

Toward an Algebraic Theory of Systems*

Christian Matt^{†1}, Ueli Maurer¹, Christopher Portmann^{‡2},
Renato Renner², and Björn Tackmann^{§3}

¹Department of Computer Science, ETH Zurich, 8092 Zurich, Switzerland.
cmatt@cs.ucsb.edu, maurer@inf.ethz.ch

²Institute for Theoretical Physics, ETH Zurich, 8093 Zurich, Switzerland.
chportma@inf.ethz.ch, renner@phys.ethz.ch

³Department of Computer Science and Engineering, University of California, San Diego,
La Jolla, CA 92093, USA.
bta@zurich.ibm.com

Abstract

We propose the concept of a system algebra with a parallel composition operation and an interface connection operation, and formalize composition-order invariance, which postulates that the order of composing and connecting systems is irrelevant, a generalized form of associativity. Composition-order invariance explicitly captures a common property that is implicit in any context where one can draw a figure (hiding the drawing order) of several connected systems, which appears in many scientific contexts. This abstract algebra captures settings where one is interested in the behavior of a composed system in an environment and wants to abstract away anything internal not relevant for the behavior. This may include physical systems, electronic circuits, or interacting distributed systems.

One specific such setting, of special interest in computer science, are functional system algebras, which capture, in the most general sense, any type of system that takes inputs and produces outputs depending on the inputs, and where the output of a system can be the input to another system. The behavior of such a system is uniquely determined by the function mapping inputs to outputs. We consider several instantiations of this very general concept. In particular, we show that Kahn networks form a functional system algebra and prove their composition-order invariance.

Moreover, we define a functional system algebra of causal systems, characterized by the property that inputs can only influence future outputs, where an abstract partial order relation captures the notion of “later”. This system algebra is also shown to be composition-order invariant and appropriate instantiations thereof allow to model and analyze systems that depend on time.

Keywords: Systems, composition, abstraction, order invariance, fixed points, causality.

*© 2018. This manuscript version is made available under the CC-BY-NC-ND 4.0 license <https://creativecommons.org/licenses/by-nc-nd/4.0/>. The version published in *Theoretical Computer Science* is available at <https://doi.org/10.1016/j.tcs.2018.06.001>.

[†]Present address: Department of Computer Science, University of California, Santa Barbara, CA 93106, USA.

[‡]Present address: Department of Computer Science, ETH Zurich, 8092 Zurich, Switzerland.

[§]Present address: IBM Research – Zurich, 8803 Rüschlikon, Switzerland.

1 Introduction

1.1 Motivation

A universal concept in many disciplines is to characterize the behavior of an object, often called a module or a system, and thereby to intentionally ignore internal aspects considered irrelevant. We will here use the term system. The purpose of a system is to be used or embedded in an environment, which can for example consist of several other systems. A system's behavior is the exact characterization of the effect a system can have when embedded in an environment. In such a consideration, anything internal, not affecting the behavior, is by definition considered irrelevant, and two systems with the same behavior are considered to be equal. Which aspects are considered irrelevant, and hence are not modeled as part of the behavior, depends strongly on the concrete investigated question. For example, if for a software module computing a function, one is only interested in the input-output behavior, then this function characterizes the behavior of the software module, and aspects like the underlying computational model, the program, the complexity, timing guarantees, etc., are irrelevant. In another consideration, for example, one may want to include the timing aspects as part of the observable (and hence relevant) behavior. In yet another consideration one may be interested in the memory requirements for the specific computational model, etc.

One can think of a system as connecting to the environment via an *interface*. More generally, if one wants to model a system composed of several (sub-)systems, then one can consider each system to have several interfaces and that systems can be composed by connecting an interface of one system with an interface of another system. The term “interface” is here used in the sense of capturing the potential connection to another system; it is indeed used in some contexts to refer to the specification of the behavior of an object, such as a software module, but here we use the term in a more general and more abstract sense. Our notion of connected systems corresponds to drawing a diagram with boxes, each having several lines (interfaces), and where some interfaces of systems are connected (by lines) and some interfaces remain free, i.e., accessible by the environment; see Figure 1. A composed system appears to the environment as having only the free (not connected) interfaces and its behavior is observed only via these free interfaces; the internal topology becomes irrelevant and is not part of the behavior.

In some applications, certain internal details of a system matter. One can then define the

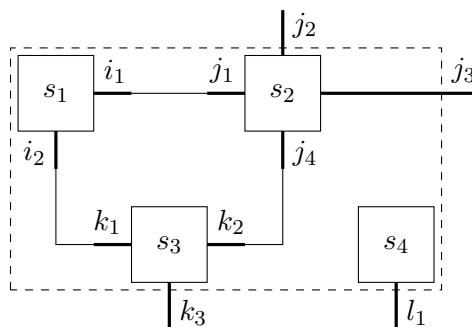


Figure 1: A system composed of the four subsystems s_1 , s_2 , s_3 , and s_4 . Interfaces are labeled to allow specifying connections, e.g., i_1 and j_1 are connected, and s_4 is not connected to the other systems. The resulting system has interfaces j_2 , j_3 , k_3 , and l_1 .

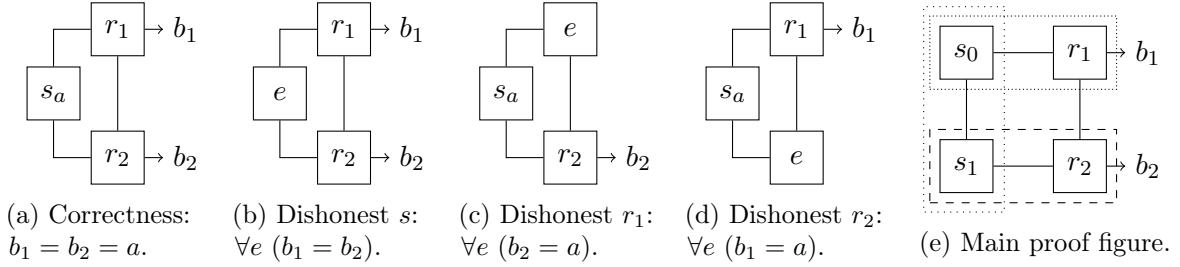


Figure 2: The impossibility of broadcast for three parties with one dishonest party. Figures 2a to 2d correspond to the requirements for a broadcast protocol. Figure 2e can be subdivided in different ways and is used in the proof. Interface labels are omitted to increase readability.

relevant internal aspects as being part of the behavior, to make them, by definition, visible from the outside. If too many internal details become relevant, our approach might be less suitable than directly using a model of systems that considers these internals. Indeed, a majority of existing work models systems by defining their internal operations, e.g., via states and transition functions. In many cases, however, such a detailed description of the internal operations is unnecessary and cumbersome, and our abstract approach would be beneficial. We now describe two examples for which ignoring internal details appear particularly useful.

Distributed systems. In distributed systems, where systems are connected to other systems with which they can communicate, one is often interested in certain properties of the composed system. As an example, we present a simple impossibility proof of bit-broadcast for three parties with one dishonest party. This famous result was first proven by Lamport et al. [LSP82; PSL80]; the proof we present here is in the spirit of that given by Fisher et al. [FLM86]. This proof only requires that the involved systems can be connected and rearranged as described; the communication between them and how they operate internally is irrelevant. Ignoring these internal details not only simplifies the proof but also makes the result more general, e.g., it also holds if the systems communicate via some sort of analog signals.

The goal of a bit-broadcast protocol is to allow a sender to send a bit such that all honest receivers output the same bit (consistency) and if the sender is honest, they output the bit that was sent (validity). Assume an honest sender uses the system s_a for broadcasting a bit $a \in \{0, 1\}$, and honest receivers use systems r_1 and r_2 that decide on a bit b_1 , and b_2 , respectively. If these systems implement a broadcast protocol with the required guarantees, each condition in Figures 2a to 2d must hold for all systems e , which capture the possible behaviors of a dishonest party. Assume toward a contradiction that this is the case and consider the system in Figure 2e. We can view the system in the dotted box composed of s_0 and s_1 as a system e in Figure 2b to obtain $b_1 = b_2$. We can also view the system in the densely dotted box composed of s_0 and r_1 as a system e in Figure 2c to obtain $b_2 = 1$. Finally, the system in the dashed box can be viewed as a system e in Figure 2d, which implies $b_1 = 0$, a contradiction. Hence, there are no systems s_0 , s_1 , r_1 , and r_2 that satisfy these constraints. In Section 3.2, we provide a more formal proof within our theory.

Cryptography. Cryptographic schemes are often defined as some sort of efficient algorithms. While efficiency is of course relevant in practice, one can separate the computational aspects from the functionality to simplify the analysis. *Constructive cryptography* by Maurer and

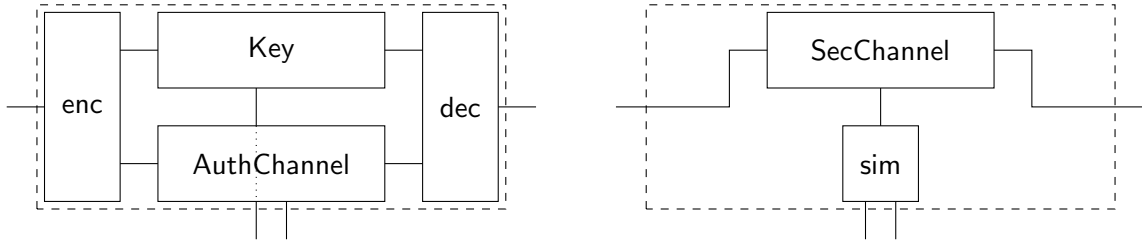


Figure 3: The systems appearing in the construction of a secure channel from an authenticated channel and a shared secret key via symmetric encryption. The construction notion requires the two systems in the dashed boxes to be indistinguishable.

Renner [MR11; Mau12] allows one to model what cryptographic protocols achieve using a system algebra that abstracts away cumbersome details. To this end, one considers so-called *resource systems*, which provide a certain functionality to the parties connected to their interfaces, and *converter systems*, which can be connected to resources to obtain a new resource. Typical resources with interfaces for two honest parties and an adversary are a shared secret key, which provides a randomly generated key at the interfaces for the honest parties and nothing at the interface for the adversary,¹ and different types of channels with different capabilities for the adversary. The goal of a cryptographic protocol is then to *construct* a resource S from a resource R . Such a protocol consists of a converter for each honest party, and it achieves the construction if the system obtained from R by connecting the protocol converters to the interfaces of the honest parties is *indistinguishable* from the system obtained from S by connecting some converter system, called *simulator*, to the interface of the adversary. The notion of indistinguishability can be defined in several ways leading to different types of security, e.g., as the systems being identical or via a certain class of distinguishers.

A crucial property of this construction notion is that it is *composable*, i.e., if some protocol constructs a resource S from a resource R and another protocol constructs a resource T from S , these protocols can be composed to obtain a construction of T from R . Turned around, one can also decompose the construction of T from R into two separate constructions. Since these two constructions can be analyzed independently, this approach provides modularity and simplifies the analysis of complex protocols by breaking them down into smaller parts.

An example of a construction is that of a secure channel, which leaks only the length of the sent messages to the adversary and does not allow modifications of them, from the resource consisting of a shared secret key and an authenticated channel, which leaks the sent messages to the adversary but also does not allow modifications of them, by a symmetric encryption scheme. See Figure 3 for an illustration of the involved systems. To achieve this construction, the simulator must, knowing only the length of the sent messages, output bit-strings that are indistinguishable from encryptions of these messages. If the used encryption scheme is, e.g., the one-time pad, the two systems in Figure 3 are identical for an appropriate simulator, i.e., they have the same input-output behavior [Mau12], which provides the strongest possible security guarantee. Note that internally, these systems are very different, but this is intentionally ignored.

¹One could also view a shared secret key as having no interface for the adversary, but as defined in constructive cryptography, all resources involved in a construction have an interface for each party.

1.2 Contributions

We develop a theory of systems with different levels of abstraction. To achieve generality and to strive for simplicity, theorems are proved at the highest level of abstraction at which they hold. See Figure 4 for an overview of the types of systems we consider. This is not meant to be a complete picture of all systems one can consider; systems we do not consider in this paper but could be treated within our theory include probabilistic systems, physical systems, circuits, etc.

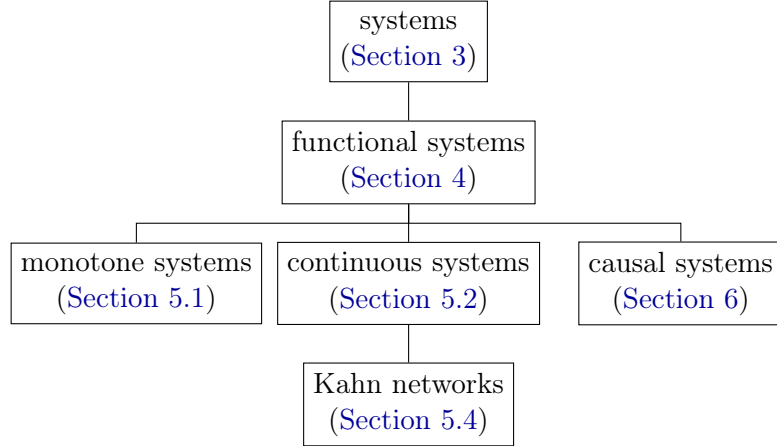


Figure 4: The hierarchy of the systems treated in this paper, where systems at lower levels are special cases of their parents.

Abstract system algebras and composition-order invariance. At the highest level of abstraction, we do not specify what systems are, but only postulate two operations \parallel and γ , depicted in Figure 5; the former for combining two systems in parallel as in Figure 5a and the latter for connecting interfaces as in Figure 5b. Using these operations one can build “graphs” such as those depicted in Figure 1, by first taking the systems in parallel and then connecting the interfaces. We call a set of systems together with these operations, and the specification which interfaces a system has and which of them can be connected, a *system algebra*, see Section 3.

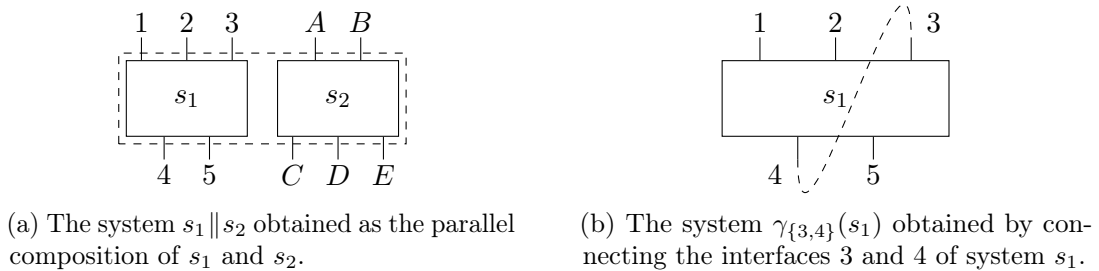


Figure 5: Composition operations in a system algebra.

A natural property of such an algebra that can be specified at this level of abstraction is *composition-order invariance*, that is, a composed system is completely described by its “graph”, and the order in which the operations are applied to build the graph does not matter. This property is not only very natural, but also necessary for many applications. For example, the

impossibility proof for broadcast sketched above relies on it since otherwise, rearranging and subdividing systems as in [Figure 2e](#) would not be allowed. To illustrate this, we formalize the systems occurring in that figure in an abstract system algebra and show how composition-order invariance appears in the proof in [Section 3.2](#). Composition-order invariance was also used by Maurer and Renner to prove the composition theorem of constructive cryptography [\[MR11\]](#). While appearing natural and innocent, examples throughout our paper indicate that composition-order invariance is actually a nontrivial property and requires a proof.

Functional system algebras. An important type of system in computer science takes inputs and produces outputs depending on these inputs. The behavior of such a system can be fully described by a function mapping inputs to outputs. We define a type of system algebra, called *functional system algebra*, in [Section 4.1](#) where the systems have input interfaces and output interfaces and correspond to such functions. Connecting an input interface to an output interface is understood as setting the value input at the former to be equal to the value output at the latter. Determining the resulting system thus involves finding a fixed point of the underlying function; if multiple fixed points exist, the system algebra has to specify which one to select. By appropriately choosing the domains and functions, various types of systems can be modeled in this way, including interactive systems that take many inputs in different rounds and systems that depend on time.

We prove several basic results at this level of abstraction, i.e., without specifying which functions are considered in [Section 4.2](#). For example, we show that not all functional system algebras are composition-order invariant, but if there are always unique fixed points for interface connections and connections can be reordered, composition-order invariance holds.

While this paper focuses on the deterministic case, we point out that functional systems can be used as a basis to model probabilistic systems. For example, one can consider systems that take randomness as an explicit input at a dedicated interface, one can include random variables in the domains of the functions, or one can consider probability distributions over deterministic systems. Systematically understanding probabilistic systems in this way is future work.

Instantiations of functional system algebras. To instantiate the concept of a functional system algebra, we need to specify the domains of the functions and the set of functions to consider. To be able to define the interface connection, we have to ensure that all functions have the required fixed points. One way to guarantee fixed points that is well-studied in mathematics, especially domain theory, is to equip the domain with a partial order such that all chains have a supremum and to consider *monotone functions*. A related concept are *continuous functions*, which are defined as preserving these suprema. In both cases the functions have a least fixed point. While continuity is a stronger requirement than monotonicity, a slightly weaker assumption on the domains is sufficient to guarantee least fixed points. We show in [Section 5.1](#) and [Section 5.2](#) that if least fixed points are chosen for interface connections, monotone and continuous functions form a functional system algebra, respectively. Under the additional assumption that nonempty chains have an infimum, we show in [Section 5.3](#) that these system algebras are composition-order invariant.

Monotone and continuous functions are not only a mathematical convenience to obtain fixed points, but they also encompass as a special case an intuitive and useful model known as *Kahn networks* [\[Kah74\]](#) (or *Kahn process networks*; in Kahn’s paper only defined for continuous functions, but monotone functions can be used as well), which we consider in [Section 5.4](#). Kahn

networks have been developed to provide a semantics for parallel programming languages [Kah74], but they have also been used in other contexts, including embedded systems [SZT+04] and signal processing [LP95]. The domains of the functions there consist of sequences of values and the partial order is defined to be the initial segment (or prefix) relation. An interpretation of a function is that it maps input histories to output histories. Such functions therefore correspond to interactive systems that take one input after the other at each input interface and produce outputs depending on the inputs. Monotonicity means that additional inputs can only yield additional outputs, an output cannot be “taken back”. Even though it appears to be a very natural question whether the order in which interfaces of Kahn networks are connected matters, we are not aware of any result in this direction. Our proof of composition-order invariance, which indeed turned out to be nontrivial, therefore also provides new insights into this well-studied model.

In Section 6, we finally provide an instantiation of functional system algebras consisting of *causal systems* in which inputs to the system can only influence “later” outputs. We formalize this by considering a partially ordered set (where “less than” can be interpreted as “before”) and letting the domains of the functions consist of subsets thereof. As an example, consider the partially ordered set containing pairs (v, t) , which can be interpreted as the value v being input (or output) at time t , where the order is naturally defined as the one induced by the second component. The domains are then sets of such pairs. This allows us, as for Kahn networks, to model systems that take several inputs at each input interface and produce several outputs. We define causality for such systems and prove that the corresponding functions have unique fixed points. Therefore, we obtain a composition-order invariant functional system algebra. This system algebra can in particular be used to model and analyze systems that depend on time, such as clocks and channels with certain delays.

1.3 Related Work

There exists a large body of work on modeling certain types of systems mathematically. Some models can be understood as special cases in our theory, but also very general theories and models that do not fit in our theory exist. The work we are aware of, however, only captures partial aspects of our theory. We now describe some of this work and compare it to ours.

Abstract models. The abstract concept of a system algebra in which complex systems are built from components has been informally described in the context of cryptography by Maurer and Renner [MR11]. They also, again informally, introduced composition-order independence, which corresponds to our composition-order invariance. We provide in this paper a formalization that matches their requirements.

Hardy has developed an abstract theory, in which composition-order invariance (there called order independence) plays an important role [Har13]. That work, however, focuses on physical systems. Lee and Sangiovanni-Vincentelli [LS98] also introduce an abstract system model, but it is specific to systems that consider some form of time and does not follow an algebraic approach.

Closely related to our abstract system algebras are block algebras as introduced by de Alfaro and Henzinger in the context of interface theories [dAH01b]. Our systems and interfaces are there called blocks and ports, respectively, and they also define parallel composition and port connection operations. A major difference compared to our system algebras is that port connections do not hide the connected ports. Moreover, while de Alfaro and Henzinger require the parallel composition to be commutative and associative, they do not define a notion that corresponds to

our composition-order invariance, i.e., their port connections not necessarily commute with other port connections and parallel composition.

