Generalizing Semiring induced Valuation Algebras TR-IIIA-2013-03

Jordi Roca Lacostena and Jesús Cerquides
Artificial Intelligence Research Institute, IIIA-CSIC
Spanish National Research Council, CSIC.
Campus Universitat Autònoma de Barcelona, Bellaterra, Spain
{jroca, cerquide}@iiia.csic.es

December 13, 2013

Abstract

Valuation algebras, are the algebraic structure associated with the treatment and management of information packages. These kind of structures are frequently used on statistics and strongly linked to artificial intelligence.

Along this dissertation we will generalize the concept of semiring induced valuation algebra so that set-based valuation algebras and some algorithms like Action-GDL [12] or the Graph Coalition Structure Generation algorithm described in [13], can be fit into a new single wider unified framework.

This document is based on the Bachelor Degree Dissertation written by Jordi Roca Lacostena under the supervision of Jesús Cerquides Bueno and Joaquim Roé i Vellvé .

1 Introduction

Description and objectives of the project

Valuation algebras, are the algebraic structure associated with the treatment and management of information packages.

These kind of structures are frequently used on statistics and strongly linked to artificial intelligence. Furthermore, there are several articles [5, 9, 10] refering valuations algebras as an algebraic structure used in order to solve Distributed Constraint Optimization Problems (DCOPs).

A DCOP [5, 9, 12] is a formalism that captures the rewards and costs of local interactions in a multiagent system where each agent chooses a set of individual actions. Thereby, it is a framework which can model a large number of coordination, scheduling and tasks allocation problems in multiagent systems [12].

In order to solve DCOPs researchers and engineers have developed lots of algorithms relying on valuation algebras. One of these algorithms is Action-GDL [12], which was also presented as finding solutions without distribute in a semiring induced valuation algebra [4, 8].

In spite of DCOPs being that important, there are still some problems that cannot be efficiently mapped there. Hence, Pouly created in [6] a wider framework called set-based valuation algebras which include the valuation algebras used in DCOPs.

Nonetheless, in [13] an algorithm similar to Action-GDL is proposed in order to solve the coalition structure generation problem (CSGP). This problem is equivalent to the complete set partitioning problem, one of the fundamental problems in combinatorial optimisation. Nevertheless, there is no known way to map this problem to a set-based valuation algebra.

Hence, the objective of this dissertation is to generalize the concept of semiring induced valuation algebra so that Action-GDL, the algorithm described in [13] and set-based valuation algebras can be fit into a new single wider unified framework.

Contents

This dissertation is divided into six chapters, being the introduction the first one.

The second chapter refreshes some properties of lattices and defines the concept of valuation algebra in the most general way.

The third one, which is based on Pouly and Kohlas book *Generic inference* [8], introduces covering join trees and describes the collect algorithm, bringing a new proof in order to show that it works.

On the fourth chapter we define valuation algebras over semirings using a partition lattices as background.

Finally the fifth chapter shows that the Action-GDL algorithm holds on the valuation algebras defined on chapter 4.

The last chapter include the conclusions of the dissertation as long as some ideas for future work on this subject

For completeness and ease of reading, the appendix contains some basic and important information about ordered sets and lattices.

2 Valuation Algebras on Modular Lattices

Information treatment and manipulation on huge domains is usually infeasible for informatic systems. The way of solving this issue is breaking the entire information package on small data packages over smaller domains in such a way that one can obtain the original information by combining the small packages.

The main advantage of this procedure, known as factorization, is the fact that the information packages created are more operable than the original package due to the difference between the sizes of their respective domains.

Valuation algebras are the algebraic structure associated to the treatment of information packages, thus they are strongly linked with information management.

In this chapter we will introduce the notion of valuation algebra as long as some other concepts which will turn out important for the objectives of our work.

2.1 Valuation algebras

Valuation Algebras are built on top of lattices. Therefore we need to know what is a lattice. Informally, we say that a lattice is a partially ordered set in which any two elements have a supremum and an infimum. Further information about lattices can be found in the appendix.

Definition 2.1.
Let D be lattice. A set Φ is called a *set of valuations* with *domain* in D if we can define three operations on Φ and D:

- *Labeling*: $\Phi \to D$; $\phi \mapsto d(\phi)$
- *Combination*: $\Phi \times \Phi \to \Phi$; $(\phi, \psi) \mapsto \phi \otimes \psi$
- *Projection*: $\Phi \times D \to \Phi$; $(\phi, a) \mapsto \phi^{\downarrow a}$ for $a \leq d(\phi)$

An element $\phi \in \Phi$ is called a *valuation*

Example 1. *Valuations over discrete variables*

In this example we are going to see the set of valuations that is used in order to define and solve a DCOP.

Consider $U = \{x_1, \dots, x_n\}$ a finite set of discrete variables.

For any variable $x_i \in U$ we denote the set of its possible values by Ω_{x_i}, and we will call it the frame of x_i. Moreover, given a subset $s \subseteq U$ we define its set of configurations or its frame as

$$\Omega_s = \prod_{x_i \in s} \Omega_{x_i}$$

Hence, we will take the powerset lattice of U, $\mathcal{P}(U)$, as the domain lattice of the valuation algebra defined over discrete variables, whereas a valuation with domain $d \in \mathcal{P}(U)$ will be any function $\phi : \Omega_d \to \mathbb{R}$ from the possible configurations of the variables in d to the real numbers set.

Before defining the operations it is necessary to remark that for any $d \in D$ and any $s \subset d$ it is possible to break the configurations of d as the configurations of s and the configurations of $s' = \{x_i \in d : x_i \notin s\}$, i.e. all $x \in \Omega_d$ can be written as (a, b) where $a \in \Omega_s$ and $b \in \Omega_{s'}$. We will denote a as $x^{\downarrow s}$ and b as $x^{\downarrow s'}$

Now we are ready to define all the operations in the valuation algebra defined over discrete variables.

Recall that labeling just maps every valuation to its domain, thus we only have to define the combination and projection:

- *Combination*: Given $\phi, \psi \in \Phi$ with respective domains s, t we define:

$$\begin{array}{rcl} (\phi \otimes \psi) : \Omega_{s\cup t} & \longrightarrow & \mathbb{R} \\ x & \longmapsto & \phi(x^{\downarrow s}) \times \psi(x^{\downarrow t}) \end{array}$$

- *Projection*:Given $\phi \in \Phi$ with domain s , for all $t \subseteq s$ we define:

$$\begin{array}{rcl} \phi^{\downarrow t} : \Omega_t & \longrightarrow & \mathbb{R} \\ x & \longmapsto & \sum_{y \in \Omega_{t'}} \phi(x, y) \end{array}$$

For instance, consider $U = \{x, y, z\}$ were x, y, z are binary variables and the valuations $\phi : \Omega_{\{x,y\}} \to \mathbb{R}$ and $\psi : \Omega_{\{y,z\}} \to \mathbb{R}$ defined as it follows:

ϕ	$y=0$	$y=1$
$x=0$	-2	5
$x=1$	1	-3

ψ	$z=0$	$z=1$
$y=0$	0	2
$y=1$	-1	1

It is clear that $d(\phi) = \{x, y\}$ and $d(\psi) = \{y, z\}$, thus we are going to calculate $(\phi \otimes \psi)(\alpha)$ for all $\alpha \in \Omega_{\{x,y,x\}}$:

α	$\alpha^{\downarrow\{x,y\}}$	$\alpha^{\downarrow\{y,z\}}$	$\phi(\alpha^{\downarrow\{x,y\}})$	$\psi(\alpha^{\downarrow\{y,z\}})$	$(\phi \otimes \psi)(\alpha)$
000	00	00	-2	0	0
001	00	01	-2	2	-4
010	01	10	5	-1	-5
011	01	11	5	1	5
100	10	00	1	0	0
101	10	01	1	2	2
110	11	10	-3	-1	3
111	11	11	-3	1	-3

Finally, we will take $s = \{y, z\}$ and we will calculate $(\phi \otimes \psi)^{\downarrow s}(\beta)$ for all $\beta \in \Omega_s$.

Notice that $\Omega_{s'} = \Omega_{\{x\}} = \{0, 1\}$, thus we obtain $(\phi \otimes \psi)^{\downarrow s}(\beta) = (\phi \otimes \psi)(0, \beta) + (\phi \otimes \psi)(1, \beta)$.
Hence:

β	$(\phi \otimes \psi)(0, \beta)$	$(\phi \otimes \psi)(1, \beta)$	$(\phi \otimes \psi)^{\downarrow s}(\beta)$
00	0	0	0
01	-4	2	-2
10	-5	3	-2
11	5	-3	2

Definition 2.2.
Let D be lattice and Φ be a set of valuations with domain in D.
We say that the tuple $\langle \Phi, D \rangle$ is a *valuation algebra* on D if the following axioms are satisfied by the three operations defined above (labeling, combination and projection): .

A1 *Commutative Semigroup*: Φ is associative and commutative under $\otimes$.

A2 *Labeling*: For $\phi, \psi \in \Phi$,

$$d(\phi \otimes \psi) = d(\phi) \vee d(\psi)$$

A3 *Projection*: For $\phi \in \Phi$, $a \in D$ such that $a \leq d(\phi)$,

$$d(\phi^{\downarrow a}) = a$$

A4 *Transitivity*: For $\phi \in \Phi$, $a, b \in D$ such that $a \leq b \leq d(\phi)$,

$$(\phi^{\downarrow b})^{\downarrow a} = \phi^{\downarrow a}$$

A5 *Combination*: For $\phi, \psi \in \Phi$, $a \in D$ such that $d(\phi) \leq a \leq d(\phi) \vee d(\psi)$,

$$(\phi \otimes \psi)^{\downarrow a} = \phi \otimes (\psi^{\downarrow a \wedge d(\psi)})$$

A6 *Domain*: For $\phi \in \Phi$,

$$\phi^{\downarrow d(\phi)} = \phi$$

It can easily be shown that the valuations and the operations defined on the example above satisfy all the axioms of valuation algebras.

Next, we are going to prove a result that we will use in the last part of the dissertation. We want to do it now because later on we will work with a specific kind of valuation algebras and we want to show that this result holds for any valuation algebra

Proposition 2.3.
Let $\langle \Phi, D \rangle$ be a valuation algebra. Let $\phi, \psi \in \Phi$. Then:

$$(\phi \otimes \psi)^{\downarrow d(\phi) \wedge d(\psi)} = \phi^{\downarrow d(\phi) \wedge d(\psi)} \otimes \psi^{\downarrow d(\phi) \wedge d(\psi)}$$

Proof.
Using transitivity and combination axioms it holds:

$$(\phi \otimes \psi)^{\downarrow d(\phi) \wedge d(\psi)} = ((\phi \otimes \psi)^{\downarrow d(\phi)})^{\downarrow d(\phi) \wedge d(\psi)} = (\phi \otimes \psi^{\downarrow d(\phi) \wedge d(\psi)})^{\downarrow d(\phi) \wedge d(\psi)} = \phi^{\downarrow d(\phi) \wedge d(\psi)} \otimes \psi^{\downarrow d(\phi) \wedge d(\psi)}$$

□

2.2 Valuation algebras can only be defined on modular lattices

In [8], Pouly and Kohlas say that valuation algebras can be defined over any kind of lattice. We will show that modular lattices are the most general lattices which admit a valuation algebra structure if we want to be able to define a valuation over any element of the domain D.

First of all, we will introduce the concept of modularity, which can be seen as a weaker notion of the distributive property. Further information about modular lattices can be found in [11].

Definition 2.4.
A lattice $L = (\mathcal{L}; \vee, \wedge)$ is said to be *modular* if for all $x, w, y \in \mathcal{L}$ such that $x \leq y$ we have:

$$x \vee (w \wedge y) = (x \vee w) \wedge y$$

Proposition 2.5.
Let L be a lattice. Then L is modular if and only if

$$x \leq z \leq x \vee y \Rightarrow x \vee (y \wedge z) = z \textit{ for all } x, y, z \in L$$

Proof.
Suppose that L is modular, and $x, y, z \in L$ such that $x \leq z \leq x \vee y$
Using $x \leq z$ and the definition of modular we obtain $x \vee (y \wedge z) = (x \vee y) \wedge z$.
As long as $z \leq x \vee y$ we have $(x \vee y) \wedge z = z$.

Now suppose that $x \vee (y \wedge z) = z$ for all $x, y, z \in L$ such that $x \leq z \leq x \vee y$.
Given $x, y, w \in L$ such that $x \leq y$, we set $z = (x \vee w) \wedge y$.

Due to $\vee$ definition we know that $z \leq x \vee w$. Equivalently we obtain $x \leq y \leq z$ using $\wedge$ definition, so in particular $x \leq z \leq x \vee w$

Using the hypothesis above we obtain $x \vee (w \wedge z) = z = (x \vee w) \wedge y$
Furthermore, since $w = w \wedge (x \vee w)$,we obtain $x \vee (w \wedge y) = x \vee ((w \wedge (x \vee w)) \wedge y) = x \vee (w \wedge ((x \vee w) \wedge y)) = x \vee (w \wedge z) = z = (x \vee w) \wedge y$

□

Theorem 2.6.
Let $\langle \Phi, D \rangle$ be a valuation algebra such that for all $d \in D$, there is a $\phi \in \Phi$ such that $d(\Phi) = d$ Then:

D is a modular lattice

Proof.
Given $x, y, z \in D$ such that $x \leq z \leq x \vee y$ we can take $\phi, \psi \in \Phi$ such that $d(\phi) = x$ and $d(\psi) = y$ due to hypothesis.

Using the combination axiom (A5) we obtain $(\phi \otimes \psi)^{\downarrow a} = \phi \otimes (\psi^{\downarrow a \wedge d(\psi)})$. Therefore, if we use the labeling and projection axioms ((A2) and (A3)), we obtain $z = d((\phi \otimes \psi)^{\downarrow z}) = d(\phi \otimes (\psi^{\downarrow z \wedge d(\psi)})) = x \vee (z \wedge y)$.

