frameworks, properties, comparison. Constraints **4** (1999) 199–240

16. Zadeh, L.: Fuzzy sets as a basis for a theory of possibility. Fuzzy Sets and Systems **1** (1978)
17. Darwiche, A.: Decomposable negation normal form. J. ACM **48** (2001) 608–647