# Subtyping constraints in quasi-lattices 

Emmanuel Coquery and François Fages<br>INRIA Rocquencourt, Projet Contraintes, BP 105, 78153 Le Chesnay Cedex, FRANCE


#### Abstract

In this paper, we show the decidability and NP-completeness of the satisfiability problem for non-structural subtyping constraints in quasi-lattices. This problem, introduced by Smolka in 1988, is important for the typing of logic and functional languages. We generalize Trifonov and Smith's algorithm over lattices, to the case of quasi-lattices with a complexity in $O\left(m^{v} M^{v} n^{3}\right)$, where $m$ (resp. $M$ ) stands for the number of minimal (resp. maximal) elements of the quasi-lattice, $v$ is the number of unbounded variables and $n$ is the number of constraints. Similarly, we extend Pottier's algorithm for computing explicit solutions to the case of quasi-lattices. Finally we evoke some applications of these results to type inference in constraint logic programming and functional programming languages.


## 1 Introduction

The search for more and more flexible type systems for programming languages goes with the search for algorithms for solving subtyping constraints in more and more complex type structures. Type checking and type inference algorithms for a program basically consist in solving systems of subtyping constraints of the form $\exists X \bigwedge_{i=1}^{n} t_{i} \leq t_{i}^{\prime}$ where $t_{i}, t_{i}^{\prime}$ are types and $X$ is the set of variables appearing in the system.

In its most general form combining subtyping and parametric polymorphisms, non-structural subtyping allows subtyping relations between type constructors of different arities. For instance, in the type system for constraint logic programming TCLP [3], the subtyping relation list $(\alpha) \leq$ term allows us to see a (homogeneous) list as a term. In a lattice of type constructors, Trifonov and Smith [10] gave a simple decomposition algorithm, with a complexity in $O\left(n^{3}\right)$, for testing the satisfiability of non-structural subtyping constraints in the lattice of infinite or regular types. Pottier [7] extended this algorithm to compute solutions explicitly when they exist. However, the lattice structure of type constructors imposes the existence of a minimal element $\perp$ and a maximal element $\top$, and thus does not treat the typing with the empty type $\perp$ as an error.

In this paper, we are interested in the resolution of non-structural subtyping constraints in more general structures than lattices, especially in the case where the type constructors form a complete quasi-lattice, that is a partially ordered set for which any non-empty subset having a lower bound (resp. an upper bound) has a greatest lower bound (resp. least upper bound). The decidability of nonstructural subtyping constraints satisfiability in quasi-lattices is an open problem
mentioned in Smolka's thesis [9]. In this paper, we bring a positive answer to this problem by generalizing Trifonov and Smith's algorithm to quasi-lattices of non-atomic types, and we prove the NP-completeness of this problem.

The rest of the paper is organized as follows. In the next section, we define the ordered set of infinite types formed upon a quasi-lattice of type constructors of different arities, and we prove that this set is a quasi-lattice. In section 3, we show that in quasi-lattices, the systems closed by Trifonov and Smith's decomposition rules are satisfiable, and we give an algorithm for testing the satisfiability of subtyping constraints with a time complexity in $O\left(m^{v} M^{v} n^{3}\right)$, where $m$ (resp. $M$ ) stands for the number of minimal (resp. maximal) elements of the quasi-lattice, $v$ is the number of unbounded variables and $n$ is the number of constraints. The NP-completeness of constraint satisfiability is shown in this section by using the result of Pratt and Tiuryn for n-crowns [8]. In section 4, we generalize Pottier's algorithm for computing explicit solutions in quasi-lattices. Section 5 presents some applications of these results to type checking and we conclude in the last section.

Note to the reviewers: the proofs which do not appear in the main text are given in the appendix.

## 2 Infinite types

### 2.1 Preliminaries

Let $(E, \leq)$ be a partially ordered set, and $S$ be a subset of $E$. We note $\downarrow S=\{x \in$ $E \mid S \neq \emptyset, \forall y \in S x \leq y\}$ the set of lower bounds of $S$ and $\uparrow S=\{x \in E \mid S \neq$ $\emptyset, \forall y \in S y \leq x\}$ the set of upper bounds of $S$. We note $\sqcap S$ (resp. $\sqcup S$ ) the greatest lower bound (resp. least upper bound) of $S$ whenever it exists. A lower quasi-lattice (resp. upper quasi-lattice) is a partially ordered set where any finite subset having a lower (resp. upper) bound has a greatest lower bound (resp. a least upper bound). A quasi-lattice is an upper and a lower quasi-lattice.

Definition 1 (Complete quasi-lattice). A partially ordered set is a complete quasi-lattice (in the sense of sets) if for all non empty subset $S \subseteq E, \sqcap S$ exists whenever $\downarrow S \neq \emptyset$ and $\sqcup S$ exists whenever $\uparrow S \neq \emptyset$.

### 2.2 Labels

As mentionned in the introduction, we are interested in type languages allowing subtyping relations between type constructors of different arities, like list $(\alpha) \leq$ term for instance. In general, such subtyping relations specify subtyping relations between specific arguments of the type constructors. For instance, by writing $k_{1}(\alpha, \beta) \leq k_{2}(\beta)$, we specify that types built with $k_{1}$ are subtypes of the ones built with $k_{2}$ when the second argument of $k_{1}$ and the argument of $k_{2}$ correspond, the first argument of $k_{1}$ being forgotten in the subtype relationship.

From a formal point of view, it is simpler (and more general) to express the relationship between arguments by working with a structure of labeled terms. In
the formalism of Pottier [6], each argument of a constructor is indicated by a label instead of a position. Moreover, positive and negative labels are distinguished in order to express the covariance or the contravariance of arguments w.r.t. the subtyping relation.

So let $\mathcal{L}^{+}$and $\mathcal{L}^{-}$be two disjoint countable sets of labels, we pose $\mathcal{L}=$ $\mathcal{L}^{+} \uplus \mathcal{L}^{-}$. Let $(\mathcal{K}, \leq \mathcal{K})$ be a complete quasi-lattice of type constructors. Let $a$ be the arity function defined from $\mathcal{K}$ into the finite parts of $\mathcal{L}$. We denote by $a^{+}$(resp. $a^{-}$) the function which associates the positive (resp. negative) labels to a constructor. We assume that there is at least one type constructor with an empty arity, $k_{0}$.
Definition 2. $\left(\mathcal{K}, \leq_{\mathcal{K}}, \mathcal{L}^{+}, \mathcal{L}^{-}, a\right)$ is a signature if:

1. for all $k_{1} \leq \mathcal{K} k_{2} \leq \mathcal{K} k_{3}, a\left(k_{1}\right) \cap a\left(k_{3}\right) \subseteq a\left(k_{2}\right)$.
2. for all $S \subseteq \mathcal{K}$, if $\sqcap S$ exists, then $a(\Pi S) \subseteq \bigcup_{s \in S} a(s)$.
3. for all $S \subseteq \mathcal{K}$, if $\sqcup S$ exists, then $a(\sqcup S) \subseteq \bigcup_{s \in S} a(s)$.
4. for all $k_{1} \leq \mathcal{K} k_{2}$, there exists $k$ s.t. $k_{1} \leq_{\mathcal{K}} k \leq \mathcal{K} k_{2}$ and $a(k)=a\left(k_{1}\right) \cap a\left(k_{2}\right)$.

Conditions 1, 2,3 express the coherence of labels w.r.t. the order relation and are similar to the ones found in [6] for lattices. Condition 4 is specific to quasilattices, its purpose is to forbid signatures like $k_{1}(\alpha) \leq \mathcal{K} k_{2}(\beta)$ which do not induce a quasi-lattice structure for types. For example, if $k_{3}$ and $k_{4}$ are not comparable, then $k_{2}\left(k_{3}\right)$ and $k_{2}\left(k_{4}\right)$ have common lower bounds, like $k_{1}\left(k_{3}\right)$ and $k_{1}\left(k_{4}\right)$, but don't have a greatest common upper bound.

For a signature $\left(\mathcal{K}, \leq \mathcal{K}, \mathcal{L}^{+}, \mathcal{L}^{-}, a\right)$, we note $\mathcal{L}^{*}$ the set of finite strings of labels, $\epsilon$ the empty string, "." the string concatenation and $|w|$ the length of $w$. We are interested in infinite types formed upon $\mathcal{K}$, where the positions of subterms are defined by strings of labels.
Definition 3. Let $\left(\mathcal{K}, \leq \mathcal{K}, \mathcal{L}^{+}, \mathcal{L}^{-}, a\right)$ be a signature. An infinite type is a partial mapping from $\mathcal{L}^{*}$ into $\mathcal{K}$ such that:

1. Its domain is prefix closed: $\forall w=w_{1} \cdot w_{2} \in \operatorname{dom}(t), w_{1} \in \operatorname{dom}(t)$.
2. $\epsilon \in \operatorname{dom}(t)$.
3. For all position $w \in \operatorname{dom}(t)$, for all $l \in \mathcal{L}$, w.l $\in \operatorname{dom}(t)$ if and only if $l \in a(t(w))$.
We note $\mathcal{T}(\mathcal{S})$ the set of infinite types built upon the signature $\mathcal{S}$. In the following, we assume a fixed signature $\mathcal{S}=\left(\mathcal{K}, \leq_{\mathcal{K}}, \mathcal{L}^{+}, \mathcal{L}^{-}, a\right)$. and we note $\mathcal{T}=\mathcal{T}(\mathcal{S})$. We note $t / w$ the type $t^{\prime}: v \mapsto(w . v)$. We note $U / l$, the set of subterms of types in $U \subseteq \mathcal{T}$ occuring at position $l \in \mathcal{L}$, that is $U / l=\{t / l \mid t \in U \wedge l \in a(t(\epsilon))\}$.
Example 1. We shall use the following example of quasi-lattice of type constructors given with their labels, $\left\{k_{0}, k_{1}, k_{2}\left(l_{1}, l_{2}, l_{3}\right), k_{3}\left(l_{2}, l_{3}\right), k_{4}\left(l_{2}\right), k_{5}\left(l_{3}\right)\right\}$, where $\mathcal{L}^{+}=\left\{l_{1}, l_{2}, l_{3}\right\}, \mathcal{L}^{-}=\emptyset$, and the subtyping relation is pictured out as follows:


Definition 4. A type constructor $k^{\prime} \in \mathcal{K}$ is a lower (resp. upper) bound of another constructor $k \in \mathcal{K}$ w.r.t. a set of labels $L \subseteq \mathcal{L}$ if $k^{\prime} \leq_{\mathcal{K}} k$ (resp. $k \leq \mathcal{K} k^{\prime}$ ) and $a(k) \cap a\left(k^{\prime}\right) \subseteq L$.

We note $\downarrow_{L} k$ (resp. $\uparrow_{L} k$ ) the set of lower (resp. upper) bounds of $k$ w.r.t. $L$. In example 1, we have $\downarrow_{\left\{l_{2}, l_{3}\right\}} k_{2}=\left\{k_{3}, k_{4}, k_{5}\right\}$ and $\uparrow_{\left\{l_{3} F\right.}=\left\{k_{3}, k_{5}\right\}$. Next, we define the subset of labels of $k$ occuring in $\downarrow_{k} L$ :

Definition 5. For a set of labels $L \subseteq \mathcal{L}$, the subset of significant labels of $L$ under (resp. over) $k$ is the set

$$
S L \downarrow_{L} k=a(k) \cap \bigcup_{k^{\prime} \in \downarrow_{I^{k}}} a\left(k^{\prime}\right)
$$

$\left(\right.$ resp. $\left.S L \uparrow_{L} k=a(k) \cap \bigcup_{k^{\prime} \in \uparrow_{L} k} a\left(k^{\prime}\right)\right)$.
In example 1, we have $S L \downarrow_{\left\{l_{1}, l_{2}\right\}} k_{2}=\left\{l_{2}\right\}$ and $S L \uparrow_{\left\{l_{1}, l_{2}\right\}} k_{2}=\emptyset$. One can easily check using the conditions of the definition 2 of a signature the following:
Proposition 1. If $\downarrow_{L} k \neq \emptyset$, then $\downarrow_{L} k$ has a maximum $\sqcup \downarrow_{L} k$ and $a\left(\sqcup \downarrow_{L} k\right)=$ $S L \downarrow_{L} k$. If $\uparrow_{L} k \neq \emptyset$, then $\uparrow_{L} k$ has a minimum $\sqcap \uparrow_{L} k$ and $a\left(\sqcap \uparrow_{L} k\right)=S L \uparrow_{L} k$.