As long as we can do it for every $x, y \in D$ we obtain that D is modular by proposition 2.5.

□

3 Finding marginals on a factorized valuation

As we previously said, valuation algebras are used to break big information packages on smaller and more operable data packages. Nonetheless, we want to do it in such a way that we can obtain the initial information by combining the small data blocks.

Thereby, along this dissertation we will study which information related to ϕ can be obtained from the elements of a given factorization factorization.

In particular, we will suppose that we are given a valuation ϕ by means of a set of valuations $\phi_1, \ldots, \phi_n$ such that $\phi = \phi_1 \otimes \cdots \otimes \phi_n$. Moreover, as long as we want the data packages to be smaller we require that the domains of ϕ_i are far smaller than the domain of ϕ.

Since ϕ itself is intractable (its domain is supposed to be too big), we will be interested in determining the projection $\phi^{\downarrow s}$ of ϕ to a smaller domain $s \leq d(\phi)$.

This is known as the *projection* (or *marginal*) of ϕ to s and it gives us information related to s which can only be obtained from ϕ. Therefore it would be pretty interesting to find how to obtain $\phi^{\downarrow s}$ for some $s \in D$ using a factorization of ϕ.

The aim of this chapter is to prove that there is an algorithm, called collect algorithm, which can be used for finding some marginals of ϕ from a given factorization.

The collect algorithm [3, 8] uses a covering join tree as its main data structure. Thus we start introducing covering join trees in section 3.1 and continue with the description of the algorithm in the section 3.2. Finally we will give a new proof about the collect algorithm which is valid on any valuation algebra on a modular lattice.

3.1 Covering Join Trees

The collect algorithm is based on message-passing between the elements of the factorization. It is important to know when the messages are sent and who the sender and the receiver are. In order to understand why the collect algorithm works we will introduce the concept of covering join tree:

Definition 3.1.
A *labeled tree* is a tuple $\mathcal{T} = (V, E, \lambda, D)$ such that:

- (V, E) is a tree, in particular $|V| = |E| + 1$
- D is a set
- $\lambda : \ V \longrightarrow D$

We say that a labeled tree is a *lattice-labeled tree* if D is a lattice.

Thinking about the example 1, one can deduce that given two valuations such that their domains share a variable x they must be able to talk about information related to this variable.

Nonetheless, when these domains are placed on a labeled tree and they are not neighbours, it turns necessary to send the information through the nodes in the path between them. In order to guarantee that no information is lost during this process we must demand the labeled tree to satisfy the property below.

Definition 3.2.
A lattice-labeled tree $\mathcal{T} = (V, E, \lambda, D)$ is said to satisfy the *running intersection property* if for any $i, j \in V$ it holds that $\lambda(i) \wedge \lambda(j) \leq \lambda(k)$ for all nodes k on the path between i and j.

A lattice-labeled tree that satisfies the running intersection property is called *join tree.*

Definition 3.3.
Let (Φ, D) be a valuation algebra and $\phi \in \Phi$ such that $\phi = \phi_1 \otimes \cdots \otimes \phi_n$.

A join tree $\mathcal{T} = (V, E, \lambda, D)$ is called *covering join tree* for the factorization $\phi_1 \otimes \cdots \otimes \phi_n$ if $|V| = n$ and for all ϕ_i there is a node $v_j \in V$ such that $d(\phi_i) \leq \lambda(j)$.

Notice that this definition of covering join tree is different to other definitions given in [8, 10] because we enforce $|V| = n$. Nevertheless one can easily deduce that it is possible to obtain a new factorization, based on the original one, with the adequate number factors for fitting the number of tree nodes. On the other side, taking $|V| = n$ allows us to link any valuation ϕ_i of the factorization with a single tree node.

Taking this into account, we are going to enumerate the nodes in such a way that it will be easier to work. After doing that we will be able to introduce the concept of node separator which will also become really helpful.

Given $\mathcal{T} = (V, E, \lambda, D)$ a covering join tree for the factorization $\phi_1 \otimes \cdots \otimes \phi_n$ we will number the nodes $v_j \in V$ in the following way:

- The root node, v_k, will be numbered as $\pi(k) = |V| = r$
- Given two nodes $v_i, v_j \in V \setminus \{v_k\}$ if v_j is in the path from node v_i to the root node, we will write $\pi(v_i) = a < b = \pi(v_j)$.

From now on we are going to assume that all the covering join trees are numbered in this way, and thus we will refer to $a = \pi(v_i)$ instead of v_j. Moreover, we will suppose that $d(\phi_i) \leq \lambda(i)$, thus we will link each valuation ϕ_i with the node numbered by i (see definition 3.6).

Before defining the separators we must remember some properties about trees.

Definition 3.4.
Let $\mathcal{T} = (V, E)$ be a tree and let r be the root node.

For each $i \neq r$ we define the *parent* of the node i, denoted by $pa(i)$, as the first node after i in the path between i and r.

Furthermore, for each node i we define the set of its *children* as $ch(i) = \{j \in V : pa(j) = i\}$.

Similary, for each node i we define the set of its *brothers* as $br(i) = \{j \in ch(pa(i)) : j \neq i\}$

Definition 3.5.
Given $\mathcal{T} = (V, E, \lambda, D)$ a covering join tree , the *separator* $sep(i)$ of a node $i < |V|$ is defined by

$$sep(i) = \lambda(i) \wedge \lambda(pa(i))$$

Although we are ready to formulate the collect algorithm and to prove that it can be used for finding some marginals using the notion of covering join tree we have just defined, we want to improve the nodes' labels. In other words, we want the domains linked with the tree nodes and the factorization valuations to be as small as possible.

For instance, it is clear that for any given factorization $\phi = \phi_1 \otimes \cdots \otimes \phi_n$ we can take any tree with n nodes and link each one of these nodes with the whole domain $d(\phi)$, obtaining a covering join tree. Nevertheless, this procedure has the problem that the collect algorithm manipulates information related to the domains linked with the covering join tree, so we gain nothing by factorizing the valuation ϕ, because we will have to use information packages domains as huge as the original one.

Definition 3.6.
Let $\mathcal{T} = (V, E, \lambda, D)$ a covering join tree for a given factorization $\phi = \phi_1 \otimes \cdots \otimes \phi_n$.

For each node i we set

- $\alpha_i = d(\phi_i) \vee \bigvee_{j \in ch(i)} \alpha_j$ for $i = 1, \dots, r$
- $\beta_i = d(\phi_i) \vee \bigvee_{\substack{j,k \in ch(i) \\ j \neq k}} (\alpha_j \wedge \alpha_k)$ for $i = 1, \dots, r$
- $\lambda'(i) = \beta_i \vee (\lambda(pa(i)) \wedge \alpha(i))$ for $i = r, \dots, 1$

Then we define the *optimized covering join tree* over $\mathcal{T}$ as $\mathcal{T}' = (V, E, \lambda', D)$

***Remark*:** Notice that α_i and β_i are well defined because $j \in ch(i)$ implies $j < i$. Similarly $\lambda'(i)$ is well defined due to the fact that $i < pa(i)$.

Before getting introduced to the collect algorithm we will prove that any optimized covering join tree is itself a covering join tree. We will use the proposition below in order to obtain that result.

Proposition 3.7.
Let $\mathcal{T} = (V, E, \lambda, D)$ be an optimized covering join tree for a given factorization $\phi = \phi_1 \otimes \cdots \otimes \phi_n$. Then:

$$\lambda(i) \leq \alpha_i, \quad \textit{for all } i \in V$$

Proof.
It is immediate to see that $\beta_i \leq \alpha_i$.
Then:

$$\lambda(i) = \beta_i \vee (\lambda(pa(i)) \wedge \alpha_i) \leq \alpha_i \vee (\lambda(pa(i)) \wedge \alpha_i) = \alpha_i$$

□

Theorem 3.8.
Let $\mathcal{T} = (V, E, \lambda, D)$ be an optimized covering join tree for a given factorization $\phi = \phi_1 \otimes \cdots \otimes \phi_n$. Then:

$$\mathcal{T} \textit{ is a covering join tree}$$

Proof.
First, one must notice that for all $\phi(i)$ it is satisfied $d(\phi(i)) \subseteq \beta_i \subseteq \lambda'(i)$. Therefore we only have to prove that the running intersection property is satisfied.

Set $p_1, p_2, \ldots, p_m$ the unique path between two given nodes $p_1, p_m \in V$. We want to see that

$$p_1 \wedge p_m \leq p_i, \quad \text{for } 1 < i < m$$

Notice that if $m \leq 2$ it is immediate to prove it. Therefore we will suppose $m \geq 3$.

As long as $\mathcal{T}$ is a tree, the path can be seen as the composition of two different paths. The first part grows up from p_1 path to some node p_i such that $p_i = pa(\overset{i-1}{\ldots}(pa(i))_{\ldots}))$. On the other side, the second part descends from p_i to p_m, satisfying $p_j = pa(\overset{m-j}{\ldots}(pa(m))_{\ldots}))$. Notice that if $p_1 \neq p_m$ at most one of these subpaths may be empty.

Then there are three possible configurations for the paths:

1. No subpath is empty, i.e. $1 \nleq i \nleq m$

   In this case, by proposition 3.7 we have that $\lambda(p_1) \leq \alpha_{p_1} \leq \alpha(p_{i-1})$ and $\lambda(p_m) \leq \alpha_{p_m} \leq \alpha(p_{i-1})$. As long as $p_{m-1}, p_{m+1} \in ch(p_i)$ we obtain:

$$\lambda(p_1) \wedge \lambda(p_m) \leq \alpha(p_{i-1}) \wedge \alpha(p_{i-1}) \leq \bigvee_{\substack{j,k \in ch(i) \\ j \neq k}} (\alpha_j \wedge \alpha_k) \leq \beta_i \leq \lambda(i)$$

2. The descendant path is empty, so $i = m$

   In this case, we use proposition 3.7 in order to obtain$\lambda(p_1) \wedge \lambda(p_m) \leq \lambda(p_1) \leq \alpha_1 \leq \alpha_{m-1}$
   On the other side we know that $\lambda(p_1) \wedge \lambda(p_m) \leq \lambda(p_m) = \lambda(pa(p_{m-1}))$
   In conclusion we obtain $\lambda(p_1) \wedge \lambda(p_m) \leq \alpha_{m-1} \wedge \lambda(pa(p_{m-1})) \leq \lambda(p_{m-1})$. Moreover:

$$\lambda(p_1) \wedge \lambda(p_m) \leq \lambda(p_1) \wedge \lambda(p_m) \wedge \lambda(p_{m-1}) \leq \lambda(p_1) \wedge \lambda(p_{m-1})$$

   Then we have that $\lambda(p_1) \wedge \lambda(p_m) \leq \lambda(p_1) \wedge \lambda(p_{m-1}) \leq \cdots \leq \lambda(p_1) \wedge \lambda(p_2)$
   Therefore $\lambda(p_1) \wedge \lambda(p_m) \leq \lambda(p_1) \wedge \lambda(p_i) \leq \lambda(p_i), \quad \text{for } 1 < i < m$

3. The ascendant path is empty so $i = 1$

   In this case we will consider the path from p_m to p_1 and use the proof for empty descendant path, as long as $p_m, p_{m-1}, \ldots, p_1$ would be an ascendant path.

□

3.2 Collect Algorithm

Given (Φ, D) be a valuation algebra and $\phi \in \Phi$ such that $\phi = \phi_1 \otimes \cdots \otimes \phi_n$, the aim of the collect algorithm is compute $\phi^{\downarrow d(\phi_i)}$ for some fixed i by only using $\phi_1, \ldots, \phi_n$.

In order to do that we suppose that we are given an optimized covering join tree for that factorization. Notice that such a tree always exists, due to definition 3.6.

The algorithm is based on sending messages up the tree, through the edges of the covering join tree, until the root node is reached. The message sent from each node summarizes the information in the corresponding subtree which is relevant to its parent. As we will see the running intersection property guarantees that no information is lost.

The collect algorithm may be described by the following rules:

R1 Each node sends a message to its parent once it has received all messages from its children.

R2 When a node is ready to send a message, it computes the message by projecting its current content to the separator.

R3 When a node receives a message it updates its current content by combining it with the incoming message

As one can deduce from these rules, the content of the nodes may change during the algorithm's run. In order to simplify the lecture we will introduce the following notation:

- $\phi_j^{(1)} = \phi_j$ will denote the initial content of node j
- $\phi_j^{(i)}$ will denote the content of node j before step i of the collect algorithm.
- $y_i = \bigvee_{j=i}^{r} \lambda(i)$

Due to the way of numbering the nodes of the covering join tree we can suppose that during the i-th step of the algorithm the node i sends a message to its parent. This fact allows us to define the message-passing as it follows:

- At step i, the node i computes the message
$$\mu_{i \to pa(i)} = (\phi_i^{(i)})^{\downarrow sep(i)}$$
- The receiving node $pa(i)$ combines the message with its node content:
$$\phi_{pa(i)}^{(i+1)} = \phi_{pa(i)}^{(i)} \otimes \mu_{i \to pa(i)}$$
- The content of all other nodes remains unchanged:
$$\phi_j^{(i+1)} = \phi_j^{(i)}, \quad \text{for all } j \neq pa(i)$$

One can easily deduce that the algorithm has $r - 1 = |V| - 1$ steps as long as $|E| = |V| - 1$. Thus the content of any node at the end of the collect algorithm will be denoted by $\phi_j^{(r)}$. Moreover, looking at the structure of the collect algorithm it is immediate to prove that:

$$\phi_j^{(r)} = \phi_j \otimes \bigotimes_{i \in ch(j)} \mu_{i \to j}$$

Proposition 3.9.
Let $\mathcal{T} = (V, E, \lambda, D)$ be an optimized covering join tree for a given factorization $\phi = \phi_1 \otimes \cdots \otimes \phi_n$. Then for $i = 1, \ldots, r$ it holds:

$$d(\phi_i^{(j)}) = \lambda(i), \quad \text{for } j \geq i$$

Proof.
Due to how the collect algorithm works, we obtain that if $j \geq i$ it holds that

$$d(\phi_i^{(j)}) = d(\phi_i^{(r)})\phi_j = d(\phi_i \otimes \bigotimes_{i \in ch(j)} \mu_{i \to j})$$

Therefore we can write $d(\phi_i^{(j)})$ as it follows:

$$d(\phi_i^{(j)}) = d(\phi_i^{(r)}) = d(\phi_i \otimes \bigotimes_{k \in ch(i)} \mu_{k \to i}) = d(\phi_i) \vee \bigvee_{k \in ch(i)} \mu_{k \to i}) = d(\phi_i) \vee \bigvee_{k \in ch(i)} sep(k) =$$
$$= d(\phi_i) \vee \bigvee_{k \in ch(i)} (\lambda(k) \wedge \lambda(i))$$

Moreover $d(\phi_i^{(j)}) = (\ d(\phi_i) \vee \bigvee_{k \in ch(i)} \lambda(k)\ ) \wedge \lambda(i)$ due to modularity (because $d(\phi_i) \leq \lambda(i)$).
Therefore $d(\phi_i^{(j)}) \leq \lambda(i)$.

