A New Linear Logic for Deadlock-Free Session-Typed Processes *

Ornela Dardha and Simon J. Gay

School of Computing Science, University of Glasgow, United Kingdom
{Ornela.Dardha,Simon.Gay}@glasgow.ac.uk

Abstract. The π-calculus, viewed as a core concurrent programming language, has been used as the target of much research on type systems for concurrency. In this paper we propose a new type system for deadlock-free session-typed π-calculus processes, by integrating two separate lines of work. The first is the propositions-as-types approach by Caires and Pfenning, which provides a linear logic foundation for session types and guarantees deadlock-freedom by forbidding cyclic process connections. The second is Kobayashi's approach in which types are annotated with priorities so that the type system can check whether or not processes contain genuine cyclic dependencies between communication operations. We combine these two techniques for the first time, and define a new and more expressive variant of classical linear logic with a proof assignment that gives a session type system with Kobayashi-style priorities. This can be seen in three ways: (i) as a new linear logic in which cyclic structures can be derived and a Cycle-elimination theorem generalises Cut-elimination; (ii) as a logically-based session type system, which is more expressive than Caires and Pfenning's; (iii) as a logical foundation for Kobayashi's system, bringing it into the sphere of the propositions-as-types paradigm.

1 Introduction

The Curry-Howard correspondence, or propositions-as-types paradigm, provides a canonical logical foundation for functional programming [42]. It identifies types with logical propositions, programs with proofs, and computation with proof normalisation. It was natural to ask for a similar account of concurrent programming, and this question was brought into focus by the discovery of linear logic [24] and Girard's explicit suggestion that it should have some connection with concurrent computation. Several attempts were made to relate π-calculus processes to the proof nets of classical linear logic [1,8], and to relate CCS-like processes to the $*$-autonomous categories that provide semantics for classical linear logic [2]. However, this work did not result in a convincing propositions-as-types framework for concurrency, and did not continue beyond the 1990s.

* Supported by the UK EPSRC grant EP/K034413/1, "From Data Types to Session Types: A Basis for Concurrency and Distribution (ABCD)", and by COST Action IC1201, "Behavioural Types for Reliable Large-Scale Software Systems (BETTY)".

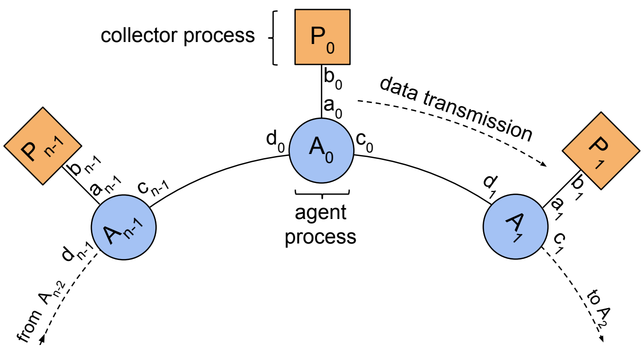

Fig. 1. Cyclic Scheduler

Meanwhile, Honda *et al.* [26,27,38] developed *session types* as a formalism for statically checking that messages have the correct types and sequence according to a communication protocol. Research on session types developed and matured over several years, eventually inspiring Caires and Pfenning [12] to discover a Curry-Howard correspondence between dual intuitionistic linear logic [7] and a form of π-calculus with session types [38]. Wadler [41] subsequently gave an alternative formulation based on classical linear logic, and related it to existing work on session types for functional languages [23]. The Caires-Pfenning approach has been widely accepted as a propositions-as-types theory of concurrent programming, as well as providing a logical foundation for session types.

Caires and Pfenning's type system guarantees deadlock-freedom by forbidding cyclic process structures. It provides a logical foundation for deadlock-free session processes, complementing previous approaches to deadlock-freedom in session type systems [9,15,21,22]. The logical approach to session types has been extended in many ways, including features such as dependent types [39], failures and non-determinism [11], sharing and races [6]. All this work relies on the acyclicity condition. However, rejecting cyclic process structures is unnecessarily strict: they are a necessary, but not sufficient, condition for the existence of deadlocked communication operations. As we will show in Ex. 1 (Fig. 1), there are deadlock-free processes that can naturally be implemented in a cyclic way, but are rejected by Caires and Pfenning's type system.

Our contribution is to define a new logic, *priority-based linear logic* (PLL), and formulate it as a type system for *priority-based* CP (PCP), which is a more expressive class of processes than Wadler's CP [41]. This is the first Curry-Howard correspondence that allows cyclic interconnected processes, while still ensuring deadlock-freedom. The key idea is that PLL includes conditions on inter-channel dependencies based on Kobayashi's type systems [29,30,32]. Our work can be viewed in three ways: (i) as a new linear logic in which cyclic proof structures can be derived; (ii) as an extension of Caires-Pfenning type systems so that they accept more processes, while maintaining the strong logical foundation; (iii) as a logical foundation for Kobayashi-style type systems.

An example of a deadlock-free cyclic process is Milner's well-known scheduler [35], described in the following Ex. 1.

Example 1 (Cyclic Scheduler, Fig. 1). A set of agents $A_0, ..., A_{n-1}$, for $n > 1$, is scheduled to perform a certain task in cyclic order, starting with agent A_0. For all $i \in \{1, ..., n-1\}$, agent A_i sends the result of computation to a collector process P_i, before transmitting further data to agent $A_{(i+1) \texttt{ mod } n}$. At the end of the round, A_0 sends the final result to P_0. Here we define a finite version of Milner's scheduler, which executes one round of communication.

$$\begin{array}{rcll} \mathit{Sched} & \triangleq & ...(\boldsymbol{\nu} a_i b_i)...(\boldsymbol{\nu} c_i d_{(i+1) \texttt{ mod } n})\big(A_0 \mid A_1 \mid ... \mid A_{n-1} \mid P_0 \mid P_1 \mid ... \mid P_{n-1}\big) & \\ A_0 & \triangleq & c_0[\mathbf{n}_0].d_0(x_0).a_0[\mathbf{m}_0].\mathsf{close}_0 & \\ A_i & \triangleq & d_i(x_i).a_i[\mathbf{m}_i].c_i[\mathbf{n}_i].\mathsf{close}_i & i \in \{1, ..., n-1\} \\ P_i & \triangleq & b_i(y_i).Q_i & i \in \{0, ..., n-1\} \end{array}$$

Prefix $c_0[\mathbf{n}_0]$ denotes an output on c_0, and $d_0(x_0)$ an input on d_0. For now, let $\mathbf{m}$ and $\mathbf{n}$ denote data. Process close_i closes the channels used by A_i: the details of this closure are irrelevant here (however, they are as in processes Q and R in Ex. 2). Process Q_i uses the message received from A_i, in internal computation. The construct $(\boldsymbol{\nu} ab)$ creates two channel endpoints a and b and binds them together. The system *Sched* is deadlock-free because $A_1, ..., A_{n-1}$ each wait for a message from the previous A_i before sending, and A_0 sends the initial message.

Sched is not typable in the original type systems by Caires-Pfenning and Wadler. To do that, it would be necessary to break A_0 into two parallel agents $A'_0 \triangleq c_0[\mathbf{n}_0].\mathsf{close}_{c_0}$ and $A''_0 \triangleq d_0(x_0).a_0[\mathbf{m}_0].\mathsf{close}_{d_0,a_0}$. This changes the design of the system, yielding a different one. Moreover, if the scheduler continues into a second round of communication, this redesign is not possible because of the potential dependency from the input on d_0 to the next output on c_0. However, *Sched* is typable in PCP; we will show the type assignment at the end of § 2.

There is a natural question at this point: *given that the cyclic scheduler is deadlock-free, is it possible to encode its semantics in* CP*, thus eliminating the need for* PCP? It is possible to define a centralised agent A that communicates with all the collectors P_i, resulting in a system that is semantically equivalent to our *Sched*. However, such an encoding has a global character, and changes the structure of the overall system from distributed to centralised. In programming terms, it corresponds to changing the software design, as we pointed out in Ex. 1, and ultimately the software architecture, which is not always desirable or even feasible. The aim of PCP is to generalise CP so that deadlock-free processes can be constructed with their natural structure. We would want any encoding of PCP into CP to be structure-preserving, which would mean translating the Cycle rule (given in Fig. 2) homomorphically; this is clearly impossible.

Contributions and Structure of the Paper In § 2 we define priority-based linear logic (PLL), which extends classical linear logic (CLL) with priorities attached to propositions. These priorities are based on Kobayashi's annotations for deadlock freedom [32]. By following the propositions-as-types paradigm, we define a term assignment for PLL proofs, resulting in priority-based classical processes (PCP), which extends Wadler's CP [41] with Mix and Cycle rules (Fig. 2). In §3 we define an operational semantics for PCP. In §4 we prove Cycle-elimination (Thm. 1) for PLL, analogous to the standard Cut-elimination theorem for CLL. Consequently, the results for PCP are subject reduction (Thm. 2), top-level deadlock-freedom (Thm. 3), and full deadlock-freedom for closed processes (Thm. 4). In § 5 we discuss related work and conclude the paper.

2 PCP: Classical Processes with Mix and Cycle

Priority-based CP (PCP) follows the style of Wadler's Classical Processes (CP) [41], with details inspired by Carbone *et al.* [14] and Caires and Pérez [11].

Types We start with types, which are based on CLL propositions. Let A, B range over types, given in Def. 1. Let $\mathsf{o}, \kappa \in \mathbb{N} \cup \{\omega\}$ range over *priorities*, which are used to annotate types. Let ω be a special element such that $\mathsf{o} < \omega$ for all $\mathsf{o} \in \mathbb{N}$. Often, we will omit ω. We will explain priorities later in this section.