Process algebras allow the modeling of communicating concurrent processes [BBR09], which correspond to our systems. Examples of process algebras include Milner’s Calculus of Communicating Systems (CCS) [Mil80] and Hoare’s Communicating Sequential Processes (CSP) [Hoa78]. These theories are similar to ours in that they consider certain operations, including parallel composition, on processes and postulate axioms they need to satisfy. These axioms also include guarantees similar to our composition-order invariance. In contrast to our theory, CCS does not have an explicit operation for connecting interfaces, there called ports. Rather, ports with matching labels are implicitly connected when systems are taken in parallel. As in the work by de Alfaro and Henzinger [dAH01b] discussed above, ports are not hidden by this connection; hiding them is an explicit operation in CCS. In CSP, processes communicate via special input and output commands. As opposed to our abstract level, these process algebras differentiate between inputs and outputs and allow to specify the behavior of processes. Thus, with respect to the level of abstraction, they belong somewhere between our abstract and functional levels, or even below.

Milner’s Flowgraphs [Mil79] model composed systems as generalized graphs, in which the nodes correspond to subsystems. This is considerably different from our theory since we abstract away all internal details of a system, in particular the subsystems that compose a system.

Functional models. A line of work on system models based on functions has been initiated by Kahn’s seminal paper [Kah74] on networks of autonomous computing systems. These systems may be sensitive to the order in which messages arrive on one interface, but they are oblivious to the relative order of incoming messages on different interfaces. He shows that least fixed points exist, based on earlier work by Tarski [Tar55], Scott [Sco70], and Milner [Mil73],² and therefore connecting systems is well-defined. Tackmann [Tac14] considered the case where systems are fully oblivious to the order of their incoming messages, which can be seen as a special case of Kahn networks where each interface contains at most one message. Micciancio and Tessaro [MT13] start from the same type as Kahn but extend it to tolerate certain types of order-dependent behavior within complex systems.

Timed models. Several works have defined *causal* system models. Lee and Sangiovanni-Vincentelli [LS98] define *delta causality*, which intuitively requires that each output must be provoked by an input that occurred at least a δ -difference earlier. They show that fixed points exist, based on Banach’s theorem. Cataldo et al. [CLL+06] generalize this to a notion of “superdense” time where multiple events may occur simultaneously. Portmann et al. [PMM+17], in the quantum scenario, describe a type of strict causality based on a *causality function* that can be seen as a generalization of delta causality. Naundorf [Nau00] considers *strict* causality without any minimal time distance, and proves that fixed points still exist. Matsikoudis and Lee [ML15] then show a *constructive* fixed point theorem for the same notion, which they refer to as *strictly contracting*. They show that it is implied by a more natural notion of (strict) causality where outputs can be influenced only by inputs that occur strictly earlier, under the assumption that the ordering of inputs is well-founded³. We show in Appendix A that the strict causality notion of [ML15] is essentially equivalent to the definition we introduce in this work.

²Other sources attribute the original theorem to Kleene or Knaster, see [LNS82].

³A partial order on a set T is *well-founded* if every nonempty subset of T has one or more minimal elements.

Except for the work of Portmann et al., none of the previously mentioned definitions of causal functions explicitly capture systems with multiple interfaces as the work of Kahn [Kah74] or our work. Also, the mentioned papers investigating causal functions do not define how to connect systems such that one obtains a system of the same type, and therefore they do not provide a system algebra as we do. The model by Portmann et al. [PMM+17] captures quantum information-processing systems and can be seen as a generalization of our causal systems. Restricting that model to classical, deterministic inputs and outputs yields, however, a more complex and less general model than our causal systems. For example, the causality definition in that paper is more restrictive and in contrast to our causal systems, the systems there are not allowed to produce infinitely many outputs in finite time.

Partial orders have also been used to model causal or temporal dependencies, for example in Pratt’s theory of partially ordered multisets (pomsets) [Pra86; Pra85]. As in our causal systems, there is a partial order on the possible events. In contrast to our causal systems, however, systems in that theory are not necessarily functions and can exhibit nondeterminism. Our model is therefore more specific, which allows a more accessible presentation.

Stateful models. Several models of systems have been proposed that model the systems as objects that explicitly contain state. I/O automata initially discussed by Lynch and Tuttle [LT89] and interface automata by de Alfaro and Henzinger [dAH01a] enhance stateful automata by interactive communication. Timed automata by Alur and Dill [AD94] and timed I/O automata by Kaynar et al. [KLSV10] extend them to include a notion of time as our causal systems. Interactive Turing machines basically equip Turing machines with additional tapes that they share with other machines and have been used widely in (complexity-theoretic) cryptography [Gol01; Can01]. All these models are substantially different from ours since we want to hide all internal details of systems, including their state.

2 Preliminaries

2.1 Functions and Notation for Sets and Tuples

A *function* $f: X \rightarrow Y$ is a subset of $X \times Y$ such that for every $x \in X$, there is exactly one $y \in Y$ such that $(x, y) \in f$, where we will usually write $f(x) = y$ instead of $(x, y) \in f$. For two sets X and Y , the set of all functions $X \rightarrow Y$ is denoted by Y^X . A *partial function* $f: X \rightarrow Y$ is a function $X' \rightarrow Y$ for some $X' \subseteq X$. For a subset $S \subseteq X$, $f(S) := \{f(s) \mid s \in S\}$ denotes the *image of S under f* . For $X' \subseteq X$, we define the *restriction of f to X'* as $f|_{X'} := \{(x, y) \in f \mid x \in X'\} \in Y^{X'}$. Note that for $z \notin X$ and $y \in Y$, we have $f \cup \{(z, y)\} \in Y^{X \cup \{z\}}$. An element in Y^X can equivalently be interpreted as a tuple of elements in Y indexed by elements in X . In case we interpret a function as a tuple, we usually use a boldface symbol to denote it. For a tuple $\mathbf{x} \in X^I$ with $\mathbf{x}(i) = x_i$ for all $i \in I$, we also write $\mathbf{x} = (x_i)_{i \in I}$ and if $I = \{i_1, \dots, i_n\}$, we write $\mathbf{x} = (x_{i_1}, \dots, x_{i_n})$. The *symmetric difference* of two sets X and Y is defined as

$$X \triangle Y := (X \setminus Y) \cup (Y \setminus X).$$

Finally, we denote the *power set* of a set X by $\mathcal{P}(X) := \{S \mid S \subseteq X\}$.

2.2 Order Relations

We first recall some basic definitions about relations.

Definition 2.1. Let X be a set. A (binary) relation on X is a subset of $X \times X$. We write $x R y$ for $(x, y) \in R$. A relation $R \subseteq X \times X$ is called *reflexive* if $x R x$ for all $x \in X$. It is called *symmetric* if $x R y \implies y R x$ for all $x, y \in X$ and *antisymmetric* if $(x R y \wedge y R x) \implies x = y$. A relation $R \subseteq X \times X$ is called *transitive* if $(x R y \wedge y R z) \implies x R z$ for all $x, y, z \in X$.

For sets X and I and a binary relation R on X , we define the relation R on X^I as the componentwise relation, i.e., for $\mathbf{x}, \mathbf{y} \in X^I$,

$$\mathbf{x} R \mathbf{y} \iff \forall i \in I (\mathbf{x}(i) R \mathbf{y}(i)).$$

Definition 2.2. A *partial order* on X is a binary relation on X that is reflexive, antisymmetric, and transitive. A *partially ordered set (poset)* (X, \preceq) is a set X together with a partial order \preceq on X .

We will typically denote partial orders by \leq , \preceq , or \sqsubseteq , and define the relation $<$ by $x < y \iff x \leq y \wedge x \neq y$, and analogously \prec and \sqsubset .

Definition 2.3. Let (X, \preceq) be a poset. Two elements $x, y \in X$ are *comparable* if $x \preceq y$ or $y \preceq x$, and *incomparable* otherwise. If all $x, y \in X$ are comparable, X is *totally ordered*. A totally ordered subset of a poset is called a *chain*.

Definition 2.4. Let (X, \preceq) be a poset. An element $x \in X$ is the *least element* of X if $x \preceq y$ for all $y \in X$. Similarly, $x \in X$ is the *greatest element* of X if $y \preceq x$ for all $y \in X$. The least element and greatest element of X are denoted by $\min X$ and $\max X$, respectively. An element $x \in X$ is a *minimal element* of X if there is no $y \in X$ with $y \prec x$, and x is a *maximal element* if there is no $y \in X$, $y \succ x$. For a subset $S \subseteq X$, $x \in X$ is a *lower bound* of S if $x \preceq s$ for all $s \in S$ and an *upper bound* of S if $x \succeq s$ for all $s \in S$. If the set of lower bounds has a greatest element, it is called the *infimum* of S , denoted $\inf S$; the *supremum* of S , denoted $\sup S$, is the least upper bound of S .

Definition 2.5. A poset (X, \preceq) is *well-ordered* if every nonempty subset of X has a least element.

Note that every well-ordered poset (X, \preceq) is totally ordered because $\{x, y\}$ has a least element.

Definition 2.6. Let (X, \preceq) and (Y, \preceq) be posets. An *order isomorphism* is a bijection $\psi: X \rightarrow Y$ such that $\psi(x_1) \preceq \psi(x_2) \iff x_1 \preceq x_2$ for all $x_1, x_2 \in X$. The posets (X, \preceq) and (Y, \preceq) are called *order isomorphic* if such order isomorphism exists.

2.3 Ordinals and Transfinite Induction

We briefly recall some basics of set theory, following Halbeisen [Hal12] and Jech [Jec03]. A *class* is a collection of sets. More formally, a class C corresponds to a logical formula and we write $x \in C$ if x satisfies that formula. Every set is a class but not all classes are sets; for example, the class of all sets and the class of all ordinals are not sets. A class that is not a set is called a *proper class*.

Definition 2.7. An *ordinal* is a set α such that $\forall x \in \alpha (x \subseteq \alpha)$, $\forall x_1, x_2 \in \alpha (x_1 \in x_2 \vee x_1 = x_2 \vee x_2 \in x_1)$, and $\forall S \subseteq \alpha, S \neq \emptyset \exists x \in S \forall y \in x (y \notin S)$.

For ordinals α, β with $\alpha \in \beta$, we write $\alpha < \beta$. For all ordinals $\alpha \neq \beta$, we have either $\alpha < \beta$ or $\beta < \alpha$ (but not both) and $\alpha \not< \alpha$. It can be shown that every nonempty class of ordinals has a least element (according to the relation $<$) [Hal12, Theorem 3.12]. For an ordinal α , we define

$$\alpha + 1 := \alpha \cup \{\alpha\}.$$

We have that $\alpha + 1$ is the least ordinal greater than α [Hal12, Corollary 3.13]. An ordinal α is called *successor ordinal* if $\alpha = \beta + 1$ for some ordinal β . A *limit ordinal* is an ordinal that is not a successor ordinal.

Every well-ordered set X is order isomorphic to exactly one ordinal [Jec03, Theorem 2.12]. This ordinal is called the *order type* of X .

We define the natural numbers as $0 := \emptyset$, $1 := 0 + 1 = \{0\}$, $2 := 1 + 1 = \{0, 1\}$, and so on. That is, a number $n \in \mathbb{N}$ is the set of all numbers less than n . The set of natural numbers \mathbb{N} is also an ordinal, denoted by ω . Note that 0 and $\omega = \mathbb{N}$ are limit ordinals and all nonzero natural numbers are successor ordinals. A method for proving a statement about all ordinals is via the transfinite induction theorem [Jec03, Theorem 2.14].

Theorem 2.8 (Transfinite Induction). *Let C be a class of ordinals such that*

- (i) $0 \in C$,
- (ii) if $\alpha \in C$, then $\alpha + 1 \in C$, and
- (iii) if α is a nonzero limit ordinal and $\beta \in C$ for all $\beta < \alpha$, then $\alpha \in C$.

Then, C is the class of all ordinals.

The following lemma will later be useful.

Lemma 2.9. *Let (X, \preceq) be a poset and assume that for every ordinal α , there is an $x_\alpha \in X$ such that $x_\beta \preceq x_\alpha$ for all $\beta \leq \alpha$.⁴ Then, there exists an ordinal $\hat{\alpha}$ such that $x_{\hat{\alpha}} = x_{\hat{\alpha}+1}$.*

Proof. A result by Hartogs implies that for any set S , there exists an ordinal $\eta(S)$ such that there is no injective function $\eta(S) \rightarrow S$ [Joh87, Lemma 7.1]. Hence, there exist ordinals $\hat{\alpha} < \beta$ such that $x_{\hat{\alpha}} = x_\beta$, since otherwise the function $\eta(X) \rightarrow X, \gamma \mapsto x_\gamma$ would be injective. Since $\hat{\alpha} \leq \hat{\alpha} + 1 \leq \beta$, we have $x_{\hat{\alpha}} \preceq x_{\hat{\alpha}+1} \preceq x_\beta$. Thus, $x_{\hat{\alpha}} = x_\beta$ implies that $x_{\hat{\alpha}} = x_{\hat{\alpha}+1}$. \square

2.4 Complete Posets and Fixed Points of Monotone and Continuous Functions

A natural requirement for functions between posets is that they preserve order. Order-preserving functions are also called monotone and are defined below.

Definition 2.10. Let (X, \leq) and (Y, \preceq) be posets. A function $f: X \rightarrow Y$ is *monotone* if

$$\forall x_1, x_2 \in X \ (x_1 \leq x_2 \rightarrow f(x_1) \preceq f(x_2)).$$

Note that a monotone bijection is not necessarily an order isomorphism: For $X = \{x_0, x_1\}$ with incomparable x_0 and x_1 and $Y = \{0, 1\}$ with $0 \preceq 1$, the bijection $f: X \rightarrow Y, x_i \mapsto i$ is trivially monotone but not an order isomorphism.

⁴More formally, we consider a *class function* F from the class of all ordinals to X , and write x_α for $F(\alpha)$.

Definition 2.11. Let X be a set and $f: X \rightarrow X$ be a function. Then, $x \in X$ is called a *fixed point* of f if $f(x) = x$.

Definition 2.12. A *complete partially ordered set (CPO)* is a poset in which every chain has a supremum.

Note that the empty set is a chain and every element is an upper bound of \emptyset . Therefore, a CPO contains a least element.

Theorem 2.13 ([Ési09, Theorem 2.5]). *Let (X, \preceq) be a CPO and $f: X \rightarrow X$ be monotone. Then, f has a least fixed point, which equals $x_{\hat{\alpha}}$ for some ordinal $\hat{\alpha}$, where $x_0 = \min(X)$, $x_{\alpha+1} = f(x_\alpha)$ for any ordinal α , and for nonzero limit ordinals α , $x_\alpha = \sup\{x_\beta \mid \beta < \alpha\}$. We further have $x_\alpha \leq x_\beta$ for $\alpha \leq \beta$.*

The above theorem is constructive in the sense that it not only guarantees the existence of a least fixed point, but also provides a procedure to find it. However, this procedure might only terminate after transfinitely many steps. The situation improves if the function is not only monotone but also continuous in the sense that it preserves suprema. In this case, a weaker requirement on the domain of the function is sufficient, namely only chains that correspond to infinite sequences need to have a supremum.

Definition 2.14. Let (X, \preceq) be a poset. An ω -chain in X is a sequence $(x_n)_{n \in \omega}$ such that $x_i \preceq x_j$ for all $i \leq j$. We say (X, \preceq) is an ω -chain complete partially ordered set (ω -CPO) if it has a least element and every ω -chain has a supremum.

Definition 2.15. Let (X, \leq) and (Y, \preceq) be ω -CPOs. A function $f: X \rightarrow Y$ is ω -continuous if for every ω -chain $C = (x_n)_{n \in \omega}$ in X , $\sup f(C)$ exists and $f(\sup C) = \sup f(C)$.

The next lemma shows that ω -continuity implies monotonicity (the converse is not true).

Lemma 2.16. *Let (X, \leq) and (Y, \preceq) be ω -CPOs and $f: X \rightarrow Y$ an ω -continuous function. Then, f is monotone.*

Proof. Let $x_1, x_2 \in X$ such that $x_1 \leq x_2$. Then, $\{x_1, x_2, x_2, x_2, \dots\}$ is an ω -chain. Therefore,

$$f(x_1) \preceq \sup f(\{x_1, x_2, x_2, \dots\}) = f(\sup\{x_1, x_2, x_2, \dots\}) = f(x_2). \quad \square$$

Theorem 2.17 ([DP02, Theorem 8.15]). *Let (X, \preceq) be an ω -CPO and $f: X \rightarrow X$ be ω -continuous. Then, f has a least fixed point, which equals $\sup\{x_n \mid n \in \omega\}$, where $x_0 = \min(X)$ and $x_{n+1} = f(x_n)$ for $n \in \omega$.*

Remark. In the literature, CPOs and continuity are often defined in terms of so-called directed subsets instead of chains [DP02; Ési09]. For our purposes, chains are more intuitive and directly applicable in our proofs. This definitional inconsistency is not an issue since the two types of definitions have been shown to be equivalent [Mar76].

3 Abstract System Algebras

We first define system algebras at an abstract level described, but not formalized, by Maurer and Renner [MR11], where systems are objects with interfaces via which they can be connected to other systems. Our motivation for defining system algebras at this level of abstraction is

twofold: First, we thereby introduce a common language and notation that can be used for more concrete system algebras in later sections as well as in future papers. Secondly, one can already make meaningful statements and prove theorems at this level of abstraction. As examples, we provide a proof of broadcast impossibility in [Section 3.2](#) and refer to the paper by Maurer and Renner [\[MR11\]](#), where the composition theorem of constructive cryptography is proven based on abstract systems. Whenever a proof is possible at this level of abstraction, it is clearly preferable to proofs at lower levels due to increased generality and simplicity.

3.1 Definitions

A system algebra is a set of systems with certain operations that allow one to compose several systems to obtain a new system. In this way, complex systems can be decomposed into independent components. At this level of abstraction, we only specify how systems can be composed, but not what systems are or how they interact with other systems. In the same sense as the elements of an algebraic ring are abstract objects without concrete meaning, abstract systems have no particular meaning attached, beyond how they can be composed. In a concrete instantiation, systems could, e.g., communicate via discrete inputs and outputs at their interfaces or via analog signals, et cetera. We define two operations; an operation \parallel for taking two systems in parallel, and an operation γ for connecting two interfaces of a system. See [Figure 5](#) for a depiction of these two operations. Several systems can be connected by first taking them in parallel and then connecting their interfaces. A similar definition has been given by Tackmann [\[Tac14, Definition 3.4\]](#).

Definition 3.1. Let Λ be a set (the set of interface labels). A Λ -system algebra $(\mathfrak{S}, \lambda, \parallel, \Gamma, \gamma)$ consists of a set \mathfrak{S} (the set of systems), a function $\lambda: \mathfrak{S} \rightarrow \mathcal{P}(\Lambda)$ (assigning to each system its set of interface labels), a partial function $\parallel: \mathfrak{S} \times \mathfrak{S} \rightarrow \mathfrak{S}$ (the *parallel composition* operation), a function $\Gamma: \mathfrak{S} \rightarrow \mathcal{P}(\{\{i, i'\} \mid i, i' \in \Lambda\})$ (specifying for each system the set of interface-label pairs that can be connected), and a partial function $\gamma: \{\{i, i'\} \mid i, i' \in \Lambda\} \times \mathfrak{S} \rightarrow \mathfrak{S}$ (the *interface connection* operation), such that

- for all $s \in \mathfrak{S}$, $\lambda(s)$ is finite and for all $\{i, i'\} \in \Gamma(s)$, we have $i, i' \in \lambda(s)$,
- for $s_1, s_2 \in \mathfrak{S}$, $\parallel(s_1, s_2)$, denoted $s_1 \parallel s_2$, is defined if and only if $\lambda(s_1) \cap \lambda(s_2) = \emptyset$, and in this case, $\lambda(s_1 \parallel s_2) = \lambda(s_1) \cup \lambda(s_2)$ and for $j \in \{1, 2\}$ and for all $i, i' \in \lambda(s_j)$, we have $\{i, i'\} \in \Gamma(s_1 \parallel s_2) \iff \{i, i'\} \in \Gamma(s_j)$, and
- for $i, i' \in \Lambda$ and $s \in \mathfrak{S}$, $\gamma(\{i, i'\}, s)$, denoted by $\gamma_{\{i, i'\}}(s)$, is defined if and only if $\{i, i'\} \in \Gamma(s)$, and in this case, $\lambda(\gamma_{\{i, i'\}}(s)) = \lambda(s) \setminus \{i, i'\}$.