On the other hand by λ definition we know that $\lambda(k) = \beta_k \vee (\lambda(pa(k)) \wedge \alpha_k) \geq \lambda(pa(k)) \wedge \alpha_k$. Then, as long as $pa(k) = i$ for all $k \in ch(i)$ we obtain:

$$d(\phi_i) \vee \bigvee_{k \in ch(i)} (\lambda(k) \wedge \lambda(i)) \geq\ d(\phi_i) \vee \bigvee_{k \in ch(i)} ((\lambda(i) \wedge \alpha(k) \wedge \lambda(i)) =\ d(\phi_i) \vee \bigvee_{k \in ch(i)} (\alpha(k) \wedge \lambda(i))$$

Finally, using again the modularity we obtain:

$$d(\phi_i^{(j)}) \geq\ d(\phi_i) \vee \bigvee_{k \in ch(i)} (\lambda(i) \wedge \alpha(k)) = (\ d(\phi_i) \vee \bigvee_{k \in ch(i)} \alpha(k)\ ) \wedge \lambda(i) = \alpha(i) \wedge \lambda(i) = \lambda(i)$$

□

The following theorem proves that some marginals can be obtained from a given factorization. A similar result can be found in the third chapter of [8], but it only holds for valuation algebras on distributive lattices (like powersets in example 1), whereas the following theorem provides the result for valuation algebras on modular lattices.

Theorem 3.10.
Let $\mathcal{T} = (V, E, \lambda, D)$ be an optimized covering join tree for a given factorization $\phi = \phi_1 \otimes \cdots \otimes \phi_n$. Then for $i = 1, \ldots, r$ it holds:

$$\bigotimes_{j=i}^{r} \phi_j^{(i)} = \phi^{\downarrow y_i},$$

In particular, for $i = r$ we obtain that at the end of the collect algorithm, the root node r contains the marginal of ϕ relative to $\lambda(r)$.

$$\phi_r^{(r)} = \phi^{\downarrow \lambda(r)}$$

Proof.
For $i = 1$ notice that $y(1) = d(\phi)$ because $\mathcal{T}$ covers a factorization $\phi_1 \otimes \cdots \otimes \phi_r$ of ϕ. Therefore we obtain:

$$\phi^{\downarrow \lambda(1)} = \phi^{\downarrow d(\phi)} = \phi = \phi_1 \otimes \dots \phi_n = \phi_1^{(1)} \otimes \dots \phi_n^{(1)} = \bigotimes_{j=1}^{n} \phi_j^{(1)}$$

Now we will suppose that the equality holds for i and we will prove it for $i + 1$:

We know that $y_{i+1} \leq y_i$ due to how we defined y_j, from that we obtain:

$$\phi^{\downarrow y_{i+1}} = (\phi^{\downarrow y_i})^{\downarrow y_{i+1}} = (\phi_i^{(i)} \otimes \phi_{i+1}^{(i)} \otimes \cdots \otimes \phi_r^{(i)})^{\downarrow y_{i+1}} = ((\phi_i^{(i)}) \otimes (\phi_{i+1}^{(i)} \otimes \cdots \otimes \phi_r^{(i)}))^{\downarrow y_{i+1}}$$

Notice that as long as $d(\phi_j^{(i)}) \leq \lambda(j)$ for all i we obtain $d(\phi_{i+1}^{(i)} \otimes \cdots \otimes \phi_r^{(i)}) \leq y_{i+1}$

In particular we can apply the combination axiom A5 obtaining:

$$((\phi_i^{(i)}) \otimes (\phi_{i+1}^{(i)} \otimes \cdots \otimes \phi_r^{(i)}))^{\downarrow y_{i+1}} = (\phi_i^{(i)})^{\downarrow d(\phi_i^{(i)}) \wedge y_{i+1}} \otimes (\phi_{i+1}^{(i)} \otimes \cdots \otimes \phi_r^{(i)})$$

Considering the way we numbered the tree nodes we obtain that $d(\phi_i^{(i)}) \wedge y_{i+1} = d(\phi_i^{(i)}) \wedge \lambda(pa(i))$ due to the running intersection property and the fact that $i < pa(i)$

Furthermore, by theorem 3.9 we obtain $d(\phi_i^{(i)}) \wedge y_{i+1} = \lambda(i) \wedge \lambda(pa(i)) = sep(i)$.
So:

$$\begin{aligned} \phi^{\downarrow y_{i+1}} &= (\phi_i^{(i)})^{\downarrow d(\phi_i^{(i)}) \wedge y_{i+1}} \otimes (\phi_{i+1}^{(i)} \otimes \cdots \otimes \phi_r^{(i)}) = ((\phi_i^{(i)})^{sep(i)} \otimes \phi_{pa(i)}^{(i)}) \otimes \bigotimes_{\substack{j \in \{i+1,\dots,r\} \\ j \neq pa(i)}} \phi_j^{(i)} = \\ &= (\mu_{i \to pa(i)} \otimes \phi_{pa(i)}^{(i)}) \otimes \bigotimes_{\substack{j \in \{i+1,\dots,r\} \\ j \neq pa(i)}} \phi_j^{(i)} = \phi_{pa(i)}^{(i+1)} \otimes \bigotimes_{\substack{j \in \{i+1,\dots,r\} \\ j \neq pa(i+1)}} \phi_j^{(i+1)} = \bigotimes_{j \in \{i+1,\dots,r\}} \phi_j^{(i+1)} \end{aligned}$$

□

4 Valuation Algebras on Partition Lattices

Next, we are going to show how to define a semiring induced valuation algebra on a partition lattice domain.

The reason to take partition lattice domains is that they are the most general finite lattices (see theorem 4.4,). Therefore any result on a partition lattice can be used on any lattice.

On the other hand, we define the valuation algebra over a semiring because Action-GDL need this algebraic structure.

Thus in section 4.1 we recall what partition lattices are, whereas next section will introduce the raise and decrease operations which will be useful to define valuations on partition lattices.

Finally, the last sections of the chapter refresh what semirings and multisets are and we use these concepts to define a semiring induced valuation algebra on a partition lattice. It is important to remark that most of the results and concepts that one can find in this section are completely original and have been designed in order to define this valuation algebra.

4.1 Partition Lattices

Definition 4.1.
A *partition* $\Pi = \{B_i : 1 \leq i \leq n\}$ of a set X called *universe* consists of a collection of subsets $B_i \subset X$ called *blocks* such that:

- $B_i \neq \emptyset$
- $B_i \cap B_j = \emptyset$ for $i \neq j$ and $1 \leq i, j \leq n$
- $\bigcup_{i=1}^{n} B_i = X$

We denote the *set of* all possible *partitions* of a universe X by $Part(X)$

Definition 4.2.
Let X be a set and $\pi_1, \pi_2 \in Part(X)$.
We say that π_1 is *finer* than π_2 ,or π_2 is *coarser* than π_1, if every block of π_1 is contained in some block of π_2 or equivalently if every block of π_1 is a union of blocks from π_2.

Figure 1 shows an example of two partitions which can be compared. Notice that each block in the finer partition is contained in some block of the coarser partition.

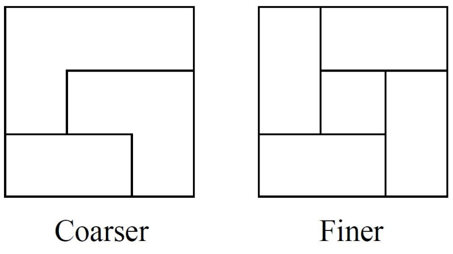


Figure 1: Partitions

Definition 4.3. Using the last definition we can induce two different partial order relations over $Part(X)$:

1. $\pi_1 \leq_1 \pi_2 \;\Leftrightarrow \pi_2$ is coarser than π_1.

2. $\pi_1 \leq_2 \pi_2 \;\Leftrightarrow \pi_2$ is finer than π_1.

It can be proved that $(Part(X), \leq_1)$ is a complete lattice with $\bot_1 = \{\{x\} : x \in X\}$ as a bottom element, and $\top_1 = \{X\}$ as a top element.

Equivalently, $(Part(X), \leq_2)$ is a complete lattice with $\bot_2 = \{X\}$ as a bottom element, and $\top_2 = \{\{x\} : x \in X\}$ as a top element.

From now on, we will call $(Part(X), \leq_2)$ the *partition lattice of a universe* X and we will denote it by $(Part(X), \leq)$ or $Part(X)$.

We will also denote it's bottom element $\{X\}$ by $\bot$, and its unique block X by $\diamond$

Next theorem will allow us to generalize valuation algebras in order to contain DCOP valuation algebra in example 1 as long as the valuation algebra related to CSG problem. The proof of this theorem can be found on [2].

Theorem 4.4. *For every finite lattice L there is a finite set X such that L can be embedded in* $Part(X)$

It is important to remark that, despite theorem 4.4 just refers to finite lattices, the same result holds for infinite lattices [2].

Although most definitions and some results in this chapter hold for partiton lattices of infinite universes, we will consider them to be finite later on, as long as one can only store a finite quantity of information.

4.2 Rise and Decrease

In this section we will talk about rise and decrease, two operations on partition lattices which we defined in order to generalize the projection of variables and the union of projections seen in example 1.

4.2.1 Definition of rise and decrease

Definition 4.5.
Let X be a set and $\pi_1, \pi_2 \in Part(X)$ such that $\pi_1 \leq \pi_2$. We define the *raise of a block* $b \in \pi_1$ *to* π_2 as:

$$b \Uparrow \pi_2 = \{x \in \pi_2 : x \subseteq b\}$$

Figure 2 shows the raise of a block using the partitions shown in figure 1:

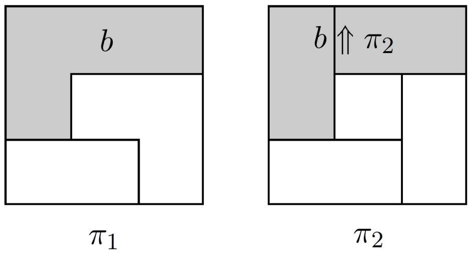


Figure 2: Raise of a Block

Definition 4.6.
Let X be a set and $\pi_1, \pi_2 \in Part(X)$ such that $\pi_1 \leq \pi_2$. We define the *decrease of a block* $b \in \pi_2$ *to* π_1 as:

$$b \Downarrow \pi_1 = \{x \in \pi_1 : b \subseteq x\}$$

Figure 3 shows the decrease of a block using the same partitions that where used in the previous figures:

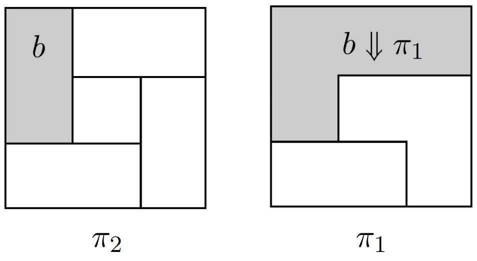


Figure 3: Decrease of a Block

Proposition 4.7.
Let X be a set and $\pi_1, \pi_2 \in Part(X)$ such that $\pi_1 \leq \pi_2$. Let $b \in \pi_2$ be a block. Then:

$b \Downarrow \pi_1$ has only one element.

Proof.
Let $x, y \in \pi_1$ such that $x, y \in b \Downarrow \pi_1$.
$(x, y) \in b \Downarrow \pi_1$ implies that $b \subseteq x$ and $b \subseteq y$, thus $b \subseteq x \cap y$, in particular $x \cap y \neq \emptyset$.
As long as $x, y \in \pi_1$ are blocks, we know that $x \neq y$ implies $x \cap y \neq \emptyset$, thus we obtain $x = y$ □

Definition 4.8.
Let X be a set and $\pi_1, \pi_2 \in Part(X)$ such that $\pi_1 \leq \pi_2$. Let $b \in \pi_2$ be a block. We will denote the only element in $b \Downarrow \pi_1$ as b_{π_1}

4.2.2 Properties of rise and decrease

As long as rise and decrease will generalize the projection it turns necessary to know which properties do they satisfy. Next proposition just enumerates some properties, whereas the following lemmas will be necessary to define a valuation algebra.