### 2.3 Subtype ordering

The subtyping relation $\leq$ is defined over types by coinduction, as the intersection of a sequence $\left(\leq_{n}\right)$ of preorders over types defined by:

```
\(-\leq_{0}=\mathcal{T} \times \mathcal{T}\)
\(-t \leq_{n+1} t^{\prime}\) if \(t(\epsilon) \leq \mathcal{K} t^{\prime}(\epsilon)\) and for all \(l \in a(t(\epsilon)) \cap a\left(t^{\prime}(\epsilon)\right)\) :
    - either \(l \in \mathcal{L}^{+}\)and \(t / l \leq_{n} t^{\prime} / l\)
    - or \(l \in \mathcal{L}^{-}\)and \(t^{\prime} / l \leq_{n} t / l\)
\(-\leq=\bigcap_{n \in \mathbb{N}} \leq_{n}\)
```

Proposition 2. $\leq$ is an order over $\mathcal{T}$.
Proof. We show by induction that for all $n \in \mathbb{N}, \leq_{n}$ is a preorder and we deduce the reflexivity and the transitivity of $\leq$. To show the antisymetry, we first show by induction that for all $n \in \mathbb{N}$, if $t_{1} \leq_{n+1} t_{2}$ and $t_{2} \leq_{n+1} t_{1}$ then for all position $w \in \operatorname{dom}\left(t_{1}\right)$ of length $|w| \leq n$, we have $w \in \operatorname{dom}\left(t_{2}\right)$ and $t_{1}(w)=t_{2}(w)$. Now let us consider $t_{1} \leq t_{2} \leq t_{1}$ : if $t_{1} \neq t_{2}$, then there exists a $w$ of minimal size such that $t_{1}(w) \neq t_{2}(w)$. However $t_{1} \leq_{|w|+1} t_{2} \leq_{|w|+1} t_{1}$, so $t_{1}(w)=t_{2}(w)$, a contradiction.

Similarly we show:
Proposition 3. Let $t_{1}, t_{2} \in \mathcal{T}$. $t_{1} \leq t_{2}$ if and only if $t_{1}(\epsilon) \leq \mathcal{K} t_{2}(\epsilon)$ and for all $l \in a\left(t_{1}(\epsilon)\right) \cap a\left(t_{2}(\epsilon)\right)$ :

- either $l \in \mathcal{L}^{+}$and $t_{1} / l \leq t_{2} / l$
- or $l \in \mathcal{L}^{-}$and $t_{2} / l \leq t_{1} / l$

Now, our goal is to show that this ordered set of types forms a quasi-lattice. First we define the set of usable labels under a set of types $S$ as the set of labels $l$ such that $S / l$ has a lower bound:

Definition 6. The set of usable labels under a set of types $S \subseteq \mathcal{T}$ is the set

$$
U L \downarrow S=\left\{l \in \mathcal{L}^{+} \mid \downarrow(S / l) \neq \emptyset\right\} \cup\left\{l \in \mathcal{L}^{-} \mid \uparrow(S / l) \neq \emptyset\right\}
$$

The set of usable labels above $S$ is the set

$$
U L \uparrow S=\left\{l \in \mathcal{L}^{+} \mid \uparrow(S / l) \neq \emptyset\right\} \cup\left\{l \in \mathcal{L}^{-} \mid \downarrow(S / l) \neq \emptyset\right\}
$$

For example with the types $t=k_{2}\left(k_{0}, k_{1}, k_{4}\left(k_{0}\right)\right)$ and $t^{\prime}=k_{3}\left(k_{1}, k_{5}\left(k_{1}\right)\right)$ formed over the constructors of example 1 , we have $U L \downarrow\left\{t, t^{\prime}\right\}=\left\{l_{1}, l_{2}\right\}$ and $U L \uparrow\left\{t, t^{\prime}\right\}=$ $\left\{l_{1}, l_{2}, l_{3}\right\}$. Second, we define what will be the head constructor of greatest lower bounds and least upper bounds in $\mathcal{T}$.

Definition 7. For a set of types $S \subseteq \mathcal{T}$, the greatest lower bound constructor of $S$ is the constructor noted $\Pi_{\epsilon} S=\sqcup \downarrow_{(U L \downarrow S)}(\sqcap\{s(\epsilon) \mid s \in S\})$, the least upper bound constructor of $S$ is the constructor noted $\sqcup_{\epsilon} S=\sqcap \uparrow_{(U L \uparrow S)}(\sqcup\{s(\epsilon) \mid s \in S\})$

Now we define sequences of types that approximate the greatest lower bound of a set of types up to a given depth, starting with an arbitrary type constant of arity $\emptyset, k_{0}$, as follows:

Definition 8. The greatest lower (resp. least upper) bound of rank $n$ of a non empty set $S \subseteq \mathcal{T}$ of types, noted $\square_{n} S$ (resp. $\sqcup_{n} S$ ), is defined by:
$-\Pi_{0} S=\sqcup_{0} S=k_{0}$
$-\left(\Pi_{n+1} S\right)(\epsilon)=\Pi_{\epsilon} S$ and $\forall l \in a\left(\Pi_{\epsilon} S\right)$ :

- either $l \in \mathcal{L}^{+}$and $\left(\Pi_{n+1} S\right) / l=\Pi_{n}(S / l)$
- or $l \in \mathcal{L}^{-}$and $\left(\square_{n+1} S\right) / l=\sqcup_{n}(S / l)$
$-\left(\sqcup_{n+1} S\right)(\epsilon)=\sqcup_{\epsilon} S$ and $\forall l \in a\left(\sqcup_{\epsilon} S\right)$ :
- either $l \in \mathcal{L}^{+}$and $\left(\sqcup_{n+1} S\right) / l=\sqcup_{n}(S / l)$
- or $l \in \mathcal{L}^{-}$and $\left(\sqcup_{n+1} S\right) / l=\Pi_{n}(S / l)$

This provides a construction of the following candidates for the greatest lower bound and the least upper bound of a set of types:

Definition 9. The partial mapping $\Pi_{\mathcal{T}}: \wp(\mathcal{T}) \rightarrow\left(\mathcal{L}^{*} \rightarrow \mathcal{K}\right)\left(\right.$ resp. $\left.\sqcup_{\mathcal{T}}\right)$ is defined by:

$$
\left(\sqcap_{\mathcal{T}} S\right)(w)=\left(\sqcap_{n+1} S\right)(w) \quad\left(\operatorname{resp} .\left(\sqcup_{\mathcal{T}} S\right)(w)=\left(\sqcup_{n+1} S\right)(w)\right)
$$

for all non empty set of types $S \subseteq \mathcal{T}$, for all $n \in \mathbb{N}$, for all position $w \in$ $\operatorname{dom}\left(\sqcap_{n+1}(S)\right)\left(\right.$ resp. $\left.\sqcup_{n+1}\right)$ such that $|w|=n$.

Proposition 4. Let $S \neq \emptyset \subseteq \mathcal{T}$. If $\downarrow S \neq \emptyset$ (resp. $\uparrow S \neq \emptyset$ ) then $\sqcap_{\mathcal{T}} S$ (resp. $\sqcup_{\mathcal{T}} S$ ) is well defined and is a type.

Proposition 5. Let $S \neq \emptyset \subseteq \mathcal{T}$ such that $\downarrow S \neq \emptyset$ (resp. $\uparrow S \neq \emptyset$ ). Then for all $s \in S, \sqcap_{\mathcal{T}} S \leq s$ (resp. $s \leq \sqcup_{\mathcal{T}} S$ ).
Proposition 6. Let $S \neq \emptyset \subseteq \mathcal{T}$ such that $\downarrow S \neq \emptyset$ (resp. $\uparrow S \neq \emptyset$ ). For all $t \in \downarrow S, t \leq \Pi_{\mathcal{T}} S$ (resp. for all $t \in \uparrow S, \sqcup_{\mathcal{T}} S \leq t$ ).
Theorem 1. $(\mathcal{T}, \leq)$ is a complete quasi-lattice, where $\square_{\mathcal{T}}$ denotes greatest lower bounds and $\sqcup_{\mathcal{T}}$ denotes least upper bounds.
Proof. Let $S \neq \emptyset \subseteq \mathcal{T}$. If $S$ has a lower bound then, by proposition 4, $\sqcap_{\mathcal{T}} S$ exists. By proposition 5, for all $s \in S, \sqcap_{\mathcal{T}} S \leq s$ and by proposition 6, for all $t \in \downarrow S, t \leq \Pi_{\mathcal{T}} S$. So $\Pi_{\mathcal{T}} S$ is the greatest lower bound of $S$. Similarly, we show that if $S$ has an upper bound, then $\sqcup_{\mathcal{T}} S$ is defined and is the least upper bound of $S$. So $(\mathcal{T}, \leq)$ is a quasi-lattice.

Concerning the subset $\mathcal{R} \subseteq \mathcal{T}$ of regular types (i.e. types having a finite number of subterms), we have the following:
Proposition 7. Let $t_{1}$ and $t_{2}$ two regular types. If $t_{1} \square_{\mathcal{T}} t_{2}$ is defined, then it is a regular type. If $t_{1} \cup_{\mathcal{T}} t_{2}$ is defined, then it is a regular type.
Theorem 2. $(\mathcal{R}, \leq)$ is a quasi-lattice.
Proof. By theorem 1, $(\mathcal{T}, \leq)$ is a quasi-lattice. By proposition 7, if $r_{1}, r_{2} \in \mathcal{R}$ and $\exists r, r \leq r_{1} \wedge r \leq r_{2}$ (resp. $\left.r_{1} \leq r \wedge r_{2} \leq r\right)$ then $r_{1} \sqcap_{\mathcal{T}} r_{2} \in \mathcal{R}$ (resp. $r_{1} \sqcup_{\mathcal{T}} r_{2} \in \mathcal{R}$ ). So ( $\mathcal{R}, \leq)$ is a quasi-lattice where $\square_{\mathcal{T}}$ denotes greatest lower bounds and $\sqcup_{\mathcal{T}}$ denotes least upper bounds.

It is worth noting however that $(\mathcal{R}, \leq)$ may not be a complete quasi-lattice. For example, let $\mathcal{K}=\{a, b\}$ with $a \leq_{\mathcal{K}} b$ and $a(a)=a(b)=\{l\}$. Let $\left(u_{n}\right)_{n \geq 0}$ be the sequence of types defined by $u_{n}(w)=b$ if $|w|=\frac{n(n+1)}{2}$, and $u_{n}(w)=a$ otherwise. One can check that $\left(u_{n}\right)_{n \geq 0}$ has no lower bound in $\mathcal{R}$.