Definition 1 (Types). *Types (A, B) are given by:*

$$A, B ::= \bot^{\mathsf{o}} \mid \mathbf{1}^{\mathsf{o}} \mid A \otimes^{\mathsf{o}} B \mid A \parr^{\mathsf{o}} B \mid \oplus^{\mathsf{o}} \{l_i : A_i\}_{i\in I} \mid \&^{\mathsf{o}}\{l_i : A_i\}_{i\in I} \mid ?^{\mathsf{o}} A \mid !^{\mathsf{o}} A$$

$\bot^{\mathsf{o}}$ and $\mathbf{1}^{\mathsf{o}}$ are associated with channel endpoints that are ready to be closed. $A \otimes^{\mathsf{o}} B$ (respectively, $A \parr^{\mathsf{o}} B$) is associated with a channel endpoint that first outputs (respectively, inputs) a channel of type A and then proceeds as B. $\oplus^{\mathsf{o}}\{l_i : A_i\}_{i\in I}$ is associated with a channel endpoint over which we can select a label from $\{l_i\}_{i\in I}$, and proceed as A_i. Dually, $\&^{\mathsf{o}}\{l_i : A_i\}_{i\in I}$ is associated with a channel endpoint that can offer a set of labelled types. $?^{\mathsf{o}} A$ types a collection of clients requesting A. Dually, $!^{\mathsf{o}} A$ types a server repeatedly accepting A.

Duality on types is total and is given in Def. 2. It preserves priorities of types.

Definition 2 (Duality). *The* duality *function* $(\cdot)^\perp$ *on types is given by:*

$$\begin{aligned} (A \parr^{\mathsf{o}} B)^\perp &= A^\perp \otimes^{\mathsf{o}} B^\perp & (\bot^{\mathsf{o}})^\perp &= \mathbf{1}^{\mathsf{o}} \\ (A \otimes^{\mathsf{o}} B)^\perp &= A^\perp \parr^{\mathsf{o}} B^\perp & (\mathbf{1}^{\mathsf{o}})^\perp &= \bot^{\mathsf{o}} \\ (\&^{\mathsf{o}}\{l_i : A_i\}_{i\in I})^\perp &= \oplus^{\mathsf{o}}\{l_i : A_i{}^\perp\}_{i\in I} & ?^{\mathsf{o}} A^\perp &= !^{\mathsf{o}} A^\perp \\ (\oplus^{\mathsf{o}}\{l_i : A_i\}_{i\in I})^\perp &= \&^{\mathsf{o}}\{l_i : A_i{}^\perp\}_{i\in I} & !^{\mathsf{o}} A^\perp &= ?^{\mathsf{o}} A^\perp \end{aligned}$$

Processes Let P, Q range over processes, given in Def. 3. Let x, y range over channel endpoints, and $\mathbf{m}, \mathbf{n}$ over channel endpoints of type either $\bot^{\mathsf{o}}$ or $\mathbf{1}^{\mathsf{o}}$.

Definition 3 (Processes). *Processes (P, Q) are given by:*

$$\begin{array}{llllll} P, Q ::= & x[y].P & \textit{(output)} & \mathbf{0} & \textit{(inaction)} \\ & x(y).P & \textit{(input)} & P \mid Q & \textit{(composition)} \\ & x \triangleleft l_j.P & \textit{(selection)} & (\boldsymbol{\nu} x^A y)P & \textit{(session restriction)} \\ & x \triangleright \{l_i : P_i\}_{i\in I} & \textit{(branching)} & x[\,].\mathbf{0} & \textit{(empty output)} \\ & x \to y^A & \textit{(forwarding)} & x().P & \textit{(empty input)} \end{array}$$

Process $x[y].P$ (respectively, $x(y).P$) outputs (respectively, inputs) y on channel endpoint x, and proceeds as P. Process $x \triangleleft l_j.P$ uses x to select l_j from a labelled choice process, typically being $x \triangleright \{l_i : P_i\}_{i\in I}$, and triggers P_j; labels indexed by the finite set I are pairwise distinct. Process $x \to y^A$ forwards communications from x to y, the latter having type A. Processes also include the inaction process $\mathbf{0}$, the parallel composition of P and Q, denoted $P \mid Q$, and the double restriction constructor $(\boldsymbol{\nu} x^A y)P$: the intention is that x and y denote dual session channel endpoints in P, and A is the type of x. Processes $x[\,].\mathbf{0}$ and $x().P$ are the empty output and empty input, respectively. They denote the closure of a session from the viewpoint of each of the two communicating participants.

Notions of bound/free names in processes are standard; we write $\mathtt{fn}(P)$ to denote the set of free names of P. Also, we write $P\{x/z\}$ to denote the (capture-avoiding) substitution of x for the free occurrences of z in P. Finally, we let $\tilde{x}$, which is different from x, denote a sequence $x_1, \ldots, x_n$ for $n > 0$.

Typing Rules Typing contexts, ranged over by Γ, Δ, Θ, are sets of typing assumptions $x : A$. We write Γ, Δ for union, requiring the contexts to be disjoint. A typing judgement $P \vdash \Gamma$ means "process P is well typed using context Γ".

Before presenting the typing rules, we need some auxiliary definitions. Our priorities are based on the annotations used by Kobayashi [32], but simplified to single priorities *à la* Padovani [37]. They obey the following laws:

(i) An action of priority o must be prefixed only by actions of priorities *strictly smaller than* o.
(ii) Communication requires *equal priorities* for the complementary actions.

Definition 4 (Priority). *The* priority *function* $\mathsf{pr}(\cdot)$ *on types is given by:*

$$\mathsf{pr}(A \mathbin{⅋}^{\mathsf{o}} B) = \mathsf{pr}(A \otimes^{\mathsf{o}} B) = \mathsf{o} \qquad \mathsf{pr}(\perp^{\mathsf{o}}) = \mathsf{pr}(\mathbf{1}^{\mathsf{o}}) = \mathsf{o}$$
$$\mathsf{pr}(\oplus^{\mathsf{o}}\{l_i : A_i\}_{i \in I}) = \mathsf{pr}(\&^{\mathsf{o}}\{l_i : A_i\}_{i \in I}) = \mathsf{o} \qquad \mathsf{pr}(\,?^{\mathsf{o}} A) = \mathsf{pr}(\,!^{\mathsf{o}} A) = \mathsf{o}$$

Definition 5 (Lift). *Let $t \in \mathbb{N}$. The* lift *operator* $\uparrow^t(\cdot)$ *on types is given by:*

$$\uparrow^t(A \mathbin{⅋}^{\mathsf{o}} B) = (\uparrow^t A) \mathbin{⅋}^{(\mathsf{o}+t)} (\uparrow^t B) \qquad \uparrow^t \perp^{\mathsf{o}} = \mathbf{1}^{(\mathsf{o}+t)}$$
$$\uparrow^t(A \otimes^{\mathsf{o}} B) = (\uparrow^t A) \otimes^{(\mathsf{o}+t)} (\uparrow^t B) \qquad \uparrow^t \mathbf{1}^{\mathsf{o}} = \perp^{(\mathsf{o}+t)}$$
$$\uparrow^t(\&^{\mathsf{o}}\{l_i : A_i\}_{i \in I}) = \&^{(\mathsf{o}+t)}\{l_i : \uparrow^t A_i\}_{i \in I} \qquad \uparrow^t(\,?^{\mathsf{o}} A) = \,?^{(\mathsf{o}+t)}(\uparrow^t A)$$
$$\uparrow^t(\oplus^{\mathsf{o}}\{l_i : A_i\}_{i \in I}) = \oplus^{(\mathsf{o}+t)}\{l_i : \uparrow^t A_i\}_{i \in I} \qquad \uparrow^t(\,!^{\mathsf{o}} A) = \,!^{(\mathsf{o}+t)}(\uparrow^t A)$$

We assume $\omega + t = \omega$ for all $t \in \mathbb{N}$.
The operator $\uparrow^t$ is extended component-wise to typing contexts: $\uparrow^t \Gamma$.

The typing rules are given in Fig. 2. Ax states that the forwarding process $x \to y^A$ is well typed if x and y have dual types, respectively $A^\perp$ and A. Mix types the parallel composition of two processes P and Q in the union of their disjoint typing contexts. Cycle is our key typing rule; it states that the restriction process is well typed, if the endpoints x and y have dual types, respectively A and $A^\perp$. By Def. 2, A and $A^\perp$ also have the same priorities, enforcing law (ii) above. In classical logic this rule would be unsound, but in PLL it allows deadlock-free cycles. Rule $\emptyset$ states that inaction is well typed in the empty context. Rules $\mathbf{1}$ and $\perp$ type channel closure actions from the viewpoint of each participant. Rule $⅋$ (respectively $\otimes$) types an input process $x(y).P$ (respectively, output process $x[y].P$), with y bound and x of type $A \mathbin{⅋}^{\mathsf{o}} B$ (respectively, $A \otimes^{\mathsf{o}} B$). The priority

$$\frac{}{x \to y^A \vdash x{:}A^\perp, y{:}A}\ \textsc{Ax} \qquad \frac{P \vdash \Gamma \quad Q \vdash \Delta}{P \mid Q \vdash \Gamma, \Delta}\ \textsc{Mix} \qquad \frac{P \vdash \Gamma, x{:}A, y{:}A^\perp}{(\boldsymbol{\nu} x^A y)P \vdash \Gamma}\ \textsc{Cycle}$$

$$\frac{}{\mathbf{0} \vdash \emptyset}\ \emptyset \qquad \frac{}{x[\,].\mathbf{0} \vdash x{:}\mathbf{1}^{\mathsf{o}}}\ \mathbf{1} \qquad \frac{P \vdash \Gamma \quad \mathsf{o} < \mathsf{pr}(\Gamma)}{x().P \vdash x{:}\perp^{\mathsf{o}}, \Gamma}\ \perp$$

$$\frac{P \vdash \Gamma, y{:}A, x{:}B \quad \mathsf{o} < \mathsf{pr}(\Gamma)}{x(y).P \vdash \Gamma, x{:}A \mathbin{⅋^{\mathsf{o}}} B}\ ⅋ \qquad \frac{P \vdash \Gamma, y{:}A, x{:}B \quad \mathsf{o} < \mathsf{pr}(\Gamma)}{x[y].P \vdash \Gamma, x{:}A \otimes^{\mathsf{o}} B}\ \otimes$$

$$\frac{\forall i \in I.(P_i \vdash \Gamma, x{:}A_i) \quad \mathsf{o} < \mathsf{pr}(\Gamma)}{x \triangleright \{l_i : P_i\}_{i \in I} \vdash \Gamma, x{:}\&^{\mathsf{o}}\{l_i : A_i\}_{i\in I}}\ \& \qquad \frac{P \vdash \Gamma, x{:}A_j \quad j \in I \quad \mathsf{o} < \mathsf{pr}(\Gamma)}{x \triangleleft l_j.P \vdash \Gamma, x{:}\oplus^{\mathsf{o}}\{l_i : A_i\}_{i\in I}}\ \oplus$$

$$\frac{P \vdash ?\Gamma, y{:}A \quad \mathsf{o} < \mathsf{pr}(?\Gamma)}{!x(y).P \vdash ?\Gamma, x{:}\,!^{\mathsf{o}}\,A}\ ! \qquad \frac{P \vdash \Gamma, y{:}A \quad \mathsf{o} < \mathsf{pr}(\Gamma)}{?x[y].P \vdash \Gamma, x{:}\,?^{\mathsf{o}}\,A}\ ?$$

$$\frac{P \vdash \Gamma}{P \vdash \Gamma, x{:}\,?^{\mathsf{o}}\,A}\ \mathrm{W} \qquad \frac{P \vdash \Gamma, y{:}\,?^{\kappa}\,A, z{:}\,?^{\kappa'}\,A \quad \mathsf{o} \leq \kappa \quad \mathsf{o} \leq \kappa' \quad \mathsf{o} < \mathsf{pr}(\Gamma)}{P\{x/y, x/z\} \vdash \Gamma, x{:}\,?^{\mathsf{o}}\,A}\ \mathrm{C}$$

