Technical Report

UCAM-CL-TR-911
ISSN 1476-2986

Number 911

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Proofs for 'Verifying Spatial Properties of Array Computations'

Dominic Orchard, Mistral Contrastin,
Matthew Danish, Andrew Rice

September 2017

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

Technical reports published by the University of Cambridge
Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/techreports/

ISSN 1476-2986

Proofs for 'Verifying Spatial Properties of Array Computations'

Dominic Orchard Mistral Contrastin Matthew Danish
Andrew Rice

Abstract

This technical report provides accompanying proofs for the paper: *Verifying Spatial Properties of Array Computations*. We show three properties of the lattice model of the stencil specification language. 1) that it is sound with respect to the equational theory of region specifications; 2) that is it sound with respect to the theory of region approximation; 3) that the inference algorithm is sound. We further derive useful identities on the specification language and properties of Union Normal Form—the data structure used to implement the model. Core definitions from the paper are restated here to make the overall report more self contained.

Numbering for definitions, lemmas, propositions, theorems, and sections is aligned to the paper [1] for easy cross-reference.

Definition 1 (Induction variables)**.** An integer variable is an *induction variable* if it is the control variable of a "*for*" loop (`do` in Fortran), incremented by 1 per iteration. A variable is interpreted as an induction variable only within the scope of the loop body. Throughout, syntactic variables `i`, `j`, `k` range over induction variables.

Definition 2 (Array subscripts and indices)**.** An *array subscript*, denoted $a(\bar{e})$, is an expression which indicates the element of an N-dimensional array a at the N-dimensional *index* $\bar{e}$, specified by a comma-separated sequence of integer expressions $e_1, \ldots, e_N$. We also refer to each one of these e_i as an index. An index e_i is called *relative* if the expression involves an induction variable. An *absolute index* is an integer expression which is constant with respect to the enclosing loop, *i.e.*, it does not involve an induction variable.

Definition 3 (Origin)**.** For an array subscript $a(\bar{e})$, the index $\bar{e}$ is called an *origin index*, if all $e \in \bar{e}$ are naked induction variable expressions, *e.g.*, `a(i, j)`. We refer to this as the "origin" since the indices are offset by 0 in each dimension relative to the induction variables.

Definition 4 (Neighbourhood and affine index)**.** For an array subscript $a(\bar{e})$, an index $e \in \bar{e}$ is a *neighbourhood index* (or *neighbour*) if e is of the form $e \equiv$ `i`, $e \equiv$ `i + c`, or $e \equiv$ `i - c`, where `c` is an integer constant and `i` is an induction variable. That is, a neighbourhood index is a constant offset/translation of an induction variable. (The relation $\equiv$ here identifies terms up-to commutativity of $+$ and the inverse relationship of $+$ and $-$ *e.g.*, `(-b) + i` $\equiv$ `i - b`).

An *affine index* is an index expression of the form `a*i + b`, where `a` and `b` are constants.

4 Equational & approximation theories

Proposition 1 (Centered approximation)**.** *For any dimension n, depth k, and pointed attribute* `p`*:*

$$\texttt{forward(dim=}n\texttt{,depth=}k\texttt{,p)} \preceq \texttt{centered(dim=}n\texttt{,depth=}k\texttt{,p)}$$
$$\texttt{backward(dim=}n\texttt{,depth=}k\texttt{,p)} \preceq \texttt{centered(dim=}n\texttt{,depth=}k\texttt{,p)}$$

Proof. Proof of the first inequality:

$$\begin{aligned}
&\texttt{fwd(dim=}n\texttt{,depth=}k\texttt{,p)} \\
&\quad \preceq \texttt{fwd(dim=}n\texttt{,depth=}k\texttt{,p) + bwd(dim=}n\texttt{,depth=}k\texttt{,p)} && \textbf{(Combined)} \\
&\quad \equiv \texttt{ctd(dim=}n\texttt{,depth=}k\texttt{,p)} && \textbf{(Centered)}
\end{aligned}$$