## 3 Testing the satisfiability of subtyping constraints

Let $\mathcal{V}$ be a countable set of variables, noted $\alpha, \beta, \ldots$ Types with variables are defined as the set, noted $\mathcal{T}$, of types built upon the signature $\left(\mathcal{K} \cup \mathcal{V}, \leq \mathcal{K}, \mathcal{L}^{+}, \mathcal{L}^{-}, a\right)$. A subtyping constraint is a pair of finite types noted $t_{1} \leq t_{2}$. For a system $C$ of subtyping constraints, we note $V(C)$ the set of variables occuring in $C$.
Definition 10. A substitution $\rho: \mathcal{V} \rightarrow \mathcal{T}$ satisfies the constraint $t_{1} \leq t_{2}$, noted $\rho=t_{1} \leq t_{2}$, if $\rho\left(t_{1}\right) \leq \rho\left(t_{2}\right)$. The subtyping constraint $t_{1} \leq t_{2}$ is satisfiable if there exist a substitution $\rho$ such that $\rho \models t_{1} \leq t_{2}$.

For the sake of simplicity, we will suppose, without loss of generality, that the constraint systems we consider contain only small terms. A small term is either a variant, a constant, or a term of depth 1 where all leaves are variables. For example, int, list( $\alpha$ ) and $\alpha$ are small terms while list(int) is not. Clearly, given a constraint system, one can find an equivalent constraint system where all terms are small terms, by introducing variables for arguments of terms that are not small terms, and by introducing equality (double inequality) constraints between these variables and the corresponding arguments.

### 3.1 Closed systems

We first define pre-closed systems as constraint systems where variables are bounded. We recall in table 1 the partial function dec used for breaking constraints in Trifonov and Smith's algorithm [10].
Definition 11 (Pre-closed systems). A constraint system $c$ is said to be upper pre-closed if for all variable $\alpha \in V(C)$, there exists $t \notin \mathcal{V}$ such that $t \leq \alpha \in C . C$ is said to be lower pre-closed if for all $\alpha \in V(C)$, there exists $t \notin \mathcal{V}$ such that $\alpha \leq t \in C$. A constraint system is said to be pre-closed if it is upper and lower pre-closed.

$$
\begin{aligned}
\operatorname{dec}(\alpha \leq \beta)= & \{\alpha \leq \beta\} \\
\operatorname{dec}(\alpha \leq t)= & \{\alpha \leq t\} \\
\operatorname{dec}(t \leq \alpha)= & \{t \leq \alpha\} \\
\operatorname{dec}\left(t_{1} \leq t_{2}\right)= & \bigcup_{l \in a+\left(t_{1}(\epsilon)\right) \cap a+\left(t_{2}(\epsilon)\right)}\left\{t_{1} / l \leq t_{2} / l\right\} \cup \bigcup_{l \in a^{-}\left(t_{1}(\epsilon)\right) \cap a^{-}\left(t_{2}(\epsilon)\right)}\left\{t_{2} / l \leq t_{1} / l\right\} \\
& \text { if } t_{1}(\epsilon) \leq \mathcal{K} t_{2}(\epsilon) .
\end{aligned}
$$

Table 1. Trifonov and Smith's decomposition function [7, 10]

Definition 12 (Closed system). A constraint system $C$ is closed if it is preclosed and if for all constraint $c \in C$, dec $(c) \in C$ and for all $\left\{t_{1} \leq \alpha, \alpha \leq t_{2}\right\} \subseteq$ $C$, $\operatorname{dec}\left(t_{1} \leq t_{2}\right)$ is defined and included in $C$.

Some technical notions and lemmas will be necessary to prove that closed systems are satisfiable in quasi-lattices (theorem 3). For a variable $\alpha$, let $\Uparrow_{C} \alpha=$ $\{t \mid t \notin \mathcal{V}, \alpha \leq t \in C\}$ and $\Downarrow_{C} \alpha=\{t \mid t \notin \mathcal{V}, t \leq \alpha \in C\}$ be the sets of types bounding $\alpha$ in $C$. For a set of variables $A$, we note $\Uparrow_{C} A=\bigcup_{\alpha \in A} \Uparrow_{C} \alpha$ and $\Downarrow_{C} A=\bigcup_{\alpha \in A} \Downarrow_{C} \alpha$. By abuse of notation, when $C$ is clear from the context, we will omit $C$ in the notations. Given a constraint system $C$, the following partial function sol associates to two sets of variables in $C$, the head constructor of a type which can be bounded by these variables in $C$ :
Definition 13. Given a constraint system $C$, sol : $\wp(V(C)) \times \wp(V(C)) \rightarrow \mathcal{K}$ is a partial function defined by $\operatorname{sol}(A, B)=\sqcup\left(\downarrow_{a(\sqcup D)}(\sqcap U)\right)$ when it exists, where $U=\{t(\epsilon) \mid t \in \Uparrow B\}$ and $D=\{t(\epsilon) \mid t \in \Downarrow A\}$.
Lemma 1. In a closed system $C$, sol $(A, B)$ is defined for all non empty sets $A, B$ such that $\forall \alpha \in A, \forall \beta \neq \alpha \in B, \alpha \leq \beta \in C$.
Proof. Let $t \in \Downarrow A$ and $t^{\prime} \in \Uparrow B$. Since $C$ is closed, $\operatorname{dec}\left(t \leq t^{\prime}\right)$ is defined, hence $t(\epsilon) \leq \mathcal{k} t^{\prime}(\epsilon)$. So $\forall k \in D, \forall k^{\prime} \in U, k \leq \mathcal{k} k^{\prime}$. Since $C$ is closed, $\Downarrow A \neq \emptyset$ and $\Uparrow B \neq$ $\emptyset$. So $\sqcup D$ and $\sqcap U$ are defined and $\sqcup D \leq \mathcal{\kappa} \sqcap U$. Therefore $\downarrow_{a(\sqcup D)}(\sqcap U) \neq \emptyset$ and $\operatorname{sol}(A, B)$ is well defined.

Lemma 2. Let $C$ be a closed constraint system and $A, B \subseteq V(C)$ verifying the conditions of lemma 1. For all label $l \in a(\operatorname{sol}(A, B))$, if $l \in \mathcal{L}^{+}$then $(\Downarrow A / l, \Uparrow B / l)$ satisfies the condition of lemma 1 , and if $l \in \mathcal{L}^{-}$then $(\Uparrow B / l, \Downarrow A / l)$ satisfies the condition of lemma 1 .

Lemma 3. Let $C$ be a closed constraint system. Let $A, B, E, F \subseteq V(C)$. If $\Downarrow A \subseteq \Downarrow E$ and $\Uparrow F \subseteq \Uparrow B$ and $A, B$ and $E, F$ satisfy the conditions of lemma 1 , then $\operatorname{sol}(A, B) \leq \mathcal{K} \operatorname{sol}(E, F)$.

Theorem 3. In a quasi-lattice, any closed constraint system is satisfiable.
Proof. Let $C$ be a closed constraint system. Let us consider the partial mapping $\Gamma: \wp(V(C)) \times \wp(V(C)) \times \mathcal{L}^{*} \rightarrow \wp(V(C)) \times \wp(V(C))$ defined for couples $(A, B)$ satisfying the conditions of lemma 1 as follows: $\Gamma(A, B, \epsilon)=(A, B)$, if $\Gamma(A, B, w)=\left(A^{\prime}, B^{\prime}\right)$ then, for all label $l \in a\left(\operatorname{sol}\left(A^{\prime}, B^{\prime}\right)\right), l \in \mathcal{L}^{+}, \Gamma(A, B, w . l)=$ $\left(\left(\Downarrow A^{\prime}\right) / l,\left(\Uparrow B^{\prime}\right) / l\right)$ and if $l \in \mathcal{L}^{-}, \Gamma(A, B, w . l)=\left(\left(\Uparrow B^{\prime}\right) / l,\left(\Downarrow A^{\prime}\right) / l\right)$. Let us consider $\gamma: \wp(V(C)) \times \wp(V(C)) \rightarrow \mathcal{T}$ defined by $\gamma(A, B)(w)=\operatorname{sol}(\Gamma(A, B, w))$. By induction, using lemma 2, one can check that $\gamma(A, B)$ is a type. Now, let us consider the substitution $\rho(\alpha)=\gamma(\{\alpha\},\{\alpha\})$.

We show that $\rho \neq C$. By induction, we show that $\forall n \in \mathbb{N}, \forall t_{1} \leq t_{2} \in$ $C, \rho\left(t_{1}\right) \leq_{n} \rho\left(t_{2}\right)$ and for all $A, B, E, F \in V(C)$ satisfying the conditions of lemma $3, \gamma(A, B) \leq_{n} \gamma(E, F)$.

The case $n=0$ is trivially true. Now, we show the case $n+1$.
Let us consider the case $t_{1}=\gamma(A, B) \leq_{n+1} \gamma(E, F)=t_{2}$. By lemma 3, $k_{1}=\gamma(A, B)(\epsilon) \leq \mathcal{K} \gamma(E, F)(\epsilon)=k_{2}$. Let $l \in a\left(k_{1}\right) \cap a\left(k_{2}\right), l \in \mathcal{L}^{+}$. Since $\Downarrow A \subseteq$ $\Downarrow E,(\Downarrow A) / l \subseteq(\Downarrow E) / l$ and thus $\Downarrow((\Downarrow A) / l) \subseteq \Downarrow((\Downarrow E) / l)$. Similarly $\Uparrow((\Uparrow F) / l) \subseteq$ $\Uparrow((\Uparrow B) / l)$. So, using the induction hypothesis, $t_{1} / l=\gamma((\Downarrow A) / l,(\Uparrow B) / l) \leq_{n}$ $\gamma((\Downarrow E) / l,(\Uparrow F) / l)=t_{2} / l$. Similarly, if $l \in \mathcal{L}^{-}, t_{2} / l=\gamma((\Uparrow F) / l,(\Downarrow E) / l) \leq_{n}$ $\gamma((\Uparrow B) / l,(\Downarrow A) / l)=t_{1} / l$. Thus we deduce $t_{1} \leq_{n+1} t_{2}$.

Let us consider $\rho(\alpha) \leq \rho(\beta)$. Since $\alpha \leq \beta \in C$ and since $C$ is closed, $\Downarrow \alpha \subseteq$ $\Downarrow \beta$ and $\Uparrow \beta \subseteq \Uparrow \alpha$. So we can apply the preceeding result obtaining $\rho(\alpha)=$ $\gamma(\{\alpha\},\{\alpha\}) \leq_{n+1} \gamma(\{\beta\},\{\beta\})=\rho(\beta)$.