Proposition 4.9. ***Properties of rise and decrease***
Let X be a set and $\pi_1, \pi_2 \in Part(X)$ such that $\pi_1 \leq \pi_2$. Let $a \in \pi_1$ and $b \in \pi_2$ be blocks. Then:

1. $x \neq b_{\pi_1} \Rightarrow b \cap x = \emptyset \ , \qquad \forall x \in \pi_1$
2. $a = \bigcup_{x \in a \Uparrow \pi_2} x$
3. $\forall x \in (a \Uparrow \pi_2) \Rightarrow x_{\pi_1} = a$
4. $b \in (b_{\pi_1} \Uparrow \pi_2)$

Proof.
They are direct from the definition of raise and decrease. □

Lemma 1.
Let X be a set and $\pi_0, \pi_1, \pi_2 \in Part(X)$ such that $\pi_0 \leq \pi_1 \leq \pi_2$. Let b be a block from π_2. Then:

$$b_{\pi_0} = (b_{\pi_1})_{\pi_0}$$

Proof.
We know that $b \subseteq b_{\pi_1} \subseteq (b_{\pi_1})_{\pi_0} \in \pi_0$. We also know that it only exist a block in π_2 that contains b and this block is b_{π_0} Proposition 4.7 □

Notice that Figure 4 shows that $b_{\pi_0} = (b_{\pi_1})_{\pi_0}$, therefore it is an example of lemma 1.

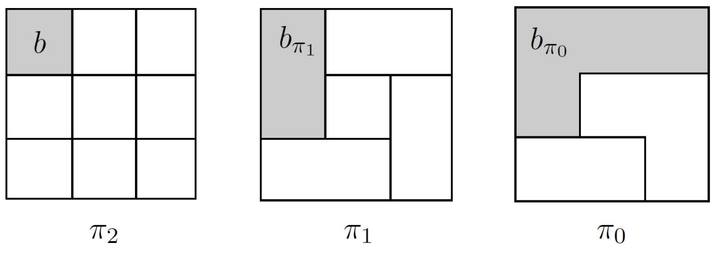


Figure 4: Lemma 1 representation

Lemma 2.
Let X be a set and $\pi_0, \pi_1, \pi_2 \in Part(X)$ such that $\pi_0 \leq \pi_1 \leq \pi_2$. Let b be a block from π_0. Then:

$$b \Uparrow \pi_2 = \bigcup_{x \in b \Uparrow \pi_1} x \Uparrow \pi_2$$

Proof.
Let $z \in \pi_2$ such that $z \in \bigcup_{x \in b \Uparrow \pi_1} x \Uparrow \pi_2$.

Then it exist a block $x \in (b \Uparrow \pi_1)$ such that $z \in (x \Uparrow \pi_2)$, which implies that $x \subseteq b$ and $z \subseteq x$, so $z \subseteq b$. Hence, in particular $z \in (b \Uparrow \pi_2)$

Suppose now that $z \in b \Uparrow \pi_2$.
We know, by property 4, that $z \in z_{\pi_1} \Uparrow \pi_2$ so we only need to prove that $z_{\pi_1} \in b \Uparrow \pi_1$.

We also know that $b = z_{\pi_0}$ property 3, and the previous lemma tells us that $z_{\pi_0} = (z_{\pi_1})_{\pi_0}$, but $z_{\pi_0} = b$ which means that $z_{\pi_1} \in (b \Uparrow \pi_1)$ property 4.

□

Figure 5 shows an example of lemma 2. Notice that if we raise both blocks in $b \Uparrow \pi_1$ we just obtain the blocks in $b \Uparrow \pi_2$.

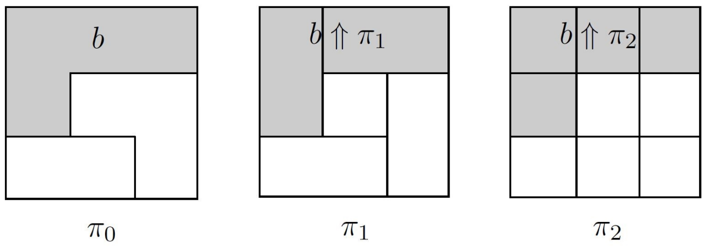


Figure 5: Lemma 2 representation

4.2.3 Rise and decrease of block sets

We have already defined how to raise and decrease a block, but looking at lemma 2 one may deduce that sometimes it turns necessary to raise or decrease a set of blocks. Next we will give a formal definition for set raising and decreasing. In addition, we will rewrite the properties already seen in such a way that they will hold for block sets.

Definition 4.10.
Let X be a set and $\pi_1, \pi_2 \in Part(X)$ such that $\pi_1 \leq \pi_2$. We define the *raise of a subset* $A \subseteq \pi_1$ *to* π_2 as:

$$A \Uparrow \pi_2 = \bigcup_{x \in A} x \Uparrow \pi_2$$

Definition 4.11.
Let X be a set and $\pi_1, \pi_2 \in Part(X)$ such that $\pi_1 \leq \pi_2$. We define the *decrease of a subset* $A \subseteq \pi_2$ *to* π_1 as:

$$A \Downarrow \pi_1 = \bigcup_{x \in A} x \Downarrow \pi_1 = \{x_{\pi_1} : x \in A\}$$

Remark: $\{b\} \Downarrow \pi_1 = \{b_{\pi_1}\}$

With these definitions above we can rewrite lemma 1 and lemma 2 as it follows:

Lemma 1
Let X be a set and $\pi_0, \pi_1, \pi_2 \in Part(X)$ such that $\pi_0 \leq \pi_1 \leq \pi_2$. Let A be a subset of π_2. Then:

$$A \Downarrow \pi_0 = (A \Downarrow \pi_1) \Downarrow \pi_0$$

Lemma 2
Let X be a set and $\pi_0, \pi_1, \pi_2 \in Part(X)$ such that $\pi_0 \leq \pi_1 \leq \pi_2$. Let A be a subset of π_0. Then:

$$A \Uparrow \pi_2 = (A \Uparrow \pi_1) \Uparrow \pi_2$$

Moreover, given X a set, $\pi_1, \pi_2 \in Part(X)$ such that $\pi_1 \leq \pi_2$, $A \subseteq \pi_1$ and $B \subseteq \pi_2$ we can also rewrite the properties 3 and 4 of the rise and decrease:

3. $A = A \Uparrow \pi_2 \Downarrow \pi_1$.
4. $B \subseteq B \Downarrow \pi_1 \Uparrow \pi_2$.

As it's clear than $A = B$ implies that their rise and decrease will be equal we obtain the following result.

Proposition 4.12.
Let X be a set and $\pi_1, \pi_2 \in Part(X)$ such that $\pi_1 \leq \pi_2$. Let $A, B \subseteq \pi_1$. Then:

$$A \Uparrow \pi_2 = B \Uparrow \pi_2 \Leftrightarrow A = B$$

Proof.
$A = B \Rightarrow A \Uparrow \pi_2 = B \Uparrow \pi_2$ is clear, so we only have to demonstrate $A \Uparrow \pi_2 = B \Uparrow \pi_2 \Rightarrow A = B$. Due to the property 3 we obtain:

$$A \Uparrow \pi_2 = B \Uparrow \pi_2 \Rightarrow A = A \Uparrow \pi_2 \Downarrow \pi_1 = B \Uparrow \pi_2 \Downarrow \pi_1 = B$$

. □

Remark: $A \Downarrow \pi_1 = B \Downarrow \pi_2$ doesn't imply that $A = B$, because property 4 doesn't give an equality but an inclusion.

As we will see later, the following lemma will allow us to know when it is possible to define a valuation algebra on a partition lattice.

Lemma 3.
Let X be a set and $\pi_0, \pi_1, \pi_2, \pi_3 \in Part(X)$ such that $\pi_0 \leq \pi_1 \leq \pi_3$ and $\pi_0 \leq \pi_2 \leq \pi_3$. Let $b \in \pi_2$ be a block . Then:

$$b \Uparrow \pi_3 \Downarrow \pi_1 \subseteq b \Downarrow \pi_0 \Uparrow \pi_1$$

Proof.
The property 3 tells us that $x \Downarrow \pi_0 \Uparrow \pi_1 = x \Downarrow \pi_0 \Uparrow \pi_1 \Uparrow \pi_3 \Downarrow \pi_1$.

Then, by lemma 2 we obtain: $x \Downarrow \pi_0 \Uparrow \pi_1 \Uparrow \pi_3 \Downarrow \pi_1 = x \Downarrow \pi_0 \Uparrow \pi_3 \Downarrow \pi_1 = x \Downarrow \pi_0 \Uparrow \pi_2 \Uparrow \pi_3 \Downarrow \pi_1 = (x \Downarrow \pi_0 \Uparrow \pi_2) \Uparrow \pi_3 \Downarrow \pi_1$ which is a superset of $x \Uparrow \pi_3 \Downarrow \pi_1$ due to property 4. □

4.3 Semirings and Multisets

Before being able to define a valuation algebra on partition lattice we need to recall what $R-$ *multisets* are, as long as define the raise and decrease of multisets. In order to do that, we will introduce the needed concepts during the following pages.

Definition 4.13.
A *semiring* is a tuple $\langle A, +, \times \rangle$, where $+$ and $\times$ are binary operations such that:

- $+$ and $\times$ are both associative
- $+$ is commutative
- for $a, b, c \in R$: $a \times (b + c) = a \times b + a \times c$
- for $a, b, c \in R$: $(a + b) \times c = a \times c + b \times c$
- $+$ has a neutral element e_+
- $\times$ has a neutral element $e_\times$
- $a \times e_+ = e_+ \times a = e_+$ for all $a \in R$

A semiring is called *commutative* if $a \times b = b \times a$ for all $a, b \in A$

A semiring is called cancellative if for all $a, b, c \in A$ we have:

$$a \times b = a \times c \Rightarrow b = c$$

Definition 4.14.
Let X be a finite set and R be a semiring. Given $\Pi \in Part(X)$ a *R-multiset* over Π is a mapping

$$\begin{aligned} \Phi : \Pi &\to R \\ b &\mapsto \Phi(b) \end{aligned}$$

We say that Π is the domain of Φ and we denote it by $d(\Phi)$

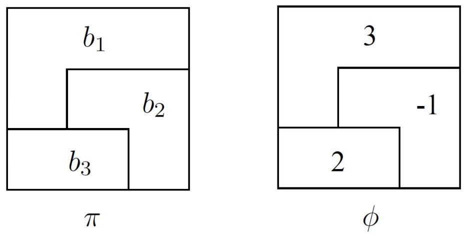
Example 2. *$R-$Multiset over a partition*

This example will show how a $R-$Multiset over a partition can be represented.

Let $R = \langle \mathbb{Z}, +, \times \rangle$ and $\Pi = \{b_1, b_2, b_3\}$ be a partiton with three blocks.
Then the R-multiset defined by

$$\begin{aligned} \Phi : \Pi &\to \mathbb{Z} \\ b_1 &\mapsto 3 \\ b_2 &\mapsto -1 \\ b_3 &\mapsto 2 \end{aligned}$$

will be represented as:



Next, we are going to define the raise and decrease of multisets, which will turn very important in order to define combination and projection operations in section 4.4.

Definition 4.15.
Let X be a finite set and R be a semiring. Let $\pi_1, \pi_2 \in Part(X)$ such that $\pi_1 \leq \pi_2$. We define the *raise of a R-multiset* $\Phi : \pi_1 \to R$ over π_1 to π_2 as the R-multiset over π_2 defined by:

$$\begin{aligned} (\Phi \Uparrow \pi_2) \ : \pi_2 &\longrightarrow R \\ b &\longmapsto \Phi(b_{\pi_1}) \end{aligned}$$

Although one may imagine how the raise of a $R-$multiset looks like, figure 6 shows it.

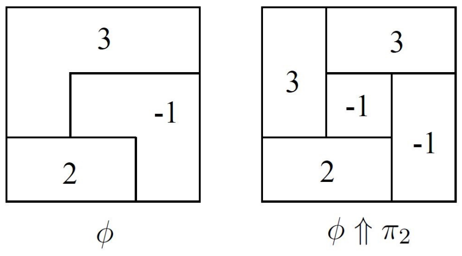


Figure 6: Raise of a Multiset

Definition 4.16.
Let X be a finite set and R be a semiring. Let $\pi_1, \pi_2 \in Part(X)$ such that $\pi_1 \leq \pi_2$. We define the *decrease of a R-multiset* $\Phi : \pi_2 \to R$ over π_2 to π_1 as the R-multiset over π_1 defined by:

$$\begin{array}{rccc} (\Phi \Downarrow \pi_1) : & \pi_1 & \longrightarrow & R \\ & b & \longmapsto & \displaystyle\sum_{a \in (b \Uparrow \pi_2)} \Phi(a) \end{array}$$

Figure 7 shows an example of a Multiset decrease.

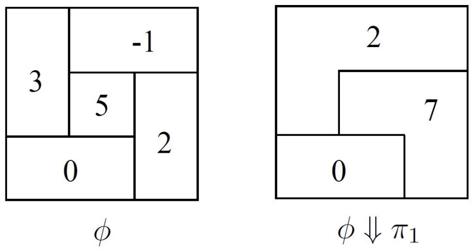


Figure 7: Decrease of a Multiset

4.4 Defining a valuation algebra on a partition lattice

Notice that $R-$multisets over partitions can be seen as information contained in each block. We will use this fact in order to define a valuation algebra on partition lattices.

First of all we must define the operations of labeling, combination and projection. This will be done in the following section, whereas in section 4.4.2 we will create the desired valuation algebra.

4.4.1 Valuations and operations on partition lattices

Definition 4.17.
Let X be a finite set and R be a semiring. Let ϕ be a R-multiset over $\pi_1 \in Part(X)$. Let ψ be a R-multiset over $\pi_2 \in Part(X)$. We define the *combination* of ϕ and ψ as the following R-multiset over $\pi_1 \vee \pi_2$:

$$\begin{array}{cccc} (\phi \otimes \psi): & \pi_1 \vee \pi_2 & \longrightarrow & R \\ & b & \longmapsto & \phi(b_{\pi_1}) \times \psi(b_{\pi_2}) \end{array}$$

Notice that as we can see in figure 8 it holds that $\phi \otimes \psi = (\phi \Uparrow (\pi_1 \vee \pi_2)) \times (\psi \Uparrow (\pi_1 \vee \pi_2))$.