We will usually identify a system algebra with the set of systems \mathfrak{S} and use the same symbols λ , \parallel , Γ , and γ for different system algebras. The parallel composition of two system is only allowed if they have disjoint interface sets. This means in particular that one cannot consider the parallel composition of a system with itself. One can imagine that each system exists only once and therefore cannot be used twice within another system. This is not an issue because \mathfrak{S} can contain many “copies” of a system with different interface labels, and different systems can have different interface labels. One could also introduce an interface-renaming operation, but we will not formalize this because it is not needed here. The set $\Gamma(s)$ determines which interfaces of a system s are compatible, i.e., can be connected to each other. It might or might not be possible to connect an interface to itself. Figuratively speaking, one could imagine that interfaces come with different types of plugs and one can only connect interfaces with matching plugs, where

$\Gamma(s)$ contains all unordered pairs of matching interfaces. For example, we will later consider system algebras with separate interfaces for inputs and outputs, where one can only connect input interfaces to output interfaces, but not two interfaces of the same type. Since the connection operation is defined for unordered pairs of interfaces, one always connects two interfaces to each other, without a direction. The condition on Γ for the parallel composition ensures that if one can connect two interfaces of a system, one can still do so after taking another system in parallel (i.e., $\{i, i'\} \in \Gamma(s_j) \implies \{i, i'\} \in \Gamma(s_1 \parallel s_2)$ for $j \in \{1, 2\}$), and additional connections are only created between the two systems, not for a single system (i.e., for $j \in \{1, 2\}$ and $i, i' \in \lambda(s_j)$, $\{i, i'\} \in \Gamma(s_1 \parallel s_2) \implies \{i, i'\} \in \Gamma(s_j)$). The intuition behind this condition is that the two systems are independent and do not influence what is possible for the other system. After connecting interfaces, however, it is possible that connections that were allowed before become disallowed. For example, one might want to consider a system algebra in which one cannot create cycles by connecting systems, e.g., when modeling systems that correspond to Boolean circuits. Then, certain connections are only allowed as long as other interfaces are not connected.

We restrict ourselves to systems with finitely many interfaces because we are only interested in systems that are composed of finitely many components. Therefore, we can define parallel composition as a binary operation and interface connection for a single pair of interfaces, whereas in general, one would define the parallel composition of potentially infinitely many systems and the connection of potentially infinitely many pairs of interfaces. In our simplified setting, repeated applications of the binary parallel composition and the connection of two interfaces are sufficient.

An important property system algebras can have is *composition-order invariance* (called composition-order independence by Maurer and Renner [MR11]). Loosely speaking, it guarantees that a system that is composed of several systems is independent of the order in which they have been composed. Put differently, a figure in which several systems are connected by lines uniquely determines the overall system; the order in which the figure was drawn is irrelevant.

Definition 3.2. For a Λ -system algebra $(\mathfrak{S}, \lambda, \parallel, \Gamma, \gamma)$, we say Γ *permits reordering* if for all $s \in \mathfrak{S}$, $\{i, i'\} \in \Gamma(s)$, and $\{j, j'\} \in \Gamma(\gamma_{\{i, i'\}}(s))$, we have $\{j, j'\} \in \Gamma(s)$ and $\{i, i'\} \in \Gamma(\gamma_{\{j, j'\}}(s))$. If additionally $\gamma_{\{j, j'\}}(\gamma_{\{i, i'\}}(s)) = \gamma_{\{i, i'\}}(\gamma_{\{j, j'\}}(s))$, $(\mathfrak{S}, \lambda, \parallel, \Gamma, \gamma)$ is called *connection-order invariant*. A connection-order invariant system algebra is called *composition-order invariant* if the operation \parallel is associative and commutative and for all $s_1, s_2 \in \mathfrak{S}$ and $\{i, i'\} \in \Gamma(s_1)$ such that $\lambda(s_1) \cap \lambda(s_2) = \emptyset$, we have $\gamma_{\{i, i'\}}(s_1) \parallel s_2 = \gamma_{\{i, i'\}}(s_1 \parallel s_2)$.

All system algebras we consider in this paper have associative and commutative parallel composition. Note however, that one can also imagine system algebras where this is not the case: Consider a set of systems that correspond to software components that are compiled together when two systems are composed in parallel. Depending on compiler optimizations, the efficiency of the resulting program might depend on the order in which components are compiled together.

3.2 An Abstract Proof of Broadcast Impossibility

In this section, we provide a formal proof at the level of abstract systems of the impossibility of bit-broadcast for three parties with one dishonest party, as sketched in the introduction. This exemplifies how a proof that involves drawing figures can be justified at the level of abstract systems and why composition-order invariance is crucial for doing so. Since there is no notion of outputting a bit for abstract systems, one cannot directly formulate the requirements for broadcast at this level. We therefore first prove a more abstract statement and afterwards argue how this implies the impossibility for more concrete systems. We assume in the following that we

have a system algebra that allows to connect systems arbitrarily, i.e., two (different) interfaces of a system can always be connected (formally, $\{i, i'\} \in \Gamma(s)$ for all $i, i' \in \lambda(s)$ with $i \neq i'$). This is a reasonable assumption, e.g., in a setting where the systems communicate with other systems by sending messages over channels.

Theorem 3.3. *Let $(\mathfrak{S}, \lambda, \parallel, \Gamma, \gamma)$ be a composition-order invariant Λ -system algebra such that $\Gamma(s) \supseteq \{\{i, i'\} \mid i, i' \in \lambda(s) \wedge i \neq i'\}$ for all $s \in \mathfrak{S}$. Let $s_0, s_1, r_1, r_2 \in \mathfrak{S}$ such that $\lambda(s_0), \lambda(s_1), \lambda(r_1)$, and $\lambda(r_2)$ are pairwise disjoint and $s_0^1, s_0^2 \in \lambda(s_0)$, $s_1^1, s_1^2 \in \lambda(s_1)$, $r_1^1, r_1^2 \in \lambda(r_1)$, and $r_2^1, r_2^2 \in \lambda(r_2)$, where these eight interface labels are distinct. Further let*

$$\begin{aligned} \mathcal{S}_0 &:= \{ \gamma_{\{r_1^2, e^2\}}(\gamma_{\{s_0^2, e^1\}}(\gamma_{\{s_0^1, r_1^1\}}(s_0 \parallel r_1 \parallel e))) \mid e \in \mathfrak{S} \wedge \lambda(e) \cap \lambda(s_0) = \emptyset = \lambda(e) \cap \lambda(r_1) \\ &\quad \wedge e^1, e^2 \in \lambda(e) \wedge e^1 \neq e^2 \}, \\ \mathcal{S}_1 &:= \{ \gamma_{\{e^2, r_2^2\}}(\gamma_{\{s_1^2, r_2^1\}}(\gamma_{\{s_1^1, e^1\}}(s_1 \parallel e \parallel r_2))) \mid e \in \mathfrak{S} \wedge \lambda(e) \cap \lambda(s_1) = \emptyset = \lambda(e) \cap \lambda(r_2) \\ &\quad \wedge e^1, e^2 \in \lambda(e) \wedge e^1 \neq e^2 \}, \\ \mathcal{S}_= &:= \{ \gamma_{\{r_1^2, r_2^2\}}(\gamma_{\{e^2, r_2^1\}}(\gamma_{\{e^1, r_1^1\}}(e \parallel r_1 \parallel r_2))) \mid e \in \mathfrak{S} \wedge \lambda(e) \cap \lambda(r_1) = \emptyset = \lambda(e) \cap \lambda(r_2) \\ &\quad \wedge e^1, e^2 \in \lambda(e) \wedge e^1 \neq e^2 \}. \end{aligned}$$

Then, $\mathcal{S}_0 \cap \mathcal{S}_1 \cap \mathcal{S}_= \neq \emptyset$.

Proof. Note that \mathcal{S}_0 corresponds to the set of systems for all possible e and $a = 0$ in Figure 2d, \mathcal{S}_1 corresponds to Figure 2c for $a = 1$, and $\mathcal{S}_=$ corresponds to the systems in Figure 2b. Let

$$t := \gamma_{\{r_1^2, r_2^2\}}(\gamma_{\{s_1^2, r_2^1\}}(\gamma_{\{s_0^1, r_1^1\}}(\gamma_{\{s_0^2, s_1^1\}}(s_0 \parallel s_1 \parallel r_1 \parallel r_2))))$$

be the system in Figure 2e. By composition-order invariance, we have for $e = \gamma_{\{s_0^2, s_1^1\}}(s_0 \parallel s_1)$,

$$e \parallel r_1 \parallel r_2 = \gamma_{\{s_0^2, s_1^1\}}(s_0 \parallel s_1) \parallel r_1 \parallel r_2 = \gamma_{\{s_0^2, s_1^1\}}(s_0 \parallel s_1 \parallel r_1 \parallel r_2).$$

Hence, we obtain for $e^1 = s_0^1$, and $e^2 = s_1^2$, that $t = \gamma_{\{r_1^2, r_2^2\}}(\gamma_{\{e^2, r_2^1\}}(\gamma_{\{e^1, r_1^1\}}(e \parallel r_1 \parallel r_2))) \in \mathcal{S}_=$. Again using composition-order invariance, we further have for $\hat{e} = \gamma_{\{s_1^2, r_2^1\}}(s_1 \parallel r_2)$, $\hat{e}^1 = s_1^1$, and $\hat{e}^2 = r_2^2$,

$$\begin{aligned} t &= \gamma_{\{r_1^2, r_2^2\}}(\gamma_{\{s_0^2, s_1^1\}}(\gamma_{\{s_0^1, r_1^1\}}(\gamma_{\{s_1^2, r_2^1\}}(s_1 \parallel r_2 \parallel s_0 \parallel r_1)))) \\ &= \gamma_{\{r_1^2, \hat{e}^2\}}(\gamma_{\{s_0^2, \hat{e}^1\}}(\gamma_{\{s_0^1, r_1^1\}}(\hat{e} \parallel s_0 \parallel r_1))) \in \mathcal{S}_0. \end{aligned}$$

Finally, we have for $\tilde{e} = \gamma_{\{s_0^1, r_1^1\}}(s_0 \parallel r_1)$, $\tilde{e}^1 = s_0^2$, and $\tilde{e}^2 = r_1^2$,

$$\begin{aligned} t &= \gamma_{\{r_1^2, r_2^2\}}(\gamma_{\{s_1^2, r_2^1\}}(\gamma_{\{s_0^2, s_1^1\}}(\gamma_{\{s_0^1, r_1^1\}}(s_0 \parallel r_1 \parallel s_1 \parallel r_2)))) \\ &= \gamma_{\{\tilde{e}^2, r_2^2\}}(\gamma_{\{s_1^2, r_2^1\}}(\gamma_{\{\tilde{e}^1, s_1^1\}}(\tilde{e} \parallel s_1 \parallel r_2))) \in \mathcal{S}_1. \end{aligned}$$

Therefore, we have $t \in \mathcal{S}_0 \cap \mathcal{S}_1 \cap \mathcal{S}_= \neq \emptyset$. □

To see why this implies the claimed impossibility, assume a protocol for broadcast exists and let s_a for $a \in \{0, 1\}$ be a system that implements the protocol for the sender to broadcast the bit a and let r_1 and r_2 be systems for the two receivers such that these systems have distinct interface labels.⁵ The validity of the broadcast protocol implies that for all systems in \mathcal{S}_0 , the

⁵Since interface labels are only used for connecting systems and typical protocols do not depend on them, it is reasonable to assume that such systems with distinct interface labels exist.

subsystem r_1 decides on the bit 0 (say with probability more than $\frac{2}{3}$) and for all systems in \mathcal{S}_1 , the subsystem r_2 decides on the bit 1 (with probability more than $\frac{2}{3}$). The consistency condition further implies that for all systems in $\mathcal{S}_=$, r_1 and r_2 decide on the same bit (with probability more than $\frac{2}{3}$). Now [Theorem 3.3](#) says that there is a system that satisfies all three constraints, which is impossible. Therefore no such protocol exists.

4 Functional System Algebras

4.1 Definitions

We now introduce special system algebras for functional systems that take inputs at dedicated input interfaces and produce outputs at their output interfaces, where the outputs are computed as a function of the inputs. This not only allows us to model systems that take a single input at each input interface and produce a single output at each output interface, but also much more general systems. For example, to model interactive systems that successively take inputs and produce outputs, one can consider the set of sequences of values as the domain of the functions. The function corresponding to a system then maps an entire input history to the output history and is a compact description of the system.

We define the parallel composition of two systems to be the function that evaluates both systems independently. Interface connection is defined in a way such that after connecting an input interface to an output interface, the input at the former equals the output at the latter. This corresponds to having a fixed point of a certain function determined by the connected interfaces and the system. One therefore has to choose Γ such that fixed points for all allowed connections exist. Ideally, there is always a unique fixed point, because in this case, the interface connection operation is uniquely determined by this condition. If there are several fixed points, one has to be chosen in each case. A functional system algebra is therefore characterized by a set of functions \mathfrak{S} , a function Γ determining the allowed interface connections, and an appropriate choice of fixed points ϕ . We use boldface letters for functional systems to distinguish them from abstract systems.

Definition 4.1. Let Λ and \mathcal{X} be sets, let for all finite disjoint $\mathcal{I}, \mathcal{O} \subseteq \Lambda$, $\mathfrak{S}_{\mathcal{I}, \mathcal{O}} \subseteq (\mathcal{X}^{\mathcal{O}})^{\mathcal{X}^{\mathcal{I}}}$ be a set of functions $\mathbf{s}: \mathcal{X}^{\mathcal{I}} \rightarrow \mathcal{X}^{\mathcal{O}}$, and let \mathfrak{S} be the union of all $\mathfrak{S}_{\mathcal{I}, \mathcal{O}}$. For $\mathbf{s} \in \mathfrak{S}_{\mathcal{I}, \mathcal{O}}$, $i \in \mathcal{I}$, $o \in \mathcal{O}$, and $\mathbf{x} \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$, let

$$\Phi_{i, o, \mathbf{x}}^{\mathbf{s}} := \{x_i \in \mathcal{X} \mid \mathbf{s}(\mathbf{x} \cup \{(i, x_i)\})(o) = x_i\}$$

be the set of fixed points of the function $x_i \mapsto \mathbf{s}(\mathbf{x} \cup \{(i, x_i)\})(o)$. Further let $\Gamma: \mathfrak{S} \rightarrow \mathcal{P}(\{\{i, o\} \mid i, o \in \Lambda\})$ such that for all $\mathbf{s} \in \mathfrak{S}_{\mathcal{I}, \mathcal{O}}$ and $\{i, o\} \in \Gamma(\mathbf{s})$, we have $i \in \mathcal{I}$, $o \in \mathcal{O}$ (or $i \in \mathcal{O}$, $o \in \mathcal{I}$),⁶ and for all $\mathbf{x} \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$, we have $\Phi_{i, o, \mathbf{x}}^{\mathbf{s}} \neq \emptyset$. Finally let $\phi_{i, o}^{\mathbf{s}}: \mathcal{X}^{\mathcal{I} \setminus \{i\}} \rightarrow \mathcal{X}$ for $\mathbf{s} \in \mathfrak{S}_{\mathcal{I}, \mathcal{O}}$ and $\{i, o\} \in \Gamma(\mathbf{s})$ be a function such that for all $\mathbf{x} \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$, we have $\phi_{i, o}^{\mathbf{s}}(\mathbf{x}) \in \Phi_{i, o, \mathbf{x}}^{\mathbf{s}}$. Then, we define

$$\mathfrak{F}(\mathfrak{S}, \Gamma, \phi) := (\mathfrak{S}, \lambda, \parallel, \Gamma, \gamma),$$

where ϕ is the set of all $\phi_{i, o}^{\mathbf{s}}$ and

- for $\mathbf{s} \in \mathfrak{S}_{\mathcal{I}, \mathcal{O}}$, we have $\lambda(\mathbf{s}) = \mathcal{I} \cup \mathcal{O}$,

⁶We formally cannot require $i \in \mathcal{I}$ and $o \in \mathcal{O}$ because $\{i, o\}$ is unordered. To simplify the notation we will, however, always assume that $i \in \mathcal{I}$ and $o \in \mathcal{O}$ when we write $\{i, o\} \in \Gamma(\mathbf{s})$, and similarly for $\{i', o'\}$, $\{i_1, o_1\}$, etc.

- for pairwise disjoint $\mathcal{I}_1, \mathcal{I}_2, \mathcal{O}_1, \mathcal{O}_2 \subseteq \Lambda$, $\mathbf{s}_1 \in \mathfrak{S}_{\mathcal{I}_1, \mathcal{O}_1}$, and $\mathbf{s}_2 \in \mathfrak{S}_{\mathcal{I}_2, \mathcal{O}_2}$, we have $\mathbf{s}_1 \parallel \mathbf{s}_2: \mathcal{X}^{\mathcal{I}_1 \cup \mathcal{I}_2} \rightarrow \mathcal{X}^{\mathcal{O}_1 \cup \mathcal{O}_2}$ with

$$\forall \mathbf{x} \in \mathcal{X}^{\mathcal{I}_1 \cup \mathcal{I}_2} \quad \forall j \in \{1, 2\} \quad \forall o_j \in \mathcal{O}_j \quad (\mathbf{s}_1 \parallel \mathbf{s}_2)(\mathbf{x})(o_j) = \mathbf{s}_j(\mathbf{x}|_{\mathcal{I}_j})(o_j), \quad (1)$$

- and for $\mathbf{s} \in \mathfrak{S}_{\mathcal{I}, \mathcal{O}}$ and $\{i, o\} \in \Gamma(\mathbf{s})$, we have $\gamma_{\{i, o\}}(\mathbf{s}): \mathcal{X}^{\mathcal{I} \setminus \{i\}} \rightarrow \mathcal{X}^{\mathcal{O} \setminus \{o\}}$ with

$$\forall \mathbf{x} \in \mathcal{X}^{\mathcal{I} \setminus \{i\}} \quad \gamma_{\{i, o\}}(\mathbf{s})(\mathbf{x}) = \mathbf{s}(\mathbf{x} \cup \{(i, \phi_{i, o}^{\mathbf{s}}(\mathbf{x}))\})|_{\mathcal{O} \setminus \{o\}}.$$

If \mathfrak{S} is closed under \parallel and γ ,⁷ and if for all $\mathbf{s}_1, \mathbf{s}_2 \in \mathfrak{S}$ with $\lambda(\mathbf{s}_1) \cap \lambda(\mathbf{s}_2) = \emptyset$, for $j \in \{1, 2\}$, and for all $i, o \in \lambda(\mathbf{s}_j)$, we have $\{i, o\} \in \Gamma(\mathbf{s}_1 \parallel \mathbf{s}_2) \iff \{i, o\} \in \Gamma(\mathbf{s}_j)$, then $\mathfrak{F}(\mathfrak{S}, \Gamma, \phi)$ is a Λ -system algebra. In this case, we say $\mathfrak{F}(\mathfrak{S}, \Gamma, \phi)$ is a *functional Λ -system algebra over \mathcal{X}* .

Remark. Our definition of interface connections via fixed points implies that the “line” connecting two systems has no effect on the values. This means in particular that it does not introduce delays or transmission errors. If, e.g., delays are required (and the domains and functions are defined at a level such that delays can be specified, see Section 6), one can allow the connection of systems only via dedicated channel systems that introduce the desired delays.

The choice of fixed points is crucial for obtaining a system algebra with desired properties. For example, if the system algebra is supposed to model a class of real-world systems, the chosen fixed point should correspond to the value generated by the real system. The choice of the fixed point can also influence whether connection-order invariance holds. If fixed points are not unique, a reasonable requirement is that they are consistently chosen in the sense that whenever two systems have the same set of fixed points for a specific interface connection, the same fixed point is chosen for both systems.