Let us consider $\alpha \leq t$, with $t \notin \mathcal{V}$. Since $t \in \Uparrow \alpha, \rho(\alpha)(\epsilon)=\operatorname{sol}(\{\alpha\},\{\alpha\}) \leq \mathcal{K}$ $\sqcap\left\{t^{\prime}(\epsilon) \mid t^{\prime} \in \Uparrow \alpha\right\} \leq \mathcal{K}^{t} t(\epsilon)$. Let $l \in a(\rho(\alpha)(\epsilon)) \cap a(t(\epsilon)), l \in \mathcal{L}^{+}$. Since we use small terms, $t / l$ is a variable $\beta$, thus $\rho(t) / l=\gamma(\{\beta\},\{\beta\})$ and $\rho(\alpha) / l=$ $\gamma((\Downarrow \alpha) / l,(\Uparrow \alpha) / l)$. Since $\beta \in(\Uparrow \alpha) / l, \Uparrow \beta \subseteq \Uparrow((\Uparrow \alpha) / l)$. Since $C$ is closed, for all $\alpha^{\prime} \in(\Downarrow \alpha) / l, \alpha^{\prime} \leq \beta \in C$. Since $C$ is closed, we have $\Downarrow((\Downarrow \alpha) / l) \subseteq \Downarrow \beta$. Thus we obtain $\rho(\alpha) / l=\gamma((\Downarrow \alpha) / l,(\Uparrow \alpha) / l) \leq_{n} \gamma(\{\beta\},\{\beta\})=\rho(t) / l$. Similarly, if $l \in \mathcal{L}^{-}$, $\rho(t) / l=\gamma(\{\beta\},\{\beta\}) \leq_{n} \gamma((\Uparrow \alpha) / l,(\Downarrow \alpha) / l)=\rho(\alpha) / l$. So $\rho(\alpha) \leq_{n+1} \rho(t)$

Similarly, we show the case $t \leq \alpha$.
The last case is $t_{1} \leq t_{2}$ with $t_{1}, t_{2} \notin V(C)$. Since $C$ is closed $t_{1}(\epsilon) \leq \mathcal{K} t_{2}(\epsilon)$. Let $l \in a\left(t_{1}(\epsilon)\right) \cap a\left(t_{2}(\epsilon)\right), l \in \mathcal{L}^{+}$. Since $C$ is closed, posing $\alpha=t_{1} / l$ and $\beta=t_{2} / l$, we have $\alpha \leq \beta \in C$ and, using the induction hypothesis, $\rho\left(t_{1}\right) / l=\rho(\alpha) \leq_{n} \rho(\beta)=$ $\rho\left(t_{2}\right) / l$. Similarly, if $l \in \mathcal{L}^{-}$, we obtain $\rho\left(t_{2}\right) / l \leq_{n} \rho\left(t_{1}\right) / l$. Thus $\rho\left(t_{1}\right) \leq_{n+1} \rho\left(t_{2}\right)$.

So for all $n \in \mathbb{N}$, for all $t_{1} \leq t_{2} \in C, \rho\left(t_{1}\right) \leq_{n} \rho\left(t_{2}\right)$, i.e. $\rho\left(t_{1}\right) \leq \rho\left(t_{2}\right)$. Thus $\rho \models C$.

Given a pre-closed system $C$, one can compute its closure, as in Trifonov and Smith's algorithm [10] in $O\left(n^{3}\right)$. The algorithm proceeds as follows: let the sequence $C, C 1, C 2, \ldots$ be defined by:

$$
C^{n+1}=C^{n} \cup \bigcup_{c \in C} d e c(c) \cup \bigcup_{\left\{t \leq \alpha, \alpha \leq t^{\prime}\right\} \subseteq C} d e c\left(t \leq t^{\prime}\right)
$$

For any $C^{n}$, it is clear that $C^{n+1}$ is equivalent to $C^{n}$ whenever it is defined. Otherwise $C^{n}$ has no solution because we try to apply dec on some constraint $t_{1} \leq t_{2}$ while $t_{1}(\epsilon) \not \subset \mathcal{K} t_{2}(\epsilon)$, which means that $t_{1} \leq t_{2}$ is not satisfiable. Thus, if an element of the sequence is not defined, $C$ is not satisfiable. Otherwise the sequence reaches a fix point which is closed, hence satisfiable by theorem 3 , and equivalent to $C$, so $C$ is satisfiable.

### 3.2 Pre-closure algorithm

The algorithm above requires a pre-closed system as an entry. This condition is automatically filled in lattices since there exists a maximal type $T$ and a minimal type $\perp$. In this case, it is sufficient to add constraints $\perp \leq \alpha$ and $\alpha \leq \top$ to obtain a pre-closed system with the same solutions [7, 10]. In quasi-lattices, the theorem 4 below provides sufficient conditions over $\mathcal{K}$ for deciding the satisfiability of a non pre-closed constraint system. Let $\overline{\mathcal{K}}$ be the set of maximal elements of $\mathcal{K}$ and $\underline{\mathcal{K}}$ the set of its minimal elements.
Theorem 4. If $\mathcal{K}$ verify the following conditions:

1. $\forall k \in \overline{\mathcal{K}} \cup \underline{\mathcal{K}}, a(k)=\emptyset$
2. For all $k \in \mathcal{K}$, there exists $k_{1} \in \underline{\mathcal{K}}$ and $k_{2} \in \overline{\mathcal{K}}$ such that $k_{1} \leq \mathcal{K} k \leq \mathcal{K} k_{2}$.

For any constraint system $C$ let the set of pre-closures $p c(C)$ be:

$$
p c(C)=\left\{C \cup \bigcup_{\alpha \in V(C)}\left\{t_{\alpha} \leq \alpha, \alpha \leq t_{\alpha}^{\prime}\right\} \mid t_{\alpha}(\epsilon) \in \underline{\mathcal{K}}, t_{\alpha}^{\prime}(\epsilon) \in \overline{\mathcal{K}}\right\}
$$

All elements in pc $(C)$ are closed and the union of their sets of solutions is equal to the set of solutions of $C$.

Proof. Since $\forall k \in \overline{\mathcal{K}} \cup \underline{\mathcal{K}}, a(k)=\emptyset$, for all $C^{\prime} \in p c(C), V\left(C^{\prime}\right)=V(C)$. So, by construction, the elements of $p c(C)$ are pre-closed. For all $C^{\prime} \in p c(C)$, we have $C \subseteq C^{\prime}$, thus $\rho \vDash C^{\prime} \Rightarrow \rho \models C$. Now we show that if $\rho \vDash C$ then there exists $C^{\prime} \in p c(C)$ such that $\rho=C^{\prime}$. By condition 2) for all $\alpha \in V(C)$, one can find $k_{\alpha} \in \underline{\mathcal{K}}$ and $k_{\alpha}^{\prime} \in \overline{\mathcal{K}}$ such that $\rho \models t \leq \alpha, \alpha \leq t^{\prime}$ with $t(\epsilon)=k_{\alpha}$ and $t^{\prime}(\epsilon)=k_{\alpha}^{\prime}$. Thus there exists $C^{\prime} \in p c(C)$ such that $\rho \models C^{\prime}$. Thus the union of the sets of solutions of the elements of $p c(C)$ is equal to the set of solutions of $C$.

If $\overline{\mathcal{K}}$ and $\underline{\mathcal{K}}$ are finite sets, it is possible to enumerate the elements of $p c(C)$. Since these elements are pre-closed, one can test their satisfiability using the
closure algorithm of the previous section. This gives an algorithm for testing the satisfiability of non-closed constraint systems in quasi-lattices with a finite number of extrema each with an empty arity. The time complexity of the satisfiability test is in $O\left(n^{3} m^{v} M^{v}\right)$ where $n$ is the size of the constraint system, $m$ is the size of $\underline{\mathcal{K}}$ and $M$ the size of $\overline{\mathcal{K}}$, and $v$ is the number of unbounded variables.

Theorem 5. The satisfiability problem for subtyping constraints in quasi-lattices with a finite number of extrema each with an empty arity is NP-complete.

Proof. The satisfiability of pre-closed system is polynomial. By theorem 4 the set of pre-closures of a system can be guessed by enumerating the possible bounds for unbounded variable among a finite set, hence the satisfiability problem is in NP. To prove the NP-completeness, we use the result of Pratt and Tiuryn [8] that the satisfiability of subtyping constraints in n -crowns is NP-complete for $n \geq 2$. An n-crown is a poset with $2 n$ elements $k_{0}, \ldots, k_{2 n-1}$, all with an empty arity and partially ordered such that the only comparisons are $k_{2 i} \leq \mathcal{K} k_{2 i \pm 1}$ and $k_{0} \leq \mathcal{K} k_{2 n-1}$. Clearly, n-crowns with $n \geq 3$ are quasi-lattices with a finite number of extrema each with an empty arity. The satisfiability problem in quasi-lattices is thus NP-complete.

The first condition imposed on $K$ in theorem 4 expresses that the extrema in the quasi-lattice of constructors have an empty arity. Without this condition, it is worth noting that the introduction of a new constraint $t_{\alpha} \leq \alpha$ (or $\alpha \leq t_{\alpha}^{\prime}$ ) may also introduce some new unbounded variables appearing in $t$, that must be bounded by introducing new constraints, which leads to introduce an infinity of variables. Thus, the above algorithm cannot be used in that case. Our result thus lefts open the decidability of the satisfiability of non-structural subtyping constraints in quasi-lattices where some extrema have a non-empty arity.

## 4 Computation of explicit solutions

In [6], Pottier describes an algorithm for simplifying subtyping constraint systems in lattices, allowing the computation of bounds for variables occuring in constraints, and the computation of solutions by identifying the variables with their bounds. We extend here this algorithm to the case of quasi-lattices.

Let us assume, as in section 3, that constraints are formed upon small terms and that the constraint system to be simplified is pre-closed. In order to solve a constraint system $C$ in a quasi-lattice $\mathcal{T}(\mathcal{K})$, we complete $\mathcal{K}$ in a lattice $\mathcal{K}^{\perp, \top}$ by adding $\perp$ and $\top$ elements with an empty arity and for all $k \in \mathcal{K}, \perp \leq_{\mathcal{K}^{\perp, \top}}$ $k \leq \mathcal{K}^{\perp, \top} T$. Then $C$ is solved in $\mathcal{T}\left(\mathcal{K}^{\perp, \top}\right)$ using Pottier's algorithm, obtaining a constraint system $C^{\prime}$ equivalent to $C$ and simplified. Finally, a set of rules is applied over $C^{\prime}$, in order to obtain bounds for the variables of $C$ in the quasilattice.

Now, we describe the form of the constraint system $C^{\prime}$ resulting from the application of the algorithm over a subtyping constraint system $C$ in the lattice $\mathcal{K}^{\perp, \top}$. Pottier's algorithm may introduce some variables representing the greatest
lower or least upper bounds of a set of original variables in $C$. For a set $A$ of variables in $C$, we note $\gamma_{A}$ the variable representing the greatest lower bound of $A$ and $\lambda_{A}$ the variable representing its least upper bound. We have recalled in table 2 the properties satisfied by $C^{\prime}[7]$. It is clear that $C^{\prime}$ is closed thus satisfiable in $\mathcal{T}\left(\mathcal{K}^{\perp, \top}\right)$.