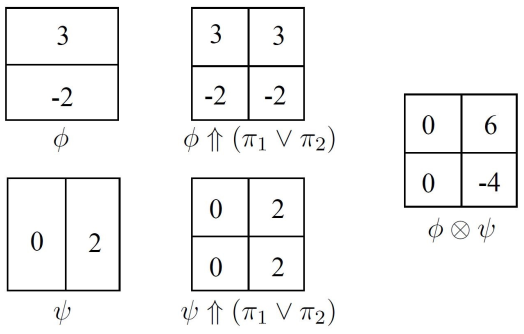

3
-2
ϕ
3 3
-2 -2
$\phi \Uparrow (\pi_1 \vee \pi_2)$
0 2
ψ
0 2
0 2
$\psi \Uparrow (\pi_1 \vee \pi_2)$
0 6
0 -4
$\phi \otimes \psi$

Figure 8: Combination

Definition 4.18.
Let X be a finite set and R be a semiring. Let $\pi_1, \pi_2 \in Part(X)$ such that $\pi_1 \leq \pi_2$ and let ϕ be a R-multiset over π_2. We define the *projection of* ϕ to π_1 as:

$$\phi^{\downarrow \pi_1} = (\phi \Downarrow \pi_1)$$

Definition 4.19.
Let X be a finite set and $D \subseteq Part(X)$
Let R be a commutative semiring.

We denote by Φ the finite set of R-multisets over elements from D

For each $\pi \in D$ we define Φ_π as the *set of R-multisets over* π.

We define a *R-valuation algebra candidate* over D as the tuple $\langle \Phi, D \rangle$ with the following operations:

- *Labeling*:$\Phi \to D$; $\phi \mapsto d(\phi)$
- *Combination*: $\Phi \otimes \Phi \to \Phi$; $(\phi, \psi) \mapsto \phi \otimes \psi$
- *Projection*: $\Phi \times D \to \Phi$; $(\phi, \pi) \mapsto \phi \Downarrow \pi$, for $\pi \leq d(\phi)$

4.4.2 Valuable partition lattices

Although we have just defined valuations on partition lattice we still must prove that the satisfy the valuation algebra axioms (see definition 2.2). Unfortunately, valuation algebra axioms don't hold for any set of valuations on partition lattices, therefore we must define when it is possible to define a valuation algebra.

Definition 4.20.
Let X a finite set. A *valuable partition lattice* over X is defined as any sublattice $D \subseteq Part(X)$ such that for all $\pi_1, \pi_2, \pi_3 \in D$ such that $\pi_1 \leq \pi_3$, $\pi_2 \leq \pi_3$ we have:

$$b \Downarrow (\pi_1 \wedge \pi_2) \Uparrow \pi_1 = b \Uparrow \pi_3 \Downarrow \pi_1 \qquad \text{for all } b \in \pi_2$$

Recall that Lemma 3 told us that $b \Downarrow (\pi_1 \wedge \pi_2) \Uparrow \pi_1 \subseteq b \Uparrow \pi_3 \Downarrow \pi_1$ is always satisfied. Therefore valuable partition lattices satisfy the equality instead of the subset relation in Lemma 3.

Figure 9 shows this property that all blocks in a valuable partition lattice must satisfy. Hence, the partition lattice $\langle \{\pi_1, \pi_2, \pi_3, \pi_1 \wedge \pi_2\}, \vee, \wedge \rangle$ is in fact a valuable partition lattice.

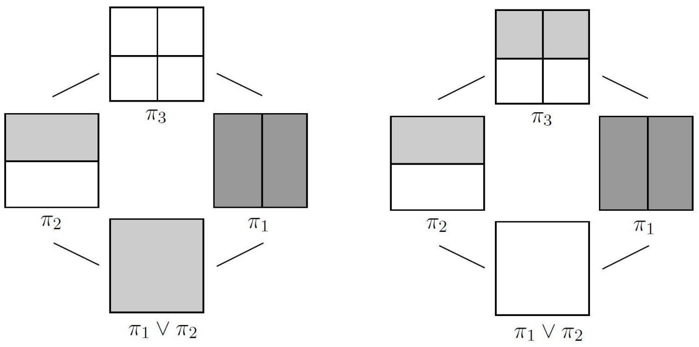


Figure 9: A valuable partition latice

Notice that in Figure 9 we have that $\pi_3 = \pi_1 \vee \pi_2$. This has been made on purpose in order to introduce the following proposition. It will give us an alternative definition for valuable partition lattices, as long as an important property which they satisfy.

Proposition 4.21.
Let X be a finite set and $D \subseteq Part(X)$ a sublattice. Then:

D is a valuable partition lattice over X
if and only if
$$b \Downarrow (\pi_1 \wedge \pi_2) \Uparrow \pi_1 = b \Uparrow (\pi_1 \vee \pi_2) \Downarrow \pi_1 \quad \textit{for all } \pi_1, \pi_2 \in D,\ b \in \pi_2$$

Proof.
It's clear that $\pi_1 \leq \pi_3$ and $\pi_2 \leq \pi_3$ implies $(\pi_1 \vee \pi_2) \leq \pi_3$.

In particular if $\pi_2 \Downarrow (\pi_1 \wedge \pi_2) \Uparrow \pi_1 = \pi_2 \Uparrow (\pi_1 \vee \pi_2) \Downarrow \pi_1$, for all $\pi_1, \pi_2 \in D$ we can use some results already proven in order to obtain that D is a valuable partition lattice over X:

$$b \Downarrow (\pi_1 \wedge \pi_2) \Uparrow \pi_1 = b \Uparrow (\pi_1 \vee \pi_2) \Downarrow \pi_1 = b \Uparrow (\pi_1 \vee \pi_2) \Uparrow \pi_3 \Downarrow (\pi_1 \vee \pi_2) \Downarrow \pi_1 = b \Uparrow \pi_3 \Downarrow \pi_1$$

On the other side, if we set $\pi_3 = \pi_1 \vee \pi_2$ it holds that $\pi_1 \leq \pi_3$ and $\pi_2 \leq \pi_3$.

So using the valuable partition lattice definition we obtain $b \Downarrow (\pi_1 \wedge \pi_2) \Uparrow \pi_1 = \Uparrow \pi_3 \Downarrow \pi_1 = \Uparrow (\pi_1 \vee \pi_2) \Downarrow \pi_1$. □

Proposition 4.22.
Let D be a valuable partition lattice over a finite set X and let $\pi_0, \pi_1, \pi_2 \in D$ such that $\pi_0 \leq \pi_2 \leq \pi_0 \vee \pi_1$. Then we have:

$$(b \Downarrow (\pi_1 \wedge \pi_2)) \Uparrow \pi_1 = (b \Uparrow (\pi_0 \vee \pi_1)) \Downarrow \pi_1 \ , \qquad \textit{for all } b \in \pi_2$$

Proof.
Using $\vee$ definition, one can easily deduce $\pi_0 \vee \pi_1 = \pi_2 \vee \pi_1$ from the expression $\pi_0 \leq \pi_2 \leq \pi_0 \vee \pi_1$. Thus from the definition of valuable partition lattice we obtain $(b \Downarrow (\pi_1 \wedge \pi_2)) \Uparrow \pi_1 = (b \Uparrow (\pi_1 \vee \pi_2)) \Downarrow \pi_1 = (b \Uparrow (\pi_0 \vee \pi_1)) \Downarrow \pi_1$. □

Thinking about valuation algebras axioms, one may deduce that most problematic axiom when defining valuation algebras is the combination axiom (A5). Recall that theorem 2.6 shows that a modular lattice is needed in order to define a good valuation algebra. On the other side last proposition will allow us to prove that valuations on a valuable partition lattice satisfy the axiom. Therefore we obtain the following theorem:

Theorem 4.23.
Let X be a finite set and $D \subseteq Part(X)$ and let $\langle \Phi, D \rangle$ be a R-valuation algebra candidate. Then:

$$D \textit{ modular valuable partition lattice} \Rightarrow \langle \Phi, D \rangle \textit{ is a valuation algebra.}$$

Proof.
We have to verify the six axioms related to valuation algebras. In order to that we will use the combination and projection definitions given in section 4.4.1. Therefore we will write the valuation algebras' axioms as it follows:

A1 Φ is associative and commutative under $\otimes$.

As long as commutativity is guaranteed by $\times$ commutativity on R we only have to verify the associativity.

Let $\phi, \psi, \varphi \in \Phi$ with domains π_1, π_2, π_3 respectively.

First of all we must verify that $d(\phi \otimes \psi) \otimes \varphi) = d(\phi \otimes (\psi \otimes \varphi))$

$d(\phi \otimes \psi) \otimes \varphi) = (\pi_1 \vee \pi_2) \vee \pi_3 = \pi_1 \vee (\pi_2 \vee \pi_3) = d(\phi \otimes (\psi \otimes \varphi))$

Now, let $b \in \pi_1 \vee \pi_2 \vee \pi_3$ be a block.

$((\phi \otimes \psi) \otimes \varphi)\ (b) = (\phi \otimes \psi)(b_{\pi_1 \vee \pi_2}) \times \varphi(b_{\pi_3}) = \phi((b_{\pi_1 \vee \pi_2})_{\pi_1}) \times \psi((b_{\pi_1 \vee \pi_2})_{\pi_2}) \times \varphi(b_{\pi_3}) = \phi(b_{\pi_1}) \times \psi(b_{\pi_2}) \times \varphi(b_{\pi_3})$

By a similar reasoning we obtain $(\phi \otimes (\psi \otimes \varphi))\ (b) = \phi(b_{\pi_1}) \times \psi(b_{\pi_2}) \times \varphi(b_{\pi_3})$

A2 For $\phi, \psi \in \Phi$ we have that $d(\phi \otimes \psi) = d(\phi) \vee d(\psi)$

It's clear due to the definition of the combination.

A3 For $\phi \in \Phi$, $\pi \in D$ such that $\pi \leq d(\phi)$ we have that $d(\phi \Downarrow \pi) = \pi$

It's clear due to the definition of the projection.

A4 For $\phi \in \Phi$, $\pi_0, \pi_1, \pi_2 \in D$ such that $\pi_0 \leq \pi_1 \leq \pi_2 = d(\phi)$ we have that $(\phi \Downarrow \pi_1) \Downarrow \pi_0 = \phi \Downarrow \pi_0$

Let $b \in \pi_0$ be a block. By the lemma 2 we have:

$$((\phi \Downarrow \pi_1) \Downarrow \pi_0)\ (b) = \sum_{a \in (b \Uparrow \pi_1)} (\phi \Downarrow \pi_1)\ (a) = \sum_{a \in (b \Uparrow \pi_1)} \sum_{x \in (a \Uparrow \pi_2)} \phi\ (x) = \sum_{x \in (\bigcup_{b \Uparrow \pi_1} x \Uparrow \pi_2)} \phi\ (x) = \sum_{x \in (b \Uparrow \pi_1) \Uparrow \pi_2} \phi\ (x) = \sum_{x \in b \Uparrow \pi_2} \phi\ (x) = (\phi \Downarrow \pi_0)\ (b)$$

A5 For $\phi, \psi \in \Phi$, with domain π_0 and π_1 respectively and for all $\pi_2 \in D$ such that $\pi_0 \leq \pi_2 \leq \pi_0 \vee \pi_1$, we have that $(\phi \otimes \psi) \Downarrow \pi_2 = \phi \otimes (\psi \Downarrow \pi_2 \wedge \pi_1))$

First of all we must verify that $d((\phi \otimes \psi) \Downarrow \pi_2) = d(\phi \otimes (\psi \Downarrow \pi_2 \wedge \pi_1)))$.

It's clear that $d((\phi \otimes \psi) \Downarrow \pi_2) = \pi_2$. On the other side $d(\phi \otimes (\psi \Downarrow \pi_2 \wedge \pi_1))) = \pi_0 \vee (\pi_2 \wedge \pi_1) = \pi_2$ due to Proposition 2.5

Now, let $b \in \pi_2$ be a block.
$(\phi \otimes (\psi \Downarrow \pi_2 \wedge \pi_1))\ (b) = \phi(b_{\pi_0}) \times (\psi \Downarrow (\pi_2 \wedge \pi_1))\ (b_{\pi_2 \wedge \pi_1}) = \phi(b_{\pi_0}) \times \sum_{x \in (b_{\pi_2 \wedge \pi_1} \Uparrow \pi_1)} \psi(x)$.

$((\phi \otimes \psi) \Downarrow \pi_2)\ (b) = \sum_{x \in (b \Uparrow (\pi_0 \vee \pi_1))} (\phi \otimes \psi)(x) = \sum_{x \in (b \Uparrow (\pi_0 \vee \pi_1))} \phi(x_{\pi_0}) \times \psi(x_{\pi_1})$.

We must remark that for all $x \in (b \Uparrow (\pi_0 \vee \pi_1))$ we have $x_{\pi_0} = x_{(\pi_0 \vee \pi_1)_{\pi_0}} = b_{\pi_0}$.

From that, we obtain $((\phi \otimes \psi) \Downarrow \pi_2)\ (b) = \phi(b_{\pi_0}) \times \sum_{x \in (b \Uparrow (\pi_0 \vee \pi_1))} \psi(x_{\pi_1})$, so for ending the proof we have to verify that

$$\sum_{x \in (b_{\pi_2 \wedge \pi_1} \Uparrow \pi_1)} \psi(x) = \sum_{x \in (b \Uparrow (\pi_0 \vee \pi_1))} \psi(x_{\pi_1})$$

Equivalently we can verify $b_{\pi_2 \wedge \pi_1} \Uparrow \pi_1 = \{x_{\pi_1} : x \in (b \Uparrow (\pi_0 \vee \pi_1))\}$ but as long as D is a valuable partition lattice we have $b_{\pi_2 \wedge \pi_1} \Uparrow \pi_1 = b \Downarrow \pi_2 \wedge \pi_1 \Uparrow \pi_1 = b \Uparrow (\pi_0 \vee \pi_1) \Downarrow \pi_1 = \{x_{\pi_1} : x \in (b \Uparrow (\pi_0 \vee \pi_1))\}$ where the second equality is due to Proposition 4.22.