Definition 4.2. Let $(\mathfrak{S}, \lambda, \parallel, \Gamma, \gamma)$ be a functional Λ -system algebra over \mathcal{X} . We say it has *unique fixed points* if $|\Phi_{i, o, \mathbf{x}}^{\mathbf{s}}| = 1$ for all $\mathbf{s} \in \mathfrak{S}_{\mathcal{I}, \mathcal{O}}$, $\{i, o\} \in \Gamma(\mathbf{s})$, and $\mathbf{x} \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$. If there exists⁸ ϕ such that $(\mathfrak{S}, \lambda, \parallel, \Gamma, \gamma) = \mathfrak{F}(\mathfrak{S}, \Gamma, \phi)$ and for all $\mathbf{s} \in \mathfrak{S}_{\mathcal{I}, \mathcal{O}}, \mathbf{s}' \in \mathfrak{S}_{\mathcal{I}', \mathcal{O}'}, \{i, o\} \in \Gamma(\mathbf{s}) \cap \Gamma(\mathbf{s}')$, $\mathbf{x} \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$, and $\mathbf{x}' \in \mathcal{X}^{\mathcal{I}' \setminus \{i\}}$,

$$\Phi_{i, o, \mathbf{x}}^{\mathbf{s}} = \Phi_{i, o, \mathbf{x}'}^{\mathbf{s}'} \implies \phi_{i, o}^{\mathbf{s}}(\mathbf{x}) = \phi_{i, o}^{\mathbf{s}'}(\mathbf{x}'),$$

we say $(\mathfrak{S}, \lambda, \parallel, \Gamma, \gamma)$ has *consistently chosen fixed points*.

4.2 Basic Properties

It is easy to find functional system algebras that are not connection-order invariant if fixed points are not chosen consistently. Consider for example a system \mathbf{s} with two fixed points x, x' for connecting interfaces i and o regardless of inputs at other interfaces. It is possible that x is chosen for $\gamma_{\{i, o\}}(\mathbf{s})$ but for the system $\mathbf{s}' := \gamma_{\{i', o'\}}(\mathbf{s})$, which has the same two fixed points for connecting i and o , x' is chosen. In this case, one can have $\gamma_{\{i', o'\}}(\gamma_{\{i, o\}}(\mathbf{s})) \neq \gamma_{\{i, o\}}(\gamma_{\{i', o'\}}(\mathbf{s}))$. As the following example shows, consistently chosen fixed points are, however, not sufficient to guarantee connection-order invariance.

⁷That is, for $\mathbf{s}_1, \mathbf{s}_2 \in \mathfrak{S}$ for which \parallel is defined, $\mathbf{s}_1 \parallel \mathbf{s}_2 \in \mathfrak{S}$, and for $\mathbf{s} \in \mathfrak{S}$ and $\{i, o\} \in \Gamma(\mathbf{s})$, $\gamma_{\{i, o\}}(\mathbf{s}) \in \mathfrak{S}$.

⁸While ϕ uniquely determines γ , the converse is not true. For example, if \mathbf{s}, i , and o are such that the input at interface i does not influence the outputs of \mathbf{s} at interfaces different from o , the choice of $\phi_{i, o}^{\mathbf{s}}(\cdot)$ is irrelevant. Hence, we only require for consistently chosen fixed points that γ can be explained consistently.

Consistently chosen fixed points do not imply connection-order invariance. Let $\mathcal{X} = \{0, 1\}$ and consider a system algebra where 0 is preferred to 1 as a fixed point, i.e., $\Phi_{i,o,\mathbf{x}}^s = \{0, 1\} \Rightarrow \phi_{i,o}^s(\mathbf{x}) = 0$. This clearly guarantees consistently chosen fixed points. Now consider the system $\mathbf{s} \in \mathfrak{S}_{\{i_1, i_2\}, \{o_1, o_2, o_3\}}$, $\mathbf{s}(\{(i_1, x_1), (i_2, x_2)\}) = \{(o_1, 1 - x_1), (o_2, 1 - x_2), (o_3, x_1)\}$. If we connect i_2 and o_1 , there is exactly one fixed point for each input at i_1 because we essentially compose two functions, which both invert their input bit. Thus, $\gamma_{\{i_2, o_1\}}(\mathbf{s})(\{(i_1, x_1)\}) = \{(o_2, x_1), (o_3, x_1)\}$. If we now connect i_1 and o_2 , both 0 and 1 are fixed points and 0 is chosen, i.e., we get $\gamma_{\{i_1, o_2\}}(\gamma_{\{i_2, o_1\}}(\mathbf{s}))(\emptyset) = \{(o_3, 0)\}$. However, first connecting i_1 and o_2 yields $\gamma_{\{i_1, o_2\}}(\mathbf{s})(\{(i_2, x_2)\}) = \{(o_1, x_2), (o_3, 1 - x_2)\}$. If we then connect i_2 and o_1 , 0 is again the preferred fixed point and therefore $\gamma_{\{i_2, o_1\}}(\gamma_{\{i_1, o_2\}}(\mathbf{s}))(\emptyset) = \{(o_3, 1)\}$, which implies $\gamma_{\{i_2, o_1\}}(\gamma_{\{i_1, o_2\}}(\mathbf{s})) \neq \gamma_{\{i_1, o_2\}}(\gamma_{\{i_2, o_1\}}(\mathbf{s}))$.

While consistently chosen fixed points are not sufficient, the next lemma shows that connection-order invariance follows from unique fixed points if Γ permits reordering.

Lemma 4.3. *Functional system algebras with unique fixed points and Γ that permits reordering are connection-order invariant.*

Proof. Let Λ and \mathcal{X} be sets, and let $(\mathfrak{S}, \lambda, \|\cdot\|, \Gamma, \gamma) = \mathfrak{F}(\mathfrak{S}, \Gamma, \phi)$ be a functional Λ -system algebra over \mathcal{X} with unique fixed points such that Γ permits reordering. Further let $\mathbf{s} \in \mathfrak{S}$, $\{i, o\} \in \Gamma(\mathbf{s})$, and $\{i', o'\} \in \Gamma(\gamma_{\{i, o\}}(\mathbf{s}))$. We have to show that $\gamma_{\{i', o'\}}(\gamma_{\{i, o\}}(\mathbf{s})) = \gamma_{\{i, o\}}(\gamma_{\{i', o'\}}(\mathbf{s}))$. To this end, let $\mathbf{x} \in \mathcal{X}^{\mathcal{I} \setminus \{i, i'\}}$. By [Definition 4.1](#), there exist $x_i, x_{i'}, x'_i, x'_{i'} \in \mathcal{X}$ such that

$$\begin{aligned} \gamma_{\{i', o'\}}(\gamma_{\{i, o\}}(\mathbf{s}))(\mathbf{x}) &= \mathbf{s}(\mathbf{x} \cup \{(i, x_i), (i', x_{i'})\})|_{\mathcal{O} \setminus \{o, o'\}}, \\ \gamma_{\{i, o\}}(\gamma_{\{i', o'\}}(\mathbf{s}))(\mathbf{x}) &= \mathbf{s}(\mathbf{x} \cup \{(i, x'_i), (i', x'_{i'})\})|_{\mathcal{O} \setminus \{o, o'\}}. \end{aligned}$$

We further have $x_i \in \Phi_{i, o, \mathbf{x} \cup \{(i', x_{i'})\}}^s$ and since fixed points are unique, x_i is chosen for connecting i and o on input $\mathbf{x} \cup \{(i', x_{i'})\}$. This implies $x_{i'} \in \Phi_{i', o', \mathbf{x}}^{\gamma_{\{i, o\}}(\mathbf{s})}$. Moreover, $x'_i \in \Phi_{i, o, \mathbf{x} \cup \{(i', x'_{i'})\}}^s$ and thus $x'_{i'} \in \Phi_{i', o', \mathbf{x}}^{\gamma_{\{i, o\}}(\mathbf{s})}$. Because $|\Phi_{i', o', \mathbf{x}}^{\gamma_{\{i, o\}}(\mathbf{s})}| = 1$, this implies $x_{i'} = x'_{i'}$. Therefore, $\Phi_{i, o, \mathbf{x} \cup \{(i', x_{i'})\}}^s = \Phi_{i, o, \mathbf{x} \cup \{(i', x'_{i'})\}}^s$ and we obtain $x_i = x'_i$. We conclude $\gamma_{\{i', o'\}}(\gamma_{\{i, o\}}(\mathbf{s}))(\mathbf{x}) = \gamma_{\{i, o\}}(\gamma_{\{i', o'\}}(\mathbf{s}))(\mathbf{x})$. \square

We next prove that the parallel composition operation of functional system algebras is always associative and commutative.

Lemma 4.4. *For any functional system algebra, parallel composition is associative and commutative.*

Proof. Let Λ and \mathcal{X} be sets, and let $(\mathfrak{S}, \lambda, \|\cdot\|, \Gamma, \gamma) = \mathfrak{F}(\mathfrak{S}, \Gamma, \phi)$ be a functional Λ -system algebra over \mathcal{X} . Further let $\mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3, \mathcal{O}_1, \mathcal{O}_2, \mathcal{O}_3 \subseteq \Lambda$ be pairwise disjoint, let $\mathbf{s}_j \in \mathfrak{S}_{\mathcal{I}_j, \mathcal{O}_j}$ for $j \in \{1, 2, 3\}$, and let $\mathbf{x} \in \mathcal{X}^{\mathcal{I}_1 \cup \mathcal{I}_2 \cup \mathcal{I}_3}$ and $o \in \mathcal{O}_k$ for some $k \in \{1, 2, 3\}$. If $k = 3$, we have by equation (1), $((\mathbf{s}_1 \parallel \mathbf{s}_2) \parallel \mathbf{s}_3)(\mathbf{x})(o) = \mathbf{s}_3(\mathbf{x}|_{\mathcal{I}_3})(o)$. If $k \neq 3$, we have $((\mathbf{s}_1 \parallel \mathbf{s}_2) \parallel \mathbf{s}_3)(\mathbf{x})(o) = (\mathbf{s}_1 \parallel \mathbf{s}_2)(\mathbf{x}|_{\mathcal{I}_1 \cup \mathcal{I}_2})(o) = \mathbf{s}_k(\mathbf{x}|_{\mathcal{I}_k})(o)$. Similarly, $(\mathbf{s}_1 \parallel (\mathbf{s}_2 \parallel \mathbf{s}_3))(\mathbf{x})(o) = \mathbf{s}_k(\mathbf{x}|_{\mathcal{I}_k})(o)$. Hence, $(\mathbf{s}_1 \parallel \mathbf{s}_2) \parallel \mathbf{s}_3 = \mathbf{s}_1 \parallel (\mathbf{s}_2 \parallel \mathbf{s}_3)$, i.e., \parallel is associative. Commutativity follows directly from equation (1): for $\mathbf{x} \in \mathcal{X}^{\mathcal{I}_1 \cup \mathcal{I}_2}$ and $o \in \mathcal{O}_k$ for some $k \in \{1, 2\}$, we have $(\mathbf{s}_1 \parallel \mathbf{s}_2)(\mathbf{x})(o) = \mathbf{s}_k(\mathbf{x}|_{\mathcal{I}_k})(o) = (\mathbf{s}_2 \parallel \mathbf{s}_1)(\mathbf{x})(o)$. \square

We further show that connection-order invariance implies composition-order invariance for functional system algebras with consistently chosen fixed points.

Lemma 4.5. *A functional system algebra with consistently chosen fixed points is composition-order invariant if and only if it is connection-order invariant.*

Proof. By definition, a functional system algebra that is not connection-order invariant is not composition-order invariant. For the other direction, let $(\mathfrak{S}, \lambda, \parallel, \Gamma, \gamma) = \mathfrak{F}(\mathfrak{S}, \Gamma, \phi)$ be a connection-order invariant functional Λ -system algebra over \mathcal{X} with consistently chosen fixed points. Associativity and commutativity of \parallel follows from Lemma 4.4. It remains to show that $\gamma_{\{i,o\}}(\mathbf{s}_1) \parallel \mathbf{s}_2 = \gamma_{\{i,o\}}(\mathbf{s}_1 \parallel \mathbf{s}_2)$ for all $\mathbf{s}_1 \in \mathfrak{S}_{\mathcal{I}_1, \mathcal{O}_1}$, $\mathbf{s}_2 \in \mathfrak{S}_{\mathcal{I}_2, \mathcal{O}_2}$, and $\{i, o\} \in \Gamma(\mathbf{s}_1)$ such that $\lambda(\mathbf{s}_1) \cap \lambda(\mathbf{s}_2) = \emptyset$. To this end, let $\mathbf{x} \in \mathcal{X}^{(\mathcal{I}_1 \cup \mathcal{I}_2) \setminus \{i\}}$, $o_1 \in \mathcal{O}_1 \setminus \{o\}$, and $o_2 \in \mathcal{O}_2$. Because inputs to \mathbf{s}_1 do not affect the outputs of \mathbf{s}_2 , we have

$$(\gamma_{\{i,o\}}(\mathbf{s}_1) \parallel \mathbf{s}_2)(\mathbf{x})(o_2) = \mathbf{s}_2(\mathbf{x}|_{\mathcal{I}_2})(o_2) = (\gamma_{\{i,o\}}(\mathbf{s}_1 \parallel \mathbf{s}_2))(\mathbf{x})(o_2).$$

By the same reasoning, the set of fixed points for connecting interfaces i and o is not influenced by \mathbf{s}_2 , i.e., $\Phi_{i,o,\mathbf{x}}^{\mathbf{s}_1 \parallel \mathbf{s}_2} = \Phi_{i,o,\mathbf{x}|_{\mathcal{I}_1}}^{\mathbf{s}_1}$. Consistently chosen fixed points imply $\phi_{i,o}^{\mathbf{s}_1 \parallel \mathbf{s}_2}(\mathbf{x}) = \phi_{i,o}^{\mathbf{s}_1}(\mathbf{x}|_{\mathcal{I}_1})$ and hence

$$\begin{aligned} (\gamma_{\{i,o\}}(\mathbf{s}_1) \parallel \mathbf{s}_2)(\mathbf{x})(o_1) &= \gamma_{\{i,o\}}(\mathbf{s}_1)(\mathbf{x}|_{\mathcal{I}_1})(o_1) = \mathbf{s}_1\left(\mathbf{x}|_{\mathcal{I}_1} \cup \left\{ \left(i, \phi_{i,o}^{\mathbf{s}_1}(\mathbf{x}|_{\mathcal{I}_1}) \right) \right\}\right)(o_1) \\ &= \mathbf{s}_1\left(\mathbf{x}|_{\mathcal{I}_1} \cup \left\{ \left(i, \phi_{i,o}^{\mathbf{s}_1 \parallel \mathbf{s}_2}(\mathbf{x}) \right) \right\}\right)(o_1) \\ &= (\mathbf{s}_1 \parallel \mathbf{s}_2)\left(\mathbf{x} \cup \left\{ \left(i, \phi_{i,o}^{\mathbf{s}_1 \parallel \mathbf{s}_2}(\mathbf{x}) \right) \right\}\right)(o_1) \\ &= (\gamma_{\{i,o\}}(\mathbf{s}_1 \parallel \mathbf{s}_2))(\mathbf{x})(o_1). \end{aligned}$$

Therefore, $\gamma_{\{i,o\}}(\mathbf{s}_1) \parallel \mathbf{s}_2 = \gamma_{\{i,o\}}(\mathbf{s}_1 \parallel \mathbf{s}_2)$ and hence composition-order invariance holds. \square

Since unique fixed points imply consistently chosen fixed points, we can combine Lemma 4.3 and Lemma 4.5 to obtain the following result.

Theorem 4.6. *Functional system algebras with unique fixed points and Γ that permits reordering are composition-order invariant.*

4.3 Merging Input and Output Interfaces

It is often desirable to have a system algebra that allows arbitrary interface connections, as we have assumed in Section 3.2. A functional system algebra does not provide this because one can only connect input interfaces to output interfaces. A single interface connection in a functional system algebra also allows only communication in one direction, while one might prefer to connect systems to each other such that they can communicate in both directions. These drawbacks can be mitigated easily by considering an interface as a pair of an input interface and an output interface and by defining interface connection as connecting both interfaces of the corresponding pair: Let $(\mathfrak{S}, \lambda, \parallel, \Gamma, \gamma)$ be a functional $(\Lambda \times \{0, 1\})$ -system algebra that allows connecting arbitrary input interface to arbitrary output interfaces (i.e., $\Gamma(\mathbf{s}) = \{\{i, o\} \mid i \in \mathcal{I}, o \in \mathcal{O}\}$ for $\mathbf{s} \in \mathfrak{S}_{\mathcal{I}, \mathcal{O}}$). We then define the Λ -system algebra $(\mathfrak{S}', \lambda', \parallel', \Gamma', \gamma')$ via $\mathfrak{S}' := \{\mathbf{s} \in \mathfrak{S}_{\mathcal{L} \times \{1\}, \mathcal{L} \times \{0\}} \mid \mathcal{L} \subseteq \Lambda\}$, $\lambda'(\mathbf{s}) = \mathcal{L}$ for $\mathbf{s} \in \mathfrak{S}_{\mathcal{L} \times \{1\}, \mathcal{L} \times \{0\}}$, $\parallel' = \parallel$, $\Gamma(\mathbf{s}) = \{\{i, i'\} \mid i, i' \in \lambda(\mathbf{s})\}$, $\gamma'_{\{i,i'\}}(\mathbf{s}) = \gamma_{\{(i',1),(i,0)\}}(\gamma_{\{(i,1),(i',0)\}}(\mathbf{s}))$. See Figure 6 for a depiction of a system and the connection operation in $(\mathfrak{S}', \lambda', \parallel', \Gamma', \gamma')$. Note that the system algebra $(\mathfrak{S}', \lambda', \parallel', \Gamma', \gamma')$ is composition-order invariant if $(\mathfrak{S}, \lambda, \parallel, \Gamma, \gamma)$ is.

By doing this, one can use functional system algebras as the ones we develop in later sections to instantiate system algebras with undirected interfaces.

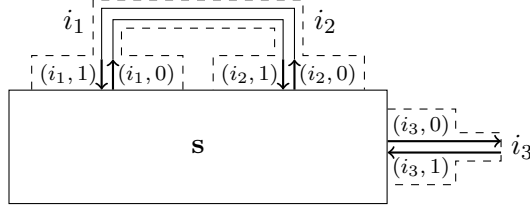


Figure 6: The system $\gamma'_{\{i_1, i_2\}}(\mathbf{s})$ in the system algebra with merged input and output interfaces.

Remark. As defined above, Γ' allows connecting an interface to itself. While this is well-defined, one might want to exclude it since it is typically not needed.

5 The System Algebras of Monotone and Continuous Systems

By [Theorem 2.13](#) and [Theorem 2.17](#), monotone functions on CPOs and ω -continuous functions on ω -CPOs have least fixed points. We can use this to define a functional system algebra consisting of such functions. While fixed points of monotone functions are well-studied in domain theory, we still need to show that choosing the least fixed point yields a functional system algebra with the desired properties. To obtain a system algebra, we need to prove that the set of system is closed under interface connections defined via choosing the least fixed point. We prove that this is the case both for monotone and continuous functions. Furthermore, we show that composition-order invariance holds for both system algebras under the additional assumption on the CPO that every nonempty chain has an infimum. Finally, we describe Kahn networks as a special case of the system algebra of continuous systems.

5.1 Monotone Systems

Let (\mathcal{X}, \preceq) be a CPO. Recall that for a set \mathcal{I} , we also write \preceq for the partial order on $\mathcal{X}^{\mathcal{I}}$ where for $\mathbf{x}, \mathbf{y} \in \mathcal{X}^{\mathcal{I}}$, $\mathbf{x} \preceq \mathbf{y}$ if $\mathbf{x}(i) \preceq \mathbf{y}(i)$ for all $i \in \mathcal{I}$. Monotonicity is therefore also defined for functions on such tuples. Let Λ be a set and define for finite disjoint $\mathcal{I}, \mathcal{O} \subseteq \Lambda$,

$$\mathfrak{M}_{\mathcal{I}, \mathcal{O}} := \{\mathbf{s}: \mathcal{X}^{\mathcal{I}} \rightarrow \mathcal{X}^{\mathcal{O}} \mid \mathbf{s} \text{ is monotone}\}.$$

We can use the existence of least fixed points to define a system algebra. Before we do so, we need the following simple lemma.

