Transfinite Step-Indexing for Termination

Technical Report

Simon Spies, Neel Krishnaswami, Derek Dreyer

October 27, 2020

Contents

1 The Language $\lambda_{\text{chan}}$ 2
2 Ordinals 5
3 Resources 7
4 Logical Relation 11
  4.1 Kripke Logical Relation 14
  4.2 Resource Interpretation Properties 15
  4.3 Properties of the Type Interpretations 20
  4.4 Semantic Typing 24
  4.5 Compatibility Lemmas 26
either nothing

Table 1: The language \( \lambda_{\text{chan}} \)

1 The Language \( \lambda_{\text{chan}} \)

In the present work, we consider the language \( \lambda_{\text{chan}} \) given in Figure 1, an extension of the simply-typed \( \lambda \)-calculus with an implementation of asynchronous channels. A fresh channel can be created with let \( (x, y) = \text{chan()} \) in \( e \), where \( x \) is a handle for receiving values over the new channel and \( y \) is a handle for sending values. The operation put can be used to send values and the operation get to receive values.

We assign expressions a type \( A \) in the linear type system \( \Gamma \vdash e : A \), defined in Figure 2. Besides the base types \( \mathbb{I}, \mathbb{B}, \mathbb{N} \), we have linear pairs \( A \otimes B \) and linear functions \( A \to B \). The type \( \text{Get} A \) is used for the receive handle of a channel, indicating that the channel will transfer a value of type \( A \). Similarly, the type \( \text{Put} A \) is used for the send handle. We allow values of arbitrary types \( A \) to be transferred through channels, including channel handles themselves and functions possibly capturing channel handles. The context \( \Gamma \) is a linear context without an ordering of the variables. We write \( \Gamma, \Delta \) for the disjoint union of the contexts \( \Gamma \) and \( \Delta \).

We equip the language with a single-threaded, heap-based operational semantics, given in Figure 3. Operationally, each channel is represented by a single location \( \ell \) in the heap, storing either nothing \( E \), a value \( V(v) \), or a continuation \( C(\lambda x.e) \). Initially, the heap stores the empty heap value \( E \). If a value is sent over channel \( \ell \) with put(\( \ell, v \)), then the value is stored in memory as \( V(v) \). If subsequently get(\( \ell, \lambda x.e \)) is executed, then the continuation \( \lambda x.e \) is invoked with argument \( v \) and the state in the heap is restored to \( E \). If from the initial state get(\( \ell, \lambda x.e \)) is executed, then the continuation is stored in the heap as \( C(\lambda x.e) \) and invoked with argument \( v \) once a corresponding put(\( \ell, v \)) is called. Besides the operations on channels, the operational semantics allows standard, pure reductions for the simply typed \( \lambda \)-calculus. Reductions are allowed to occur in any evaluation context \( K \), making it a call-by-value, left-to-right operational semantics.

We write \( h[\ell \mapsto hv] \) for the heap which returns \( hv \) for argument \( \ell \) and \( h(\ell') \) for any argument \( \ell' \neq \ell \). Analogously, we use the notation to update finite and infinite maps with a new binding in the remainder of this work. We assume substitution is capture avoiding, write \( e[v/x] \) for the single-point substitution replacing \( x \) with \( v \) in \( e \), and \( e[\theta] \) for the parallel substitution replacing each free variable \( x \) in \( e \) with \( \theta \).
<table>
<thead>
<tr>
<th>Head</th>
<th>Body</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\Gamma \vdash e : A)</td>
<td>(x : A \vdash x : A)</td>
</tr>
<tr>
<td></td>
<td>(\Gamma, x : A \vdash e : B)</td>
</tr>
<tr>
<td></td>
<td>(\vdash () : 1)</td>
</tr>
<tr>
<td></td>
<td>(\Gamma \vdash e_1 : 1)</td>
</tr>
<tr>
<td></td>
<td>(\Delta \vdash e_2 : A)</td>
</tr>
<tr>
<td></td>
<td>(\vdash b : B)</td>
</tr>
<tr>
<td></td>
<td>(\Gamma \vdash e : B)</td>
</tr>
<tr>
<td></td>
<td>(\Delta \vdash e_1 : A)</td>
</tr>
<tr>
<td></td>
<td>(\Delta \vdash e_2 : A)</td>
</tr>
<tr>
<td></td>
<td>(\Gamma, \Delta \vdash \text{if } e \text{ then } e_1 \text{ else } e_2 : A)</td>
</tr>
<tr>
<td></td>
<td>(\vdash n : N)</td>
</tr>
<tr>
<td></td>
<td>(\Gamma, \Delta \vdash e : N)</td>
</tr>
<tr>
<td></td>
<td>(\Delta \vdash e_0 : A)</td>
</tr>
<tr>
<td></td>
<td>(x : A \vdash e_3 : A)</td>
</tr>
<tr>
<td></td>
<td>(\Gamma, \Delta \vdash \text{iter}(e, e_0, x, e_3) : A)</td>
</tr>
<tr>
<td></td>
<td>(\Gamma \vdash e_1 : A_1 \otimes A_2)</td>
</tr>
<tr>
<td></td>
<td>(\Delta \vdash e_2 : A_1 \otimes A_2)</td>
</tr>
<tr>
<td></td>
<td>(\Gamma, \Delta \vdash (e_1, e_2) : A_1 \otimes A_2)</td>
</tr>
<tr>
<td></td>
<td>(\Gamma, x : A \vdash e : B)</td>
</tr>
<tr>
<td></td>
<td>(\Gamma, \Delta \vdash \text{let } (x,y) = e_1 \text{ in } e_2 : B)</td>
</tr>
</tbody>
</table>
| | \(\Gamma \vdash e_1 : A 
\rightarrow B\) |
| | \(\Delta \vdash e_2 : A\) |
| | \(\Gamma, \Delta \vdash e_1, e_2 : B\) |
| | \(\Gamma \vdash \text{Get } A\) |
| | \(\Delta \vdash e_2 : A \rightarrow 1\) |
| | \(\Gamma, \Delta \vdash \text{get}(e_1, e_2) : 1\) |
| | \(\Gamma, x : \text{Get } A, y : \text{Put } A \vdash e : B\) |
| | \(\Gamma \vdash \text{let } (x,y) = \text{chan}() \text{ in } e : B\) |

Figure 2: Typing Rules
Evaluation Contexts  

\[
K ::= \cdot | K; e | K e' | vK | (K, e) | (v, K) | \text{let } (x, y) = K \text{ in } e' \\
| K + e | v + K | \text{iter}(K, e, e') | \text{iter}(v, K, x, e) \\
| \text{if } K \text{ then } e_2 \text{ else } e_3 | \text{get}(K, e') | \text{get}(v, K) | \text{put}(K, e') | \text{put}(v, K)
\]

\[
\begin{align*}
\begin{array}{c}
e \rightsquigarrow_p e' \\
(e, h) \rightsquigarrow (e', h')
\end{array}
\end{align*}
\]

Pure Reduction

\[
\begin{align*}
() & \rightsquigarrow_p e \\
(\lambda x. e) & \rightsquigarrow_p e[v/x] \\
\text{if true then } e_1 \text{ else } e_2 & \rightsquigarrow_p e_1 \\
\text{if false then } e_1 \text{ else } e_2 & \rightsquigarrow_p e_2
\end{align*}
\]

\[
\begin{align*}
\text{let } (x, y) = (v_1, v_2) \text{ in } e & \rightsquigarrow_p e[v_1/x, v_2/y] \\
n + m & \rightsquigarrow_p n + m \\
\text{iter}(0, v, x, e) & \rightsquigarrow_p v \\
\text{iter}(n + 1, v, x, e) & \rightsquigarrow_p \text{iter}(n, e[v/x], x, e)
\end{align*}
\]

Channel Reduction

\[
\begin{align*}
\text{let } (x, y) = \text{chan}() \text{ in } e, h & \rightsquigarrow_c (e[\ell/x, \ell/y], h[\ell \mapsto E]) \quad \text{if } \ell \notin \text{dom } h \\
\text{get}(\ell, \lambda x. e), h & \rightsquigarrow_c () \quad \text{if } h\ell = E \\
\text{get}(\ell, \lambda x. e), h & \rightsquigarrow_c (e[v/x], h[\ell \mapsto E]) \quad \text{if } h\ell = V(v) \\
\text{put}(\ell, v), h & \rightsquigarrow_c () \quad \text{if } h\ell = E \\
\text{put}(\ell, v), h & \rightsquigarrow_c (e[v/x], h[\ell \mapsto E]) \quad \text{if } h\ell = C(\lambda x. e)
\end{align*}
\]

Figure 3: Operational Semantics
2 Ordinals

In this work, we only consider ordinals strictly smaller than $\omega^\omega$. Each such ordinal, denoted by $\alpha, \beta, \gamma$ in the following, can be expressed in its Cantor normal form. That is, for each ordinal $\alpha < \omega^\omega$ there is some $k \in \mathbb{N}$ and coefficients $a_0, \ldots, a_k \in \mathbb{N}$ such that $\alpha = \sum_{i=0}^k a_i \omega^i \triangleq \omega^k a_k + \cdots + \omega^0 a_0$. We combine ordinals using natural addition [3].

**Definition 2.1** (Natural Addition). Let $\alpha = \sum_{i=0}^k \omega^i a_i$ and $\beta = \sum_{i=0}^l \omega^i b_i$. We define natural addition by $\alpha \oplus \beta \triangleq \sum_{i=0}^{\max(k,l)} \omega^i (a_i + b_i)$ where $a_i \triangleq 0$ for $i = k + 1, \ldots, l$ and $b_i \triangleq 0$ for $i = l + 1, \ldots, k$.

**Lemma 2.1.** Natural addition is associative and commutative, and 0 is an identity. Natural addition is compatible with $<$ on ordinals, meaning $\alpha < \beta$ implies $\alpha \oplus \gamma < \beta \oplus \gamma$ for all $\alpha, \beta, \gamma$.

Natural addition is cancellative, meaning $\alpha \oplus \gamma = \beta \oplus \gamma$ implies $\alpha = \beta$.

**Proof.** Let $\alpha = \sum_{i=0}^m \omega^i a_i$, $\beta = \sum_{i=0}^n \omega^i b_i$, and $\gamma = \sum_{i=0}^p \omega^i c_i$. We define $k \triangleq \max(m, n, p)$ and extend $\alpha, \beta, \gamma$ with zeros by $a_i \triangleq 0$ for $i = m+1, \ldots, k$ and $b_i \triangleq 0$ for $i = n+1, \ldots, k$ and $c_i \triangleq 0$ for $i = p+1, \ldots, k$. As $\omega^i \cdot 0 = 0$ and $0 + \alpha = \alpha$ for all $i$ and $\alpha$, we have $\alpha = \sum_{i=0}^k \omega^i a_i$, $\beta = \sum_{i=0}^k \omega^i b_i$, and $\gamma = \sum_{i=0}^k \omega^i c_i$.

1. **Associativity.** $\alpha \oplus (\beta \oplus \gamma) = \alpha \oplus \sum_{i=0}^k \omega^i (b_i + c_i) = \sum_{i=0}^k \omega^i (a_i + (b_i + c_i)) = \sum_{i=0}^k \omega^i ((a_i + b_i) + c_i) = (\sum_{i=0}^k \omega^i (a_i + b_i)) \oplus \gamma = (\alpha \oplus \beta) \oplus \gamma.$

2. **Commutativity.** $\alpha \oplus \beta = \sum_{i=0}^k \omega^i (a_i + b_i) = \sum_{i=0}^k \omega^i (b_i + a_i) = \beta \oplus \alpha.$

3. **Identity.** $\alpha \oplus 0 = \sum_{i=0}^k \omega^i (a_i + 0) = \sum_{i=0}^k \omega^i a_i = \alpha.$

4. **Compatibility.** As $\alpha < \beta$, we have $\alpha \neq \beta$. Let $j$ be the largest number such that $a_j \neq b_j$. Clearly $a_j < b_j$ as otherwise $\beta > \alpha$. Since $\sum_{i=0}^{j-1} \omega^i (a_i + c_i) < \omega^j$, we have:

\[
\alpha \oplus \gamma = \sum_{i=0}^k \omega^i (a_i + c_i)
= (\sum_{i=j+1}^k \omega^i (a_i + c_i)) + \omega^j (a_j + c_j) + \left(\sum_{i=0}^{j-1} \omega^i (a_j + c_j)\right)
= (\sum_{i=j+1}^k \omega^i (b_i + c_i)) + \omega^j (a_j + c_j) + \left(\sum_{i=0}^{j-1} \omega^i (a_i + c_i)\right)
< (\sum_{i=j+1}^k \omega^i (b_i + c_i)) + \omega^j (1 + a_j + c_j)
\leq (\sum_{i=j+1}^k (b_i + c_i) \omega^j) + \omega^j (b_j + c_j)
\leq \sum_{i=0}^k \omega^i (b_i + c_i) = \beta \oplus \gamma
\]

5. **Cancellation.** We first show that if $\sum_{i=0}^k \omega^i m_i = \sum_{i=0}^k \omega^i n_i$, then $m_i = n_i$ for all $i = 0, \ldots, k$ by induction on $k$. For $k = 0$ the claim is trivial. For $k > 0$, we first show $m_k = n_k$. By way of contradiction assume $m_k \neq n_k$. Without loss of generality, let $m_k > n_k$. Then $\sum_{i=0}^k \omega^i m_i \geq \omega^k m_k \geq \omega^k (n_k + 1) = \omega^k n_k + \omega^k > \omega^k n_k + \sum_{i=0}^{k-1} \omega^i n_i =$
\[ \sum_{i=0}^{k} \omega^i n_i, \] a contradiction. Thus \( m_k = n_k \). By left cancellation of ordinal addition, we obtain \( \sum_{i=0}^{k-1} \omega^i m_i = \sum_{i=0}^{k-1} \omega^i n_i \). By induction \( m_i = n_i \) for all \( i = 0, \ldots, k-1 \).

Let \( \alpha \oplus \gamma = \beta \oplus \gamma \). By assumption \( \sum_{i=0}^{k} \omega^i (a_i + c_i) = \alpha \oplus \gamma = \sum_{i=0}^{k} \omega^i (b_i + c_i) \).

Thus, we have \( a_i + c_i = b_i + c_i \) for all \( i = 0, \ldots, k \). The claim follows with right cancellation of + on natural numbers.

Natural addition gives rise to natural multiplication, a commutative, associative multiplication operation which distributes over natural addition. In the present work, we resort to two special cases, multiplication by natural numbers and multiplication by \( \omega \).

**Definition 2.2 (Natural Multiplication).** Let \( \alpha = \sum_{i=0}^{m} \omega^i a_i \). We define \( n \otimes \alpha \triangleq \sum_{i=0}^{m} \omega^i a_i n \) and \( \omega \otimes \alpha \triangleq \sum_{i=0}^{m} \omega^{i+1} a_i \).

**Lemma 2.2.** Natural multiplication has the following properties:

\[ 0 \otimes \alpha = 0 \quad 1 \otimes \alpha = \alpha \quad (n_1 + n_2) \otimes \alpha = n_1 \otimes \alpha \oplus n_2 \otimes \alpha \quad n \otimes \alpha \leq \omega \otimes \alpha \]

**Proof.** Let \( \alpha = \sum_{i=0}^{m} \omega^i a_i \).

1. \( 0 \otimes \alpha = 0 \). Immediate from the definition.

2. \( 1 \otimes \alpha = \alpha \). Immediate from the definition.

3. \( (n_1 + n_2) \otimes \alpha = n_1 \otimes \alpha \oplus n_2 \otimes \alpha \). We have \( (n_1 + n_2) \otimes \alpha = \sum_{i=0}^{m} \omega^i a_i (n_1 + n_2) = \sum_{i=0}^{m} \omega^i (a_i n_1 + a_i n_2) = (\sum_{i=0}^{m} \omega^i a_i n_1) \oplus (\sum_{i=0}^{m} \omega^i a_i n_2) = (n_1 \otimes \alpha) \oplus (n_2 \otimes \alpha) \).