Proof of the second inequality:

$$\begin{aligned}
&\texttt{bwd(dim=}n\texttt{,depth=}k\texttt{,p)} \\
&\quad \preceq \texttt{fwd(dim=}n\texttt{,depth=}k\texttt{,p) + bwd(dim=}n\texttt{,depth=}k\texttt{,p)} && \textbf{(Combined)} \\
&\quad \equiv \texttt{ctd(dim=}n\texttt{,depth=}k\texttt{,p)} && \textbf{(Centered)}
\end{aligned}$$

□

Proposition 2 (Point approximation)**.** *Let* `R` *be one of* `forward`*,* `backward`*, and* `centered`*, n a fixed dimension, and k a fixed depth, then we have*

$$\texttt{pointed(dim=}n\texttt{)} \preceq \texttt{R(dim=}n\texttt{,depth=}k\texttt{)}$$
$$\texttt{R(dim=}n\texttt{,depth=}k\texttt{,nonpointed)} \preceq \texttt{R(dim=}n\texttt{,depth=}k\texttt{)}$$

Proof. Proof of the first inequality:

$$\begin{aligned}
&\texttt{ptd(dim=}n\texttt{)} \\
&\quad \preceq \texttt{R(dim=}n\texttt{,depth=}k\texttt{,nonpointed) + ptd(dim=}n\texttt{)} && \textbf{(Combined)} \\
&\quad \equiv \texttt{R(dim=}n\texttt{,depth=}k\texttt{)} && \textbf{(Overlapping pointed)}
\end{aligned}$$

Proof of the second inequality:

$$\begin{aligned}
&\texttt{R(dim=}n\texttt{,depth=}k\texttt{,nonpointed)} \\
&\quad \preceq \texttt{R(dim=}n\texttt{,depth=}k\texttt{,nonpointed) + ptd(dim=}n\texttt{)} && \textbf{(Combined)} \\
&\quad \equiv \texttt{R(dim=}n\texttt{,depth=}k\texttt{)} && \textbf{(Overlapping pointed)}
\end{aligned}$$

□

5 Semantic model

5.1 Lattice model or regions

The underlying domain of our semantics is the set $\mathcal{P}(\mathbb{Z}^N)$ that is, sets of integer vectors of length N. We characterise various subsets of this space used in our model, called: *index schemes*, *interval schemes*, *regions*, and *subscript schemes*.

Definition 5 (Index scheme)**.** An N-dimensional *index scheme* S is a set of integer vectors ($S \in \mathcal{P}(\mathbb{Z}^N)$) which can be written as the Cartesian product of N subsets of $\mathbb{Z}$, *i.e.*

$$\exists S_1 \subseteq \mathbb{Z}, \ldots, S_N \subseteq \mathbb{Z} \,.\; S = \prod_{i=1}^{N} S_i$$

with the additional condition that every S_i is either finite (*i.e.*, $\exists n.|S_i| = n$) or $S_i = \mathbb{Z}$.

We use S, T, U to denote index schemes throughout. Henceforth, we implicitly assume index schemes are N-dimensional[1] for some N when it is clear from the context.

Index schemes can be *projected* in the i^{th} dimension by $\pi_i : \mathcal{P}(\mathbb{Z}^N) \to \mathcal{P}(\mathbb{Z})$. For an index scheme S, we refer to $\pi_i(S)$ as the i^{th} *component* of S. We assume that i, when used for projection, always lies between 1 and N.

Lemma 1. *Intersection distributes over index schemes. That is, for index schemes S, T*

$$S \cap T = \prod_{i=1}^{N} \pi_i(S) \cap \pi_i(T)$$

Thus intersection is closed for index schemes: the intersection of two index schemes is an index scheme.

Proof.

$$\begin{aligned} S \;\cap\; T &= \left\{ x \;\middle|\; \bigwedge_{1 \le i \le N} x_i \in \pi_i(S) \right\} \bigcap \left\{ x \;\middle|\; \bigwedge_{1 \le i \le N} x_i \in \pi_i(T) \right\} \\ &= \left\{ x \;\middle|\; \bigwedge_{1 \le i \le N} (x_i \in \pi_i(S) \wedge x_i \in \pi_i(T)) \right\} \\ &= \prod_{i=1}^{N} \pi_i(S) \cap \pi_i(T) \end{aligned}$$

□

Union however is not itself an index scheme because union does not distribute over Cartesian product (recall, index schemes must be definable as a Cartesian product of integer sets). However, a more restricted property holds.