Lemma 5.1. *Let $\mathcal{I}, \mathcal{O} \subseteq \Lambda$ be finite disjoint sets, let $\mathbf{s} \in \mathfrak{M}_{\mathcal{I}, \mathcal{O}}$, and let $i \in \mathcal{I}$, $o \in \mathcal{O}$. Then, for all $\mathbf{x} \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$, there exists a least $x_i \in \mathcal{X}$ such that $\mathbf{s}(\mathbf{x} \cup \{(i, x_i)\})(o) = x_i$.*

Proof. For $\mathbf{x} \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$, let $f: \mathcal{X} \rightarrow \mathcal{X}, x_i \mapsto \mathbf{s}(\mathbf{x} \cup \{(i, x_i)\})(o)$ and note that the fixed points of f precisely correspond to the values $x_i \in \mathcal{X}$ such that $\mathbf{s}(\mathbf{x} \cup \{(i, x_i)\})(o) = x_i$. Since \mathbf{s} is monotone, so is f , and therefore f has a least fixed point by [Theorem 2.13](#). \square

Definition 5.2. We define $\mathfrak{F}(\mathfrak{M}, \Gamma, \phi)$, the Λ -system algebra of monotone systems over \mathcal{X} , as follows: For $\mathbf{s} \in \mathfrak{M}_{\mathcal{I}, \mathcal{O}}$, $\Gamma(\mathbf{s}) := \{\{i, o\} \mid i \in \mathcal{I}, o \in \mathcal{O}\}$ and $\phi_{i, o}^{\mathbf{s}}(\mathbf{x})$ is the least element of $\Phi_{i, o, \mathbf{x}}^{\mathbf{s}}$ for all $i \in \mathcal{I}$, $o \in \mathcal{O}$, and $\mathbf{x} \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$.

For this to actually be a functional system algebra, we need that the parallel composition of systems and connecting interfaces of a system again yield a system in our algebra, i.e., a monotone function. While this is straightforward for parallel composition, it is nontrivial for interface connections. The essence of the proof we provide below is showing that $\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_1) \preceq \phi_{i,o}^{\mathbf{s}}(\mathbf{x}_2)$ whenever $\mathbf{x}_1 \preceq \mathbf{x}_2$. To this end, we exploit the constructive nature of [Theorem 2.13](#) that can be used to find the fixed points $\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_1)$ and $\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_2)$.

Theorem 5.3. $(\mathfrak{M}, \lambda, \|\cdot\|, \Gamma, \gamma) = \mathfrak{F}(\mathfrak{M}, \Gamma, \phi)$ is a functional Λ -system algebra over \mathcal{X} .

Proof. Since the relation \preceq is defined componentwise, it is clear from the definition of parallel composition by equation (1) that \mathfrak{M} is closed under $\|\cdot\|$. It remains to show that \mathfrak{M} is also closed under connecting interfaces. To this end, let $\mathcal{I}, \mathcal{O} \subseteq \Lambda$ be finite disjoint sets and let $\mathbf{s} \in \mathfrak{M}_{\mathcal{I}, \mathcal{O}}$. Further let $i \in \mathcal{I}$, $o \in \mathcal{O}$. We have to show that $\gamma_{\{i,o\}}(\mathbf{s}) \in \mathfrak{M}_{\mathcal{I} \setminus \{i\}, \mathcal{O} \setminus \{o\}}$. To show that $\gamma_{\{i,o\}}(\mathbf{s})$ is monotone, let $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$, such that $\mathbf{x}_1 \preceq \mathbf{x}_2$. We then have $\gamma_{\{i,o\}}(\mathbf{s})(\mathbf{x}_k) = \mathbf{s}(\mathbf{x}_k \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{x}_k))\})|_{\mathcal{O} \setminus \{o\}}$ for $k \in \{1, 2\}$, where $\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_k)$ is the least fixed point of the function $x_i \mapsto \mathbf{s}(\mathbf{x}_k \cup \{(i, x_i)\})(o)$. Let x_k^α for ordinals α be as in [Theorem 2.13](#) such that $\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_k) = x_k^{\hat{\alpha}_k}$ for some ordinal $\hat{\alpha}_k$. We show by transfinite induction over α that $x_1^\alpha \preceq x_2^\alpha$ for all ordinals α .

(i) For $\alpha = 0$, we have $x_1^\alpha = \min(\mathcal{X}) = x_2^\alpha$.

(ii) Assume $x_1^\alpha \preceq x_2^\alpha$ for some ordinal α . We then have $\mathbf{x}_1 \cup \{(i, x_1^\alpha)\} \preceq \mathbf{x}_2 \cup \{(i, x_2^\alpha)\}$ and since \mathbf{s} is monotone,

$$x_1^{\alpha+1} = \mathbf{s}(\mathbf{x}_1 \cup \{(i, x_1^\alpha)\})(o) \preceq \mathbf{s}(\mathbf{x}_2 \cup \{(i, x_2^\alpha)\})(o) = x_2^{\alpha+1}.$$

(iii) Let α be a nonzero limit ordinal and assume $x_1^\beta \preceq x_2^\beta$ for all $\beta < \alpha$. We then have

$$x_1^\alpha = \sup\{x_1^\beta \mid \beta < \alpha\} \preceq \sup\{x_2^\beta \mid \beta < \alpha\} = x_2^\alpha.$$

Thus, $x_1^\alpha \preceq x_2^\alpha$ for all ordinals α . It is easy to see (by transfinite induction) that $x_1^\alpha = x_1^{\hat{\alpha}_1}$ for all $\alpha \geq \hat{\alpha}_1$, and $x_2^\alpha = x_2^{\hat{\alpha}_2}$ for all $\alpha \geq \hat{\alpha}_2$. We can therefore conclude

$$\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_1) = x_1^{\hat{\alpha}_1} = x_1^{\max\{\hat{\alpha}_1, \hat{\alpha}_2\}} \preceq x_2^{\max\{\hat{\alpha}_1, \hat{\alpha}_2\}} = x_2^{\hat{\alpha}_2} = \phi_{i,o}^{\mathbf{s}}(\mathbf{x}_2).$$

Using monotonicity of \mathbf{s} , this implies

$$\gamma_{\{i,o\}}(\mathbf{s})(\mathbf{x}_1) = \mathbf{s}(\mathbf{x}_1 \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{x}_1))\})|_{\mathcal{O} \setminus \{o\}} \preceq \mathbf{s}(\mathbf{x}_2 \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{x}_2))\})|_{\mathcal{O} \setminus \{o\}} = \gamma_{\{i,o\}}(\mathbf{s})(\mathbf{x}_2).$$

Hence, $\gamma_{\{i,o\}}(\mathbf{s})$ is monotone and therefore $\gamma_{\{i,o\}}(\mathbf{s}) \in \mathfrak{M}_{\mathcal{I} \setminus \{i\}, \mathcal{O} \setminus \{o\}}$. \square

5.2 Continuous Systems

Let Λ be a set and let (\mathcal{X}, \preceq) be an ω -CPO. It is easily verified that $\mathcal{X}^{\mathcal{I}}$ is again an ω -CPO for $\mathcal{I} \subseteq \Lambda$, and the supremum of an ω -chain in $\mathcal{X}^{\mathcal{I}}$ corresponds to the tuple of the component-wise suprema. We define for finite disjoint $\mathcal{I}, \mathcal{O} \subseteq \Lambda$,

$$\mathfrak{N}_{\mathcal{I}, \mathcal{O}} := \{\mathbf{s}: \mathcal{X}^{\mathcal{I}} \rightarrow \mathcal{X}^{\mathcal{O}} \mid \mathbf{s} \text{ is } \omega\text{-continuous}\}^9$$

The following lemma can be proven analogously to [Lemma 5.1](#).

⁹We use the symbol \mathfrak{N} because continuity is stronger than monotonicity and \mathfrak{C} will be used for the system algebra of causal systems in [Section 6](#).

Lemma 5.4. Let $\mathcal{I}, \mathcal{O} \subseteq \Lambda$ be finite disjoint sets, let $\mathbf{s} \in \mathfrak{N}_{\mathcal{I}, \mathcal{O}}$, and let $i \in \mathcal{I}$, $o \in \mathcal{O}$. Then, for all $\mathbf{x} \in \mathcal{X}^{\mathcal{T} \setminus \{i\}}$, there exists a least $x_i \in \mathcal{X}$ such that $\mathbf{s}(\mathbf{x} \cup \{(i, x_i)\})(o) = x_i$.

Definition 5.5. We define $\mathfrak{F}(\mathfrak{N}, \Gamma, \phi)$, the Λ -system algebra of continuous systems over \mathcal{X} , as follows: For $\mathbf{s} \in \mathfrak{N}_{\mathcal{I}, \mathcal{O}}$, $\Gamma(\mathbf{s}) := \{\{i, o\} \mid i \in \mathcal{I}, o \in \mathcal{O}\}$ and $\phi_{i,o}^{\mathbf{s}}(\mathbf{x})$ is the least element of $\Phi_{i,o,\mathbf{x}}^{\mathbf{s}}$ for all $i \in \mathcal{I}$, $o \in \mathcal{O}$, and $\mathbf{x} \in \mathcal{X}^{\mathcal{T} \setminus \{i\}}$.

We again need to show that connecting interfaces of a system yields a system in our algebra, i.e., in this case an ω -continuous function. We do so by proving that for an ω -chain $(\mathbf{x}_n)_{n \in \omega}$, $\{\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n) \mid n \in \omega\}$ is an ω -chain as well, and $\phi_{i,o}^{\mathbf{s}}(\sup\{\mathbf{x}_n \mid n \in \omega\}) = \sup\{\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n) \mid n \in \omega\}$. We then use ω -continuity of \mathbf{s} to conclude that $\gamma_{\{i,o\}}(\mathbf{s})$ is ω -continuous.

Theorem 5.6. $(\mathfrak{N}, \lambda, \|\cdot\|, \Gamma, \gamma) = \mathfrak{F}(\mathfrak{N}, \Gamma, \phi)$ is a functional Λ -system algebra over \mathcal{X} .

Proof. As in the proof of [Theorem 5.3](#), it is straightforward to verify that \mathfrak{N} is closed under parallel composition. To show that \mathfrak{N} is closed under connecting interfaces, let $\mathcal{I}, \mathcal{O} \subseteq \Lambda$ be finite disjoint sets and let $\mathbf{s} \in \mathfrak{N}_{\mathcal{I}, \mathcal{O}}$. Further let $i \in \mathcal{I}$, $o \in \mathcal{O}$. To show that $\gamma_{\{i,o\}}(\mathbf{s}) \in \mathfrak{N}_{\mathcal{I} \setminus \{i\}, \mathcal{O} \setminus \{o\}}$, let $(\mathbf{x}_n)_{n \in \omega}$ be an ω -chain in $\mathcal{X}^{\mathcal{T} \setminus \{i\}}$. We then have $\gamma_{\{i,o\}}(\mathbf{s})(\mathbf{x}_n) = \mathbf{s}(\mathbf{x}_n \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n))\})|_{\mathcal{O} \setminus \{o\}}$ for all $n \in \omega$, where $\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n)$ is the least fixed point of the function $x_i \mapsto \mathbf{s}(\mathbf{x}_n \cup \{(i, x_i)\})(o)$. Let $\hat{\mathbf{x}} := \sup\{\mathbf{x}_n \mid n \in \omega\}$. Since \mathbf{s} is monotone by [Lemma 2.16](#), one can show as in the proof of [Theorem 5.3](#) that $\phi_{i,o}^{\mathbf{s}}(\mathbf{x}) \preceq \phi_{i,o}^{\mathbf{s}}(\mathbf{x}')$ for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}^{\mathcal{T} \setminus \{i\}}$ with $\mathbf{x} \preceq \mathbf{x}'$. Thus, $\{\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n) \mid n \in \omega\}$ is an ω -chain. Because we also have $\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n) \preceq \phi_{i,o}^{\mathbf{s}}(\hat{\mathbf{x}})$ for all $n \in \omega$, we obtain $\sup\{\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n) \mid n \in \omega\} \preceq \phi_{i,o}^{\mathbf{s}}(\hat{\mathbf{x}})$. Using ω -continuity of \mathbf{s} , we further have

$$\begin{aligned} \mathbf{s}\left(\hat{\mathbf{x}} \cup \left\{(i, \sup\{\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n) \mid n \in \omega\})\right\}\right)(o) &= \mathbf{s}\left(\sup\left\{\mathbf{x}_n \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n))\} \mid n \in \omega\right\}\right)(o) \\ &= \sup\left\{\mathbf{s}\left(\mathbf{x}_n \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n))\}\right)(o) \mid n \in \omega\right\} \\ &= \sup\{\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n) \mid n \in \omega\}. \end{aligned}$$

This implies that $\sup\{\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n) \mid n \in \omega\}$ is a fixed point of $x_i \mapsto \mathbf{s}(\hat{\mathbf{x}} \cup \{(i, x_i)\})(o)$. Since $\phi_{i,o}^{\mathbf{s}}(\hat{\mathbf{x}})$ is the least fixed point of that function, we have $\phi_{i,o}^{\mathbf{s}}(\hat{\mathbf{x}}) \preceq \sup\{\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n) \mid n \in \omega\}$, and thus $\phi_{i,o}^{\mathbf{s}}(\hat{\mathbf{x}}) = \sup\{\phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n) \mid n \in \omega\}$. Together with ω -continuity of \mathbf{s} , this implies

$$\begin{aligned} \gamma_{\{i,o\}}(\mathbf{s})(\sup\{\mathbf{x}_n \mid n \in \omega\}) &= \gamma_{\{i,o\}}(\mathbf{s})(\hat{\mathbf{x}}) = \mathbf{s}(\hat{\mathbf{x}} \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\hat{\mathbf{x}}))\})|_{\mathcal{O} \setminus \{o\}} \\ &= \mathbf{s}\left(\sup\left\{\mathbf{x}_n \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n))\} \mid n \in \omega\right\}\right)|_{\mathcal{O} \setminus \{o\}} \\ &= \sup\left\{\mathbf{s}\left(\mathbf{x}_n \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{x}_n))\}\right)|_{\mathcal{O} \setminus \{o\}} \mid n \in \omega\right\} \\ &= \sup\{\gamma_{\{i,o\}}(\mathbf{s})(\mathbf{x}_n) \mid n \in \omega\}. \end{aligned}$$

We conclude that $\gamma_{\{i,o\}}(\mathbf{s})$ is ω -continuous and therefore $\gamma_{\{i,o\}}(\mathbf{s}) \in \mathfrak{N}_{\mathcal{I} \setminus \{i\}, \mathcal{O} \setminus \{o\}}$. \square

5.3 Composition-Order Invariance of Monotone and Continuous Systems

We want to show that the system algebras of monotone and continuous systems are composition-order invariant. We can do so if we additionally require that all nonempty chains in \mathcal{X} have an infimum. Before we prove the main result of this section, we need the following lemma about fixed points (cf. [\[GD03, \(A.6\) in Chapter I, §2.6\]](#)).

Lemma 5.7. *Let (\mathcal{X}, \preceq) be a poset such that every nonempty chain in \mathcal{X} has an infimum. Let $f: \mathcal{X} \rightarrow \mathcal{X}$ be a monotone function and let $x \in \mathcal{X}$ such that $f(x) \preceq x$. Then there exists $\hat{x} \in \mathcal{X}$ such that $f(\hat{x}) = \hat{x} \preceq x$.*

Proof. Let $x_0 := x$, $x_{\alpha+1} := f(x_\alpha)$ for any ordinal α , and $x_\alpha := \inf\{x_\beta \mid \beta < \alpha\}$ for nonzero limit ordinals α (where we show below that this infimum exists). We claim that we have for all ordinals α

$$\forall \beta \leq \alpha \quad f(x_\alpha) \preceq x_\alpha \preceq x_\beta.$$

We prove this claim by transfinite induction over α . For $\alpha = 0$, the claim follows since $f(x) \preceq x$. Assume the claim is true for some ordinal α . We then have $x_{\alpha+1} = f(x_\alpha) \preceq x_\alpha$. For $\beta \leq \alpha$, we therefore have by assumption $x_{\alpha+1} \preceq x_\alpha \preceq x_\beta$. For $\beta = \alpha + 1$, we trivially have $x_{\alpha+1} \preceq x_\beta$. Since f is monotone, we thus have for all $\beta \leq \alpha + 1$,

$$f(x_{\alpha+1}) \preceq f(x_\alpha) = x_{\alpha+1} \preceq x_\beta.$$

Now let α be a nonzero limit ordinal and assume that for all $\alpha' < \alpha$, we have $f(x_{\alpha'}) \preceq x_{\alpha'} \preceq x_\beta$ for all $\beta \leq \alpha'$. This implies that $\{x_\beta \mid \beta < \alpha\}$ is a chain and therefore the infimum exists. We then have by definition of x_α that $x_\alpha \preceq x_\beta$ for all $\beta \leq \alpha$. By monotonicity of f , we further have for all $\alpha' < \alpha$,

$$f(x_\alpha) = f(\inf\{x_\beta \mid \beta < \alpha\}) \preceq f(x_{\alpha'}) \preceq x_{\alpha'}.$$

Hence, $f(x_\alpha)$ is a lower bound of $\{x_\beta \mid \beta < \alpha\}$, and therefore $f(x_\alpha) \preceq \inf\{x_\beta \mid \beta < \alpha\} = x_\alpha \preceq x_\beta$ for all $\beta \leq \alpha$. Thus, the claim holds for all ordinals α .

Our claim and [Lemma 2.9](#) applied to the poset (\mathcal{X}, \succeq) yield that there exists an ordinal $\hat{\alpha}$ such that $x_{\hat{\alpha}} = x_{\hat{\alpha}+1}$. This implies $f(x_{\hat{\alpha}}) = x_{\hat{\alpha}+1} = x_{\hat{\alpha}} \preceq x_0 = x$. \square

We now use this lemma to prove composition-order invariance. For showing connection-order invariance, we consider a system \mathbf{s} and connecting interfaces i to o , and i' to o' . For an input \mathbf{x} at the remaining interfaces, consider the function $f: \mathcal{X}^2 \rightarrow \mathcal{X}^2$ that maps (x_0, x_1) to the outputs of \mathbf{s} at interfaces o and o' on input $\mathbf{x} \cup \{(i, x_0), (i', x_1)\}$. When we first connect i to o , and then i' to o' , the system algebra chooses fixed points x_i and $x_{i'}$ for the two connections, where $(x_i, x_{i'})$ is a fixed point of f . We show that it is the least fixed point, and that the same is true for connecting the interfaces in the other order, which implies that the two resulting systems are equal.

Note that $(x_i, x_{i'})$ being the least fixed point of f is not trivial: We know by definition of the connection operation that x_i is the least fixed point of $f(\cdot, x_{i'})(0)$, but for some $x_1 \prec x_{i'}$, the function $f(\cdot, x_1)(0)$ could have a fixed point $x_0 \prec x_i$ such that (x_0, x_1) is a fixed point of f that does not get chosen for the connections since x_0 is not the least fixed point of $f(\cdot, x_1)(0)$. Our proof proceeds as follows: For the least fixed point (\hat{x}_0, \hat{x}_1) of f , we prove that \hat{x}_0 is the least fixed point of $f(\cdot, \hat{x}_1)(0)$ by showing that if there were a fixed point less than \hat{x}_0 , we could use the lemma above to obtain a fixed point of f less than (\hat{x}_0, \hat{x}_1) . We then use this to conclude that $(x_i, x_{i'}) = (\hat{x}_0, \hat{x}_1)$.

Theorem 5.8. *Let Λ be a set, let (\mathcal{X}, \preceq) be a CPO and let $(\mathfrak{S}, \lambda, \|\cdot\|, \Gamma, \gamma) = \mathfrak{F}(\mathfrak{M}, \Gamma, \phi)$ be the Λ -system algebra of monotone systems over \mathcal{X} , or let (\mathcal{X}, \preceq) be an ω -CPO and let $(\mathfrak{S}, \lambda, \|\cdot\|, \Gamma, \gamma) = \mathfrak{F}(\mathfrak{N}, \Gamma, \phi)$ be the Λ -system algebra of continuous systems over \mathcal{X} . If every nonempty chain in \mathcal{X} has an infimum, the system algebra is composition-order invariant.*

Proof. Since the least fixed points are chosen, $\mathfrak{F}(\mathfrak{M}, \Gamma, \phi)$ and $\mathfrak{F}(\mathfrak{N}, \Gamma, \phi)$ have consistently chosen fixed points. Hence, [Lemma 4.5](#) implies that we only have to prove connection-order invariance. Let $\mathcal{I}, \mathcal{O} \subseteq \Lambda$ be finite disjoint sets and let $\mathbf{s} \in \mathfrak{S}_{\mathcal{I}, \mathcal{O}}$. Further let $i, i' \in \mathcal{I}$ and $o, o' \in \mathcal{O}$ be distinct elements. We have to show that $\gamma_{\{i', o'\}}(\gamma_{\{i, o\}}(\mathbf{s})) = \gamma_{\{i, o\}}(\gamma_{\{i', o'\}}(\mathbf{s}))$. To this end, let $\mathbf{x} \in \mathcal{X}^{\mathcal{I} \setminus \{i, i'\}}$. By [Definition 4.1](#), there exist $x_i, x_{i'}, x'_i, x'_{i'} \in \mathcal{X}$ such that

$$\begin{aligned}\gamma_{\{i', o'\}}(\gamma_{\{i, o\}}(\mathbf{s}))(\mathbf{x}) &= \mathbf{s}(\mathbf{x} \cup \{(i, x_i), (i', x_{i'})\})|_{\mathcal{O} \setminus \{o, o'\}}, \\ \gamma_{\{i, o\}}(\gamma_{\{i', o'\}}(\mathbf{s}))(\mathbf{x}) &= \mathbf{s}(\mathbf{x} \cup \{(i, x'_i), (i', x'_{i'})\})|_{\mathcal{O} \setminus \{o, o'\}}.\end{aligned}$$

Let $f: \mathcal{X}^2 \rightarrow \mathcal{X}^2$ be the function with

$$f(x_0, x_1) = \left(\mathbf{s}(\mathbf{x} \cup \{(i, x_0), (i', x_1)\})(o), \mathbf{s}(\mathbf{x} \cup \{(i, x_0), (i', x_1)\})(o') \right).$$

We then have by definition of ϕ and γ that x_i is the least fixed point of $f(\cdot, x_{i'})(0)$ and that $x'_{i'}$ is the least fixed point of $f(x'_i, \cdot)(1)$. We further have that $(x_i, x_{i'})$ and $(x'_i, x'_{i'})$ are fixed points of f . We will show that $(x_i, x_{i'})$ and $(x'_i, x'_{i'})$ are the least fixed point of f and therefore equal.