4. \( n \otimes \alpha \leq \omega \otimes \alpha \). If \( \alpha = 0 \), the claim is trivial. If \( \alpha > 0 \), then one of the coefficients \( a_0, \ldots, a_m \) must be larger than zero. Let \( k \) be the largest \( k \) such that \( a_k > 0 \). Then \( n \otimes \alpha = \sum_{i=0}^{m} \omega^i a_i n = \sum_{i=0}^{k} \omega^i a_i n < \omega^{k+1} \leq \sum_{i=0}^{k} \omega^{i+1} a_i = \sum_{i=0}^{m} \omega^{i+1} a_i = \omega \otimes \alpha \).

\[ \square \]
3 Resources

<table>
<thead>
<tr>
<th>Capabilities</th>
<th>p, q [\vDash] get(\ell) \mid put(\ell) \mid al(\ell)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capability Sets</td>
<td>C, D [\vDash] \emptyset \mid C \uplus {p}</td>
</tr>
<tr>
<td>Invariant Maps</td>
<td>\Phi [\vDash] \emptyset \mid \Phi, \ell : A</td>
</tr>
<tr>
<td>Resources</td>
<td>R [\vDash] (C, \Phi, \alpha) \mid \frac{}{}</td>
</tr>
<tr>
<td>Resource Maps</td>
<td>\rho [\vDash] \emptyset \mid \rho, \ell \mapsto R</td>
</tr>
<tr>
<td>Step-Indices</td>
<td>i, j, k [\vDash] (\alpha, C)</td>
</tr>
</tbody>
</table>

In the logical relation, we incorporate the linearity of the type system in the form of resources [2, 4]. In our setting, each resource \(R\) is either invalid, meaning \(R = \frac{}{}\), or \(R\) is a triple \((C, \Phi, \alpha)\) where \(C\) is a set of capabilities, \(\Phi\) an invariant map, and \(\alpha\) an ordinal. In the logical relation, we relate values, expressions, and heaps with the resources they own. Intuitively, we interpret owning a resource \(R = (C, \Phi, \alpha)\) as the knowledge that the invariants \(\ell : A \in \Phi\) are enforced in the heap typing relation, the right to execute instructions corresponding to capabilities in \(C\), and the right to allocate \(\alpha\) new channels. The capability \(\text{get}(\ell)\) corresponds to the right to receive a value on channel \(\ell\), the capability \(\text{put}(\ell)\) to the right to send a value over channel \(\ell\), and the capability \(\text{al}(\ell)\) to the right to physically allocate \(\ell\), meaning to the right to add \(\ell\) to the heap. To combine two resources \(R\) and \(R'\), we combine their components.

**Definition 3.1 (Resource Addition).**

\[
\begin{align*}
(C, \Phi, \alpha) \oplus (C', \Phi', \alpha') & \triangleq (C \cup C', \Phi \cup \Phi', \alpha \oplus \alpha') & \text{if } C \not\equiv C' \text{ and } \Phi \text{ ag } \Phi' \\
R \oplus R' & \triangleq \frac{}{} & \text{otherwise}
\end{align*}
\]

where we write \(C \not\equiv C'\), if \(C\) and \(C'\) are disjoint, meaning \(C \cap C' = \emptyset\) and we write \(\Phi \text{ ag } \Phi'\), if two invariant maps \(\Phi\) and \(\Phi'\) agree, meaning \(\forall \ell \in \dom \Phi \cap \dom \Phi'. \Phi \ell = \Phi' \ell\).

We write \(\sqrt{R}\), if \(R\) is a valid resource, meaning \(R \not\equiv \frac{}{}\). A resource \(R\) is a subresource of \(R'\), written \(R \sqsubseteq R'\), if there exists some resource \(R''\) with \(R' = R \oplus R''\). We denote the empty resource by \(\varepsilon \triangleq (\emptyset, \emptyset, 0)\). In the following, it will sometimes be convenient to consider resources which only consist of capabilities \(R_{\text{cap}(C)} \triangleq (C, \emptyset, 0)\), invariants \(R_{\text{inv}(\Phi)} \triangleq (\emptyset, \Phi, 0)\), or ordinals \(R_{\text{ord}(\alpha)} \triangleq (\emptyset, \emptyset, \alpha)\). We write \(R_{\text{cap}(p_1, \ldots, p_n)}\) for the resource \(R_{\text{cap}(\{p_1, \ldots, p_n\})}\) and \(R_{\text{inv}(\ell_1: A_1, \ldots, \ell_n: A_n)}\) for the resource \(R_{\text{inv}(\{\ell_1: A_1, \ldots, \ell_n: A_n\})}\).

In the remainder of this section, we establish some basic properties about the use of invariants and resources in general.

**Lemma 3.1.**

1. If \(\Phi_1 \text{ ag } \Phi_2\), then \(\Phi_1 \cup \Phi_2\) is an invariant map. Further, \(\dom(\Phi_1 \cup \Phi_2) = \dom \Phi_1 \cup \dom \Phi_2\) and \((\Phi_1 \cup \Phi_2) \ell = \Phi_1 \ell \text{ for all } \ell \in \dom \Phi_1\) and \((\Phi_1 \cup \Phi_2) \ell = \Phi_2 \ell \text{ for all } \ell \in \dom \Phi_2\).
2. \(\Phi_1 \text{ ag } (\Phi_2 \cup \Phi_3)\) and \(\Phi_2 \text{ ag } \Phi_3\) iff \(\Phi_1 \text{ ag } \Phi_2 \text{ and } \Phi_1 \text{ ag } \Phi_3\) and \(\Phi_2 \text{ ag } \Phi_3\).
3. If \(\Phi_1 \text{ ag } \Phi_2\), then \(\Phi_2 \text{ ag } \Phi_1\).
4. If \(\Phi_1 \subseteq \Phi_2\), then \(\Phi_1 \text{ ag } \Phi_2\).
5. \(\Phi \text{ ag } \emptyset\).
Proof.

1. Assume $\Phi_1 \sqcup \Phi_2$, that is $\forall \ell \in \text{dom } \Phi_1 \cap \Phi_2. \Phi_1 \ell = \Phi_2 \ell$. Then $\Phi_1 \cup \Phi_2$ is a function. Clearly $\text{dom}(\Phi_1 \cup \Phi_2) = \text{dom } \Phi_1 \cup \text{dom } \Phi_2$. If $\ell \in \text{dom } \Phi_1$, then $(\Phi_1 \cup \Phi_2)\ell = \Phi_1 \ell$ regardless of whether $\ell \in \text{dom } \Phi_2$ or not with $\Phi_1 \sqcup \Phi_2$. Similarly, if $\ell \in \text{dom } \Phi_2$, then $(\Phi_1 \cup \Phi_2)\ell = \Phi_2 \ell$ regardless of whether $\ell \in \text{dom } \Phi_1$ or not with $\Phi_1 \sqcup \Phi_2$.

2. Let $\Phi_1 \sqcup (\Phi_2 \sqcup \Phi_3)$ and $\Phi_2 \sqcup \Phi_3$. With the first claim we know that $\Phi_2 \sqcup \Phi_3$ is an invariant map. We show $\Phi_1 \sqcup \Phi_2$. Let $\ell \in \text{dom } \Phi_1 \cap \text{dom } \Phi_2$. Then $\Phi_1 \ell = (\Phi_2 \sqcup \Phi_3) \ell = \Phi_2 \ell$ by the first claim. Analogously $\Phi_1 \sqcup \Phi_3$.

Let $\Phi_1 \sqcup \Phi_2$ and $\Phi_1 \sqcup \Phi_3$ and $\Phi_2 \sqcup \Phi_3$. Let $\ell \in \text{dom } \Phi_1 \cap (\text{dom } \Phi_2 \cup \text{dom } \Phi_3)$. If $\ell \in \text{dom } \Phi_2$, then $\Phi_1 \ell = \Phi_2 \ell = (\Phi_2 \sqcup \Phi_3) \ell$ by the first claim. If $\ell \in \text{dom } \Phi_3$, then $\Phi_1 \ell = \Phi_3 \ell = (\Phi_2 \sqcup \Phi_3) \ell$ by the first claim.

3. Let $\Phi_1 \sqcup \Phi_2$ and $\ell \in \text{dom } \Phi_2 \cap \text{dom } \Phi_1$. Then $\ell \in \text{dom } \Phi_1 \cap \text{dom } \Phi_2$. The claim follows.

4. Since $\Phi_1 \subseteq \Phi_2$, we have $\Phi_1 \ell = \Phi_2 \ell$ for all $\ell \in \text{dom } \Phi_1 \subseteq \text{dom } \Phi_2$.

5. The set $\text{dom } \Phi \cap \text{dom } \emptyset$ is empty. Thus, the claim holds trivially.

\[ \square \]

Lemma 3.2. The resource addition $\oplus$ is associative, commutative, and $\epsilon$ is an identity. The subresource relation $\sqsubseteq$ is reflexive, transitive, and $\epsilon \sqsubseteq R$ and $R \sqsubseteq R \cup R'$ for all $R, R'$. Validity extends to subresources, meaning if $\sqrt{R}$ and $R' \sqsubseteq R$, then $\sqrt{R'}$. The subresource relation is compatible with resource addition, meaning if $R \sqsubseteq R'$, then $R \oplus R_f \sqsubseteq R' \oplus R_f$. If $(C, \Phi, \alpha) \sqsubseteq (C', \Phi', \alpha')$, then $C \subseteq C'$ and $\Phi \subseteq \Phi'$ and $\alpha \leq \alpha'$.

Proof.

1. Associativity $\oplus$. Let $R_1, R_2, R_3$ be resources. We show $R_1 \oplus (R_2 \oplus R_3) = (R_1 \oplus R_2) \oplus R_3$.

If both sides are $\bot$, the claim follows.

Case $R_1 \oplus (R_2 \oplus R_3) = (C, \Phi, \alpha)$ for some $C, \Phi, \alpha$. Then by definition of $\oplus$ we have $C = C_1 \cup C_2 \cup C_3$ and $\Phi = \Phi_1 \cup \Phi_2 \cup \Phi_3$ and $\alpha = \alpha_1 \oplus \alpha_2 \oplus \alpha_3$ where $R_1 = (C_1, \Phi_1, \alpha_1)$ and $R_2 = (C_2, \Phi_2, \alpha_2)$ and $R_3 = (C_3, \Phi_3, \alpha_3)$ such that $C_1 \not\equiv C_2 \cup C_3$ and $C_2 \not\equiv C_3$ and $\Phi_1 \sqcup (\Phi_2 \cup \Phi_3)$ and $\Phi_2 \sqcup \Phi_3$. Thus, $C_1 \not\equiv C_2$ and $C_1 \not\equiv C_3$. Hence $(C_1 \cup C_2) \not\equiv C_3$.

Further, by Lemma 3.1 we have $\Phi_1 \sqcup \Phi_2$ and $\Phi_1 \sqcup \Phi_3$. Thus, $(\Phi_1 \cup \Phi_2) \sqcup \Phi_3$. With with Lemma 2.1, we obtain: $(R_1 \oplus R_2) \oplus R_3 = ((C_1 \cup C_2) \cup C_3, (\Phi_1 \cup \Phi_2) \cup \Phi_3, (\alpha_1 \oplus \alpha_2) \oplus \alpha_3) = R_1 \oplus (R_2 \oplus R_3)$

Case $(R_1 \oplus R_2) \oplus R_3 = (C, \Phi, \alpha)$ for some $C, \Phi, \alpha$. Then by definition of $\oplus$ we have $C = C_1 \cup C_2 \cup C_3$ and $\Phi = \Phi_1 \cup \Phi_2 \cup \Phi_3$ and $\alpha = \alpha_1 \oplus \alpha_2 \oplus \alpha_3$ where $R_1 = (C_1, \Phi_1, \alpha_1)$ and $R_2 = (C_2, \Phi_2, \alpha_2)$ and $R_3 = (C_3, \Phi_3, \alpha_3)$ such that $(C_1 \cup C_2) \not\equiv C_3$ and $C_1 \not\equiv C_2$ and $(\Phi_1 \cup \Phi_2) \sqcup \Phi_3$ and $\Phi_1 \sqcup \Phi_3$ and $\Phi_2 \sqcup \Phi_3$. Thus, $C_1 \not\equiv C_2$ and $C_2 \not\equiv C_3$. Hence $(C_1 \cup C_2) \not\equiv C_3$.

Further, by Lemma 3.1 $\Phi_1 \sqcup \Phi_3$ and $\Phi_2 \sqcup \Phi_3$ and thus $\Phi_1 \sqcup (\Phi_2 \cup \Phi_3)$.

With with Lemma 2.1, we obtain: $R_1 \oplus (R_2 \oplus R_3) = (C_1 \cup (C_2 \sqcup C_3), (\Phi_1 \cup (\Phi_2 \sqcup \Phi_3), (\alpha_1 \oplus (\alpha_2 \oplus \alpha_3)) = R_1 \oplus (R_2 \oplus R_3)$.
2. **Commutativity** $\oplus$. Let $R_1, R_2$ be resources. We show $R_1 \oplus R_2 = R_2 \oplus R_1$. If $R_1 = \emptyset$ or $R_2 = \emptyset$, the claim is trivial. Let $R_1 = (C_1, \Phi_1, \alpha_1)$ and $R_2 = (C_2, \Phi_2, \alpha_2)$. If not $C_1 \neq C_2$, then $R_1 \oplus R_2 = \emptyset = R_2 \oplus R_1$. If not $\Phi_1 \sqsubseteq \Phi_2$, then by Lemma 3.1 not $\Phi_2 \sqsubseteq \Phi_1$ and thus $R_1 \oplus R_2 = \emptyset = R_2 \oplus R_1$. Otherwise, $R_1 \oplus R_2 = (C_1 \cup C_2, \Phi_1 \cup \Phi_2, \alpha_1 \oplus \alpha_2) = (C_2 \cup C_1, \Phi_2 \cup \Phi_1, \alpha_2 \oplus \alpha_1) = R_2 \oplus R_1$ with Lemma 2.1.

3. **Identity** $\oplus$. Let $R$ be some resource. We show $R \oplus \emptyset = R$. If $R = \emptyset$, the claim is trivial.

   If $R = (C, \Phi, \alpha)$, we have $R \oplus \emptyset = (C \cup \emptyset, \Phi \cup \emptyset, \alpha \oplus 0) = (C, \Phi, \alpha) = R$ since $C \neq \emptyset$ and $\Phi \sqsubseteq \emptyset$ by Lemma 3.1.

4. **Reflexivity** $\sqsubseteq$. For any resource $R$, we have $R \oplus \emptyset = R$. Thus $R \sqsubseteq R$.

5. **Transitivity** $\sqsubseteq$. Let $R_1 \subseteq R_2$ and $R_2 \subseteq R_3$. Then there are $R_f$ and $R_g$ such that $R_1 \oplus R_f = R_2$ and $R_2 \oplus R_g = R_3$. Thus $R_1 \oplus (R_f \oplus R_g) = R_3$ and thus $R_1 \subseteq R_3$.

6. $\emptyset \sqsubseteq R$. Follows with $\emptyset \oplus R = R$.

7. **Validity**. Assume $\checkmark R$ and $R' \sqsubseteq R$. Then there is some $R_f$ such that $R = R' \oplus R_f$. Since $\checkmark R$, we have $R = (C, \Phi, \alpha)$ for some $C, \Phi, \alpha$. Thus $R' = (C', \Phi', \alpha')$ for some $C', \Phi', \alpha'$ since $R = R' \oplus R_f$.

8. **Compatibility**. Let $R \sqsubseteq R'$. Then there is some $R''$ such that $R \oplus R'' = R'$. Hence $R \oplus R_f \oplus R'' = R \oplus R'' \oplus R_f = R' \oplus R_f$. Thus $R \oplus R_f \subseteq R' \oplus R_f$.

9. **Decomposition**. Let $(C, \Phi, \alpha) \sqsubseteq (C', \Phi', \alpha')$. Then there is some $R_f$ such that $(C', \Phi', \alpha') = (C, \Phi, \alpha) \oplus R_f$. By definition of $\oplus$, we have $R_f = (C_f, \Phi_f, \alpha_f)$ for some $C_f, \Phi_f, \alpha_f$ such that $C' = C \cup C_f$ and $\Phi' = \Phi \cup \Phi_f$ and $\alpha' = \alpha \oplus \alpha_f$. Since $0 \leq \alpha_f$, we have $\alpha = \alpha \oplus 0 \leq \alpha \oplus \alpha_f = \alpha'$.

\[\square\]

**Lemma 3.3.**

1. $(C, \Phi, \alpha) = R_{\text{cap}}(C) \oplus R_{\text{inv}}(\Phi) \oplus R_{\text{ord}}(\alpha)$

2. If $\checkmark R_1 \oplus R_{\text{cap}}(C)$ and $R_2 \oplus R_{\text{cap}}(C) = R_1 \oplus R_{\text{cap}}(C)$, then $R_2 = R_1$.

3. If $\checkmark R_1 \oplus R_{\text{cap}}(C)$ and $R_2 \oplus R_{\text{cap}}(C) \sqsubseteq R_1 \oplus R_{\text{cap}}(C)$, then $R_2 \sqsubseteq R_1$.

4. If $R_2 \oplus R_{\text{ord}}(\alpha) = R_1 \oplus R_{\text{ord}}(\alpha)$, then $R_2 = R_1$.

5. If $R_2 \oplus R_{\text{ord}}(\alpha) \sqsubseteq R_1 \oplus R_{\text{ord}}(\alpha)$, then $R_2 \sqsubseteq R_1$.

6. If $\emptyset \sqsubseteq \Psi$, then $R_{\text{inv}}(\Phi) \oplus R_{\text{inv}}(\Psi) = R_{\text{inv}}(\Psi)$.

7. $\checkmark R$ iff $\checkmark R \oplus R_{\text{ord}}(\alpha)$.