Lemma 2. *If S and T are index schemes where $\pi_i(S) = \pi_i(T)$ for all $1 \le i \le N$ apart from some dimension k, then:*

$$S \cup T = \pi_1(S) \times \cdots \times (\pi_k(S) \cup \pi_k(T)) \times \cdots \times \pi_N(S)$$

[1]Whenever the model is used, the particular dimensionality N is derived from source code, from the static type of the array being analysed.

Proof.

$$
\begin{aligned}
S \cup T &= \left\{ x \;\middle|\; \bigwedge_{1\le i\le N} x_i \in \pi_i(S) \right\} \bigcup \left\{ x \;\middle|\; \bigwedge_{1\le i\le N} x_i \in \pi_i(T) \right\} \\
&= \left\{ x \;\middle|\; \bigwedge_{\substack{1\le i\le N \\ i\neq k}} x_i \in \pi_i(S) \wedge x_k \in \pi_k(S) \vee \bigwedge_{\substack{1\le i\le N \\ i\neq k}} x_i \in \pi_i(T) \wedge x_k \in \pi_k(T) \right\} \\
&= \left\{ x \;\middle|\; \bigwedge_{\substack{1\le i\le N \\ i\neq k}} x_i \in \pi_i(S) \wedge x_k \in \pi_k(S) \cup \pi_k(T) \right\} \\
&= \pi_1(S) \times \cdots \times (\pi_k(S) \cup \pi_k(T)) \times \cdots \times \pi_N(S)
\end{aligned}
$$

□

Definition 6 (Intervals with an optional hole)**.** We define an extended notion of closed interval on $\mathbb{Z}$ which may contain a *hole* at the origin, written $[a \ldots b]^c$ where a and b are drawn from $\mathbb{Z}$ with $a \le 0 \le b$ and c is drawn from $\mathbb{B} = \{\mathsf{true}, \mathsf{false}\}$. Intervals are interpreted as sets as follows:

$$[a \ldots b]^c \triangleq \{n \mid a \le n \le b \wedge (c \vee n \neq 0)\}$$

If the superscript to the interval is omitted it is treated as true (no hole). We also add the distinguished interval $[-\infty \ldots \infty]$, which is simply an alias for $\mathbb{Z}$, but this notation avoids separate handling of the infinite interval in the following definitions, lemmas, and their proofs.

We denote the set of all intervals with an optional hole as *Interval*.

Lemma 3. *Union and intersection are closed for Interval, where we have the following dual identities:*

$$
\begin{aligned}
[a \ldots b]^c \cap [d \ldots e]^f &= [\max\{a, d\} \ldots \min\{b, e\}]^{c \wedge f} \\
[a \ldots b]^c \cup [d \ldots e]^f &= [\min\{a, d\} \ldots \max\{b, e\}]^{c \vee f}
\end{aligned}
$$

Proof. We give the proof of the first identity and the second one is similar.

$$
\begin{aligned}
[a \ldots b]^c \cap [d \ldots e]^f &= \left\{ n \;\middle|\; a \le n \le b \wedge (\neg c \implies n \neq 0) \right\} \cap \\
&\quad \left\{ n \;\middle|\; d \le n \le e \wedge (\neg f \implies n \neq 0) \right\} \\
&= \left\{ n \;\middle|\; \max\{a, d\} \le n \le \min\{b, e\} \wedge (\neg c \vee \neg f \implies n \neq 0) \right\} \\
&= [\max\{a, d\} \ldots \min\{b, e\}]^{c \wedge f}
\end{aligned}
$$

□

We define a subset of index schemes called *interval schemes*, where each component of an index scheme is a member of *Interval*.