Note that if (\mathcal{X}, \preceq) is a CPO or an ω -CPO, then (\mathcal{X}^2, \preceq) is a CPO or an ω -CPO, respectively. Moreover, if \mathbf{s} is monotone, so is f , and if \mathbf{s} is ω -continuous, so is f ; in both cases, f is monotone by [Lemma 2.16](#). Hence, f has a least fixed point (\hat{x}_0, \hat{x}_1) by [Theorem 2.13](#) and [Theorem 2.17](#). This implies that \hat{x}_0 is a fixed point of $f(\cdot, \hat{x}_1)(0)$. We claim that it is the least fixed point of that function. Assume toward a contradiction that $\tilde{x}_0 \prec \hat{x}_0$ is the least fixed point of $f(\cdot, \hat{x}_1)(0)$. We then have by monotonicity of f for some $y_1 \in \mathcal{X}$,

$$(\tilde{x}_0, y_1) = f(\tilde{x}_0, \hat{x}_1) \preceq f(\hat{x}_0, \hat{x}_1) = (\hat{x}_0, \hat{x}_1).$$

Therefore, $f(\tilde{x}_0, \hat{x}_1) = (\tilde{x}_0, y_1) \preceq (\tilde{x}_0, \hat{x}_1)$. Thus, [Lemma 5.7](#) implies that f has a fixed point $(\bar{x}_0, \bar{x}_1) \preceq (\tilde{x}_0, \hat{x}_1) \prec (\hat{x}_0, \hat{x}_1)$, which contradicts (\hat{x}_0, \hat{x}_1) being the least fixed point of f .

Since \hat{x}_0 is the least fixed point of $f(\cdot, \hat{x}_1)(0)$ and (\hat{x}_0, \hat{x}_1) is a fixed point of f , we have $\hat{x}_1 \in \Phi_{i', o', \mathbf{x}}^{\gamma_{\{i, o\}}(\mathbf{s})}$. We also have that $x_{i'}$ is the least element of $\Phi_{i', o', \mathbf{x}}^{\gamma_{\{i, o\}}(\mathbf{s})}$ and $\hat{x}_1 \preceq x_{i'}$. This implies $\hat{x}_1 = x_{i'}$. As we have seen before, x_i is the least fixed point of $f(\cdot, x_{i'})(0)$ and \hat{x}_0 is the least fixed point of $f(\cdot, \hat{x}_1)(0)$. Thus, $x_i = \hat{x}_0$. An analogous argument shows that $(x'_i, x'_{i'}) = (\hat{x}_0, \hat{x}_1) = (x_i, x_{i'})$. We therefore have $\gamma_{\{i', o'\}}(\gamma_{\{i, o\}}(\mathbf{s}))(\mathbf{x}) = \gamma_{\{i, o\}}(\gamma_{\{i', o'\}}(\mathbf{s}))(\mathbf{x})$. \square

5.4 Kahn Networks

Kahn networks [[Kah74](#)] are an instantiation of the system algebra of continuous systems. There, the set \mathcal{X} consists of sequences over some set of values \mathcal{V} . The interpretation is that the systems successively take inputs from \mathcal{V} at each interface and produce outputs in \mathcal{V} . An input $(v_1, v_2, v_3) \in \mathcal{X}$ to an interface then corresponds to the input history where v_1 was input first, then v_2 , and finally v_3 . Hence, in this model the order of inputs at each interface is relevant, but the order of inputs at different interfaces is not modeled.

More formally, let \mathcal{V} be some nonempty set of values, let

$$\mathcal{X} := \{(v_\alpha)_{\alpha \in \beta} \in \mathcal{V}^\beta \mid \beta \leq \omega\},$$

and let \sqsubseteq be the initial segment relation on \mathcal{X} , i.e.,

$$(v_\alpha^1)_{\alpha \in \beta_1} \sqsubseteq (v_\alpha^2)_{\alpha \in \beta_2} \iff \beta_1 \leq \beta_2 \wedge \forall \alpha \in \beta_1 (v_\alpha^1 = v_\alpha^2).$$

It is straightforward to verify that $(\mathcal{X}, \sqsubseteq)$ is a CPO: For a chain $\mathcal{C} \subseteq \mathcal{X}$, the supremum of \mathcal{C} is the sequence v in which for $\alpha \leq \omega$, v_α is defined if and only if there is a sequence v' in \mathcal{C} in which v'_α is defined, and in this case, $v_\alpha = v'_\alpha$. This is well-defined since all elements in \mathcal{C} are comparable and thus, if two sequences in \mathcal{C} define v'_α and \tilde{v}_α , we have $v'_\alpha = \tilde{v}_\alpha$. One can therefore consider the system algebra of continuous systems over \mathcal{X} . Such systems correspond to Kahn networks [Kah74]. By Theorem 5.8, the system algebra is composition-order invariant since every chain also has an infimum (replace “there is a sequence in \mathcal{C} ” by “for all sequences in \mathcal{C} ” in the argument for the supremum).

Example. As a simple example of a system in this model, consider the system \mathbf{s} with input interface i and output interfaces o_1, o_2 that forwards all inputs to both o_1 and o_2 . Formally, the system is described by the function $\mathbf{s}: \mathcal{X}^{\{i\}} \rightarrow \mathcal{X}^{\{o_1, o_2\}}$ with

$$\mathbf{s}(\mathbf{x})(o_1) = \mathbf{s}(\mathbf{x})(o_2) = \mathbf{x}(i).$$

This function is clearly ω -continuous and therefore \mathbf{s} is a Kahn network. We can now connect interfaces i and o_1 to obtain the system $\gamma_{\{i, o_1\}}(\mathbf{s})$ with no input interfaces and a single output interface o_2 . To determine the output at that interface, we have to find the least fixed point of the function $x_i \mapsto \mathbf{s}(\{(i, x_i)\})(o_1) = x_i$. Since all $x \in \mathcal{X}$ are fixed points of this function, the least fixed point is $\min \mathcal{X}$, i.e., the empty sequence. This also matches the intuition that $\gamma_{\{i, o_1\}}(\mathbf{s})$ should not output anything because every other output would have to be caused by itself.

Generalizations. Kahn only considered ω -continuous systems and argued that continuity in contrast to monotonicity prevents a system from only outputting a value after receiving infinitely many inputs [Kah74]. Note, however, that allowing all monotone systems also yields a valid composition-order invariant system algebra.

Another generalization is to not only consider finite and infinite sequences, but also include transfinite sequences of inputs and outputs. By doing so, one can model systems that output something after infinitely many other outputs. This makes sense if one considers systems that can produce infinitely many outputs in finite time. Conversely, one can also place a finite upper bound on the number of inputs and outputs per interface, i.e., $\mathcal{X} := \{(v_\alpha)_{\alpha \in \beta} \in \mathcal{V}^\beta \mid \beta \leq n\}$ for some $n \in \mathbb{N}$. The special case $n = 1$ has been considered by Tackmann [Tac14]. Note, however, that it is not possible in this model to allow arbitrarily many but only finitely many inputs and outputs, because this does not yield an ω -CPO: The supremum of an ω -chain $C_0 \sqsubseteq C_1 \sqsubseteq C_2 \sqsubseteq \dots$, where C_n contains n elements, is infinite.

6 The System Algebra of Causal Systems

6.1 Causal Systems

We now develop a system algebra where the inputs and outputs of systems are (partially) ordered, i.e., a single input or output of a system is an element of some poset (P, \preceq) . An input or output history at some interface is then a subset of P . We interpret $p_1 \prec p_2$ as p_2 *could* causally depend on p_1 , but not vice versa. This can (but does not need to) mean that p_1 occurred at an earlier time than p_2 . For the ease of presentation, we will say that p_1 is before p_2 and p_2 is after p_1 if $p_1 \prec p_2$, keeping in mind that \preceq does not necessarily correspond to a relation on time. As a

natural instantiation in which time is made explicit and \prec does mean at an earlier time, consider $P = \mathcal{V} \times \mathcal{T}$ for some set \mathcal{V} and a poset (\mathcal{T}, \leq) and let

$$(v_1, t_1) \preceq (v_2, t_2) \quad :\iff \quad t_1 < t_2 \vee (v_1, t_1) = (v_2, t_2). \quad (2)$$

We can then interpret (v, t) as the value v being input or output at time t .

We again fix some arbitrary set Λ of interface labels and define a functional system algebra where a system is a function that for each input interface takes an input history as an input and outputs the output history for each output interface. That is, we define a functional system algebra over \mathcal{X} , where \mathcal{X} consists of subsets of P . We only allow well-ordered subsets, which means that at each input interface, if there are inputs, there is a first one, and one input is given after another. Inputs at different interfaces (which can be connected to different systems) can however be incomparable (e.g., occur at the same time). Only considering well-ordered inputs per interface seems to be a reasonable restriction if systems in computer science are to be modeled, because such systems are typically discrete and are started at some point, i.e., there is a first output. This restriction is discussed further below. Formally, we define

$$\mathcal{X} := \{X \subseteq P \mid (X, \preceq) \text{ is well-ordered}\}.$$

The systems we consider respect causality, i.e., an output can only depend on inputs that are given before that output. We formalize this by requiring that for every change in the output history, there is an earlier change in the input history.

Definition 6.1. For finite disjoint $\mathcal{I}, \mathcal{O} \subseteq \Lambda$, a *causal $(\mathcal{I}, \mathcal{O})$ -system over \mathcal{X}* is a function $\mathbf{s}: \mathcal{X}^{\mathcal{I}} \rightarrow \mathcal{X}^{\mathcal{O}}$ such that

$$\forall \mathbf{X}, \mathbf{X}' \in \mathcal{X}^{\mathcal{I}} \forall o \in \mathcal{O} \forall y \in (\mathbf{s}(\mathbf{X})(o) \Delta \mathbf{s}(\mathbf{X}')(o)) \exists i \in \mathcal{I} \exists x \in (\mathbf{X}(i) \Delta \mathbf{X}'(i)) \quad x \prec y.$$

We denote the set of causal $(\mathcal{I}, \mathcal{O})$ -systems over \mathcal{X} by $\mathfrak{C}_{\mathcal{I}, \mathcal{O}}$.

As we show in [Appendix A](#), our definition corresponds to a generalization of strict causality as considered by Matsikoudis and Lee [\[ML15\]](#) to systems with several interfaces.

We define the relation \sqsubseteq on \mathcal{X} such that $X_1 \sqsubseteq X_2$ if X_1 is an *initial segment* of X_2 , i.e., X_2 contains all elements from X_1 and all additional elements are greater than all elements in X_1 :

$$X_1 \sqsubseteq X_2 \quad :\iff \quad X_1 \subseteq X_2 \wedge \forall x_1 \in X_1 \forall x_2 \in X_2 \setminus X_1 \quad (x_1 \prec x_2).$$

It is easy to verify that $(\mathcal{X}, \sqsubseteq)$ is a poset.

Examples and comparison to Kahn networks. In this paragraph, we consider the intuitive special case $P = \mathcal{V} \times \mathbb{R}$ with the partial order as defined in (2). In contrast to Kahn networks, causal systems for this choice of P directly allow modeling systems that produce outputs depending on the precise time of the inputs. As we will see below, there are also more subtle differences.

If we ignore the concrete times and only keep the order of the elements in an input or output $X \in \mathcal{X}$, we obtain a sequence of values in \mathcal{V} , i.e., an element of the domain of Kahn networks. Moreover, the relation \sqsubseteq on \mathcal{X} then corresponds to the relation \sqsubseteq on sequences we defined for

Kahn networks. However, causality does not imply that the function is monotone with respect to \sqsubseteq . To see this, consider the function $\mathbf{s}: \mathcal{X}^{\{i\}} \rightarrow \mathcal{X}^{\{o\}}$ with

$$\mathbf{s}(\mathbf{X})(o) = \begin{cases} \{(1, 1)\}, & (1, 0) \in \mathbf{X}(i), \\ \{(0, 1)\}, & (1, 0) \notin \mathbf{X}(i). \end{cases}$$

This function corresponds to a system with one input interface i and one output interface o that outputs at time 1 a bit indicating whether 1 has been input at time 0. This is a causal system with respect to [Definition 6.1](#) because outputs can only change at time 1, and if they do, the inputs must have changed at time 0, which is before time 1. On the other hand, \mathbf{s} is not monotone with respect to \sqsubseteq : We have $\emptyset \sqsubseteq \{(1, 0)\}$, but on input \emptyset , the system outputs $\{(0, 1)\}$, and on input $\{(1, 0)\}$, the system outputs $\{(1, 1)\}$, where $\{(0, 1)\} \not\sqsubseteq \{(1, 1)\}$.

Note that our definition of causality requires that for every change in the outputs, there is a change in the inputs *strictly* before that. Including systems that produce outputs without any delay makes it impossible to allow arbitrary connections: Consider a system that at one interface takes as input a bit b and outputs $1 - b$ at another interface *at the same time*, and outputs there the bit 0 at time 1 if no inputs are given at that time. If we now connect these two interfaces, one cannot consistently assign a value to that interface.

This restriction means in particular that the function $\mathcal{X}^{\{i\}} \rightarrow \mathcal{X}^{\{o_1, o_2\}}$ that maps the input to itself at both output interface is not a causal system, because changing an input at some point in time results in a change of the outputs at the same time. To model a causal system that corresponds to the Kahn network we gave as an example in [Section 5.4](#), which forwards all inputs to both output interfaces, we therefore need to add some delay to the inputs. For the sake of simplicity, we here add a fixed delay $\delta > 0$ to all inputs. We obtain the function $\mathbf{s}: \mathcal{X}^{\{i\}} \rightarrow \mathcal{X}^{\{o_1, o_2\}}$ with

$$\mathbf{s}(\mathbf{X})(o_1) = \mathbf{s}(\mathbf{X})(o_2) = \{(v, t + \delta) \mid (v, t) \in \mathbf{X}(i)\}.$$

One can easily verify that this function does satisfy [Definition 6.1](#). For connecting interfaces i and o_1 , \emptyset is the *unique* fixed point: It clearly is a fixed point, and there is no other fixed point since if there is some input, the output is delayed by δ and hence different from the input. Therefore, the resulting system produces no output, which is consistent with our analysis of the corresponding Kahn network. Note, however, that in contrast to the Kahn network, we here have a unique fixed point, so the output \emptyset is not only intuitive, but also formally the only option. We show in [Section 6.2](#) that causality always guarantees unique fixed points.

On the restriction to well-ordered subsets. One cannot in general define a consistent system algebra of causal systems for $\mathcal{X} = \mathcal{P}(P)$. To see this, let $P = \mathbb{Q}$ and let \preceq be the usual order on \mathbb{Q} . For this poset, an output $t \in \mathbb{Q}$ can be interpreted as an output from a unary set at time t . Consider a system \mathbf{s} with input interface i and output interfaces o and o' that for all $n \in \mathbb{N} \setminus \{0\}$ generates an output at time $1/n$ at both output interfaces if and only if it did not receive an input before time $1/n$. Formally, $\mathbf{s}(\{(i, X)\}) = \{(o, Y), (o', Y)\}$ with $Y = \{1/n \mid n \in \mathbb{N} \setminus \{0\} \wedge \forall t \in X (t \geq 1/n)\}$. This system clearly respects causality, but, e.g., for $X = \emptyset$, the output is not well-ordered because for each output, there is another output before. When we connect interfaces i and o , the resulting system does not have a well-defined behavior: If it outputs something at time $1/n$ at interface o' , it did not receive any input before, i.e., it did not output anything at interface o before. In particular, it did not output anything before time $1/2n$. But then, it would output something at time $1/2n$, a contradiction. Hence, the

system cannot output anything at all. But then, it would output something at time 1, also a contradiction.

This example shows that we at least have to exclude infinite descending chains, i.e., \mathcal{X} can only contain subsets of P on which the ordering is well-founded. This is implied by only considering well-ordered subsets as we do. One could generalize \mathcal{X} to include non-well-ordered sets that contain no infinite descending chains, as done by Matsikoudis and Lee [ML15]. A nonempty subset of such sets contains minimal elements, but not necessarily a unique minimum, allowing for incomparable or concurrent inputs and outputs. Since our systems, in contrast to the functions considered by Matsikoudis and Lee, can have multiple interfaces, we do not lose much generality by restricting ourselves to well-ordered sets: At different interfaces, we can still have inputs or outputs at incomparable or equal times. We also find it natural to have several interfaces for potentially simultaneous inputs or outputs because this makes the assumptions on the concurrency explicit. Moreover, well-ordered sets are easier to handle and allow more intuitive proofs.

6.2 Fixed Points of Causal Functions

To define interface connection for causal systems, we need to prove a result about fixed points. We show that causal functions, which correspond to a single-interface variant of Definition 6.1, have a unique fixed point. We also show how this fixed point can be obtained, i.e., we provide a constructive proof. A similar result was proven non-constructively by Naundorf [Nau00]; Matsikoudis and Lee have provided a constructive proof of a related result [ML15]. However, we find that our construction is simpler and better applicable in our setting.

The intuition is similar to that of other constructive fixed point theorems: Start with $X_0 = \emptyset$ and then apply the function f over and over again until a fixed point is reached. In contrast to monotone functions (see Theorem 2.13 and Theorem 2.17), simply setting $X_{\alpha+1} = f(X_\alpha)$ does not work because an input can prevent future outputs; consider for example a function f that on input \emptyset outputs $\{x_0, x_1\}$ with $x_0 \prec x_1$, and on input $\{x_0\}$ outputs $\{x_0\}$. This does not contradict causality but the fixed point $\{x_0\}$ is not reached if we set $X_{\alpha+1} = f(X_\alpha)$ and $f(\{x_0, x_1\}) \neq \{x_0\}$. We therefore only add the least new element in $f(X_\alpha)$ to X_α to obtain $X_{\alpha+1}$. This means for the example above that $X_1 = X_0 \cup \{\min(f(X_0) \setminus X_0)\} = \{x_0\}$. For nonzero limit ordinals α , we set $X_\alpha = \bigcup_{\beta < \alpha} X_\beta$.

Theorem 6.2. *Let \mathcal{X} be as above and $f: \mathcal{X} \rightarrow \mathcal{X}$ such that*

$$\forall X, X' \in \mathcal{X} \forall y \in (f(X) \triangle f(X')) \exists x \in (X \triangle X') \quad x \prec y. \quad (3)$$

Then, f has a unique fixed point. This fixed point equals $X_{\hat{\alpha}}$ for some ordinal $\hat{\alpha}$, where

$$\begin{aligned} X_0 &= \emptyset, \\ \text{for any ordinal } \alpha, \quad X_{\alpha+1} &= \begin{cases} X_\alpha, & f(X_\alpha) \setminus X_\alpha = \emptyset, \\ X_\alpha \cup \{\min(f(X_\alpha) \setminus X_\alpha)\}, & \text{otherwise,} \end{cases} \\ \text{and for nonzero limit ordinals } \alpha, \quad X_\alpha &= \bigcup_{\beta < \alpha} X_\beta. \end{aligned}$$

Proof. Note that $X_{\alpha+1}$ is well-defined if X_α is well-ordered, which is implied by the following claim.