**Proof.**

1. By definition since $C \neq \emptyset$ and $\Phi \sqsubseteq \emptyset$ by Lemma 3.1.

2. Assuming $\checkmark R_1 \oplus R_{\text{cap}}(C)$, we have $R_1 = (C_1, \Phi_1, \alpha_1)$ for some $C_1, \Phi_1, \alpha_1$ such that $C_1 \neq C$. Further, $R_2 = (C_2, \Phi_2, \alpha_2)$ such that $C_2 \neq C$. By assumption $(C_1 \cup C, \Phi_1, \alpha_1) = $
$(C_2 \cup C, \Phi_2, \alpha_2)$. Since $C_1 \neq C$ and $C_2 \neq C$, we have $C_1 = C_2$.

3. By assumption $R_2 \oplus R_{\text{cap}(C)} \oplus R_f = R_1 \oplus R_{\text{cap}(C)}$. Thus by the second claim, $R_2 \oplus R_f = R_1$. Hence $R_2 \subseteq R_1$.

4. If $R_1 = \frac{\pi}{2}$, then $R_2 = \frac{\pi}{2}$ and the claim follows. If $R_2 = \frac{\pi}{2}$, then $R_1 = \frac{\pi}{2}$ and the claim follows. Let $R_1 = (C_1, \Phi_1, \alpha_1)$ and $R_2 = (C_2, \Phi_2, \alpha_2)$ for some $C_1, C_2, \Phi_1, \Phi_2, \alpha_1, \alpha_2$. By assumption $(C_1, \Phi_1, \alpha_1 \oplus \alpha) = (C_2, \Phi_2, \alpha_2 \oplus \alpha)$. The claim follows with right cancellation of natural addition, see Lemma 2.1.

5. By assumption $R_2 \oplus R_{\text{ord}(\alpha)} \oplus R_f = R_1 \oplus R_{\text{ord}(\alpha)}$. Thus by the second claim, $R_2 \oplus R_f = R_1$. Hence $R_2 \subseteq R_1$.

6. If $\Phi \subseteq \Psi$, then $\Phi \sqsubseteq \Psi$ by Lemma 3.1. Thus $R_{\text{inv}(\Phi)} \oplus R_{\text{inv}(\Psi)} = R_{\text{inv}(\Phi \cup \Psi)} = R_{\text{inv}(\Psi)}$.

7. Follows by case analysis on $R$. 

\[\Box\]
4 Logical Relation

In this section, we define the logical relation of the language $\lambda_{\text{chan}}$. In Figure 4, we give the definition of the type interpretations. In Section 4.1, we prove that the type interpretations are Kripke logical relations. In Section 4.2 and Section 4.3, we prove several properties about the type interpretations which enable compact proofs of the compatibility lemmas. In Section 4.4, we define the semantic typing judgement $\Gamma \mid e : A$ and in Section 4.5, we prove the compatibility lemmas required to prove that its a sound model of the syntactic typing judgement $\Gamma \vdash e : A$.

**Step-Indices** As step-indices, we use pairs of ordinals and sets of capabilities $i, j, k ::= (\alpha, C)$. On step-indices, we define the lexicographic ordering:

$$(\alpha, C) < (\alpha', C') \iff \alpha < \alpha' \lor (\alpha = \alpha' \land C \subseteq C')$$

and as usual define $i \leq j \iff i < j \lor i = j$.

We define the value relation $V[A]_i$, the heap typing $H[\Phi]_i$, and the expression relation $E[A]_i$ by recursion on the step-index $i$. For a fixed step-index $i$, we define the value relation $V[A]_i$ and the expression relation $E[A]_i$ by recursion on the type $A$. More precisely, at step-index $i$ the heap typing $H[\Phi]_i$, depends on $V[B]_j$ for $j < i$ and arbitrary types $B$, the value relation $V[A]_i$, depends on $E[B]_j$ for $j < i$ and arbitrary types $B$, and the expression relation $E[A]_i$ depends on $V[A]_j$ for $j \leq i$ and on $H[\Phi]_j$ for $j < i$ and arbitrary $\Phi$.

**Lemma 4.1.** If $(\alpha, C) \leq (\alpha', C')$, then $(\alpha, C \setminus D) \leq (\alpha', C' \setminus D)$.

**Proof.** If $\alpha < \alpha'$, then $(\alpha, C \setminus D) \leq (\alpha, C') < (\alpha', C' \setminus D)$. If $\alpha = \alpha'$, then $C \subseteq C'$. Hence $C \setminus D \subseteq C' \setminus D$. Thus $(\alpha, C \setminus D) \leq (\alpha, C' \setminus D) = (\alpha', C' \setminus D)$. □

**Value Relation** In the value relation $V[A]_i$, we relate values that semantically inhabit the type $A$ with the resources they own. Unit, Booleans, and natural numbers only own the empty resource $\epsilon$. Inhabitants of Get $A$ are locations $\ell$ that own the resource $R_{\text{get}}(\ell, A) \triangleq (\{\text{get}(\ell)\}, \{\ell : A\}, 0)$. Ownership of get$(\ell)$ entails the right to perform a get on location $\ell$ and ownership of $\ell : A$ guarantees that the invariant $\ell : A$ is satisfied by the heap in the heap typing relation $H[\Phi]_i$. Analogously, inhabitants of Put $A$ are locations $\ell$ that own the resource $R_{\text{put}}(\ell, A) \triangleq (\{\text{put}(\ell)\}, \{\ell : A\}, 0)$. For linear pairs $A \otimes B$, we combine the resources owned by the individual components. In the interpretation of the linear function type $A \rightarrow B$, a function is related to those resources that are required to execute the body safely, provided the resources owned by the argument are added.

**Logical State** There is no physical difference between a channel in its initial state and a channel after it has been used to exchange a value: in both cases, the channel location stores the value $E$. However, there is a logical difference, which matters for keeping track of resource
Step-Indices and Logical State

Step-Indices \( i, j, k \) ::= (\( \alpha, C \) )

Logical State \( s \) ::= Start | Cont | Val | Done

Logical State Map \( \sigma \) ::= \( \emptyset \) | \( \sigma, \ell \mapsto s \)

\( \sigma \rightarrow \sigma[\ell \mapsto \text{Start}] \) if \( \ell \notin \text{dom} \sigma \)

\( \sigma \rightarrow \sigma[\ell \mapsto \text{Done}] \) if \( \sigma \ell = \text{Val} \)

\( \sigma \rightarrow \sigma[\ell \mapsto \text{Val}] \) if \( \sigma \ell = \text{Start} \)

\( \sigma \rightarrow \sigma[\ell \mapsto \text{Done}] \) if \( \sigma \ell = \text{Cont} \)

\( \sigma \rightarrow \sigma[\ell \mapsto \text{Cont}] \) if \( \sigma \ell = \text{Start} \)

Value Relation

\( \mathcal{V}[\mathbb{I}]_i \triangleq \{((), \epsilon)\} \)

\( \mathcal{V}[\mathbb{B}]_i \triangleq \{(\text{true}, \epsilon), (\text{false}, \epsilon)\} \)

\( \mathcal{V}[\mathbb{N}]_i \triangleq \{(n, \epsilon) | n \in \mathbb{N}\} \)

\( \mathcal{V} \{\text{Get } A\}_i \triangleq \{(\ell, (\{\text{get}(\ell)\}, \{\ell : A\}, 0))\} \)

\( \mathcal{V} \{\text{Put } A\}_i \triangleq \{(\ell, (\{\text{put}(\ell)\}, \{\ell : A\}, 0))\} \)

\( \mathcal{V} \{A \otimes B\}_i \triangleq \{(v_1, v_2) | (v_1, v_2) \in \mathcal{V}[A]_i \text{ and } (v_2, v_2) \in \mathcal{V}[B]_i\} \)

\( \mathcal{V} \{A \rightarrow B\}_i \triangleq \{((\lambda x.e), R_e) | \forall j \leq i, (v, R_v) \in \mathcal{V}[A]_j, (e[v/x], R_v + R_e) \in \mathcal{E}[B]_j\} \)

Heap Typing

\( \mathcal{H}[\Phi]_i \triangleq \{(h, \sigma, \rho) | \text{dom } \Phi = \text{dom } h = \text{dom } \sigma = \text{dom } \rho \text{ and } \forall \ell : A \in \Phi. (h, \ell, \sigma, \rho) \in \mathcal{H}[\ell : A]_i\} \)

\( \mathcal{H} \{\ell : A\}_i \triangleq \{(E, \text{Start}, \epsilon), (E, \text{Done}, \epsilon)\} \)

\( \mathcal{H} \{\ell : A\}_i \triangleq \{(V(v), \text{Val}, R) | \exists C'. C = C' \cup \{\text{put}(\ell)\} \text{ and } (v, R) \in \mathcal{V}[A]_i\} \)

\( \mathcal{H} \{\ell : A\}_i \triangleq \{(C(v), \text{Cont}, R) | \exists C'. C = C' \cup \{\text{get}(\ell)\} \text{ and } (v, R) \in \mathcal{V}[A \rightarrow B]_i\} \)

Expression Relation

\( \mathcal{E}[\ell : A]_i \triangleq \{(e, R_e) \mid \forall j \leq i, R_f, \Phi, h, \sigma, \rho. \mathcal{R}_j(R_e, R_f, \Phi, h, \sigma, \rho) \Rightarrow \}

\text{and } (e, h) \leadsto^* (v, h') \text{ and } \sigma \rightarrow^* \sigma' \text{ and } (v, R_v) \in \mathcal{V}[A]_k \}

where

\( \mathcal{R}_{i, C}(R_e, R_f, \Phi, h, \sigma, \rho) \triangleq (h, \sigma, \rho) \in \mathcal{H}[\Phi]_i \text{ and used } \sigma \# C \text{ and } \}

\( \exists D, C = D \upsilon \text{ idx } \sigma \text{ and } R_e + R_f + \bigoplus_{\ell \in \text{dom } \rho} \rho \ell \subseteq (D, \Phi, \alpha) \)

id(\ell, \text{Val}) = \{\text{put}(\ell)\} \quad \text{used}(\ell, \text{Done}) = \{\text{al}(\ell), \text{get}(\ell), \text{put}(\ell)\}

id(\ell, \text{Cont}) = \{\text{get}(\ell)\} \quad \text{used}(\ell, \text{s}) = \{\text{al}(\ell)\} \quad \text{othw.}

id(\ell, s) = \emptyset \quad \text{othw.} \quad \text{used } \sigma = \bigcup_{\ell \in \text{dom } \sigma} \text{used}(\ell, \sigma \ell)

Figure 4: Logical Relation
usage in our logical relation. Thus, to distinguish both states, we introduce the notion of the logical state of a channel. The logical state of every channel in the heap is tracked in a state map \( \sigma \). We evolve it according to the transition relation \( \sigma \rightarrow \sigma' \). Each fresh channel starts in the state \( \text{Start} \) and is then advanced to \( \text{Done} \), either by storing a value or a continuation in between.

**Heap Typing** The heap typing relation \( H[A] \), ensures that the values in the heap satisfy the invariants \( \ell : A \in \Phi \). More precisely, the heap typing relation \( H[A] \), relates the logical state \( \sigma \) with the physical heap \( h \) and the resources owned by values in the heap \( \rho \) such that the invariants in \( \Phi \) are upheld.

**Expression Relation** In the expression relation \( E[A] \), to execute an expression \( e \) owning resource \( R_e \), we assume some global invariant map \( \Phi \), a current logical state map \( \sigma \), a current heap \( h \), and a current resource map \( \rho \). To remain compositional, following the approach of Ahmed et al. [1], we additionally assume some frame resource \( R_f \), representing the resource owned by a potential context in which \( e \) could be executed. We then show that the resource interpretation \( \text{RI} \) is preserved during the execution with a potential decrease in the step-index. That is, we assume the resource interpretation is initially satisfied and show that at the end of the execution the resource interpretation is satisfied for some resource \( R_v \) owned by the result \( v \), some extended invariant map \( \Phi' \), some updated logical state map \( \sigma' \), some updated heap \( h' \), and some updated resource map \( \rho' \). After the execution, we ensure that the logical state map was advanced according to the transition relation \( \rightarrow \) and that the result \( v \) is contained in the value relation.

The resource interpretation \( \text{RI}_{\alpha,C}(R_e, R_f, \Phi, h, \sigma, \rho) \) serves three purposes: First, it ensures that the heap is well-typed given the current invariant map \( \Phi \) with \( (h, \sigma, \rho) \in H[A]_{\alpha,C}. \) Second, with \( R_e \oplus R_f \oplus \bigoplus_{\ell \in \text{dom} \rho} \rho \ell \subseteq (D, \Phi, \alpha) \) we ensure ownership of resources has the intuitive meaning. For the resources owned by different program components, the invariant maps must all agree and be contained in the global invariant map \( \Phi \), the capability sets must be pairwise disjoint and contained in the step-index since \( D \subseteq C \), and the sum of all ordinals must be at most \( \alpha \). The latter guarantees that we can compositionally decrease the global step-index by locally decreasing the ordinal in the resource of an expression. For example, if the expression resource is \( R_e = R' \oplus R_{\text{ord}(1)} \), we can decrease \( \alpha \) and allocate new capabilities by giving up the resource \( R_{\text{ord}(1)} \).

Third, the resource interpretation enforces that capabilities are used according to the following state transition system with tokens [5], where “tokens” in our case are capabilities:

```
Start
\[ al(\ell) \]

\[ \text{Cont} \]
\[ al(\ell), get(\ell) \]

\[ \text{Val} \]
\[ al(\ell), put(\ell) \]

put(\ell)

get(\ell)

\[ \text{Done} \]
\[ al(\ell), get(\ell), put(\ell) \]
```
Below every state are the capabilities currently owned by that state. Formally, for a state map \( \sigma \), the capabilities currently owned by \( \sigma \) are given by \( \text{idx} \sigma \cup \text{used} \sigma \). The functions \( \text{idx} \sigma \) and \( \text{used} \sigma \) distinguish between two modes in which capabilities can be owned by \( \sigma \), depending on whether they are still contained in the step-index or not. We transfer the capability \( \text{put}(\ell) \) (resp. \( \text{get}(\ell) \)) to \( \text{idx} \sigma \) at the point where a value (resp. continuation) is stored in the heap. We move them to \( \text{used} \sigma \) at the point where the continuation or value is retrieved during the execution. If the state map \( \sigma \) owns capabilities, then no program component can own those capabilities since 

\[
R_e \oplus R_f \oplus \bigoplus_{\ell \in \text{dom } \rho} \rho \ell \subseteq (D, \Phi, \alpha)
\]

where \( D \# \text{idx} \sigma \) and \( D \# \text{used} \sigma \).

**Lemma 4.2.**

1. \( \text{idx}(\ell, s) \# \text{used}(\ell, s) \)
2. If \( \ell \neq \ell' \), then \( \text{idx}(\ell, s) \# \text{used}(\ell', s') \).
3. \( R_{\text{cap}(\text{idx} \sigma)} = \bigoplus_{\ell \in \text{dom } \sigma} R_{\text{cap}(\text{idx}(\ell, \sigma \ell))} \).
4. If \( \ell \in \text{dom } \sigma \), then \( \text{al}(\ell) \in \text{used } \sigma \).
5. \( \text{al}(\ell) \notin \text{idx}(\ell, s) \) and \( \text{al}(\ell) \notin \text{idx} \sigma \).

**Proof.**

1. Immediate from the definition of \( \text{idx} \) and \( \text{used} \).
2. Immediate from the definition of \( \text{idx} \) and \( \text{used} \) since the capabilities in \( \text{idx}(\ell, s) \) only mention the location \( \ell \) and the capabilities in \( \text{used}(\ell', s') \) only mention the location \( \ell' \).
3. For any location \( \ell \), the set \( \text{idx}(\ell, \sigma \ell) \) contains only capabilities with the location \( \ell \). Thus

\[
\text{idx}(\ell, \sigma \ell) \# \bigcup_{\ell' \in L, \ell \neq \ell'} \text{idx}(\ell', \sigma \ell')
\]

for all \( \ell \in \text{dom } \sigma \) and \( L \subseteq \text{dom } \sigma \). Thus, \( \bigoplus_{\ell \in \text{dom } \sigma} R_{\text{cap}(\text{idx}(\ell, \sigma \ell))} \) is valid and:

\[
\bigoplus_{\ell \in \text{dom } \sigma} R_{\text{cap}(\text{idx}(\ell, \sigma \ell))} = R_{\text{cap}(\bigcup_{\ell \in \text{dom } \sigma} \text{idx}(\ell, \sigma \ell))} = R_{\text{cap}(\text{idx} \sigma)}
\]

4. We have \( \text{al}(\ell) \in \text{used}(\ell, s) \) for any \( s \). Thus, if \( \ell \in \text{dom } \sigma \), we have \( \text{al}(\ell) \in \text{used}(\ell, \sigma \ell) \subseteq \bigcup_{\ell \in \text{dom } \sigma} \text{used}(\ell, \sigma \ell) = \text{used } \sigma \).
5. Immediate from the definition of \( \text{idx} \).

\[ \square \]

### 4.1 Kripke Logical Relation

The logical logical relation is a Kripke relation in the step-index. More precisely, the value relation \( \mathcal{V}[A] \), and the expression relation \( \mathcal{E}[A] \), are closed under smaller step-indices. The heap typing relation \( \mathcal{H}[\Phi] \) is almost closed under smaller step-indices as the following lemma shows:
Lemma 4.3. Let $i = (\alpha, C) \geq (\beta, D) = j$.

