Continuation Semantics for Concurrency

Gabriel Ciobanu and Eneia Nicolae Todoran

Approved for public release; further dissemination unlimited.

Prepared by
Formal Methods Laboratory.

Series of
Technical Reports (Institute of Computer Science Iași)

ISSN 1842 - 1490
FML is the Formal Methods Laboratory from the Institute of Computer Science of the Romanian Academy and is located in Iași. It is a research institution aimed at developing new formalisms for challenging open problems in computer science, systems biology and other emerging research fields. FML is headed by Dr. Gabriel Ciobanu.

IIT is the Institute of Computer Science of the Romanian Academy and is concerned with basic research in computer science.

Romanian Academy of Science is the supreme forum of science from Romania.

Calea Victoriei 125, Sector 1, 010071, Bucharest, ROMANIA.

Telephone: +40 21 2128640
Telefax: +40 21 2116608
Web: http://www.academiaromana.ro/

Printed in Romania and reproduced directly from the authors original copy.
Continuation Semantics for Concurrency

by

Gabriel Ciobanu 1,2
Institute of Computer Science, Romanian Academy, Iaşi

gabriel@iit.tuiasi.ro

Eneia Nicolae Todoran
Technical University of Cluj-Napoca,
Department of Computer Science
400027 Cluj-Napoca, Romania

Eneia.Todoran@cs.utcluj.ro

ABSTRACT

This paper presents a continuation semantics satisfying the basic laws of concurrent systems. This semantics is illustrated for a simple CSP-like language extended with communication on multiple channels and synchronization based on join patterns, which also provides operators for nondeterministic choice, sequential and parallel composition. For the language under investigation we present a denotational semantics. Then we prove that the semantic operators designed with continuations obey the concurrency laws such as the associativity or the commutativity of parallel composition. The significance of the results is given mainly by the flexibility provided by the continuations technique which can thus be used to describe concurrent behaviour.

1Head of Formal Methods Laboratory (FML)
2Contact person
## Contents

1 Introduction ......................................................... 2

2 Preliminaries ........................................................ 4

3 Syntax and Continuation Structure for *MCC* ................ 6

4 Continuation Semantics for *MCC* ............................... 8

5 Concurrency Laws .................................................. 12

6 Concluding Remarks ............................................... 49
1 Introduction

The paper deals with fundamental aspects of systems exhibiting concurrency, namely systems consisting of several computing processes interacting among each other. By “process” we understand the behaviour of a (software) system, the execution of a program. Since the concurrent systems are usually complex, it is useful to have the possibility of describing, analyzing and reasoning about the concurrent systems in a precise way. Generally the interactions among processes cannot be predicted, and so the concurrent computation involves nondeterminism. The traditional view of computation as a function from input to output cannot deal properly with nondeterminism; the models of concurrency are provided by process calculi like ACP [5], CCS [10] and CSP [9]. These process calculi work with terms, and use an operational semantics to describe the computational dynamics. They are also called process algebras because they use algebraic operators and equations among terms to describe the system behaviour.

In general, starting out from a given set of atomic actions, the basic operators are used to compose the actions into more complicated processes. The basic operators are + (denoting alternative composition), ; (denoting sequential composition) and ∥ (denoting parallel composition). The following basic laws express the general properties of the operators involved in a concurrent system. These are static laws because they do describe the action executions explicitly.

\[
\begin{align*}
  x + y &= y + x \text{(commutativity of alternative composition)} \\
  x + (y + z) &= (x + y) + z \text{(associativity of alternative composition)} \\
  x + x &= x \text{(idempotency of alternative composition)} \\
  (x + y); z &= x; z + y; z \text{(left distributivity of + over ;)} \\
  x; (y + z) &= x; y + x; z \text{(right distributivity of + over ;)} \\
  (x; y); z &= x; (y; z) \text{(associativity of sequential composition)} \\
  x || y &= y || x \text{ (commutativity of parallel composition)} \\
  (x || y); z &= x || (y || z) \text{(associativity of parallel composition)}
\end{align*}
\]

The operator ; binds more tightly than +. The axiom for the left distributivity of + over ; is not always included in concurrency theories. When this axiom is present one gets a trace or linear-time semantics where the nondeterministic choice points are not recorded. Without this axiom, one gets a branching time model that can provide better flexibility in the modeling of process synchronization.\(^1\)

1When considering an approach to concurrency theory, parallel composition is the central operator, and so the importance of its proper treatment in a continuation semantics for concurrency.

\(^1\)In this paper we use the notions of linear time and branching time semantics in the context of denotational semantics. In this context, an element of a linear time domain is a collection of execution traces. An element of a branching time domain is a tree-like structure. For a discussion of linear time versus branching time semantics the reader may consult [2] or the monograph [3].
The denotational semantics of concurrent systems uses the observations of a process (e.g., traces, failures). The meaning of a process is given by a set of possible observations. These denotations are compositional and provide an operational intuition. However the denotational semantics are more abstract than the operational semantics, and are not close to any specific implementation for a concurrent system or language. In this paper we present a continuation semantics for concurrency (CSC) which satisfies the static laws of concurrent systems, and which can provide flexibility in designing concurrent languages.

Continuations are known in denotational semantics for the flexibility they provide as a language design tool. Traditional continuations [12] can be used to model a variety of advanced control concepts, including non-local exists, coroutines and even multitasking [16]. However, according to [8], the traditional continuations do not work well-enough in the presence of concurrency. The CSC technique [13] is a general tool for representing control in concurrent systems. A CSC continuation is a structure of computations (partially evaluated denotations) that can be evaluated in parallel.

The structure of CSC continuations is representative for the control concepts of the (concurrent) language under study. In the case of a simple sequential language a continuation is a stack of computations. For handling parallel composition, CSC continuations can be structured as multisets of computations. Communication and synchronization information can also be encoded in continuations. Thanks to the use of continuations, we do not need a branching time domain to model communication and synchronization behavior. The final yield of the denotational semantics is an element of a linear-time domain, i.e. a simple collection of execution traces. The CSC technique provides a "pure" continuation-based approach to communication and concurrency, in which all control concepts are modeled as operations manipulating continuations. In the CSC approach the heavy part of the computation is moved from the powerdomains [11] to the domains of continuations.

In this paper we consider a simple imperative concurrent language \(MCC\) extended with communication on multiple channels in the style of Join calculus [7]. \(MCC\) is an abbreviation for \(Multiple\ Channels\ Communication\). The language \(MCC\) was first investigated in [14] (although in [14] it was named \(L_J\)). \(MCC\) provides operators for sequential composition, parallel composition and nondeterministic choice. It also provides two primitives for concurrent interaction on multiple channels: \(c!e\), and \(c?v_1\&\cdots\&c_n?v_n\) which we call a join pattern. Synchronized execution of \(n + 1\) actions \(c_1!e_1,\cdots,c_ne_n\) and \(c_1?v_1\&\cdots\&c_n?v_n\) occurring in parallel processes, results in the transmission of the value of each expression \(e_i\) along the channel \(c_i\) from the process executing the \(c_i!e_i\) statement to the process executing the \(c_1?v_1\&\cdots\&c_n?v_n\) statement. The latter assigns the \(n\) received values to the variables \(v_1,\cdots,v_n\): the value of each expression \(e_i\) is transmitted along the channel \(c_i\) and assigned to the corresponding variable \(v_i\). The whole interaction behaves like a distributed multi-assignment. When \(n = 1\) the interaction is a simple CSP-like point-to-point communication [9].

We present a denotational model designed with CSC for \(MCC\). The denotational model is built within the mathematical framework of 1-bounded complete metric spaces [3]. We use the general theory developed in [1] as the domain of denotations is defined as solution
of an equation in where the domain variable occurs in the left-hand side of a function space construction. We prove that the semantic operators - that are designed with CSC continuations - obey the laws that are usually given in the algebraic theories of concurrency, such as the associativity and the commutativity of parallel composition.

Some of the properties are simple. For example, the semantics of nondeterministic choice is given by the standard union operator, which is associative, commutative and idempotent. Other properties, e.g. the associativity of sequential and parallel execution, require more complex proofs. The basic idea of each of the main proofs is to show that the property under consideration is preserved by the computation steps (it is an invariant of the computation). In metric semantics it is customary to attach a $\frac{1}{2}$ -contracting factor to each computation step. In each case we obtain a relation of the kind $\epsilon \leq \frac{1}{2} \cdot \epsilon$ where $\epsilon$ is the distance between two behaviorally equivalent continuations. Therefore $\epsilon = 0$ and the desired property follows. This proof technique is general and can be applied to every language designed with CSC. Only the structure of continuations and the computation invariants (which are language specific and are related to the structure of continuations) need to be adapted.

2 Preliminaries

The notation $(x \in) X$ introduces the set $X$ with typical element $x$ ranging over $X$. For any set $X$ we denote by $|X|$ the cardinal number of $X$. Let $f \in X \to Y$ be a function. The function $[f \mid x \mapsto y] : X \to Y$, is defined (for $x, x' \in X, y \in Y$) by: $[f \mid x \mapsto y](x') = \begin{cases} y & \text{if } x' = x \\ f(x') & \text{else} \end{cases}$. We also use the notation $[f \mid x_1 \mapsto y_1 | \cdots | x_n \mapsto y_n]$ as an abbreviation for $[\cdots [f \mid x_1 \mapsto y_1 \mid \cdots | x_n \mapsto y_n] \cdots]$. If $f : X \to X$ and $f(x) = x$ we call $x$ a fixed point of $f$. When this fixed point is unique (see Theorem 2.1) we write $x = \text{fix}(f)$.

The study presented in this paper takes place in the mathematical framework of 1-bounded complete metric spaces. We work with the following notions which we assume to be known: metric and ultrametric spaces, isometry (distance preserving bijection between metric spaces) denoted by ‘$\cong$‘, and complete metric space. For further explanations the reader may consult the monograph [3].

A simple example of ultrametric space can be obtained by endowing an arbitrary nonempty set $(x, y \in) X$ with the metric $d : X \times X \to [0, 1]$ defined by $d(x, y) = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{else} \end{cases}$. $d$ is the so-called discrete metric on $X$. $(X, d)$ is a complete ultrametric space.

Let $(a \in) A$ be a nonempty set. We denote by $A^*$ the set of all finite sequences over $A$, and we denote by $A^\omega$ the set of all infinite sequences over $A$. We obtain another example of ultrametric space by endowing the set $(x, y \in) A^\omega = A^* \cup A^\omega$ with the metric $d : A^\omega \times A^\omega \to [0, 1]$, given by: $d(x, y) = 2^{-\sup \{n \mid x[n] = y[n]\}}$, where $x[n]$ denotes the prefix of $x$ of length $n$ if $\text{length}(x) \geq n$, and $x$ otherwise. By convention, $2^{-\infty} = 0$. Such a $d$ is a Baire metric, and $(A^\omega, d)$ is a complete ultrametric space.

We recall that if $(X, d_X), (Y, d_Y)$ are metric spaces, a function $f : X \to Y$ is a contraction if $\exists c \in \mathbb{R}, 0 \leq c < 1, \forall x_1, x_2 \in X : d_Y(f(x_1), f(x_2)) \leq c \cdot d_X(x_1, x_2)$. When $c = 1$ the function $f$ is called non-expansive. In what follows we denote the set of all nonexpansive functions
from $X$ to $Y$ by $X \rightarrow Y$. The following Theorem is at the core of metric semantics.

**THEOREM 2.1 (Banach)** Let $(X, d_X)$ be a complete metric space. Each contraction $f : X \rightarrow X$ has a unique fixed point.

**DEFINITION 2.1** Let $(X, d_X), (Y, d_Y)$ be (ultra) metric spaces with $d_X, d_Y \leq 1$.

We define the following metrics for $x \in X$, $f \in X \rightarrow Y$ (the function space), $(x, y) \in X \times Y$ (the Cartesian product), $u, v \in X + Y$ (the disjoint union of $X$ and $Y$), and $U, V \in \mathcal{P}(X)$ (the power set of $X$):

(a) $d_{\frac{1}{2}} : X \times X \rightarrow [0, 1]$ 
\[ d_{\frac{1}{2}}(x_1, x_2) = \frac{1}{2} \cdot d_X(x_1, x_2) \]

(b) $d_{X \rightarrow Y} : (X \rightarrow Y) \times (X \rightarrow Y) \rightarrow [0, 1]$ 
\[ d_{X \rightarrow Y}(f_1, f_2) = \sup_{x \in X} d_Y(f_1(x), f_2(x)) \]

(c) $d_{X \times Y} : (X \times X) \times (Y \times Y) \rightarrow [0, 1]$ 
\[ d_{X \times Y}((x_1, y_1), (x_2, y_2)) = \max\{d_X(x_1, x_2), d_Y(y_1, y_2)\} \]

(d) $d_{X + Y} : (X + Y) \times (X + Y) \rightarrow [0, 1]$ 
\[ d_{X + Y}(u, v) = \begin{cases} d_X(u, v) & \text{if } (u, v) \in X \\ d_Y(u, v) & \text{else if } (u, v) \in Y \end{cases} \]

(The disjoint union can be defined by: $X + Y = (\{1\} \times X) \cup (\{2\} \times Y)$)

(e) $d_H : \mathcal{P}(X) \times \mathcal{P}(X) \rightarrow [0, 1]$:
\[ d_H(U, V) = \max\{\sup_{u \in U} d(u, V), \sup_{v \in V} d(u, U)\} \]

where $d(u, W) = \inf_{w \in W} d(u, w)$ and by convention $\sup\emptyset = 0$, $\inf\emptyset = 1$ 

($d_H$ is the Hausdorff distance).

We use the abbreviations $\mathcal{P}_{nco}(\cdot)$, $\mathcal{P}_{fin}(\cdot)$ and $\mathcal{P}_{nfin}(\cdot)$, to denote the power sets of non-empty and compact, finite and non-empty and finite subsets of ‘·’, respectively. In general, the constructs $\mathcal{P}_{fin}(\cdot)$ and $\mathcal{P}_{nfin}(\cdot)$ do not give rise to complete spaces. In our study, we only use the power set of finite subsets to create structures that we endow with the discrete metric; any set endowed with the discrete metric is trivially a complete ultrametric space. Also, we often suppress the metrics part in domain definitions, and write, e.g., $\frac{1}{2} \cdot X$ instead of $(X, d_{\frac{1}{2}} X)$.

**Remark 1** Let $(X, d_X), (Y, d_Y), d_{\frac{1}{2}} : X \rightarrow Y, d_{X \times Y}, d_{X + Y}$ and $d_H$ be as in definition 2.1. In case $d_X, d_Y$ are ultrametrics, so are $d_{\frac{1}{2}} : X \rightarrow Y, d_{X \times Y}, d_{X + Y}$ and $d_H$. Moreover, if $(X, d_X), (Y, d_Y)$ are complete then $\frac{1}{2} \cdot X, X \rightarrow Y, X \rightarrow Y, X \times Y, X + Y$, and $\mathcal{P}_{nco}(X)$ (with the metrics defined above) are also complete metric spaces [3].
3 Syntax and Continuation Structure for MCC

The syntax of MCC is given in BNF in 3.1. The basic components are a set \( (v \in \text{Var}) \) of variables, a set \( (e \in \text{Exp}) \) of expressions, a set \( (c \in \text{Ch}) \) of communication channels and a set \( (x \in \text{Pvar}) \) of procedure variables. \( \text{Exp} \) is a class of numeric expressions without side effects that evaluate to integer values \( z \in \mathbb{Z} \). We assume that the evaluation of an expression always terminates.

**DEFINITION 3.1** (Syntax of MCC)

(a) (Join patterns) \( j(\in J) ::= c?v \mid j \& j \)

For a program to be valid the channels \( c_1, \cdots, c_n \) and the variables \( v_1, \cdots, v_n \) in a join pattern \( j = (c_1?v_1 & \cdots & c_n?v_n) \) must be pairwise distinct.

(b) (Statements) \( s(\in \text{Stat}) ::= a \mid x \mid s + s \mid s \parallel s \mid s \parallel s \)

where \( a ::= \text{skip} \mid v := e \mid c!e \mid j \)

(c) (Guarded statements) \( g(\in \text{GStat}) ::= a \mid g + g \mid g \parallel g \parallel g \)

(d) (Declarations) \( (D \in \text{Decl}) = \text{PVar} \to \text{GStat} \)

(e) (Programs) \( (\rho \in \text{MCC}) = \text{Decl} \times \text{Stat} \)

The language MCC provides assignment \( (v := e) \), recursion, sequential composition \( (s; s) \), nondeterministic choice \( (s + s) \), parallel composition \( (s \parallel s) \) and the communication mechanism that was explained informally in the introduction. Following a standard approach, we assume the meanings of expressions is given by a valuation \( \mathcal{E}[.] : \text{Exp} \to \Sigma \to \mathbb{Z} \), where \( (\sigma \in \Sigma) = \text{Var} \to \mathbb{Z} \) is a set of states.

We employ an approach to recursion based on declarations and guarded statements as in [3]. In a guarded statement each recursive call is preceded by at least one elementary action, which guarantees the fact that the semantic operators are contracting functions in the present metric setting. For the sake of brevity, in what follows we assume a fixed declaration \( D \in \text{Decl} \), and all considerations in any given argument refer to this fixed \( D \).

For inductive proofs we introduce a complexity measure that decreases upon recursive calls. A function such as \( c \) is well-defined due to our restriction to guarded recursion (Definition 3.1(c), 3.1(d)).

**DEFINITION 3.2** (Complexity measure) The function \( c : \text{Stat} \to \mathbb{N} \) is given by

\[
\begin{align*}
c(a) &= 1 \\
c(s_1; s_2) &= 1 + c(s_1) \\
c(x) &= 1 + c(D(x)) \\
c(s_1 \text{ op } s_2) &= 1 + \max\{c(s_1), c(s_2)\}, \text{ op } \in \{+ \parallel\}
\end{align*}
\]
In the CSC approach a continuation is an application-specific structure of computations. The language MCC provides a general combination of parallel and sequential composition. As shown in [13], in order to model this combination of concepts a CSC continuation must be a tree of computations with active computations at the leaves. The major issue that gives rise to a tree-like structure is the presence of statements such as \((s_1 \parallel s_2); s_3\). In such a statement, \(s_3\) can only execute after both \(s_1\) and \(s_2\) have terminated. Following [13], in order to define such domains of trees of computations we use a (partially ordered) set of identifiers.

**DEFINITION 3.3**

(a) Let \((\alpha \in)\Id = \{1, 2\}^*\) be a set of identifiers, equipped with the following partial ordering:
\[
\alpha \leq \alpha' \iff \alpha' = \alpha \cdot i_1 \cdots i_n \text{ for } i_1, \ldots, i_n \in \{1, 2\}, n \geq 0.
\]

(b) We define a function \(\text{max} : \mathcal{P}(\Id) \rightarrow \mathcal{P}(\Id)\) by:
\[
\text{max}(A) = \{\alpha \mid \alpha \text{ is a maximal element of } (A, \leq_A)\}
\]

where \(A \in \mathcal{P}(\Id)\) and \(\leq_A\) is the restriction of \(\leq\) to the subset \(A\) of \(\Id\).

\(\Id\) is the set of all finite, possibly empty, sequences over \(\{1, 2\}\) and \(\alpha \leq \alpha'\) iff \(\alpha\) is a prefix of \(\alpha'\). We use the symbol \(\lambda\) to represent the empty sequence over \(\{1, 2\}\) (i.e. \(\lambda \in \Id = \{1, 2\}^*\)). Throughout this paper we use the symbol \(\cdot\) as a concatenation operator over sequences.

We can represent tree-like structures over \((\Id, \leq)\). Let \(A = \{\alpha, \alpha \cdot 1, \alpha \cdot 2, \alpha \cdot 1 \cdot 1, \alpha \cdot 1 \cdot 2, \alpha \cdot 2 \cdot 1, \alpha \cdot 2 \cdot 2\}\); the maximal elements of \((A, \leq_A)\) are exactly the leaves of the tree: \(\text{max}(A) = \{\alpha \cdot 1 \cdot 1, \alpha \cdot 1 \cdot 2, \alpha \cdot 2 \cdot 1, \alpha \cdot 2 \cdot 2\}\). The elements of the set \(\Id\) are used to distinguish between multiple occurrences of a computation in a continuation. The partial ordering \(\leq\) defined on \(\Id\) is used to implement the semantics of sequential composition in the presence of parallelism as explained above.

Let \((\pi) \in \Pi = \mathcal{P}_{\text{fin}}(\Id)\) and let \((x) \in \mathcal{X}\) be a metric domain, i.e. a complete metric space. We use the following notation: \(\|\mathcal{X}\| \not= \Pi \times (\Id \rightarrow \mathcal{X})\). Let \(\alpha \in \Id\), \((\pi, \phi) \in \|\mathcal{X}\|\) with \(\pi \in \Pi\), \(\phi \in \Id \rightarrow \mathcal{X}\). We define \(id : \|\mathcal{X}\| \rightarrow \Pi\), \(id(\pi, \phi) = \pi\). We also use the following abbreviations:
\[
\begin{align*}
(\pi, \phi)(\alpha) & \not= \phi(\alpha) \quad \text{ (in } \mathcal{X}) \\
(\pi, \phi) \setminus \pi' & \not= (\pi \setminus \pi', \phi) \quad \text{ (in } \|\mathcal{X}\|) \\
[(\pi, \phi) \mid \alpha \mapsto x] & \not= (\pi \cup \{\alpha\}, [\phi \mid \alpha \mapsto x]) \quad \text{ (in } \|\mathcal{X}\|)
\end{align*}
\]

The basic idea is that we treat \((\pi, \phi)\) as a 'function' with finite graph \(\{(\alpha, \phi(\alpha)) \mid \alpha \in \pi\}\), thus ignoring the behaviour of \(\phi\) for any \(\alpha \notin \pi\) (\(\pi\) is the 'domain' of \((\pi, \phi)\)). We find convenient to use this mathematical structure to represent finite partially ordered bags (or multisets)\(^2\) of computations. As mentioned before, \(\Id\) is used to distinguish between multiple

\(^2\)We avoid using the notion of a partially ordered multiset which is a more refined structure – see [4], or chapter 16 of [3].
occurrences of a computation in such a bag. We endow both sets $Id$ and $\Pi$ with discrete metrics; every set with a discrete metric is a complete ultrametric space. According to Definition 2.1 and Remark 1, $(Id \to X, d_{Id\to X})$ becomes also a domain (i.e., a complete ultrametric space). It follows that $\Pi \times (Id \to X)$ becomes a domain. The metric over $\{X\} = \Pi \times (Id \to X)$ is $d' = d_\Pi \times d_{Id\to X}$ given by $d'((\pi, \phi), (\pi', \phi')) = \max\{d_\Pi(\pi, \pi'), d_{Id\to X}(\phi, \phi')\}$, where $d_{Id\to X}(\phi, \phi') = \sup_{\alpha \in Id} d_X(\phi(\alpha), \phi'(\alpha))$ and the metric $d_X$ over $X$ is known. Intuitively, the operators behave as follows. $id(\pi, \phi)$ returns the collection of identifiers for the valid computations contained in the bag $(\pi, \phi)$, $(\pi, \phi)(\alpha)$ returns the computation with identifier $\alpha$, $(\pi, \phi) \setminus \pi'$ removes the computations with identifiers in $\pi'$, and $[(\pi, \phi) \mid \alpha \mapsto x]$ replaces the computation with identifier $\alpha$.