Claim 1. For all ordinals α , X_α is well-ordered, we have $X_\beta \sqsubseteq X_\alpha$ for all $\beta \leq \alpha$, and

$$\forall x_1 \in X_\alpha \forall x_2 \in f(X_\alpha) \setminus X_\alpha (x_1 \prec x_2).$$

Proof of claim. The proof is by transfinite induction over α .

(i) For $\alpha = 0$, there is nothing to show since $X_\alpha = \emptyset$.

(ii) Assume the claim holds for some ordinal α . If $X_{\alpha+1} = X_\alpha$, the claim also holds for $\alpha + 1$. Otherwise, $X_{\alpha+1} = X_\alpha \cup \{\min(f(X_\alpha) \setminus X_\alpha)\}$. To see that $X_{\alpha+1}$ is well-ordered, let $S \subseteq X_{\alpha+1}$, $S \neq \emptyset$. If $S \cap X_\alpha = \emptyset$, S contains only one element, which is then the least element. Otherwise, the least element of S is $\min(S \cap X_\alpha)$ because $X_{\alpha+1} \setminus X_\alpha$ only contains the element $\min(f(X_\alpha) \setminus X_\alpha)$, and since the claim holds for α , this element is greater than all elements in X_α .

Since the claim holds for α , we have $\forall x_1 \in X_\alpha \forall x_2 \in f(X_\alpha) \setminus X_\alpha (x_1 \prec x_2)$. This implies $X_\alpha \sqsubseteq X_{\alpha+1}$. Since $X_\beta \sqsubseteq X_\alpha$ for all $\beta \leq \alpha$, we also have $X_\beta \sqsubseteq X_{\alpha+1}$ for all $\beta \leq \alpha + 1$.

Let $x_1 \in X_{\alpha+1}$ and $x_2 \in f(X_{\alpha+1}) \setminus X_{\alpha+1}$. We then have $x_1 \preceq \min(f(X_\alpha) \setminus X_\alpha)$. If $x_2 \in f(X_\alpha)$, then $x_2 \in (f(X_\alpha) \setminus X_\alpha) \setminus \{\min(f(X_\alpha) \setminus X_\alpha)\}$; hence, $x_1 \prec x_2$. Otherwise, $x_2 \in f(X_{\alpha+1}) \setminus f(X_\alpha)$. By (3), we then have $\min(f(X_\alpha) \setminus X_\alpha) \prec x_2$, and since $x_1 \preceq \min(f(X_\alpha) \setminus X_\alpha)$, we also have $x_1 \prec x_2$. Thus, the claim holds for $\alpha + 1$.

(iii) Now let α be a nonzero limit ordinal and assume the claim holds for all $\beta < \alpha$. To show that X_α is well-ordered, let $S \subseteq X_\alpha$, $S \neq \emptyset$. Let β be the least ordinal such that $S \cap X_\beta \neq \emptyset$. By definition of X_α , we have $\beta < \alpha$ and therefore, $s := \min(S \cap X_\beta)$ exists. To see that s is the least element of S , let $x \in S \setminus X_\beta$. Then, $x \in X_\gamma$ for some γ with $\beta < \gamma < \alpha$. This implies $X_\beta \sqsubseteq X_\gamma$, and therefore $s \prec x$. Hence, X_α is well-ordered.

To show that $X_\beta \sqsubseteq X_\alpha$ for all $\beta \leq \alpha$, note that $X_\beta \subseteq X_\alpha$ and let $x_1 \in X_\beta$, $x_2 \in X_\alpha \setminus X_\beta$. Then, there exists some ordinal γ such that $x_2 \in X_\gamma \setminus X_\beta$ and $\beta < \gamma < \alpha$. Because we assume that the claim holds for all ordinals less than α , we obtain $X_\beta \sqsubseteq X_\gamma$. Hence, $x_1 \prec x_2$ and thus $X_\beta \sqsubseteq X_\alpha$.

Let $x_1 \in X_\alpha$ and $x_2 \in f(X_\alpha) \setminus X_\alpha$. By definition of X_α , there exists some $\beta < \alpha$ such that $x_1 \in X_\beta$. As shown above, we have $X_\beta \sqsubseteq X_\alpha$. Therefore, $x_2 \in f(X_\alpha) \setminus X_\beta$. If $x_2 \in f(X_\beta)$, we have $x_1 \prec x_2$ since $x_2 \in f(X_\beta) \setminus X_\beta$ and the claim holds for β . Otherwise, we have $x_2 \in f(X_\alpha) \setminus f(X_\beta)$. By (3), this implies $\min(X_\alpha \setminus X_\beta) \prec x_2$. Because $x_1 \in X_\beta \sqsubseteq X_\alpha$, we conclude $x_1 \prec \min(X_\alpha \setminus X_\beta) \prec x_2$. Altogether, the claim holds for X_α .

By [Theorem 2.8](#), the claim holds for all ordinals α . ◇

We now have by [Claim 1](#) and [Lemma 2.9](#) that there exists an ordinal $\hat{\alpha}$ such that $X_{\hat{\alpha}} = X_{\hat{\alpha}+1}$. Therefore, $f(X_{\hat{\alpha}}) \setminus X_{\hat{\alpha}} = \emptyset$. To conclude that $X_{\hat{\alpha}}$ is a fixed point of f , we show that $X_{\hat{\alpha}} \subseteq f(X_{\hat{\alpha}})$, and hence $f(X_{\hat{\alpha}}) = X_{\hat{\alpha}}$.

Claim 2. For all ordinals α , $X_\alpha \subseteq f(X_\alpha)$.

Proof of claim. We prove the claim by transfinite induction over α .

(i) For $\alpha = 0$, the claim trivially holds.

- (ii) Assume the claim holds for some ordinal α and let $x \in X_{\alpha+1}$. Then, $x \in f(X_\alpha)$ because either $x \in X_\alpha \subseteq f(X_\alpha)$ or $x = \min(f(X_\alpha) \setminus X_\alpha) \in f(X_\alpha)$. Assume toward a contradiction that $x \notin f(X_{\alpha+1})$. We then have $x \in f(X_\alpha) \setminus f(X_{\alpha+1})$. By (3) and the definition of $X_{\alpha+1}$, this implies $\min(f(X_\alpha) \setminus X_\alpha) \prec x$. Since $x \in X_{\alpha+1}$, we either have $x = \min(f(X_\alpha) \setminus X_\alpha)$ or $x \in X_\alpha$ and therefore $x \prec \min(f(X_\alpha) \setminus X_\alpha)$ by Claim 1. In both cases, we obtain $x \preceq \min(f(X_\alpha) \setminus X_\alpha) \prec x$, a contradiction. Hence, $x \in f(X_{\alpha+1})$.
- (iii) Let α be a nonzero limit ordinal and assume $X_\beta \subseteq f(X_\beta)$ for all $\beta < \alpha$. Let $x \in X_\alpha$. Then, $x \in X_\beta \subseteq f(X_\beta)$ for some $\beta < \alpha$. By Claim 1, we have $X_\beta \sqsubseteq X_\alpha$. Assume toward a contradiction that $x \notin f(X_\alpha)$. Then, (3) implies $\min(X_\alpha \setminus X_\beta) \prec x$. However, since $x \in X_\beta \sqsubseteq X_\alpha$, we also have $x \prec \min(X_\alpha \setminus X_\beta)$, a contradiction. Thus, $X_\alpha \subseteq f(X_\alpha)$. \diamond

To prove uniqueness, let Y_1 and Y_2 be fixed points of f . If $Y_1 \neq Y_2$, we can assume without loss of generality that $Y_1 \setminus Y_2 \neq \emptyset$ and

$$\forall y \in Y_2 \setminus Y_1 \ (y \not\prec \min(Y_1 \setminus Y_2)) \quad (4)$$

(otherwise, swap the roles of Y_1 and Y_2). Since $\min(Y_1 \setminus Y_2) \in Y_1 \setminus Y_2 = f(Y_1) \setminus f(Y_2)$, we have by (3) that there exists some $y \in Y_1 \triangle Y_2$ such that $y \prec \min(Y_1 \setminus Y_2)$. Then, $y \in Y_1 \setminus Y_2$ implies $y \prec y$, and $y \in Y_2 \setminus Y_1$ contradicts (4). Therefore, we must have $Y_1 = Y_2$. This concludes the proof. \square

While for some functions, $\hat{\alpha}$ will be a small finite number, using ordinals beyond ω cannot be avoided in general. To illustrate this, and to show that the iteration can still be carried out in this case, consider the following example. Again let $P = \mathbb{Q}$ and let \preceq be the usual order on \mathbb{Q} . Let f be the function that on the empty input outputs \mathbb{N} , and for each input $x \in \mathbb{Q}$, additionally outputs $(x + \lfloor x + 1 \rfloor)/2 \succ x$. That is

$$f(X) = \mathbb{N} \cup \{(x + \lfloor x + 1 \rfloor)/2 \mid x \in X\}.$$

We then have

$$X_0 = \emptyset, X_1 = \{\min f(\emptyset)\} = \{0\}, X_2 = \{0, \frac{1}{2}\}, X_3 = \{0, \frac{1}{2}, \frac{3}{4}\}, \dots, X_n = \{0, \frac{1}{2}, \dots, \frac{2^{n-1}-1}{2^{n-1}}\}, \dots$$

Taking the first limit, we obtain $X_\omega = \bigcup_{\beta < \omega} X_\beta = \{\frac{2^n-1}{2^n} \mid n \in \mathbb{N}\}$. We then continue as before:

$$X_{\omega+1} = X_\omega \cup \{1\}, X_{\omega+2} = X_\omega \cup \{1, 1 + \frac{1}{2}\}, \dots, X_{\omega+n} = X_\omega \cup \{1, 1 + \frac{1}{2}, \dots, 1 + \frac{2^{n-1}-1}{2^{n-1}}\}, \dots$$

For the next limit we have $X_{\omega+\omega} = X_{\omega \cdot 2} = \{m + \frac{2^n-1}{2^n} \mid m \in \{0, 1\}, n \in \mathbb{N}\}$. Continuing this process, we arrive at

$$X_{\omega \cdot \omega} = \{m + \frac{2^n-1}{2^n} \mid m, n \in \mathbb{N}\}.$$

It is easy to see that $X_{\omega \cdot \omega}$ is a fixed point of f and therefore $\hat{\alpha} = \omega \cdot \omega$.

If we again interpret an output $t \in \mathbb{Q}$ as an output from a unary set at time t , the system corresponding to the function f after connecting interfaces generates infinitely many outputs in a finite time. This is often called *Zeno behavior*, named after Zeno's paradoxes. Such behavior can be avoided by requiring *delta causality* instead of just causality, which places a lower bound δ on the reaction time of a system [LS98; ML15]. While this seems to be a reasonable restriction in practice, it makes the definitions more cumbersome. Also, the system described above does have a well-defined behavior, and the fixed point is what one expects. Having less restrictions on the type of system can also improve mathematical convenience when modeling systems.

6.3 Defining the System Algebra

We can apply [Theorem 6.2](#) to show that causal systems have a unique fixed point for arbitrary interface connections.

Corollary 6.3. *Let $\mathcal{I}, \mathcal{O} \subseteq \Lambda$ be finite disjoint sets, let $\mathbf{s} \in \mathfrak{C}_{\mathcal{I}, \mathcal{O}}$, and let $i \in \mathcal{I}$, $o \in \mathcal{O}$. Then, for all $\mathbf{X} \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$, there exists a unique $X_i \in \mathcal{X}$ such that $\mathbf{s}(\mathbf{X} \cup \{(i, X_i)\})(o) = X_i$.*

Proof. For $\mathbf{X} \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$, let $f: \mathcal{X} \rightarrow \mathcal{X}, X \mapsto \mathbf{s}(\mathbf{X} \cup \{(i, X)\})(o)$ and note that the fixed points of f precisely correspond to the values $X_i \in \mathcal{X}$ such that $\mathbf{s}(\mathbf{X} \cup \{(i, X_i)\})(o) = X_i$. It is therefore sufficient to show that f has a unique fixed point. To verify that f satisfies the condition of [Theorem 6.2](#), let $X, X' \in \mathcal{X}$ and let $y \in (f(X) \Delta f(X'))$. Then, $y \in (\mathbf{s}(\mathbf{X} \cup \{(i, X)\})(o) \Delta \mathbf{s}(\mathbf{X} \cup \{(i, X')\})(o))$. We therefore have by [Definition 6.1](#) that there exists $x \in X \Delta X'$ such that $x \prec y$. Hence, [Theorem 6.2](#) implies that f has a unique fixed point. \square

Definition 6.4. We define $\mathfrak{F}(\mathfrak{C}, \Gamma, \phi)$, the Λ -system algebra of causal systems over \mathcal{X} , as follows: For $\mathbf{s} \in \mathfrak{C}_{\mathcal{I}, \mathcal{O}}$, $\Gamma(\mathbf{s}) := \{\{i, o\} \mid i \in \mathcal{I}, o \in \mathcal{O}\}$ and $\phi_{i,o}^{\mathbf{s}}(\mathbf{X})$ is the unique element of $\Phi_{i,o,\mathbf{X}}^{\mathbf{s}}$ for all $i \in \mathcal{I}$, $o \in \mathcal{O}$, and $\mathbf{X} \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$.

To show that causal systems form a functional system algebra, we have to prove that connecting interfaces of a causal system again yields a causal system. To this end, we first show that changing an input \mathbf{X} can only change the fixed point after that change in \mathbf{X} .

Lemma 6.5. *Let $\mathcal{I}, \mathcal{O} \subseteq \Lambda$ be finite disjoint sets, let $\mathbf{s} \in \mathfrak{C}_{\mathcal{I}, \mathcal{O}}$, and let $i \in \mathcal{I}$, $o \in \mathcal{O}$. Further let $\mathbf{X}, \mathbf{X}' \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$. We then have*

$$\forall y \in \phi_{i,o}^{\mathbf{s}}(\mathbf{X}) \Delta \phi_{i,o}^{\mathbf{s}}(\mathbf{X}') \exists i' \in \mathcal{I} \setminus \{i\} \exists x \in \mathbf{X}(i') \Delta \mathbf{X}'(i') (x \prec y).$$

Proof. Let

$$E := \{y \in \phi_{i,o}^{\mathbf{s}}(\mathbf{X}) \Delta \phi_{i,o}^{\mathbf{s}}(\mathbf{X}') \mid \forall i' \in \mathcal{I} \setminus \{i\} \forall x \in \mathbf{X}(i') \Delta \mathbf{X}'(i') (x \not\prec y)\}$$

and assume toward a contradiction that $E \neq \emptyset$. Then, there exists $y_0 \in E$ such that $\forall y \in E (y \not\prec y_0)$ because $\phi_{i,o}^{\mathbf{s}}(\mathbf{X})$ and $\phi_{i,o}^{\mathbf{s}}(\mathbf{X}')$ are well-ordered and $E \subseteq \phi_{i,o}^{\mathbf{s}}(\mathbf{X}) \cup \phi_{i,o}^{\mathbf{s}}(\mathbf{X}')$. Since $\phi_{i,o}^{\mathbf{s}}(\mathbf{X}) = \mathbf{s}(\mathbf{X} \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{X}))\})(o)$ and $\phi_{i,o}^{\mathbf{s}}(\mathbf{X}') = \mathbf{s}(\mathbf{X}' \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{X}'))\})(o)$, [Definition 6.1](#) implies that there exists $i' \in \mathcal{I}$ and $x \in (\mathbf{X} \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{X}))\})(i') \Delta (\mathbf{X}' \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{X}'))\})(i')$ such that $x \prec y_0$. Because $y_0 \in E$, we have $i' = i$, i.e., $x \in \phi_{i,o}^{\mathbf{s}}(\mathbf{X}) \Delta \phi_{i,o}^{\mathbf{s}}(\mathbf{X}')$. We further have $x \notin E$ since $x \prec y_0$ and $\forall y \in E (y \not\prec y_0)$. Thus, there exist $i'' \in \mathcal{I} \setminus \{i\}$ and $x' \in \mathbf{X}(i'') \Delta \mathbf{X}'(i'')$ such that $x' \prec x$. This implies $x' \prec y_0$, contradicting $y_0 \in E$. Hence, we have $E = \emptyset$. \square

Theorem 6.6. $(\mathfrak{C}, \lambda, \parallel, \Gamma, \gamma) = \mathfrak{F}(\mathfrak{C}, \Gamma, \phi)$ is a composition-order invariant functional Λ -system algebra over \mathcal{X} .

Proof. Note that \mathfrak{C} is closed under parallel composition since the causality condition in [Definition 6.1](#) is placed on each output interface separately. To show that \mathfrak{C} is also closed under connecting interfaces, let $\mathcal{I}, \mathcal{O} \subseteq \Lambda$ be finite disjoint sets, let $\mathbf{s} \in \mathfrak{C}_{\mathcal{I}, \mathcal{O}}$, $i \in \mathcal{I}$, and let $o \in \mathcal{O}$. Further let $\mathbf{X}, \mathbf{X}' \in \mathcal{X}^{\mathcal{I} \setminus \{i\}}$, $o' \in \mathcal{O} \setminus \{o\}$, and $y \in (\gamma_{\{i,o\}}(\mathbf{s})(\mathbf{X})(o') \Delta \gamma_{\{i,o\}}(\mathbf{s})(\mathbf{X}')(o'))$. We have to show that $\exists i' \in \mathcal{I} \setminus \{i\} \exists x \in (\mathbf{X}(i') \Delta \mathbf{X}'(i')) x \prec y$. By the definition of $\gamma_{\{i,o\}}(\mathbf{s})$ and [Definition 6.1](#), we have

$$\exists i' \in \mathcal{I} \exists x \in ((\mathbf{X} \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{X}))\})(i') \Delta (\mathbf{X}' \cup \{(i, \phi_{i,o}^{\mathbf{s}}(\mathbf{X}'))\})(i')) x \prec y.$$

If $i' \neq i$, we are done. Otherwise, let $x \in \phi_{i,o}^s(\mathbf{X}) \triangle \phi_{i,o}^s(\mathbf{X}')$ such that $x \prec y$. Lemma 6.5 then implies that $\exists i'' \in \mathcal{I} \setminus \{i\} \exists x' \in (\mathbf{X}(i'') \triangle \mathbf{X}'(i'')) x' \prec x \prec y$. Hence, we have $\gamma_{\{i,o\}}(\mathbf{s}) \in \mathfrak{C}$. This shows that $\mathfrak{F}(\mathfrak{C}, \Gamma, \phi)$ is a functional system algebra. Using Corollary 6.3 and Theorem 4.6, we conclude that it is composition-order invariant. \square

7 Conclusion and Future Work

We have introduced the concept of a *system algebra*, which captures the composition of systems as algebraic operations, and identified *composition-order invariance* as an important property of a system algebra. We have then introduced *functional system algebras*, which consist of systems that correspond to functions mapping inputs to outputs. As instantiations of functional system algebras, we have considered monotone and continuous systems, where Kahn networks are an important special case of them. We have shown their composition-order invariance, providing further insights into these well-studied systems. We have finally introduced the system algebra of causal systems, which allows to model systems that depend on time.

Future work includes using the system algebra of causal systems to analyze protocols that deal with time such as clock synchronization protocols. We are further developing a theory of probabilistic system algebras that can be used to describe and study cryptographic protocols.

Acknowledgments. Ueli Maurer was supported by the Swiss National Science Foundation (SNF), project No. 200020-132794. Christopher Portmann and Renato Renner are supported by the European Commission FP7 Project RAQUEL (grant No. 323970), US Air Force Office of Scientific Research (AFOSR) via grant FA9550-16-1-0245, the Swiss National Science Foundation (via the National Centre of Competence in Research ‘Quantum Science and Technology’), and the European Research Council – ERC (grant No. 258932). Björn Tackmann was supported by the Swiss National Science Foundation (SNF) via Fellowship No. P2EZP2_155566 and in part by the NSF grants CNS-1228890 and CNS-1116800.

A Comparison with Strict Causality by Matsikoudis and Lee

We show below that the definition of strict causality by Matsikoudis and Lee [ML15] and our definition of causality are equivalent up to syntactical differences for functions that are compatible with both formalisms. Beforehand, we recall the relevant definitions from [ML15]. The domains of the functions considered there consist of so-called *signals*. A signal is a partial function $\sigma: \tilde{\mathcal{T}} \rightarrow \mathcal{V}$, for some poset $(\tilde{\mathcal{T}}, \preceq)$ and some set \mathcal{V} [ML15, Definition 2.2]. We denote the set of all signals by Σ . Strict causality is then defined as follows.