1. $\mathcal{E}[A]_i \subseteq \mathcal{E}[A]_j$
2. $\mathcal{V}[A]_i \subseteq \mathcal{V}[A]_j$
3. If $(hv, s, R) \in \mathcal{H}_\mathcal{V}[\ell : A]_i$ and $\text{idx}(\ell, s) \subseteq D$, then $(hv, s, R) \in \mathcal{H}_\mathcal{V}[\ell : A]_j$.
4. If $(h, \sigma, \rho) \in \mathcal{H}_\mathcal{P}[]_i$ and $\text{idx} \sigma \subseteq D$, then $(h, \sigma, \rho) \in \mathcal{H}_\mathcal{P}[]_j$.

Proof:

1. The claim $\mathcal{E}[A]_i \subseteq \mathcal{E}[A]_j$ is immediate from the definition: For any $k \leq j$, we have $k \leq i$ by transitivity. The claim then follows by assumption.

2. By induction on $A$. The claim is trivial for $\mathbb{1}, \mathbb{B}, \mathbb{N}, \text{Get A}$, and $\text{Put A}$.

For $A \otimes B$, let $(v, R) \in \mathcal{V}[A \otimes B]_i$. Then there are $v_1, v_2$ and $R_1, R_2$ such that $v = (v_1, v_2)$, $R = R_1 \oplus R_2$, $(v_1, R_1) \in \mathcal{V}[A]_i$, and $(v_2, R_2) \in \mathcal{V}[B]_i$. By induction, we know $(v_1, R_1) \in \mathcal{V}[A]_j$ and $(v_2, R_2) \in \mathcal{V}[B]_j$. Hence $((v_1, v_2), R_1 \oplus R_2) \in \mathcal{V}[A \otimes B]_j$.

For $A \rightarrow B$ the claim follows immediately from the definition of $\mathcal{V}[A \rightarrow B]$: Let $(\lambda x.e, R_e) \in \mathcal{V}[A \rightarrow B]_i$ and $k \leq j$ and $(v, R_v) \in \mathcal{V}[A]_k$. Then $k \leq i$. Hence, by $(\lambda x.e, R_e) \in \mathcal{V}[A \rightarrow B]_i$, we have $(e[v/x], R_v \oplus R_e) \in \mathcal{V}[B]_k$.

3. By case analysis on $s$. Trivial for Start and Done.

For $\text{Val}$, we have $hv = \mathcal{V}(v)$ and $C = C' \uplus \{\text{put}(\ell)\}$ for some $C'$ and $v$ such that $(v, R) \in \mathcal{V}[A]_{\alpha, C'}$. Since $\text{put}(\ell) \in \text{idx}(\ell, \text{Val}) \subseteq D$, we know that $D = D' \uplus \{\text{put}(\ell)\}$ for some $D'$. Since $(\beta, D) \leq (\alpha, C)$, we have $(\beta, D') \leq (\alpha, C')$ by Lemma 4.1. By the second claim, it follows that $(v, R) \in \mathcal{V}[A]_{\beta, D'}$.

For $\text{Cont}$, we have $hv = C(v)$ and $C = C' \uplus \{\text{get}(\ell)\}$ for some $C'$ and $v$ such that $(v, R) \in \mathcal{V}[A \rightarrow 1]_{\alpha, C'}$. Since $\text{get}(\ell) \in \text{idx}(\ell, \text{Cont}) \subseteq D$, we know that $D = D' \uplus \{\text{get}(\ell)\}$ for some $D'$. Since $(\beta, D) \leq (\alpha, C)$, we have $(\beta, D') \leq (\alpha, C')$ by Lemma 4.1. By the second claim, it follows that $(v, R) \in \mathcal{V}[A \rightarrow 1]_{\beta, D'}$.

4. Follows immediately from the first claim since $\text{idx}(\ell, s) \subseteq \text{idx} \sigma$ for all $\ell \mapsto s \in \sigma$.

4.2 Resource Interpretation Properties

In the compatibility lemmas for the channel operations, we update the resource interpretation during the execution. The lemmas in this section capture how we update the resource interpretation.

For the resource interpretation $\text{RI}_i(R_e, R_f, \Phi, h, \sigma, \rho)$ we use the following equivalent formulation in the proofs:
Lemma 4.4. \( \text{RL}_{\alpha,C}(R_e, R_f, \Phi, h, \sigma, \rho) \) iff  
\[(h, \sigma, \rho) \in \mathcal{H}[\Phi]_{\alpha,C} \text{ and used } \sigma \neq C \text{ and } \sum(\rho) \oplus R_e \oplus R_f \oplus R_{\text{cap}(\text{idx}, \sigma)} \subseteq (C, \Phi, \alpha) \]
where \( \sum(\rho) \triangleq \bigoplus_{\ell \in \text{dom } \rho} \rho^\ell \)

Proof. For the forward direction, let \( \text{RL}_{\alpha,C}(R_e, R_f, \Phi, h, \sigma, \rho) \). Then \((h, \sigma, \rho) \in \mathcal{H}[\Phi]_{\alpha,C} \) and used \( \sigma \neq C \) and \( C = D \uplus \text{idx } \sigma \) and \( R_e \oplus R_f \oplus \sum(\rho) \subseteq (D, \Phi, \alpha) \). By Lemma 3.2, we have:
\[(R_e \oplus R_f \oplus \sum(\rho)) \oplus R_{\text{cap}(\text{idx}, \sigma)} \subseteq (D, \Phi, \alpha) \oplus R_{\text{cap}(\text{idx}, \sigma)} = (C, \Phi, \alpha) \]
The claim follows with commutativity of \( \oplus \).

For the backward direction, let \((h, \sigma, \rho) \in \mathcal{H}[\Phi]_{\alpha,C} \) and used \( \sigma \neq C \) and \( C = D \uplus \text{idx } \sigma \) and \( R_e \oplus R_f \oplus \sum(\rho) \subseteq (D, \Phi, \alpha) \). Then \( \text{idx } \sigma \subseteq C \) by Lemma 3.2. Thus \( C = D \uplus \text{idx } \sigma \) for some capability set \( D \). Hence, we have \( \sum(\rho) \oplus R_e \oplus R_f \oplus R_{\text{cap}(\text{idx}, \sigma)} \subseteq (D, \Phi, \alpha) \oplus R_{\text{cap}(\text{idx}, \sigma)} \). Thus, by Lemma 3.3, we have \( \sum(\rho) \oplus R_e \oplus R_f \subseteq (D, \Phi, \alpha) \). The claim follows with commutativity of \( \oplus \).

We can move resources from the expression into the frame and back:

Lemma 4.5. \( \text{RL}_1(R_1 \oplus R_2, R_f, \Phi, h, \sigma, \rho) \) iff \( \text{RL}_1(R_1, R_2 \oplus R_f, \Phi, h, \sigma, \rho) \)

Proof. Follows immediately from associativity of \( \oplus \):
\[
\sum(\rho) \oplus (R_1 \oplus R_2) \oplus R_f \oplus R_{\text{cap}(\text{idx}, \sigma)} = \sum(\rho) \oplus R_1 \oplus (R_2 \oplus R_f) \oplus R_{\text{cap}(\text{idx}, \sigma)}
\]

The resource interpretation is affine, in particular in the ordinal resources. For example, we can trade in the right to allocate \( \omega \) fresh locations for the right to allocate \( n \) fresh locations for some \( n \in \mathbb{N} \).

Lemma 4.6. If \( \text{RL}_{\beta,C}(R_e \oplus R_{\text{ord}(\alpha)}, R_f, \Phi, h, \sigma, \rho) \) and \( \alpha' \leq \alpha \), then there is some \( \beta' \leq \beta \) such that \( \text{RL}_{\beta',C}(R_e \oplus R_{\text{ord}(\alpha')}, R_f, \Phi, h, \sigma, \rho) \).

Proof. By Lemma 3.2, we have \( R_{\text{ord}(\alpha)} \subseteq \sum(\rho) \oplus R_e \oplus R_{\text{ord}(\alpha)} \oplus R_f \oplus R_{\text{cap}(\text{idx}, \sigma)} \subseteq (C, \Phi, \beta) \) and thus \( \beta = \beta_f \oplus \alpha \) for some \( \beta_f \). We define \( \beta' \triangleq \beta_f \oplus \alpha' \). Then \( \beta' \leq \beta \) by Lemma 2.1. Thus, \( (\beta', C) \leq (\beta, C) \). By Lemma 3.2, we have \( \text{idx } \sigma \subseteq C \). Since \((h, \sigma, \rho) \in \mathcal{H}[\Phi]_{\beta,C} \), we have \( (h, \sigma, \rho) \in \mathcal{H}[\Phi]_{\beta',C} \) by Lemma 4.3. Lastly:
\[
\sum(\rho) \oplus R_e \oplus R_{\text{ord}(\alpha)} \oplus R_f \oplus R_{\text{cap}(\text{idx}, \sigma)} \subseteq (C, \Phi, \beta) \\
\Rightarrow \sum(\rho) \oplus R_e \oplus R_{\text{ord}(\alpha)} \oplus R_f \oplus R_{\text{cap}(\text{idx}, \sigma)} \subseteq (C, \Phi, \beta_f) \oplus R_{\text{ord}(\alpha)} \quad (\beta = \beta_f \oplus \alpha) \\
\Rightarrow \sum(\rho) \oplus R_e \oplus R_f \oplus R_{\text{cap}(\text{idx}, \sigma)} \subseteq (C, \Phi, \beta_f) \quad (\text{Lemma 3.3}) \\
\Rightarrow \sum(\rho) \oplus R_e \oplus R_{\text{ord}(\alpha')} \oplus R_f \oplus R_{\text{cap}(\text{idx}, \sigma)} \subseteq (C, \Phi, \beta_f) \oplus R_{\text{ord}(\alpha')} \quad (\text{Lemma 3.2}) \\
\Rightarrow \sum(\rho) \oplus R_e \oplus R_{\text{ord}(\alpha')} \oplus R_f \oplus R_{\text{cap}(\text{idx}, \sigma)} \subseteq (C, \Phi, \beta') \quad (\beta' = \beta_f \oplus \alpha') \]

\[16\]
Lemma 4.7. If $\text{RI}_{\alpha,C}(R_1 \oplus R_2, R_f, \Phi, h, \sigma, \rho)$, then $\text{RI}_{\alpha,C}(R_1, R_f, \Phi, h, \sigma, \rho)$.

Proof. By assumption, we have $(h, \sigma, \rho) \in H[\Phi]_{\alpha,C}$ and used $\sigma \neq C$ and $\text{sum}(\rho) \oplus R_1 \oplus R_2 \oplus R_f \oplus R_{\text{cap}(\text{idx} \sigma)} \subseteq (C, \Phi, \alpha)$. Thus $\text{sum}(\rho) \oplus R_1 \oplus R_f \oplus R_{\text{cap}(\text{idx} \sigma)} \subseteq (C, \Phi, \alpha)$ and the claim follows. \qed

We can trade in one ordinal resource for the capabilities associated with some fresh channel $\ell$.

Lemma 4.8. If $\text{RI}_{\alpha,C}(R_e \oplus R_{\text{ord}(1)}, R_f, \Phi, h, \sigma, \rho)$, then there is some $\ell$ and $\beta$ with $\alpha = \beta \oplus 1$ and $\text{RI}_{\beta,C \cup \{\text{al}(\ell), \text{put}(\ell), \text{get}(\ell)\}}(R_e \oplus R_{\text{cap}(\text{get}(\ell), \text{put}(\ell), \text{al}(\ell))), R_f, \Phi, h, \sigma, \rho)$.

Proof. By Lemma 3.2, we have $R_{\text{ord}(1)} \subseteq \text{sum}(\rho) \oplus R_e \oplus R_{\text{ord}(1)} \oplus R_f \ominus R_{\text{cap}(\text{idx} \sigma)} \subseteq (C, \Phi, \alpha)$ and thus $\alpha = \beta \oplus 1$ for some $\beta$. Let $\ell$ be some fresh location such that $\ell \notin \text{dom} \sigma$ and $\text{al}(\ell), \text{get}(\ell), \text{put}(\ell) \notin C$. We abbreviate $D \equiv \{\text{al}(\ell), \text{get}(\ell), \text{put}(\ell)\}$. We have:

\[
\text{sum}(\rho) \oplus R_e \oplus R_{\text{ord}(1)} \oplus R_f \oplus R_{\text{cap}(\text{idx} \sigma)} \subseteq (C, \Phi, \alpha)
\]

\[
\Rightarrow \text{sum}(\rho) \oplus R_e \oplus R_{\text{ord}(1)} \oplus R_f \oplus R_{\text{cap}(\text{idx} \sigma)} \subseteq (C, \Phi, \beta) \oplus R_{\text{cap}(D)} \quad (\text{Lemma 3.3})
\]

By assumption, we have $(h, \sigma, \rho) \in H[\Phi]_{\alpha,C}$. Further, $\text{idx} \sigma \subseteq C \cup D$ by Lemma 3.2. Since $(\beta, C \cup D) < (\alpha, C)$, we have $(h, \sigma, \rho) \in H[\Phi]_{\beta,C \cup D}$ by Lemma 4.3. Since by assumption used $\sigma \neq C$, we have used $\sigma \neq C \cup D$ as $D = \{\text{al}(\ell), \text{get}(\ell), \text{put}(\ell)\}$ and $\ell \notin \text{dom} \sigma$. \qed

In the above lemma, we extend the capability set $C$ in the step-index with the capabilities $\text{al}(\ell), \text{get}(\ell)$, and $\text{put}(\ell)$. Due to the lexicographic ordering, we obtain the resource interpretation at a smaller step-index by decreasing $\alpha$. We refer to the above lemma as the logical allocation since the heap $h$ is left unchanged. The following lemma allows for the physical allocation of $\ell$, meaning trading the capability $\text{al}(\ell)$ in for extending the heap $h$ by the fresh location $\ell$.

Lemma 4.9. If $\text{RI}_{\alpha,C}(R_e \oplus R_{\text{cap}(\text{al}(\ell))}, R_f, \Phi, h, \sigma, \rho)$, then $\text{RI}_{\alpha,C \setminus \{\text{al}(\ell)\}}(R_e \oplus R_{\text{inv}(\ell)} A, R_f, (\Phi, \ell : A), h[\ell \mapsto E], \sigma[\ell \mapsto \text{Start}], \rho[\ell \mapsto e])$.

Proof. By Lemma 3.2, we have $R_{\text{cap}(\text{al}(\ell))} \subseteq \text{sum}(\rho) \oplus R_e \oplus R_{\text{cap}(\text{al}(\ell))} \oplus R_f \oplus R_{\text{cap}(\text{idx} \sigma)} \subseteq (C, \Phi, \alpha)$. Hence, we have $\{\text{al}(\ell)\} \subseteq C$ and thus $C = C' \cup \{\text{al}(\ell)\}$ for some $C'$. Since used $\sigma \neq C$, we have $\text{al}(\ell) \notin \text{used} \sigma$. Thus $\ell \notin \text{dom} \sigma$ by Lemma 4.2. From $(\sigma, h, \rho) \in H[\Phi]_{\alpha,C}$, we know $\text{dom} \Phi = \text{dom} \rho = \text{dom} \sigma$ and thus $\ell \notin \text{dom} \Phi$ and $\ell \notin \text{dom} \rho$. We
have:

\[
\text{sum}(\rho) \oplus R_e \oplus R_{\text{cap}(al(\ell))} \oplus R_f \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \subseteq (C, \Phi, \alpha) \\
\Rightarrow \text{sum}(\rho) \oplus R_e \oplus R_{\text{cap}(al(\ell))} \oplus R_f \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \subseteq (C', \Phi, \alpha) \oplus R_{\text{cap}(al(\ell))} \\
\Rightarrow \text{sum}(\rho) \oplus R_e \oplus R_f \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \subseteq (C', \Phi, \alpha) \\
\Rightarrow \text{sum}(\rho) \oplus R_e \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \oplus R_f \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \subseteq (C', \Phi, \alpha) \oplus R_{\text{inv}(\ell, A)} \\
\Rightarrow \text{sum}(\rho) \oplus R_e \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \oplus R_f \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \subseteq (C', \Phi, \ell : A, \alpha) \\
\Rightarrow \text{sum}(\rho[\ell \mapsto \epsilon]) \oplus R_e \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \oplus R_f \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \subseteq (C', \Phi, \ell : A, \alpha) 
\]

\text{(Lemma 3.3)}

where \((\ast)\) is true since \(\ell \notin \text{dom } \rho \) and \(\ell \notin \text{dom } \sigma\). We have \(\text{idx}(\sigma[\ell \mapsto \text{Start}]) \subseteq C'\) by \text{Lemma 3.2}. Thus, by \text{Lemma 4.3}, we have \((h, \sigma, \rho) \in H[\Phi]_{\alpha, C'}\). Since \((E, \text{Start}, \epsilon) \in H[\Phi]_{\ell : [A]}_{\alpha, C'},\) we have \((h[\ell \mapsto E], \sigma[\ell \mapsto \text{Start}], \rho[\ell \mapsto \epsilon]) \in H[\Phi, \ell : A]_{\alpha, C'}\). By assumption used \(\alpha \neq C\). Hence used \(\sigma[\ell \mapsto \text{Start}] = \{\text{al}(\ell)\} \cup \text{used } \sigma \neq C'\).