Definition 7 (Interval scheme)**.** An *interval scheme* is an N-dimensional index scheme such that each component of the scheme is a holed interval (Definition 6) (since intervals define either finite subsets of $\mathbb{Z}$ or $\mathbb{Z}$ in its entirety).

$Interval_N$ denotes the set of all N-dimensional interval schemes.

Interval schemes are composed by union and intersection to form a *region*, which describes the extent of a specification in our language.

Definition 8 (Region)**.** The set of all N-dimensional regions, denoted $Region_N$ is defined as the smallest set satisfying the following:

1. If R is in $Interval_N$, then R is in $Region_N$.
2. If R and S are in $Region_N$, then so are $R \cap S$ and $R \cup S$.

Proposition 3. *($Region_N, \cup, \cap, \top, \bot$) is a bounded distributive lattice with top element $\top = \mathbb{Z}^N$ and bottom element $\bot = \emptyset$.*

Proof. Straightforward, the join and meet are mapped to $\cup$ and $\cap$, the set is inductively defined to be closed under these operations. Union and intersection are associative, commutative, and absorptive under closed sets and then they are also for $Region_N$. This is enough to show that, it is a lattice.

Further, we have $\mathbb{Z}^N \cap R = R$ and $\emptyset \cup R = R$ with $\mathbb{Z}^N$ and $\emptyset$ belonging to $Region_N$. This makes the lattice bounded.

Finally, the lattice is distributive since union distributes over intersection and vice versa when the set is closed under these operations. □

The set of regions $Region_N$ is the target of our specification language model. However, the model must also incorporate information about approximation and multiplicity specification modifiers. This information is provided by the following labelled variants.

Definition 9. Mult and Approx are parametric labelled variant types with injections given by:

$$\mathsf{Mult}\ a \triangleq \mathsf{mult}\ a \mid \mathsf{only}\ a \qquad \mathsf{Approx}\ a \triangleq \mathsf{exact}\ a \mid \mathsf{lower}\ a \mid \mathsf{upper}\ a$$

e.g., lower is an injection into Approx, of type $\mathsf{lower} : a \to \mathsf{Approx}\ a$.

These injection functions are used to mark the model of regions ($Region_N$) with *multiplicity* and *approximation* information.

During specification and consistency checking, Mult is used to determine whether repeated indices should be allowed, while Approx is used to determine if the region in the specification should be treated as a lower bound, an upper bound, or in exact correspondence with the region defined by the stencil computation.
Finally, we define another subset of index schemes called *subscript schemes* where each component of an index scheme is either a singleton set or $\mathbb{Z}$. This is used to model array index terms from source code.

Definition 10 (Subscript scheme)**.** A *subscript scheme* S is an index scheme where:

$$\forall i.\ 1 \leq i \leq N \implies \exists p.\ \pi_i(S) = \{p\} \ \vee\ \pi_i(S) = \mathbb{Z}$$

That is, the i^{th} component of the set is either a singleton in $\mathbb{Z}$ or all of $\mathbb{Z}$.

$$
\begin{array}{ll}
[\![-]\!]^{\mathsf{a}} : \mathit{approx} \to (A \to \mathsf{Approx}\, A) & \\
[\![\texttt{atLeast}]\!]^{\mathsf{a}} & = \mathsf{lower} \\
[\![\texttt{atMost}]\!]^{\mathsf{a}} & = \mathsf{upper} \\
[\![\epsilon]\!]^{\mathsf{a}} & = \mathsf{exact}
\end{array}
$$

$$
\begin{array}{ll}
[\![-]\!]^{\mathsf{m}} : \mathit{mult} \to (A \to \mathsf{Mult}\, A) & \\
[\![\texttt{readOnce}]\!]^{\mathsf{m}} & = \mathsf{once} \\
[\![\epsilon]\!]^{\mathsf{m}} & = \mathsf{mult} \\
\\
[\![\texttt{nonpointed}]\!] & = \mathsf{false} \\
[\![\epsilon]\!] & = \mathsf{true}
\end{array}
$$