By a slight abuse we use the same notations (including the operator $id$ and the abbreviations $(\cdot)(\alpha), (\cdot) \setminus \pi, [\cdot \mid \alpha \mapsto x]$) when $(x \in) X$ is an ordinary set: $\{X\} = \Pi \times (Id \to X)$; in this case we do not endow $\{X\}$ with a metric structure.

### 4 Continuation Semantics for MCC

We design a continuation-based denotational semantics for $MCC$. As a semantic universe for the final yield of our denotational model we employ a standard linear-time domain $(p \in )P = P_{nco}(Q)$, where $Q = \Sigma^* \cup \Sigma^* \{\delta\} \cup \Sigma^\infty$. Here $\Sigma^*(\Sigma^\infty)$ denotes the collection of all finite (infinite) sequences over $\Sigma$. $Q$ contains finite sequences, possibly terminated with the symbol $\delta$ which denotes deadlock, or infinite sequences over $\Sigma$. We use the symbol $\lambda$ to represent the empty sequence over $\Sigma$. This is a slight abuse, since we also use the symbol $\lambda$ to represent the empty sequence over $\{1, 2\}$; however, it is always clear from the context which is the type of $\lambda$ (either $\lambda \in Id = \{1, 2\}^*$, or $\lambda \in \Sigma^*$). As it is well-known [3], by endowing $\Sigma$ with the discrete metric we obtain a Baire metric on $Q$. The type of the denotational semantics $D$ for $MCC$ is $Sem_D = Stat \to D$, where:

$$D \cong \text{Cont} \to \Sigma \to P$$

$$\begin{align*}
(\gamma \in )\text{Cont} &= Id \times Kont \\
(\kappa \in )\text{Kont} &= \{\text{Comp}\} \\
\text{Comp} &= J^\otimes + \text{Snd} + \frac{1}{2} \cdot D
\end{align*}$$

Here $J^\otimes = \{\otimes\} \times J$, $\text{Snd} = Ch \times \Xi$ and $(\xi \in )\Xi = \Sigma \to \mathbb{Z}$. We use the notation $(j) = (\otimes, j)$. Also, for readability we denote typical elements $(c, \xi)$ of $\text{Snd}$ by $c!\xi$. In the equations above the sets $Id$ (and $\Pi$), $\Sigma, J^\otimes$ and $\text{Snd}$ are endowed with the discrete metric, which is an ultrametric. The composed metric spaces are built up using the composite metrics of Definition 2.1. To conclude that such a system of equations has a (unique) solution (up to isometry) we rely on the general theory developed in [1]. The solution for $D$ is obtained as a complete ultrametric space.

$\text{Comp}$ is the domain of computations. A computation is either a (partially evaluated) denotation or a value used for (join) synchronization. The construction $\{\text{Comp}\} = \Pi \times (Id \to \text{Comp})$ was introduced in Section 3. In the sequel $\varphi$ ranges over $Id \to \text{Comp}$. 
We call an element of Kont a closed continuation and an element of Cont an open continuation. A (closed or open) continuation is a representation of what remains to be computed from the program [12]. A closed continuation $\kappa \in Kont$ is a self-contained structure of computations. An open continuation $(\alpha, \kappa) \in Cont$ behaves like an evaluation context [6] for the denotational mapping. In an expression $D(s)(\alpha, \kappa)$, $D(s)$ is the active computation which is evaluated with respect to $(\alpha, \kappa)$. Intuitively, an open continuation $(\alpha, \kappa)$ is a structure of computations which contains a hole, indicating the conceptual position of the active computation. The position of the 'hole' is given in this representation by the identifier $\alpha$, which is not an element of $id(\kappa)$ ($D$ is designed to preserve this invariant property: $\alpha \notin id(\kappa)$ and $\alpha \in \max(\{\alpha\} \cup id(\kappa))$).

The denotational function $D$ is given in Definition 4.1 with the aid of a mapping $kc$, which is called a scheduler. The denotational function maps an open continuation to a program behavior. After producing an elementary step the denotational function transmits the control to the scheduler. The scheduler receives as parameter a closed continuation which it maps to a corresponding program behavior. If the continuation is empty the scheduler terminates the computation. If the continuation is not empty but there is no possibility to continue the computation then the scheduler detects a deadlock. Otherwise the scheduler can either activate a computation (denotation) or it can perform a join synchronization. In order to activate a computation the scheduler decomposes a closed continuation into a computation and a corresponding open continuation and then it executes the former with the latter as continuation. The activation of a computation can follow after zero or more (a finite number of) join synchronization steps.

We introduce three auxiliary mappings that are used by the scheduler. A schedule is a finite and non-empty set of identifiers. A schedule with only 1 element is an activation schedule; it contains exactly one identifier $\alpha$, and the computation with identifier $\alpha$ is activated by the scheduler function $kc$. A schedule that contains more than 1 element is a synchronization schedule; it is used to define a pattern matching synchronization on multiple channels. Activation schedules are computed by the mapping $\Omega_A$; synchronization schedules are computed by the mapping $\Omega_S$. For all $\kappa \in Kont$ we have $\Omega_A(\kappa) \cap \Omega_S(\kappa) = \emptyset$.

Let $(\varsigma \in) Sched = P_{fin}(Id)$ be the set of schedules. We use different symbols to represent typical elements of the sets $(\varsigma \in) Sched$ and $(\pi \in) \Pi$ so it is always clear when a (non-empty and) finite set of identifiers is treated as a schedule.³ Let $\hat{\cdot} : Sched \to \Pi$, $\hat{\varsigma} = \varsigma$. With the aid of an auxiliary predicate $Sync : (Sched \times Kont) \to Bool$ we define $\Omega, \Omega_A, \Omega_S : Kont \to P_{fin}(Sched)$ as follows:

$$Sync(\{\alpha\}, \kappa) = false$$
$$Sync(\{\alpha, \alpha_1, \ldots, \alpha_n\}, \kappa) = (\kappa(\alpha) = \langle c_1?v_1 \& \cdots \& c_n?v_n \rangle \in J^\circ) \land$$
$$\quad (\kappa(\alpha_1) = c_1!\xi_1 \in \Snd) \land \cdots \land (\kappa(\alpha_n) = c_n!\xi_n \in \Snd)$$
$$\Omega(\kappa) = \Omega_A(\kappa) \cup \Omega_S(\kappa)$$

³$\Pi = Sched \cup \{\emptyset\}$; the set $\Pi = P_{fin}(Id)$ was introduced in Section 3.
We use the mapping (· | · ⇒ ·) : (Σ × Π × Kont) → Σ to model the state update effect of a multichannel synchronous communication:

\[
(σ | ς \Rightarrow κ) = \begin{cases} 
[σ | v_1 \mapsto ξ_1(σ) | \cdots | v_n \mapsto ξ_n(σ)] & \text{if } ς \subseteq \text{max}(id(κ)), \text{Sync}(ς, κ), \\
κ(α) = \{α, α_1, \ldots, α_n\}, & ς = ∅, κ = c_1?v_1 \& \cdots \& c_n?v_n, \\
k(α_1) = c_1!ξ_1, \ldots, κ(α_n) = c_n!ξ_n & \text{otherwise}
\end{cases}
\]

We also define the predicates Terminates, Blocks : Kont → Bool by:

Terminates(κ) = (id(κ) = ∅)
Blocks(κ) = (id(κ) ≠ ∅) ∧ (Ω(κ) = ∅)

**DEFINITION 4.1 (Denotational Semantics (D) for MCC)**

(a) Let kc : Kont → Σ → P be given by:

\[kc(κ)(σ) = \begin{cases} 
\text{if } \text{Terminates}(κ) & \text{then } \{λ\} \\
\text{else if } \text{Blocks}(κ) & \text{then } \{δ\} \\
\text{else } & \bigcup_{(α) ∈ Ω_κ} κ(α, κ \setminus \{α\})(σ) \cup \\
& \bigcup_{ς ∈ Ω_ς} (σ \mid ς \Rightarrow κ) \cdot kc(κ \setminus ς)(σ \mid ς \Rightarrow κ)
\end{cases}\]

(b) We define Φ : Sem_D → Sem_D (for (S ∈) Sem_D = Stat → D) by:

\[
\begin{align*}
Φ(S)(\text{skip})(α, κ)(σ) &= σ \cdot kc(κ)(σ) \\
Φ(S)(v := e)(α, κ)(σ) &= σ' \cdot kc(κ)(σ') & \text{where } σ' = [σ | v \mapsto E[e]](σ) \\
Φ(S)(c!e)(α, κ)(σ) &= σ \cdot kc[κ | α \mapsto c!E[e]](σ) \\
Φ(S)(j)(α, κ)(σ) &= σ \cdot kc[κ | α \mapsto \langle j \rangle](σ) \\
Φ(S)(D(x))(α, κ)(σ) &= Φ(S)(D(x))(α, κ)(σ) \\
Φ(S)(s + s_2)(α, κ)(σ) &= Φ(S)(s_1)(α, κ)(σ) \cup Φ(S)(s_2)(α, κ)(σ) \\
Φ(S)(s_1; s_2)(α, κ)(σ) &= Φ(S)(s_1)(α \cdot 1, [κ | α \mapsto S(s_2)])(σ) \\
Φ(S)(s_1 ∥ s_2)(α, κ)(σ) &= Φ(S)(s_1)(α \cdot 1, [κ | α \cdot 2 \mapsto S(s_2)])(σ) \cup Φ(S)(s_2)(α \cdot 2, [κ | α \cdot 1 \mapsto S(s_1)])(σ)
\end{align*}
\]
(c) We put $\mathcal{D} = \text{fix}(\Phi)$. Let $\alpha_0 = \lambda$ and $\kappa_0 = (\emptyset, \varphi_0)$, where $\varphi_0(\alpha) = \mathcal{D}(\text{skip})$, $\forall \alpha \in \text{Id}$. We also define $\mathcal{D}[\cdot]: \text{Stat} \rightarrow \Sigma \rightarrow \text{P}$ by: $\mathcal{D}[s] = \mathcal{D}(s(\alpha_0, \kappa_0))$.

In the case of $\text{MCC}$ a CSC continuation is a tree of computations with active elements at the leaves (the maximal elements with respect to $'\leq'$). In the case of a sequential composition $(s_1; s_2)$ the computations $\mathcal{D}(s_1)$ and $\mathcal{D}(s_2)$ are given the identifiers $\alpha \cdot 1$ and $\alpha$, respectively ($\alpha \cdot 1 > \alpha$). The scheduler function $kc$ gives priority to the computations at the leaves of the tree that represents the continuation. Therefore $\mathcal{D}(s_2)$ will only be evaluated after the completion of the evaluation of $\mathcal{D}(s_1)$. In the case of a parallel composition $(s_1 \parallel s_2)$ the computations $\mathcal{D}(s_1)$ and $\mathcal{D}(s_2)$ are given the identifiers $\alpha \cdot 1$ and $\alpha \cdot 2$, respectively; $\alpha \cdot 1$ and $\alpha \cdot 2$ are incomparable (with respect to $\leq$) therefore the computations $\mathcal{D}(s_1)$ and $\mathcal{D}(s_2)$ are evaluated in an interleaved manner.

It may not be obvious that the scheduler function $kc$ is well-defined, since it occurs in the right-hand side of its definition. The recursive occurrence of $kc$ is preceded by the (contracting) synchronization step "$(\sigma | \zeta \Rightarrow \kappa) \ldots$". We could define $kc$ as fixed-point of an appropriate higher-order contraction. However, it is easier to define $kc$ by induction on the number of communication patterns (of the type $J^{\circ}$) that are contained in the continuation $\kappa$. A continuation is a finite structure of $\text{Comp}$ computations ($\text{Comp} = J^{\circ} + \text{Snd} + \frac{1}{2} \cdot \mathcal{D}$).

After each synchronization step the number of $J^{\circ}$ communication patterns that are contained in a continuation decreases by 1. When the continuation contains no communication pattern then $\Omega_S(\kappa) = \emptyset$. If $\Omega_S(\kappa) = \emptyset$ then the evaluation either terminates, or blocks, or $\Omega_A(\kappa) \neq \emptyset$. If $\Omega_A(\kappa) \neq \emptyset$ then a computation (of the type $\frac{1}{2} \cdot \mathcal{D}$) contained in $\kappa$ is activated for evaluation.

**LEMMA 4.1**

(a) The mapping $kc$ (as introduced in 4.1) is well-defined.

(b) $\forall \kappa_1, \kappa_2 \in \text{Kont}: d(kc(\kappa_1), kc(\kappa_2)) \leq 2 \cdot d(\kappa_1, \kappa_2)$

**LEMMA 4.2** For all $s \in \text{Sem}_D$, $s \in \text{Stat}$, $\alpha \in \text{Id}$, $\kappa \in \text{Kont}$, $\sigma \in \Sigma$:

(a) $\Phi(S)(s)(\alpha, \kappa)(\sigma) \in \text{P}$ (it is well defined).

(b) $\Phi(S)(s)$ is nonexpansive (in $(\alpha, \kappa)$), and

(c) $\Phi$ is $\frac{1}{2}$-contractive (in $S$).

The proofs of 4.1 and 4.2 are omitted. Similar results are given in [14, 13].

**EXAMPLE 4.1** Let $j \in J, s_1, s_2 \in \text{Stat}, j = c_1 \cdot v_1 \& c_2 \cdot v_2$, $s_1 = c_1!1 || c_2!2$ and $s_2 = j; (v := v_1 + v_2 || v := 10)$. Let also $\sigma_1 = [\sigma \mid v_1 \mapsto 1 \mid v_2 \mapsto 2]$, $\sigma_2 = [\sigma_1 \mid v \mapsto 3]$, $\sigma_3 = [\sigma_1 \mid v \mapsto 10]$. ($\sigma, \sigma_1, \sigma_2, \sigma_3 \in \Sigma$).

One can check that $\mathcal{D}[s_1 || s_2](\sigma) = \{\sigma \sigma \sigma \sigma_1 \sigma_2 \sigma_3, \sigma \sigma \sigma \sigma_1 \sigma_3 \sigma_2\}$
5 Concurrency Laws

We present a technique of describing the behaviour of concurrent systems in a denotational model designed with CSC, using a representation of continuations as structures of computations (denotations). For $MCC$ we prove that the semantic operators designed with CSC continuations satisfy the basic concurrency laws mentioned in the introduction. The semantics of nondeterministic choice $'+'$ is given by the standard union operator, which is associative, commutativity and idempotent. Also, it is easy to prove the distributivity of the non-deterministic choice over sequential composition.

$$D((s_1 + s_2); s_3)(\alpha, \kappa)(\sigma)$$

$$= D(s_1 + s_2)(\alpha \cdot 1, [\kappa | \alpha \mapsto D(s_3)])(\sigma)$$

$$= D(s_1)(\alpha \cdot 1, [\kappa | \alpha \mapsto D(s_3)])(\sigma) \cup D(s_2)(\alpha \cdot 1, [\kappa | \alpha \mapsto D(s_3)])(\sigma)$$

$$= D(s_1; s_3)(\alpha, \kappa)(\sigma) \cup D(s_2; s_3)(\alpha, \kappa)(\sigma)$$

$$= D((s_1; s_3) + (s_2; s_3))(\alpha, \kappa)(\sigma)$$

This implies $D((s_1 + s_2); s_3) = D(s_1; s_3 + s_2; s_3)$, because the proof is given for arbitrary $(\alpha, \kappa) \in \text{Cont}$ and $\sigma \in \Sigma$.

But the flexibility provided by continuations comes at a price. The other properties of sequential and parallel composition require more complex arguments. We introduce and use the notion of a resumption. In this paper, a resumption is a structure of $MCC$ statements.\(^4\) A continuation can contain arbitrary values of the type $D$. We prove the properties of sequential and parallel composition for continuations that can be obtained as semantified versions of resumptions, i.e. for continuations that contain only denotations of $MCC$ statements. The function $K$ defined in 5.1(b) maps a resumption to a corresponding continuation. We also introduce a notion of isomorphism over resumptions.

**DEFINITION 5.1**

(a) Let $(\theta \in)\text{Comp} = J^\Diamond \cup \text{Snd} \cup \text{Stat}$. Notice that $J \subseteq \text{Stat}$, but $J^\Diamond \cap \text{Stat} = \emptyset$ (the set $J^\Diamond$ was defined at the beginning of section 4). We define the set of closed resumptions $(k \in)\text{KRes} = \{|\text{Comp}\}|$.\(^5\) We define the set $\text{CRes}$ of open resumptions by:

$$\text{CRes} = \{(\alpha, k) | (\alpha, k) \in (\text{Id} \times \text{KRes}), \alpha \notin \text{id}(k), \alpha \in \text{max} \{|\alpha\| \cup \text{id}(k)\}\}$$

\(^4\)The terminology is similar to the one employed in [3], where the term resumption is used as an operational counterpart of the term continuation. For example, in [3] in the case of a sequential language a resumption may be a list (sequence) of program statements. However, in this technical report we do not present an operational semantics; we only present a denotational semantics.

\(^5\)In this case the construct $\{|\cdot|\}$ is used to define an ordinary set; see the explanation given in the final part of section 3. The sets $J^\Diamond$, $\text{Snd}$ and $\text{Stat}$ are disjoint.
We define the mappings $\mathsf{sync} : \mathsf{Predicate}$ with the aid of an auxiliary predicate $\mathsf{sync} : (\mathsf{Predicate} \times \mathsf{KRes}) \to \mathsf{Bool}$

$$\mathsf{sync}(\{\alpha\}, k) = \mathsf{false}$$
$$\mathsf{sync}(\{\alpha, \alpha_1, \ldots, \alpha_n\}, k) = (k(\alpha) = \langle c_1?v_1 \& \cdots \& c_n?v_n \rangle \in J^\wedge) \wedge$$

$$(k(\alpha_1) = c_1!\xi_1 \in \mathsf{Snd}) \wedge \cdots \wedge (k(\alpha_n) = c_n!\xi_n \in \mathsf{Snd})$$

$$\omega(k) = \omega_A(k) \cup \omega_S(k)$$

$$\omega_A(k) = \{\alpha \mid \alpha \in \max(id(k)), k(\alpha) \in \mathsf{Stat}\}$$

$$\omega_S(k) = \{\varsigma \mid \varsigma \subseteq \max(id(k)), \mathsf{sync}(\varsigma, k)\}$$

We define $(\cdot \mid \cdot \rightarrow \cdot) : (\Sigma \times \mathsf{Predicate} \times \mathsf{KRes}) \to \Sigma$ as follows:

$$(\sigma \mid \varsigma \rightarrow k) = \begin{cases} [\sigma \mid v_1 \mapsto \xi_1(\sigma) \mid \cdots \mid v_n \mapsto \xi_n(\sigma)] & \text{if } \varsigma \subseteq \max(id(k)), \mathsf{sync}(\varsigma, k), \\
\varsigma = \{\alpha, \alpha_1, \ldots, \alpha_n\}, \\
k(\alpha) = \langle c_1?v_1 \& \cdots \& c_n?v_n \rangle, \\
k(\alpha_1) = c_1!\xi_1, \ldots, k(\alpha_n) = c_n!\xi_n & \text{otherwise} \\
\sigma & \text{otherwise} \\
\end{cases}$$

We also define the predicates $\mathsf{terminates}, \mathsf{blocks} : \mathsf{KRes} \to \mathsf{Bool}$ by

$$\mathsf{terminates}(k) = (id(k) = \emptyset)$$

$$\mathsf{blocks}(k) = (id(k) \neq \emptyset) \wedge (\omega(k) = \emptyset)$$

Some properties of these mappings are stated in lemmas 5.6 and 5.7.

**DEFINITION 5.2**

(a) We say that two closed resumptions $k_1, k_2 \in \mathsf{KRes}$ are isomorphic, and we write $k_1 \cong k_2$, if there is a bijection $\mu : id(k_1) \to id(k_2)$ such that:

(i) $\mu(\alpha_1) \leq \mu(\alpha_2) \Leftrightarrow \alpha_1 \leq \alpha_2, \ \forall \alpha_1, \alpha_2 \in id(k_1)$

(ii) $k_2(\mu(\alpha)) = k_1(\alpha), \ \forall \alpha \in id(k_1)$

(b) We say that two open resumptions $(\alpha_1, k_1), (\alpha_2, k_2) \in \mathsf{CRes}$ are isomorphic, and we write $(\alpha_1, k_1) \cong (\alpha_2, k_2)$ iff there is a bijection $\mu : (\{\alpha_1\} \cup id(k_1)) \to (\{\alpha_2\} \cup id(k_2))$ such that:

$^6(\varsigma \in) \mathsf{Sched} = \mathcal{P}_{\mathsf{fin}}(\mathsf{Id})$, see section 4.
(i) $\mu(\alpha_1) = \alpha_2$

(ii) $\mu(\alpha') \leq \mu(\alpha'') \iff \alpha' \leq \alpha''$, $\forall \alpha', \alpha'' \in \{\alpha_1\} \cup \text{id}(k_1)$

(iii) $k_2(\mu(\alpha)) = k_1(\alpha)$, $\forall \alpha \in \text{id}(k_1)$

Obviously, $\forall (\alpha, k) \in CRes : (\alpha, k) \cong (\alpha, k)$ and if $(\alpha_1, k_1), (\alpha_2, k_2) \in CRes$ then $(\alpha_1, k_1) \cong (\alpha_2, k_2) \Rightarrow k_1 \cong k_2$. Also, the following Lemma is easily established.

**Lemma 5.1** For all $k \in \text{KRes}$, $\alpha \in \text{Id}, \theta \in \text{Comp}$:

$$\text{K}[k \mid \alpha \mapsto \theta] = [K(k) \mid \alpha \mapsto \Theta(\theta)]$$

In Corollary 1 we show that any two continuations that correspond to isomorphic resumptions behave the same. This result is obtained by combining Lemma 5.2 with an argument '$\epsilon \leq \frac{1}{2} \cdot \epsilon \Rightarrow \epsilon = 0'$. Lemma 5.2 identifies the property (in this case the isomorphism between resumptions) which is preserved at each computation step. The effect of each computation step is given in the present metric setting by the $\frac{1}{2}$-contracting factor that appears in the statement of Lemma 5.2(b).

In the proof of Lemma 5.2 we use some results presented later in Lemma 5.6 and Lemma 5.7.