1. For all $\alpha \in V\left(C^{\prime}\right)$, there exists exactly one type $t \notin \mathcal{V}$ (noted $\Downarrow_{C^{\prime}} \alpha$ ) such that $t \leq \alpha \in C^{\prime}$ and exactly one type $t^{\prime} \notin \mathcal{V}$ (noted $\left.\Uparrow_{C^{\prime}} \alpha\right)$ such that $\alpha \leq t^{\prime} \in C^{\prime}$.
2. For all $\{\alpha \leq \beta, \beta \leq \delta\} \subseteq C^{\prime}, \alpha \leq \delta \in C^{\prime}$.
3. For all $\alpha \in V\left(C^{\prime}\right)$, for all label $l \in a\left(\left(\Uparrow_{C^{\prime}} \alpha\right)(\epsilon)\right)$, either $l \in \mathcal{L}^{+}$and $\exists A, t / l=\gamma_{A}$, or $l \in \mathcal{L}^{-}$and $\exists A, t / l=\lambda_{A}$. For all label $l \in a\left(\left(\Downarrow_{C^{\prime}} \alpha\right)(\epsilon)\right)$, either $l \in \mathcal{L}^{+}$and $\exists A, t / l=\lambda_{A}$, or $l \in \mathcal{L}^{-}$and $\exists A, t / l=\gamma_{A}$.
4. If $\alpha \leq \beta \in C^{\prime}$, or if $\alpha \equiv \gamma_{A}$ and $\beta \equiv \gamma_{B}$ with $B \subseteq A$ then $k_{\alpha}=\left(\Uparrow_{C^{\prime}} \alpha\right)(\epsilon) \leq_{\mathcal{K}^{\perp,}, \top}$ $k_{\beta}=\left(\Uparrow_{C^{\prime}} \beta\right)(\epsilon)$ and for all label $l \in a\left(k_{\alpha}\right) \cap a\left(k_{\beta}\right)$, if $l \in \mathcal{L}^{+},\left(\Uparrow_{C^{\prime}} \alpha\right) / l=\gamma_{E}$, $\left(\Uparrow_{C^{\prime}} \beta\right) / l=\gamma_{F}$, then $F \subseteq E$. If $l \in \mathcal{L}^{-},\left(\Uparrow_{C^{\prime}} \alpha\right) / l=\lambda_{E},\left(\Uparrow_{C^{\prime}} \beta\right) / l=\lambda_{F}$, then $F \subseteq E$.
5. If $\alpha \leq \beta \in C^{\prime}$, or if $\alpha \equiv \lambda_{A}$ and $\beta \equiv \lambda_{B}$ with $A \subseteq B$ then $k_{\alpha}=\left(\Downarrow_{C^{\prime}} \alpha\right)(\epsilon) \leq_{\mathcal{K}^{\perp}, \top}$ $k_{\beta}=\left(\Downarrow_{C^{\prime}} \beta\right)(\epsilon)$ and for all label $l \in a\left(k_{\alpha}\right) \cap a\left(k_{\beta}\right)$, if $l \in \mathcal{L}^{+},\left(\Downarrow_{C^{\prime}} \alpha\right) / l=\lambda_{E}$, $\left(\Downarrow_{C^{\prime}} \beta\right) / l=\lambda_{F}$, then $E \subseteq F$. If $l \in \mathcal{L}^{-},\left(\Downarrow_{C^{\prime}} \alpha\right) / l=\gamma_{E},\left(\Downarrow_{C^{\prime}} \beta\right) / l=\gamma_{F}$, then $E \subseteq F$.
6. For all variable $\gamma_{A}, \Uparrow_{C^{\prime}} \gamma_{A} \neq \top$ and for all variable $\lambda_{A}, \Downarrow_{C^{\prime}} \lambda_{A} \neq \perp$.
7. For all $\alpha \in V(C), \operatorname{dec}\left(\Downarrow_{C^{\prime}} \alpha \leq \Uparrow_{C^{\prime}} \alpha\right)$ is defined and included in $C^{\prime}$.
8. $\forall t, t^{\prime}, t \leq t^{\prime} \notin C^{\prime}$

Table 2. Properties verified by the result of Pottier's algorithm [7].

In table 3 we present a set of rules for computing a solution to a constraint system $D$ in a quasi-lattice.

Proposition 8. The application of the rules of table 3 preserve the solution which co-domain is included in $\mathcal{T}(\mathcal{K}) \cup\{\perp, \top\}$.

Proof. Let us consider the case (Down $\perp$ ), the other cases being similar. Let us assume that $\rho(\alpha)=D, \alpha \leq t, \gamma_{A} \leq \perp$ with $\operatorname{dom}(\rho) \subseteq \mathcal{T}(\mathcal{K}) \cup\{\perp, \top\}$ and let us show that $\rho \models D, \gamma_{A} \leq \perp, \alpha \leq t^{\prime}$. Clearly, $\rho\left(\gamma_{A}\right)=\perp$. Since for all label $l^{\prime} \in t^{\prime}(\epsilon), t / l^{\prime}=t^{\prime} / l^{\prime}$, it is sufficient to show that $\rho(\alpha)(\epsilon) \leq \mathcal{K} t^{\prime}(\epsilon) . t / l=$ $\gamma_{A}$, so $\rho(t) / l=\perp$. Since there is no type in $\mathcal{T}(\mathcal{K})$ smaller than $\perp$, and by proposition $3, l \notin a(\rho(\alpha)(\epsilon))$. Thus $\rho(\alpha)(\epsilon) \leq_{\mathcal{K}, \perp, \top} \sqcup\left(\downarrow_{a(t(\epsilon)) \backslash\{i\}} t(\epsilon)\right)$ if it exists and $\rho(\alpha)(\epsilon) \leq_{\mathcal{K}^{\perp, \top}} \perp$ otherwise. Thus $\rho(\alpha)(\epsilon) \leq_{\mathcal{K}^{\perp, \top}} t^{\prime}(\epsilon)$.

Proposition 9. The rules of table 3 terminate and are confluent.
Proof. By proposition 1, if $D, \alpha \leq t \rightarrow D, \alpha \leq t^{\prime}$ then $a\left(t^{\prime}(\epsilon)\right) \subset a(t(\epsilon))$, thus one can apply rules (Down $\perp$ ) and (Down $\top$ ) only a finite number of times for each variable. As well, one can apply rules ( $\mathrm{Up} \perp$ ) and ( $\mathrm{Up} \top$ ) only a finite number of times for each variable. Since no rules increases the arity of the variables bounds,

```
(Down \(\perp) D, \gamma_{A} \leq \perp, \alpha \leq t \rightarrow D, \gamma_{A} \leq \perp, \alpha \leq t^{\prime}\)
If \(t / l=\gamma_{A}\).
\(t^{\prime}(\epsilon)=\sqcup\left(\downarrow_{a(t(\epsilon)) \backslash\{l\}} t(\epsilon)\right)\)
    if it is defined, \(t^{\prime}(\epsilon)=\perp\) otherwise.
    For all label \(l^{\prime} \in t^{\prime}(\epsilon), t^{\prime} / l^{\prime}=t / l^{\prime}\).
(Down T) \(D, \top \leq \lambda_{A}, \alpha \leq t \rightarrow D, \top \leq \lambda_{A}, \alpha \leq t^{\prime}\)
If \(t / l=\lambda_{A}\).
\(t^{\prime}(\epsilon)=\sqcup\left(\downarrow_{a(t(\epsilon)) \backslash\left\{{ }_{l}\right\}} t(\epsilon)\right)\)
if it is defined, \(t^{\prime}(\epsilon)=\perp\) otherwise.
For all label \(l^{\prime} \in t^{\prime}(\epsilon), t^{\prime} / l^{\prime}=t / l^{\prime}\).
(Up T) \(D, \top \leq \lambda_{A}, t \leq \alpha \rightarrow D, \top \leq \lambda_{A}, t^{\prime} \leq \alpha\)
If \(t / l=\lambda_{A}\).
\(t^{\prime}(\epsilon)=\Pi\left(\uparrow_{a(t(\epsilon)) \backslash\{l\}} t(\epsilon)\right)\)
if it is defined, \(t^{\prime}(\epsilon)=\perp\) otherwise.
For all label \(l^{\prime} \in t^{\prime}(\epsilon), t^{\prime} / l^{\prime}=t / l^{\prime}\).
\((\mathrm{Up} \perp) \quad D, \gamma_{A} \leq \perp, t \leq \alpha \rightarrow D, \gamma_{A} \leq \perp, t^{\prime} \leq \alpha\)
If \(t / l=\gamma_{A}\).
\(t^{\prime}(\epsilon)=\Pi\left(\uparrow_{a(t(\epsilon)) \backslash\{l\}} t(\epsilon)\right)\)
if it is defined, \(t^{\prime}(\epsilon)=\perp\) otherwise.
For all label \(l^{\prime} \in t^{\prime}(\epsilon), t^{\prime} / l^{\prime}=t / l^{\prime}\).
```

Table 3. Rules for computing bounds in a quasi-lattice
the rewriting system terminates in $O(n)$ steps where $n$ is the size of the constraint system. The convergence can be checked by remarking that $\sqcup\left(\downarrow_{L^{\prime}}\left(\sqcup\left(\downarrow_{L} k\right)\right)\right)=$ $\sqcup\left(\downarrow_{L \cap L^{\prime}} k\right)=\sqcup\left(\downarrow_{L}\left(\sqcup\left(\downarrow_{L^{\prime}} k\right)\right)\right)$.

Proposition 10. Let $C^{\prime}$ be the result of Pottier's algorithm applied to $C$ and let $C^{\prime \prime} \rightarrow^{*} C^{\prime \prime} \nrightarrow$. Then $C^{\prime \prime}$ verifies the properties of table 2.

Proof. ${ }^{1}$ One can check that properties 1), 2), 3), 6), 7) and 8) are conserved during the application of the rules of table 3. If the application of a rule breaks the property 4) or 5), it is possible to reestablish it by applying this rule on some other variables. Since $\rightarrow^{*}$ is convergent, we obtain that $C^{\prime \prime}$ verifies properties 4) and 5).

Theorem 6. Let $C^{\prime}$ be the result of Pottier's algorithm applied to a pre-closed constraint system $C$, and let $C^{\prime} \rightarrow^{*} C^{\prime \prime} \nrightarrow$. If $\mathcal{L}^{-}=\emptyset$ then the upper bounds $\Uparrow_{C^{\prime \prime}} \alpha$ (resp. lower bounds $\Downarrow_{C^{\prime \prime}} \alpha$ ) in $C^{\prime \prime}$ define a maximal (resp. minimal) solution of $C$ in $\mathcal{T}(\mathcal{K})$.

Proof. Let be the following equation system: $\left\{\alpha=t \mid \alpha \leq t \in C^{\prime \prime}\right\}$. This system admit a unique solution $\rho$ because each variable appears only once on the left hand side of $=$. Let us show that $\rho$ is a solution of $C^{\prime \prime}$. In order to do this, we first show by induction that for all $n \in \mathbb{N}$, for all constraint $t \leq t^{\prime} \in C^{\prime \prime}$, $\rho(t) \leq_{n} \rho\left(t^{\prime}\right)$, and if $A \subseteq B, \rho\left(\gamma_{B}\right) \leq_{n} \rho\left(\gamma_{A}\right)$. The case $n=0$ is trivially

[^0]verified. Let $\alpha \leq \beta \in C^{\prime \prime}$. By proposition 10, $C^{\prime \prime}$ verifies the properties of table 2. Thus $\rho(\alpha)(\epsilon)=\left(\Uparrow_{C^{\prime \prime}} \alpha\right)(\epsilon) \leq_{\mathcal{K}}\left(\Uparrow_{C^{\prime \prime}} \beta\right)(\epsilon)=\rho(\alpha)(\epsilon)$. Let us consider $l \in$ $a(\rho(\alpha)(\epsilon)) \cap a(\rho(\beta)(\epsilon))$. By properties 3) and 4) of table 2, $\rho(\alpha) / l=\gamma_{A}$ for some $A$ and $\rho(\beta) / l=\gamma_{B}$ for some $B$ with $B \subseteq A$. By induction $\rho\left(\gamma_{A}\right) \leq_{n} \rho\left(\gamma_{B}\right)$. Thus $\rho(\alpha) \leq_{n+1} \rho(\beta)$. Similarly, if $A \subseteq B, \rho\left(\gamma_{B}\right) \leq_{n+1} \rho\left(\gamma_{A}\right)$. Let $\alpha \leq t \in C^{\prime \prime}$. We have $\rho(\alpha)=\rho(t)$ so $\rho(\alpha) \leq_{n+1} \rho(t)$. Let $t \leq \alpha \in C^{\prime \prime}$. Using property 7) of table 2, we have $\rho(t)(\epsilon) \leq \mathcal{K} \rho(\alpha)(\epsilon)$. Moreover, for all label $l \in a(r h o(t)(\epsilon)) \cap a(\rho(\alpha)(\epsilon))$, $t / l \leq \Uparrow_{C^{\prime \prime}} \alpha \in C$, thus $\rho(t) / l \leq_{n} \rho(\alpha) / l$. So $\rho(t) \leq_{n+1} \rho(\alpha)$. So for all constraint $t \leq t^{\prime} \in C^{\prime \prime}, \rho(t) \leq \rho\left(t^{\prime}\right)$, thus $\rho \neq C^{\prime \prime}$. Moreover, since $\mathcal{L}^{-}=\emptyset$, there are only variables of the form $\gamma_{A}$ occuring on the right hand side of the system defining $\rho$, thus, by using property 6 ) of table 2 , the fact that $C$ is pre-closed with bounds in $\mathcal{K}$ and that rule (Down $\perp$ ) can not be applied to $C^{\prime \prime}$, we obtain that for all $\alpha \in V(C), \rho(\alpha) \in \mathcal{T}(\mathcal{K})$. Then we check by induction that for all substitution $\rho^{\prime} \models C^{\prime \prime}$, for all variable $\alpha \in V\left(C^{\prime \prime}\right)$, for all $n \in \mathbb{N}, \rho^{\prime}(\alpha) \leq_{n} \rho(\alpha)$, i.e. $\rho^{\prime}(\alpha) \leq \rho(\alpha)$. $\rho$ is thus a maximal solution of $C$ in $\mathcal{T}(\mathcal{K})$.