(a) Interpretation of modifiers

$$
\begin{array}{ll}
[\![-]\!]_N : \mathit{region} \to \mathit{Region}_N & \\
[\![\texttt{r + s}]\!]_N & = [\![\texttt{r}]\!]_N \cup [\![\texttt{s}]\!]_N \\
[\![\texttt{r * s}]\!]_N & = [\![\texttt{r}]\!]_N \cap [\![\texttt{s}]\!]_N \\
[\![\texttt{rconst}]\!]_N & = \mathit{promote}_N(\mathit{dim}(\texttt{rconst}), [\![\texttt{rconst}]\!])
\end{array}
$$

$$
\begin{array}{ll}
[\![-]\!] : \mathit{rconst} \to \mathit{Interval} & \\
[\![\texttt{pointed(dim=}i\texttt{)}]\!] & = [0 \ldots 0]^{\mathsf{true}} \\
[\![\texttt{centered(dim=}i\texttt{,depth=}k\texttt{,p)}]\!] & = [-k \ldots k]^{[\![\texttt{p}]\!]} \\
[\![\texttt{forward(dim=}i\texttt{,depth=}k\texttt{,p)}]\!] & = [0 \ldots k]^{[\![\texttt{p}]\!]} \\
[\![\texttt{backward(dim=}i\texttt{,depth=}k\texttt{,p)}]\!] & = [-k \ldots 0]^{[\![\texttt{p}]\!]}
\end{array}
$$

(b) Interpretation of regions and region constants

Figure 1: Semantic model of specifications

5.2 Denotational semantics for specifications

Figure 1 shows the interpretation of the syntactic components.

Definition 11 (Semantics of specifications)**.** An interpretation function $[\![-]\!]_N$ maps closed[2] specifications to an element of $\mathsf{Mult}(\mathsf{Approx}(\mathit{Region}_N))$ as follows:

$$[\![\texttt{mult, approx, region}]\!]_N = [\![\texttt{mult}]\!]^{\mathsf{m}}\ ([\![\texttt{approx}]\!]^{\mathsf{a}}\ [\![\texttt{region}]\!]_N)$$

Definition 12. Let $\mathit{promote}_N : \mathbb{N}_{>0} \times \mathit{Interval} \to \mathit{Interval}_N$ be a function generating an interval scheme such that if v is $\mathit{promote}_N(i, [a \ldots b]^c)$, then $\pi_i(v) = [a \ldots b]^c$ and $\pi_j(v) = \mathbb{Z}$ in all other dimensions j.

Definition 13. Given a region constant `rconst`, let $\mathit{dim}(\texttt{rconst}) \in \mathbb{N}_{>0}$ be the dimension for which that region constant is defined, *e.g.*, $\mathit{dim}(\texttt{centered(dim=}d\texttt{,depth=}k\texttt{)}) = d$.

Theorem 1 (Equational soundness)**.** *The lattice model is sound with respect to the equational theory. Let R and S be N-dimensional region terms, then we have*

$$\forall R, S, N.\quad R \equiv S \implies [\![R]\!]_N = [\![S]\!]_N$$

Proof. Both $\equiv$ and $=$ are equivalence relations. We only need to show that each property associated with specifications equivalence holds as an equality in the model.

Basic & Subsumption These follow from basic properties of lattices.

Dist This is immediate from the definition of distributive lattice (see Proposition 3).

[2]That is, we assume there are no occurrences of *rvar* in a specification being modelled. Any *open* specification containing region variables can be made closed by straightforward syntactic substitution with a (closed) *region*.

Overlapping pointed Counterparts of individual regions for some Boolean p are given below:

$$\begin{aligned}
[\![\texttt{R(dim=}n\texttt{,depth=}k\texttt{,nonpointed)}]\!]_N &= \mathit{promote}_N(n, [-k \ldots k]^{\mathsf{false}}) \\
[\![\texttt{pointed(dim=}n\texttt{)}]\!]_N &= \mathit{promote}_N(n, [0 \ldots 0]^{\mathsf{true}}) \\
[\![\texttt{R(dim=}n\texttt{,depth=}k\texttt{)}]\!]_N &= \mathit{promote}_N(n, [-k \ldots k]^{\mathsf{true}})
\end{aligned}$$

Using Lemma 2, it is sufficient to show $[-k \ldots k]^p \ \cup \ [0 \ldots 0]^{\mathsf{true}} = [-k \ldots k]^{\mathsf{true}}$ holds. This immediately follows from Lemma 3.