**Lemma 5.2**

(a) For all $k_1, k_2 \in \text{KRes}$ with $k_1 \cong k_2$ and $\sigma \in \Sigma$, there exists $\overline{\sigma} \in \text{Stat}$, $(\overline{\alpha_1}, \overline{k_1}), (\overline{\alpha_2}, \overline{k_2}) \in CRes$ with $(\overline{\alpha_1}, \overline{k_1}) \cong (\overline{\alpha_2}, \overline{k_2})$ and $\overline{\sigma} \in \Sigma$ such that:

$$d(kc(K(k_1)(\sigma)), \text{K}(K(k_2))(\sigma)) \leq$$

$$d(D(\overline{\sigma})(\overline{\alpha_1}, \overline{k_1})((\overline{\sigma}), D(\overline{\sigma})(\overline{\alpha_2}, \overline{k_2}))((\overline{\sigma}))$$

(b) For all $s \in \text{Stat}, (\alpha_1, k_1), (\alpha_2, k_2) \in CRes$ with $(\alpha_1, k_1) \cong (\alpha_2, k_2)$ and $\sigma \in \Sigma$, there exists $\overline{\sigma} \in \text{Stat}$, $(\overline{\alpha_1}, \overline{k_1}), (\overline{\alpha_2}, \overline{k_2}) \in CRes$ with $(\overline{\alpha_1}, \overline{k_1}) \cong (\overline{\alpha_2}, \overline{k_2})$ and $\overline{\sigma} \in \Sigma$ such that:

$$d(D(s)(\alpha_1, K(k_1))(\sigma)), D(s)(\alpha_2, K(k_2))(\sigma)) \leq$$

$$\frac{1}{2} \cdot d(D(\overline{\sigma})(\overline{\alpha_1}, K(\overline{k_1}))((\overline{\sigma}), D(\overline{\sigma})(\overline{\alpha_2}, K(\overline{k_2}))((\overline{\sigma}))$$

**Proof:** For Lemma 5.2(a) we distinguish the following subcases:

- **Case terminates($k_1$).** By Lemma 5.7(d) we also have terminates($k_2$). In this case by 5.6(e):

$$d(kc(K(k_1))(\sigma), \text{K}(K(k_2))(\sigma)) = d(\{\lambda\}, \{\lambda\}) = 0$$

- **Case \neg terminates($k_1$) \land blocks($k_1$).** By 5.7(d) and 5.7(e) we also have \neg terminates($k_2$) \land blocks($k_2$). By 5.6(e) and 5.6(f):
\[
d(kc(K(k_1))(\sigma), kc(K(k_2))(\sigma)) = d(\{\delta\}, \{\delta\}) = 0
\]

- Case \(\neg \text{terminates}(k_1) \land \neg \text{blocks}(k_1)\). In this case we also have \(\neg \text{terminates}(k_2)\) and \(\neg \text{blocks}(k_2)\) and \(\omega(k_1) \neq \emptyset\) and \(\omega(k_2) \neq \emptyset\).

\[
d(kc(K(k_1))(\sigma), kc(K(k_2))(\sigma))
\]
\[
= d\left(\bigcup_{\{\alpha\} \in \Omega_A(K(k_1))} K(k_1)(\alpha)(\alpha, K(k_1) \setminus \{\alpha\})(\sigma) \cup \bigcup_{\varsigma \in \Omega_S(K(k_1))} (\sigma | \varsigma \Rightarrow K(k_1)), K(k_1)(\varsigma)(\sigma | \varsigma \Rightarrow K(k_1)) \right) \cup \bigcup_{\{\alpha\} \in \Omega_A(K(k_2))} K(k_2)(\alpha)(\alpha, K(k_2) \setminus \{\alpha\})(\sigma) \cup \bigcup_{\varsigma \in \Omega_S(K(k_2))} (\sigma | \varsigma \Rightarrow K(k_2)), K(k_2)(\varsigma)(\sigma | \varsigma \Rightarrow K(k_2))
\]
\[
[\text{Lemma } 5.6(b), 5.6(c)]
\]
\[
= d\left(\bigcup_{\{\alpha\} \in \Omega_A(k_1)} K(k_1)(\alpha)(\alpha, K(k_1) \setminus \{\alpha\})(\sigma) \cup \bigcup_{\varsigma \in \Omega_S(k_1)} (\sigma | \varsigma \Rightarrow K(k_1)), K(k_1)(\varsigma)(\sigma | \varsigma \Rightarrow K(k_1)) \right) \cup \bigcup_{\{\alpha\} \in \Omega_A(k_2)} K(k_2)(\alpha)(\alpha, K(k_2) \setminus \{\alpha\})(\sigma) \cup \bigcup_{\varsigma \in \Omega_S(k_2)} (\sigma | \varsigma \Rightarrow K(k_2)), K(k_2)(\varsigma)(\sigma | \varsigma \Rightarrow K(k_2))
\]
\[
[\text{Lemma } 5.7(a), 5.7(b), \text{Lemma } 5.6(j)]
\]
\[
\leq \max\{\max\{d(K(k_1)(\alpha)(\alpha, K(k_1) \setminus \{\alpha\})(\sigma), K(k_2)(\mu(\alpha))(\mu(\alpha), K(k_2) \setminus \{\mu(\alpha)\})(\sigma) \mid \{\alpha\} \in \omega_A(k_1)\}, \max\{d((\sigma | \varsigma \Rightarrow k_1)), K(k_1)(\varsigma)(\sigma | \varsigma \Rightarrow k_1) \mid \varsigma \in \omega_S(k_1)\})\}^{(5.2.a.1)}
\]

where \(\mu : id(k_1) \rightarrow id(k_2)\) is a bijection that satisfies the properties given in definition 5.2(a), and we use the notation

\[
\mu(\{\alpha_1, \ldots, \alpha_n\}) \equiv \{\mu(\alpha_1), \ldots, \mu(\alpha_n)\}
\]

for any schedule \(\{\alpha_1, \ldots, \alpha_n\} \in Sched\). We proceed by induction on \(|I(k_1)|\), where

\[
I(k_1) = \{\alpha \mid \alpha \in id(k_1), k_1(\alpha) \in J^\circ\}
\]
If \(|I(k_1)| = 0\) then, by Definition 5.1 and Lemma 5.6, \(\omega_S(k_1) = \emptyset\). In this case, by Lemma 5.6(g) we have

\[(5.2.a.1) = \max \{ d(D(s)(\alpha, K(k_1 \setminus \{\alpha\})))(\sigma), D(s)(\mu(\alpha), K(k_2 \setminus \{\mu(\alpha)\}))(\sigma) \mid \{\alpha\} \in \omega_A(k_1), s = k_1(\alpha) = k_2(\mu(\alpha)) \}\]

As \(k_1 \cong k_2, \forall \{\alpha\} \in \omega_A(k_1): (\alpha, k_1 \setminus \{\alpha\}), (\mu(\alpha), k_2 \setminus \{\mu(\alpha)\}) \in CRes\) and \( (\alpha, k_1 \setminus \{\alpha\}) \cong (\mu(\alpha), k_2 \setminus \{\mu(\alpha)\}) \). In this subcase we conclude that \(\exists \overline{\sigma} \in Stat, \overline{\sigma} \in Id\) such that \(\overline{\sigma} = k_1(\overline{\alpha}) = k_2(\mu(\overline{\alpha}))\), \((\overline{\alpha}, k_1 \setminus \{\overline{\alpha}\}) \cong (\mu(\overline{\alpha}), k_2 \setminus \{\mu(\overline{\alpha})\})\) and:

\[d(kc(K(k_1))(\sigma), kc(K(k_2))(\sigma)) \leq d(D(\overline{\sigma})(\overline{\alpha}, K(k_1 \setminus \{\overline{\alpha}\}))(\sigma), D(\overline{\sigma})(\mu(\overline{\alpha}), K(k_2 \setminus \{\mu(\overline{\alpha})\}))(\sigma))\]

Next we assume that \(|I(k_1)| > 0\). We have \(\omega(k_1) = \omega_A(k_1) \cup \omega_S(k_1)\). By Lemma 5.6(b) and 5.6(c) \(\omega_A(k_1) = \Omega_A(K(k_1)), \omega_S(k_1) = \Omega_S(K(k_1))\). In this case we have:

\[(5.2.a.1) = \]

\[= \max \{\]

\[\max \{ d(K(k_1)(\alpha)(\alpha, K(k_1 \setminus \{\alpha\}))(\sigma), K(k_2)(\mu(\alpha))(\mu(\alpha), K(k_2 \setminus \{\mu(\alpha)\}))(\sigma) \mid \{\alpha\} \in \omega_A(k_1) \}^{(5.2.a.2)}, \]

\[\max \{ d((\sigma \mid \zeta \rightarrow k_1) \cdot kc(K(k_1) \setminus \zeta)(\sigma \mid \zeta \rightarrow k_1),\]

\[\quad \quad \quad \quad (\sigma \mid \zeta \rightarrow k_1) \cdot kc(K(k_2) \setminus \mu(\zeta))(\sigma \mid \zeta \rightarrow k_1)) \mid \zeta \in \omega_S(k_1) \}^{(5.2.a.3)} \}\]

If \((5.2.a.2) \geq (5.2.a.3)\) then we proceed as in the basic case (i.e. \(|I(k_1)| = 0\)) and we obtain the desired result. Now assume that \((5.2.a.2) < (5.2.a.3)\). By using Lemma 5.6(g) we can compute as follows:

\[(5.2.a.3) = \]

\[= \frac{1}{2} \cdot \max \{ d(kc(K(k_1) \setminus \zeta))(\sigma \mid \zeta \rightarrow k_1), kc(K(k_2) \setminus \mu(\zeta))(\sigma \mid \zeta \rightarrow k_1)) \]

\[\mid \zeta \in \omega_S(k_1)\}\]

Notice that \(\forall \zeta \in Sched\)

\[k_1 \setminus \zeta \cong k_2 \setminus \mu(\zeta)\]

and \(|I(k_1 \setminus \zeta)| < |I(k_1)|\), hence, by the induction hypothesis \(\exists \overline{\sigma} \in Stat, \overline{\sigma} \in \Sigma, (\overline{\sigma_1}, k_1), (\overline{\sigma_2}, k_2) \in CRes\) such that \((\overline{\sigma_1}, k_1) \cong (\overline{\sigma_2}, k_2)\) and

\[d(kc(K(k_1) \setminus \zeta)(\sigma \mid \zeta \rightarrow k_1), kc(K(k_2) \setminus \mu(\zeta))(\sigma \mid \zeta \rightarrow k_1)) \leq d(D(\overline{\sigma})(\overline{\sigma}, K(k_1))(\sigma), D(\overline{\sigma})(\overline{\sigma}, K(k_2))(\sigma))\]

It follows immediately that \(\exists \overline{\sigma} \in Stat, \overline{\sigma} \in \Sigma, (\overline{\sigma_1}, k_1), (\overline{\sigma_2}, k_2) \in CRes\) such that \((\overline{\sigma_1}, k_1) \cong (\overline{\sigma_2}, k_2)\) and

\[(5.2.a.3) \leq d(D(\overline{\sigma})(\overline{\sigma}, K(k_1))(\sigma), D(\overline{\sigma})(\overline{\sigma}, K(k_2))(\sigma))\]

Next we prove Lemma 5.2(b) by induction on \(c(s)\). We treat three subcases.
Case \( s=j \).

\[
d(D(j)(\alpha_1,K(k_1))(\sigma),D(j)(\alpha_2,K(k_2))(\sigma))
\]
\[
= d(\sigma \cdot kc[K(k_1) | \alpha_1 \mapsto \langle j \rangle](\sigma),\sigma \cdot kc[K(k_2) | \alpha_2 \mapsto \langle j \rangle](\sigma)) \quad \text{[Lemma 5.1]}
\]
\[
= \frac{1}{2} \cdot d(kc[K[k_1] | \alpha_1 \mapsto \langle j \rangle])(\sigma),kc(K[k_2] | \alpha_2 \mapsto \langle j \rangle)](\sigma)) \quad (5.2.b.1)
\]
It is easy to check that \((\alpha_1,k_1) \cong (\alpha_2,k_2)\) implies \([k_1 | \alpha_1 \mapsto \langle j \rangle] \cong [k_2 | \alpha_2 \mapsto \langle j \rangle]\). Thus, by Lemma 5.2(a) \( \exists \bar{\sigma} \in \text{Stat}, \sigma \in \Sigma, (\bar{\sigma}_1,\bar{K}_1), (\bar{\sigma}_2,\bar{K}_2) \in CRes \) such that \((\bar{\sigma}_1,\bar{K}_1) \cong (\bar{\sigma}_2,\bar{K}_2)\) and:

(5.2.b.1) \quad \leq \frac{1}{2} \cdot d(D(\bar{\sigma})(\bar{\sigma}_1,K(\bar{K}_1))(\sigma),D(\bar{\sigma})(\bar{\sigma}_2,K(\bar{K}_2))(\sigma))

Case \( s=x \).

\[
d(D(x)(\alpha_1,K(k_1))(\sigma),D(x)(\alpha_2,K(k_2))(\sigma))
\]
\[
= d(D(D(x))(\alpha_1,K(k_1))(\sigma),D(D(x))(\alpha_2,K(k_2))(\sigma)) \quad (5.2.b.2)
\]
By the induction hypothesis \((c(x) < c(D(x))) \exists \bar{\sigma} \in \text{Stat}, \sigma \in \Sigma, (\bar{\sigma}_1,\bar{K}_1), (\bar{\sigma}_2,\bar{K}_2) \in CRes \) with \((\bar{\sigma}_1,\bar{K}_1) \cong (\bar{\sigma}_2,\bar{K}_2)\) such that:

(5.2.b.2) \quad \leq \frac{1}{2} \cdot d(D(\bar{\sigma})(\bar{\sigma}_1,K(\bar{K}_1))(\sigma),D(\bar{\sigma})(\bar{\sigma}_2,K(\bar{K}_2))(\sigma))

Case \( s=s_1 || s_2 \).

\[
d(D(s_1 || s_2)(\alpha_1,K(k_1))(\sigma),D(s_1 || s_2)(\alpha_2,K(k_2))(\sigma))
\]
\[
= d(D(s_1)(\alpha_1 \cdot 1,[K(k_1) | \alpha_1 \cdot 2 \mapsto D(s_2)])(\sigma) \cup \\
D(s_2)(\alpha_1 \cdot 2,[K(k_1) | \alpha_1 \cdot 1 \mapsto D(s_1)])(\sigma), \\
D(s_1)(\alpha_2 \cdot 1,[K(k_2) | \alpha_2 \cdot 2 \mapsto D(s_2)])(\sigma) \cup \\
D(s_2)(\alpha_2 \cdot 2,[K(k_2) | \alpha_2 \cdot 1 \mapsto D(s_1)])(\sigma)) \quad \text{[Lemma 5.1, 'U' is nonexpansive]}
\]
\[
\leq \max\{d(D(s_1)(\alpha_1 \cdot 1,K[k_1 | \alpha_1 \cdot 2 \mapsto s_2])(\sigma) \cup \\
D(s_1)(\alpha_2 \cdot 1,K[k_2 | \alpha_2 \cdot 2 \mapsto s_2])(\sigma)) \quad (5.2.b.3), \\
d(D(s_2)(\alpha_1 \cdot 2,K[k_1 | \alpha_1 \cdot 1 \mapsto s_1])(\sigma) \cup \\
D(s_2)(\alpha_2 \cdot 2,K[k_2 | \alpha_2 \cdot 1 \mapsto s_1])(\sigma)) \quad (5.2.b.4) \}
\]
It is not difficult to check that \((\alpha_1,k_1) \cong (\alpha_2,k_2)\) implies

\[(\alpha_1 \cdot 1,[k_1 | \alpha_1 \cdot 2 \mapsto s_2]) \cong (\alpha_2 \cdot 1,[k_2 | \alpha_2 \cdot 2 \mapsto s_2])\]

and

\[(\alpha_1 \cdot 2,[k_1 | \alpha_1 \cdot 1 \mapsto s_1]) \cong (\alpha_2 \cdot 2,[k_2 | \alpha_2 \cdot 1 \mapsto s_1]).\]

For example, in the first case we can use a bijection that maps \(\alpha_1 \cdot 1\) to \(\alpha_2 \cdot 1\) and \(\alpha_1 \cdot 2\) to \(\alpha_2 \cdot 2\). Therefore, as \(c(s_1) < c(s_1 || s_2)\) we can use the induction hypothesis for \((5.2.b.3)\) and we infer that \(\exists \bar{\sigma} \in \text{Stat}, \sigma \in \Sigma, (\bar{\sigma}_1,\bar{K}_1), (\bar{\sigma}_2,\bar{K}_2) \in CRes\) such that \((\bar{\sigma}_1,\bar{K}_1) \cong (\bar{\sigma}_2,\bar{K}_2)\) and:
(5.2.b.3) \[ \leq \frac{1}{2} \cdot d(D(\overline{x}, K(\overline{1})), \overline{\sigma}), D(\overline{x}, K(\overline{2})), \overline{\sigma}) \]

The expression (5.2.b.4) can be handled in a similar manner and the desired result follows immediately.

Corollary 1

(a) For all \( s \in Stat, (\alpha_1, k_1), (\alpha_2, k_2) \in CRes \) with \((\alpha_1, k_1) \cong (\alpha_2, k_2), \sigma \in \Sigma:\)

\[ D(s)(\alpha_1, K(k_1))(\sigma) = D(s)(\alpha_2, K(k_2))(\sigma) \]

(b) For all \( k_1, k_2 \in KRes \) with \( k_1 \cong k_2 \) and \( \sigma \in \Sigma: \)

\[ kc(K(k_1))(\sigma) = kc(K(k_2))(\sigma) \]

Proof: Let \((w \in) W_\Sigma = Stat \times CRes \times CRes \times \Sigma.\) For \((s, (\alpha_1, k_1), (\alpha_2, k_2), \sigma) \in W_\Sigma\) we use the notation:

\[ \epsilon_D(s, (\alpha_1, k_1), (\alpha_2, k_2), \sigma) \overset{\text{not}}{=} d(D(s)(\alpha_1, K(k_1))(\sigma), D(s)(\alpha_2, K(k_2))(\sigma)) \]

By lemma 5.2(b) for all \((s, (\alpha_1, k_1), (\alpha_2, k_2), \sigma) \in W_\Sigma\) with \((\alpha_1, k_1) \cong (\alpha_2, k_2)\) there exists \((\overline{x}, (\overline{1}, \overline{k_1}), (\overline{2}, \overline{k_2}), \overline{\sigma}) \in W_\Sigma\) with \((\overline{1}, \overline{k_1}) \cong (\overline{2}, \overline{k_2})\) such that:

\[ \epsilon_D(s, (\alpha_1, k_1), (\alpha_2, k_2), \sigma) \leq \frac{1}{2} \cdot \epsilon_D(\overline{x}, (\overline{1}, \overline{k_1}), (\overline{2}, \overline{k_2}), \overline{\sigma}) \]

This means that we have \( sup_{w \in W_\Sigma} \epsilon_D(w) \leq \frac{1}{2} \cdot sup_{w \in W_\Sigma} \epsilon_D(\overline{w}), \) where \( w = (s, (\alpha_1, k_1), (\alpha_2, k_2), \sigma) \) and \( \overline{w} = (\overline{x}, (\overline{1}, \overline{k_1}), (\overline{2}, \overline{k_2}), \overline{\sigma});\) this implies that \( sup_{w \in W_\Sigma} \epsilon_D(w) = 0, i.e., d(D(s)(\alpha_1, K(k_1))(\sigma), D(s)(\alpha_2, K(k_2))(\sigma)) = 0 \) and thus \( D(s)(\alpha_1, K(k_1))(\sigma) = D(s)(\alpha_2, K(k_2))(\sigma), \)

which proves 1(a). 1(b) follows immediately from Lemma 5.2(a) and Corollary 1(a).

Corollary 2 If \( \theta \in Comp, (\alpha, k) \in CRes, \alpha' \in Id, \alpha' \notin id(k) \) (in general \( \alpha' \neq \alpha \)) then:

(a) \[ D(s)(\alpha, K(k))(\sigma) = D(s)(\alpha, [K(k) | \alpha' \mapsto \Theta(\theta)] \setminus \{\alpha'\})(\sigma) \]

(b) \[ kc(K(k))(\sigma) = kc([K(k) | \alpha' \mapsto \Theta(\theta)] \setminus \{\alpha'\})(\sigma) \]

Proof: This follows easily by using Corollary 1(b) and Lemma 5.1 if we notice that

\[ (\alpha, k) \cong (\alpha, [k | \alpha' \mapsto \theta] \setminus \{\alpha'\}) \]

\[ k \cong [k | \alpha' \mapsto \theta] \setminus \{\alpha'\} \]

\[
\begin{align*}
\text{Lemmas 5.3 and 5.4 can be approached with the same technique that was used in the proofs of Lemma 5.2 and Corollary 1, by establishing appropriate properties that are preserved by the computation steps and by using the } & \epsilon \leq \frac{1}{2} \cdot \epsilon \Rightarrow \epsilon = 0 \text{ argument. In the proofs of 5.3 and 5.4 we use some results presented later in Lemma 5.6 and Lemma 5.7.}
\end{align*}
\]

\textbf{Lemma 5.3 For all } s_0, s_1, s_2 \in Stat, \sigma \in \Sigma, \alpha_0, \alpha \in Id, k \in KRes \text{ such that } (\alpha_0, k) \in CRes, (\alpha, k) \in CRes, \ (\alpha_0 \neq \alpha), \neg(\alpha_0 \leq \alpha) \text{ and } \neg(\alpha \leq \alpha_0), \text{ we have:}
(a) $kc([K(k) | \alpha \mapsto D(s_1 \parallel s_2)])(\sigma) =$

$k(c([K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma)$

(b) $D(s_0)(\alpha_0, [K(k) | \alpha \mapsto D(s_1 \parallel s_2)])(\sigma) =$

$D(s_0)(\alpha_0, [K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma)$

**Proof:** We use the notation $Q(\alpha_0, \alpha)$, for $\alpha_0, \alpha \in Id$, to express the fact that $\alpha_0$ and $\alpha$ are different and incomparable with respect to $\preceq$.

$Q(\alpha_0, \alpha) \not= (\alpha_0 \neq \alpha) \land (\neg (\alpha_0 \leq \alpha)) \land (\neg (\alpha \leq \alpha_0))$

Also, we use the notation

$P_{\parallel}(\alpha_0, \alpha, k) \not= Q(\alpha_0, \alpha) \land ((\alpha_0, k) \in CRes) \land ((\alpha, k) \in CRes)$

$P_{\parallel}$ is the invariant property which is preserved by the computation steps.