Fig. 2. Typing rules for PCP.

o is strictly smaller than any priorities in the continuation process P, enforcing law (i) above. This is captured by $\mathsf{o} < \mathsf{pr}(\Gamma)$ in the premises of both rules, abbreviating "for all $z \in \mathtt{dom}(\Gamma), \mathsf{o} < \mathsf{pr}(\Gamma(z))$". Rules & and $\oplus$ type external and internal choice, respectively, and follow the previous two rules. Rule ! types a server and states that if P communicates along y following protocol A, then $!x(y).P$ communicates along x following protocol $!^{\mathsf{o}}\,A$. The three remaining rules type different numbers of clients. Rule ? is for a single client: if P communicates along y following A, then $?x[y].P$ communicates along x following $?^{\mathsf{o}}\,A$. Rule W is for no client: if P does not communicate along any channel following A, then it may be regarded as communicating along x following $?^{\mathsf{o}}\,A$, for some priority o. Rule C is for multiple clients: if P communicates along y following $?^{\kappa}\,A$, and z following protocol $?^{\kappa'}\,A$, then $P\{x/y, x/z\}$ communicates along a single channel x following $?^{\mathsf{o}}\,A$, where $\mathsf{o} \leqslant \kappa$ and $\mathsf{o} \leqslant \kappa'$. The last two conditions are necessary to deal with some cases in the proof of Cycle-elimination (Thm. 1).

Lifting preserves typability, by an easy induction on typing derivations.

Lemma 1. *If $P \vdash \Gamma$ then $P \vdash \uparrow^t \Gamma$.*

We will use this result in the form of an admissible rule: $\dfrac{P \vdash \Gamma}{P \vdash \uparrow^t \Gamma}\ \uparrow^t$

The Design of PCP We have included Mix and Cycle, which allow derivation of both the standard Cut and the Multicut by Abramsky *et al.* [2].

$$\left.\begin{array}{c}\dfrac{\dfrac{\vdash \Gamma, A_1, \ldots, A_n \quad \vdash \Delta, A_1^\perp, \ldots, A_n^\perp}{\vdash \Gamma, \Delta, A_1, \ldots, A_n, A_1^\perp, \ldots, A_n^\perp}\ \textsc{Mix}}{\vdash \Gamma, \Delta}\ \textsc{Cycle}^n\end{array}\right\}\textsc{Multicut}$$

Conversely, MIX is the nullary case of MULTICUT, and CYCLE can be derived from AX and MULTICUT:

$$\left.\dfrac{\vdash \Gamma, A, A^\perp \qquad \overline{\vdash A^\perp, A}\ \textsc{Ax}}{\vdash \Gamma}\ \textsc{Multicut}\qquad\right\}\textsc{Cycle}$$

Having included MIX, we choose CYCLE instead of MULTICUT, as CYCLE is more primitive.

In the presence of MIX and CYCLE, there is an isomorphism between $A \otimes B$ and $A \parr B$ in CLL. Both $A \otimes B \multimap A \parr B$ and $A \parr B \multimap A \otimes B$, are derivable, where $C \multimap D \ \triangleq\ C^\perp \parr D$ in CLL. Equivalently, both $(A^\perp \parr B^\perp) \parr (A \parr B)$ and $(A^\perp \otimes B^\perp) \parr (A \otimes B)$ are derivable. For simplicity, let $\mathsf{pr}(A) = \mathsf{pr}(B) = \omega$; by duality also $\mathsf{pr}(A^\perp) = \mathsf{pr}(B^\perp) = \omega$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\overline{\vdash A^\perp, A} \qquad \overline{\vdash B^\perp, B}}{\vdash A^\perp, B^\perp, A, B}\ \textsc{Mix} \qquad \mathsf{o}_1 < \omega}{\vdash A^\perp \parr^{\mathsf{o}_1} B^\perp, A, B}\ \parr \qquad \mathsf{o}_2 < \mathsf{o}_1}{\vdash A^\perp \parr^{\mathsf{o}_1} B^\perp, A \parr^{\mathsf{o}_2} B}\ \parr}{\vdash (A^\perp \parr^{\mathsf{o}_1} B^\perp) \parr^{\mathsf{o}} (A \parr^{\mathsf{o}_2} B)}\ \parr$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overline{\vdash A^\perp, A} \quad \overline{\vdash B^\perp, B}}{\vdash A^\perp, B^\perp, A, B}\ \textsc{Mix} \quad \mathsf{o}_1 < \omega}{\vdash A^\perp \otimes^{\mathsf{o}_1} B^\perp, A, B}\ \otimes \qquad \dfrac{\dfrac{\overline{\vdash A^\perp, A} \quad \overline{\vdash B^\perp, B}}{\vdash A^\perp, B^\perp, A, B}\ \textsc{Mix} \quad \mathsf{o}_2 < \omega}{\vdash A^\perp, B^\perp, A \otimes^{\mathsf{o}_2} B}\ \otimes}{\vdash A^\perp \otimes^{\mathsf{o}_1} B^\perp, A \otimes^{\mathsf{o}_2} B, A^\perp, A, B^\perp, B}\ \textsc{Mix}}{\vdash A^\perp \otimes^{\mathsf{o}_1} B^\perp, A \otimes^{\mathsf{o}_2} B}\ \textsc{Cycle}^2}{\vdash (A^\perp \otimes^{\mathsf{o}_1} B^\perp) \parr^{\mathsf{o}} (A \otimes^{\mathsf{o}_2} B)}\ \parr$$

The above derivations *without* priorities show the isomorphism between $A \otimes B$ and $A \parr B$ in CLL, which does not hold in our PLL, in particular as $\mathsf{o}_1 \neq \mathsf{o}_2$. The distinction between $\otimes$ and $\parr$, preserves the distinction between output and input in the term assignment. However, to simplify derivations, both typing rules (Fig. 2) have the same form. The usual tensor rule, where there are two separate derivations in the premise rather than just one, is derivable by using MIX.

Our type system performs priority-checking. Priorities can be inferred, as in Kobayashi's type system [32] and the tool TyPiCal [28]. We have opted for priority checking over priority inference, as the presentation is more elegant.

The following two examples illustrate the use of priorities. We first establish the structure of the typing derivation, then calculate the priorities. We conclude the section by showing the typing for the cyclic scheduler from § 1.

Example 2 (Cyclic process: deadlock-free). Consider the following process

$$P \ \triangleq\ (\boldsymbol{\nu} x_1 y_1)(\boldsymbol{\nu} x_2 y_2)\big[x_1(v).x_2(w).R \mid y_1[\mathbf{n}].y_2[\mathbf{n}'].Q\big]$$

where $R \triangleq x_1().v().x_2().w().\mathbf{0}$ and $Q \triangleq y_1[].\mathbf{0} \mid \mathbf{n}[].\mathbf{0} \mid y_2[].\mathbf{0} \mid \mathbf{n}'[].\mathbf{0}$. First, we show the typing derivation for the left-hand side of the parallel, $x_1(v).x_2(w).R$:

$$\dfrac{\dfrac{\dfrac{\overline{\mathbf{0} \vdash \emptyset \qquad \kappa_4 < \kappa_3 < \kappa_2 < \kappa_1}\ \emptyset}{R \vdash x_1 \colon \bot^{\kappa_4}\, v \colon \bot^{\kappa_3}, x_2 \colon \bot^{\kappa_2}, w \colon \bot^{\kappa_1} \qquad \mathsf{o}_1 < \kappa_4}\ \bot^4}{x_2(w).R \vdash x_1 \colon \bot^{\kappa_4}, v \colon \bot^{\kappa_3}, x_2 \colon \bot^{\kappa_1} \parr^{\mathsf{o}_1} \bot^{\kappa_2} \qquad \mathsf{o}_2 < \mathsf{o}_1}\ \parr}{x_1(v).x_2(w).R \vdash x_2 \colon \bot^{\kappa_1} \parr^{\mathsf{o}_1} \bot^{\kappa_2}, x_1 \colon \bot^{\kappa_3} \parr^{\mathsf{o}_2} \bot^{\kappa_4}}\ \parr \tag{1}$$