**Lemma 4.10.** If \(\text{Rl}_{\alpha, C}(R_e, R_f, \Phi, h, \sigma, \rho)\) and \(\ell : A \in \Phi\) and \((h, s, R) \in H[\Phi]_{\ell : [A]}_{\alpha, C}\) and used \((\ell, s) \setminus D \text{ and} \)

\[
\rho \ell \oplus R_e \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \oplus R_{\text{inv}(\Phi)} = R \oplus R'_{e} \oplus R_{\text{cap}(\text{idx} \cdot s)} \oplus R_{\text{cap}(D)} \oplus R_{\text{inv}(\Phi)},
\]

then \(\text{Rl}_{\alpha, C \setminus D}(R'_{e, R_f, \Phi, h}[\ell \mapsto h], \sigma[\ell \mapsto s], \rho[\ell \mapsto R])\).

**Proof.** By assumption \(\text{sum}(\rho) \oplus R_e \oplus R_f \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \subseteq (C, \Phi, \alpha)\). We abbreviate the resources that remain unchanged \(\text{R}_F := (\bigoplus_{\ell' \in \text{dom } \rho, \ell' \neq \ell} \rho \ell') \oplus R_f \ominus (\bigoplus_{\ell' \in \text{dom } \sigma, \ell' \neq \ell} R_{\text{cap}(\text{idx} \cdot \sigma'))\).

Then \(
R_F \oplus \rho \ell \oplus R_e \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \subseteq (C, \Phi, \alpha) \text{ by Lemma 4.2}\).

\[
R_F \oplus \rho \ell \oplus R_e \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \subseteq (C, \Phi, \alpha) \\
\Rightarrow R_F \oplus \rho \ell \oplus R_e \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \oplus R_{\text{inv}(\Phi)} \subseteq (C, \Phi, \alpha) \oplus R_{\text{inv}(\Phi)} \text{ (Lemma 3.2)} \\
\Rightarrow R_F \oplus \rho \ell \oplus R_e \oplus R_{\text{cap}(\text{idx} \cdot \sigma)} \oplus R_{\text{inv}(\Phi)} \subseteq (C, \Phi, \alpha) \text{ (Lemma 3.3)} \\
\Rightarrow R_F \oplus R \oplus R'_{e, R_{\text{cap}(\text{idx} \cdot s)} \oplus R_{\text{cap}(D)} \subseteq (C, \Phi, \alpha) \oplus R_{\text{cap}(D)} \text{ (Lemma 3.2)} \\
\Rightarrow R_F \oplus R \oplus R'_{e, R_{\text{cap}(\text{idx} \cdot s)} \oplus R_{\text{cap}(D)} \subseteq (C \setminus D, \Phi, \alpha) \oplus R_{\text{cap}(D)} \text{ (Lemma 3.3)} \\
\Rightarrow \text{sum}(\rho[\ell \mapsto R]) \oplus R'_{e} \oplus R_f \oplus R_{\text{cap}(\text{idx} \cdot \sigma[\ell \mapsto s])} \subseteq (C \setminus D, \Phi, \alpha) \text{ (Lemma 4.2)}
\]

where \((\ast)\) follows with \(D \subseteq C\) which holds given \(R_F \oplus R \oplus R'_{e} \oplus R_{\text{cap}(\text{idx} \cdot s)} \oplus R_{\text{cap}(D)} \subseteq (C, \Phi, \alpha)\) by Lemma 3.2.

By assumption, we have \(C \# \) used \(\sigma\). We show \((C \setminus D) \# \) used \(\sigma[\ell \mapsto s]\). By way of contradiction, assume \(p \in C \setminus D\) and \(p \in \text{used } (\sigma[\ell \mapsto s])\). Since \(C \# \) used \(\sigma\), we have \(p \notin \text{used } \sigma\). Thus \(p \notin \text{used } (\ell, \sigma)\) and \(p \in \text{used } (\ell, s)\). Hence \(p \in \text{used } (\ell, s) \setminus \text{used } (\ell, \sigma) \subseteq D\), a contradiction.

By assumption we have \((h, \sigma, \rho) \in H[\Phi]_{\alpha, C}\) and \((h, s, R) \in H[\Phi]_{\ell : [A]}_{\alpha, C}\). Thus, by definition we have \((h[\ell \mapsto h], \sigma[\ell \mapsto s], \rho[\ell \mapsto R]) \in H[\Phi, \ell : A]_{\alpha, C}\). With \text{Lemma 4.3}, it suffices to show \(\text{idx}(\sigma[\ell \mapsto s]) \subseteq C \setminus D\) which follows from \(\text{sum}(\rho[\ell \mapsto R]) \oplus R'_{e} \oplus R_f \oplus R_{\text{cap}(\text{idx} \cdot \sigma[\ell \mapsto s])} \subseteq (C \setminus D, \Phi, \alpha)\) by \text{Lemma 3.2}.\qed
Once a location has been allocated, the following lemma, Lemma 4.11, characterises how we can update the resource interpretation. From the state Start, the resource interpretation may be updated to the Val state by giving up the resources for the value \( v \) and the resource \( R_{\text{put}(\ell,A)} \) which contains the capability \( \text{put}(\ell) \). The resource \( R_v \) associated with the value is transferred to the heap while the capability \( \text{put}(\ell) \) is transferred to the logical state and remains part of the step-index. Analogously, we can store a continuation in the heap and advance the logical state to Cont. If the heap currently stores a value with logical state Val, we can advance the state to Done and remove the resource \( R_v \) from the resource map. We decrease the current step-index \((\alpha,C)\) to the smaller step-index \((\alpha,C\\setminus \{\text{get}(\ell),\text{put}(\ell)\})\). Analogously, we can advance from the state where a continuation is stored to the Done state.

**Lemma 4.11.** Let \( \mathcal{R}_{\alpha,C}(R_e, R_f, \Phi, h, \sigma, \rho) \).

1. If \( \sigma \ell = \text{Start} \) and \( \rho \ell = \epsilon \) and \( R_e = R_v \oplus R_{\text{put}(\ell,A)} \) and \( (v, R_v) \in \mathcal{V}[A]_{\alpha,C} \), then
   \[
   \mathcal{R}_{\alpha,C}(\epsilon, R_f, \Phi, h[\ell \mapsto \mathcal{V}(v)], \sigma[\ell \mapsto \text{Val}], \rho[\ell \mapsto R_v]).
   \]

2. If \( \sigma \ell = \text{Start} \) and \( \rho \ell = \epsilon \) and \( R_e = R_\lambda \oplus R_{\text{get}(\ell,A)} \) and \( (v, R_\lambda) \in \mathcal{V}[A \rightarrow 1]_{\alpha,C} \), then
   \[
   \mathcal{R}_{\alpha,C}(\epsilon, R_f, \Phi, h[\ell \mapsto \text{C}(v)], \sigma[\ell \mapsto \text{Cont}], \rho[\ell \mapsto R_\lambda]).
   \]

3. If \( \sigma \ell = \text{Val} \) and \( \rho \ell = R_v \) and \( R_e = R_\lambda \oplus R_{\text{get}(\ell,A)} \), then
   \[
   \mathcal{R}_{\alpha,C}(\text{\{get}(\ell),\text{put}(\ell)\})(R_v \oplus R_\lambda, R_f, \Phi, h[\ell \mapsto \text{E}], \sigma[\ell \mapsto \text{Done}], \rho[\ell \mapsto \epsilon]).
   \]

4. If \( \sigma \ell = \text{Cont} \) and \( \rho \ell = R_\lambda \) and \( R_e = R_v \oplus R_{\text{put}(\ell,A)} \), then
   \[
   \mathcal{R}_{\alpha,C}(\text{\{get}(\ell),\text{put}(\ell)\})(R_v \oplus R_\lambda, R_f, \Phi, h[\ell \mapsto \text{E}], \sigma[\ell \mapsto \text{Done}], \rho[\ell \mapsto \epsilon]).
   \]

**Proof.** We prove each of the claims using Lemma 4.10. For each case, we have \( \ell : A \in \Phi \) because \( R_{\text{get}(\ell,A)} \subseteq R_e \) or \( R_{\text{put}(\ell,A)} \subseteq R_e \) from which \( \ell : A \in \Phi \) follows by Lemma 3.2 since \( R_e \subseteq \sum(\rho) \oplus R_v \oplus R_\lambda \oplus R_{\text{cap}(\text{idx},\sigma)} \subseteq (C, \Phi, \alpha) \).

1. We pick \( D \triangleq \emptyset = \text{used}(\ell, \text{Val}) \setminus \text{used}(\ell, \text{Start}) \). We show \( (\mathcal{V}(v), \text{Val}, R_v) \in \mathcal{V}[\mathcal{A}]_{\alpha,C} \). We have \( \text{put}(\ell) \in \mathcal{C} \) as \( R_{\text{put}(\ell,A)} \subseteq R_e \subseteq (C, \Phi, \alpha) \) by Lemma 3.2. Thus, there is some \( C' \) such that \( C = C' \uplus \{\text{put}(\ell)\} \). By Lemma 4.3, we have \((v, R_v) \in \mathcal{V}[A]_{\alpha,C'}\).

   Lastly, we have by Lemma 3.2 and Lemma 3.3:
   
   \[
   \rho \ell \oplus R_v \oplus R_{\text{cap}(\text{idx}(\ell,\sigma),\ell)} \oplus R_{\text{inv}(\Phi)}
   \]
   
   \[
   = \epsilon \oplus R_v \oplus R_{\text{put}(\ell,A)} \oplus \epsilon \oplus R_{\text{inv}(\Phi)} \quad \text{(Def. idx)}
   \]
   
   \[
   = \epsilon \oplus R_v \oplus R_{\text{cap}(\text{put}(\ell))} \oplus R_{\text{inv}(\ell,A)} \oplus \epsilon \oplus R_{\text{inv}(\Phi)}
   \]
   
   \[
   = R_v \oplus \epsilon \oplus R_{\text{cap}(\text{put}(\ell))} \oplus R_{\text{cap}(D)} \oplus R_{\text{inv}(\Phi)} \quad (\ell : A \in \Phi, R_{\text{cap}(\emptyset)} = \epsilon)
   \]
   
   \[
   = R_v \oplus \epsilon \oplus R_{\text{cap}(\text{idx}(\ell,\text{Val}))} \oplus R_{\text{cap}(D)} \oplus R_{\text{inv}(\Phi)} \quad \text{(Def. idx)}
   \]

2. We pick \( D \triangleq \emptyset = \text{used}(\ell, \text{Cont}) \setminus \text{used}(\ell, \text{Start}) \). We show \( (\mathcal{C}(v), \text{Cont}, R_\lambda) \in \mathcal{V}[\mathcal{A}]_{\alpha,C} \). We have \( \ell \in \mathcal{C} \) as \( R_{\text{get}(\ell,A)} \subseteq R_e \subseteq (C, \Phi, \alpha) \) by Lemma 3.2. Thus, there is some \( C' \) with \( C = C' \uplus \{\text{get}(\ell)\} \). By Lemma 4.3, we have \((v, R_\lambda) \in \mathcal{V}[A \rightarrow 1]_{\alpha,C'}\).
Lastly, we have by Lemma 3.2 and Lemma 3.3:

\[
\rho \ell + R_e + R_{\text{cap}(\text{idx}(\ell, \sigma \ell))} + R_{\text{inv}(\Phi)}
\]

\[
= \epsilon + R_{\lambda} + R_{\text{get}(\ell, A)} + \epsilon + R_{\text{inv}(\Phi)}
\]

\[
= \epsilon + R_{\lambda} + R_{\text{cap}(\text{get}(\ell))} + R_{\text{inv}(\ell, A)} + \epsilon + R_{\text{inv}(\Phi)}
\]

\[
= R_{\lambda} + \epsilon + R_{\text{cap}(\text{get}(\ell))} + R_{\text{cap}(D)} + R_{\text{inv}(\Phi)}
\]

(Def. idx)

\[
= R_{\lambda} + \epsilon + R_{\text{cap}(\text{idx}(\ell, \text{Cont}))} + R_{\text{cap}(D)} + R_{\text{inv}(\Phi)}
\]

(Def. idx)

3. We pick \( D \triangleq \{ \text{get}(\ell), \text{put}(\ell) \} = \text{used}(\ell, \text{Done}) \setminus \text{used}(\ell, \text{Val}) \). By definition, we have \((E, \text{Done}, \epsilon) \in \mathcal{H}_{\nu}^n[\ell : A]_{\alpha, C} \). Lastly, we have by Lemma 3.2 and Lemma 3.3:

\[
\rho \ell + R_e + R_{\text{cap}(\text{idx}(\ell, \sigma \ell))} + R_{\text{inv}(\Phi)}
\]

\[
= R_v + R_{\lambda} + R_{\text{cap}(\text{get}(\ell), \text{put}(\ell))} + R_{\text{inv}(\Phi)}
\]

\[
= \epsilon + (R_v + R_{\lambda}) + R_{\text{cap}(\text{get}(\ell), \text{put}(\ell))} + R_{\text{inv}(\ell, A)} + R_{\text{inv}(\Phi)}
\]

\[
= \epsilon + (R_v + R_{\lambda}) + R_{\text{cap}(D)} + R_{\text{inv}(\Phi)}
\]

(\( \ell : A \in \Phi \))

\[
= \epsilon + (R_v + R_{\lambda}) + R_{\text{cap}(\text{idx}(\ell, \text{Done}))} + R_{\text{cap}(D)} + R_{\text{inv}(\Phi)}
\]

(Def. idx)

4. We pick \( D \triangleq \{ \text{get}(\ell), \text{put}(\ell) \} = \text{used}(\ell, \text{Done}) \setminus \text{used}(\ell, \text{Cont}) \). By definition, we have \((E, \text{Done}, \epsilon) \in \mathcal{H}_{\nu}^n[\ell : A]_{\alpha, C} \). Lastly, we have by Lemma 3.2 and Lemma 3.3:

\[
\rho \ell + R_e + R_{\text{cap}(\text{idx}(\ell, \sigma \ell))} + R_{\text{inv}(\Phi)}
\]

\[
= R_v + R_{\lambda} + R_{\text{cap}(\text{get}(\ell))} + R_{\text{inv}(\Phi)}
\]

(Def. idx)

\[
= \epsilon + (R_v + R_{\lambda}) + R_{\text{cap}(\text{get}(\ell))} + R_{\text{inv}(\ell, A)} + R_{\text{inv}(\Phi)}
\]

(\( \ell : A \in \Phi \))

\[
= \epsilon + (R_v + R_{\lambda}) + R_{\text{cap}(\text{idx}(\ell, \text{Done}))} + R_{\text{cap}(D)} + R_{\text{inv}(\Phi)}
\]

(Def. idx)

\[\square\]

4.3 Properties of the Type Interpretations

The expression relation is closed under larger resources, in particular in the number of locations that can be allocated. We can always weaken the number to a larger bound:

**Lemma 4.12.** 1. If \((e, R_e + R_{\text{ord}(\alpha)}) \in \mathcal{E}[A]_i \) and \( \alpha \leq \beta \), then \((e, R_e + R_{\text{ord}(\beta)}) \in \mathcal{E}[A]_i \).

2. If \((e, R_e) \in \mathcal{E}[A]_i \), then \((e, R_e + R) \in \mathcal{E}[A]_i \).

**Proof.** 1. Let \((\gamma, C) \leq i \) and \( R_{\gamma, C}(R_e + R_{\text{ord}(\beta)}, R_f, \Phi, h, \sigma, \rho) \). Then there is some \( \gamma' \leq \gamma \) such that \( R_{\gamma', C}(R_e + R_{\text{ord}(\alpha)}, R_f, \Phi, h, \sigma, \rho) \) by Lemma 4.6. Since \( \gamma' \leq \gamma \), we have \((\gamma', C) \leq (\gamma, C) \leq i \). By assumption, there is some \( k \leq (\gamma', C) \) and \( \Phi' \supseteq \Phi \) and \( h', \sigma', \rho' \) and \((v, R_v) \in \mathcal{V}[A]_k \) such that \( R_{\gamma', C}(R_v, R_f, \Phi', h', \sigma', \rho') \) and \( \sigma \rightarrow^* \sigma' \) and \((e, h) \leadsto^*(v, h') \). The claim follows with \( k \leq (\gamma', C) \leq (\gamma, C) \).
Lemma 4.14. Let $j \leq i$ and $\text{Rl}_j(R_e \oplus R_f, \Phi, h, \sigma, \rho)$. Then $\text{Rl}_j(R_e, R_f, \Phi, h, \sigma, \rho)$ by Lemma 4.7. The claim follows by the assumption $(e, R_e) \in E[A]_i$.

\[ \square \]

The following lemma is useful in proofs of standard compatibility lemmas. First, every value in the value relation is already contained in the expression relation. Second, we can take steps which do not manipulate the heap, defined by $e' \leadsto e \triangleq \forall h. (e, h) \leadsto (e', h)$, to prove that an expression is contained in the logical relation. Third, we can reason about composite expressions $K[e]$ by reasoning about $e$ and the remaining expression $K[v]$ after $e$ has been executed, if we abstract over the result $v$.