We proceed as follows. For 5.3(a) we show that $\forall s_1, s_2 \in Stat, \sigma \in \Sigma, \alpha \in Id, k \in KRes$ with $(\alpha, k) \in CRes$, $\exists s' \in Stat, \sigma' \in \Sigma, \alpha' \in Id, k' \in KRes$ with $P_{\parallel}(\alpha', \alpha, k')$ such that:

$$e_{kc}(k, \alpha, s_1, s_2, \sigma) \not= d(kc([K(k) | \alpha \mapsto D(s_1 \parallel s_2)])(\sigma),$$

$$k(c(K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma))$$

$$\leq d(D(s')(\alpha', [K(k') | \alpha \mapsto D(s_1 \parallel s_2)])(\sigma'),$$

$$D(s')(\alpha', [K(k') | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma')) \quad (5.3.1)$$

Also, for 5.3(b) we show that $\forall s_0, s_1, s_2 \in Stat, \sigma \in \Sigma, \alpha_0, \alpha \in Id, k \in KRes$ such that $P_{\parallel}(\alpha_0, \alpha, k), \exists s' \in Stat, \sigma' \in \Sigma, \alpha' \in Id, k' \in KRes$ with $P_{\parallel}(\alpha', \alpha, k')$ such that:

$$e_{D}(s_0, \alpha_0, k, s_1, s_2, \sigma)$$

$$= d(D(s_0)(\alpha_0, [K(k) | \alpha \mapsto D(s_1 \parallel s_2)])(\sigma),$$

$$D(s_0)(\alpha_0, [K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma))$$

$$\leq \frac{1}{2} \cdot d(D(s')(\alpha', [K(k') | \alpha \mapsto D(s_1 \parallel s_2)])(\sigma'),$$

$$D(s')(\alpha', [K(k') | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma')) \quad (5.3.2)$$

$$\not= e_{D}(s', \alpha', k', s_1, s_2, \sigma')$$

If we put $(w \in) W = Stat \times Id \times CRes \times Id \times Stat \times Stat \times \Sigma$ we infer that:
\[
\sup_{w \in W: p_j(\alpha_0, \alpha, k)} \epsilon_D(w) \leq \frac{1}{2} \cdot \sup_{w' \in W: p_j(\alpha', \alpha, k')} \epsilon_D(w')
\]
where \( w = (s_0, \alpha_0, k, \alpha, s_1, s_2, \sigma) \) and \( w' = (s', \alpha', k', \alpha, s_1, s_2, \sigma') \). However \( \sup_{w \in W: p_j(\alpha_0, \alpha, k)} \epsilon_D(w) = \sup_{w' \in W: p_j(\alpha', \alpha, k')} \epsilon_D(w') \). So we infer that:
\[
\sup_{w \in W: p_j(\alpha_0, \alpha, k)} \epsilon_D(w) = 0
\]
which implies immediately 5.3(b). Next, by using this result and (5.3.1) we obtain immediately 5.3(a). In the sequel we prove (5.3.1) and (5.3.2).

First, we prove (5.3.1). Let \( s_1, s_2 \in \text{Stat}, \sigma \in \Sigma, \alpha \in Id, k \in KRes \) with \( (k, \alpha) \in CRes \). Let also \( \kappa = [K(k) | \alpha \mapsto \mathcal{D}(s_1 \parallel s_2)], \pi = [K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(s_1) | \alpha \cdot 2 \mapsto \mathcal{D}(s_2)], k'' = [k | \alpha \mapsto s_1 \parallel s_2] \) and \( k'' = [k | \alpha \cdot 1 \mapsto s_1 | \alpha \cdot 2 \mapsto s_2] \). By Lemma 5.1 \( \kappa = K(k''), \pi = K(k'') \). We compute as follows:
\[
d(kc(\kappa)(\sigma), kc(\pi)(\sigma))
\]
\[
= d(kc[K(k) | \alpha \mapsto \mathcal{D}(s_1 \parallel s_2)](\sigma),
\]
\[
kc[K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(s_1) | \alpha \cdot 2 \mapsto \mathcal{D}(s_2)](\sigma)
\]
\[
= d(\mathcal{D}(s_1 \parallel s_2)(\alpha, [K(k) | \alpha \mapsto \mathcal{D}(s_1 \parallel s_2)] \{\alpha\})(\sigma) \cup 
\bigcup_{\alpha' \in \Omega(k''_{\alpha \neq \alpha})} K(k'')(\alpha')(K(k'') \\{\alpha'\})(\sigma) \cup 
\bigcup_{\zeta \in \Omega(s)(k'') \ {\zeta}} (\sigma | \zeta \Rightarrow K(k'')) \cdot kc(K(k'') \\{\zeta\})(\sigma) \Rightarrow K(k''))
\]
\[
= \max\{d(\mathcal{D}(s_1 \parallel s_2)(\alpha, K(k))(\sigma),
\]
\[
\mathcal{D}(s_1)(\alpha \cdot 1, [K(k) | \alpha \cdot 2 \mapsto \mathcal{D}(s_2)])(\sigma) \cup 
\mathcal{D}(s_2)(\alpha \cdot 2, [K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(s_1)])(\sigma) \}
\]
\[
(5.3.3),
\]
\[
d(\bigcup_{\alpha' \in \Omega(k''_{\alpha \neq \alpha})} K(k'')(\alpha')(K(k'') \\{\alpha'\})(\sigma),
\]
\[
\bigcup_{\alpha' \in \Omega(k''_{\alpha \neq \alpha})} K(k'')(\alpha')(K(k'') \\{\alpha'\})(\sigma) \}
\]
\[
(5.3.4)
\]
\[
d (\bigcup_{\varsigma \in \Omega_S(K(k''))} (\sigma \mid \varsigma \Rightarrow K(k'')) \cdot kc(K(k'') \setminus \varsigma)) (\sigma \mid \varsigma \Rightarrow K(k'')) ,
\]

\[
\bigcup_{\varsigma \in \Omega_S(K(k''))} (\sigma \mid \varsigma \Rightarrow K(k'')) \cdot kc(K(k'') \setminus \varsigma))(\sigma \mid \varsigma \Rightarrow K(k'')) (5.3.5)
\]
\[
\{\alpha'\} \in \omega_A(k) \land Q(\alpha', \alpha) \Rightarrow K(k')(\alpha') = K(k'')(\alpha') = K(k)(\alpha'),
\]

'\cup' is nonexpansive

\[
= \max\{d(K(k)(\alpha'))(\alpha'), (K[ k | \alpha \mapsto (s_1 || s_2)]) \setminus \{\alpha'\})(\sigma),
\]

\[
K(k)(\alpha')(\alpha'), (K[ k | \alpha \cdot 1 \mapsto s_1 | \alpha \cdot 2 \mapsto s_2]) \setminus \{\alpha'\})(\sigma)
\]

\[| \{\alpha'\} \in \omega_A(k), Q(\alpha', \alpha)\]

\[\{\alpha'\} \in \omega_A(k), (k, k) \in CRes, Q(\alpha', \alpha) \Rightarrow \]

\[\alpha' \neq \alpha, \alpha' \neq \alpha \cdot 1, \alpha' \neq \alpha \cdot 2; \text{Lemma } 5.1, \text{Lemma } 5.6(h), \text{Lemma } 5.6(g)\]

\[
= \max\{d(D(s')(\alpha'), [K(k \setminus \{\alpha'\}) | \alpha \mapsto D(s_1 || s_2)])(\sigma),
\]

\[
D(s')(\alpha', [K(k \setminus \{\alpha'\}) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma)
\]

\[| \{\alpha'\} \in \omega_A(k), Q(\alpha', \alpha), s' = k(\alpha')(5.3.4')\]

As \((\alpha, k) \in CRes\), it is easy to check that \(\forall \{\alpha'\} \in \omega_A(k)\) with \(Q(\alpha', \alpha)\) we have \(P(\alpha', \alpha, k \setminus \{\alpha'\})\). Therefore, by taking the maximum of \((5.3.4')\) we obtain a result needed in the proof of \((5.3.1)\), namely we infer that \(\exists s'_1 \in \text{Stat}, \sigma'_1 \in \Sigma, \alpha'_1 \in \text{Id}, k'_1 \in KRes, (k'_1 = k \setminus \{\alpha'_1\})\) satisfying \(P(\alpha'_1, \alpha, k'_1)\) such that:

\[
d(kc(K(k) | \alpha \mapsto D(s_1 || s_2)))(\sigma),
\]

\[
kc(K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)))(\sigma)
\]

\[
\leq d(D(s'_1)(\alpha'_1, [K(k'_1) | \alpha \mapsto D(s_1 || s_2)])(\sigma'_1),
\]

\[
D(s'_1)(\alpha'_1, [K(k'_1) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma'_1))(5.3.4''')
\]

In order to prove \((5.3.1)\) we proceed by induction on \(|I(k)|\)\(^7\) where:

\[
I(k) = \{\alpha | \alpha \in id(k), k(\alpha) \in J^0\}
\]

If \(|I(k)| = 0\) then by using \((5.3.4''')\) we obtain immediately the desired result \((5.3.1)\), because when \(|I(k)| = 0\) we have \(\omega_S(k) = \emptyset\) which implies also \(\omega_S(k'') = \emptyset\) and \(\omega_S(k''') = \emptyset\).

Next we treat the case when \(|I(k)| > 0\). \(\omega(k) = \omega_A(k) \cup \omega_S(k)\) and, in this case at least one of \(\omega_A(k)\) or \(\omega_S(k)\) is nonempty. Also, by Lemma 5.6(b) and Lemma 5.6(c), \(\omega_A(k) = \Omega_A(K(k))\) and \(\omega_S(k) = \Omega_S(K(k))\). If \((5.3.4) \geq (5.3.5)\) then we can proceed as in the basic case (i.e. \(|I(k)| = 0\)) and we obtain the desired result \((5.3.1)\). If \((5.3.4) < (5.3.5)\) then we compute as follows:

\[
(5.3.5) = d(\bigcup_{\zeta \in \Omega_S(K(k''))}(\sigma | \zeta \Rightarrow K(k'')) \cdot kc(K(k'') \setminus \zeta))(\sigma | \zeta \Rightarrow K(k'')),
\]

\(^7\)Recall that \(|\cdot|\) is the cardinal number of \(\cdot\).
\[
\bigcup_{\varsigma \in \Omega_S(K^\prime)} \left( \sigma \mid \varsigma \Rightarrow K(\overline{k}''') \right) \cdot \text{kc}(K(\overline{k}''') \setminus \varsigma) \left( \sigma \mid \varsigma \Rightarrow K(\overline{k}'') \right)
\]

"\Upsilon" is nonexpansive, \( \Omega_S(K''') = \Omega_S(K'') = \omega_S(k''') = \omega_S(k') \)

\[
= \left\{ \{\alpha_1, \ldots, \alpha_m\} \mid \{\alpha_1, \ldots, \alpha_m\} \in \omega_S(k) \land (\forall 1 \leq i \leq m : Q(\alpha_i, \alpha)) \right\},
\]

Lemma 5.6(j)]

\[
= \max \left\{ d((\sigma \mid \varsigma \Rightarrow k''') \cdot \text{kc}(K''') \setminus \varsigma)((\sigma \mid \varsigma \Rightarrow k'''),
\right.
\]

\[
(\sigma \mid \varsigma \Rightarrow \overline{k}'') \cdot \text{kc}(K(\overline{k}'') \setminus \varsigma)((\sigma \mid \varsigma \Rightarrow \overline{k}''))
\]

\[
\mid \varsigma \in \left\{ \{\alpha_1, \ldots, \alpha_m\} \mid \{\alpha_1, \ldots, \alpha_m\} \in \omega_S(k) \land (\forall 1 \leq i \leq m : Q(\alpha_i, \alpha)) \right\} \}
\]

(5.3.5')

As \((\alpha, k) \in CRes, k''' = [k \mid \alpha \mapsto s_1 || s_2], \overline{k}'' = [k \mid \alpha \cdot 1 \mapsto s_1 \mid \alpha \cdot 2 \mapsto s_2], \) for any

\[
\varsigma \in \left\{ \{\alpha_1, \ldots, \alpha_m\} \mid \{\alpha_1, \ldots, \alpha_m\} \in \omega_S(k), (\forall 1 \leq i \leq m : Q(\alpha_i, \alpha)) \right\} \}
\]

we have:

\[
\sigma_{ck}^{not} = (\sigma \mid \varsigma \Rightarrow k''') = (\sigma \mid \varsigma \Rightarrow \overline{k}'') = (\sigma \mid \varsigma \Rightarrow k)
\]

because \(\forall \{\alpha_1, \ldots, \alpha_m\} \in \omega_S(k) \) such that \(\forall 1 \leq i \leq m : Q(\alpha_i, \alpha), \) we have: \(k'''(\alpha_i) = \overline{k}''(\alpha_i) = k(\alpha_i), \forall 1 \leq i \leq m.\) Hence:

(5.3.5') \[
= \max \left\{ d(\sigma_{ck} \cdot \text{kc}((K[k \mid \alpha \mapsto s_1 || s_2]) \setminus \{\alpha', \alpha'_1, \ldots, \alpha'_n\})(\sigma_{ck}),
\right.
\]

\[
\sigma_{ck} \cdot \text{kc}((K[k \mid \alpha \cdot 1 \mapsto s_1 \mid \alpha \cdot 2 \mapsto s_2]) \setminus \{\alpha', \alpha'_1, \ldots, \alpha'_n\})(\sigma_{ck}))
\]

\[
\mid \{\alpha', \alpha'_1, \ldots, \alpha'_n\} \in \omega_S(k),
\]

\[
\left( Q(\alpha', \alpha) \land (\forall 1 \leq i \leq n : Q(\alpha'_i, \alpha)),
\right.
\]

\[
k(\alpha') = c_1 v_1 \land \cdots \land c_n v_n,
\]

\[
k(\alpha'_1) = c_1 \xi_1, \ldots, k(\alpha'_n) = c_1 ! \xi_n,
\]

\[
\sigma_{ck} = [\sigma \mid v_1 \mapsto \xi_1(\sigma) \mid \cdots \mid v_n \mapsto \xi_n(\sigma)]
\]

\[
\left\{ \{\alpha', \alpha'_1, \ldots, \alpha'_n\} \in \omega_S(k), (\alpha, k) \in CRes \Rightarrow
\right.
\]

\[
\left. \alpha \notin \{\alpha', \alpha'_1, \ldots, \alpha'_n\}, \alpha \cdot 1 \notin \{\alpha', \alpha'_1, \ldots, \alpha'_n\},
\right.
\]

\[
\alpha \cdot 2 \notin \{\alpha', \alpha'_1, \ldots, \alpha'_n\}; \text{Lemma 5.1, Lemma 5.6(h), Lemma 5.6(g)}\]

\[
= \frac{1}{2} \max \left\{ d(\text{kc}[K[k \setminus \{\alpha', \alpha'_1, \ldots, \alpha'_n\}] \mid \alpha \mapsto D(s_1 || s_2)](\sigma_{ck}),
\right.
\]

\[
\text{kc}[K[k \setminus \{\alpha', \alpha'_1, \ldots, \alpha'_n\}] \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \cdot 2 \mapsto D(s_2)](\sigma_{ck}))
\]

\[
\mid \{\alpha', \alpha'_1, \ldots, \alpha'_n\} \in \omega_S(k),
\]
(Q(α', α) ∧ (∀1 ≤ i ≤ n : Q(α_i, α))),

k(α') = c_1 \cdot v_1 \land \cdots \land c_n \cdot v_n,

k(α'_i) = c_1 \cdot ξ_1, \ldots, k(α'_n) = c_n \cdot ξ_n,

σ_{ck} = [σ | v_1 → ξ_1(σ) | \cdots | v_n → ξ_n(σ)] \ (5.3.5''')

Notice that ∀{α', α_1, •••, α_n} ∈ ωS(k) : |I(k\{α', α_1, •••, α_n})| < |I(k)| and (α, k\{α', α_1, •••, α_n}) ∈ CRes. Therefore we can use the induction hypothesis for (5.3.1) and we infer that ∃s' ∈ Stat, α' ∈ Id, σ' ∈ S, k' ∈ KRes such that P_{∥}(α', α, k') and

d(kc[K(k \{α', α_1, •••, α_n})] | α → D(s_1 || s_2]][σ_{ck}),

kc[K(k \{α', α_1, •••, α_n})] | α \cdot 1 → D(s_1) | α \cdot 2 → D(s_2)](σ_{ck})

≤ d(D(σ')).[K(k') | α \cdot 1 → D(s_1) | α \cdot 2 → D(s_2)](σ)

for any ω ∈ ωS(k), ω = {α', α_1, •••, α_n}, Q(α', α), Q(α_i, α), ∀1 ≤ i ≤ n. By taking the maximum over (5.3.5'') it follows immediately that ∃s' ∈ Stat, α' ∈ Id, s' ∈ S, k' ∈ KRes such that P_{∥}(α', α, k') and:

(5.3.5) = (5.3.5'')

≤ d(D(σ_1).[K(k') | α \cdot 1 → D(s_1) | α \cdot 2 → D(s_2)])(σ_1),

D(σ_1)(σ_1, [K(k') | α \cdot 1 → D(s_1) | α \cdot 2 → D(s_2)])(σ_1))

This implies that we have

d(kc[K(k) | α → D(s_1 || s_2)](σ),

kc[K(k) | α \cdot 1 → D(s_1) | α \cdot 2 → D(s_2)](σ))

≤ d(D(σ_1).[K(k') | α → D(s_1) | α \cdot 2 → D(s_2)](σ_1),

D(σ_1)(σ_1, [K(k') | α \cdot 1 → D(s_1) | α \cdot 2 → D(s_2)](σ_1))

which completes the proof of (5.3.1), for any k ∈ KRes, |I(k)| ≥ 0.

Next, we prove (5.3.2). We proceed by induction on c(s_0) using (5.3.1). In the computations given below, by assumption P_{∥}(α_0, α, k).

Case [s_0 = skip]. Simple enough; left to the reader.

Case [s_0 = v := e]. Let σ' = [σ | v → E[e](σ)].

\[ d(D(v := e)(\alpha_0, [K(k) \mid \alpha \mapsto D(s_1 \parallel s_2)])(\sigma), \]
\[ D(v := e)(\alpha_0, [K(k) \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \cdot 2 \mapsto D(s_2)])(\sigma) \]
\[ = d(\bar{\sigma} \cdot kc[K(k) \mid \alpha \mapsto D(s_1 \parallel s_2)](\bar{\sigma}), \]
\[ \bar{\sigma} \cdot kc[K(k) \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \cdot 2 \mapsto D(s_2)](\bar{\sigma})) \]
\[ = \frac{1}{2} \cdot d(kc[K(k) \mid \alpha \mapsto D(s_1 \parallel s_2)](\sigma), \]
\[ kc[K(k) \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \cdot 2 \mapsto D(s_2)](\sigma)) \] (5.3.8)

As \((\alpha, k) \in CRes\), by (5.3.1), \(\exists s' \in Stat, \sigma' \in \Sigma, \alpha' \in Id, k' \in KRes\) such that \(P_p(\alpha', \alpha, k')\) and:
\[ (5.3.8) \leq \frac{1}{2} \cdot d(D(s')(\alpha', [K(k') \mid \alpha \mapsto D(s_1 \parallel s_2)])(\sigma'), \]
\[ D(s')(\alpha', [K(k') \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \cdot 2 \mapsto D(s_2)])(\sigma')) \]

Case \([s_0 = j]\).
\[ d(D(j)(\alpha_0, [K(k) \mid \alpha \mapsto D(s_1 \parallel s_2)])(\sigma), \]
\[ D(j)(\alpha_0, [K(k) \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \cdot 2 \mapsto D(s_2)])(\sigma)) \]
\[ = d(\sigma \cdot kc[K(k) \mid \alpha \mapsto D(s_1 \parallel s_2)](\sigma), \]
\[ \sigma \cdot kc[K(k) \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \cdot 2 \mapsto D(s_2)](\sigma)) \]
\[ [ P_p(\alpha_0, \alpha, k) \Rightarrow \alpha_0 \neq \alpha, \alpha_0 \neq \alpha \cdot 1, \alpha_0 \neq \alpha \cdot 2; \text{Lemma 5.1} ] \]
\[ = \frac{1}{2} \cdot d(kc[K(k) \mid \alpha \mapsto \langle j \rangle \mid \alpha \mapsto D(s_1 \parallel s_2)](\sigma), \]
\[ kc[K(k) \mid \alpha \mapsto \langle j \rangle \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \cdot 2 \mapsto D(s_2)](\sigma) \] (5.3.9)

Notice that \(P_p(\alpha_0, \alpha, k)\) implies \((\alpha, [k \mid \alpha_0 \mapsto \langle j \rangle]) \in CRes\), therefore, by (5.3.1)
\[ \exists s' \in Stat, \sigma' \in \Sigma, \alpha' \in Id, k' \in KRes\) such that \(P_p(\alpha', \alpha, k')\) and:
\[ (5.3.9) \leq \frac{1}{2} \cdot d(D(s')(\alpha', [K(k') \mid \alpha \mapsto D(s_1 \parallel s_2)])(\sigma'), \]
\[ D(s')(\alpha', [K(k') \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \cdot 2 \mapsto D(s_2)])(\sigma')) \]

Case \([s_0 = c!e]\). Similar to the case \(s_0 = j\).

Case \([s_0 = x]\).
\[ d(D(x)(\alpha_0, [K(k) \mid \alpha \mapsto D(s_1 \parallel s_2)])(\sigma), \]
\[ D(x)(\alpha_0, [K(k) \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \cdot 2 \mapsto D(s_2)])(\sigma)) \]
\[ = d(D(D(x))(\alpha_0, [K(k) \mid \alpha \mapsto D(s_1 \parallel s_2)])(\sigma), \]
\[ D(D(x))(\alpha_0, [K(k) \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \cdot 2 \mapsto D(s_2)])(\sigma)) \] (5.3.10)

By the induction hypothesis \((c(D(x)) < c(x)) \exists s' \in Stat, \sigma' \in \Sigma, \alpha' \in Id, k' \in KRes\) such that \(P_p(\alpha', \alpha, k')\) and:
\[ (5.3.10) \leq \frac{1}{2} \cdot d(D(s')(\alpha', [K(k') \mid \alpha \mapsto D(s_1 \parallel s_2)])(\sigma'), \]
\[D(s')(\alpha', [K(k') | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma')\]

Case \([s_0 = s_0^1 + s_0^2]\). It has any difficulty, and so we skip it.