Centered Let $[\![\texttt{p1}]\!]$, $[\![\texttt{p2}]\!]$, and $[\![\texttt{p3}]\!]$ be $p1$, $p2$, and $p3$ respectively. We know that $p1$ is $p2 \vee p3$ from (**Centered**) definition. Then we get the following interpretation for region constants involved in the equivalence:

$$\begin{aligned}
[\![\texttt{centered(dim=}k\texttt{,depth=}n\texttt{,}p1\texttt{)}]\!]_N &= \mathit{promote}_N(n, [-k \ldots k]^{p2 \vee p3}) \\
[\![\texttt{forward(dim=}k\texttt{,depth=}n\texttt{,}p2\texttt{)}]\!]_N &= \mathit{promote}_N(n, [0 \ldots k]^{p2}) \\
[\![\texttt{backward(dim=}k\texttt{,depth=}n\texttt{,}p3\texttt{)}]\!]_N &= \mathit{promote}_N(n, [0 \ldots k]^{p3})
\end{aligned}$$

Using Lemma 2, it is sufficient to show $[-k \ldots k]^{p2 \vee p3} = [0 \ldots k]^{p2} \cup [-k \ldots 0]^{p3}$ holds. This immediately follows from Lemma 3.

□

Theorem 2 (Approximation soundness)**.** *The lattice model is sound with respect to the theory of approximation. Let R and S be N-dimensional regions, then we have*

$$\forall R, S, N. \quad R \preceq S \implies [\![R]\!]_N \subseteq [\![S]\!]_N$$

Proof. Both $\preceq$ and $\subseteq$ are partial orders on sets. We only need to show that each inequality holds in the model.

Equivalence Region equivalence is modeled with set equality and the partial order on regions is modeled with subset relation. Partial orders are reflexive and the model of equivalence agrees with that of the partial order, so this rule holds.

Combined `*` and `+` maps to meet and join operations in the lattice. All meets and joins in a lattice satisfy these inequalities.

Depth When m is k and l (with $k \leq l$) in `R(dim=`n`,depth=`m`,p)`, we obtain the interpretations $\mathit{promote}_N(n, \mathit{int}_k)$ and $\mathit{promote}_N(n, \mathit{int}_l)$ for some N as interpretations of regions. Call these *ks* and *ls* respectively. One way to show $ks \subseteq ls$ is to show that the intersection of *ks* and *ls* is *ks*.

To show that the intersection is *ks*, we use Lemma 1 which reduces the problem to showing that the intersection of int_k and int_l is the former since at every point except n the intervals agree, hence the intersection remains the same at these dimensions. Then we use Lemma 3 to calculate the intersection. From the initial assumptions, we have $k \leq l$, so the maximum of the lower bounds is $-k$ and the minimum of lower bounds is k. This equivalent to the interval int_k. Note that pointedness is the same for both intervals and depths are positive integers, so the pointedness is preserved under intersection. This shows the intersection of *ls* and *ks* is indeed *ks* and concludes the proof.

□

5.3 Modelling array subscripts

Definition 14 (Individual array terms)**.** Recall array subscript terms of the form $a(\bar{e})$ from Definition 2. We interpret these terms with a partial interpretation to subscript schemes (a subset of index schemes) $[\![-]\!]^{aterm} : \textit{array-term} \nrightarrow \mathcal{P}(\mathbb{Z}^N)$. The interpretation is defined when all indices are either absolute or neighbourhood indices (Definition 4).

$$[\![a(\bar{e})]\!]^{aterm} = \prod_{1 \leq i \leq N} \textit{subscript}(\bar{e}_i) \qquad \textit{subscript}(e) = \begin{cases} \{c\} & e \equiv \mathtt{j} \pm c \\ \mathbb{Z} & e \text{ is absolute} \end{cases}$$

where $\mathtt{j}$ is an induction variable. Note that this interpretation function produces subscript schemes, but these are not necessarily members of $\textit{Interval}_N$ or $\textit{Region}_N$.