Now, the typing derivation for the right-hand side of the parallel, $y_1[\mathbf{n}].y_2[\mathbf{n}'].Q$, and recall that $\kappa_4 < \kappa_3 < \kappa_2 < \kappa_1$:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{y_1[\,].\mathbf{0} \vdash y_1 \colon \mathbf{1}^{\kappa_4}}\;\mathbf{1} \quad \dfrac{}{\mathbf{n}[\,].\mathbf{0} \vdash \mathbf{n} \colon \mathbf{1}^{\kappa_3}}\;\mathbf{1} \quad \dfrac{}{y_2[\,].\mathbf{0} \vdash y_1 \colon \mathbf{1}^{\kappa_2}}\;\mathbf{1} \quad \dfrac{}{\mathbf{n}'[\,].\mathbf{0} \vdash \mathbf{n}' \colon \mathbf{1}^{\kappa_1}}\;\mathbf{1}}{\overline{y_1[\,].\mathbf{0} \mid \mathbf{n}[\,].\mathbf{0} \mid y_2[\,].\mathbf{0} \mid \mathbf{n}'[\,].\mathbf{0} \vdash y_1 \colon \mathbf{1}^{\kappa_4}, \mathbf{n} \colon \mathbf{1}^{\kappa_3}, y_2 \colon \mathbf{1}^{\kappa_2}, \mathbf{n}' \colon \mathbf{1}^{\kappa_1}}}\;\textsc{Mix}^3 \qquad \mathsf{o}_3 < \kappa_4}{y_2[\mathbf{n}'].Q \vdash y_1 \colon \mathbf{1}^{\kappa_4}, \mathbf{n} \colon \mathbf{1}^{\kappa_3}, y_2 \colon \mathbf{1}^{\kappa_1} \otimes^{\mathsf{o}_3} \mathbf{1}^{\kappa_2}}\;\otimes \qquad \mathsf{o}_4 < \mathsf{o}_3}{y_1[\mathbf{n}].y_2[\mathbf{n}'].Q \vdash y_2 \colon \mathbf{1}^{\kappa_1} \otimes^{\mathsf{o}_3} \mathbf{1}^{\kappa_2}, y_1 \colon \mathbf{1}^{\kappa_3} \otimes^{\mathsf{o}_4} \mathbf{1}^{\kappa_4}}\;\otimes \tag{2}$$

Finally, the typing derivation for process P is as follows:

$$\dfrac{\dfrac{\dfrac{(1) \qquad\qquad (2)}{\begin{array}{l} x_1(v).x_2(w).R \mid y_1[\mathbf{n}].y_2[\mathbf{n}'].Q \vdash \\ \quad x_2 \colon \perp^{\kappa_1} \mathbin{⅋}^{\mathsf{o}_1} \perp^{\kappa_2}, x_1 \colon \perp^{\kappa_3} \mathbin{⅋}^{\mathsf{o}_2} \perp^{\kappa_4}, y_2 \colon \mathbf{1}^{\kappa_1} \otimes^{\mathsf{o}_3} \mathbf{1}^{\kappa_2}, y_1 \colon \mathbf{1}^{\kappa_3} \otimes^{\mathsf{o}_4} \mathbf{1}^{\kappa_4} \end{array}}\;\textsc{Mix} \qquad \mathsf{o}_1 = \mathsf{o}_3}{\begin{array}{l} (\boldsymbol{\nu} x_2 y_2)\big[x_1(v).x_2(w).R \mid y_1[\mathbf{n}].y_2[\mathbf{n}'].Q\big] \vdash \\ \quad x_1 \colon \perp^{\kappa_3} \mathbin{⅋}^{\mathsf{o}_2} \perp^{\kappa_4}, y_1 \colon \mathbf{1}^{\kappa_3} \otimes^{\mathsf{o}_4} \mathbf{1}^{\kappa_4} \end{array}}\;\textsc{Cycle} \qquad \mathsf{o}_2 = \mathsf{o}_4}{(\boldsymbol{\nu} x_1 y_1)(\boldsymbol{\nu} x_2 y_2)\big[x_1(v).x_2(w).R \mid y_1[\mathbf{n}].y_2[\mathbf{n}'].Q\big] \vdash \emptyset}\;\textsc{Cycle}$$

The system of equations

$$\mathsf{o}_2 < \mathsf{o}_1 \qquad \mathsf{o}_4 < \mathsf{o}_3 \qquad \mathsf{o}_1 = \mathsf{o}_3 \qquad \mathsf{o}_2 = \mathsf{o}_4$$

can be solved by the assignment $\mathsf{o}_1 = \mathsf{o}_3 = 1$ and $\mathsf{o}_2 = \mathsf{o}_4 = 0$.

Example 3 (Cyclic process: deadlocked!). Now consider the process

$$P' = (\boldsymbol{\nu} x_1 y_1)(\boldsymbol{\nu} x_2 y_2)\big[x_1(v).x_2(w).R \mid y_2[\mathbf{n}'].y_1[\mathbf{n}].Q\big]$$

where $R = x_1().v().x_2().w().\mathbf{0}$ and $Q = y_1[\,].\mathbf{0} \mid \mathbf{n}[\,].\mathbf{0} \mid y_2[\,].\mathbf{0} \mid \mathbf{n}'[\,].\mathbf{0}$. Notice that the order of actions on channels y_1 and y_2 is now swapped, thus causing a deadlock! If we tried to construct a typing derivation for process P', we would have for the right-hand side of the parallel the following:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{y_1[\,].\mathbf{0} \vdash y_1 \colon \mathbf{1}^{\kappa_4}}\;\mathbf{1} \quad \dfrac{}{\mathbf{n}[\,].\mathbf{0} \vdash \mathbf{n} \colon \mathbf{1}^{\kappa_3}}\;\mathbf{1} \quad \dfrac{}{y_2[\,].\mathbf{0} \vdash y_1 \colon \mathbf{1}^{\kappa_2}}\;\mathbf{1} \quad \dfrac{}{\mathbf{n}'[\,].\mathbf{0} \vdash \mathbf{n}' \colon \mathbf{1}^{\kappa_1}}\;\mathbf{1}}{\overline{y_1[\,].\mathbf{0} \mid \mathbf{n}[\,].\mathbf{0} \mid y_2[\,].\mathbf{0} \mid \mathbf{n}'[\,].\mathbf{0} \vdash y_1 \colon \mathbf{1}^{\kappa_4}, \mathbf{n} \colon \mathbf{1}^{\kappa_3}, y_2 \colon \mathbf{1}^{\kappa_2}, \mathbf{n}' \colon \mathbf{1}^{\kappa_1}}}\;\textsc{Mix}^3 \qquad \mathsf{o}_4 < \kappa_4}{y_1[\mathbf{n}].Q \vdash \mathbf{n}' \colon \mathbf{1}^{\kappa_1}, y_2 \colon \mathbf{1}^{\kappa_2}, y_1 \colon \mathbf{1}^{\kappa_3} \otimes^{\mathsf{o}_4} \mathbf{1}^{\kappa_4}}\;\otimes \qquad \mathsf{o}_3 < \mathsf{o}_4}{y_2[\mathbf{n}'].y_1[\mathbf{n}].Q \vdash y_1 \colon \mathbf{1}^{\kappa_3} \otimes^{\mathsf{o}_4} \mathbf{1}^{\kappa_4}, y_2 \colon \mathbf{1}^{\kappa_1} \otimes^{\mathsf{o}_3} \mathbf{1}^{\kappa_2}}\;\otimes$$

Then, the system of equations

$$\mathsf{o}_2 < \mathsf{o}_1 \qquad \mathsf{o}_3 < \mathsf{o}_4 \qquad \mathsf{o}_1 = \mathsf{o}_3 \qquad \mathsf{o}_2 = \mathsf{o}_4$$

has no solution because it requires $\mathsf{o}_2 < \mathsf{o}_3$ and $\mathsf{o}_3 < \mathsf{o}_2$, which is impossible.

Example 1 continued (Cyclic Scheduler).

$$\begin{array}{rcll} \mathit{Sched} & \triangleq & \ldots(\boldsymbol{\nu} a_i b_i)\ldots(\boldsymbol{\nu} c_i d_{(i+1) \texttt{ mod } n})\big(A_0 \mid A_1 \mid \ldots \mid A_{n-1} \mid P_0 \mid P_1 \mid \ldots \mid P_{n-1}\big) & \\ A_0 & \triangleq & c_0[\mathbf{n}_0].d_0(x_0).a_0[\mathbf{m}_0].\mathsf{close}_0 & \\ A_i & \triangleq & d_i(x_i).a_i[\mathbf{m}_i].c_i[\mathbf{n}_i].\mathsf{close}_i & i \in \{1, \ldots, n-1\} \\ P_i & \triangleq & b_i(y_i).Q_i & i \in \{0, \ldots, n-1\} \end{array}$$

By applying the typing rules in Fig. 2 we can derive $\mathit{Sched} \vdash \emptyset$, since it is a closed process, and assign the following types and priorities:

$c_0 : \mathbf{1} \otimes^0 \mathbf{1}$	$d_0 : \bot \parr^{2(n-1)} \bot$	$a_0 : \mathbf{1} \otimes^{2(n-1)+1} \mathbf{1}$	for A_0
$d_i : \bot \parr^{2i-2} \bot$	$a_i : \mathbf{1} \otimes^{2i-1} \mathbf{1}$	$c_i : \mathbf{1} \otimes^{2i} \mathbf{1}$	for A_i, $0 < i < n$
$b_0 : \bot \parr^{2(n-1)+1} \bot$	$b_i : \bot \parr^{2i-1} \bot$		for P_0 and P_i, $0 < i < n$

The priorities of types $\bot$ and $\mathbf{1}$ could be easily assigned as Ex. 2. As the priority of d_{i+1} is $2(i+1) - 2 = 2i$, we can connect it to a_i with a Cycle.

3 Operational Semantics of PCP

In this section we define structural equivalence, the principal β-reduction rules and commuting conversions. The detailed derivations can be found in [18].

We define structural equivalence to be the smallest congruence relation satisfying the following axioms. SC-Ax-Swp allows swapping channels in the forwarding process. SC-Ax-Cycle states that cycle applied to a forwarding process is equivalent to inaction. This allows elimination of unnecessary cycles. Axioms SC-Mix-Nil, SC-Mix-Comm and SC-Mix-Asc state that parallel composition uses the inaction as the neutral element and is commutative and associative. SC-Cycle-Ext is the standard scope extrusion rule. SC-Cycle-Swp allows swapping channels and SC-Cycle-Comm states the commutativity of restriction[1].