Case \([s_0 = s_0^1; s_0^2]\).

\[d(D(s_0^1; s_0^2)(\alpha_0, [K(k) | \alpha \mapsto D(s_1 || s_2)])(\sigma), \]

\[\mathcal{D}(s_0^1; s_0^2)(\alpha_0, [K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma))\]

\[= d(D(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \mapsto D(s_1 || s_2) | \alpha_0 \mapsto D(s_0^2)])(\sigma), \]

\[\mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2) | \alpha_0 \mapsto D(s_0^2)])(\sigma))\]

\[-P_\parallel(\alpha_0, \alpha, k) \Rightarrow \alpha_0 \neq \alpha, \alpha_0 \neq \alpha \cdot 1, \alpha_0 \neq \alpha \cdot 2; \text{Lemma 5.1}\]

\[= d(D(s_0^1)(\alpha_0 \cdot 1, [K[k | \alpha_0 \mapsto s_0^2] | \alpha \mapsto D(s_1 || s_2)])(\sigma), \]

\[\mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K[k | \alpha_0 \mapsto s_0^2] | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma))\] (5.3.11)

Notice that \(P_\parallel(\alpha_0, \alpha, k)\) implies \(P_\parallel(\alpha_0 \cdot 1, \alpha, [k | \alpha_0 \mapsto s_0^2]\). Therefore, we can use the induction hypothesis \((c(s_0^1) < c(s_0^1; s_0^2))\) and we infer that \(\exists s' \in \text{Stat}, \sigma' \in \Sigma, \alpha' \in Id, k' \in KRes\) such that \(P_\parallel(\alpha', \alpha, k')\) and

\[(5.3.11) \leq \frac{1}{2} \cdot d(D(s')(\alpha', [K(k') | \alpha \mapsto D(s_1 || s_2)])(\sigma'), \]

\[\mathcal{D}(s')(\alpha', [K(k') | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma')\]

Case \([s_0 = s_0^1 || s_0^2]\).

\[d(D(s_0^1 || s_0^2)(\alpha_0, [K(k) | \alpha \mapsto D(s_1 || s_2)])(\sigma), \]

\[\mathcal{D}(s_0^1 || s_0^2)(\alpha_0, [K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2)])(\sigma))\]

\[= d(D(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \mapsto D(s_1 || s_2) | \alpha_0 \cdot 2 \mapsto D(s_0^2)])(\sigma) \cup \]

\[\mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) | \alpha \mapsto D(s_1 || s_2) | \alpha_0 \cdot 1 \mapsto D(s_0^1)])(\sigma), \]

\[\mathcal{D}(s_0^1)(\alpha_0 \cdot 1, \]

\[K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2) | \alpha_0 \cdot 2 \mapsto D(s_0^2)])(\sigma) \cup \]

\[\mathcal{D}(s_0^2)(\alpha_0 \cdot 2, \]

\[K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2) | \alpha_0 \cdot 1 \mapsto D(s_0^1)])(\sigma))\]

[ ' \cup ' is nonexpansive ]

\[\leq \max\{d(D(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \mapsto D(s_1 || s_2) | \alpha_0 \cdot 2 \mapsto D(s_0^2)])(\sigma), \]

\[\mathcal{D}(s_0^1)(\alpha_0 \cdot 1, \]

\[K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2) | \alpha_0 \cdot 2 \mapsto D(s_0^2)])(\sigma)), \]

\[d(D(s_0^2)(\alpha_0 \cdot 2, [K(k) | \alpha \mapsto D(s_1 || s_2) | \alpha_0 \cdot 1 \mapsto D(s_0^1)])(\sigma), \]

\[\mathcal{D}(s_0^2)(\alpha_0 \cdot 2, \]

\[K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \mapsto D(s_2) | \alpha_0 \cdot 1 \mapsto D(s_0^1)])(\sigma))\}
LEMMA 5.4. For all $s_0, s_1, s_2 \in \text{Stat}, \sigma \in \Sigma, \alpha, \alpha \in Id, k \in KRes$ such that $(\alpha, k) \in CRes, \alpha \notin id(k), \alpha \cdot 1 \notin id(k)$ and $(-\alpha(0 \leq \alpha \cdot 1))$ we have:

(a) $\kappa([K(k) | \alpha \mapsto D(s_1; s_2)](\sigma) =

\kappa([K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \mapsto D(s_2)](\sigma)

(b) $D(s_0)(\alpha_0, [K(k) | \alpha \mapsto D(s_1; s_2)](\sigma) =

$D(s_0)(\alpha_0, [K(k) | \alpha \cdot 1 \mapsto D(s_1) | \alpha \mapsto D(s_2)](\sigma)

Proof: The proof of this Lemma is similar to the proof of Lemma 5.3. In this case the invariant property which is preserved by the computation steps is $P_i$: $P_i(\alpha, k, k) \equiv ((\alpha, k) \in CRes) \land (\alpha \notin id(k)) \land (\alpha \cdot 1 \notin id(k)) \land (-(\alpha(0 \leq \alpha \cdot 1))$.

We proceed as follows. For 5.4(a) we show that $\forall s_1, s_2 \in \text{Stat}, \sigma \in \Sigma, \alpha \in Id, k \in KRes$ such that $\alpha \notin id(k)$ and $\alpha \cdot 1 \notin id(k)$, $\exists s' \in \text{Stat}, \sigma' \in \Sigma, \alpha' \in Id, k' \in KRes$ such that $P_i(\alpha', k, k')$ such that:

$\epsilon_{kc}(k, \alpha, s_1, s_2, \sigma)$.
\[ d(kc[ K(k) \mid \alpha \mapsto D(s_1; s_2)])(\sigma), \]

\[ kc[ K(k) \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \mapsto D(s_2)](\sigma) \]

\[ \leq d(D(s')(\alpha', [ K(k') \mid \alpha \mapsto D(s_1; s_2)])(\sigma'), \]

\[ D(s')(\alpha', [ K(k') \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \mapsto D(s_2)])(\sigma')) (5.4.1) \]

\[ \overset{\text{not.}}{=} \epsilon_D(s', \alpha', k', \alpha, s_1, s_2, \sigma') \]

Also, for 5.4(b) we show that \( \forall s_0, s_1, s_2 \in \text{Stat}, \sigma \in \Sigma, \alpha_0, \alpha \in \text{Id}, k, k' \in KRes \) such that \( P_i(\alpha_0, \alpha, k), \exists s' \in \text{Stat}, \sigma' \in \Sigma, \alpha', \in \text{Id}, k', k' \in KRes \) such that \( P_i(\alpha', \alpha, k) \) and:

\[ \epsilon_D(s_0, \alpha_0, k, \alpha, s_1, s_2, \sigma) \]

\[ = d(D(s_0)(\alpha_0, [ K(k) \mid \alpha \mapsto D(s_1; s_2)])(\sigma), \]

\[ D(s_0)(\alpha_0, [ K(k) \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \mapsto D(s_2)])(\sigma) \]

\[ \leq \frac{1}{2} \cdot d(D(s')(\alpha', [ K(k') \mid \alpha \mapsto D(s_1; s_2)])(\sigma'), \]

\[ D(s')(\alpha', [ K(k') \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \mapsto D(s_2)])(\sigma') (5.4.2) \]

\[ \overset{\text{not.}}{=} \epsilon_D(s', \alpha', k', \alpha, s_1, s_2, \sigma') \]

If we put \( (w \in) W = \text{Stat} \times \text{Id} \times KRes \times \text{Id} \times \text{Stat} \times \Sigma \) we infer that:

\[ \sup_{w \in W : P_i(\alpha_0, \alpha, k)} \epsilon_D(w) \leq \frac{1}{2} \cdot \sup_{w' \in W : P_i(\alpha', \alpha, k')} \epsilon_D(w') \]

where \( w = (s_0, \alpha_0, k, \alpha, s_1, s_2, \sigma) \) and \( w' = (s', \alpha', k', \alpha, s_1, s_2, \sigma') \).

However \( \sup_{w \in W : P_i(\alpha_0, \alpha, k)} \epsilon_D(w) = \sup_{w' \in W : P_i(\alpha', \alpha, k')} \epsilon_D(w') \). So we infer that:

\[ \sup_{w \in W : P_i(\alpha_0, \alpha, k)} \epsilon_D(w) = 0 \]

which implies immediately 5.4(b). Next, by using this result and (5.4.1) we obtain immediately 5.4(a).

As in the case of property (5.3.1) (used in the proof of Lemma 5.3(a)), in the proof of (5.4.1) one can proceed by induction on \(|I(k)|\), where \( I(k) = \{ \alpha \mid \alpha \in \text{Id}(k), k(\alpha) \in J^\Omega \} \).

Here we only give the proof of (5.4.2). We proceed by induction on \( c(s_0) \), using (5.4.1). In the computations given below, we assume \( P_i(\alpha_0, \alpha, k) \), and have three subcases.

Case \([s_0 = j]\).

\[ d(D(j)(\alpha_0, [ K(k) \mid \alpha \mapsto D(s_1; s_2)])(\sigma), \]

\[ D(j)(\alpha_0, [ K(k) \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \mapsto D(s_2)])(\sigma) \]

\[ = d(\sigma \cdot kc[ K(k) \mid \alpha \mapsto D(s_1; s_2) \mid \alpha_0 \mapsto \langle j \rangle ])(\sigma), \]

\[ \sigma \cdot kc[ K(k) \mid \alpha \cdot 1 \mapsto D(s_1) \mid \alpha \mapsto D(s_2) \mid \alpha_0 \mapsto \langle j \rangle ])(\sigma) \]
\[ P_i(\alpha_0, \alpha, k) \Rightarrow \alpha_0 \neq \alpha, \alpha_0 \neq \alpha \cdot 1; \text{Lemma 5.1} \]
\[
= \frac{1}{2} \cdot \: d(kc[K[k | \alpha_0 \mapsto \langle j \rangle ] | \alpha \mapsto \mathcal{D}(s_1; s_2)](\sigma),
\]
\[
kc[K[k | \alpha_0 \mapsto \langle j \rangle ] | \alpha \cdot 1 \mapsto \mathcal{D}(s_1) | \alpha \mapsto \mathcal{D}(s_2)](\sigma) \quad (5.4.3)
\]
\[
P_i(\alpha_0, \alpha, k) \text{ implies } \alpha \notin \text{id([k | \alpha_0 \mapsto \langle j \rangle ]), } \alpha \cdot 1 \notin \text{id([k | \alpha_0 \mapsto \langle j \rangle ])}. \text{ Therefore, by} \quad (5.4.1) \quad \exists s' \in \text{Stat, } \sigma' \in \Sigma, \alpha' \in \text{Id, k' } \in \text{KRes such that } P_i(\alpha', \alpha, k') \text{ and:}
\[
(5.4.3) \quad \leq \frac{1}{2} \cdot \: d(\mathcal{D}(s')(\alpha', [K(k') | \alpha \mapsto \mathcal{D}(s_1; s_2)])(\sigma'),
\]
\[
\mathcal{D}(s')(\alpha', [K(k') | \alpha \cdot 1 \mapsto \mathcal{D}(s_1) | \alpha \mapsto \mathcal{D}(s_2)])(\sigma'))
\]
Case \([s_0 = x] \).
\[
d(\mathcal{D}(x)(\alpha_0, [K(k) | \alpha \mapsto \mathcal{D}(s_1; s_2)])(\sigma),
\]
\[
\mathcal{D}(x)(\alpha_0, [K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(s_1) | \alpha \mapsto \mathcal{D}(s_2)])(\sigma)
\]
\[
= d(\mathcal{D}(D(x))(\alpha_0, [K(k) | \alpha \mapsto \mathcal{D}(s_1; s_2)])(\sigma),
\]
\[
\mathcal{D}(D(x))(\alpha_0, [K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(s_1) | \alpha \mapsto \mathcal{D}(s_2)])(\sigma) \quad (5.4.4)
\]
By the induction hypothesis \((c(D(x)) < c(x)) \exists s' \in \text{Stat, } \sigma' \in \Sigma, \alpha' \in \text{Id, k' } \in \text{KRes such that } P_i(\alpha', \alpha, k') \text{ and:}
\[
(5.4.4) \quad \leq \frac{1}{2} \cdot \: d(\mathcal{D}(s')(\alpha', [K(k') | \alpha \mapsto \mathcal{D}(s_1; s_2)])(\sigma'),
\]
\[
\mathcal{D}(s')(\alpha', [K(k') | \alpha \cdot 1 \mapsto \mathcal{D}(s_1) | \alpha \mapsto \mathcal{D}(s_2)])(\sigma'))
\]
Case \([s_0 = s_0^1; s_0^2] \).
\[
d(\mathcal{D}(s_0^1; s_0^2)(\alpha_0, [K(k) | \alpha \mapsto \mathcal{D}(s_1; s_2)])(\sigma),
\]
\[
\mathcal{D}(s_0^1; s_0^2)(\alpha_0, [K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(s_1) | \alpha \mapsto \mathcal{D}(s_2)])(\sigma)
\]
\[
= d(\mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \mapsto \mathcal{D}(s_1; s_2)] | \alpha \mapsto \mathcal{D}(s_0^2)])(\sigma),
\]
\[
\mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(s_1) | \alpha \mapsto \mathcal{D}(s_2)] | \alpha \mapsto \mathcal{D}(s_0^2)])(\sigma) \quad (5.4.5)
\]
\[
P_i(\alpha_0, \alpha, k) \text{ implies } P_i(\alpha_0 \cdot 1, \alpha, [k | \alpha_0 \mapsto s_0^2]). \text{ Therefore, we can use the induction hypothesis } (c(s_0^1) < c(s_0^1; s_0^2)) \text{ and we infer that } \exists s' \in \text{Stat, } \sigma' \in \Sigma, \alpha' \in \text{Id, k' } \in \text{KRes such that } P_i(\alpha', \alpha, k') \text{ and}
\[
(5.4.5) \quad \leq \frac{1}{2} \cdot \: d(\mathcal{D}(s')(\alpha', [K(k') | \alpha \mapsto \mathcal{D}(s_1; s_2)])(\sigma'),
\]
\[
\mathcal{D}(s')(\alpha', [K(k') | \alpha \cdot 1 \mapsto \mathcal{D}(s_1) | \alpha \mapsto \mathcal{D}(s_2)])(\sigma'))
\]

\[ \square \]

**Lemma 5.5** For all \(s_0, s_1, s_2 \in \text{Stat, } \sigma \in \Sigma, \alpha_0, \alpha \in \text{Id, k } \in \text{KRes such that}
\]
\((\alpha_0, k) \in \text{CRes, } \alpha \notin \text{id(k) and } (\neg (\alpha_0 \leq \alpha)) \text{ we have:}
\]
\[ (a) \: kc([K(k) | \alpha \mapsto \mathcal{D}(s_1 + s_2)])(\sigma) \]
\[
= \kappa_c([K(k) \mid \alpha \mapsto D(s_1)])(\sigma) \cup \kappa_c([K(k) \mid \alpha \mapsto D(s_2)])(\sigma)
\]

(b) \(D(s_0)([\alpha_0, [K(k) \mid \alpha \mapsto D(s_1 + s_2)]](\sigma)\)

\[
= D(s_0)([\alpha_0, [K(k) \mid \alpha \mapsto D(s_1)]](\sigma) \cup D(s_0)([\alpha_0, [K(k) \mid \alpha \mapsto D(s_2)]](\sigma))
\]

**Proof:** First we introduce the invariant property which is preserved by the computation steps.

\[
P_{+}(\alpha_0, \alpha, k) \overset{\text{not.}}{=} ((\alpha_0, k) \in CRes) \land (\alpha \notin \text{id}(k)) \land (\neg(\alpha_0 \leq \alpha))
\]

For 5.5(a) we show that \(\forall s_1, s_2 \in \text{Stat}, \sigma \in \Sigma, \alpha \in Id, k \in KRes\) such that \(\alpha \notin \text{id}(k)\), \(\exists s' \in \text{Stat}, \sigma' \in \Sigma, \alpha' \in Id, k' \in KRes\) such that \(P_{+}(\alpha', \alpha, k')\) and:

\[
\epsilon_{\kappa c}^{i+}(k, \alpha, s_1, s_2, \sigma) \overset{\text{not.}}{=} \kappa c([K(k) \mid \alpha \mapsto D(s_1 + s_2)])(\sigma),
\]

\[
d([K(k) \mid \alpha \mapsto D(s_1 + s_2)])(\sigma) \cup \kappa c([K(k) \mid \alpha \mapsto D(s_2)])(\sigma)
\]

\[
\leq d(D(s')(\alpha', [K(k') \mid \alpha \mapsto D(s_1 + s_2)])(\sigma'),
\]

\[
D(s')(\alpha', [K(k') \mid \alpha \mapsto D(s_1)])(\sigma') \cup D(s')(\alpha', [K(k') \mid \alpha \mapsto D(s_2)])(\sigma')) \overset{\text{5.5.1}}{=} \epsilon_{D}^{i+}(s', \alpha', k', \alpha, s_1, s_2, \sigma')
\]

Also, for 5.5(b) we show that \(\forall s_0, s_1, s_2 \in \text{Stat}, \sigma \in \Sigma, \alpha_0, \alpha \in Id, k \in KRes\) such that \(P_{+}(\alpha_0, \alpha, k)\), \(\exists s' \in \text{Stat}, \sigma' \in \Sigma, \alpha' \in Id, k' \in KRes\) such that \(P_{+}(\alpha', \alpha, k')\) and:

\[
\epsilon_{D}^{i+}(s_0, \alpha_0, k, \alpha, s_1, s_2, \sigma)
\]

\[
d(D(s_0)([\alpha_0, [K(k) \mid \alpha \mapsto D(s_1 + s_2)]](\sigma),
\]

\[
D(s_0)([\alpha_0, [K(k) \mid \alpha \mapsto D(s_1)]](\sigma) \cup D(s_0)([\alpha_0, [K(k) \mid \alpha \mapsto D(s_2)]](\sigma))
\]

\[
\leq \frac{1}{2} \cdot d(D(s')(\alpha', [K(k') \mid \alpha \mapsto D(s_1 + s_2)])(\sigma'),
\]

\[
D(s')(\alpha', [K(k') \mid \alpha \mapsto D(s_1)])(\sigma') \cup D(s')(\alpha', [K(k') \mid \alpha \mapsto D(s_2)])(\sigma')) \overset{\text{5.5.2}}{=} \epsilon_{D}^{i+}(s', \alpha', k', \alpha, s_1, s_2, \sigma')
\]

If we put \((w) \in W = \text{Stat} \times Id \times KRes \times Id \times \text{Stat} \times \text{Stat} \times \Sigma\) we infer that:

\[
\sup_{w \in \text{W} : P_{+}(\alpha_0, \alpha, k)} \epsilon_{D}^{i+}(w) \leq \frac{1}{2} \cdot \sup_{w' \in \text{W} : P_{+}(\alpha', \alpha, k')} \epsilon_{D}^{i+}(w')
\]

where \(w = (s_0, \alpha_0, k, \alpha, s_1, s_2, \sigma)\) and \(w' = (s', \alpha', k', \alpha, s_1, s_2, \sigma')\).

Obviously, \(\sup_{w \in \text{W} : P_{+}(\alpha_0, \alpha, k)} \epsilon_{D}^{i+}(w) = \sup_{w' \in \text{W} : P_{+}(\alpha', \alpha, k')} \epsilon_{D}^{i+}(w').\) We infer that:
\[ \sup_{w \in W, \rho^+(\alpha_0, \alpha, k)} \epsilon^+_{\mathcal{D}}(w) = 0 \]

which implies 5.5(b). Next, by using this result and and (5.5.1) we obtain 5.5(a).

In the proof of (5.5.1) one can proceed by induction on \(|I(k)|\), where \(I(k) = \{\alpha \mid \alpha \in id(k), k(\alpha) \in J^0\}\). One can follow the same line of reasoning as in the proofs of 5.3(a) (5.3.1) and 5.4(a) (5.4.1). Here, we only give the proof of (5.5.2). We proceed by induction on \(c(s_0)\), using (5.5.1). In the computations given below, we assume \(P^+_\downarrow(\alpha_0, \alpha, k)\). We only treat two subcases.

**Case \([s_0 = \text{skip}]\).**

\[ d(\mathcal{D}(\text{skip})(\alpha_0, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1 + s_2)])(\sigma), \]
\[ \mathcal{D}(\text{skip})(\alpha_0, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)])(\sigma) \cup \]
\[ \mathcal{D}(\text{skip})(\alpha_0, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)])(\sigma) \]
\[ = d(\sigma \cdot kc[K(k) \mid \alpha \mapsto \mathcal{D}(s_1 + s_2)])(\sigma), \]
\[ \sigma \cdot kc[K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]
\[ \sigma \cdot kc[K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \]
\[ = d(\sigma \cdot kc[K(k) \mid \alpha \mapsto \mathcal{D}(s_1 + s_2)])(\sigma), \]
\[ \sigma \cdot (kc[K(k) \mid \alpha \mapsto \mathcal{D}(s_1)])(\sigma) \cup \]
\[ kc[K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \]
\[ = \frac{1}{2} \cdot d(kc[K(k) \mid \alpha \mapsto \mathcal{D}(s_1 + s_2)])(\sigma), \]
\[ kc[K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]
\[ kc[K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \]

As \(P^+_\downarrow(\alpha_0, \alpha, k) \Rightarrow \alpha \notin id(k)\), by (5.5.1) \(\exists s' \in \text{Stat}, \sigma' \in \Sigma, \alpha' \in Id, k' \in KRes\) such that \(P^+_\downarrow(\alpha', \alpha, k')\) and:

\[ \frac{1}{2} \cdot d(\mathcal{D}(s')(\alpha', [K(k') \mid \alpha \mapsto \mathcal{D}(s_1 + s_2)])(\sigma'), \]
\[ \mathcal{D}(s')(\alpha', [K(k') \mid \alpha \mapsto \mathcal{D}(s_1)])(\sigma') \cup \]
\[ \mathcal{D}(s')(\alpha', [K(k') \mid \alpha \mapsto \mathcal{D}(s_2)])(\sigma') \]

**Case \([s_0 = s_0^1 \parallel s_0^2]\).**

\[ d(\mathcal{D}(s_0^1 \parallel s_0^2)(\alpha_0, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1 + s_2)])(\sigma), \]
\[ \mathcal{D}(s_0^1 \parallel s_0^2)(\alpha_0, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)])(\sigma) \cup \]
\[ \mathcal{D}(s_0^1 \parallel s_0^2)(\alpha_0, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)])(\sigma) \]
\[ = d(\mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1 + s_2)](\sigma) \cup \]
\[ \mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1 + s_2)](\sigma) \cup \]
\[ \mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]
\[ \mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]

\[ \mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \cup \]
\[ \mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \cup \]

\[ \mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]
\[ \mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \cup \]

\[ \mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]
\[ \mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \cup \]

\[ \mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]
\[ \mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \cup \]

\[ \mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]
\[ \mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \cup \]

\[ \mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]
\[ \mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \cup \]

\[ \mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]
\[ \mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \cup \]

\[ \mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]
\[ \mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \cup \]

\[ \mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]
\[ \mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \cup \]

\[ \mathcal{D}(s_0^1)(\alpha_0 \cdot 1, [K(k) \mid \alpha \mapsto \mathcal{D}(s_1)](\sigma) \cup \]
\[ \mathcal{D}(s_0^2)(\alpha_0 \cdot 2, [K(k) \mid \alpha \mapsto \mathcal{D}(s_2)](\sigma) \cup \]
\[ D(s_0^2)(\alpha_0 \cdot 2, [K(k) | \alpha \mapsto D(s_1) | \alpha_0 \cdot 1 \mapsto D(s_0^1)])(\sigma) \cup D(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \mapsto D(s_1) | \alpha_0 \cdot 2 \mapsto D(s_0^2)])(\sigma) \cup D(s_0^2)(\alpha_0 \cdot 2, [K(k) | \alpha \mapsto D(s_2) | \alpha_0 \cdot 1 \mapsto D(s_0^1)])(\sigma) \]

\[ \cup \] is associative, commutative and non-expansive

\[ \leq \max \{ d(D(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \mapsto D(s_1 + s_2) | \alpha_0 \cdot 2 \mapsto D(s_0^2)])(\sigma), D(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \mapsto D(s_1) | \alpha_0 \cdot 2 \mapsto D(s_0^2)])(\sigma) \cup D(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \mapsto D(s_2) | \alpha_0 \cdot 2 \mapsto D(s_0^2)])(\sigma) \}, \]

\[ \sum \text{ is associative, commutative and non-expansive} \]

\[ = \max \{ d(D(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \mapsto D(s_1 + s_2) | \alpha \mapsto D(s_1 + s_2)])(\sigma), D(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \mapsto D(s_1 + s_2)](\sigma) \cup D(s_0^1)(\alpha_0 \cdot 1, [K(k) | \alpha \mapsto D(s_2)](\sigma) \}, \]

\[ \sum \text{ is associative, commutative and non-expansive} \]

\[ \leq \max \{ \frac{1}{2} \cdot d(D(s_1^1)(\alpha_1', [K(k_1') | \alpha \mapsto D(s_1 + s_2)])(\sigma_1'), D(s_1')(\alpha_1', [K(k_1') | \alpha \mapsto D(s_2)])(\sigma_1') \cup D(s_2')(\alpha_1', [K(k_1') | \alpha \mapsto D(s_1)])(\sigma_2') \}, \]

\[ \sum \text{ is associative, commutative and non-expansive} \]

\[ \leq \max \{ \frac{1}{2} \cdot d(D(s_2^1)(\alpha_2', [K(k_2') | \alpha \mapsto D(s_1 + s_2)])(\sigma_1'), D(s_1')(\alpha_1', [K(k_1') | \alpha \mapsto D(s_2)])(\sigma_1') \cup D(s_2')(\alpha_1', [K(k_1') | \alpha \mapsto D(s_1)])(\sigma_2') \}, \]

\[ \sum \text{ is associative, commutative and non-expansive} \]

\[ \leq \max \{ \sum, \sum \} \text{ we obtain the desired result} \]

\[ \square \]

The next two lemmas are presented without proofs (which are simple enough and can be used by the reader as proof exercises).
LEMMA 5.6 For all $k \in KRes$

(a) $id(k) = id(K(k))$
(b) $\omega_A(k) = \Omega_A(K(k))$
(c) $\omega_S(k) = \Omega_S(K(k))$
(d) $\omega(k) = \Omega(K(k))$
(e) $\text{terminates}(k) = \text{Terminates}(K(k))$
(f) $\text{blocks}(k) = \text{Blocks}(K(k))$

(g) $K(k) \setminus \pi = K(k \setminus \pi)$, for any $\pi \in \Pi$
(h) $\{ k \mid \alpha \mapsto \theta \} \setminus \{ \alpha' \} = \{ k \setminus \{ \alpha' \} \mid \alpha \mapsto \theta \}$, for $\alpha, \alpha' \in Id, \theta \in \text{Comp}$, with $\alpha' \neq \alpha$

(i) $\text{Sync}(\varsigma, K(k)) = \text{sync}(\varsigma, k)$, for any $\varsigma \in \text{Sched}$

(j) $(\sigma \mid \varsigma \Rightarrow K(k)) = (\sigma \mid \varsigma \Rightarrow k)$, for any $\sigma \in \Sigma, \varsigma \in \text{Sched}$.