Corollary 1. A pre-closed constraint system $C$ is satisfiable in infinite types in and only if it is satisfiable in regular types.

Proof. The substitution used in the proof theorem 6 above is a regular type.
Note that in the case where some type constructors have contravariant labels (in $\mathcal{L}^{-}$), there may not be one maximal solution. For example, let us take $\mathcal{K}=$ $\{$ int, float,$\rightarrow\}$ with int $\leq$ float $, \mathcal{L}^{+}=\{r\}, \mathcal{L}^{-}=\{a\}, a($ int $)=a($ float $)=\emptyset$, $a(\rightarrow)=\{a, r\}$. Let $C=\{\alpha \rightarrow \alpha \leq \beta, \beta \leq \alpha \rightarrow \alpha$, int $\leq \alpha, \alpha \leq$ float $\}$. $C$ is preclosed and has two incomparable solutions, namely $\rho(\beta)=$ int $\rightarrow$ int, $\rho(\alpha)=$ int and $\rho^{\prime}(\beta)=$ float $\rightarrow$ float, $\rho^{\prime}(\alpha)=$ float.

By combining Pottier's simplification algorithm with the rules of table 3, we thus obtain a simplification algorithm for pre-closed systems in quasi-lattices. Moreover, in the case where all type constructors are covariant ( $\mathcal{L}^{-}=\emptyset$ ), it gives maximal and minimal solutions. The combination of the above algorithm with the pre-closure algorithm of section 3.2, gives a set of maximal and minimal solutions for non-pre-closed systems in quasi-lattices with a finite number of extrema each with an empty arity.

## 5 Applications

A first application of the previous algorithm for solving subtyping constraints in quasi-lattices is our type system TCLP [3, 1] for constraint logic programs. This algorithm allows us to remove the emtpy type $\perp$ from the type structure. It also allows us to use type structures that are more complex than the ones used until now, especially type structures without the type term used for metaprogramming. The constraint solving algorithm of the previous section has been implemented in CHR by a simple modification of our previous constraint solver of TCLP in lattices [2]. The implementation in CHR shows very good practical performances [5].

A second application can be found in the framework of type inference with subtyping for languages a la ML. In [6], Pottier uses subtyping constraints for type inference in a variant of ML with rows. However, in a lattice, the bottom element $\perp$ denotes the empty type, hence a function typed by $\perp \rightarrow \tau$ cannot be applied to any argument. The algorithm for solving subtyping constraints described in this paper allows one to use the quasi-lattice obtained by removing the $\perp$ element from the lattice as a type structure. A type error can then be produced instead of a typing with the empty type.

## 6 Conclusion

We have studied general forms of non-structural subtyping relations in the quasilattice of infinite (regular) types formed over a quasi-lattice of type constructors. We have shown the decidability of the satisfiability problem for subtyping constraints in quasi-lattices, by generalizing Trifonov and Smith's algorithm for testing the satisfiability of subtyping constraints in lattices to the case of quasilattices, with a time complexity in $O\left(m^{v} M^{v} n^{3}\right)$ where $m$ (resp. $M$ ) is the number of minimal (resp. maximal) elements of the quasi-lattice and $v$ the number of unbounded variables. It is worth noting that the complexity of this algorithm is in $O\left(n^{3}\right)$ for constraint systems where all variables are bounded. In the general case we have shown the NP-completeness of this problem.

We have also extended Pottier's algorithm for computing solutions to the case of quasi-lattices, and have shown that the computed solutions are minimal (resp. maximal) solutions when all type constructors are covariant. Finally we have mentionned some applications of these algorithms to type inference problems in constraint logic programming and in functional programming languages.

As for future work, one can mention several problems left open in this paper. We have already mentionned the case where the extrema of the quasi-lattice of constructors have a non empty arity. The decidability of constraint satisfiability in finite types is also an open problem. In the homogeneous case (i.e. type constructors in a subtype relation have the same arity), Frey has shown that this problem is Pspace complete in arbitrary posets [4].

## References

1. E. Coquery. Tclp: a generic type checker for constraint logic programs. http://pauillac.inria.fr/~coquery/tclp/.
2. E. Coquery and F. Fages. From typing constraints to typed constraint systems in CHR. In Third workshop on Rule-based Constraint Reasoning and Programming, CP'01, Cyprus, December 2001.
3. F. Fages and E. Coquery. Typing constraint logic programs. Theory and Practice of Logic Programming, 1, November 2001.
4. A. Frey. Satisfying subtype inequalities in polynomial space. In Proceedings of the 1997 International Static Analysis Symposium, number 1302 in LNCS, 1997.
5. Th. Frühwirth. Theory and practice of constraint handling rules. Journal of Logic Programming, Special Issue on Constraint Logic Programming, 37(1-3):95138, October 1998.
6. F. Pottier. A versatile constraint-based type inference system. Nordic Journal of Computing, 7(4):312-347, November 2000
7. F. Pottier. Simplifying subtyping constraints: a theory. Information \& Computation, 170(2):153-183, November 2001.
8. V. Pratt and J. Tiuryn. Satisfiability of inequalities in a poset. Fundamenta Informaticae, 28(1-2):165-182, 1996.
9. G. Smolka. Logic Programming over Polymorphically Order-Sorted Types. PhD thesis, Universitat Kaiserslautern, 1989.
10. V. Trifonov and S. Smith. Subtyping constrained types. In Proc. 3rd Int'l Symposium on Static Analysis, number 1145 in LNCS, pages 349-365. Springer, 1996.

## A Proofs of section 2

Lemma 4. Let $S \neq \emptyset \subseteq \mathcal{T}$. If $t \in \downarrow S$, then $t(\epsilon) \in \downarrow_{U L \downarrow S} \sqcap\{s(\epsilon) \mid s \in S\}$. Similary, if $t \in \uparrow(S)$, then $t(\epsilon) \in \uparrow_{U L \uparrow S} \sqcup\{s(\epsilon) \mid s \in S\}$.

Proof. One can check that $t(\epsilon) \leq \mathcal{K} \sqcap\{s(\epsilon) \mid s \in S\}$ and that $a(t(\epsilon)) \cap a(\sqcap\{s(\epsilon) \mid$ $s \in S\}) \subseteq U L \downarrow S$.

Corollary 2. Let $S \neq \emptyset \subseteq \mathcal{T}$. If $\downarrow S \neq \emptyset$ (resp. $\uparrow S \neq \emptyset$ ) then $\sqcap_{\epsilon} S$ (resp. $\sqcup_{\epsilon} S$ ) is defined.

Proof. Since $\downarrow S \neq \emptyset$, there exists $t \in \downarrow S$ and, by lemma $4, t(\epsilon) \in \downarrow_{U L \downarrow S} \sqcap\{s(\epsilon) \mid s \in S\}$. Thus $\downarrow_{U L \downarrow S} \sqcap\{s(\epsilon) \mid s \in S\} \neq \emptyset$ and by definition, it has an upper bound. Thus it has a least upper bound $\Pi_{\epsilon} S$.

Corollary 3. Let $S \neq \emptyset \subseteq \mathcal{T}$. If $\downarrow S \neq \emptyset$ (resp. $\uparrow S \neq \emptyset$ ) then for all $n \in \mathbb{N}, \sqcap_{n} S$ (resp. $\sqcup_{n} S$ ) is defined.

Proof. We show it by induction over $n$ : the case $n=0$ is trivial. For the case $n+1$, let us consider $\Pi_{n+1} S$ (the case $\sqcup_{n+1} S$ is similar). By corollary $2, \Pi_{\epsilon} S$ is defined. Let $l \in a\left(\Pi_{\epsilon} S\right)$, by proposition $1 l \in S L \downarrow_{U L \downarrow S} k \subseteq U L \downarrow S$. Thus $S / l$ has a lower bound (or an upper bound, depending of the sign of $l$ ). Thus, by induction, $\left(\square_{n+1} S\right) / l$ is defined.

Lemma 5. Let $S \neq \emptyset \subseteq \mathcal{T}$. $\forall t \in \mathcal{T}$, if $t \in \downarrow S$ (resp. $t \in \uparrow S$ ) then $t(\epsilon) \leq \mathcal{K} \sqcap_{\epsilon} S$ (resp. $\sqcup_{\epsilon} S \leq \mathcal{K} t(\epsilon)$ ).

Proof. Let $t \in \downarrow S$. By lemma $4, k \in \downarrow_{U L \downarrow S} \sqcap\{s(\epsilon) \mid s \in S\}$. Thus, by definition of $\Pi_{\epsilon} S, k \leq \mathcal{K}^{\prime} \Pi_{\epsilon}(S)$. The proof is similar for $\sqcup_{\epsilon} S$.

Corollary 4. Let $S \neq \emptyset \subseteq \mathcal{T} . \forall t \in \downarrow S$ (resp. $t \in \uparrow S$ ), $\forall n \in \mathbb{N} t \leq_{n} \sqcap_{n} S$ (resp. $\Pi_{n} S \leq_{n} t$ ).

Proof. By induction over $n$. The case $n=0$ is trivial. Let us consider $t \leq_{n+1}$ $\sqcap_{n+1} S$. Since $t \in \downarrow S$ with $t(\epsilon)=k$ then, by lemma $5, k \leq \mathcal{\kappa} \Pi_{\epsilon} S$. Let $l \in a(t(\epsilon)) \cap$ $a\left(\Pi_{\epsilon} S\right)$. If $l \in \mathcal{L}^{+}$, then , since $t \in \downarrow S$, by proposition $3, t / l \in \downarrow(S / l)$. By induction, we obtain $t / l \leq_{n} \Pi_{n}(S / l)=\left(\square_{n+1} S\right) / l$. The case $l \in \mathcal{L}^{-}$, is similar. Thus $t \leq_{n+1} \sqcap_{n+1} S$. Similary, if $t \in \uparrow S$, then $t \geq_{n+1} \sqcup_{n+1} S$.

Lemma 6. Let $S \neq \emptyset \subseteq \mathcal{T}$. If $\downarrow S \neq \emptyset$ (resp. $\uparrow S \neq \emptyset$ ) then $\forall s \in S, \sqcap_{\epsilon} S \leq \mathcal{K} s(\epsilon)$ (resp. $s(\epsilon) \leq \mathcal{K} \sqcup_{\epsilon} S$ ).