A6 For $\phi \in \Phi$, $\phi \Downarrow d(\phi) = \phi$

It's clear that for each $\pi \in Part(X)$ we have $b_\pi = b$ for all $b \in \pi$.

$\square$

Although we have just shown how to define a valuation algebra on a partition lattice, we would like to prove that it has been done in the most general way possible.

Notice that it can easily be proven that if $\langle \Phi, D \rangle$ is a valuation algebra it holds that D is a modular valuable partition lattice. We already know that modularity is needed and we only have to accept the axiom 5 in order to obtain that valuable partition lattices are also needed.

Theorem 4.24.
Let X be a finite set and $D \subseteq Part(X)$ and let $\langle \Phi, D \rangle$ be a R-valuation algebra candidate. Then:

$$D \text{ modular valuable partition lattice} \Leftrightarrow \langle \Phi, D \rangle \text{ is a valuation algebra.}$$

From now on we will do a distinction between the valuation algebras notation and the $R-$multisets and partition notation.
Therefore, when talking about valuation algebras domains we will write elements of the lattice x, y, z instead of the partition notation π, π_1, π_2. Moreover, we will write $\phi^{\downarrow x}$ instead of $\phi \Downarrow \pi$.

5 Dynamic Programming

As we saw in chapter 3, the collect algorithm can be generalized from distributive lattices to modular lattices. In this chapter we will generalize the Action-GDL algorithm, which was created to solve DCOPs.

This chapter relies in the chapter 8 of [8], where it is shown the Action-GDL under the name of *computing solutions without distribute*. Therefore, the domain lattices in that chapter are powerset lattices, which are boolean lattices. In fact, most of the proofs and definitions are made using the existence of a complementary domain.

On the other hand, we will work on modular valuable partition lattices domains, which in general do not satisfy neither the existence of a complementary nor the distributive property, therefore we will do a generalization of the contents in [8].

It is important to remark that during the study of chapter 8 in [8] we came across with a counterexample of theorem 8.1. That theorem plays a very important role on that chapter because all the algorithms proposed there rely on it.

Fortunately we found how to fix the problem with a little change in the theorem statement, but it has a little cost: the semiring $\langle A, +, \times\rangle$ employed to define the valuation algebra must be cancellative.

Nevertheless, in most practical applications, Action-GDL uses the tropical or the arctic semirings, defined as $\langle \mathbb{R} \cup \{+\infty\}, min, +\rangle$ and $\langle \mathbb{R} \cup \{-\infty\}, max, +\rangle$ respectively, which are cancellative semirings [4].

5.1 Idempotent Valuation Algebras on Partition Lattices

As long as Action-GDL works on tropical or arctic semirings induced valuation algebras it makes sense to think that it would work on any valuation algebra over a semiring with the same properties.

These properties, apart from cancellativity, are a total order and an idempotent $+$. We will define these concepts as long as some important results related to them in this section.

Definition 5.1.
A semiring $\langle A, +, \times\rangle$ is said to be *idempotent* if $a + a = a$ for all $a \in A$.

In order to know whether $+$ is idempotent or not we will write $\otimes$ instead of $+$ when it must be an idempotent operation.

Definition 5.2.
Given $R = \langle A, +, \times\rangle$ a semiring, we define the *canonical partial order* $\leq$ in R by setting

$$a \leq b \Leftrightarrow \text{there is } c \in A \text{ such that } a + c = b$$

If this order is a total order, then we will say that our semiring is a totally ordered semiring.

The proposition below will be useful in order to work with idempotent semirings.

Proposition 5.3.
Let $\langle A, \oplus, \times\rangle$ be an idempotent semiring. Then:

$$a \leq b \Leftrightarrow a \oplus b = b$$

Proof.
It is clear that $a \oplus b = b \Rightarrow a \leq b$. Therefore we only have to prove the opposite implication.

Suppose $a \leq b$, i.e. there is $c \in A$ such that $a \oplus c = b$. Then by idempotency we obtain:

$$a \oplus b = a \oplus a \oplus c = a \oplus c = b$$

□

Notice that in any totally ordered idempotent semiring we have that $a \oplus b = max_{\oplus}(a, b)$, where $max_{\oplus}$ is the maximum according to the canonical order defined by $\oplus$.

The propositions below generalize this idea showing that $\bigoplus\limits_{x \in X} x = \ \max_{\otimes}\{x : x \in X\}$.

Proposition 5.4.
Let $\langle A, \oplus, \times\rangle$ be a totally ordered, idempotent semiring and let $X = \{x_1, \ldots, x_n\} \subset A$. Then:

$$\textit{There is } x_i \in X \textit{ such that } \bigoplus_{x \in X} x = x_i$$

Proof.
The case $n = 1$ is immediate.

For $|X| \geq 2$ we will suppose that our proposition holds for any $Y \subseteq A$ such that $|Y| \leq n - 1$.

By the associative property of $\oplus$ we obtain $\bigoplus\limits_{x \in X} x = x_1 \oplus \bigoplus\limits_{x \in X\backslash\{x_1\}} x$
As long as $\langle A, \oplus, \times\rangle$ is a totally ordered semiring, at least one of the following expressions must be satisfied:

- $x_1 \geq \bigoplus\limits_{x \in X\backslash\{x_1\}} x$, which would imply that $x_1 \oplus \bigoplus\limits_{x \in X\backslash\{x_1\}} x = x_1$
- $x_1 \leq \bigoplus\limits_{x \in X\backslash\{x_1\}} x$, which would imply that $x_1 \oplus \bigoplus\limits_{x \in X\backslash\{x_1\}} x = \bigoplus\limits_{x \in X\backslash\{x_1\}} x = x_j$ for some $1 \lneq j$

□

***Remark*:** Notice that due to the proof, it is immediate to see that $x_i \geq x_j$ for all $x_j \in X$. In particular we obtain:

$$\bigoplus_{x \in X} x = \ \max_{\otimes}\{x : x \in X\}$$

Proposition 5.5.
Let $\langle A, \oplus, \times \rangle$ be a totally ordered, idempotent semiring and let $X \subseteq Y \subseteq A$ with $|X| < \infty$ and $|Y| < \infty$. Then:

$$\bigoplus_{x \in X} x \leq \bigoplus_{y \in Y} y$$

Proof.
As long as $X \subseteq Y$ we have $\bigoplus\limits_{y \in Y} y = \bigoplus\limits_{x \in X} x \oplus \bigoplus\limits_{z \in Y-X} z$. In order to simplify the notation we can write $\alpha = \bigoplus\limits_{x \in X} x$ and $\beta = \bigoplus\limits_{z \in Y-X} z$.

In particular $\bigoplus\limits_{y \in Y} y = \alpha \oplus \beta$
Using that $\langle A, \oplus, \times \rangle$ is totally ordered, at least one of the following expressions must be satisfied:

- $\alpha \leq \beta$: Then $\alpha \oplus \beta = \beta$, so $\alpha \leq \beta = \bigoplus\limits_{y \in Y} y$
- $\beta \leq \alpha$: Then $\alpha \oplus \beta = \alpha$. Therefore, we have $\alpha = \alpha \oplus \beta = \bigoplus\limits_{y \in Y} y$. In particular $\alpha \leq \bigoplus\limits_{y \in Y} y$

□

From now on we will consider $\langle \Phi, D \rangle$ to be a valuation algebra on a modular valuable partition lattice defined over a totally ordered, idempotent semiring.

5.2 Solutions and Extensions

Given ϕ a valuation, the Action-GDL aim is finding the configuration of its domain which maximizes or minimizes ϕ.

In fact, Action-GDL computes $max_{\oplus}$ using the tropical or the arctic semiring depending on what are we looking for, i.e. we use $\oplus = min$ if we want the minimum configuration whereas we use $\oplus = max$ if we want to maximize the valuation.

Notice that finding the maximum configuration is the same that find a configuration x such that $\phi(x) = \max_{\otimes}\{\phi(x) : x \in d(\phi)\}$. Therefore, according to proposition 5.4 remark we have that finding the maximum value is equivalent to find $x \in d(\phi)$ such that

$$\phi(x) = \max_{\otimes}\{\phi(x) : x \in X\} = \bigoplus_{x \in d(\phi)} \phi(x) = \phi^{\downarrow \perp}(\diamond)$$

On the other hand, compute the projection from $d(\phi)$ to $\perp$ could be so difficult due to $d(\phi)$ size. Therefore, we will use the projection axiom of valuation algebras in order to compute ϕ from $\phi^{\downarrow \perp}$ with some intermediate steps which we will call extensions.

In this section we are going to define the notion of extension sets and the notion of solution, which will allow us to do the process defined above.

Definition 5.6.
Given ϕ a valuation, $s \in D$ such that $s \leq d(\phi)$ and $\beta \in s$ we define the *set of extensions of β with respect to ϕ* as:

$$W_\phi^s(\beta) = \{b \in (\beta \Uparrow d(\phi)) : \phi(b) = \phi^{\downarrow s}(\beta)\}$$

Lemma 4.
Let ϕ be a valuation and $s \in D$ such that $s \leq d(\phi)$. Let $\beta \in s$ and $b \in (\beta \Uparrow d(\phi))$. Then, for all u such that $s \leq u \leq d(\phi)$ it holds:

$$\phi(b) = \phi^{\downarrow u}(b_u) = \phi^{\downarrow s}(\beta) \;\Leftrightarrow\; \phi(b) = \phi^{\downarrow s}(\beta)$$

Proof.
It's clear that $\phi(b) = \phi^{\downarrow u}(b_u) = \phi^{\downarrow s}(\beta) \;\Rightarrow\; \phi(b) = \phi^{\downarrow s}(\beta)$.

Therefore we only have to prove $\phi(b) = \phi^{\downarrow s}(\beta) \Rightarrow \phi(b) = \phi^{\downarrow u}(b_u) = \phi^{\downarrow s}(\beta)$

We must remark that $b \in (\beta \Uparrow d(\phi)) \Rightarrow b \subseteq \beta$, moreover, as long as $s \leq u \leq d(\phi)$, we obtain $b \subseteq b_u \subseteq b_s = \beta$. Thus, in particular $\{b_u\} \subseteq (\beta \Uparrow u)$.

Using that fact, we can secure that $b_u \Uparrow d(\phi) \subseteq (\beta \Uparrow u) \Uparrow d(\phi)$, obtaining:

$$\phi^{\downarrow u}(b_u) = \bigoplus_{a \in b_u \Uparrow d(\phi)} \phi(a) \leq \bigoplus_{a \in (\beta \Uparrow u) \Uparrow d(\phi)} \phi(a) = \phi^{\downarrow s}(\beta) = \phi(b)$$

But as long as $b \in (b_u \Uparrow s)$ we have $\bigoplus_{a \in (b_u \Uparrow d(\phi))} \phi(a) \geq \phi(b)$, hence $\phi(b) = \phi^{\downarrow u}(b_u)$.

□

The following theorem shows that the extension from a block β to a bigger domain $d(\phi)$ can be made in two steps: first we extend β to some domain u bigger than β's domain but smaller than $d(\phi)$, and then we extend the elements of this extension to $d(\phi)$.

Theorem 5.7.
Let ϕ be a valuation, $s \in D$ such that $s \leq d(\phi)$ and $\beta \in s$. Then, for all $u \in D$ such that $s \leq u \leq d(\phi)$ it holds:

$$W_\phi^s(\beta) = \{b \in (\beta \Uparrow d(\phi)) : b_u \in W_{\phi^{\downarrow u}}^s(\beta) \text{ and } b \in W_\phi^u(b_u)\}$$

Proof.
Using definition 5.6 we have that

$$W_{\phi \Downarrow u}^s(\beta) = \{\alpha \in (\beta \Uparrow u) : \phi^{\downarrow u}(\alpha) = \phi^{\downarrow u \downarrow s}(\beta)\}$$

$$W_\phi^u(b_u) = \{\alpha \in \{b_u \Uparrow d(\phi)\} : \phi(\alpha) = \phi^{\downarrow u}(b_u)\}$$

In particular:

$$\alpha \in W_{\phi^{\downarrow u}}^s(\beta) \Leftrightarrow \phi^{\downarrow u}(\alpha) = \phi^{\downarrow s}(\beta)$$

$$\alpha \in W_\phi^u(b_u) \Leftrightarrow \phi(\alpha) = \phi^{\downarrow u}(\alpha_u)$$

.

Hence, we obtain:

$$\{b \in (\beta \Uparrow d(\phi)) : b_u \in W^s_{\phi^{\downarrow u}}(\beta) \textit{ and } b \in W^u_\phi(b_u)\} =$$
$$\{b \in (\beta \Uparrow d(\phi)) : \phi^{\downarrow u}(b_u) = \phi^{\downarrow s}(\beta) \textit{ and } \phi(b) = \phi^{\downarrow u}(b_u)\} =$$
$$\{b \in (\beta \Uparrow d(\phi)) : \phi(b) = \phi^{\downarrow u}(b_u) = \phi^{\downarrow s}(\beta)\} = W^s_\phi(\beta)$$

Where the last equality is due to lemma 4

□

Definition 5.8. Given ϕ a valuation, we define it's solution set as follows:

$$c_\phi = W^\perp_\phi(\diamond)$$

Remark

$$c_\phi = W^\perp_\phi(\diamond) = \{b \in (\diamond \Uparrow d(\phi)) : \phi(b) = \phi^{\downarrow\perp}(\diamond)\} = \{b \in d(\phi) : \phi(b) = \phi^{\downarrow\perp}(\diamond)\} =$$
$$\{b \in d(\phi) : \phi(b) = \phi^{\downarrow s}(b_\pi) = \phi^{\downarrow\perp}(\diamond)\} \textit{ for all } s \le d(\phi)$$

Lemma 5.
Let ϕ be a valuation and $s \in D$ such that $s \le d(\phi)$
Then:

$$c_{\phi^{\downarrow s}} = (c_\phi) \Downarrow s$$

Proof.
Given $\alpha \in c_\phi \Downarrow s$, then, for all $a \in (\alpha \Uparrow d(\phi))$ it holds $\phi(a) = \phi^{\downarrow s}(a_s) = \phi^{\downarrow\perp}(\diamond) = \phi^{\downarrow s \downarrow\perp}(\diamond)$. In particular $\alpha = a_s \in c_{\phi^{\downarrow s}}$

On the other side, $\beta \in c_{\phi^{\downarrow s}} \Rightarrow \phi^{\downarrow s}(\beta) = \phi^{\downarrow s\downarrow\perp}(\diamond) = \phi^{\downarrow\perp}(\diamond)$.