LEMMA 5.7 Let $k_1, k_2 \in KRes$ be such that $k_1 \cong k_2$. In (a), (b) and (c) $\mu : id(k_1) \rightarrow id(k_2)$ is a bijection that satisfies the properties given in Definition 5.2(a). We have:

(a) $\omega_A(k_2) = \{(\mu(\alpha)) \mid (\alpha) \in \omega_A(k_1)\}$
(b) $\omega_S(k_2) = \{(\mu(\alpha), \mu(\alpha_1), \ldots, \mu(\alpha_n)) \mid (\alpha, \alpha_1, \ldots, \alpha_n) \in \omega_S(k_1), n \geq 1\}$
(c) $\omega(k_2) = \{(\mu(\alpha_1), \ldots, \mu(\alpha_m)) \mid (\alpha_1, \ldots, \alpha_m) \in \omega(k_1), m \geq 1\}$
(d) $\text{terminates}(k_1) = \text{terminates}(k_2)$
(e) $\text{blocks}(k_1) = \text{blocks}(k_2)$

Also, if $\theta \in \text{Comp}, (\alpha_1, k_1), (\alpha_2, k_2) \in CRes, (\alpha_1, k_1) \cong (\alpha_2, k_2)$ then

(f) $[k_1 \mid \alpha_1 \mapsto \theta] \cong [k_2 \mid \alpha_2 \mapsto \theta]$
(g) $(\alpha_1 \cdot 1, [k_1 \mid \alpha_1 \mapsto \theta]) \cong (\alpha_2 \cdot 1, [k_2 \mid \alpha_2 \mapsto \theta])$
(h) $(\alpha_1 \cdot 1, [k_1 \mid \alpha_1 \cdot 2 \mapsto \theta]) \cong (\alpha_2 \cdot 1, [k_2 \mid \alpha_2 \cdot 2 \mapsto \theta])$
(i) $(\alpha_1 \cdot 2, [k_1 \mid \alpha_1 \cdot 1 \mapsto \theta]) \cong (\alpha_2 \cdot 2, [k_2 \mid \alpha_2 \cdot 1 \mapsto \theta])$

If $k_1, k_2 \in KRes$ are such that $k_1 \cong k_2$, $\sigma \in \Sigma, \varsigma \in \omega_S(k_1)$ and $\mu : id(k_1) \rightarrow id(k_2)$ is a bijection that satisfies the properties given in Definition 5.2(a) then

(j) $(\sigma \mid \varsigma \Rightarrow k_1) = (\sigma \mid \mu(\varsigma) \Rightarrow k_2)$
The following lemma shows that parallel composition is commutative and associative and that sequential composition is associative with respect to all isomorphic continuations that contain only denotations of program statements. In Theorem 5.11 we show that these properties are also preserved in arbitrary syntactic contexts (a notion to be introduced below).

**Lemma 5.8** If $s_1, s_2, s_3 \in \text{Stat}, (\alpha, k), (\overline{\alpha}, k) \in \text{CRes}$, $(\alpha, k) \cong (\overline{\alpha}, k)$ and $\sigma \in \Sigma$ then:

(a) $\mathcal{D}(s_1 \parallel s_2)(\alpha, K(k))(\sigma) = \mathcal{D}(s_2 \parallel s_1)(\overline{\alpha}, K(k))(\sigma)$

(b) $\mathcal{D}(s_1 \parallel (s_2 \parallel s_3))(\alpha, K(k))(\sigma) = \mathcal{D}((s_1 \parallel s_2) \parallel s_3)(\overline{\alpha}, K(k))(\sigma)$

(c) $\mathcal{D}(s_1; (s_2; s_3))(\alpha, K(k))(\sigma) = \mathcal{D}((s_1; s_2); s_3)(\overline{\alpha}, K(k))(\sigma)$

(d) $\mathcal{D}(s_1; (s_2 + s_3))(\alpha, K(k))(\sigma) = \mathcal{D}(s_1; s_2 + s_1; s_3)(\overline{\alpha}, K(k))(\sigma)^8$

**Proof:**

(a) $d(\mathcal{D}(s_1 \parallel s_2)(\alpha, K(k))(\sigma), \mathcal{D}(s_2 \parallel s_1)(\overline{\alpha}, K(k))(\sigma))$

$$= d(\mathcal{D}(s_1)(\alpha \cdot 1, [K(k) | \alpha \cdot 2 \mapsto \mathcal{D}(s_2)])(\sigma)) \cup$$

$$\mathcal{D}(s_2)(\alpha \cdot 2, [K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(s_1)])(\sigma),$$

$$\mathcal{D}(s_2)(\overline{\alpha} \cdot 1, [K(\overline{k}) | \overline{\alpha} \cdot 2 \mapsto \mathcal{D}(s_1)])(\sigma) \cup$$

$$\mathcal{D}(s_1)(\overline{\alpha} \cdot 2, [K(\overline{k}) | \overline{\alpha} \cdot 1 \mapsto \mathcal{D}(s_2)])(\sigma)) \quad \text{[Lemma 5.1]}$$

$$= d(\mathcal{D}(s_1)(\alpha \cdot 1, K[k | \alpha \cdot 2 \mapsto s_2])(\sigma)) \quad (5.8.1) \cup$$

$$\mathcal{D}(s_2)(\alpha \cdot 2, K[k | \alpha \cdot 1 \mapsto s_1])(\sigma) \quad (5.8.2),$$

$$\mathcal{D}(s_2)(\overline{\alpha} \cdot 1, K[\overline{k} | \overline{\alpha} \cdot 2 \mapsto s_1])(\sigma) \quad (5.8.3) \cup$$

$$\mathcal{D}(s_1)(\overline{\alpha} \cdot 2, K[\overline{k} | \overline{\alpha} \cdot 1 \mapsto s_2])(\sigma) \quad (5.8.4) \quad \text{['} \cup \text{' is nonexpansive]}$$

$$\leq \max\{d( (5.8.1) , (5.8.4) , d( (5.8.2) , (5.8.3) ) \}$$

It is easy to check that $(\alpha, k) \cong (\overline{\alpha}, k)$ implies

$$(\alpha \cdot 1, [k | \alpha \cdot 2 \mapsto s_2]) \cong (\overline{\alpha} \cdot 2, [\overline{k} | \overline{\alpha} \cdot 1 \mapsto s_2]) \quad \text{and}$$

$$(\alpha \cdot 2, [k | \alpha \cdot 1 \mapsto s_1]) \cong (\overline{\alpha} \cdot 1, [\overline{k} | \overline{\alpha} \cdot 2 \mapsto s_1]).$$

For example, we can take a bijection that maps $\alpha \cdot 1$ to $\overline{\alpha} \cdot 2$ and $\alpha \cdot 2$ to $\overline{\alpha} \cdot 1$. Therefore, by Corollary 1(a) $\max\{d( (5.8.1) , (5.8.4) , d( (5.8.2) , (5.8.3) ) \} = 0$, which implies $\mathcal{D}(s_1 \parallel s_2)(\alpha, K(k))(\sigma) = \mathcal{D}(s_2 \parallel s_1)(\overline{\alpha}, K(k))(\sigma)$.

---

8Recall that $;\!$ binds stronger than $+$, therefore $s_1; s_2 + s_1; s_3$ means $(s_1; s_2) + (s_1; s_3)$. 
(b) Notice that it suffices to prove that \(\forall (\alpha, k) \in CRes:\)

\[
\mathcal{D}(s_1 \| s_2 \| s_3)(\alpha, K(k))(\sigma) = \mathcal{D}(s_1 \| s_2 \| s_3)(\alpha, K(k))(\sigma).
\]

because, by Corollary 1(a): \(\mathcal{D}(s_1 \| s_2 \| s_3)(\alpha, K(k))(\sigma) = \mathcal{D}(s_1 \| s_2 \| s_3)(\overline{\alpha}, K(\overline{k}))(\sigma).\)

We obtain the desired result by showing that:

\[
d(\mathcal{D}(s_1 \| s_2 \| s_3)(\alpha, K(k))(\sigma) \quad (5.8.5), \mathcal{D}(s_1 \| s_2 \| s_3)(\alpha, K(k))(\sigma) \quad (5.8.6) = 0.
\]

We expand (5.8.5) and (5.8.6) as follows

\[
(5.8.5) = \mathcal{D}(s_1)(\alpha \cdot 1, [K(k) \mid \alpha \cdot 2 \mapsto \mathcal{D}(s_2 \| s_3)])(\sigma) \cup \mathcal{D}(s_2)(\alpha \cdot 2, [K(k) \mid \alpha \cdot 1 \mapsto \mathcal{D}(s_1)])(\sigma)
\]

\[
= \mathcal{D}(s_1)(\alpha \cdot 1, [K(k) \mid \alpha \cdot 2 \mapsto \mathcal{D}(s_2 \| s_3)])(\sigma) \quad (5.8.7) \cup \mathcal{D}(s_2)(\alpha \cdot 2, [K(k) \mid \alpha \cdot 1 \mapsto \mathcal{D}(s_1)])(\sigma)
\]

\[
(5.8.6) = \mathcal{D}(s_1 \| s_2)(\alpha \cdot 1, [K(k) \mid \alpha \cdot 2 \mapsto \mathcal{D}(s_3)])(\sigma) \cup \mathcal{D}(s_3)(\alpha \cdot 2, [K(k) \mid \alpha \cdot 1 \mapsto \mathcal{D}(s_1)])(\sigma)
\]

\[
= \mathcal{D}(s_1)(\alpha \cdot 1, [K(k) \mid \alpha \cdot 2 \mapsto \mathcal{D}(s_2)])(\sigma) \cup \mathcal{D}(s_2)(\alpha \cdot 2, [K(k) \mid \alpha \cdot 1 \mapsto \mathcal{D}(s_3)])(\sigma)
\]

\[
(5.8.7) = (5.8.10), (5.8.8) = (5.8.11) \quad \text{and} \quad (5.8.9) = (5.8.12), (5.8.8) =
\]

We prove that (5.8.7) = (5.8.10), (5.8.8) = (5.8.11) and (5.8.9) = (5.8.12).

We prove that (5.8.10) follows easily from Corollary 1(a). Indeed

\[
(5.8.10) = [\text{Lemma 5.1} \quad \mathcal{D}(s_2)(\alpha \cdot 2 \cdot 1, K[k \mid \alpha \cdot 1 \mapsto s_1 \mid \alpha \cdot 2 \cdot 2 \mapsto s_3])(\sigma)
\]

\[
(5.8.11) = [\text{Lemma 5.1} \quad \mathcal{D}(s_2)(\alpha \cdot 1 \cdot 2, K[k \mid \alpha \cdot 2 \mapsto s_3 \mid \alpha \cdot 1 \cdot 1 \mapsto s_1])(\sigma)
\]

and it is easy to check that

\[
(\alpha \cdot 2 \cdot 1, [k \mid \alpha \cdot 1 \mapsto s_1 \mid \alpha \cdot 2 \cdot 2 \mapsto s_3]) \cong (\alpha \cdot 1 \cdot 2, [k \mid \alpha \cdot 2 \mapsto s_3 \mid \alpha \cdot 1 \cdot 1 \mapsto s_1])
\]

This isomorphism can be established by considering the following bijection:

\[
\mu : \{\alpha \cdot 2 \cdot 1, \alpha \cdot 1, \alpha \cdot 2 \cdot 2\} \cup id(k) \to \{\alpha \cdot 1 \cdot 2, \alpha \cdot 2, \alpha \cdot 1 \cdot 1\} \cup id(k)
\]

\[
\mu(\alpha \cdot 2 \cdot 1) = \alpha \cdot 1 \cdot 2
\]

\[
\mu(\alpha \cdot 1) = \alpha \cdot 1 \cdot 1
\]

\[
\mu(\alpha \cdot 2 \cdot 2) = \alpha \cdot 2
\]

\[
(\alpha, k) \in CRes \Rightarrow \alpha \cdot 2 \cdot 1, \alpha \cdot 1, \alpha \cdot 2 \cdot 2, \alpha \cdot 1 \cdot 2, \alpha \cdot 2, \alpha \cdot 1 \cdot 1 \notin id(k).
\]
\[ \mu(\alpha') = \alpha', \text{ for } \alpha' \in \text{id}(k) \]

In order to approach the relationships \((5.8.7) = (5.8.10)\) and \((5.8.9) = (5.8.12)\), we notice that:

\[
\begin{align*}
(\alpha \cdot 1 \cdot 1, [k | \alpha \cdot 2 \mapsto s_3 | \alpha \cdot 1 \cdot 2 \mapsto s_2]) \cong \\
(\alpha \cdot 1, [k | \alpha \cdot 2 \mapsto s_3 | \alpha \cdot 2 \cdot 1 \mapsto s_2]) \\
(\alpha \cdot 2 \cdot 2, [k | \alpha \cdot 1 \mapsto s_1 | \alpha \cdot 2 \cdot 1 \mapsto s_2]) \cong \\
(\alpha \cdot 1, [k | \alpha \cdot 2 \cdot 1 \mapsto s_1 | \alpha \cdot 2 \cdot 2 \mapsto s_2]) \\
(\alpha \cdot 2, [k | \alpha \cdot 1 \mapsto s_1 \parallel s_2]) \cong (\alpha \cdot 1, [k | \alpha \cdot 2 \mapsto s_1 \parallel s_2])
\end{align*}
\]

Therefore, in order to prove \((5.8.7) = (5.8.10)\) and \((5.8.9) = (5.8.12)\), by Corollary 1(a) and Lemma 5.1 it suffices to show that \(\forall s', s_1, s_2 \in \text{Stat}, \alpha \in \text{Id}, k \in \text{KRes}, \sigma \in \Sigma\) such that \((\alpha, k) \in \text{CRes}\):

\[
\begin{align*}
D(s') (\alpha \cdot 1, [K(k) | \alpha \cdot 2 \mapsto D(s_1 \parallel s_2)])(\sigma) &= \\
D(s') (\alpha \cdot 1, [K(k) | \alpha \cdot 2 \cdot 1 \mapsto D(s_1) | \alpha \cdot 2 \cdot 2 \mapsto D(s_2)])(\sigma)
\end{align*}
\]

This follows immediately from Lemma 5.3(b).

(c) It suffices to prove the following equality

\[ D(s_1; s_2; s_3)(\alpha, K(k)) (\sigma) = D((s_1; s_2); s_3)(\alpha, K(k)) (\sigma). \]

because, by Corollary 1(a): \(D((s_1; s_2); s_3)(\alpha, K(k)) (\sigma) = D((s_1; s_2); s_3)(\overline{\sigma}, \overline{K(k)}) (\sigma)\).

We compute as follows:

\[
\begin{align*}
D(s_1; s_2; s_3)(\alpha, K(k)) (\sigma) &= \\
&= D(s_1)(\alpha \cdot 1, [K(k) | \alpha \mapsto D(s_2; s_3)])(\sigma) \quad \text{[Lemma 5.1]} \\
&= D(s_1)(\alpha \cdot 1, K[k | \alpha \mapsto s_2; s_3])(\sigma) \quad \text{[Lemma 5.13]} \\
\end{align*}
\]

It is easy to check that: \((\alpha \cdot 1, [k | \alpha \mapsto s_2; s_3]) \cong (\alpha \cdot 1 \cdot 1, [k | \alpha \mapsto s_2; s_3])\). Therefore, by 1(a), \((5.8.13) = D(s_1)(\alpha \cdot 1 \cdot 1, K[k | \alpha \mapsto s_2; s_3])(\sigma), \text{ and} \)

\[
\begin{align*}
D(s_1)(\alpha \cdot 1 \cdot 1, K[k | \alpha \mapsto s_2; s_3])(\sigma) &= \\
&= D(s_1)(\alpha \cdot 1 \cdot 1, [K(k) | \alpha \mapsto D(s_2; s_3)])(\sigma) \quad \text{[Lemma 5.1]} \\
&= D(s_1)(\alpha \cdot 1 \cdot 1, [K(k) | \alpha \cdot 1 \mapsto D(s_2) | \alpha \mapsto D(s_3)])(\sigma) \\
&= D(s_1; s_2)(\alpha \cdot 1, [K(k) | \alpha \mapsto D(s_3)])(\sigma) \\
&= D((s_1; s_2); s_3)(\alpha, K(k))(\sigma)
\end{align*}
\]

(d) By Corollary 1(a) \(D(s_1; s_2 + s_1; s_3)(\alpha, K(k))(\sigma) = D(s_1; s_2 + s_1; s_3)(\overline{\sigma}, \overline{K(k)})(\sigma)\). Therefore, it suffices to prove:
We denote by \( \mathcal{D}(s_1; s_2 + s_3)(\alpha, K(k))(\sigma) \) for all \((\alpha, k) \in \Sigma\) the result of substituting \( s \) for \((\cdot)\) in \( C \). Formally, this substitution can be defined inductively: \((\cdot)(s) = s\), \( a(s) = a\), \( x(s) = x\), and \((C_1 \text{op} C_2)(s) = C_1(s) \text{op} C_2(s)\) where \( \text{op} \in \{;, +, \parallel\}\).

Lemma 5.9 shows that program properties are preserved in any syntactic context by all CSC continuations containing only computations denotable by program statements. In the proof of Lemma 5.9 we use some results presented later in Lemma 5.10.

**LEMMA 5.9** If \( s_1, s_2 \in \text{Stat} \) such that \( \mathcal{D}(s_1)(\alpha, K(k))(\sigma) = \mathcal{D}(s_2)(\alpha, K(k))(\sigma) \) for all \((\alpha, k) \in \text{CRes} \) and \( \sigma \in \Sigma \), then we have

\[
\mathcal{D}(C(s_1))(\alpha, K(k))(\sigma) = \mathcal{D}(C(s_2))(\alpha, K(k))(\sigma)
\]

for all \((\alpha, k) \in \text{CRes} \), \( \sigma \in \Sigma \) and all contexts \( C \).

**Proof:** By structural induction on \( C \). Cases \([C = a]\) and \([C = x]\) are trivial; the case \([C = (\cdot)]\) follows immediately by the inductive assumption. We also treat the following subcases.