SC-Ax-Swp	$x \rightarrow y^A \vdash x : A^\perp, y : A \;\equiv\; y \rightarrow x^{A^\perp} \vdash x : A^\perp, y : A$	
SC-Ax-Cycle	$(\boldsymbol{\nu} x^{A^\perp} y)\, x \rightarrow y^A \vdash \emptyset \;\equiv\; \mathbf{0} \vdash \emptyset$	
SC-Mix-Nil	$\mathbf{0} \mid P \vdash \Gamma \;\equiv\; P \vdash \Gamma$	
SC-Mix-Comm	$P \mid Q \vdash \Gamma, \Delta \;\equiv\; Q \mid P \vdash \Gamma, \Delta$	
SC-Mix-Asc	$P \mid (Q \mid R) \vdash \Gamma, \Delta, \Theta \;\equiv\; (P \mid Q) \mid R \vdash \Gamma, \Delta, \Theta$	
SC-Cycle-Ext	$(\boldsymbol{\nu} x^A y)(P \mid Q) \vdash \Gamma, \Delta \;\equiv\; P \mid (\boldsymbol{\nu} x^A y) Q \vdash \Gamma, \Delta$	$x, y \notin \mathtt{fn}(P)$
SC-Cycle-Swp	$(\boldsymbol{\nu} x^A y) P \vdash \Gamma \;\equiv\; (\boldsymbol{\nu} y^{A^\perp} x) P \vdash \Gamma$	
SC-Cycle-Comm	$(\boldsymbol{\nu} x^A y)(\boldsymbol{\nu} z^B w) P \vdash \Gamma \;\equiv\; (\boldsymbol{\nu} z^B w)(\boldsymbol{\nu} x^A y) P \vdash \Gamma$	

The core of the operational semantics consists of β-reductions. In π-calculus terms these are communication steps; in logical terms they are Cycle-elimination steps. $\beta_{\otimes\parr}$ is given in Fig. 3 to illustrate priorities. It simplifies a cycle connecting x of type $A \otimes^{\mathsf{o}} B$ and y of type $A \parr^{\mathsf{o}} B$, which corresponds to communication between an output on x and an input on y, respectively. Both actions have priority o, which is strictly smaller than any priorities in their typing contexts, respecting the fact that they are top-level prefixes. The remaining β-reductions are summarised below. β_{AxCycle} simplifies a Cycle involving an axiom. $\beta_{\mathbf{1}\bot}$ closes and eliminates channels. $\beta_{\oplus\&}$, similarly to $\beta_{\otimes\parr}$, simplifies a communication between a selection and a branching. $\beta_{!?}$ simplifies a cycle between one server of type $!^{\mathsf{o}} A$ and one client of type $?^{\mathsf{o}} A$. The last two rules differ in the number of clients involved: rule $\beta_{!\text{W}}$ considers no clients, whether $\beta_{!\text{C}}$ considers multiple clients.

[1] Note that associativity of restriction is derived from SC-Mix-Comm and SC-Cycle-Comm.

$$
\dfrac{\dfrac{\dfrac{\begin{array}{c}\mathsf{o} < \mathsf{pr}(\Gamma)\\ P \vdash \Gamma, v\!:\!A, x\!:\!B\end{array}}{x[v].P \vdash \Gamma, x\!:\!A \otimes^{\mathsf{o}} B}\otimes \qquad \dfrac{\begin{array}{c}\mathsf{o} < \mathsf{pr}(\Delta)\\ Q \vdash \Delta, w\!:\!A^{\perp}, y\!:\!B^{\perp}\end{array}}{y(w).Q \vdash \Delta, y\!:\!A^{\perp} \parr^{\mathsf{o}} B^{\perp}}\parr}{x[v].P \mid y(w).Q \vdash \Gamma, \Delta, x\!:\!A \otimes^{\mathsf{o}} B, y\!:\!A^{\perp} \parr^{\mathsf{o}} B^{\perp}}\textsc{Mix}}{(\boldsymbol{\nu} x^{A \otimes^{\mathsf{o}} B} y)\big(x[v].P \mid y(w).Q\big) \vdash \Gamma, \Delta}\textsc{Cycle}
$$

$$
\longrightarrow \qquad \dfrac{\dfrac{P \vdash \Gamma, v\!:\!A, x\!:\!B \qquad Q \vdash \Delta, w\!:\!A^{\perp}, y\!:\!B^{\perp}}{P \mid Q \vdash \Gamma, \Delta, v\!:\!A, x:B, w\!:\!A^{\perp}, y:B^{\perp}}\textsc{Mix}}{(\boldsymbol{\nu} v^{A} w)(\boldsymbol{\nu} x^{B} y)\big(P \mid Q\big) \vdash \Gamma, \Delta}\textsc{Cycle}^2
$$

Fig. 3. β-reduction for $\otimes$ and $\parr$.

$$
\begin{array}{rl}
\beta_{\textsc{AxCycle}} & (\boldsymbol{\nu} y^{A} z)(x \to y^{A} \mid P) \vdash \Gamma, x\!:\!A^{\perp} \;\longrightarrow\; P\{x/z\} \vdash \Gamma, x\!:\!A^{\perp}\\
\beta_{\mathbf{1}\perp} & (\boldsymbol{\nu} x^{A} y)(x[\,].\mathbf{0} \mid y().P) \vdash \Gamma \;\longrightarrow\; P \vdash \Gamma\\
\beta_{\oplus\&} & (\boldsymbol{\nu} x^{\oplus^{\mathsf{o}}\{l_i:B_i\}_{i\in I}} y)\big(x \triangleleft l_j.P \mid y \triangleright \{l_i : Q_i\}_{i\in I}\big) \vdash \Gamma, \Delta \longrightarrow\\
& \hfill (\boldsymbol{\nu} x^{B_j} y)\big(P \mid Q_j\big) \vdash \Gamma, \Delta\\
\beta_{!?} & (\boldsymbol{\nu} x^{!^{\mathsf{o}} A} y)\big(!x(v).P \mid ?y[w].Q\big) \vdash ?\Gamma, \Delta \;\longrightarrow\; (\boldsymbol{\nu} v^{A} w)\big(P \mid Q\big) \vdash ?\Gamma, \Delta\\
\beta_{!\mathrm{W}} & (\boldsymbol{\nu} x^{!^{\mathsf{o}} A} y)\big(!x(v).P \mid Q\big) \vdash ?\Gamma, \Delta \;\longrightarrow\; Q \vdash ?\Gamma, \Delta\\
\beta_{!\mathrm{C}} & (\boldsymbol{\nu} x^{!^{\mathsf{o}} A} y)\big(!x(v).P \mid Q\{y/y', y/y''\}\big) \vdash ?\Gamma, \Delta \;\longrightarrow\\
& \hfill (\boldsymbol{\nu} x'^{\,!^{\mathsf{o}} A} y')(!x'(v').P' \mid (\boldsymbol{\nu} x''^{\,!^{\mathsf{o}} A} y'')(!x''(v'').P'' \mid Q)) \vdash ?\Gamma, \Delta
\end{array}
$$

Commuting conversions, following [12,41], allow communication prefixes to be moved to the conclusion of a typing derivation, corresponding to pulling them out of the scope of Cycle rules. In order to account for the sequence of Cycles, here we use $\widetilde{\cdot}$. Due to this movement, if a prefix on a channel endpoint x with priority o is pulled out at top level, then to preserve priority conditions in the typing rules in Fig. 2, it is necessary to increase priorities of all actions after the prefix on x. This increase is achieved by using $\uparrow^{\mathsf{o}+1}(\cdot)$ in the typing contexts.

$$
\begin{array}{rl}
\kappa_{\perp} & (\boldsymbol{\nu} \widetilde{x}^{\widetilde{A}} \widetilde{y})\big(x().P \mid Q\big) \vdash \Gamma, \Delta, x\!:\perp^{\mathsf{o}} \;\longrightarrow\\
& \hfill x().[(\boldsymbol{\nu} \widetilde{x}^{\widetilde{A}} \widetilde{y})\big(P \mid Q\big)] \vdash \uparrow^{\mathsf{o}+1}\Gamma, \uparrow^{\mathsf{o}+1}\Delta, x\!:\perp^{\mathsf{o}}\\
\kappa_{\otimes} & (\boldsymbol{\nu} \widetilde{x}^{\widetilde{A}} \widetilde{y})\big(x[v].P \mid Q\big) \vdash \Gamma, \Delta, x\!:\!A \otimes^{\mathsf{o}} B \;\longrightarrow\\
& x[v].\big[(\boldsymbol{\nu} \widetilde{x}^{\widetilde{A}} \widetilde{y})\big(P \mid Q\big)\big] \vdash (\uparrow^{\mathsf{o}+1}\Gamma), (\uparrow^{\mathsf{o}+1}\Delta), x\!:\!(\uparrow^{\mathsf{o}+1}A) \otimes^{\mathsf{o}} (\uparrow^{\mathsf{o}+1}B)\\
\kappa_{\parr} & (\boldsymbol{\nu} \widetilde{x}^{\widetilde{A}} \widetilde{y})\big(x(w).P \mid Q\big) \vdash \Gamma, \Delta, x\!:\!A \parr^{\mathsf{o}} B \;\longrightarrow\\
& x(w).\big[(\boldsymbol{\nu} \widetilde{x}^{\widetilde{A}} \widetilde{y})\big(P \mid Q\big)\big] \vdash (\uparrow^{\mathsf{o}+1}\Gamma), (\uparrow^{\mathsf{o}+1}\Delta), x\!:\!(\uparrow^{\mathsf{o}+1}A) \parr^{\mathsf{o}} (\uparrow^{\mathsf{o}+1}B)\\
\kappa_{\oplus} & (\boldsymbol{\nu} \widetilde{x}^{\widetilde{A}} \widetilde{y})(x \triangleleft l_j.P \mid Q) \vdash \Gamma, \Delta, x\!:\!\oplus^{\mathsf{o}}\{l_i : B_i\}_{i\in I} \;\longrightarrow\\
& x \triangleleft l_j.\big[(\boldsymbol{\nu} \widetilde{x}^{\widetilde{A}} \widetilde{y})\big(P \mid Q\big)\big] \vdash (\uparrow^{\mathsf{o}+1}\Gamma), (\uparrow^{\mathsf{o}+1}\Delta), x\!:\!\oplus^{\mathsf{o}}\{l_i : \uparrow^{\mathsf{o}+1}B_i\}_{i\in I}\\
\kappa_{\&} & (\boldsymbol{\nu} \widetilde{x}^{\widetilde{A}} \widetilde{y})(x \triangleright \{l_i : P_i\}_{i\in I} \mid Q) \vdash \Gamma, \Delta, x\!:\!\&^{\mathsf{o}}\{l_i : B_i\}_{i\in I} \;\longrightarrow\\
& x \triangleright \{l_i : (\boldsymbol{\nu} \widetilde{x}^{\widetilde{A}} \widetilde{y})\big(P_i \mid Q\big)\}_{i\in I} \vdash (\uparrow^{\mathsf{o}+1}\Gamma), (\uparrow^{\mathsf{o}+1}\Delta), x\!:\!\&^{\mathsf{o}}\{l_i : \uparrow^{\mathsf{o}+1}B_i\}_{i\in I}\\
\kappa_{?} & (\boldsymbol{\nu} \widetilde{x}^{\widetilde{A}} \widetilde{y})\big(?x[w].P \mid Q\big) \vdash \Gamma, \Delta, x\!:\,?^{\mathsf{o}} A \;\longrightarrow\\
& \hfill ?x[w].\big[(\boldsymbol{\nu} \widetilde{x}^{\widetilde{A}} \widetilde{y})\big(P \mid Q\big)\big] \vdash (\uparrow^{\mathsf{o}+1}\Gamma), (\uparrow^{\mathsf{o}+1}\Delta), x\!:\,?^{\mathsf{o}}(\uparrow^{\mathsf{o}+1}A)\\
\kappa_{!} & (\boldsymbol{\nu} \widetilde{x}^{\,\widetilde{?^{\mathsf{o}} A}} \widetilde{y})\big(!x(v).P \mid Q\big) \vdash ?\Gamma, \Delta, x\!:\,!^{\mathsf{o}} A \;\longrightarrow\\
& \hfill !x(v).\big[(\boldsymbol{\nu} \widetilde{x}^{\,\widetilde{?^{\mathsf{o}} A}} \widetilde{y})\big(P \mid Q\big)\big] \vdash (\uparrow^{\mathsf{o}+1}\Gamma), (\uparrow^{\mathsf{o}+1}\Delta), x\!:\,!^{\mathsf{o}}(\uparrow^{\mathsf{o}+1}A)
\end{array}
$$