Aditionally $\phi^{\downarrow s}(\beta) = \bigoplus\limits_{b \in (\beta \Uparrow d(\phi))} \phi(b) = max_\otimes\{\phi(b) : b \in (\beta \Uparrow d(\phi))\}$.

By proposition 5.4 we can secure that it exist some $b \in (\beta \Uparrow d(\phi))$ such that $\phi(b) = \phi^{\downarrow s}(\beta) = \phi^{\downarrow\perp}(\diamond)$. Then, as long as $\beta = b_s$ we obtain $\beta \in ((c_\phi) \Downarrow s)$ □

5.3 Optimization on General Valuation Algebras

Notice that given a factorized valuation $\phi = \phi_1 \otimes \cdots \otimes \phi_n$ the collect algorithm allows us to compute $\phi^{\downarrow\lambda(r)}$, where r denotes the factor linked with the root node of the optimized covering join tree (see section 3.2).

As long as $\lambda(r)$ is supposed to be small enough, we are able to compute $\phi^{\downarrow(\perp)}$ by projecting $\phi^{\downarrow\lambda(r)}$ to $\perp$. Therefore we know which is the best value that ϕ can take.

In this section we will show and give different algorithms in order to compute elements $b \in d(\phi)$ such that $b \in c_\phi$. Notice that if we work with an idempotent induced valuation algebra it is equivalent to compute elements such that $\phi(b)$ is the best, according to the canonical order defined in the semiring.

In order to do it we will suppose that we work with a valuation algebra $\langle\Phi, D\rangle$ defined on a modular valuable Partition Lattice such that the semiring that induces it is a totally ordered, idempotent and cancellative semiring $R = \langle A, \oplus, \times\rangle$.

The following theorem shows us that given a factorized valuation ϕ and a domain $u \leq d(\phi)$, we can compute $c_{(\phi^{\downarrow u})}$ and extend its elements to $d(\phi)$ in order to obtain the whole solution set.

Theorem 5.9.
Let $\phi \in \Phi$ such that there are $\phi_s, \phi_t \in \Phi$ with domains s and t such that $\phi = \phi_s \otimes \phi_t$.
Then, for all $u \in D$ such that $u \wedge t \leq s \leq u \leq d(\phi)$ we have:

$$c_\phi = \{b \in (s \vee t) : b_u \in c_{(\phi^{\downarrow u})} \textit{ and } b_t \in W^{u \wedge t}_{\phi^{\downarrow t}}(b_{u\wedge t})\}$$

Proof.

Set $m := \phi^{\downarrow \perp}(\diamond)$, and u such that $u \wedge t \leq s \leq u \leq d(\phi)$.

First we will prove that $c_\phi \subseteq \{b \in (s \vee t) : b_u \in c_{(\phi^{\downarrow u})} \textit{ and } b_t \in W^{u \wedge t}_{\phi^{\downarrow t}}(b_{u\wedge t})\}$

Given $b \in c_\phi$ we can guarantee that $\phi^{\downarrow u}(b_u) = m$ for all $u \leq d(\phi)(= (s \vee t))$ due the remark of definition 5.8. In particular we obtain:

- $\phi^{\downarrow s}(b_s) = m$
- $\phi^{\downarrow u}(b_u) = m \Rightarrow b_u \in c_{(\phi^{\downarrow u})}$
- $\phi^{\downarrow t}(b_t) = m$ and $\phi^{\downarrow (u \wedge t)}(b_{u \wedge t}) = m \Rightarrow \ \phi^{\downarrow t}(b_t) = \phi^{\downarrow (u \wedge t)}(b_{u \wedge t}) \ \Rightarrow \ b_t \in W^{u \wedge t}_{\phi^{\downarrow t}}(b_{u\wedge t})$

So in order to complete the proof we must demonstrate the opposite inclusion

Given $b \in \{b \in (s \vee t) : b_u \in c_{(\phi^{\downarrow u})} \textit{ and } b_t \in W^{u \wedge t}_{\phi^{\downarrow t}}(b_{u\wedge t})\}$ we have:

- $b_u \in c_{(\phi^{\downarrow u})} \ \Rightarrow \phi^{\downarrow u}(b_u) = (\phi^{\downarrow u})^{\downarrow \perp}(\diamond) = \phi^{\downarrow bot}(\diamond) = m$

  $\Rightarrow b_{u \wedge t} \in c_{(\phi^{\downarrow u \wedge t})} \Rightarrow (\phi^{\downarrow u \wedge t})(b_{u \wedge t}) = m$
- $b_t \in W^{u \wedge t}_{\phi^{\downarrow t}}(b_{u\wedge t}) \Rightarrow \phi^{\downarrow t}(b_t) = \phi^{\downarrow u \wedge t}(b_{u \wedge t}) = m$

Then using the combination axiom (A5)we obtain:

$m = \phi^{\downarrow u}(b_u) = (\phi_s \otimes \phi_t)^{\downarrow u}(b_u) = (\phi_s \otimes (\phi_t^{\downarrow u \wedge t}))(b_u) = \phi_s(b_s) \times \phi_t^{\downarrow u \wedge t}(b_{u \wedge t})$
$m = \phi^{\downarrow t}(b_t) = (\phi_u \otimes \phi_t)^{\downarrow t}(b_t) = (\phi_t \otimes (\phi_s^{\downarrow u \wedge t}))(b_t) = \phi_t(b_t) \times \phi_s^{\downarrow u \wedge t}(b_{u \wedge t})$

Finally, using the associative and commutative property of $\times$ we obtain:

$m \times m = \phi_s(b_s) \times \phi_t^{\downarrow u \wedge t}(b_{u \wedge t}) \times \phi_t(b_t) \times \phi_s^{\downarrow u \wedge t}(b_{u \wedge t}) = (\phi_s(b_s) \times \phi_t(b_t)) \times (\phi_t^{\downarrow u \wedge t}(b_{u \wedge t}) \times \phi_s^{\downarrow u \wedge t}(b_{u \wedge t})) = \phi(b) \times \phi^{\downarrow u \wedge t}(b_{u \wedge t}))$ due to proposition. 2.3

Hence $m \times m = \phi(b) \times \phi^{\downarrow u \wedge t}(b_{u \wedge t})) = \phi(b) \times m$

Recall that we supposed that we work on a valuation algebra induced by an idempotent totally ordered and cancellative semiring, thereby we obtain $m = \phi(b)$ due to the cancellativity. In particular, $b \in c_\phi$

□

Recall that both, arctic and tropical semiring, are cancellative, therefore, as long as the theorem above works for modular domains instead of boolean domains we will consider that it is a generalized and corrected version of theorem 8.1 in [8].

As it is done in [8] with theorem 8.1, we will use theorem 5.9 to create all the desired algorithms. In order to do that we will suppose that we are given an optimized covering join tree for a fixed factorization $\phi_1 \otimes \dots \phi_r$

Corollary 5.10.
For $i = 1, ..., r-1$ and $s = \lambda(i+1) \vee \dots \vee \lambda(r)$ it holds:

$$c_\phi^{\downarrow s\vee\lambda(i)} = \{b \in s \vee \lambda(i) : \; b_s \in c_\phi^{\downarrow s} \; and \; b_{\lambda(i)} \in W_{\phi^{\downarrow\lambda(i)}}^{sep(i)}(b_{sep(i)})\}$$

Proof.
By Theorem 2.6 we have $c_\phi^{\downarrow s\vee\lambda(i)} = c_{(\phi^{\downarrow s\vee\lambda(i)})} = c_{(\phi_i^{(i)} \otimes \phi_{i+1}^{(i)} \otimes \dots \otimes \phi_r^{(i)})} = c_{((\phi_i^{(i)}) \otimes (\phi_{i+1}^{(i)} \otimes \dots \otimes \phi_r^{(i)}))}$

Using the proposition 3.9 we obtain $d((\phi_i^{(i)})) = \lambda(i)$. Therefore, if we take $s = d(\phi_{i+1}^{(i)} \otimes \dots \otimes \phi_r^{(i)})$ $\lambda(i+1) \vee \dots \vee \lambda(r) = u$ and $t = \lambda(i)$ we have that $u \wedge t = s \wedge t \leq s \leq u \leq s \vee t$. Thus we can apply the previous theorem obtaining:

$$c_\phi^{\downarrow s\vee\lambda(i)} = \{b \in (s \vee t) : b_u \in c_{(\phi^{\downarrow u})} \; and \; b_t \in W_{\phi^{\downarrow t}}^{u\wedge t}(b_{u\wedge t})\}$$

Finally, as long as $i < pa(i)$ we can apply the running intersection property and obtain:

$$u \wedge t = s \wedge t = s \wedge \lambda(i) = \lambda(pa(i)) \wedge \lambda(i) = sep(i)$$

□

Recall that the collect algorithm allows us to know $\phi^{\downarrow\lambda(r)}$. As long as for any given tree one can fix any node as root node it makes sense to think that one compute $\phi^{\downarrow\lambda(i)}$ for $i = 1, \dots, r$ by running the collect algorithm r times. Nevertheless a multi-query procedure was created in order to do that using a single covering join tree.

Although in [8] this procedure, called collect-distribute algorithm, is only described on valuation algebras with boolean domains, in [11] it is given a generalization to modular domains. Therefore we can use the corollary above to compute all the solutions with the following algorithm:

1. Execute the collect-distribute algorithm on $\{\phi_1, \dots, \phi_n\}$
2. Identify $c_\phi^{\downarrow\lambda(r)}$ in the root node.
3. For $i = r-1, \dots, 1$
   a) Compute $W_{\phi^{\downarrow\lambda(i)}}^{sep(i)}$ in node i
   b) Build $c_\phi^{\downarrow\lambda(r)\vee\dots\vee\lambda(i)}$ by application of Corollary 5.10
4. Return $c_\phi = c_\phi^{\downarrow\lambda(r)\vee\dots\vee\lambda(1)}$

This procedure allows us to create the following algorithm:

Algorithm 1. *Computing all solutions with distribute*

```
input: $\phi^{\downarrow\lambda(r)}$ for $i = 1, \dots, r$
begin:
    $c := W^{\perp}_{\phi^{\downarrow\lambda(r)}}(\diamond)$
    for $i = r-1, \dots, 1$ do
        $s := \lambda(r) \vee \cdots \vee \lambda(i+1)$
        $c := \{b \in s \vee \lambda(i) :\ b_s \in c_\phi^{\downarrow s}$ and $b_{\lambda(i)} \in W^{sep(i)}_{\phi^{\downarrow\lambda(i)}}(b_{sep(i)})\}$
    endfor
    return $c$
end
```

Nevertheless we want to be able to compute solutions using only the collect algorithm instead of the Collect-Distribute algorithm. Moreover, in most cases we will be interested on finding a single solution. Next we will bring some results in order to do it, and finally we will show the algorithms without distribute phase.

Theorem 5.11. *If the configuration extension sets satisfy that for all $\phi_s, \phi_t \in D$ with domains s and t such that $t \leq s$ it holds that $W^t_{\phi_s}(x) \subseteq W^t_{\phi_s \otimes \phi_t}(x)$*
Then for $i = 1, ..., r-1$ and $s = \lambda(i+1) \vee \cdots \vee \lambda(r)$ it holds:

$$c_\phi^{\downarrow s \vee \lambda(i)} \supseteq \{b \in s \vee \lambda(i) :\ b_s \in c_\phi^{\downarrow s} \text{ and } b_{\lambda(i)} \in W^{sep(i)}_{\phi_i^{(r)}}(b_{sep(i)})\}$$

Proof.
The previous theorem tells us that $c_\phi^{\downarrow s \vee \lambda(i)} = \{b \in s \vee \lambda(i) :\ b_s \in c_\phi^{\downarrow s} \text{ and } b_{\lambda(i)} \in W^{sep(i)}_{\phi^{\downarrow\lambda(i)}}(b_{sep(i)})\}$.

Pouly and Kohlas proved in [8] that $\phi^{\downarrow\lambda(i)} = \phi_i^{(r)} \otimes \psi^{\downarrow sep(i)}$ due to the the message-passing in distribute algorithm.

As long as $sep(i) \leq \lambda(pa(i)) \leq s$ we obtain $W^{sep(i)}_{\phi^{\downarrow\lambda(i)}}(b_{sep(i)}) \supseteq W^{sep(i)}_{\phi_i^{(r)}}(b_{sep(i)})$ due to the theorem hypothesis.