Case \([C = C_1 + C_2]\)

\[
\mathcal{D}((C_1 + C_2)(s_1))(\alpha, K(k))(\sigma)
\]

\[= \mathcal{D}(C_1(s_1) + C_2(s_1))(\alpha, K(k))(\sigma)\]

\[= \mathcal{D}(C_1(s_1))(\alpha, K(k))(\sigma) \cup \mathcal{D}(C_2(s_1))(\alpha, K(k))(\sigma) \quad \text{[Ind. hyp.]}\]

\[= \mathcal{D}(C_1(s_2))(\alpha, K(k))(\sigma) \cup \mathcal{D}(C_2(s_2))(\alpha, K(k))(\sigma)\]
LEMMA 5.10

If \( s_1, s_2 \in \text{Stat} \) are such that for all \((\alpha, k) \in \text{CRes}, \sigma \in \Sigma\):

\[
\mathcal{D}(C_1(s_2) + C_2(s_2))(\alpha, K(k))(\sigma) \quad = \quad \mathcal{D}((C_1 + C_2)(s_2))(\alpha, K(k))(\sigma)
\]

Case \([C = C_1 \parallel C_2]\). Let \((\alpha, k) \in \text{CRes}, \sigma \in \Sigma\). We have to prove that:

\[
\mathcal{D}((C_1 \parallel C_2)(s_1))(\alpha, K(k))(\sigma) = \mathcal{D}((C_1 \parallel C_2)(s_2))(\alpha, K(k))(\sigma)
\]

We compute as follows:

\[
\mathcal{D}((C_1 \parallel C_2)(s_1))(\alpha, K(k))(\sigma) = \mathcal{D}(C_1(s_1) \parallel C_2(s_1))(\alpha, K(k))(\sigma)
\]

\[
= \mathcal{D}(C_1(s_1))(\alpha \cdot 1, [K(k) | \alpha \cdot 2 \mapsto \mathcal{D}(C_2(s_1))])(\sigma) \quad \text{[Lemma 5.1]}
\]

\[
= \mathcal{D}(C_1(s_1))(\alpha, K(k)), \quad \text{[Ind. hyp.]} \quad \text{\((5.9.1)\)}
\]

\[
= \mathcal{D}(C_2(s_1))(\alpha \cdot 1, K(k) | \alpha \cdot 2 \mapsto \mathcal{D}(C_1(s_1)))(\sigma) \quad \text{[Lemma 5.1]}
\]

\[
= \mathcal{D}(C_2(s_1))(\alpha \cdot 2, [K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(C_1(s_1))])(\sigma) \quad \text{\((5.9.2)\)}
\]

We handle \((5.9.1)\) first.

\[
= \mathcal{D}(C_1(s_1))(\alpha \cdot 1, [K(k) | \alpha \cdot 2 \mapsto \mathcal{D}(C_2(s_1))])(\sigma) \quad \text{[Lemma 5.1]}
\]

\[
= \mathcal{D}(C_1(s_1))(\alpha \cdot 1, K(k) | \alpha \cdot 2 \mapsto \mathcal{D}(C_2(s_1)))(\sigma) \quad \text{[Ind. hyp.]} \quad \text{\((5.9.1)\)}
\]

\[
= \mathcal{D}(C_2(s_1))(\alpha \cdot 1, K(k) | \alpha \cdot 2 \mapsto \mathcal{D}(C_1(s_1)))(\sigma) \quad \text{[Lemma 5.1]}
\]

\[
= \mathcal{D}(C_2(s_1))(\alpha \cdot 2, [K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(C_1(s_1))])(\sigma) \quad \text{\((5.9.2)\)}
\]

\((\alpha, k) \in \text{CRes} \) implies \((\alpha \cdot 1, k) \in \text{CRes} \). Hence, by the induction hypothesis \(\mathcal{D}(C_2(s_1))(\alpha \cdot 1, K(k))(\sigma) = \mathcal{D}(C_2(s_2))(\alpha \cdot 1, K(k))(\sigma)\). By Lemma 5.10(b)

\[
\mathcal{D}(C_1(s_2))(\alpha \cdot 1, [K(k) | \alpha \cdot 2 \mapsto \mathcal{D}(C_2(s_1))])(\sigma)
\]

\[
= \mathcal{D}(C_1(s_2))(\alpha \cdot 1, [K(k) | \alpha \cdot 2 \mapsto \mathcal{D}(C_2(s_1))])(\sigma)
\]

Similarly

\[
\mathcal{D}(C_1(s_2))(\alpha \cdot 2, [K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(C_1(s_1))])(\sigma)
\]

\[
= \mathcal{D}(C_1(s_2))(\alpha \cdot 2, [K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(C_1(s_1))])(\sigma)
\]

Therefore

\[
\mathcal{D}((C_1 \parallel C_2)(s_1))(\alpha, K(k))(\sigma) = \mathcal{D}(C_1(s_2))(\alpha \cdot 1, [K(k) | \alpha \cdot 2 \mapsto \mathcal{D}(C_2(s_2))])(\sigma) \bigcup \mathcal{D}(C_1(s_2))(\alpha \cdot 2, [K(k) | \alpha \cdot 1 \mapsto \mathcal{D}(C_1(s_2))])(\sigma)
\]

\[
= \mathcal{D}(C_1(s_2)) \parallel C_2(s_2))(\alpha, K(k))(\sigma)
\]

\[
= \mathcal{D}((C_1 \parallel C_2)(s_2))(\alpha, K(k))(\sigma)
\]

\(\square\)
\[ \mathcal{D}(s_1)(\alpha, K(k))(\sigma) = \mathcal{D}(s_2)(\alpha, K(k))(\sigma) \]

then

(a) for all \( k \in K\text{Res}, \overline{\alpha} \in \text{Id}, \sigma \in \Sigma \)

\[ kc([ K(k) | \overline{\alpha} \mapsto \mathcal{D}(s_1) ])(\sigma) = kc([ K(k) | \overline{\alpha} \mapsto \mathcal{D}(s_2) ])(\sigma) \]

(b) for all \( s \in \text{Stat}, (\alpha, k) \in C\text{Res}, \sigma \in \Sigma, \overline{\alpha} \in \text{Id}, \neg(\overline{\alpha} \geq \alpha) \):

\[ \mathcal{D}(s)(\alpha, [ K(k) | \overline{\alpha} \mapsto \mathcal{D}(s_1) ])(\sigma) = \mathcal{D}(s)(\alpha, [ K(k) | \overline{\alpha} \mapsto \mathcal{D}(s_2) ])(\sigma) \]

**Proof:** For (a) we show that \( \forall k \in K\text{Res}, \sigma \in \Sigma, \overline{\alpha} \in \text{Id}, \exists s' \in \text{Stat}, \alpha' \in \text{Id}, k' \in K\text{Res}, \neg(\overline{\alpha} \geq \alpha') \) such that:

\[
\epsilon^{C}_{kc}(k, \overline{\alpha}, s_1, s_2, \sigma) \not= d(kc([ K(k) | \overline{\alpha} \mapsto \mathcal{D}(s_1) ])(\sigma), kc([ K(k) | \overline{\alpha} \mapsto \mathcal{D}(s_2) ])(\sigma)) \\
\leq d(\mathcal{D}(s')(\alpha', [ K(k') | \overline{\alpha} \mapsto \mathcal{D}(s_1) ])(\sigma'), \mathcal{D}(s')(\alpha', [ K(k') | \overline{\alpha} \mapsto \mathcal{D}(s_2) ])(\sigma')) \\
\not= \epsilon^{C}_{D}(s', \alpha', k', \overline{\alpha}, s_1, s_2, \sigma') \quad (5.10.1)
\]

For (b) we show that \( \forall s \in \text{Stat}, \alpha \in \text{Id}, k \in K\text{Res}, \overline{\alpha} \in \text{Id}, \sigma \in \Sigma, \neg(\overline{\alpha} \geq \alpha), \exists s' \in \text{Stat}, \alpha' \in \text{Id}, k' \in K\text{Res}, \sigma' \in \Sigma, \neg(\overline{\alpha} \geq \alpha') \) such that:\(^{10}\)

\[
\epsilon^{C}_{D}(s, \alpha, k, \overline{\alpha}, s_1, s_2, \sigma) \\
= d(\mathcal{D}(s)(\alpha, [ K(k) | \overline{\alpha} \mapsto \mathcal{D}(s_1) ])(\sigma), \mathcal{D}(s)(\alpha, [ K(k) | \overline{\alpha} \mapsto \mathcal{D}(s_2) ])(\sigma)) \\
\leq \frac{1}{2} \cdot d(\mathcal{D}(s')(\alpha', [ K(k') | \overline{\alpha} \mapsto \mathcal{D}(s_1) ])(\sigma'), \mathcal{D}(s')(\alpha', [ K(k') | \overline{\alpha} \mapsto \mathcal{D}(s_2) ])(\sigma')) \\
= \frac{1}{2} \cdot \epsilon^{C}_{D}(s', \alpha', k', \overline{\alpha}, s_1, s_2, \sigma') \quad (5.10.2)
\]

If we put \((w) \in W = \text{Stat} \times \text{Id} \times K\text{Res} \times \text{Id} \times \text{Stat} \times \text{Stat} \times \Sigma \) we infer that

\[
\sup_{w \in W:-(\overline{\alpha} \geq \alpha)} \epsilon^{C}_{D}(w) \leq \frac{1}{2} \cdot \sup_{w' \in W:-(\overline{\alpha} \geq \alpha')} \epsilon^{C}_{D}(w')
\]

where \( w = (s, \alpha, k, \overline{\alpha}, s_1, s_2, \sigma), w' = (s', \alpha', k', \overline{\alpha}, s_1, s_2, \sigma'). \) But

\[
\sup_{w \in W:-(\overline{\alpha} \geq \alpha)} \epsilon^{C}_{D}(w) = \sup_{w' \in W:-(\overline{\alpha} \geq \alpha')} \epsilon^{C}_{D}(w')
\]

Therefore we infer \( \sup_{w \in W:-(\overline{\alpha} \geq \alpha)} \epsilon^{C}_{D}(w) = 0 \), which implies Lemma 5.10(b). Next, by using this result and \(^{(5.10.1)}\) we obtain immediately Lemma 5.10(a). In the sequel we prove \(^{(5.10.1)}\) and \(^{(5.10.2)}\).

First, we prove \(^{(5.10.1)}\) (i.e. \( \epsilon^{C}_{kc}(k, \overline{\alpha}, s_1, s_2, \sigma) \leq \epsilon^{C}_{D}(s', \alpha', k', \overline{\alpha}, s_1, s_2, \sigma') \)). Let

\[
\kappa_1 = [ K(k) | \overline{\alpha} \mapsto \mathcal{D}(s_1) ] \\
= K[k | \overline{\alpha} \mapsto \mathcal{D}(s_1)] , \text{ by Lemma 5.1}
\]

\(^{10}\neg(\overline{\alpha} \geq \alpha) \) is the invariant property that is preserved by the computation steps.
\[ \kappa_2 = [ K(k) \mid \bar{\alpha} \mapsto D(s_2) ] \quad (= K[k \mid \bar{\alpha} \mapsto D(s_2)], \text{ by Lemma 5.1}) \]

We compute as follows:

\[ \varepsilon_{\bar{\alpha}}^C(\kappa_1, \bar{\alpha}, s_1, s_2, \sigma) = d(kc(\kappa_1)(\sigma), kc(\kappa_2)(\sigma)) \]

Obviously, \( \neg \text{Terminates}(\kappa_1) \) and \( \neg \text{Terminates}(\kappa_2) \). Notice that:

\[ \Omega(\kappa_1) = \Omega(\kappa_2) \]

\[ = \begin{cases} \omega(k) \cup \{\bar{\alpha}\} \backslash \{\alpha_1, \ldots, \alpha_m\} : \{\alpha_1, \ldots, \alpha_m\} \in \omega(k), \exists 1 \leq i \leq m : \bar{\alpha} \geq \alpha_i \\ \omega(k) \end{cases} \]

We prove \((5.10.1)\) by induction on \(|I(k)|\) where:\(^{11}\)

\[ I(k) = \{ \alpha \mid \alpha \in id(k), k(\alpha) \in J^0 \} \]

We begin with the case \(|I(k)| = 0\). In this case \( \Omega_1(\kappa_1) = \Omega_1(\kappa_2) = \emptyset \), and \( \Omega(\kappa_1) = \Omega_A(\kappa_1) = \Omega_A(\kappa_2) = \Omega(\kappa_2) \neq \emptyset \), which implies \( \neg \text{Blocks}(\kappa_1) \) and \( \neg \text{Blocks}(\kappa_2) \). Two subcases. If \( (\bar{\alpha} \notin (\max(id(k) \cup \{\bar{\alpha}\})) \) then

\[ d(kc(\kappa_1)(\sigma), kc(\kappa_2)(\sigma)) \]

\[ = d(\bigcup_{\{\alpha\} \in \Omega_A(\kappa_1) } \kappa_1(\alpha, \kappa_1 \backslash \{\alpha\})(\sigma), \]

\[ \bigcup_{\{\alpha\} \in \Omega_A(\kappa_2) } \kappa_2(\alpha, \kappa_2 \backslash \{\alpha\})(\sigma) \]

\[ [ \cap \bigcup \text{ is nonexpansive}] \]

\[ \leq \max \{ d(\kappa_1(\alpha, \kappa_1 \backslash \{\alpha\})(\sigma), \kappa_2(\alpha, \kappa_2 \backslash \{\alpha\})(\sigma)) \mid \{\alpha\} \in \omega(k) \} \]

Let \( s_\alpha = k(\alpha), \forall \{\alpha\} \in \omega(k) \). As \( \bar{\alpha} \notin (\max(id(k) \cup \{\bar{\alpha}\)) \) it follows that \( \{\bar{\alpha}\} \notin \omega(k) \). Also, notice that \( \kappa_1 \backslash \{\alpha\} = [ K(k \backslash \{\alpha\}) \mid \bar{\alpha} \mapsto D(s_1) ], \kappa_2 \backslash \{\alpha\} = [ K(k \backslash \{\alpha\}) \mid \bar{\alpha} \mapsto D(s_2) ] \).

Therefore

\[ \max \{ d(\kappa(\alpha, \kappa \backslash \{\alpha\})(\sigma), D(s_\alpha)(\alpha, [ K(k \backslash \{\alpha\}) \mid \bar{\alpha} \mapsto D(s_1) ]) \mid \{\alpha\} \in \omega(k) \} \]

Clearly, this means that \( \exists s' \in \text{Stat}, \alpha' \in Id, k' \in KRes, \sigma' \in \Sigma \) such that:\(^{12}\)

\[ \varepsilon_{\bar{\alpha}}^C(\kappa_1, \bar{\alpha}, s_1, s_2, \sigma) = d(kc(\kappa_1)(\sigma), kc(\kappa_2)(\sigma)) \]

\(^{11}\)Recall that \( | \cdot | \) is the cardinal number of \( \cdot \).

\(^{12}\)More precisely, for some \( \alpha \in Id \) such that \( \{\alpha\} \in \omega(k), s' = s_\alpha, \alpha' = \alpha, k' = k \backslash \{\alpha\}, \sigma' = \sigma \).
\[
\leq d(D(s')(\alpha', [ K(k') | \overline{c} \mapsto D(s_1) ])(\sigma'), D(s')(\alpha', [ K(k') | \overline{c} \mapsto D(s_2) ])(\sigma'))
= \varepsilon_D(s', \alpha', k', \overline{c}, s_1, s_2, \sigma')
\]

Notice that the invariant property \( \neg(\overline{c} \geq \alpha') \) is preserved, because \( \alpha' = \alpha \) for some \( \{\alpha\} \in \omega(k), \) and \( \overline{c} \notin (max(id(k) \cup \{\overline{c}\})). \) Therefore (5.10.1) holds in this (sub)case.

Next, we treat the subcase when \( \overline{c} \in (max(id(k) \cup \{\overline{c}\})). \) In this subcase:

\[
d(kc(\kappa_1)(\sigma), kc(\kappa_2)(\sigma))
= d(\bigcup_{\{\alpha\} \in \omega(k), \neg(\overline{c} \geq \alpha)} \kappa_1(\alpha, \kappa_1 \setminus \{\alpha\}) \cup D(s_1)(\overline{c}, \kappa_1 \setminus \{\overline{c}\})(\sigma),
\quad \bigcup_{\{\alpha\} \in \omega(k), \neg(\overline{c} \geq \alpha)} \kappa_2(\alpha, \kappa_2 \setminus \{\alpha\}) \cup D(s_2)(\overline{c}, \kappa_2 \setminus \{\overline{c}\})(\sigma))
\]

\[\{\cup\} \text{ is nonexpansive, Corollary 2}\]

\[
\leq \max\{d(D(s_1)(\overline{c}, K(k))(\sigma), D(s_2)(\overline{c}, K(k))(\sigma)),
\quad \max\{d(\kappa_1(\alpha, \kappa_1 \setminus \{\alpha\})(\sigma), \kappa_2(\alpha, \kappa_2 \setminus \{\alpha\})(\sigma))
\quad \mid \{\alpha\} \in \omega(k), \neg(\overline{c} \geq \alpha)\}\}^{(5.10.1.2)}
\]

By the assumption of Lemma 5.10, \( d(D(s_1)(\overline{c}, K(k))(\sigma), D(s_2)(\overline{c}, K(k))(\sigma)) = 0. \) So let \( s_\alpha = k(\alpha), \forall\{\alpha\} \in \omega(k), \neg(\overline{c} \geq \alpha). \) For any \( \{\alpha\} \in \omega(k), \) such that \( \neg(\overline{c} \geq \alpha), \quad \kappa_1 \setminus \{\alpha\} = \big[ K(k \setminus \{\alpha\}) \mid \overline{c} \mapsto D(s_1) \big] \) and \( \kappa_2 \setminus \{\alpha\} = \big[ K(k \setminus \{\alpha\}) \mid \overline{c} \mapsto D(s_2) \big]. \) Therefore

\[
\big(5.10.1.2\big) = \max\{d(D(s_\alpha)(\alpha, [ K(k \setminus \{\alpha\}) \mid \overline{c} \mapsto D(s_1) ])(\sigma),
\quad D(s_\alpha)(\alpha, [ K(k \setminus \{\alpha\}) \mid \overline{c} \mapsto D(s_2) ])(\sigma))
\quad \mid \{\alpha\} \in \omega(k), \neg(\overline{c} \geq \alpha)\}\}
\]

Clearly, this means that \( \exists s' \in Stat, \alpha' \in Id, k' \in KRes, \sigma' \in \Sigma \) (more precisely, for some \( \alpha \in Id \) such that \( \{\alpha\} \in \omega(k) \) with \( \neg(\overline{c} \geq \alpha) \), \( s' = s_\alpha, \alpha' = \alpha, k' = k \setminus \{\alpha\}, \sigma' = \sigma \)) such that:

\[
\epsilon_{kc}(k, \overline{c}, s_1, s_2, \sigma)
= d(kc(\kappa_1)(\sigma), kc(\kappa_2)(\sigma))
= d(kc([ K(k) \mid \overline{c} \mapsto D(s_1) ])(\sigma), kc([ K(k) \mid \overline{c} \mapsto D(s_2) ])(\sigma))
\leq d(D(s')(\alpha', [ K(k') \mid \overline{c} \mapsto D(s_1) ])(\sigma'), D(s')(\alpha', [ K(k') \mid \overline{c} \mapsto D(s_2) ])(\sigma'))
= \epsilon_D(s', \alpha', k', \overline{c}, s_1, s_2, \sigma')
\]
Again, notice that \( \neg(\overline{\alpha} \geq \alpha') \) (because \( \alpha' = \alpha \)) therefore the invariant is preserved by the computation and thus (5.10.1) holds in this (sub)case also.

The induction step follows. So assume that \( |I(k)| > 0 \), in which case \( \Omega(\kappa_1) = \Omega_A(\kappa_1) \cup \Omega_S(\kappa_1) \) and \( \Omega(\kappa_2) = \Omega_A(\kappa_2) \cup \Omega_S(\kappa_2) \). In this case at least one of \( \Omega_A(\kappa_1) \) or \( \Omega_S(\kappa_1) \) (respectively \( \Omega_A(\kappa_2) \) or \( \Omega_S(\kappa_2) \)) is nonempty. Also, it is easy to see that \( \text{Blocks}(\kappa_1) = \text{Blocks}(\kappa_2) \), and obviously \( \neg \text{Termiates}(\kappa_1) \) and \( \neg \text{Termiates}(\kappa_2) \). The case when \( \text{Blocks}(\kappa_1) = \text{Blocks}(\kappa_2) = True \) is clear, because \( f_{\kappa_c}(k, \overline{\alpha}, s_1, s_2, \sigma) = 0 \). When \( \neg \text{Blocks}(\kappa_1) \) and \( \neg \text{Blocks}(\kappa_2) \) we compute as follows. We consider two subcases. First, we treat the subcase when \( \overline{\alpha} \notin (\max(id(k) \cup \{\overline{\alpha}\})) \), and recall that in this case \( \Omega(\kappa_1) = \Omega(\kappa_2) = \omega(k) \), and also \( \Omega_A(\kappa_1) = \Omega_A(\kappa_2) = \omega_A(k) \) and \( \Omega_S(\kappa_1) = \Omega_S(\kappa_2) = \omega_S(k) \). We have:

\[
\begin{align*}
d(kc(\kappa_1)(\sigma), kc(\kappa_2)(\sigma)) &= d(U_{\{\alpha\} \in \omega_A(k)} \kappa_1(\alpha)(\alpha, \kappa_1 \setminus \{\alpha\})(\sigma)) \cup \Upsilon_{\kappa \in \omega_S(k)}(\sigma | \varsigma \Rightarrow \kappa_1) \cdot kc(\kappa_1 \setminus \varsigma)(\sigma | \varsigma \Rightarrow \kappa_1), \\
& \Upsilon_{\{\alpha\} \in \omega_A(k)} \kappa_2(\alpha)(\alpha, \kappa_2 \setminus \{\alpha\})(\sigma) \cup \Upsilon_{\kappa \in \omega_S(k)}(\sigma | \varsigma \Rightarrow \kappa_2) \cdot kc(\kappa_2 \setminus \varsigma)(\sigma | \varsigma \Rightarrow \kappa_2))
\end{align*}
\]

[\( 'U' \) is nonexpansive]

\[
\leq \max\{\max\{d(\kappa_1(\alpha)(\alpha, \kappa_1 \setminus \{\alpha\})(\sigma), \kappa_2(\alpha)(\alpha, \kappa_2 \setminus \{\alpha\})(\sigma)) | \alpha \in \omega_A(k)\} \}^{(5.10.1.3)} , \\
\max\{d((\sigma | \varsigma \Rightarrow \kappa_1) \cdot kc(\kappa_1 \setminus \varsigma)(\sigma | \varsigma \Rightarrow \kappa_1), \\
(\sigma | \varsigma \Rightarrow \kappa_2) \cdot kc(\kappa_2 \setminus \varsigma)(\sigma | \varsigma \Rightarrow \kappa_2)) | \varsigma \in \omega_S(k)\} \}^{(5.10.1.4)}
\]

If \( (5.10.1.3) \geq (5.10.1.4) \) then we proceed as in the basic case (i.e. \( |I(k)| = 0 \)) and we obtain the desired result. Next, assume that \( (5.10.1.3) < (5.10.1.4) \). We compute as follows. First, we notice that for any \( \varsigma = \{\alpha, \alpha_1, \cdots, \alpha_n\} \in \omega_S(k) \) we have \( \sigma_{\varsigma k} = (\sigma | \varsigma \Rightarrow \kappa_1) = (\sigma | \varsigma \Rightarrow \kappa_2) \), where \( k(\alpha) = c_1?v_1 \& \cdots \& c_n?v_n = \kappa_1(\alpha) = \kappa_2(\alpha), k(\alpha_i) = \kappa_1(\alpha_i) = \kappa_2(\alpha_i) = c_i!\xi_i, 1 \leq i \leq n \). Hence:

\[
(5.10.1.4) = \max\{d(\sigma_{\varsigma k} \cdot \kappa_1 \setminus \varsigma)(\sigma_{\varsigma k}), \sigma_{\varsigma k} \cdot \kappa_2 \setminus \varsigma)(\sigma_{\varsigma k})) | \varsigma \in \omega_S(k), \\
k(\alpha) = c_1?v_1 \& \cdots \& c_n?v_n, \\
k(\alpha_1) = c_1!\xi_1, \cdots, k(\alpha_n) = c_n!\xi_n,
\]
Now notice that \( \forall \{ \alpha, \alpha_1, \ldots, \alpha_n \} \in \omega_S(k) \) : \(|I(k \setminus \{ \alpha, \alpha_1, \ldots, \alpha_n \})| < |I(k)|\). Therefore, by the induction hypothesis \( \forall \{ \alpha, \alpha_1, \ldots, \alpha_n \} \in \omega_S(k), \exists \bar{\sigma}' \in \text{Stat}, \bar{\sigma}' \in Id, \bar{k}' \in KRes, \bar{\sigma} \in \Sigma, \neg(\bar{\alpha} \geq \bar{\sigma}') \) such that:

\[
d(kc[K(k \setminus \{ \alpha, \alpha_1, \ldots, \alpha_n \})] | \bar{\sigma} \mapsto D(s_1))(\sigma_{ck}),
\]

\[
kc[K(k \setminus \{ \alpha, \alpha_1, \ldots, \alpha_n \})] | \bar{\sigma} \mapsto D(s_2))(\sigma_{ck})
\]

\[
\leq d(D(\bar{\sigma}')(\bar{\sigma}), [K(k') | \bar{\sigma} \mapsto D(s_1)])(\sigma'), D(\bar{\sigma}')(\bar{\sigma}, [K(k') | \bar{\sigma} \mapsto D(s_2)])(\sigma'))
\]

It follows immediately that \( \exists s' \in \text{Stat}, \alpha' \in Id, k' \in KRes, \sigma' \in \Sigma, \neg(\bar{\alpha} \geq \alpha') \), such that:

\[
e_{kc}^{C}(k, \bar{\alpha}, s_1, s_2, \sigma)
\]

\[
= d(kc(\kappa_1')(\sigma), kc(\kappa_2')(\sigma))
\]

\[
\leq (5.10.1.4)
\]

\[
\leq d(D(s')(\alpha', [K(k') | \bar{\sigma} \mapsto D(s_1)])(\sigma'), D(s')(\alpha', [K(k') | \bar{\sigma} \mapsto D(s_2)])(\sigma'))
\]

\[
= e_{D}^{C}(s', \alpha', k', \bar{\alpha}, s_1, s_2, \sigma')
\]