**Lemma 4.13.** 1. $\forall[A]_i \subseteq E[A]_i$.
2. If $(e, R_e) \in E[A]_i$ and $e' \leadsto e$, then $(e', R_e) \in E[A]_i$.
3. If $(e, R_e) \in E[A]_i$ and $\forall j \leq i, (v, R_v) \in \forall[A]_j$, $(K[v], R_v \oplus R_K) \in E[B]_j$, then $(K[e], R_e \oplus R_K) \in E[B]_i$.

**Proof.** 1. Let $(v, R_v) \in \forall[A]_i$ and $j \leq i$ and $\text{Rl}_j(R_e, R_f, \Phi, h, \sigma, \rho)$. Then $(v, R_v) \in \forall[A]_j$ by Lemma 4.3. The claim follows with $(v, h) \leadsto^* (v, h)$ and $\sigma \leadsto^* \sigma$.
2. Let $(e, R_e) \in E[A]_i$ and $e' \leadsto e$ and $j \leq i$ and $\text{Rl}_j(R_e, R_f, \Phi, h, \sigma, \rho)$. By assumption, there exist some $j' \leq j$ and $\Phi' \supseteq \Phi$ and $h', \sigma', \rho', v, R_v$ such that $\text{Rl}_j(R_v, R_f, \Phi', h', \sigma', \rho')$ and $(v, R_v) \in \forall[A]_j$ and $(e, h) \leadsto^* (v, h')$. The claim follows with $(e', h) \leadsto^* (e, h) \leadsto^* (v, h')$.
3. Let $j \leq i$ and $\text{Rl}_j(R_e \oplus R_K, R_f, \Phi, h, \sigma, \rho)$. By Lemma 4.5, we have $\text{Rl}_j(R_e, R_K \oplus R_f, \Phi, h, \sigma, \rho)$. By assumption, there exist some $j' \leq j$ and $\Phi' \supseteq \Phi$ and $h', \sigma', \rho', v, R_v$ such that $\text{Rl}_j(R_v, R_K \oplus R_f, \Phi', h', \sigma', \rho')$ and $\sigma \leadsto^* \sigma'$ and $(v, R_v) \in \forall[A]_{j'}$. By Lemma 4.5, $\text{Rl}_{j'}(R_e, R_K, R_f, \Phi', h', \sigma', \rho')$ follows. By assumption, $(K[v], R_v \oplus R_K) \in E[B]_j$. Thus, there exists some $k \leq j'$ and $\Phi'' \supseteq \Phi'$ and $h'', \sigma'', \rho'', v', R_v'$ such that $\text{Rl}_k(R_v', R_f, \Phi'', h'', \sigma'', \rho'')$ and $\sigma \leadsto^* \sigma''$ and $(v, R_v') \in \forall[B]_{k'}$ and $(K[v], h') \leadsto^* (v', h'')$. We have $(K[e], h) \leadsto^* (K[v], h') \leadsto^* (v', h'')$ and $\sigma \leadsto^* \sigma' \leadsto^* \sigma''$ and $k \leq j' \leq j$.

\[ \square \]

The following four lemmas are the compatibility lemmas on closed expressions for creating channels, for receiving values over them, for sending values over them, and for closing them.

**Lemma 4.14.**

\[ \forall \ell. (e[\ell/x, \ell/y], R_e \oplus R_{\text{get}(\ell,A)} \oplus R_{\text{put}(\ell,A)}) \in E[B]_i \]

**Proof.** Let $(\alpha, C) \leq i$ and $\text{Rl}_{\alpha,C}(R_e \oplus R_{\text{ord}(1)}, R_f, \Phi, h, \sigma, \rho)$. By Lemma 4.8, we have $\text{Rl}_{\beta,C \cup \{\text{al}(\ell), \text{put}(\ell), \text{get}(\ell)\}}(R_e \oplus R_{\text{cap}(\text{al}(\ell), \text{get}(\ell), \text{put}(\ell))}, R_f, \Phi, h, \sigma, \rho)$.

21
for some $\beta$ with $\alpha = \beta + 1$ and some location $\ell$. We have $(C \cup \{a_0(\ell), \text{put}(\ell), \text{get}(\ell)\}) \neq \emptyset$ used $\sigma$ by the definition of $R$ and thus $\ell \notin \text{dom} \sigma$ by Lemma 4.2. With $(h, \sigma, \rho) \in \mathcal{H}[\Phi]_{\alpha,C}$, we further have $\ell \notin \text{dom} h$ and $\ell \notin \text{dom} \Phi$ and $\ell \notin \text{dom} \rho$. By Lemma 4.9, we have:

$$R_{\beta,C \cup \{\text{get}(\ell), \text{put}(\ell)\}}(R_c \oplus R_{\text{cap}(\text{get}(\ell), \text{put}(\ell))} \oplus R_{\text{inv}(\ell, A)}) \cdot R_f, (\Phi, \ell : A), h', \sigma', \rho')$$

for $h' \triangleq h[\ell \mapsto E]$ and $\sigma' \triangleq \sigma[\ell \mapsto \text{Start}]$ and $\rho' \triangleq \rho[\ell \mapsto \epsilon]$. By assumption $(e[\ell/x, \ell/y], R_c \oplus R_{\text{get}(\ell, A)} \oplus R_{\text{put}(\ell, A)}) \in \mathcal{E}[B]_i$. Since $\beta < \alpha$ by Lemma 2.1, we have $(\beta, C \cup \{\text{get}(\ell), \text{put}(\ell)\}) \prec (\alpha, C) \leq i$. Hence, there is some $(\beta', C') \leq (\beta, C \cup \{\text{get}(\ell), \text{put}(\ell)\})$ and $\Phi' \supseteq (\Phi, \ell : A)$ with

$$R_{\beta', C'}(R_v, R_f, h', \sigma'', \rho'')$$

for some $h'', \sigma'', \rho''$, $(v, R_v) \in \mathcal{V}[B]_{\beta', C'}$. Further, $\sigma' \rightsquigarrow \sigma''$ and $(e[\ell/x, \ell/y], h') \rightsquigarrow (v, h'').$ We have $(\beta', C') \leq (\beta, C \cup \{\text{get}(\ell), \text{put}(\ell)\}) \leq (\alpha, C)$ and $\Phi' \supseteq (\Phi, \ell : A) \supseteq \Phi$. Since $\ell \notin \text{dom} \sigma$ and $\ell \notin \text{dom} h$, we have $\sigma \rightarrow \sigma' \rightarrow \sigma''$ and $(x, y) = \text{chan}(\epsilon, e, h) \rightsquigarrow (e[\ell/x, \ell/y], h') \rightsquigarrow (v, h'').$

$\square$

Lemma 4.15.

$$\frac{(v_{\text{ch}}, R_{\text{ch}}) \in \mathcal{V}[\text{Get} A]_i, (v_{\lambda}, R_{\lambda}) \in \mathcal{V}[A \rightarrow 1]_i}{(\text{get}(v_{\text{ch}}, v_{\lambda}), R_{\text{ch}} \oplus R_{\lambda}) \in \mathcal{E}[1]_i}$$

Proof. Let $(\alpha, C) \leq i$ and $R_{\alpha,C}(R_v \oplus R_f, H, \sigma, \rho)$. By $(v_{\text{ch}}, R_{\text{ch}}) \in \mathcal{V}[\text{Get} A]_i$, we know $v_{\text{ch}} = \ell$ and $R_v = R_{\text{ch}} = R_{\text{get}(\ell, A)}$ for some $\ell$. By Lemma 3.2, we have $R_{\text{get}(\ell, A)} \subseteq (C, H, \sigma, \rho)$ and $\sqrt{R_{\text{get}(\ell, A)} \oplus R_{\text{cap}(\ell, \sigma)}}$ and thus get(\ell) $\subseteq C$ and get(\ell) $\notin \text{idx} \sigma$ and $\ell : A \in \Phi$. Further, we have used $\sigma \\neq C$. Thus, get(\ell) $\notin \text{used} \sigma$ follows. Further, unfolding the definition of $R_{\alpha,C}$, we have $(h\ell, \sigma\ell, \rho\ell) \in \mathcal{H}[\ell : A]_{\alpha,C}$. Since get(\ell) $\notin \text{used} \sigma$ and get(\ell) $\notin \text{idx} \sigma$, either $\sigma\ell = \text{Start}$ or $\sigma\ell = \text{Val}$.

1. Let $\sigma\ell = \text{Start}$. By $(h\ell, \sigma\ell, \rho\ell) \in \mathcal{H}[\ell : A]_{\alpha,C}$, we have $h\ell = E$ and $\rho\ell = \epsilon$. By Lemma 4.3, we have $(v_{\lambda}, R_{\lambda}) \in \mathcal{V}[A \rightarrow 1]_{\alpha,C}$. Hence, with Lemma 4.11 we have:

$$R_{\alpha,C}(\epsilon, R_f, H, h[\ell \mapsto C(v_{\lambda})], \sigma[\ell \mapsto \text{Cont}], \rho[\ell \mapsto R_{\lambda}])$$

We define $h' \triangleq h[\ell \mapsto C(v_{\lambda})], \sigma' \triangleq \sigma[\ell \mapsto \text{Cont}],$ and $\rho' \triangleq \rho[\ell \mapsto R_{\lambda}]$. By definition $(\text{get}(\ell, v_{\lambda}), h) \rightsquigarrow (v, h')$ and $\sigma \rightarrow \sigma'.$ Further, by definition $(\epsilon, e) \in \mathcal{V}[1]_{\alpha,C}.$

2. Let $\sigma\ell = \text{Val}$. By $(h\ell, \sigma\ell, \rho\ell) \in \mathcal{H}[\ell : A]_{\alpha,C}$, we have $C = C_1 \cup \{\text{put}(\ell)\}$ for some $C_1$ and $h\ell = V(v)$ and $\rho\ell = R_v$ for some $(v, R_v) \in \mathcal{V}[A]_{\alpha,C_1}$. Since get(\ell) $\subseteq C$, there is some $C_2$ such that $C_2 = C_1 \cup \{\text{get}(\ell)\}$. By Lemma 4.3 $(v, R_v) \in \mathcal{V}[A]_{\alpha,C_2}$. Thus, by definition of the value relation $\mathcal{V}[A \rightarrow 1]_i$, we have $v_{\lambda} = \lambda.x.e$ for some $x$ and $e$ such that $(e[v/x], R_v \oplus R_{\lambda}) \in \mathcal{E}[1]_{\alpha,C_2}$. By Lemma 4.11, we obtain:

$$R_{\alpha,C_2}(R_v \oplus R_{\lambda}, R_f, H, h[\ell \mapsto E], \sigma[\ell \mapsto \text{Done}], \rho[\ell \mapsto \epsilon])$$

Thus, there are some $(\beta, C_3) \leq (\alpha, C_2)$ and $\Phi' \geq \Phi, h', \sigma', \rho'$ and $(v', R_{\epsilon}') \in \mathcal{V}[1]_{\beta,C_3}$, such that $R_{\beta,C_3}(R_v', R_f, H', h', \sigma', \rho')$ and $(e[v/x], h[\ell \mapsto E]) \rightsquigarrow (v', h')$ and $\sigma[\ell \mapsto \text{Done}] \rightarrow \sigma'.$
We have \((\beta, C_3) \leq (\alpha, C_2) \leq (\alpha, C)\) and \(\Phi' \supseteq \Phi\). Further, we have \((\text{get}(\ell, \lambda x. e), h) \leadsto (e[v/x], h[\ell \mapsto E]) \leadsto^* (v', h')\). Lastly, \(\sigma \leadsto \sigma[\ell \mapsto \text{Done}] \leadsto^* \sigma'\).

\[\square\]

**Lemma 4.16.**

\[
\begin{align*}
(v_{ch}, R_{ch}) &\in \mathcal{V}[\text{Put } A], \\
(v, R_v) &\in \mathcal{V}[A]
\end{align*}
\]

\[
(\text{put}(v_{ch}, v), R_{ch} \oplus R_v) \in \mathcal{E}[\mathbb{I}]
\]

**Proof.** Let \((\alpha, C) \leq i\) and \(\text{RI}_{\alpha,C}(R_{ch} \oplus R_v, R_f, \Phi, h, \sigma, \rho)\). By \((v_{ch}, R_{ch}) \in \mathcal{V}[\text{Put } A]\), we know \(v_{ch} = \ell\) and \(R_{ch} = R_{\text{put}(\ell, A)}\) for some \(\ell\). By **Lemma 3.2**, we have \(R_{\text{put}(\ell, A)} \subseteq (C, \Phi, \alpha)\) and \(\check{v} R_{\text{put}(\ell, A)} \oplus R_{\text{cap}(idx, \sigma)}\) and thus \(\text{put}(\ell) \in C\) and \(\text{put}(\ell) \not\in \text{idx} \sigma\) and \(\ell : A \in \Phi\). Further, we have used \(\sigma \neq C\). Thus, \(\text{put}(\ell) \not\in \text{used} \sigma\) follows. Further, unfolding the definition of \(\text{RI}_{\alpha,C}\), we have \((h\ell, \sigma, \rho, \ell) \in \mathcal{H}[\mathcal{V}[A] : \alpha, C\). Since \(\text{put}(\ell) \not\in \text{used} \sigma\) and \(\text{put}(\ell) \not\in \text{idx} \sigma\), either \(\sigma = \text{Start} \) or \(\sigma = \text{Cont}\).

1. Let \(\sigma \ell = \text{Start}\). By \((h\ell, \sigma, \rho, \ell) \in \mathcal{H}[\mathcal{V}[A] : \alpha, C\), we have \(h\ell = E\) and \(\rho \ell = \epsilon\).

By **Lemma 4.3**, we have \((v, R_v) \in \mathcal{V}[A]_{\alpha,C}\). Hence, with **Lemma 4.11** we have:

\[
\text{RI}_{\beta,C}(\epsilon, R_f, \Phi, h[\ell \mapsto \text{Val}], \sigma[\ell \mapsto \text{Val}], \rho[\ell \mapsto R_v])
\]

We define \(h' \triangleq h[\ell \mapsto V(\ell)], \sigma' \triangleq \sigma[\ell \mapsto \text{Val}],\) and \(\rho' \triangleq \rho[\ell \mapsto R_v]\). By definition \((\text{put}(\ell, v), h) \leadsto^* (v', h')\) and \(\sigma \leadsto^* \sigma'\). Further, by definition \(((), \epsilon) \in \mathcal{V}[\mathbb{I}]_{\alpha,C}\).

2. Let \(\sigma \ell = \text{Cont}\). By \((h\ell, \sigma, \rho, \ell) \in \mathcal{H}[\mathcal{V}[A] : \alpha, C\), we have \(C = C_1 \cup \{\text{get}(\ell)\}\) for some \(C_1\) and \(h\ell = C(v_\lambda)\) and \(\rho \ell = R_3\) for some \((v_\lambda, R_3) \in \mathcal{V}[A \rightarrow \mathbb{I}]_{\alpha,C_1}\). Since \(\text{put}(\ell) \in C\), there is some \(C_2\) such that \(C_2 = C_1 \cup \{\text{put}(\ell)\}\). By **Lemma 4.3**, we have \((v, R_v) \in \mathcal{V}[A]_{\alpha,C_2}\). Thus, by definition of the value relation \(\mathcal{V}[A \rightarrow \mathbb{I}]_{\alpha,C_1}\) we have \(v_\lambda = \lambda x. e\) for some \(x\) and \(e\) such that \((e[v/x], R_v \oplus R_3) \in \mathcal{E}[\mathbb{I}]_{\alpha,C_2}\). By **Lemma 4.11**, we obtain:

\[
\text{RI}_{\alpha,C_2}(R_v \oplus R_3, R_f, \Phi, h[\ell \mapsto E], \sigma[\ell \mapsto \text{Done}], \rho[\ell \mapsto \epsilon])
\]

Thus, there are some \((\beta, C_3) \leq (\alpha, C_2)\) and \(\Phi' \supseteq \Phi, h', \sigma', \rho' \) and \((v', \rho_\prime) \in \mathcal{V}[\mathbb{I}]_{\beta,C_3}\), such that \(\text{RI}_{\beta,C_3}(R_v', R_f, \Phi', h', \sigma', \rho')\) and \((e[n/x], h[\ell \mapsto E]) \leadsto^* (v', h')\) and \(\sigma[\ell \mapsto \text{Done}] \leadsto^* \sigma'\).