Definition 15 (Collections of array terms)**.** Given an set of $\mathcal{A}$ of array terms flowing to a particular statement in code, we take the union of the interpretations of array terms to give a model of the array access in the computation by:

$$[\![\mathcal{A}]\!]^{aterm} = \bigcup_{a \in \mathcal{A}} [\![a]\!]^{aterm}$$

5.4 Union Normal Form

Definition 16 (Union Normal Form)**.** A set $V \in \mathcal{P}(\mathbb{Z}^N)$ is in union normal form (UNF) if it can be expressed as the union of a finite number of N-dimensional index schemes; that is, there is a finite set of index schemes $\mathcal{S}$ such that $V = \bigcup_{T \in \mathcal{S}} T$. We refer to $\mathcal{S}$ as the *support* of V.

Lemma 4. *A set in UNF has a finite representation.*

Proof. By their definition, N-dimensional index schemes can be written as the Cartesian product of N sets which are either finite subsets of $\mathbb{Z}$ or $\mathbb{Z}$ itself. Thus, index schemes have a finite representation as a finite list of its components which are either finite sets or $\mathbb{Z}$, which can be marked with a single representative symbol.

Thus, a set V in UNF can then be represented finitely by its support $\mathcal{S}$, whose elements (which are index schemes) are all finitely represented. □

Lemma 5. *The model of a region specification produces a $\textit{Region}_N$ which can be converted into UNF.*

Proof. Since $\textit{Region}_N$ is a distributive lattice (with $\cup$ and $\cap$) (Proposition 3), its elements can be converted into a union of intersections of index schemes via the property: $(S \cup S') \cap (T \cup T') = (S \cap T) \cup (S \cap T')$ and because the base elements in the inductive definition of $\textit{Region}_N$ are $\textit{Interval}_N$ elements, which are index schemes. Since $\cap$ is closed over index schemes (Lemma 1), the result is a union of index schemes, *i.e.* in UNF. □

Lemma 6. *The model of array subscripts in code (Definition 14) already produces a set in union normal form: a UNF of subscript schemes (by Definition 15).*

Proof. Definition 15 takes the union of interpretations of array terms (drawn from a set $\mathcal{A}$), using the interpretation $[\![-]\!]^{aterm}$ (Definition 14). The interpretation of array terms produces either a singleton, or $\mathbb{Z}$, thus it maps array subscripts to subscript schemes (which are also index schemes). Thus the result of $[\![\mathcal{A}]\!]^{aterm}$ is in UNF. □

6 Algorithms: analysis, checking, and inference

6.3 Inferring specifications automatically

Definition 17 (Adjacent)**.** Two index schemes S and T are *adjacent* written $adjacent(S, T)$, iff $\pi_i(S) = \pi_i(T)$ for all $1 \leq i \leq N$ apart from some dimension k such that $\pi_k(S) = [a, b]$ and $\pi_k(T) = [b + 1, c]$ (these are standard closed intervals, rather than holed intervals of *Interval*).

(See the paper, Section 6.3, for the full definition of *coalesce* and *coalesceStep*).

Lemma 7. *For a set $\mathcal{S}$ of index schemes, coalesceStep($\mathcal{S}$) is a set of index schemes. Furthermore, if every index scheme in $\mathcal{S}$ has components which are all either finite closed intervals or $\mathbb{Z}$, then every index scheme in coalesceStep($\mathcal{S}$) similarly has all components as finite closed intervals or $\mathbb{Z}$.*

Proof. From the assumptions of the lemma, the input set $\mathcal{S}$ is a set of index schemes. We proceed inductively on the size of this input set.

- Base case, $\mathcal{S} = \emptyset$, returning $\emptyset$. Trivially satisfies the property that the output set contains index schemes.
- Inductive step, *coaleseStep*($\{T\} \cup \mathcal{S}$).