□

Proposition 5.12.
Let (Φ, D) is a valuation algebra defined on a modular valuable partition lattice over a totally ordered, idempotent,cancellative semiring. Then for all $\phi_s, \phi_t \in D$ with domains s and t such that $t \leq s$ it holds that

$$W^t_{\phi_s}(x) \subseteq W^t_{\phi_s \otimes \phi_t}(x)$$

Proof.
Fixed $x \in t$ we know that
$W^t_{\phi_s}(x) = \{b \in (x \Uparrow t) : \phi_s(b) = \phi_s^{\downarrow t}(x)\}$
$W^t_{\phi_s \otimes \phi_t}(x) = \{b \in (x \Uparrow t) : (\phi_s \otimes \phi_t)(b) = (\phi_s \otimes \phi_t)^{\downarrow t}(x)\}$

Then, given $b \in W^t_{\phi_s}(x) \subseteq s$ it holds that:

$$(\phi_s \otimes \phi_t)(b) = \phi_s(b) \times \phi_t(x_t) = \phi_s(b) \times \phi_t(x_t) = \phi_s^{\downarrow t}(x) \times \phi_t(x_t) = (\phi_s^{\downarrow t} \otimes \phi_t)(x) = (\phi_s \otimes \phi_t)^{\downarrow t}(x)$$
□

Therefore, the proposition above allows us to use theorem 5.11 in order to compute some solutions after using the collect algorithm. This can be done with the following procedure:

1. Execute the collect algorithm on $\{\phi_1, \ldots, \phi_n\}$
2. Compute $c_\phi^{\downarrow\lambda(r)}$ in the root node.
3. For $i = r-1, \ldots, 1$
   a) Compute $W_{\phi_i^{(r)}}^{sep(i)}$ in node i
   b) Build a subset of $c_\phi^{\downarrow\lambda(r)\vee\cdots\vee\lambda(i)}$ by application of Corollary 5.11
4. Return the subset of$c_\phi = c_\phi^{\downarrow\lambda(r)\vee\cdots\vee\lambda(1)}$

Notice that some solutions can mean a single one, therefore, by using the procedure above we can obtain two algorithms. One of them will compute some solutions while the other one will compute a single solution (as Action-GDL does).

Algorithm 2. *Computing some solutions without distribute*

```
    input: $\phi_i^{(r)}$ for $i = 1, \ldots, r$
    begin:
        $c := W_{\phi_r^{(r)}}^{\perp}(\diamond)$
        for $i = r-1, \ldots, 1$ do
            $s := \lambda(r) \vee \cdots \vee \lambda(i+1)$
            $c := \{b \in s \vee \lambda(i) : \ b_s \in c_\phi^{\downarrow s}$ and $b_{\lambda(i)} \in W_{\phi_i^{(r)}}^{sep(i)}(b_{sep(i)})\}$
        endfor
        return $x$
    end
```

It is important to notice that if $b \in W_{\phi_s\otimes\phi_t}^t(x)$ then $\phi_s(b) \times \phi_t(x_t) = (\phi_s \otimes \phi_t)^{\downarrow t}(x) = \phi_s^{\downarrow t}(x) \times \phi_t(x_t)$.

Thus, if $\phi_t(x_t)$ is invertible for all x then we do not have an inclusion but an equality in theorem 5.11 and the following proposition. Therefore algorithm 2 will compute all solutions instead of some solutions.

To finish the content of the dissertation we show the Action-GDL generalization to modular lattices:

Algorithm 3. *Computing one solutions without distribute (General Action-GDL)*

```
    input: $\phi_i^{(r)}$ for $i = 1, \ldots, r$
    begin:
        choose $x \in W_{\phi_r^{(r)}}^{\perp}(\diamond)$
        for $i = r-1, \ldots, 1$ do
            $s := \lambda(r) \vee \cdots \vee \lambda(i+1)$
            choose $x \in \{b \in s \vee \lambda(i) : \ b_s \in c_\phi^{\downarrow s}$ and $b_{\lambda(i)} \in W_{\phi_i^{(r)}}^{sep(i)}(b_{sep(i)})\}$
        endfor
        return $c$
    end
```

6 Conclusions and future work

Along this dissertation we have introduced a new valuation algebra on partition lattices and we have proved that some algorithms related to valuation algebras hold there. This leads us to several original results:

- We proved that the only requirement for defining a (good enough) valuation algebra is a modular domain in the second chapter.
- On the third chapter we generalized the collect algorithm from distributive domains to modular domains.
- Although optimized covering join trees have been already used by engineers and statisticians ,to the best of our knowledge they have never been formalized. We bring a formal description of these trees on definition 3.6
- During chapter 4 we defined the raise and decrease in different environments, such as blocks, block sets and multisets; which leaded us to define a new valuation algebra, which can be used as the most general valuation algebra over semirings (see theorem 4.24).
- In chapter five we generalized some algorithms to some specific valuation algebra (we ask for idempotent, totally ordered and cancellative semirings). Moreover, taking into account that previous algorithms do use valuation algebras with boolean domains we were greatly surprised that we can generalize the algorithms to modular domains.

In addition, it is important to remark that we found a wrong theorem in [8] (theorem 8.1 or theorem 1 in [7]), as long as a theorem with an incomplete proof (theorem 8.4):

- Theroem 8.1 has been generalized and fixed in this dissertation (see theorem 5.9).
- Theorem 8.4 has been generalized and its proof has been completed using lemma 4 (see lemma 4 and theorem 5.7 in chapter 5).

On the other side there are still several chapter and results in [3, 8] that might be generalized.

Moreover, there are also some generalized results in chapter 5 of this dissertation that can be improved. In particular, we would like to proof theorem 5.9 without using the cancellative property, obtaining a better generalization. Furthermore, we believe that it would be possible to prove the results in section 5.3 using the property of extensions defined in theorem 5.7. In that case we should be able to generalize the results in [7].

In addition, we think that it is possible to generalize the results in chapters 4 and 5 to partition lattices with infinite universes if we use an adequate semiring.

Finally, in order to complete all the aims of the dissertation we would like to show how to embed the valuations in example 1, the valuations used in CSGP and the set-based valuation algebras described in [6] in the generalized valuations described on chapter 4, proving that we have created a wider unified framework.

Appendix A: A brief summary on lattices

Definition A.1.
A *partial order* is a binary relation $\leq$ over a set P such that:

1. $a \leq a,\ \forall a \in P$ ($\leq$ is reflexive)
2. $a \leq b$ *and* $b \leq c \ \Rightarrow\ a \leq c\ \forall a, b, c \in P$ ($\leq$ is transitive)
3. $a \leq b$ *and* $b \leq a \ \Rightarrow\ a = b\ \forall a, b \in P$ ($\leq$ is antisymmetric)

An *ordered set* is a tuple $(P, \leq)$ where P is a set and $\leq$ is a partial order over P. We can also denote it by P.

Definition A.2.
A *partial order* over a set P is called total order if for all $x, y \in P$ it holds $x \leq y$ or $y \leq x$

A *totally ordered set* is a tuple $(P, \leq)$ where P is a set and $\leq$ is a total order over P. We can also denote it by P.

Definition A.3.
Let P be an ordered set.

- P has a *bottom element* if there exists an element $\bot \in P$ such that $\bot \leq a\ \forall a \in P$.
- P has a *top element* if there exists an element $\top \in P$ such that $x \leq \top\ \forall a \in P$.

Definition A.4.
Let P be an ordered set and $S \subseteq P$.

- An element $x \in P$ is called *supremum, least upper bound* or *join of S* if:
  1. $a \leq x\ \forall a \in S$.
  2. for any $y \in P$ such that $a \leq y\ \forall a \in S$ it holds that $x \leq y$.
- An element $x \in P$ is called *infimum, greatest lower bound* or *meet of S* if:
  1. $x \leq a\ \forall a \in S$.
  2. for any $y \in P$ such that $y \leq a\ \forall a \in S$ it holds that $y \leq x$.

If supremum or infimum of a subset $S \subseteq P$ exist, then they are always unique and we write $\bigvee S$ for supremum and $\bigwedge S$ for infimum .

Moreover, if $S = \{a, b\}$ consists of two elements, we generally write $a \wedge b$ or $\sup\{a, b\}$ for supremum and $a \vee b$ or $\inf\{a, b\}$ for infimum.

Definition A.5.
Let $\mathcal{L}$ be a non-empty ordered set. Then $L = (\mathcal{L}; \vee, \wedge)$ is called a *lattice* if $a \wedge b$ *and* $a \vee b$ exist $\forall a, b \in P$.

Example:
Let X be a set, then $(\mathcal{P}(X), \subseteq)$ is a lattice with $\bot = \emptyset$, $\top = X$, $a \vee b = a \cup b$ and $a \wedge b = a \cap b$

Definition A.6.
A lattice $K = (\mathcal{K}; \vee_k \wedge_k)$ is said to be a *sublattice* of the lattice $L = (\mathcal{L}; \vee_L, \wedge_L)$ if

- $a, b \in \mathcal{K} \Rightarrow a \vee_L b, a \wedge_L b \in \mathcal{K}$
- $\vee_K$ and $\wedge_K$ are the restrictions to $\mathcal{K}$ of $\vee_L$ and $\wedge_L$ respectively.

In that case we will write $K \subseteq L$

Definition A.7.
Let L be a lattice.

- L is said to be *bounded* if it has a bottom and top element.
- L is called a *complete lattice* if $\bigwedge S$ *and* $\bigvee S$ exist $\forall S \subseteq P$.
- L is said to be *modular* if for all $x, w, y \in L$ such that $x \leq y$ we have:

$$x \vee (w \wedge y) = (x \vee w) \wedge y$$

- L is said to be *distributive* if for all $a, b, c \in L$:

$$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$$
$$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$$

Remark: If a lattice is distributive then it is also modular.

Proposition A.8.
Let L be a lattice and $K \subseteq L$ a sublattice. Then:

$$K \text{ complete} \Rightarrow L \text{ complete}$$

Proposition A.9.
Let L be a lattice and $K \subseteq L$ a sublattice. Then:

$$K \text{ modular} \Rightarrow L \text{ modular}$$

Proposition A.10.
Let L be a lattice and $K \subseteq L$ a sublattice. Then:

$$K \text{ distributive} \Rightarrow L \text{ distributive}$$

Proposition A.11. *Let L be a lattice. Then L is modular if and only if*

$$x \leq z \leq x \vee y \Rightarrow x \vee (y \wedge z) = z \textit{ for all } x, y, x \in L$$

Proof. Suppose that L is modular, and $x, y, z \in L$ such that $x \leq z \leq x \vee y$
Using $x \leq z$ and the definition of modular we obtain $x \vee (y \wedge z) = (x \vee y) \wedge z$.
As long as $z \leq x \vee y$ we have $(x \vee y) \wedge z = z$.

Now suppose that $x \vee (y \wedge z) = z$ for all $x, y, x \in L$ such that $x \leq z \leq x \vee y$.
Given $x, y, w \in L$ such that $x \leq y$, we set $z = (x \vee w) \wedge y$.

Due to $\vee$ definition we know that $z \leq x \vee w$. Equivalently we obtain $x \leq y \leq z$ using $\wedge$ definition, so in particular $x \leq z \leq x \vee w$

Using the hypothesis above we obtain $x \vee (w \wedge z) = z = (x \vee w) \wedge y$
Furthermore, since $w = w \wedge (x \vee w)$,we obtain $x \vee (w \wedge y) = x \vee ((w \wedge (x \vee w)) \wedge y) = x \vee (w \wedge ((x \vee w) \wedge y)) = x \vee (w \wedge z) = z = (x \vee w) \wedge y$

□

Definition A.12.
Two lattices $L = (\mathcal{L}, \vee_L, \wedge_L)$ and $K = (\mathcal{K}, \vee_K, \wedge_K)$ are said to be *isomorphs* if it exist a bijective function $\phi : \mathcal{L} \to \mathcal{K}$ such that:

$$\phi(a \vee_L b) = \phi(a) \vee_K \phi(b) \text{ for all } a, b, \in \mathcal{L}$$
$$\phi(a \wedge_L b) = \phi(a) \wedge_K \phi(b) \text{ for all } a, b, \in \mathcal{L}$$

In that case we will write $L \cong K$ and the function ϕ is called an *isomorphism.*

Remark: Every isomorphism is a bijective $\vee$-homomorphism and $\wedge$-homomorphism.

Definition A.13.
We say that a lattice D can be *embedded in* a lattice L if there is a sublattice $K \subseteq L$ such that $D \cong K$.

Theorem A.14. ***Grätzer 2011** [2]*
For every finite lattice L there is a finite set X such that L can be embedded in $Part(X)$ [1]

Bibliography

[1] G. Grätzer. *General Lattice Theory.* Academic Press, 1978.

[2] G. Grätzer. *Lattice Theory: Foundation.* Springer, 2011.

[3] J. Kohlas. *Information Algebras: Generic Structures For Inference.* Springer, 2003.

[4] J. Kohlas and N. Wilson. Semiring induced valuation algebras. *Artificial Intelligence*, 172(11):1360–1399, July 2008.

[5] P.J. Modi, W.M. Shen, M. Tambe, and M. Yokoo. Adopt: Asynchronous distributed constraint optimization with quality guarantees. *ARTIFICIAL INTELLIGENCE*, 161:149–180, 2006.

[6] M. Pouly. *A Generic Framework for Local Computation.* PhD thesis, Department of Informatics, University of Fribourg, 2008.

[7] M. Pouly. Generic solution construction in valuation-based systems. In *Proceedings of the 24th Canadian conference on Advances in artificial intelligence*, Canadian AI'11, pages 335–346. Springer-Verlag, 2011.

[8] M. Pouly and J. Kohlas. *Generic Inference - A unifying Theory for Automated Reasoning.* John Wiley & Sons, Inc., 2011.

[9] R.Mailler and V.Lesser. Solving distributed constraint optimization problems using cooperative mediation. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 438–445, 2004.

[10] C. Schneuwly, M. Pouly, and J. Kohlas. Local computation in covering join trees. Technical Report 04-16, Department of Informatics, University of Fribourg, 2004.

[11] G. Shafer. An axiomatic study of computation in hypertrees, 1991.

[12] M. Vinyals, J.A. Rodriguez-Aguilar, and J. Cerquides. Constructing a unifying theory of dynamic programming DCOP algrithms via the generalized distributive law. *Autonomous Agents and Multi-Agent Systems*, (22):439–464, May 2010.

[13] T. Voice, M. Polukarov, and N. R. Jennings. Graph coalition structure generation. *CoRR*, abs/1102.1747, 2011.