Next, we consider the subcase when \( \bar{\sigma} \in (\max(id(k) \cup \{ \bar{\alpha} \})) \). Let \( P_{\text{sched}} : (\text{Id} \times \text{Sched}) \rightarrow \text{Bool}, P_{\text{sched}}(\bar{\alpha}, \{ \alpha_1, \ldots, \alpha_m \}) = \forall 1 \leq i \leq m : \neg(\bar{\alpha} \geq \alpha_i) \). In this (sub)case we have:

\[
e_{kc}^{C}(k, \bar{\alpha}, s_1, s_2, \sigma) = d(kc(\kappa_1')(\sigma), kc(\kappa_2')(\sigma))
\]

\[
= d(\bigcup_{\alpha} \in \omega_{\lambda}(k), \neg(\bar{\alpha} \geq \alpha) \kappa_1(\alpha)(\alpha, \kappa_1 \setminus \{ \alpha \})(\sigma)) \cup
\]
\[
\mathcal{D}(s_1)(\alpha, \kappa_1 \setminus \{\alpha\})(\sigma) \cup \\
\bigcup_{\kappa \in \omega_S(k), P_sched(\alpha, \kappa)} (\sigma \mid \kappa \Rightarrow \kappa_1) \cdot k\mathcal{C}(\kappa_1 \setminus \kappa)(\sigma \mid \kappa \Rightarrow \kappa_1), \\
\mathcal{D}(s_2)(\alpha, \kappa_2 \setminus \{\alpha\})(\sigma) \cup \\
\bigcup_{\kappa \in \omega_S(k), P_sched(\alpha, \kappa)} (\sigma \mid \kappa \Rightarrow \kappa_2) \cdot k\mathcal{C}(\kappa_2 \setminus \kappa)(\sigma \mid \kappa \Rightarrow \kappa_2))
\]

[\text{U} \text{ is nonexpansive, Corollary 2}]

\[
\leq \max \{d(\mathcal{D}(s_1)(\alpha, K(k))(\sigma), \mathcal{D}(s_2)(\alpha, K(k))(\sigma)), \\
\max \{d(\kappa_1(\alpha) \setminus \{\alpha\})(\sigma), \kappa_2(\alpha) \setminus \{\alpha\})(\sigma) \mid \{\alpha\} \in \omega_A(k), \neg(\alpha \geq \alpha)\} \quad \text{(5.10.1.6)}, \\
\max \{d((\sigma \mid \kappa \Rightarrow \kappa_1) \cdot k\mathcal{C}(\kappa_1 \setminus \kappa)(\sigma \mid \kappa \Rightarrow \kappa_1)), \\
(\sigma \mid \kappa \Rightarrow \kappa_2) \cdot k\mathcal{C}(\kappa_2 \setminus \kappa)(\sigma \mid \kappa \Rightarrow \kappa_2)) \mid \kappa \in \omega_S(k), P_sched(\alpha, \kappa)\} \quad \text{(5.10.1.7)}
\]

By the assumption of Lemma 5.10, \(d(\mathcal{D}(s_1)(\alpha, K(k))(\sigma), \mathcal{D}(s_2)(\alpha, K(k))(\sigma)) = 0\). Next, if \(\text{(5.10.1.6)} \geq \text{(5.10.1.7)}\) then we proceed as in the basic case (i.e. \(|I(k)| = 0\) and we obtain the desired result. Finally, assume that \(\text{(5.10.1.6)} < \text{(5.10.1.7)}\). We compute as follows. First, we notice that for any \(\kappa = \{\alpha, \alpha_1, \cdots, \alpha_n\} \in \omega_S(k)\) we have \(\kappa \Rightarrow \kappa_1 = (\sigma \mid \kappa \Rightarrow \kappa_1) = (\sigma \mid \kappa \Rightarrow \kappa_2)\), where \(\kappa \Rightarrow \kappa_1 = [\sigma \mid v_1 \mapsto \xi_1(\sigma) \cdots \mid v_n \mapsto \xi_n(\sigma)]\), \(k(\alpha) = c_1?v_1\ & \cdots\ & c_n?v_n = \kappa_1(\alpha) = \kappa_2(\alpha), k(\alpha_i) = \kappa_1(\alpha_i) = \kappa_2(\alpha_i) = c_i!\xi_i, 1 \leq i \leq n\). Hence:

\[
\text{(5.10.1.7)} = \max \{d(\kappa \cdot k\mathcal{C}(\kappa_1 \setminus \{\alpha, \alpha_1, \cdots, \alpha_n\})(\sigma), \kappa \cdot k\mathcal{C}(\kappa_2 \setminus \{\alpha, \alpha_1, \cdots, \alpha_n\})(\sigma) \mid \{\alpha, \alpha_1, \cdots, \alpha_n\} \in \omega_S(k), \\
\neg(\alpha \geq \alpha), \forall \ 1 \leq i \leq n: \neg(\alpha_i), \\
k(\alpha) = c_1?v_1\ & \cdots\ & c_n?v_n, \\
k(\alpha_1) = c_1!\xi_1, \cdots, k(\alpha_n) = c_n!\xi_n, \\
\kappa \Rightarrow \kappa_1 = [\sigma \mid v_1 \mapsto \xi_1(\sigma) \cdots \mid v_n \mapsto \xi_n(\sigma)]\}
\]

\[
= \frac{1}{2} \cdot \max \{d(\kappa \cdot k\mathcal{C}(\kappa_1 \setminus \{\alpha, \alpha_1, \cdots, \alpha_n\}), \kappa \cdot k\mathcal{C}(\kappa_2 \setminus \{\alpha, \alpha_1, \cdots, \alpha_n\}) \mid \alpha \mapsto \mathcal{D}(s_1)) \mid (\sigma), \\
\kappa \cdot k\mathcal{C}(\kappa_1 \setminus \{\alpha, \alpha_1, \cdots, \alpha_n\}) \mid \alpha \mapsto \mathcal{D}(s_2)) \mid (\sigma)\}
\]
\[
| \{ \alpha, \alpha_1, \ldots, \alpha_n \} \in \omega_S(k), \\
\neg(\overline{\sigma} \geq \alpha), \forall 1 \leq i \leq n : \neg(\overline{\sigma} \geq \alpha_i), \\
k(\alpha) = c_1 ? v_1 \& \cdots \& c_n ? v_n, \\
k(\alpha_1) = c_1 ! \xi_1, \ldots, k(\alpha_n) = c_n ! \xi_n, \\
\sigma_{\cdot k} = [\sigma \mid v_1 \mapsto \xi_1(\sigma) \mid \cdots \mid v_n \mapsto \xi_n(\sigma) ]
\]

Notice that \( \forall \{ \alpha, \alpha_1, \ldots, \alpha_n \} \in \omega_S(k) : |I(k \setminus \{ \alpha, \alpha_1, \ldots, \alpha_n \})| < |I(k)| \). Therefore, by the induction hypothesis \( \forall \{ \alpha, \alpha_1, \ldots, \alpha_n \} \in \omega_S(k), \exists \overline{\sigma}' \in \text{Stat}, \overline{\alpha} \in \text{Id}, k' \in KRes, \overline{\sigma}' \in \Sigma, \neg(\overline{\alpha} \geq \overline{\alpha}') \), such that:

\[
d(kc([K(k \setminus \{ \alpha, \alpha_1, \ldots, \alpha_n \}) \mid \overline{\alpha} \mapsto D(s_1)])(\sigma_{\cdot k}), \\
kc([K(k \setminus \{ \alpha, \alpha_1, \ldots, \alpha_n \}) \mid \overline{\alpha} \mapsto D(s_2)])(\sigma_{\cdot k})) \\
\leq d(D(\overline{\sigma}')(\overline{\alpha}', [K(k') \mid \overline{\alpha} \mapsto D(s_1)])(\sigma'), D(\overline{\sigma}')(\overline{\alpha}', [K(k') \mid \overline{\alpha} \mapsto D(s_2)])(\sigma')) \\
= e_D(s', \alpha', k', \overline{\alpha}, s_1, s_2, \sigma')
\]

which completes the proof of (5.10.1).

In the sequel we prove (5.10.2), i.e. we show that \( \forall s' \in \text{Stat}, \alpha \in \text{Id}, k' \in KRes, \overline{\alpha} \in \text{Id}, \sigma \in \Sigma \) with \( \neg(\overline{\alpha} \geq \alpha) \), \( \exists \overline{\sigma}' \in \text{Stat}, \alpha' \in \text{Id}, k' \in KRes, \sigma' \in \Sigma \) with \( \neg(\overline{\alpha} \geq \alpha') \) such that:

\[
e_D(s, \alpha, k, \overline{\alpha}, s_1, s_2, \sigma) \leq \frac{1}{2} \cdot e_D(s', \alpha', k', \overline{\alpha}, s_1, s_2, \sigma')
\]

We proceed by induction on \( c(s) \). Three subcases.

Case \( [s = j] \)

\[
d(D(j)(\alpha, [K(k) \mid \overline{\alpha} \mapsto D(s_1)])(\sigma), D(j)(\alpha, [K(k) \mid \overline{\alpha} \mapsto D(s_2)])(\sigma)) \\
= d(\sigma \cdot k( [K(k) \mid \overline{\alpha} \mapsto D(s_1)] \mid \alpha \mapsto \langle j \rangle))(\sigma), \\
\sigma \cdot k( [K(k) \mid \overline{\alpha} \mapsto D(s_2)] \mid \alpha \mapsto \langle j \rangle))(\sigma) \\
[\neg(\overline{\alpha} \geq \alpha)] \\
= \frac{1}{2} \cdot d(kc([K(k) \mid \alpha \mapsto \langle j \rangle] \mid \overline{\alpha} \mapsto D(s_1)])(\sigma), \\
k( [K(k) \mid \alpha \mapsto \langle j \rangle] \mid \overline{\alpha} \mapsto D(s_2)])(\sigma) \quad \text{[Lemma 5.1]}
\]
\[
= \frac{1}{2} \cdot d(kc([K([k \mid \alpha \mapsto \langle j \rangle]) \mid \overline{\alpha} \mapsto D(s_1)])(\sigma),
\]
\[kc([K([k \mid \alpha \mapsto \langle j \rangle]) \mid \overline{\alpha} \mapsto D(s_2)])(\sigma)) \quad (5.10.2.4)
\]
And by (5.10.1), \(\exists s' \in Stat, \alpha' \in Id, k' \in KRes, \sigma' \in \Sigma\) with \(\neg(\overline{\alpha} \geq \alpha')\) such that:
\[\leq \frac{1}{2} \cdot d(D(s')(\alpha', [K(k') \mid \overline{\alpha} \mapsto D(s_1)])(\sigma'),
\]
\[D(s')(\alpha', [K(k') \mid \overline{\alpha} \mapsto D(s_2)])(\sigma')))
\]
Case \([s = x]\)
\[d(D(x)(\alpha, [K(k) \mid \overline{\alpha} \mapsto D(s_1)])(\sigma), D(x)(\alpha, [K(k) \mid \overline{\alpha} \mapsto D(s_2)])(\sigma))
\]
\[= d(D(D(x))(\alpha, [K(k) \mid \overline{\alpha} \mapsto D(s_1)])(\sigma),
\]
\[D(D(x))(\alpha, [K(k) \mid \overline{\alpha} \mapsto D(s_2)])(\sigma)) \quad (5.10.2.5)
\]
By the induction hypothesis \(c(D(x)) < c(x)\), \(\exists s' \in Stat, \alpha' \in Id, k' \in KRes, \sigma' \in \Sigma\) with \(\neg(\overline{\alpha} \geq \alpha')\) such that:
\[\leq \frac{1}{2} \cdot d(D(s')(\alpha', [K(k') \mid \overline{\alpha} \mapsto D(s_1)])(\sigma'),
\]
\[D(s')(\alpha', [K(k') \mid \overline{\alpha} \mapsto D(s_2)])(\sigma')))
\]
Case \([s = s^1 \parallel s^2]\)
\[\epsilon^C_D(s^1 \parallel s^2, \alpha, k, \overline{\alpha}, s_1, s_2, \sigma)
\]
\[= d(D(s^1 \parallel s^2)(\alpha, [K(k) \mid \overline{\alpha} \mapsto D(s_1)])(\sigma),
\]
\[D(s^1 \parallel s^2)(\alpha, [K(k) \mid \overline{\alpha} \mapsto D(s_2)])(\sigma))
\]
\[= d(D(s^1)(\alpha \cdot 1, [K(k) \mid \overline{\alpha} \mapsto D(s_1) \mid \alpha \cdot 2 \mapsto D(s^2)])(\sigma)\cup
\]
\[D(s^2)(\alpha \cdot 2, [K(k) \mid \overline{\alpha} \mapsto D(s_1) \mid \alpha \cdot 1 \mapsto D(s^1)])(\sigma),
\]
\[D(s^1)(\alpha \cdot 1, [K(k) \mid \overline{\alpha} \mapsto D(s_2) \mid \alpha \cdot 2 \mapsto D(s^2)])(\sigma)\cup
\]
\[D(s^2)(\alpha \cdot 2, [K(k) \mid \overline{\alpha} \mapsto D(s_2) \mid \alpha \cdot 1 \mapsto D(s^1)])(\sigma))
\]
\[\text{[}'\cup'\text{ is nonexpansive; }\neg(\overline{\alpha} \geq \alpha) \Rightarrow \neg(\overline{\alpha} \geq \alpha \cdot 1), \neg(\overline{\alpha} \geq \alpha \cdot 2); \text{ Lemma 5.1}]
\]
\[\leq \max\{d(D(s^1)(\alpha \cdot 1, [K[k \mid \alpha \cdot 2 \mapsto s^2] \mid \overline{\alpha} \mapsto D(s_1)])(\sigma),
\]
\[D(s^1)(\alpha \cdot 1, [K[k \mid \alpha \cdot 2 \mapsto s^2] \mid \overline{\alpha} \mapsto D(s_2)])(\sigma)\} \quad (5.10.2.6)
\]
\[d(D(s^2)(\alpha \cdot 2, [K[k \mid \alpha \cdot 1 \mapsto s^1] \mid \overline{\alpha} \mapsto D(s_1)])(\sigma),
\]
\[D(s^2)(\alpha \cdot 2, [K[k \mid \alpha \cdot 1 \mapsto s^1] \mid \overline{\alpha} \mapsto D(s_2)])(\sigma)\} \quad (5.10.2.7)
\]
As \(\neg(\overline{\alpha} \geq \alpha \cdot 1)\) and \(\neg(\overline{\alpha} \geq \alpha \cdot 2)\), we can apply the induction hypothesis and we infer that \(\exists s'_1 \in Stat, \alpha'_1 \in Id, k'_1 \in KRes, \sigma'_1 \in \Sigma\) with \(\neg(\overline{\alpha} \geq \alpha'_1)\) and \(\exists s'_2 \in Stat, \alpha'_2 \in Id, k'_2 \in KRes, \sigma'_2 \in \Sigma\) with \(\neg(\overline{\alpha} \geq \alpha'_2)\), such that:
\[\leq \frac{1}{2} \cdot d(D(s'_1)(\alpha'_1, [K(k'_1) \mid \overline{\alpha} \mapsto D(s_1)])(\sigma'_1),
\]
\[D(s'_1)(\alpha'_1, [K(k'_1) \mid \overline{\alpha} \mapsto D(s_2)])(\sigma'_1)) \quad (5.10.2.6')\]
\[ \frac{1}{2} \cdot d(D(s_2)(\alpha_2', [K(k_2') | \alpha \mapsto D(s_1)])(\sigma_2'), D(s_2')(\alpha_2', [K(k_2) | \alpha \mapsto D(s_2)])(\sigma_2')) \]

Finally, by taking \( \max\{ (5.10.2.6'), (5.10.2.7') \} \) we obtain:

\[ \epsilon_D^C(s^1 \parallel s^2, \alpha, k, \overline{\alpha}, s_1, s_2, \sigma) \leq \max\{ (5.10.2.6'), (5.10.2.7') \} \]

\[ = \max\{ \frac{1}{2} \cdot \epsilon_D^C(s'_1, \alpha'_1, k'_1, \overline{\alpha}, s_1, s_2, \sigma'_1), \frac{1}{2} \cdot \epsilon_D^C(s'_2, \alpha'_2, k'_2, \overline{\alpha}, s_1, s_2, \sigma'_2) \} \]

This implies immediately the desired result. \( \square \)

The main results are provided by Theorem 5.11. By using this theorem one can reason in a compositional manner upon the behavior of \( MCC \) programs. In general, the properties stated in Theorem 5.11 hold for all continuations containing only computations denotable by program statements. In practice this may be sufficient, because the initial continuation \( (\alpha_0, \kappa_0) \) is empty (contains no computations) and according to the semantic equations, the denotational mapping \( D(\cdot) \) adds only denotations of statements to the continuation. When evaluated with respect to the initial (empty) continuation, the denotational mapping preserves the following invariant property: a CSC continuation contains only computations denotable by program statements. Properties 5.11(a)-(d) hold for all continuations.

**THEOREM 5.11** For all \( s, s_1, s_2, s_3 \in Stat \):

(a) \( D(s_1 + s_2) = D(s_2 + s_1) \)

(b) \( D((s_1 + s_2) + s_3) = D(s_1 + (s_2 + s_3)) \)

(c) \( D(s + s) = D(s) \)

(d) \( D((s_1 + s_2); s_3) = D((s_1; s_3) + (s_2; s_3)) \)

Also, for all \( s_1, s_2, s_3 \in Stat, (\alpha, k), (\overline{\alpha}, \overline{k}) \in CRes, (\alpha, k) \equiv (\overline{\alpha}, \overline{k}) \) and for all contexts \( C \):

(e) \( D(C(s_1; s_2))(\alpha, K(k)) = D(C(s_2; s_1))(\overline{\alpha}, K(\overline{k})) \)

(f) \( D(C(s_1; (s_2 || s_3)))(\alpha, K(k)) = D(C((s_1; s_2) || s_3))(\overline{\alpha}, K(\overline{k})) \)

(g) \( D(C(s_1; (s_2; s_3)))(\alpha, K(k)) = D(C((s_1; s_2); s_3))(\overline{\alpha}, K(\overline{k})) \)

(h) \( D(C(s_1; (s_2 + s_3)))(\alpha, K(k)) = D(C((s_1; s_2) + (s_1; s_3)))(\overline{\alpha}, K(\overline{k})) \)

**Proof:** (a), (b) and (c) follow immediately from the properties of the union operator \( 'U' \). For (d) the proof was given at the beginning of section 5 for all continuations and \( (\alpha, \kappa) \in Cont \) and for all states \( \sigma \in \Sigma \). Here we only treat (e); (f), (g) and (h) can be handled similarly.
\[ D(C(s_1 \parallel s_2))(\alpha, K(k))(\sigma) \]
\[ = D(C(s_1 \parallel s_2))(\alpha, K(k))(\sigma) \quad \text{[Lemma 5.8(a), Lemma 5.9]} \]
\[ = D(C(s_2 \parallel s_1))(\alpha, K(k))(\sigma) \quad \text{[Corollary 1(a)]} \]
\[ = D(C(s_2 \parallel s_1)(\pi, K(k))(\sigma) \]

which implies \[ D(C(s_1 \parallel s_2))(\alpha, K(k)) = D(C(s_2 \parallel s_1)(\pi, K(k)), \] because the proof is given for an arbitrary \( \sigma \).

The function \( D[ \cdot ] \) (see Definition 4.1) is not defined in a compositional manner; only \( D(\cdot) \) is a compositional (denotational) mapping. However, \( D[s] \) is useful because it evaluates a (whole) program \( s \) with respect to the initial (empty) continuation. The following corollary can be used to reason about the behavior of \( D[s] \).

**Corollary 3** For all \( s, s_1, s_2, s_3 \in \text{Stat} \) and for all contexts \( C \):

\( a) \ D[C(s_1 + s_2)] = D[C(s_2 + s_1)] \)

\( b) \ D[C((s_1 + s_2) + s_3)] = D[C(s_1 + (s_2 + s_3)) \]

\( c) \ D[C(s + s)] = D[C(s)] \)

\( d) \ D[C((s_1 + s_2); s_3)] = D[C((s_1; s_3) + (s_2; s_3))] \)

\( e) \ D[C(s_1 \parallel s_2)] = D[C(s_2 \parallel s_1)] \)

\( f) \ D[C(s_1 \parallel (s_2 \parallel s_3))] = D[C((s_1 \parallel s_2) \parallel s_3)] \)

\( g) \ D[C(s_1; (s_2; s_3))] = D[(s_1; s_2); s_3)] \)

\( h) \ D[C(s_1; (s_2 + s_3))] = D[(s_1; s_2) + (s_1; s_3))]] \]

**Proof:** It suffices to consider one case.

\[ D[C(s_1; (s_2; s_3))](\sigma) \]
\[ = D(C(s_1; (s_2; s_3)))(\alpha_0, \kappa_0)(\sigma) \quad \text{[Theorem 5.11(g)]} \]
\[ = D(C((s_1; s_2); s_3))(\alpha_0, \kappa_0)(\sigma) \]
\[ = D[C((s_1; s_2); s_3)](\sigma) \]

which implies \( D[C(s_1; (s_2; s_3))] = D[(s_1; s_2); s_3)], \] because the proof is given for an arbitrary \( \sigma \).
6 Concluding Remarks

The first steps toward a continuation semantics for concurrency (CSC) are presented in [13]. Our previous work suggests that the (CSC) technique can provide a very good flexibility in the denotational design of parallel and distributed systems. We developed denotational models designed with CSC for a couple of advanced control concepts like the nondeterministic promotion in Andorra-like languages [15], and the communication with synchronization on multiple channels [14] in the style of Join calculus [7]. As far as we know these concepts have not been modeled denotationally until now without CSC. But the distinctive characteristic of the CSC technique is the representation of continuations as structures of computations rather than the straightforward functions to some answer type that are used in the classic technique of continuations.

This paper describes the concurrent systems in denotational models designed with CSC by using standard techniques from metric semantics [3]. We illustrate these ideas on the particular example of a CSP-like language MCC extended with a mechanism for synchronization on multiple channels in the style of Join calculus [7]. For such a language we have proved that the semantic operators designed with continuations obey the basic laws given in the algebraic theories of concurrency, such as the associativity and the commutativity of parallel composition.

The basic idea of the proofs was to show that the properties under consideration are preserved by the computation steps. We also used an \( \epsilon \leq \frac{1}{2} \cdot \epsilon \Rightarrow \epsilon = 0 \) argument which is standard in metric semantics [3]. The identification of the semantic properties from the invariants of the computation is common in classic bisimulation semantics [10].

As further work, we think to use the elegance of the semantic domains in expressing the CSC technique. In this paper the domain of continuations was modeled with the aid of a function space from a set of identifiers (endowed with a partial order) to the domain of computations: \( \text{Id} \rightarrow \text{Comp} \). According to Corollary 1, any two isomorphic continuations behave the same. Intuitively, the domain of continuations could be defined in terms of isomorphism classes \([\text{Id} \rightarrow \text{Comp}]\) of such structures. Since the existing metric models based on isomorphism classes of semantic structures (in particular the metric pomset model [4]) do not involve domains defined by reflexive equations (like \text{Comp}), such a construction requires further work.

References