We have \((\beta, C_3) \leq (\alpha, C_2) \leq (\alpha, C)\) and \(\Phi' \supseteq \Phi\). Further, we have \((\text{put}(\ell, v), h) \leadsto (e[v/x], h[\ell \mapsto E]) \leadsto^* (v', h')\). Lastly, \(\sigma \leadsto \sigma[\ell \mapsto \text{Done}] \leadsto^* \sigma'\).

\[\square\]

For the remaining expressions, properties about their behavior follow from the lemmas above.

We showcase iteration. For iteration \(\text{iter}(n, e', x.e)\), the function that is iterated \(\lambda x. e\) cannot make use of capabilities of existing channels. Otherwise, the capabilities would have to be duplicated for repeated iteration. The function may however allocate fresh channels. If \(\alpha\) is an upper bound on the number of locations that are allocated by \(\lambda x. e\), regardless of the argument,
then $n \otimes \alpha$ is an upper bound on the number of locations that are allocated when iterating the function $n$-times. If the number of iterations is not known before the execution, we can bound it by $\omega \otimes \alpha$.

**Lemma 4.17.**

$$
\frac{(e', R'_v) \in E[A]_i}{\text{(iter}(n, e', x.e), R'_v \oplus R_{\text{ord}(n \otimes \alpha)}) \in E[A]_i}
$$

**Proof.** By induction on $n$ for arbitrary $i$, $e'$, and $R'_v$. With Lemma 4.13, it suffices to show $(\text{iter}(n, v, x.e), R_v \oplus R_{\text{ord}(n \otimes \alpha)}) \in E[A]_i$ for all $j \leq i$ and $(v, R_v) \in V[A]_j$.

1. **Case** $n = 0$. As $R_{\text{ord}(0 \otimes \alpha)} = \epsilon$ and $\text{iter}(0, v, x.e) \rightsquigarrow v$, the claim follows with Lemma 4.13 and $(v, R_v) \in V[A]_0$.

2. **Case** $n > 0$. By definition $\text{iter}(n, v, x.e) \rightsquigarrow \text{iter}(n-1, e[v/x], x.e)$. With Lemma 4.13 and Lemma 2.2, it suffices to show $(\text{iter}(n-1, e[v/x], x.e), R_v \oplus R_{\text{ord}(n-1 \otimes \alpha)}) \in E[A]_j$. Since $(v, R_v) \in V[A]_j$ and $(\lambda x.e, R_{\text{ord}(\alpha)}) \in V[A \rightarrow A]_j$ and $j \leq i$, we have $(e[v/x], R_v \oplus R_{\text{ord}(\alpha)}) \in E[A]_j$. The claim follows by induction and Lemma 4.3.

**Lemma 4.18.**

$$
\frac{(e, R) \in E[N]_i, (e_0, R_0) \in E[A]_i, (\lambda x.e_S, R_{\text{ord}(\alpha_S)}) \in V[A \rightarrow A]_i}{\text{(iter}(e, e_0, e.e_S), R \oplus R_0 \oplus R_{\text{ord}(\alpha \otimes \alpha_S)}) \in E[A]_i}
$$

**Proof.** By Lemma 4.13, it suffices to show:

$$
\text{iter}(v, e_0, e.e_S), R_v \oplus R_0 \oplus R_{\text{ord}(\alpha \otimes \alpha_S)}) \in E[A]_j
$$

for all $j \leq i$ and $(v, R_v) \in V[N]_j$. By definition of $V[N]_j$, we have $v = n$ for some $n \in \mathbb{N}$ and $R_v = \epsilon$. Since $n \otimes \alpha_S \leq \omega \otimes \alpha_S$ by Lemma 2.2, by Lemma 4.12 it suffices to show:

$$
\text{iter}(n, e_0, e.e_S), R_0 \oplus R_{\text{ord}(n \otimes \alpha_S)}) \in E[A]_j
$$

The claim follows with Lemma 4.3 and Lemma 4.17.

**4.4 Semantic Typing**

To define the semantic typing judgement $\Gamma \vdash e : A$, we close the expression with a substitution from the context interpretation $G[\Gamma]_i$:

$$
G[\Gamma]_i \triangleq \{(\theta, e)\}
$$

$$
G[\Gamma, x : A]_i \triangleq \{(\theta, R_\theta \oplus R) \mid \theta, R_\theta \in G[\Gamma]_i \text{ and } (\theta x, R) \in V[A]_i\}
$$

$$
\Gamma \vdash e : A \triangleq \exists \alpha. \forall i. \forall (\theta, R_\theta) \in G[\Gamma]_i, (e[\theta], R_\theta \oplus R_{\text{ord}(\alpha)}) \in E[A]_i
$$
**Context Interpretation** Before we proceed to prove semantic soundness, we establish the following properties of the context interpretation $G[e]_i$.

**Lemma 4.19.** If $(\theta, R_\theta) \in G[\Gamma, \Delta]_i$, then $R_\theta = R_1 \oplus R_2$ for some resources $R_1, R_2$ such that $(\theta, R_1) \in G[\Gamma]_i$ and $(\theta, R_2) \in G[\Delta]_i$.

**Proof.** By induction on $\Delta$.

1. Case $\cdot$. Then $(\theta, R_\theta) \in G[\Gamma]_i$. The claim follows with $R_1 \triangleq R_\theta$ and $R_2 \triangleq \epsilon$.

2. Case $\Delta, x : A$. Then $(\theta, R_\theta) \in G[\Gamma, \Delta, x : A]_i$. Thus, $R_\theta = R_\theta' \oplus R$ for some $R_\theta', R$ with $(\theta x, R) \in \mathcal{V}[A]_i$ and $(\theta, R_\theta') \in G[\Gamma, \Delta]_i$. By induction, there are $R_1, R_2$ such that $R_\theta' = R_1 \oplus R_2'$ and $(\theta, R_1) \in G[\Gamma]_i$ and $(\theta, R_2') \in G[\Delta]_i$. We define $R_2 \triangleq R_2' \oplus R$. By definition, $(\theta, R_2) \in G[\Delta, x : A]_i$.

$\square$

**Lemma 4.20.**

1. If $\theta =_{\text{dom } \Gamma} \theta'$ and $(\theta, R) \in G[\Gamma]_i$, then $(\theta', R) \in G[\Gamma]_i$.

2. If $i \leq j$, then $G[\Gamma]_j \subseteq G[\Gamma]_i$.

**Proof.** Both by induction on $\Gamma$. For the second, we use Lemma 4.3.$\square$

**Lemma 4.21.** If $\Gamma, x : A$ is defined and $(\theta, R_\theta) \in G[\Gamma]_i$ and $(v, R_v) \in \mathcal{V}[A]_i$, then $(\theta[x \mapsto v], R_\theta \oplus R_v) \in G[\Gamma, x : A]_i$.

**Proof.** By definition of $G[\Gamma, x : A]_i$, it suffices to show $(\theta[x \mapsto v], R_\theta) \in G[\Gamma]_i$. As type contexts are linear, we know $x \notin \text{dom } \Gamma$. Thus $\theta[x \mapsto v] =_{\text{dom } \Gamma} \theta$. The claim follows with Lemma 4.20.$\square$

**Semantic Soundness** In the following, we prove semantic soundness of the type system $\Gamma \vdash e : A$ with respect to the semantic interpretation $\Gamma \vDash e : A$. The proof proceeds by induction using compatibility lemmas in each case. We prove those compatibility lemmas in Section 4.5. Given the semantic soundness proof, it is then straightforward to derive termination.

**Theorem 4.1.** If $\Gamma \vdash e : A$, then $\Gamma \vDash e : A$.

**Proof.** By induction on $\Gamma \vdash e : A$ using Lemmas 4.22 to 4.37.$\square$

**Corollary 4.1.** If $\vdash e : A$, then $(e, \cdot) \leadsto^\star (v, h)$ for some value $v$ and heap $h$.

**Proof.** By Lemma 4.1, we have $\cdot \vdash e : A$. Thus, there is some $\alpha$ such that:

$$\forall i, (e[i|id], \epsilon \oplus R_{\text{ord}(\alpha)}) \in \mathcal{E}[A]_i,$$

since $(id, \epsilon) \in G[\cdot]_i$, for all $i$ by definition. We pick $i \triangleq (\alpha, \emptyset)$. Thus $(e, R_{\text{ord}(\alpha)}) \in \mathcal{E}[A]_i$. By definition of $\mathcal{E}[A]_i$, it suffices to show $R_i(R_{\text{ord}(\alpha)}, \epsilon, \emptyset, \emptyset, \emptyset)$. By definition $\cdot, \emptyset, \emptyset \in \mathcal{H}[\emptyset]_i$. Further, used $\emptyset = \emptyset \neq \emptyset$. Lastly, $\text{sum}(\emptyset) \oplus R_{\text{ord}(\alpha)} \oplus \epsilon \oplus R_{\text{cap}(idk)} = R_{\text{ord}(\alpha)} \subseteq (\emptyset, \emptyset, \alpha)$.$\square$
4.5 Compatibility Lemmas

Lemma 4.22.

\[ x : A \vdash x : A \]

**Proof.** Pick \( \alpha \triangleq 0 \). Let \( (\theta, R) \in \mathcal{G}[x : A] \). Then \( (\theta x, R) \in \mathcal{V}[A] \). It remains to show \( (x[\theta], R \oplus R_{\text{ord}(0)}) \in \mathcal{E}[A] \). As \( (x[\theta], R \oplus R_{\text{ord}(0)}) = (\theta x, R) \), the claim follows with Lemma 4.13. \( \square \)

Lemma 4.23.

\[ \Gamma \vdash e : B \]

\[ \Gamma, x : A \vdash e : B \]

**Proof.** By assumption, we have some \( \alpha_e \) for \( e \). We pick \( \alpha \triangleq \alpha_e \). Let \( (\theta, R_\theta) \in \mathcal{G}[\Gamma, x : A] \). Then \( R_\theta = R_1 \oplus R_2 \) for some \( R_1, R_2 \) such that \( (\theta, R_1) \in \mathcal{G}[\Gamma] \) and \( (\theta, R_2) \in \mathcal{G}[x : A] \) by Lemma 4.19. It remains to show \( (e[\theta], R_1 \oplus R_{\text{ord}(\alpha)} \oplus R_2) \in \mathcal{E}[B] \). By Lemma 4.12, it suffices to show:

\[ (e[\theta], R_1 \oplus R_{\text{ord}(\alpha)}) \in \mathcal{E}[B] \]

which follows by assumption with \( (\theta, R_1) \in \mathcal{G}[\Gamma] \). \( \square \)

Lemma 4.24.

\[ \vdash () : 1 \]

**Proof.** Pick \( \alpha \triangleq 0 \). Let \( (\theta, R) \in \mathcal{G}[\Gamma] \). It remains to show \( ((\emptyset)[\theta], R_{\text{ord}(0)}) \in \mathcal{E}[1] \). As \( ((\emptyset)[\theta], R_{\text{ord}(0)}) = (\emptyset, \epsilon) \in \mathcal{V}[1] \), the claim follows with Lemma 4.13. \( \square \)

Lemma 4.25.

\[ \Gamma \vdash e_1 : 1 \quad \Delta \vdash e_2 : A \]

\[ \Gamma, \Delta \vdash e_1 \cdot e_2 : A \]

**Proof.** By assumption, we have \( \alpha_1 \) for \( e_1 \) and \( \alpha_2 \) for \( e_2 \). We pick \( \alpha \triangleq \alpha_1 \oplus \alpha_2 \). Let \( (\theta, R_\theta) \in \mathcal{G}[\Gamma, \Delta] \). Then \( R_\theta = R_1 \oplus R_2 \) for some \( R_1, R_2 \) such that \( (\theta, R_1) \in \mathcal{G}[\Gamma] \) and \( (\theta, R_2) \in \mathcal{G}[\Delta] \) by Lemma 4.19. It remains to show \( (e_1[\theta]; e_2[\theta], R_1 \oplus R_{\text{ord}(\alpha_1)} \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A] \). By assumption \( (e_1[\theta], R_1 \oplus R_{\text{ord}(\alpha_1)}) \in \mathcal{E}[1] \). By Lemma 4.13, it suffices to show:

\[ (v_1; e_2[\theta], R_{v_1} \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A] \]

for all \( j \leq i \) and \( (v_1, R_{v_1}) \in \mathcal{V}[1] \). By definition of \( \mathcal{V}[1] \), we know \( v_1 = () \) and \( R_{v_1} = \epsilon \). By assumption \( (e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A] \). The claim follows with Lemma 4.13 and Lemma 4.3 given the pure reduction \( () \); \( e_2[\theta] \leadsto e_2[\theta] \). \( \square \)


\[ \vdash b : \mathbb{B} \]
Proof. Pick \( \alpha \triangleq 0 \). Let \((\theta, R) \in \mathcal{G}^\theta\). It remains to show \((b[\theta], R_{\text{ord}(0)}) \in \mathcal{E}[\mathbb{B}]_i\). As \((b[\theta], R_{\text{ord}(0)}) = (b, e) \in \mathcal{V}[\mathbb{B}]_i\), the claim follows with Lemma 4.13.

Lemma 4.27.

\[
\begin{array}{c}
\Gamma \vdash e : \mathbb{B} \\
\Delta \vdash e_1 : A \\
\Delta \vdash e_2 : A
\end{array}
\]

\[\Gamma, \Delta \vdash \text{if } e \text{ then } e_1 \text{ else } e_2 : A\]

Proof. By assumption, we have \(\alpha_e \) for \( e, \alpha_1 \) for \( e_1 \), and \( \alpha_2 \) for \( e_2 \). We pick \( \alpha \triangleq \alpha_e \oplus \alpha_1 \oplus \alpha_2 \). Let \((\theta, R_0) \in \mathcal{G}^\theta\). Then \( R_0 = R_1 \oplus R_2 \) for some \( R_1, R_2 \) such that \((\theta, R_1) \in \mathcal{G}^\theta_i\) and \((\theta, R_2) \in \mathcal{G}^\theta_i\) by Lemma 4.19. Thus, it remains to show:

\[(if \ e[\theta] \ then \ e_1[\theta] \ else \ e_2[\theta], R_1 \oplus R_{\text{ord}(\alpha_1)} \oplus R_2 \oplus R_{\text{ord}(\alpha_1 \oplus \alpha_2)}) \in \mathcal{E}[\mathbb{A}]_i\]

By assumption \((e[\theta], R_1 \oplus R_{\text{ord}(\alpha_1)}) \in \mathcal{E}[\mathbb{B}]_i\). Thus, by Lemma 4.13, it suffices to show:

\[(if \ v \ then \ e_1[\theta] \ else \ e_2[\theta], R_v \oplus R_2 \oplus R_{\text{ord}(\alpha_1 \oplus \alpha_2)}) \in \mathcal{E}[\mathbb{A}]_j\]

for all \( j \leq i \) and \((v, R_v) \in \mathcal{V}[\mathbb{B}]_j\). By definition of the value relation, we have \( R_v = e \) and \( v = \text{true} \) or \( v = \text{false} \).

1. Let \( v = \text{true} \). By assumption \((e_1[\theta], R_2 \oplus R_{\text{ord}(\alpha_1)}) \in \mathcal{E}[\mathbb{A}]_i\). By Lemma 4.12 and Lemma 4.3, we have \((e_1[\theta], R_2 \oplus R_{\text{ord}(\alpha_1 \oplus \alpha_2)}) \in \mathcal{E}[\mathbb{A}]_j\). The claim follows with Lemma 4.13 since if true then \( e_1[\theta] \) else \( e_2[\theta] \sim e_1[\theta] \).

2. Let \( v = \text{false} \). By assumption \((e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[\mathbb{A}]_i\). By Lemma 4.12 and Lemma 4.3, we have \((e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_1 \oplus \alpha_2)}) \in \mathcal{E}[\mathbb{A}]_j\). The claim follows with Lemma 4.13 since if false then \( e_1[\theta] \) else \( e_2[\theta] \sim e_2[\theta] \).

\[\square\]

Lemma 4.28.

\[
\vdash n : \mathbb{N}
\]

Proof. Pick \( \alpha \triangleq 0 \). Let \((\theta, R) \in \mathcal{G}^\theta\). It remains to show \((n[\theta], R_{\text{ord}(0)}) \in \mathcal{E}[\mathbb{N}]_i\). As \((n[\theta], R_{\text{ord}(0)}) = (n, e) \in \mathcal{V}[\mathbb{N}]_i\), the claim follows with Lemma 4.13.

\[\square\]

Lemma 4.29.

\[
\begin{array}{c}
\Gamma \vdash e_1 : \mathbb{N} \\
\Delta \vdash e_2 : \mathbb{N}
\end{array}
\]

\[\Gamma, \Delta \vdash e_1 + e_2 : \mathbb{N}\]

Proof. By assumption, we have \(\alpha_1 \) for \( e_1 \) and \( \alpha_2 \) for \( e_2 \). We pick \( \alpha \triangleq \alpha_1 \oplus \alpha_2 \). Let \((\theta, R_\theta) \in \mathcal{G}^\theta\). Then \( R_\theta = R_1 \oplus R_2 \) for some \( R_1, R_2 \) such that \((\theta, R_1) \in \mathcal{G}^\theta_i\) and \((\theta, R_2) \in \mathcal{G}^\theta_i\) by Lemma 4.19. It remains to show \((e_1[\theta] + e_2[\theta], R_1 \oplus R_{\text{ord}(\alpha_1)} \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[\mathbb{N}]_i\).