Finally, we give the following additional reduction rules: closure under structural equivalence, and two congruence rules, for restriction and for parallel.

Close-Equiv	$P \equiv Q \quad Q \longrightarrow R \quad R \equiv S$ implies $P \longrightarrow S$
Cong-Cycle	$P \longrightarrow Q$ implies $(\boldsymbol{\nu} x^A y)P \longrightarrow (\boldsymbol{\nu} x^A y)Q$
Cong-Mix	$P \longrightarrow Q$ implies $P \mid R \longrightarrow Q \mid R$

4 Results for PLL and PCP

4.1 Cycle-elimination for PLL

We start with results for Cycle-elimination for PLL; thus here we refer to A, B as propositions, rather than types. The detailed proofs can be found in [18].

Definition 6. *The* degree *function* $\partial(\cdot)$ *on propositions is defined by:*
– $\partial(\mathbf{1}^\circ) = \partial(\perp^\circ) = 1$
– $\partial(A \otimes^\circ B) = \partial(A \parr^\circ B) = \partial(A) + \partial(B) + 1$
– $\partial(\&^\circ\{l_i : A_i\}_{i\in I}) = \partial(\oplus^\circ\{l_i : A_i\}_{i\in I}) = \sum_{i\in I}\{\partial(A_i)\} + 1$
– $\partial(\,?^\circ A) = \partial(\,!^\circ A) = \partial(A) + 1.$

Definition 7. *A* Maxicut *is a maximal sequence of* Mix *and* Cycle *rules, ending with a* Cycle *rule.*

Maximality means that the rules applied immediately before a Maxicut are any rules in Fig. 2, other than Mix or Cycle. The order in which Mix and Cycle rules are applied within a Maxicut is irrelevant. However, Prop. 1, which follows directly from structural equivalence (§ 3), allows us to simplify a Maxicut.

Proposition 1 (Canonical Maxicut). *Given an arbitrary* Maxicut*, it is always possible to obtain from it a* canonical Maxicut *consisting of a sequence of only* Mix *rules followed by a sequence of only* Cycle *rules.*

Definition 8. *A* single-Mix Maxicut *contains only one* Mix *rule.*
$A_1, \ldots, A_n, A$ *are* Maxicut propositions *if they are eliminated by a* Maxicut*.*
The degree of a sequence of Cycles *is the sum of the degrees of the eliminated propositions.*
The degree of a Maxicut *is the sum of the degrees of the* Cycles *in it.*
The degree of a proof π, $d(\pi)$, *is the* sup *of the degrees of its* Maxicuts*, implying* $d(\pi) = 0$ *if and only if proof* π *has no* Cycles*.*
The height of a proof π, $h(\pi)$, *is the height of its tree, and it is defined as* $h(\pi) = \sup\big(h(\pi_i)\big)_{i\in I} + 1$, *where* $\{\pi_i\}_{i\in I}$ *are the subproofs of* π*.*

Maxicut has some similarities with the derived Multicut: it generalises Multicut in the number of Mixes, and a single-Mix Maxicut is an occurrence of Multicut.

The core of Cycle-elimination for our PLL, as for Cut-elimination for CLL [10,25], is the Principal Lemma (Lem. 3), which eliminates a Cycle by either (i) replacing it with another Cycle on simpler propositions, or (ii) pushing it further up the proof tree. Item (i) corresponds to (the logical part of) β-reductions (§ 3); and (ii) corresponds to (the logical part of) commuting conversions (§ 3).

Exceptionally, $\beta_{!\mathrm{C}}$ reduces the original proof in a way that neither (i) nor (ii) are respected. In order to cope with this case, we introduce Lem. 2, which is inspired by Lem B.1.3 in Bräuner [10], and adapted to our PLL. Lem. 2 allows us to reduce the degree of a proof ending with a single-Mix Maxicut and having the same degree as the whole proof, and where the last rule applied on the left hand-side immediate subproof is !. Let $[n]$ denote the set $\{1, \ldots, n\}$.

Lemma 2 (Inspired by B.1.3 in Bräuner [10]). *Let τ be a proof of the following form, ending with a single-*Mix Maxicut*:*

$$
\dfrac{\dfrac{\dfrac{\begin{matrix}\pi \\ \vdots \\ \mathsf{o} < \mathsf{pr}(?\Gamma) \\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i \\ \vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{matrix}}{\vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A}\ ! \qquad \dfrac{\begin{matrix}\pi' \\ \vdots \\ \mathsf{o} < \mathsf{pr}(\Delta) \\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i \quad \forall j \in [k] : \mathsf{o} \leqslant \kappa_j \\ \vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (\,?^{\kappa_j} A^{\perp})_{j\in[k]}\end{matrix}}{\vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \mathrm{C}^{k-1}}{\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \textsc{Mix}}{\vdash ?\Gamma, \Delta}\ \textsc{Cycle}
$$

where $d(\pi) < d(\tau)$ and $d(\pi') < d(\tau)$. Then, there is a proof τ' of $\vdash ?\Gamma, \Delta$ such that $d(\tau') < d(\tau)$.

Proof. Induction on $h(\pi')$, with a case-analysis on the last rule applied in π'. $\square$

Lemma 3 (The Principal Lemma). *Let τ be a proof of $\vdash \Gamma$, ending with a canonical* Maxicut*:*

$$
\dfrac{\dfrac{\pi_1 \ldots \pi_m}{\vdash \Gamma, A_1, ..., A_n, A, A_1^{\perp}, ..., A_n^{\perp}, A^{\perp}}\ \textsc{Mix}}{\vdash \Gamma}\ \textsc{Cycle}
$$

such that for all $i \in [m]$, $d(\pi_i) < d(\tau)$. Then there is a proof τ' of $\vdash \uparrow^t \Gamma$, for some $t \geqslant 0$, such that $d(\tau') < d(\tau)$.

Proof. The proof is by induction on $\sum_{i\in[m]} h(\pi_i)$. Let r_i be the last rule applied in π_i, for $i \in [m]$ and let C_{r_i} be the proposition introduced by r_i. Consider the proposition with the *smallest* priority. If the proposition is not unique, just pick one. Let this proposition be C_{r_k}. Then, π_k is the following proof: $\dfrac{\cdots}{\vdash \Gamma', C_{r_k}}\ r_k$

We proceed by cases on π_k.

– r_k is $\otimes$ on one of the Maxicut propositions $A_1, \ldots, A_n, A$. Without loss of generality, suppose r_k is applied on A, meaning $A = E \otimes^{\mathsf{o}} F$ for some E and F and $\mathsf{o} \geqslant 0$. By $\otimes$ rule in Fig. 2, $\mathsf{o} < \mathsf{pr}(\Gamma')$. Since A is a Maxicut proposition, by Def. 2, $A^{\perp} = E^{\perp} \parr^{\mathsf{o}} F^{\perp}$. Since $\mathsf{o} < \mathsf{pr}(\Gamma')$ and $\mathsf{pr}(A^{\perp}) = \mathsf{o}$, it must be that $A^{\perp}$ is in another proof, say π_h: $\dfrac{\cdots}{\vdash \Gamma'', E^{\perp} \parr^{\mathsf{o}} F^{\perp}}\ r_h$

Consider the case where r_h is a multiplicative, additive, exponential or $\perp$ rule in Fig. 2. Suppose r_h is applied on C_{r_h} which is not $A^{\perp}$. All the mentioned rules require $\mathsf{pr}(C_{r_h}) < \mathsf{pr}(\Gamma'', E^\perp \parr^{\mathsf{o}} F^\perp \setminus C_{r_h})$, implying $\mathsf{pr}(C_{r_h}) < \mathsf{pr}(E^\perp \parr^{\mathsf{o}} F^\perp) = \mathsf{pr}(E \otimes^{\mathsf{o}} F) = \mathsf{o}$. This contradicts the fact that o is the smallest priority. Hence, r_h must be a $\parr$ introducing $A^\perp$.