Definition A.1. Let $F: \Sigma \rightarrow \Sigma$ be a partial function. Then, F is *strictly causal* if there is a partial function $f: \Sigma \times \tilde{\mathcal{T}} \rightarrow \Sigma$ such that for all σ in the domain of F and for all $\tau \in \tilde{\mathcal{T}}$,

$$F(\sigma)(\tau) = f(\sigma|_{\{\tau' \in \tilde{\mathcal{T}} \mid \tau' \prec \tau\}}, \tau),$$

where equality here means that either both terms are undefined, or they are both defined and equal.

To be compatible with these definitions, we only consider the special case of our system algebra where $\mathcal{X} = \{X \subseteq P \mid (X, \preceq) \text{ is well-ordered}\}$ for a poset (P, \preceq) with $P = \mathcal{V} \times \mathcal{T}$ for

some set \mathcal{V} and a poset (\mathcal{T}, \leq) , and $(v_1, t_1) \preceq (v_2, t_2)$ if and only if $t_1 < t_2$ or $(v_1, t_1) = (v_2, t_2)$. Let Λ be a set and consider the poset $(\tilde{\mathcal{T}}, \preceq)$, where $\tilde{\mathcal{T}} := \Lambda \times \mathcal{T}$ and $(i_1, t_1) \preceq (i_2, t_2)$ if and only if $t_1 < t_2$ or $(i_1, t_1) = (i_2, t_2)$. This means that, intuitively, the interface identifier $i \in \Lambda$ in our model becomes part of the “time” $\tau \in \tilde{\mathcal{T}}$ during the translation to the model of [ML15]. For $\mathcal{I} \subseteq \Lambda$, a tuple $\mathbf{X} \in \mathcal{X}^{\mathcal{I}}$ can then be viewed as a signal via the injection

$$\varphi_{\mathcal{I}}: \mathcal{X}^{\mathcal{I}} \rightarrow \Sigma, \mathbf{X} \mapsto \{((i, t), v) \in (\mathcal{I} \times \mathcal{T}) \times \mathcal{V} \mid i \in \mathcal{I} \wedge (v, t) \in \mathbf{X}(i)\}.$$

Note that $\varphi_{\mathcal{I}}(\mathbf{X})$ is indeed a partial function $\tilde{\mathcal{T}} \rightarrow \mathcal{V}$ because for each $(i, t) \in \mathcal{I} \times \mathcal{T}$, there is at most one $v \in \mathcal{V}$ such that $(v, t) \in \mathbf{X}(i)$ since $\mathbf{X}(i)$ is well-ordered.

The following lemma shows that the Definitions 6.1 and A.1 are essentially equivalent for the functions considered here.

Lemma A.2. *Let \mathcal{I} and \mathcal{O} be finite disjoint sets and let $\mathbf{s}: \mathcal{X}^{\mathcal{I}} \rightarrow \mathcal{X}^{\mathcal{O}}$. Then, \mathbf{s} is a causal $(\mathcal{I}, \mathcal{O})$ -system over \mathcal{X} in the sense of Definition 6.1 if and only if $\varphi_{\mathcal{O}} \circ \mathbf{s} \circ \varphi_{\mathcal{I}}^{-1}$ is strictly causal in the sense of Definition A.1.¹⁰*

Proof. First assume that \mathbf{s} is a causal $(\mathcal{I}, \mathcal{O})$ -system over \mathcal{X} and consider the partial function $f: \Sigma \times \mathcal{T} \rightarrow \Sigma$ defined by

$$f(\sigma, \tau) := \varphi_{\mathcal{O}} \circ \mathbf{s} \circ \varphi_{\mathcal{I}}^{-1}(\sigma)(\tau).$$

Now let $\sigma \in \Sigma$ and $\mathbf{X} \in \mathcal{X}^{\mathcal{I}}$ such that $\varphi_{\mathcal{I}}(\mathbf{X}) = \sigma$. Further let $\tau = (o, t) \in \mathcal{O} \times \mathcal{T}$. We then have by Definition 6.1 for $\mathbf{X}' := \varphi_{\mathcal{I}}^{-1}(\sigma|_{\{\tau' \in \tilde{\mathcal{T}} \mid \tau' \prec \tau\}})$,

$$\forall y \in (\mathbf{s}(\mathbf{X})(o) \Delta \mathbf{s}(\mathbf{X}')(o)) \exists i \in \mathcal{I} \exists x \in (\mathbf{X}(i) \Delta \mathbf{X}'(i)) \quad x \prec y.$$

By definition of \mathbf{X}' , there are no $i \in \mathcal{I}$ and $(v', t') \in (\mathbf{X}(i) \Delta \mathbf{X}'(i))$ with $t' < t$. This implies that there is no $v \in \mathcal{V}$ with $(v, t) \in (\mathbf{s}(\mathbf{X})(o) \Delta \mathbf{s}(\mathbf{X}')(o))$, and therefore

$$\begin{aligned} \varphi_{\mathcal{O}} \circ \mathbf{s} \circ \varphi_{\mathcal{I}}^{-1}(\sigma)(\tau) &= \varphi_{\mathcal{O}} \circ \mathbf{s}(\mathbf{X})(\tau) = \varphi_{\mathcal{O}} \circ \mathbf{s}(\mathbf{X}')(o, t) = \varphi_{\mathcal{O}} \circ \mathbf{s} \circ \varphi_{\mathcal{I}}^{-1}(\sigma|_{\{\tau' \in \tilde{\mathcal{T}} \mid \tau' \prec \tau\}})(\tau) \\ &= f(\sigma|_{\{\tau' \in \tilde{\mathcal{T}} \mid \tau' \prec \tau\}}, \tau). \end{aligned}$$

Hence, $\varphi_{\mathcal{O}} \circ \mathbf{s} \circ \varphi_{\mathcal{I}}^{-1}$ is strictly causal.

To prove the opposite direction, assume $\varphi_{\mathcal{O}} \circ \mathbf{s} \circ \varphi_{\mathcal{I}}^{-1}$ is strictly causal and let $f: \Sigma \times \mathcal{T} \rightarrow \Sigma$ be a partial function such that for all σ in the domain of $\varphi_{\mathcal{O}} \circ \mathbf{s} \circ \varphi_{\mathcal{I}}^{-1}$ and for all $\tau \in \tilde{\mathcal{T}}$, we have $\varphi_{\mathcal{O}} \circ \mathbf{s} \circ \varphi_{\mathcal{I}}^{-1}(\sigma)(\tau) = f(\sigma|_{\{\tau' \in \tilde{\mathcal{T}} \mid \tau' \prec \tau\}}, \tau)$. Now let $\mathbf{X}, \mathbf{X}' \in \mathcal{X}^{\mathcal{I}}$, $o \in \mathcal{O}$, and $(v, t) = y \in (\mathbf{s}(\mathbf{X})(o) \Delta \mathbf{s}(\mathbf{X}')(o))$. We then have $\varphi_{\mathcal{O}}(\mathbf{s}(\mathbf{X}))(o, t) \neq \varphi_{\mathcal{O}}(\mathbf{s}(\mathbf{X}'))(o, t)$ and thus for $\tau := (o, t)$,

$$f(\varphi_{\mathcal{I}}(\mathbf{X})|_{\{\tau' \in \tilde{\mathcal{T}} \mid \tau' \prec \tau\}}, \tau) = \varphi_{\mathcal{O}} \circ \mathbf{s}(\mathbf{X})(\tau) \neq \varphi_{\mathcal{O}} \circ \mathbf{s}(\mathbf{X}')(o, t) = f(\varphi_{\mathcal{I}}(\mathbf{X}')|_{\{\tau' \in \tilde{\mathcal{T}} \mid \tau' \prec \tau\}}, \tau).$$

Hence, $\varphi_{\mathcal{I}}(\mathbf{X})|_{\{\tau' \in \tilde{\mathcal{T}} \mid \tau' \prec \tau\}} \neq \varphi_{\mathcal{I}}(\mathbf{X}')|_{\{\tau' \in \tilde{\mathcal{T}} \mid \tau' \prec \tau\}}$. Therefore, there exist $i \in \mathcal{I}$ and $x \in (\mathbf{X}(i) \Delta \mathbf{X}'(i))$ with $x \prec y$. This shows that \mathbf{s} satisfies Definition 6.1 and concludes the proof. \square

Note that for all finite $\mathcal{I} \subseteq \Lambda$ and for all $\mathbf{X} \in \mathcal{X}^{\mathcal{I}}$, the relation \prec is well-founded on the domain of $\varphi_{\mathcal{I}}(\mathbf{X})$: If it contained an infinite descending chain, then there would also be an infinite descending chain in $\mathbf{X}(i)$ for some $i \in \mathcal{I}$ (since \mathcal{I} is finite), contradicting that $\mathbf{X}(i)$ is well-ordered. Thus, [ML15, Theorem 4.10] can be applied, which together with Lemma A.2 implies that for all causal $(\mathcal{I}, \mathcal{O})$ -systems \mathbf{s} over \mathcal{X} , the partial function $\varphi_{\mathcal{O}} \circ \mathbf{s} \circ \varphi_{\mathcal{I}}^{-1}$ is *strictly contracting*. Hence, all results in [ML15] about strictly contracting partial functions can be used.

¹⁰The domain of $\varphi_{\mathcal{O}} \circ \mathbf{s} \circ \varphi_{\mathcal{I}}^{-1}$ is the set of all $\sigma \in \Sigma$ such that there exists $\mathbf{X} \in \mathcal{X}^{\mathcal{I}}$ with $\varphi_{\mathcal{I}}(\mathbf{X}) = \sigma$.

References

- [AD94] R. Alur and D. Dill, “A theory of timed automata”, *Theoretical Computer Science*, vol. 126, no. 2, pp. 183–235, 1994. DOI: [10.1016/0304-3975\(94\)90010-8](https://doi.org/10.1016/0304-3975(94)90010-8).
- [BBR09] J. C. M. Baeten, T. Basten, and M. A. Reniers, *Process Algebra: Equational Theories of Communicating Processes*. Cambridge University Press, 2009, Cambridge Books Online. DOI: [10.1017/CB09781139195003](https://doi.org/10.1017/CB09781139195003).
- [Can01] R. Canetti, “Universally composable security: A new paradigm for cryptographic protocols”, in *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, Oct. 2001, pp. 136–145. DOI: [10.1109/SFCS.2001.959888](https://doi.org/10.1109/SFCS.2001.959888).
- [CLL+06] A. Cataldo, E. Lee, X. Liu, E. D. Matsikoudis, and H. Zheng, “A constructive fixed-point theorem and the feedback semantics of timed systems”, in *Discrete Event Systems, 2006 8th International Workshop on*, Jul. 2006, pp. 27–32. DOI: [10.1109/WODES.2006.1678403](https://doi.org/10.1109/WODES.2006.1678403).
- [DP02] B. A. Davey and H. A. Priestley, *Introduction to Lattices and Order*, Second. Cambridge University Press, 2002. DOI: [10.1017/CB09780511809088](https://doi.org/10.1017/CB09780511809088).
- [dAH01a] L. de Alfaro and T. A. Henzinger, “Interface automata”, in *Proceedings of the 8th European Software Engineering Conference Held Jointly with 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, ser. ESEC/FSE-9, Vienna, Austria: ACM, 2001, pp. 109–120. DOI: [10.1145/503209.503226](https://doi.org/10.1145/503209.503226).
- [dAH01b] —, “Interface theories for component-based design”, in *Embedded Software: First International Workshop, EMSOFT 2001 Tahoe City, CA, USA, October 8–10, 2001 Proceedings*, T. A. Henzinger and C. M. Kirsch, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 148–165. DOI: [10.1007/3-540-45449-7_11](https://doi.org/10.1007/3-540-45449-7_11).
- [Ési09] Z. Ésik, “Fixed point theory”, in *Handbook of Weighted Automata*, M. Droste, W. Kuich, and H. Vogler, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 29–65. DOI: [10.1007/978-3-642-01492-5_2](https://doi.org/10.1007/978-3-642-01492-5_2).
- [FLM86] M. J. Fischer, N. A. Lynch, and M. Merritt, “Easy impossibility proofs for distributed consensus problems”, *Distributed Computing*, vol. 1, no. 1, pp. 26–39, 1986. DOI: [10.1007/BF01843568](https://doi.org/10.1007/BF01843568).
- [Gol01] O. Goldreich, *Foundations of Cryptography: Basic Tools*. Cambridge University Press, 2001, vol. 1. DOI: [10.1017/CB09780511546891](https://doi.org/10.1017/CB09780511546891).
- [GD03] A. Granas and J. Dugundji, *Fixed Point Theory*. Springer New York, 2003. DOI: [10.1007/978-0-387-21593-8](https://doi.org/10.1007/978-0-387-21593-8).
- [Hal12] L. J. Halbeisen, *Combinatorial Set Theory: With a Gentle Introduction to Forcing*, ser. Springer Monographs in Mathematics. Springer London, 2012. DOI: [10.1007/978-1-4471-2173-2](https://doi.org/10.1007/978-1-4471-2173-2).
- [Har13] L. Hardy, “On the theory of composition in physics”, in *Computation, Logic, Games, and Quantum Foundations. The Many Facets of Samson Abramsky: Essays Dedicated to Samson Abramsky on the Occasion of His 60th Birthday*, B. Coecke, L. Ong, and P. Panangaden, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 83–106. DOI: [10.1007/978-3-642-38164-5_7](https://doi.org/10.1007/978-3-642-38164-5_7).

- [Hoa78] C. A. R. Hoare, “Communicating sequential processes”, *Commun. ACM*, vol. 21, no. 8, pp. 666–677, Aug. 1978. DOI: [10.1145/359576.359585](https://doi.org/10.1145/359576.359585).
- [Jec03] T. Jech, *Set Theory*, Third Millennium, ser. Springer Monographs in Mathematics. Springer Berlin Heidelberg, 2003. DOI: [10.1007/3-540-44761-X](https://doi.org/10.1007/3-540-44761-X).
- [Joh87] P. T. Johnstone, *Notes on Logic and Set Theory*. Cambridge University Press, 1987. DOI: [10.1017/CB09781139172066](https://doi.org/10.1017/CB09781139172066).
- [Kah74] G. Kahn, “The semantics of a simple language for parallel programming”, in *Information Processing 74*, J. L. Rosenfeld, Ed., North-Holland, 1974, pp. 471–475.
- [KLSV10] D. Kaynar, N. Lynch, R. Segala, and F. Vaandrager, “The theory of timed I/O automata, second edition”, *Synthesis Lectures on Distributed Computing Theory*, vol. 1, no. 1, pp. 1–137, 2010. DOI: [10.2200/S00310ED1V01Y201011DCT005](https://doi.org/10.2200/S00310ED1V01Y201011DCT005).
- [LSP82] L. Lamport, R. Shostak, and M. Pease, “The byzantine generals problem”, *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, Jul. 1982. DOI: [10.1145/357172.357176](https://doi.org/10.1145/357172.357176).
- [LNS82] J.-L. Lassez, V. L. Nguyen, and E. A. Sonenberg, “Fixed point theorems and semantics: A folk tale”, *Information Processing Letters*, vol. 14, no. 3, pp. 112–116, May 1982. DOI: [10.1016/0020-0190\(82\)90065-5](https://doi.org/10.1016/0020-0190(82)90065-5).
- [LP95] E. A. Lee and T. M. Parks, “Dataflow process networks”, *Proceedings of the IEEE*, vol. 83, no. 5, pp. 773–801, May 1995. DOI: [10.1109/5.381846](https://doi.org/10.1109/5.381846).
- [LS98] E. A. Lee and A. Sangiovanni-Vincentelli, “A framework for comparing models of computation”, *IEEE Transactions on Computer-Aided Design or Integrated Circuits and Systems*, vol. 17, no. 12, pp. 1217–1229, Dec. 1998. DOI: [10.1109/43.736561](https://doi.org/10.1109/43.736561).
- [LT89] N. A. Lynch and M. R. Tuttle, “An introduction to input/output automata”, *CWI Quarterly*, vol. 2, no. 3, pp. 219–246, Sep. 1989.
- [Mar76] G. Markowsky, “Chain-complete posets and directed sets with applications”, *algebra universalis*, vol. 6, no. 1, pp. 53–68, Dec. 1976. DOI: [10.1007/BF02485815](https://doi.org/10.1007/BF02485815).
- [ML15] E. Matsikoudis and E. A. Lee, “The fixed-point theory of strictly causal functions”, *Theoretical Computer Science*, vol. 574, pp. 39–77, 2015. DOI: [10.1016/j.tcs.2015.01.036](https://doi.org/10.1016/j.tcs.2015.01.036).
- [Mau12] U. Maurer, “Constructive cryptography – a new paradigm for security definitions and proofs”, in *Theory of Security and Applications: Joint Workshop, TOSCA 2011, Saarbrücken, Germany, March 31 - April 1, 2011, Revised Selected Papers*, S. Mödersheim and C. Palamidessi, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 33–56. DOI: [10.1007/978-3-642-27375-9_3](https://doi.org/10.1007/978-3-642-27375-9_3).
- [MR11] U. Maurer and R. Renner, “Abstract cryptography”, in *The Second Symposium on Innovations in Computer Science, ICS 2011*, B. Chazelle, Ed., Tsinghua University Press, Jan. 2011, pp. 1–21.
- [MT13] D. Micciancio and S. Tessaro, “An equational approach to secure multi-party computation”, in *The Fourth Symposium on Innovations in Theoretical Computer Science, ITCS 2013*, R. Kleinberg, Ed., ACM Press, Jan. 2013, pp. 355–372. DOI: [10.1145/2422436.2422478](https://doi.org/10.1145/2422436.2422478).

- [Mil73] R. Milner, “Models for LCF”, Stanford Artificial Intelligence Laboratory, Tech. Rep. STAN-CS-73-332, Jan. 1973, pp. 1–10.
- [Mil79] ———, “Flowgraphs and flow algebras”, *J. ACM*, vol. 26, no. 4, pp. 794–818, Oct. 1979. DOI: [10.1145/322154.322167](https://doi.org/10.1145/322154.322167).
- [Mil80] ———, *A Calculus of Communicating Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1980. DOI: [10.1007/3-540-10235-3](https://doi.org/10.1007/3-540-10235-3).
- [Nau00] H. Naundorf, “Strictly causal functions have a unique fixed point”, *Theoretical Computer Science*, vol. 238, no. 1–2, pp. 483–488, 2000. DOI: [10.1016/S0304-3975\(99\)00165-6](https://doi.org/10.1016/S0304-3975(99)00165-6).
- [PSL80] M. Pease, R. Shostak, and L. Lamport, “Reaching agreement in the presence of faults”, *J. ACM*, vol. 27, no. 2, pp. 228–234, Apr. 1980. DOI: [10.1145/322186.322188](https://doi.org/10.1145/322186.322188).
- [PMM+17] C. Portmann, C. Matt, U. Maurer, R. Renner, and B. Tackmann, “Causal boxes: Quantum information-processing systems closed under composition”, *IEEE Transactions on Information Theory*, vol. 63, no. 5, pp. 3277–3305, May 2017. DOI: [10.1109/TIT.2017.2676805](https://doi.org/10.1109/TIT.2017.2676805).
- [Pra85] V. Pratt, “The pomset model of parallel processes: Unifying the temporal and the spatial”, in *Seminar on Concurrency*, S. D. Brookes, A. W. Roscoe, and G. Winskel, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 1985, pp. 180–196.
- [Pra86] ———, “Modeling concurrency with partial orders”, *International Journal of Parallel Programming*, vol. 15, no. 1, pp. 33–71, Feb. 1986. DOI: [10.1007/BF01379149](https://doi.org/10.1007/BF01379149).
- [Sco70] D. Scott, “Outline of a mathematical theory of computation”, in *4th Annual Princeton Conference on Information Sciences and Systems*, 1970, pp. 169–176.
- [SZT+04] T. Stefanov, C. Zissulescu, A. Turjan, B. Kienhuis, and E. Deprettere, “System design using Kahn process networks: The Compaan/Laura approach”, in *Design, Automation and Test in Europe Conference and Exhibition, 2004. Proceedings*, vol. 1, Feb. 2004, 340–345 Vol.1. DOI: [10.1109/DATE.2004.1268870](https://doi.org/10.1109/DATE.2004.1268870).
- [Tac14] B. Tackmann, “A theory of secure communication”, PhD thesis, ETH Zürich, 2014. DOI: [10.3929/ethz-a-010250961](https://doi.org/10.3929/ethz-a-010250961).
- [Tar55] A. Tarski, “A lattice-theoretical fixpoint theorem and its applications”, *Pacific Journal of Mathematics*, vol. 5, no. 2, pp. 285–309, 1955. DOI: [10.2140/pjm.1955.5.285](https://doi.org/10.2140/pjm.1955.5.285).