27
By assumption \((e_1[\theta], R_1 \oplus R_{\text{ord}(\alpha_1)}) \in \mathcal{E}[A]_i\). By Lemma 4.13, it suffices to show:

\[(v_1 + v_2[\theta], R_{v_1} \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[N]_j\]

for all \(j \leq i\) and \((v_1, R_{v_1}) \in \mathcal{V}[N]_j\). By assumption \((e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[N]_i\) and by Lemma 4.3 \((e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[N]_j\). Thus, by Lemma 4.13, it suffices to show:

\[(v_1 + v_2, R_{v_1} \oplus R_{v_2}) \in \mathcal{E}[N]_k\]

for all \(k \leq j\) and \((v_1, R_{v_1}) \in \mathcal{V}[N]_k\). By definition of \(\mathcal{V}[N]_j\), we know \(v_1 = n_1\) and \(v_2 = n_2\) and \(R_1 = e\) and \(R_2 = e\) for some \(n_1, n_2 \in \mathbb{N}\). As \((n_1 + n_2, e) \in \mathcal{V}[N]_k\), the claim follows with Lemma 4.13 given the pure reduction \(n_1 + n_2 \leadsto n_1 + n_2\).

\[\square\]

**Lemma 4.30.**

\[
\Gamma \models e : \mathbb{N} \quad \Delta \models e_0 : A \quad x : A \models e_S : A \\
\Gamma, \Delta \models \text{iter}(e, e_0, x.e_S) : A
\]

**Proof.** By assumption, we have \(\alpha_e\) for \(e, \alpha_0\) for \(e_0\), and \(\alpha_S\) for \(e_S\). We pick \(\alpha \triangleq \alpha_e \oplus \alpha_0 \oplus \omega \oplus \alpha_S\). Let \((\theta, R_\theta) \in \mathcal{G}[\Gamma, \Delta]_i\). Then \(R_\theta = R_1 \oplus R_2\) for some \(R_1, R_2\) such that \((\theta, R_1) \in \mathcal{G}[\Gamma]_i\) and \((\theta, R_2) \in \mathcal{G}[\Delta]_i\) by Lemma 4.19. Thus, it remains to show:

\[(\text{iter}(e[\theta], e_0[\theta], x.e_S[\theta[x \mapsto v]])], R_1 \oplus R_{\text{ord}(\alpha_1)} \oplus R_2 \oplus R_{\text{ord}(\alpha_0)} \oplus R_{\text{ord}(\omega \oplus \alpha_S)}) \in \mathcal{E}[A]_i\]

We have \((e[\theta], R_1 \oplus R_{\text{ord}(\alpha_1)}) \in \mathcal{E}[N]_i\) and \((e_0[\theta], R_2 \oplus R_{\text{ord}(\alpha_0)}) \in \mathcal{E}[A]_i\) by assumption. Hence, by Lemma 4.18, it suffices to show:

\[(\lambda x.e_S[\theta[x \mapsto v]])], R_{\text{ord}(\omega \oplus \alpha_S)}) \in \mathcal{V}[A \rightarrow A]_i\]

Let \(j \leq i\) and \((v, R_v) \in \mathcal{V}[A]_j\). We show \((e_S[\theta[x \mapsto v]])][v/x], R_{\text{ord}(\alpha_S)} \oplus R_v \in \mathcal{E}[A]_j\). By definition \((\theta[x \mapsto v], R_v) \in \mathcal{G}[x : A]_j\) and thus by assumption \((e_S[\theta[x \mapsto v]], R_v \oplus R_{\text{cap}(\alpha_S)}) \in \mathcal{E}[A]_j\). The claim follows with \(e_S[\theta[x \mapsto v]])[v/x] = e_S[\theta[x \mapsto v]]\).

\[\square\]

**Lemma 4.31.**

\[
\Gamma \models e_1 : A_1 \quad \Delta \models e_2 : A_2 \\
\Gamma, \Delta \models (e_1, e_2) : A_1 \otimes A_2
\]

**Proof.** By assumption, we have \(\alpha_1\) for \(e_1\) and \(\alpha_2\) for \(e_2\). We pick \(\alpha \triangleq \alpha_1 \oplus \alpha_2\). Let \((\theta, R_\theta) \in \mathcal{G}[\Gamma, \Delta]_i\). Then \(R_\theta = R_1 \oplus R_2\) for some \(R_1, R_2\) and \((\theta, R_1) \in \mathcal{G}[\Gamma]_i\) and \((\theta, R_2) \in \mathcal{G}[\Delta]_i\) by Lemma 4.19. It remains to show \(((e_1[\theta], e_2[\theta]), R_1 \oplus R_{\text{ord}(\alpha_1)} \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A_1 \otimes A_2]_i\).

By assumption \((e_1[\theta], R_1 \oplus R_{\text{ord}(\alpha_1)}) \in \mathcal{E}[A_1]_i\). By Lemma 4.13, it suffices to show:

\[((v_1, v_2[\theta]), R_{v_1} \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A_1 \otimes A_2]_j \]

for all \(j \leq i\) and \((v_1, R_{v_1}) \in \mathcal{V}[A_1]_j\). We have \((e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A_2]_i\) by assumption and thus \((e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A_2]_j\) by Lemma 4.3. By Lemma 4.13, it suffices to show:

\[((v_1, v_2), R_{v_1} \oplus R_{v_2}) \in \mathcal{E}[A_1 \otimes A_2]_k \]

for all \(k \leq j\) and \((v_2, R_{v_2}) \in \mathcal{V}[A_2]_k\). The claim follows with Lemma 4.12 and Lemma 4.3.  \[\square\]
Lemma 4.32.
\[ \Gamma \models e_1 : A_1 \otimes A_2 \quad \Delta, x : A_1, y : A_2 \models e_2 : B \]
\[ \Gamma, \Delta \models \text{let } (x, y) = e_1 \text{ in } e_2 : B \]

Proof. By assumption, we have \( \alpha_1 \) for \( e_1 \) and \( \alpha_2 \) for \( e_2 \). We pick \( \alpha \triangleq \alpha_1 \oplus \alpha_2 \). Let \( (\theta, R_\theta) \in \mathcal{G}[\Gamma, \Delta]_i \). Then \( R_\theta = R_1 \oplus R_2 \) for some \( R_1, R_2 \) and \( (\theta, R_1) \in \mathcal{G}[\Gamma ]_i \) and \( (\theta, R_2) \in \mathcal{G}[\Delta]_i \) by Lemma 4.19. It remains to show:
\[ (\text{let } (x, y) = e_1[\theta] \text{ in } e_2[\theta[x \mapsto x, y \mapsto y]], R_1 \oplus R_{\text{ord}(\alpha_1)} \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[B]_i \]

By assumption \((e_1[\theta], R_1 \oplus R_{\text{ord}(\alpha_1)}) \in \mathcal{E}[A_1 \otimes A_2]_i\). By Lemma 4.13, it suffices to show:
\[ (\text{let } (x, y) = v \text{ in } e_2[\theta[x \mapsto x, y \mapsto y]], R_v \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[B]_j \]
for all \( j \leq i \) and \((v, R_v) \in \mathcal{V}[A_1 \otimes A_2]_j\). By definition of the value relation, we have \( v = (v_1, v_2) \) and \( R = R_{v_1} \oplus R_{v_2} \) for some \((v_1, R_{v_1}) \in \mathcal{V}[A_1]_j\) and \((v_2, R_{v_2}) \in \mathcal{V}[A_2]_j\).

By Lemma 4.13, it suffices to show
\[ (e_2[\theta[x \mapsto v_1, y \mapsto v_2]], R_{v_1} \oplus R_{v_2} \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[B]_j \]

since let \( (x, y) = (v_1, v_2) \) in \( e_2[\theta[x \mapsto x, y \mapsto y]] \mapsto e_2[\theta[x \mapsto x, y \mapsto y]][v_1/x, v_2/y] = e_2[\theta[x \mapsto v_1, y \mapsto v_2]]. \) By Lemma 4.20, we have \( (\theta, R_2) \in \mathcal{G}[\Delta]_j \). Thus, \((\theta[x \mapsto v_1, y \mapsto v_2], R_2 \oplus R_{v_1} \oplus R_{v_2}) \in \mathcal{G}[\Delta, x : A_1, y : A_2]_j \) by Lemma 4.21. The claim follows by the assumption for \( e_2 \).

\[ \square \]

Lemma 4.33.
\[ \Gamma, x : A \models e : B \]
\[ \Gamma \models \lambda x. e : A \rightarrow B \]

Proof. By assumption we have some \( \alpha_e \) for \( e \). We pick \( \alpha \triangleq \alpha_e \). Let \( (\theta, R_\theta) \in \mathcal{G}[\Gamma]_i \). It remains to show \((\lambda x. (e[\theta[x \mapsto x]]), R_\theta \oplus R_{\text{ord}(\alpha)}) \in \mathcal{E}[A \rightarrow B]_i \). By Lemma 4.13, \((\lambda x. (e[\theta[x \mapsto x]]), R_\theta \oplus R_{\text{ord}(\alpha)}) \in \mathcal{V}[A \rightarrow B]_j \), suffices. Let \( j \leq i \) and \((v, R_v) \in \mathcal{V}[A]_j \). We show:
\[ (e[\theta[x \mapsto x]][v/x], R_\theta \oplus R_{\text{ord}(\alpha)} \oplus R_v) \in \mathcal{E}[B]_j \]

By Lemma 4.20, we have \((\theta, R_\theta) \in \mathcal{G}[\Gamma]_j \). Hence \((\theta[x \mapsto v], R_\theta \oplus R_v) \in \mathcal{G}[\Gamma, x : A]_j \) by Lemma 4.21. By assumption, we have \((e[\theta[x \mapsto v]], R_\theta \oplus R_v \oplus R_{\text{ord}(\alpha)}) \in \mathcal{E}[B]_j \). The claim follows with \( e[\theta[x \mapsto v]] = e[\theta[x \mapsto x]][v/x] \).

\[ \square \]

Lemma 4.34.
\[ \Gamma \models e_1 : A \rightarrow B \quad \Delta \models e_2 : A \]
\[ \Gamma, \Delta \models e_1 \ e_2 : B \]
Lemma 4.35.

\[
\Gamma \vdash e_1 : \text{Get } A \quad \Delta \vdash e_2 : A \to 1 \\
\Gamma, \Delta \vdash (e_1, e_2) : 1
\]

Proof. By assumption, we have \(\alpha_1\) for \(e_1\) and \(\alpha_2\) for \(e_2\). We pick \(\alpha \triangleq \alpha_1 \oplus \alpha_2\). Let \((\theta, R_\theta) \in G[\Gamma, \Delta]_i\). Then \(R_\theta = R_1 \oplus R_2\) for some \(R_1, R_2\) such that \((\theta, R_1) \in G[\Gamma]_i\) and \((\theta, R_2) \in G[\Delta]_i\) by Lemma 4.19. It remains to show \((e_1[\theta], e_2[\theta], R_1 \oplus R_{\text{ord}(\alpha_1)} \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[B]_i\).

By assumption \((e_1[\theta], R_1 \oplus R_{\text{ord}(\alpha_1)}) \in \mathcal{E}[A \to B]_i\). By Lemma 4.13, it suffices to show:

\[
(v_1 e_2[\theta], R v_1 \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[B]_j
\]

for all \(j \leq i\) and \((v_1, R_{v_1}) \in \mathcal{V}[A \to B]_j\). By assumption, we have \((e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A]_j\) and thus \((e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A]_j\) by Lemma 4.3. Hence, by Lemma 4.13, it suffices to show:

\[
(v_1 v_2, R v_1 \oplus R v_2) \in \mathcal{E}[B]_k
\]

for all \(k \leq j\) and \((v_2, R_{v_2}) \in \mathcal{V}[A]_k\). By definition of \(\mathcal{V}[A \to B]_j\), we know \(v_1 = \lambda x.e\) for some \(x\) and \(e\). Further, we have \((e[v_2/x], R v_1 \oplus R v_2) \in \mathcal{E}[B]_k\). The claim follows by Lemma 4.13 given the pure reduction \((\lambda x.e) v_2 \rightsquigarrow e[v_2/x]\).

\[
\square
\]

Lemma 4.36.

\[
\Gamma \vdash e_1 : \text{Put } A \quad \Delta \vdash e_2 : A \\
\Gamma, \Delta \vdash \text{put}(e_1, e_2) : 1
\]

Proof. By assumption, we have \(\alpha_1\) for \(e_1\) and \(\alpha_2\) for \(e_2\). We pick \(\alpha \triangleq \alpha_1 \oplus \alpha_2\). Let \((\theta, R_\theta) \in G[\Gamma, \Delta]_i\). Then \(R_\theta = R_1 \oplus R_2\) for some \(R_1, R_2\) such that \((\theta, R_1) \in G[\Gamma]_i\) and \((\theta, R_2) \in G[\Delta]_i\) by Lemma 4.19. It remains to show \((\text{put}(e_1[\theta], e_2[\theta]), R_1 \oplus R_{\text{ord}(\alpha_1)} \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[B]_i\).

By assumption \((e_1[\theta], R_1 \oplus R_{\text{ord}(\alpha_1)}) \in \mathcal{E}[\text{Put } A]_i\). By Lemma 4.13, it suffices to show:

\[
(\text{put}(v_1, e_2[\theta]), R v_1 \oplus R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[B]_j
\]

for all \(j \leq i\) and \((v_1, R_{v_1}) \in \mathcal{V}[\text{Put } A]_j\). By assumption, we have \((e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A]_j\) and thus by Lemma 4.3 it follows \((e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A \to 1]_j\). Hence, by Lemma 4.13, it suffices to show:

\[
(\text{put}(v_1, v_2), R v_1 \oplus R v_2) \in \mathcal{E}[1]_k
\]

for all \(k \leq j\) and \((v_2, R_{v_2}) \in \mathcal{V}[A \to 1]_k\). The claim follows with Lemma 4.15 and Lemma 4.3.

\[
\square
\]
for all $j \leq i$ and $(v_1, R_{v_1}) \in \mathcal{V}[\text{Put } A]_j$. By assumption, we have $(e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A]_j$. Thus, $(e_2[\theta], R_2 \oplus R_{\text{ord}(\alpha_2)}) \in \mathcal{E}[A]_j$ by Lemma 4.3. Hence, by Lemma 4.13, it suffices to show:

$$(\text{put}(v_1, v_2), R_{v_1} \oplus R_{v_2}) \in \mathcal{E}[1]_k$$

for all $k \leq j$ and $(v_2, R_{v_2}) \in \mathcal{V}[A]_k$. The claim follows with Lemma 4.16 and Lemma 4.3. □

**Lemma 4.37.**

$$\Gamma, x : \text{Get } A, y : \text{Put } A \vdash e : B$$

$$\Gamma \vdash \text{let } (x, y) = \text{chan}() \text{ in } e : B$$

**Proof.** By assumption we have some $\alpha_c$ for $e$. We pick $\alpha = \alpha_c \oplus 1$. Let $(\theta, R_\theta) \in \mathcal{G}[\Gamma]_i$. We show (let $(x, y) = \text{chan}()$ in $(e[\theta[x \mapsto x, y \mapsto y]]), R_\theta \oplus R_{\text{ord}(\alpha_c)} \oplus R_{\text{ord}(1)}) \in \mathcal{E}[B]_i$. To show the claim, we use Lemma 4.14. Let $\ell$ be some location. It remains to show:

$$(e[\theta[x \mapsto x, y \mapsto y]][\ell/x, \ell/y], R_\theta \oplus R_{\text{cap}(\alpha_c)} \oplus R_{\text{get}(\ell, A)} \oplus R_{\text{put}(\ell, A)}) \in \mathcal{E}[B]_i$$

By definition $(\ell, R_{\text{get}(\ell, A)}) \in \mathcal{V}[\text{Get } A]_i$, and $(\ell, R_{\text{put}(\ell, A)}) \in \mathcal{V}[\text{Put } A]_i$. Thus, we have:

$$(\theta[x \mapsto \ell, y \mapsto \ell], R_\theta \oplus R_{\text{get}(\ell, A)} \oplus R_{\text{put}(\ell, A)}) \in \mathcal{G}[\Gamma, x : \text{Get } A, y : \text{Put } A]_i$$

with Lemma 4.21. The claim follows by assumption with $e[\theta[x \mapsto x, y \mapsto y]][\ell/x, \ell/y] = e[\theta[x \mapsto \ell, y \mapsto \ell]].$ □
References