We construct proof τ_A ending with a single-Mix Maxicut applied on *at least* A:

$$\dfrac{\dfrac{\dfrac{\overset{\pi_\otimes}{\vdots}}{\vdash \Gamma', E, F \quad \mathsf{o} < \mathsf{pr}(\Gamma')}}{\vdash \Gamma', E \otimes^{\mathsf{o}} F}\otimes \quad \dfrac{\dfrac{\overset{\pi_\parr}{\vdots}}{\vdash \Gamma'', E^\perp, F^\perp \quad \mathsf{o} < \mathsf{pr}(\Gamma'')}}{\vdash \Gamma'', E^\perp \parr^{\mathsf{o}} F^\perp}\parr}{\dfrac{\vdash \Gamma', \Gamma'', E \otimes^{\mathsf{o}} F, E^\perp \parr^{\mathsf{o}} F^\perp}{\vdash \Gamma'''}\ \textsc{Cycle}}\ \textsc{Mix}$$

Then, by structural equivalence, we can rewrite τ in terms of τ_A. By applying $\beta_{\otimes\parr}$ on τ_A (only considering the logical part), we obtain a proof τ'_A such that $d(\tau'_A) < d(\tau_A) \leq d(\tau)$, because $\partial(E)+\partial(F) < \partial(E \otimes^{\mathsf{o}} F)$. We can then construct τ' by substituting τ'_A for τ_A in τ, which concludes this case.

– r_k is ! on one of the Maxicut propositions $A_1, \ldots, A_n, A$. Without loss of generality, suppose r_k introduces A, implying that $A = !^{\mathsf{o}} A'$ for some A' and $\mathsf{o} \geqslant 0$. Then π_k is the following proof:

$$\dfrac{\overset{\pi_!}{\vdots} \\ \vdash ?\Theta, A' \quad \mathsf{o} < \mathsf{pr}(?\Theta)}{\vdash ?\Theta, !^{\mathsf{o}} A'}\ !$$

where $\Gamma' = ?\Theta$. Since A is a Maxicut proposition, by duality $A^\perp = ?^{\mathsf{o}} A'^\perp$. Since $\mathsf{o} < \mathsf{pr}(\Gamma')$ and $\mathsf{pr}(A^\perp) = \mathsf{o}$, it must be that $A^\perp$ is in another proof. Let it be π_h for $h \in [m]$ and $h \neq k$. Then we apply Lem. 2 to π_k and π_h, obtaining a proof which we use to construct τ', as we did in the previous case. □

Lemma 4. *Given a proof τ of $\vdash \Gamma$, such that $d(\tau) > 0$, then for some $t \geqslant 0$ there is a proof τ' of $\vdash \uparrow^t \Gamma$ such that $d(\tau') < d(\tau)$.*

Proof. By induction on $h(\tau)$. We have the following cases.

– If τ ends in a Maxicut whose degree is *the same as* the degree of τ:

$$\dfrac{\dfrac{\pi_1 \ldots \pi_m}{\vdash \Gamma, A_1, ..., A_n, A, A_1^\perp, ..., A_n^\perp, A^\perp}\ \textsc{Mix}^m}{\vdash \Gamma}\ \textsc{Cycle}^{n+1}$$

we can apply the induction hypothesis to the subproofs of τ right before the last Mix preceding the sequence of Cycle. This allows us to reduce their degrees to become smaller than $d(\tau)$. Then we use Lem. 3.

– Otherwise, by using the inductive hypothesis on the immediate subproofs to reduce their degree, we also reduce the degree of the whole proof. □

Theorem 1 (Cycle-elimination). *Given any proof of $\vdash \Gamma$, we can construct a Cycle-free proof of $\vdash \uparrow^t \Gamma$, for some $t \geqslant 0$.*

Proof. Iteration on Lem. 4. □

Cycle-elimination increases the priorities of the propositions in Γ. This is solely due to the (logical part of) our commuting conversions in § 3.

4.2 Deadlock-Freedom for PCP

Theorem 2 (Subject Reduction). *If $P \vdash \Gamma$ and $P \longrightarrow Q$, then $Q \vdash {\uparrow^t} \Gamma$, for some $t \geqslant 0$.*

Proof. Follows from the β-reductions and commuting conversions in § 3. □

Definition 9. *A process is a* Cycle *if it is of the form $(\boldsymbol{\nu} x^A y)P$.*

Theorem 3 (Top-Level Deadlock-Freedom). *If $P \vdash \Gamma$ and P is a* Cycle*, then there is some Q such that $P \longrightarrow^* Q$ and Q is not a* Cycle*.*

Proof. The interpretation of Lem. 3 for PCP is that either (i) a top-level communication occurs, corresponding to a β-reduction, or (ii) commuting conversions are used to push Cycle further inwards in a process. Consequently, iterating Lem. 3 results in eliminating top-level Cycles. □

Eliminating all Cycles, as specified by Thm. 1, would correspond to a semantics in which reduction occurs under prefixes, as discussed by Wadler [41]. In order to achieve this, we would need to introduce additional congruence rules, such as:

$$\frac{P \longrightarrow Q}{x(y).P \longrightarrow x(y).Q}$$

and similarly for other actions. Reductions of this kind are not present in the π-calculus, and we also omit them in our framework.

However, we can eliminate all Cycles in a proof of $\vdash \emptyset$, corresponding to full deadlock-freedom for closed processes. Kobayashi's type system [32] satisfies the same property.

Theorem 4 (Deadlock-Freedom for Closed Processes). *If $P \vdash \emptyset$, then either $P \equiv \mathbf{0}$ or there is Q such that $P \longrightarrow Q$.*

Proof. This follows from Thm. 2 and Thm. 3, because if $Q \vdash \emptyset$ and Q is not a Cycle then Q must be a parallel composition of $\mathbf{0}$ processes. □

5 Related Work and Conclusion

Cycle and Multicut rules were explored by Abramsky *et al.* [2,3,4] in the context of $*$-autonomous categories. That work is not directly comparable with ours, as it only presented a typed semantics for CCS-like processes and did not give a type system for a language or a term assignment for a logical system. Atkey *et al.* [5] added a Multicut rule to CP, producing an isomorphism between $\otimes$ and $\invamp$, but they did not consider deadlock-freedom.

In Kobayashi's original type-theoretic approach to deadlock-freedom [29], priorities were abstract tags from a partially ordered set. In later work abstract tags were simplified to natural numbers, and priorities were replaced by pairs of obligations and capabilities [30,32]. The latter change allows more processes to be typed, at the expense of a more complex type system. Padovani [36] adapted

Kobayashi's approach to session types, and later on he simplified it to a single priority for linear π-calculus [37]. Then, the single priority technique can be transferred to session types by the encoding of session types into linear types [33,19,16,17]. For simplicity, we have opted for single priorities, as Padovani [37].

The first work on progress for session types, by Dezani-Ciancaglini *et al.* [22,15], guaranteed the property by allowing only one active session at a time. Later work [21] introduced a partial order on channels in Kobayashi-style [29]. Bettini *et al.* [9] applied similar ideas to multiparty session types. The main difference with our work is that we associate priorities with individual communication operations, rather than with entire channels. Carbone *et al.* [13] proved that progress is a compositional form of lock-freedom and introduced a new technique for progress in session types by adopting Kobayashi's type system and the encoding of session types [19]. Vieira and Vasconcelos [40] used single priorities and an abstract partial order in session types to guarantee deadlock-freedom.

The linear logic approach to deadlock-free session types started with Caires and Pfenning [12], based on dual intuitionistic linear logic, and was later formulated for classical linear logic by Wadler [41]. All subsequent work on linear logic and session types enforces deadlock-freedom by forbidding cyclic connections. In their original work, Caires and Pfenning commented that it would be interesting to compare process typability in their system with other approaches including Kobayashi's and Dezani-Ciancaglini's. However, we are aware of only one comparative study of the expressivity of type systems for deadlock-freedom, by Dardha and Pérez [20]. They compared Kobayashi-style typing and CLL typing, and proved that CLL corresponds to Kobayashi's system with the restriction that only single cuts, not multicuts, are allowed.

In this paper, we have presented a new logic, priority-based linear logic (PLL), and a term assignment system, priority-based CP (PCP), that increase the expressivity of deadlock-free session type systems, by combining Caires and Pfenning's linear logic-based approach and Kobayashi's priority-based type system. The novel feature of PLL and PCP is Cycle, which allows cyclic process structures to be formed if they do not violate ordering conditions on the priorities of prefixes. Following the propositions-as-types paradigm, we prove a Cycle-elimination theorem analogous to the standard Cut-elimination theorem. As a result of this theorem, we obtain deadlock-freedom for a class of π-calculus processes which is larger than the class typed by Caires and Pfenning. In particular, these are processes that typically share more than one channel in parallel.

There are two main directions for future work. First, develop a type system for a functional language, priority-based GV, and translate it into PCP, along the lines of Lindley and Morris' [34] translation of GV [41] into CP. Second, extend PCP to allow recursion and sharing [6], in order to support more general concurrent programming, while maintaining deadlock-freedom, as well as termination, or typed behavioural equivalence.

Acknowledgements We are grateful for suggestions and feedback from the anonymous reviewers and colleagues: Wen Kokke, Sam Lindley, Roly Perera, Frank Pfenning, Carsten Schürmann and Philip Wadler.