  The output set is always built from members of the input set, which are always index schemes as shown in the two cases of *coalesceStep*.

  In the first case, the output set is built from the singleton set $\{T\}$ (where T is an index scheme by the input property) and the recursive call to *coalesceStep*(S). Applying the inductive hypothesis, a set of index schemes is thus produced.

  In the second case, *coalesce* is non-empty. The application of coalesce($T, \mathcal{S}$) produces a set of index schemes: by Lemma 2, the union of two index schemes is an index scheme if all components are equal except at most one component. Contiguity (Definition 17) matches this precondition, thus *coalesce* maps sets of index schemes to sets of index schemes. Thus, the recursive call to *coalesceStep* is applied again to a set of index schemes.

□

Corollary 1. *A subscript scheme has components which are either $\mathbb{Z}$ or a singleton set (which is a closed interval* e.g.*, $\{n\} = [n, n]$) thus the fixed-point of coalesceStep on $\mathcal{S}_0$ (the initial set of subscript schemes from the analysis) yields a set of index schemes whose components are finite intervals or $\mathbb{Z}$.*

Proof. Let C be the property that a set contains all closed integer inverals or $\mathbb{Z}$.

Firstly, *coalesce* preserves the C-property since it takes the union of elements, where a union of two closed sets which are adjacent yields another closed set.

Then *coalesceStep* produces a C set if it is applied to a C set, by induction on the cardinality of the input set. The base case is trivial. In the inductive step:

- if $\mathit{coalesce}(T, \mathcal{S}) = \emptyset$ then $\{T\}$ (which satisfies C) is unioned onto $\mathit{coalesceStep}(\mathcal{S} - T)$ which preserves C (by induction).
- if $\mathit{coalesce}(T, \mathcal{S}) \neq \emptyset$ then *coalesceStep* is applied to a set built from the union of $\mathit{coalesce}(T, \mathcal{S})$ which is C preserving and $\mathcal{S}$ - $\{T\}$ (also C preserving). Thus, the result has the C-property (since the output set is built only from the input set).

Note that we first apply *coalesceStep* to a set $\mathcal{S}_0$ of subscript schemes, which satisfies C as they are either singleton intervals of $\mathbb{Z}$. Thus, $\mathit{coalesceStep}(\mathcal{S}_0)$ yields a C-set and therefore the fixed point $\mu\mathit{coalesceStep}(\mathcal{S}_0)$ satisfies the C property: that the output set of index schemes are all expressible via closed integer intervals or $\mathbb{Z}$. □

Definition 18. (Convert) We map a component of an interval scheme to a region constant using $[\![-]\!]^{-1}$, combining each component with `*` and combining the resulting regions by `+`:

$$\mathit{convert}_N(\mathcal{S}) = \sum_{S \in \mathcal{S}} \prod_{\pi_i(S) \neq \mathbb{Z}} [\![\pi_i(S)]\!]^{-1} \tag{1}$$

Where summation is by the syntactic `+` and product is by the syntactic `*`.

Theorem 3 (Inference soundness)**.** *For all region specs R, then $\mathit{infer}([\![R]\!]_N)_N \equiv R$*

Proof. This follows by induction on the definition of regions syntax R and by definitions of inference and the semantics sharing the same underlying model and using inverse mappings between each other.

For constants, we map to $\mathit{promote}_N(i, [\![\texttt{rconst}]\!])$ which by *convert*, in (1) is mapped back to `rconst` since *convert* is defined in terms of the inverse mapping $[\![-]\!]^{-1}$.

For $[\![r\texttt{+}s]\!]_N = [\![r]\!]_N \vee [\![s]\!]$ we map to a Union Normal Form of interval spaces which is converted back directly by *convert*. Similarly for `*`. □

References

[1] Dominic Orchard, Mistral Contrastin, Matthew Danish, and Andrew Rice, *Verifying Spatial Properties of Array Computations.* PACM Progr. Lang. 1, OOPSLA, Article 75 (October 2017), 30 pages.