Proof. Since $\exists t \in \downarrow S, \Pi_{\epsilon} S$ is defined. We note $K_{S}=\{s(\epsilon) \mid s \in S\}$. $\Pi_{\epsilon} S=$ $\sqcup\left(\downarrow_{U L_{\downarrow} S} \sqcap K_{S}\right)$. Moreover $\forall s \in S, \sqcap K_{S} \leq_{\mathcal{K}} s(\epsilon)$. But, by definition, $\Pi_{\epsilon} S \leq \mathcal{K} \sqcap K_{S}$. Thus $\forall s \in S, \sqcap_{\epsilon} S \leq \mathcal{K}^{s}(\epsilon)$. The proof is similar for $\sqcup_{\epsilon} S$.

Corollary 5. Let $S \neq \emptyset \subseteq \mathcal{T}$. If $\downarrow S \neq \emptyset$ (resp. $\uparrow S \neq \emptyset$ ) then for all $n \in \mathbb{N}$, $\forall s \in S, \sqcap_{n} S \leq_{n} s\left(\right.$ resp. $\left.\sqcup_{n} S \geq_{n} s\right)$.

Proof. By induction over $n$. The case $n=0$ is trivial. Let $s \in S$. Let us consider $\Pi_{n+1} S \leq_{n+1} s$. We have $\left(\Pi_{n+1} S\right)(\epsilon)=\Pi_{\epsilon} S$, and, by lemma 6, $\Pi_{\epsilon} S \leq_{\mathcal{K}} s(\epsilon)$. Let $l \in a\left(\Pi_{\epsilon} S\right) \cap a(s(\epsilon))$. By proposition $1, l \in S L \downarrow_{U L \downarrow S} k \subseteq U L \downarrow(S)$. If $l \in \mathcal{L}^{+}$, $\left(\square_{n+1} S\right) / l=\Pi_{n}(S / l)$. We have $s / l \in S / l$. Thus, by induction, $\left(\square_{n+1} S\right) / l=$ $\Pi_{n}(S / l) \leq_{n} s / l$. The case $l \in \mathcal{L}^{-}$is symetrical. Similary, $\sqcup_{n+1} S \geq_{n+1} s$.

The following lemma tells that if $m \leq n$ then $\Pi_{m} S$ is an approximation of $\Pi_{n} S$ until depth $m$.

Lemma 7. Let $S \neq \emptyset \subseteq \mathcal{T}$ such that $\downarrow S \neq \emptyset$ (resp. $\uparrow S \neq \emptyset$ ). Let $m, n \in \mathbb{N}$, $m \leq n, \forall w \in \mathcal{L}^{*}$, if $|w|<m$ then $w \in \operatorname{dom}\left(\Pi_{m} S\right) \Leftrightarrow w \in \operatorname{dom}\left(\Pi_{n} S\right)$ and $\left(\Pi_{m} S\right)(w)=\left(\square_{n} S\right)(w)$ (resp. $w \in \operatorname{dom}\left(\sqcup_{m} S\right) \Leftrightarrow w \in \operatorname{dom}\left(\sqcup_{n} S\right)$ and $\left.\left(\sqcup_{m} S\right)(w)=\left(\sqcup_{n} S\right)(w)\right)$.

Proof. By induction over $m$ : the case $n=0$ is trivial because there is no $w$ such that $|w|<0$. Let us consider the case $m+1 \leq n+1$. If $w=\epsilon$, by definition $\epsilon \in \operatorname{dom}\left(\sqcap_{m+1} S\right), \epsilon \in \operatorname{dom}\left(\sqcap_{n+1} S\right)$ and $\left(\sqcap_{m+1} S\right)(\epsilon)=\sqcap_{\epsilon} S=\left(\sqcap_{n+1} S\right)(\epsilon)$. If $w=l . w^{\prime}$, since $\left(\sqcap_{m+1} S\right)(\epsilon)=\Pi_{\epsilon} S=\left(\Pi_{n+1} S\right)(\epsilon)$ with $a\left(\sqcap_{\epsilon} S\right)=L$, we have $l \in L$. If $l \in \mathcal{L}^{+}$, then $\left(\Pi_{m+1} S\right) / l=\Pi_{m}(S / l)$ and $\left(\Pi_{n+1} S\right) / l=\Pi_{n}(S / l)$. By induction, $w^{\prime} \in \operatorname{dom}\left(\Pi_{m}(S / l)\right) \Leftrightarrow w^{\prime} \in \operatorname{dom}\left(\Pi_{n}(S / l)\right)$. Since $l \in L$, we thus have $\left.w=l . w^{\prime} \in \operatorname{dom}\left(\square_{m+1} S\right)\right) \Leftrightarrow w \in \operatorname{dom}\left(\square_{n+1} S\right)$ ). By induction, we have $\left(\square_{m}(S / l)\right)\left(w^{\prime}\right)=\left(\sqcap_{n}(S / l)\right)\left(w^{\prime}\right)$, thus $\left(\sqcap_{m+1} S\right)\left(l . w^{\prime}\right)=\left(\left(\sqcap_{m+1} S\right) / l\right)\left(w^{\prime}\right)=$ $\left(\square_{m}(S / l)\right)\left(w^{\prime}\right)=\left(\square_{n}(S / l)\right)\left(w^{\prime}\right)=\left(\left(\square_{n+1} S\right) / l\right)\left(w^{\prime}\right)=\left(\square_{n+1} S\right)\left(l . w^{\prime}\right)$. The case $l \in \mathcal{L}^{-}$is symetrical. The proof is similar for $\sqcup_{m+1}$

Proof (of proposition 4). Since $\downarrow S \neq \emptyset$, by corollary 3 , for all $n \in \mathbb{N}, \square_{n} S$ is defined. Now we show that $\square_{\mathcal{T}} S$ is a type:

- Let $w . l \in \operatorname{dom}\left(\sqcap_{\mathcal{T}} S\right)$. We have $w . l \in \operatorname{dom}\left(\sqcap_{|w . l|+1} S\right)$. Thus $w \in \operatorname{dom}\left(\sqcap_{|w . l|+1} S\right)$. By lemma 7 we obtain $w \in \operatorname{dom}\left(\Pi_{|w|+1} S\right)$, thus $w \in \operatorname{dom}\left(\Pi_{\mathcal{T}} S\right)$.
$-\epsilon \in \operatorname{dom}\left(\Pi_{1} S\right)$, thus $\epsilon \in \operatorname{dom}\left(\Pi_{\mathcal{T}} S\right)$
- Let $w \in \operatorname{dom}\left(\Pi_{\mathcal{T}} S\right)$ such that $\left(\Pi_{\mathcal{T}} S\right)(w)=k=\left(\Pi_{|w|+1} S\right)(w)$. We have $w . l \in \operatorname{dom}\left(\Pi_{\mathcal{T}} S\right) \Leftrightarrow w . l \in \operatorname{dom}\left(\Pi_{|w . l|+1} S\right)$. By lemma $7,\left(\Pi_{|w . l|+1} S\right)(w)=$ $\left(\Pi_{|w|+1} S\right)(w)=k$. Thus $w . l \in \operatorname{dom}\left(\Pi_{|w . l|+1} S\right) \Leftrightarrow l \in a(k)$.

Proposition 11. Let $S \subseteq \mathcal{T}$ with $S \neq \emptyset$ such that $\downarrow S \neq \emptyset . \forall l \in a\left(\left(\sqcap_{\mathcal{T}} S\right)(\epsilon)\right)$ :

- Either $l \in \mathcal{L}^{+}$and $\left(\Pi_{\mathcal{T}} S\right) / l=\Pi_{\mathcal{T}}(S / l)$
- Or $l \in \mathcal{L}^{-}$and $\left(\Pi_{\mathcal{T}} S\right) / l=\sqcup_{\mathcal{T}}(S / l)$

Similary, if $\uparrow S \neq \emptyset$ then $\forall l \in a\left(\left(\sqcup_{\mathcal{T}} S\right)(\epsilon)\right)$ :

- Either $l \in \mathcal{L}^{+}$and $\left(\square_{\mathcal{T}} S\right) / l=\sqcup_{\mathcal{T}}(S / l)$
- Or $l \in \mathcal{L}^{-}$and $\left(\square_{\mathcal{T}} S\right) / l=\square_{\mathcal{T}}(S / l)$

Proof. $\left(\left(\Pi_{\mathcal{T}} S\right) / l\right)(w)=\left(\Pi_{\mathcal{T}} S\right)(l . w)=\left(\Pi_{|l . w|+1} S\right)(l . w)$. If $l \in \mathcal{L}^{+}$then $\left(\Pi_{|l . w|+1} S\right) / l=$ $\Pi_{|w|+1}(S / l)$. In this case $\left(\Pi_{|l . w|+1} S\right)(l . w)=\left(\left(\Pi_{|l . w|+1} S\right) / l\right)(w)=\left(\Pi_{|w|+1}(S / l)\right)(w)=$ $\left(\Pi_{\mathcal{T}}(S / l)\right)(w)$. The case $l \in \mathcal{L}^{-}$is symetrical. We show in the same way the proposition for $\sqcup_{\mathcal{T}} S$.

Lemma 8. Let $S \neq \emptyset \subseteq \mathcal{T}$ such that $\downarrow S \neq \emptyset$ (resp. $\uparrow S \neq \emptyset$ ). $\forall n \in \mathbb{N}, \sqcap_{n} S \leq_{n}$ $\sqcap_{\mathcal{T}} S$ and $\sqcap_{n}(S) \geq_{n} \sqcap_{\mathcal{T}} S$ (resp. $\sqcup_{n} S \leq_{n} \sqcup_{\mathcal{T}} S$ and $\sqcup_{n} S \geq_{n} \sqcup_{\mathcal{T}} S$ ).

Proof. By induction over $n$ : the case $n=0$ is trivial. Let us consider the case $\sqcap_{\mathcal{T}} S \leq_{n+1} \sqcap_{n+1} S$. By lemma $7,\left(\sqcap_{\mathcal{T}} S\right)(\epsilon)=\left(\sqcap_{1} S\right)(\epsilon)=\left(\sqcap_{n+1} S\right)(\epsilon)$. We have $\left(\square_{\mathcal{T}} S\right)(\epsilon) \leq_{\mathcal{K}}\left(\square_{n+1} S\right)(\epsilon)$ and $\left(\sqcap_{\mathcal{T}} S\right)(\epsilon) \geq_{\mathcal{K}}\left(\sqcap_{n+1} S\right)(\epsilon)$. Let $l \in a\left(\left(\square_{\mathcal{T}} S\right)(\epsilon)\right)$. If $l \in \mathcal{L}^{+}$, then, by proposition $11,\left(\square_{\mathcal{T}} S\right) / l=\Pi_{\mathcal{T}}(S / l)$. We also have $\left(\square_{n+1} S\right) / l=$ $\Pi_{n}(S / l)$. By induction $\Pi_{\mathcal{T}}(S / l) \leq_{n} \Pi_{n}(S / l)$ and $\Pi_{\mathcal{T}}(S / l) \geq_{n} \Pi_{n}(S / l)$. Thus $\left(\sqcap_{n+1} S\right) / l \leq_{n}\left(\sqcap_{\mathcal{T}} S\right) / l$ and $\left(\square_{n+1} S\right) / l \geq_{n}\left(\sqcap_{\mathcal{T}} S\right) / l$. The case $l \in \mathcal{L}^{-}$is symetrical. Thus $\sqcap_{\mathcal{T}} S \leq_{n+1} \sqcap_{n+1} S$ et $\sqcap_{\mathcal{T}} S \geq_{n+1} \sqcap_{n+1} S$. Similary $\sqcup_{\mathcal{T}} S \leq_{n+1} \sqcup_{n+1} S$ and $\sqcup_{\mathcal{T}} S \geq_{n+1} \sqcup_{n+1} S$.

Proof (of proposition 5). Let $s \in S$. Let $n \in \mathbb{N}$. By corollary $5, \Pi_{n} S \leq_{n} s$. By lemma $8, \sqcap_{\mathcal{T}} S \leq_{n} \sqcap_{n} S$, thus $\sqcap_{\mathcal{T}} S \leq_{n} s$. Thus $\sqcap_{\mathcal{T}} S \leq s$. Similary $s \leq \sqcup_{\mathcal{T}} S$.

Proof (of proposition6). Let $t \in \downarrow S$. Let $n \in \mathbb{N}$. By corollary $4, t \leq_{n} \Pi_{n} S$ and by lemma $8, \sqcap_{n} S \leq_{n} \Pi_{\mathcal{T}} S$, thus $t \leq_{n} \Pi_{\mathcal{T}} S$. Thus $t \leq \Pi_{\mathcal{T}} S$. Similary, if $t \in \uparrow S$, then $\sqcup_{\mathcal{T}} S \leq t$.

Proof (of proposition 7). Let $S$ be the set of subterms of $t_{1}$ and $t_{2}$. Let $S_{\Pi}=$ $\left\{u \sqcap_{\mathcal{T} v} v \mid u \in S \wedge v \in S\right\}$ and $S_{\sqcup}=\left\{u \sqcup_{\mathcal{T}} v \mid u \in S \wedge v \in S\right\}$.

We show that for all $t \in S_{\sqcap} \cup S_{\sqcup}$, for all $w \in \operatorname{dom}(t), t / w \in S_{\sqcap} \cup S_{\sqcup}$, by induction over $w$ : if $w=\epsilon$ then $t / \epsilon=t \in S_{\sqcap} \cup S_{\sqcup}$. If $w=l . w^{\prime}$ : Let us assume that $t=u \Pi_{\mathcal{T}} v$, with $u \in S$ and $v \in S$ (the proof is similar for $t=u \sqcup_{\mathcal{T}} v$ ). Let us also assume that $l \in \mathcal{L}^{+}$(the proof is similar for $\in \mathcal{L}^{-}$). By proposition 11, $t / l=\left(\sqcap_{\mathcal{T}}\{u, v\}\right) / l=\sqcap_{\mathcal{T}}(\{u, v\} / l)$.

- If $u / l$ is defined but not $v / l$ then $t / l=u / l \in S \subseteq S_{\Pi} \cup S_{\sqcup}$
- If $v / l$ is defined but not $u / l$ then $t / l=v / l \in S \subseteq S_{\square} \cup S_{\sqcup}$
- If $u / l$ and $v / l$ are both defined, then $u / l \in S$ and $v / l \in S$ and $t / l=$ $u / l \sqcap_{\mathcal{T}} v / l \in S_{\sqcap} \cup S_{\sqcup}$.

Thus $t / l \in S_{\sqcap} \cup S_{\sqcup}$. By induction, $(t / l) / w^{\prime} \in S_{\sqcap} \cup S_{\sqcup}$, thus $t / w \in S_{\Pi} \cup S_{\sqcup}$.
Since $t_{1}$ and $t_{2}$ are regular types, $S$ is finite and thus $S_{\sqcap} \cup S_{\sqcup}$ is finite. However $t_{1} \sqcap_{\mathcal{T}} t_{2} \in S_{\sqcap}$, all its subterms are in $S_{\sqcap} \cup S_{\sqcup}$, thus there is only a finite number of such terms, thus $t_{1} \square_{\mathcal{T}} t_{2}$ is a regular type. Similary $t_{1} \sqcup_{\mathcal{T}} t_{2}$ a regular type.

## B Proofs of section 3

Proof (of lemma 2). Since all terms occuring in $C$ are small terms, $(\Downarrow A) / l \subseteq$ $V(C)$ and $(\Uparrow B) / l \subseteq V(C)$. Moreover, since $C$ is closed and $\forall \alpha \in A, \forall \beta \in$ $B, \alpha \leq \beta \in C$, for all $t \in \Downarrow A, t^{\prime} \in \Uparrow B, \operatorname{dec}\left(t \leq t^{\prime}\right)$ is defined and included in $C$. Let $l \in a(\operatorname{sol}(A, B))$ and $l \in \mathcal{L}^{+}$. Let $\alpha \in(\Downarrow A) / l$ and $\beta \neq \alpha \in(\Uparrow B) / l$. There exists $t \in \Downarrow A$ such that $t / l=\alpha$ and $t^{\prime} \in \Uparrow B$ such that $t^{\prime} / l=\beta$. Since $\operatorname{dec}\left(t \leq t^{\prime}\right) \subseteq C$ and $l \in a(t(\epsilon)) \cap a\left(t^{\prime}(\epsilon)\right), \alpha \leq \beta \in C$. Similary, if $l \in \mathcal{L}^{-}$, $\forall \beta \in(\Uparrow B) / l, \forall \alpha \in(\Downarrow A) / l, \beta \leq \alpha \in C$.

Lemma 9. Let $k_{1}, k_{2}, k_{3}, k_{4} \in \mathcal{K}$ such that $k_{1} \leq \mathcal{K} k_{2},, k_{1} \leq \mathcal{K} k_{3}, k_{2} \leq \mathcal{K} k_{4}$ and $k_{3} \leq \mathcal{K} k_{4}$. Then $\sqcup\left(\downarrow_{a\left(k_{1}\right)} k_{2}\right) \leq \mathcal{K} \sqcup\left(\downarrow_{a\left(k_{3}\right)} k_{4}\right)$.

Proof. We note $k=\sqcup\left(\downarrow_{a\left(k_{3}\right)} k_{4}\right)$ and $k^{\prime}=\sqcup\left(\downarrow_{a\left(k_{1}\right)} k_{2}\right), L_{i}=a\left(k_{i}\right), L=a(k)$ and $L^{\prime}=a\left(k^{\prime}\right)$. By proposition 1, we have $L^{\prime}=S L \downarrow_{L_{1}} k_{2} \subseteq L_{2} \cap L_{1}$. Since $k_{1} \leq \mathcal{K} k_{3} \leq \mathcal{K} k_{4}, L_{1} \cap L_{4} \subseteq L_{3}$. Thus $L^{\prime} \cap L_{4} \subseteq L_{2} \cap L_{1} \cap L_{4} \subseteq L_{2} \cap L_{3} \cap L_{4} \subseteq L_{3} \cap L_{4}$. Moreover $k^{\prime} \leq_{\mathcal{K}} k_{2} \leq_{\mathcal{K}} k_{4}$, thus $k^{\prime} \in \downarrow_{L_{3}} k_{4}$, d'o $k^{\prime} \leq_{\mathcal{K}} k$.

Proof (of lemma 3). Let $U_{B}=\{t(\epsilon) \mid t \in \Uparrow B\}, D_{A}=\{t(\epsilon) \mid t \in \Downarrow A\}, U_{F}=$ $\{t(\epsilon) \mid t \in \Uparrow F\}, D_{E}=\{t(\epsilon) \mid t \in \Downarrow E\}$. We have $D_{A} \subseteq D_{E}$ and $U_{F} \subseteq U_{B}$. Thus $\sqcup D_{A} \leq \mathcal{K} \sqcup D_{E}$ and $\sqcap U_{B} \leq_{\mathcal{K}} \sqcap U_{F}$. Moreover $\sqcup D_{A} \leq \mathcal{K} \sqcap U_{B}$ and $\sqcup D_{E} \leq \mathcal{K} \sqcap U_{F}$. Thus, by lemma 9 ,
$\operatorname{sol}(A, B)=\sqcup\left(\downarrow_{a\left(\cup D_{A}\right)} \sqcap U_{B}\right) \leq \mathcal{K} \sqcup\left(\downarrow_{a\left(\sqcup D_{E}\right)} \sqcap U_{F}\right)=\operatorname{sol}(E, F)$.

## C Proofs of section 4

Proof (of proposition 10 (more detailed)). Properties 1), 2), 3), 6) and 8) are not modified by the application of the rules of table 3 . Let us consider the application of the rule (Down $\perp$ ): $C^{\prime} \rightarrow^{*} D_{1}=D, \alpha \leq t \rightarrow D_{2}=D, \alpha \leq t^{\prime}$. By proposition $8, D, \alpha \leq t^{\prime}$ is satisfiable. Thus there exists $\rho$ such that $t_{i}=\rho\left(\Downarrow_{D_{1}} \alpha\right) \leq t_{\alpha}=$ $\rho(\alpha) \leq \rho(t)$ and $t_{i} \leq t_{\alpha} \leq \rho\left(t^{\prime}\right)$. Thus we have $t_{i}(\epsilon) \leq \mathcal{K} t_{\alpha}(\epsilon) \leq \mathcal{K} t^{\prime}(\epsilon)$, thus $\left(\Downarrow_{D_{2}} \alpha\right)(\epsilon) \leq \mathcal{K}\left(\Uparrow_{D_{2}} \alpha\right)(\epsilon)$. Since for all $l^{\prime} \in a\left(t^{\prime}(\epsilon)\right), t^{\prime} / l^{\prime}=t / l$, property 7$)$ is preserved (the proof for the other rules is similar). Let us show that $C^{\prime \prime}$ verifies property 4 ). Let us consider the following application of the rule (Down $\perp$ ): $C^{\prime} \rightarrow^{*} D_{1}=D, \beta \leq t \rightarrow D_{2}=D, \beta \leq t^{\prime}$. This transition can break property 4). Let us take $\alpha \leq \beta \in D_{1}$ (other cases are similar) and let us show that is it possible to make a transition (Down $\perp$ ) to reestablish property 4) w.r.t. $\alpha$ and $\beta$. The only way property 4) can be broken by (Down $\perp$ ) is that $k_{\alpha}=\left(\Uparrow_{D_{2}} \alpha\right)(\epsilon) \not \mathcal{K}_{\mathcal{K}} t^{\prime}(\epsilon)$. Let $l$ be the label use when applying the rule (Down $\perp$ ). We pose $t^{\prime}(\epsilon)=k_{\beta}^{\prime}$ and $t(\epsilon)=k_{\beta}$. If $l \notin a\left(k_{\alpha}\right)$, then $k_{\alpha} \in \downarrow_{\left(a\left(k_{\beta}\right) \backslash\{l\}\right)} k_{\beta}$ and $k_{\alpha} \leq \mathcal{K} k_{\beta}^{\prime}$. Otherwise,we have $t / l \neq \beta$, because $t \neq \perp$. Thus property 4 ) is verified for $\left(\Uparrow_{D_{2}} \alpha\right) / l$ and $\left(\Uparrow_{D_{1}} \beta\right) / l$, thus $\Uparrow_{D_{2}}\left(\left(\Uparrow_{D_{2}} \alpha\right) / l\right)=\perp$ and we can apply rule (Down $\perp$ ) upon $D_{2}$ and $\alpha$, thus obtaining $D_{3}$. Thus we have $l \notin a\left(\left(\Uparrow_{D_{3}} \alpha\right)(\epsilon)\right)$, and thus $\left(\Uparrow_{D_{3}} \alpha\right)(\epsilon) \leq \mathcal{K} k_{\beta}^{\prime}$. Thus we can always apply rule (Down $\perp$ ) to reestablish property 4 ). Similary, this can be done for the other rules and for property 5). Since by proposition 9 , the rewriting system is convergent et terminates, $C^{\prime \prime}$ verifies the properties 4) et 5).


[^0]:    ${ }^{1}$ a more detailed version of this proof is available in the appendix