References

1. S. Abramsky. Proofs as processes. *Theoretical Computer Science*, 135(1):5–9, 1994.
2. S. Abramsky, S. J. Gay, and R. Nagarajan. Interaction categories and the foundations of typed concurrent programming. In M. Broy, editor, *Proceedings of the NATO Advanced Study Institute on Deductive Program Design*, pages 35–113, 1996.
3. S. Abramsky, S. J. Gay, and R. Nagarajan. A type-theoretic approach to deadlock-freedom of asynchronous systems. In *TACS*, volume 1281 of *LNCS*, pages 295–320. Springer, 1997.
4. S. Abramsky, S. J. Gay, and R. Nagarajan. A specification structure for deadlock-freedom of synchronous processes. *Theoretical Computer Science*, 222(1–2):1–53, 1999.
5. R. Atkey, S. Lindley, and J. G. Morris. Conflation confers concurrency. In *A List of Successes That Can Change the World—Essays Dedicated to Philip Wadler on the Occasion of His 60th Birthday*, volume 9600 of *LNCS*, pages 32–55. Springer, 2016.
6. S. Balzer and F. Pfenning. Manifest sharing with session types. *Proceedings of the ACM on Programming Languages*, 1(ICFP):37:1–37:29, 2017.
7. A. Barber. Dual intuitionistic linear logic. Technical Report ECS-LFCS-96-347, University of Edinburgh, 1996. www.lfcs.inf.ed.ac.uk/reports/96/ECS-LFCS-96-347.
8. G. Bellin and P. J. Scott. On the pi-calculus and linear logic. *Theoretical Computer Science*, 135(1):11–65, 1994.
9. L. Bettini, M. Coppo, L. D'Antoni, M. D. Luca, M. Dezani-Ciancaglini, and N. Yoshida. Global progress in dynamically interleaved multiparty sessions. In *CONCUR*, volume 5201 of *LNCS*, pages 418–433. Springer, 2008.
10. T. Bräuner. Introduction to linear logic. Technical Report BRICS LS-96-6, Basic Research Institute in Computer Science, University of Aarhus, 1996.
11. L. Caires and J. A. Pérez. Linearity, control effects, and behavioral types. In *ESOP*, volume 10201 of *LNCS*, pages 229–259. Springer, 2017.
12. L. Caires and F. Pfenning. Session types as intuitionistic linear propositions. In *CONCUR*, volume 6269 of *LNCS*, pages 222–236. Springer, 2010.
13. M. Carbone, O. Dardha, and F. Montesi. Progress as compositional lock-freedom. In *COORDINATION*, volume 8459 of *LNCS*, pages 49–64. Springer, 2014.
14. M. Carbone, S. Lindley, F. Montesi, C. Schürmann, and P. Wadler. Coherence generalises duality: A logical explanation of multiparty session types. In *CONCUR*, volume 59 of *LIPIcs*, pages 33:1–33:15. Schloss Dagstuhl—Leibniz-Zentrum für Informatik, 2016.
15. M. Coppo, M. Dezani-Ciancaglini, and N. Yoshida. Asynchronous session types and progress for object-oriented languages. In *FMOODS*, volume 4468 of *LNCS*, pages 1–31. Springer, 2007.
16. O. Dardha. Recursive session types revisited. In *BEAT*, volume 162 of *EPTCS*, pages 27–34, 2014.
17. O. Dardha. *Type Systems for Distributed Programs: Components and Sessions*, volume 7 of *Atlantis Studies in Computing*. Springer / Atlantis Press, 2016.
18. O. Dardha and S. J. Gay. A New Linear Logic for Deadlock-Free Session Typed Processes. In *21st International Conference on Foundations of Software Science and Computation Structures, FoSSaCS*, 2018. (Extended Version) http://www.dcs.gla.ac.uk/~ornela/publications/DG18-Extended.pdf.

19. O. Dardha, E. Giachino, and D. Sangiorgi. Session types revisited. In *PPDP*, pages 139–150. ACM, 2012.
20. O. Dardha and J. A. Pérez. Comparing deadlock-free session typed processes. In *EXPRESS/SOS*, volume 190 of *EPTCS*, pages 1–15, 2015.
21. M. Dezani-Ciancaglini, U. de'Liguoro, and N. Yoshida. On progress for structured communications. In *TGC*, volume 4912 of *LNCS*, pages 257–275. Springer, 2009.
22. M. Dezani-Ciancaglini, D. Mostrous, N. Yoshida, and S. Drossopoulou. Session types for object-oriented languages. In *ECOOP*, volume 4067 of *LNCS*, pages 328–352. Springer, 2006.
23. S. J. Gay and V. T. Vasconcelos. Linear type theory for asynchronous session types. *Journal of Functional Programming*, 20(1):19–50, 2010.
24. J. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.
25. J.-Y. Girard, P. Taylor, and Y. Lafont. *Proofs and Types*. Cambridge University Press, New York, NY, USA, 1989.
26. K. Honda. Types for dyadic interaction. In *CONCUR*, volume 715 of *LNCS*, pages 509–523. Springer, 1993.
27. K. Honda, V. T. Vasconcelos, and M. Kubo. Language primitives and type discipline for structured communication-based programming. In *ESOP*, volume 1381 of *LNCS*, pages 122–138. Springer, 1998.
28. N. Kobayashi. TyPiCal: Type-based static analyzer for the pi-calculus. `www-kb.is.s.u-tokyo.ac.jp/~koba/typical`.
29. N. Kobayashi. A partially deadlock-free typed process calculus. *ACM Transactions on Programming Languages and Systems*, 20(2):436–482, 1998.
30. N. Kobayashi. A type system for lock-free processes. *Information and Computation*, 177(2):122–159, 2002.
31. N. Kobayashi. Type systems for concurrent programs. In *10th Anniversary Colloquium of UNU/IIST*, pages 439–453, 2002.
32. N. Kobayashi. A new type system for deadlock-free processes. In *CONCUR*, volume 4137 of *LNCS*, pages 233–247. Springer, 2006.
33. N. Kobayashi. Type systems for concurrent programs. Extended version of [31], Tohoku University, 2007.
34. S. Lindley and J. G. Morris. A semantics for propositions as sessions. In *ESOP*, pages 560–584, 2015.
35. R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.
36. L. Padovani. From lock freedom to progress using session types. In *PLACES*, volume 137 of *EPTCS*, pages 3–19, 2013.
37. L. Padovani. Deadlock and Lock Freedom in the Linear π-Calculus. In *CSL-LICS*, pages 72:1–72:10. ACM, 2014.
38. K. Takeuchi, K. Honda, and M. Kubo. An interaction-based language and its typing system. In *PARLE '94*, volume 817 of *LNCS*, pages 398–413. Springer, 1994.
39. B. Toninho, L. Caires, and F. Pfenning. Dependent session types via intuitionistic linear type theory. In *PPDP*, pages 161–172. ACM, 2011.
40. H. T. Vieira and V. T. Vasconcelos. Typing progress in communication-centred systems. In *COORDINATION*, volume 7890 of *LNCS*, pages 236–250. Springer, 2013.
41. P. Wadler. Propositions as sessions. In *ICFP*, pages 273–286. ACM, 2012.
42. P. Wadler. Propositions as types. *Communications of the ACM*, 58(12):75–84, 2015.

A Structural Equivalence

SC-Ax-Swp

$$\frac{}{x \to y^{A} \vdash x : A^{\perp}, y : A}\,\textsc{Ax} \qquad \equiv \qquad \frac{}{y \to x^{A^{\perp}} \vdash x : A^{\perp}, y : A}\,\textsc{Ax}$$

SC-Ax-Cycle

$$\frac{\dfrac{}{x \to y^{A} \vdash x : A^{\perp}, y : A}\,\textsc{Ax}}{(\boldsymbol{\nu} x^{A^{\perp}} y)\, x \to y^{A} \vdash \emptyset}\,\textsc{Cycle} \qquad \equiv \qquad \mathbf{0} \vdash \emptyset$$

SC-Mix-Nil

$$\frac{\dfrac{}{\mathbf{0} \vdash \emptyset}\,\emptyset \qquad P \vdash \Gamma}{\mathbf{0} \mid P \vdash \Gamma}\,\textsc{Mix} \qquad \equiv \qquad P \vdash \Gamma$$

SC-Mix-Comm

$$\frac{P \vdash \Gamma \qquad Q \vdash \Delta}{P \mid Q \vdash \Gamma, \Delta}\,\textsc{Mix} \qquad \equiv \qquad \frac{Q \vdash \Delta \qquad P \vdash \Gamma}{Q \mid P \vdash \Gamma, \Delta}\,\textsc{Mix}$$

SC-Mix-Asc

$$\frac{P \vdash \Gamma \qquad \dfrac{Q \vdash \Delta \qquad R \vdash \Theta}{Q \mid R \vdash \Delta, \Theta}\,\textsc{Mix}}{P \mid (Q \mid R) \vdash \Gamma, \Delta, \Theta}\,\textsc{Mix} \qquad \equiv \qquad \frac{\dfrac{P \vdash \Gamma \qquad Q \vdash \Delta}{P \mid Q \vdash \Gamma, \Delta}\,\textsc{Mix} \qquad R \vdash \Theta}{(P \mid Q) \mid R \vdash \Gamma, \Delta, \Theta}\,\textsc{Mix}$$

SC-Cycle-Ext

$$\frac{\dfrac{P \vdash \Gamma \qquad Q \vdash \Delta, x : A, y : A^{\perp} \qquad x, y \notin \mathtt{fn}(P)}{P \mid Q \vdash \Gamma, \Delta, x : A, y : A^{\perp}}\,\textsc{Mix}}{(\boldsymbol{\nu} x^{A} y)(P \mid Q) \vdash \Gamma, \Delta}\,\textsc{Cycle} \equiv \frac{P \vdash \Delta \qquad \dfrac{Q \vdash \Gamma, x : A, y : A^{\perp}}{(\boldsymbol{\nu} x^{A} y) P \vdash \Gamma}\,\textsc{Cycle} \qquad x, y \notin \mathtt{fn}(P)}{P \mid (\boldsymbol{\nu} x^{A} y) Q \vdash \Gamma, \Delta}\,\textsc{Mix}$$

SC-Cycle-Swp

$$\frac{P \vdash \Gamma, x : A, y : A^{\perp}}{(\boldsymbol{\nu} x^{A} y) P \vdash \Gamma}\,\textsc{Cycle} \qquad \equiv \qquad \frac{P \vdash \Gamma, x : A, y : A^{\perp}}{(\boldsymbol{\nu} y^{A^{\perp}} x) P \vdash \Gamma}\,\textsc{Cycle}$$

SC-Cycle-Comm

$$\frac{P \vdash \Gamma, x : A, y : A^{\perp}, z : B, w : B^{\perp}}{(\boldsymbol{\nu} x^{A} y)(\boldsymbol{\nu} z^{B} w) P \vdash \Gamma}\,\textsc{Cycle}^{2} \qquad \equiv \qquad \frac{P \vdash \Gamma, x : A, y : A^{\perp}, z : B, w : B^{\perp}}{(\boldsymbol{\nu} z^{B} w)(\boldsymbol{\nu} x^{A} y) P \vdash \Gamma}\,\textsc{Cycle}^{2}$$

Fig. 4. Structural equivalence.

B Principal β-Reductions

$\beta_{\textsc{AxCycle}}$

$$\dfrac{\dfrac{\dfrac{}{x\to y^{A} \vdash x:A^{\perp}, y:A}\;\textsc{Ax} \qquad P\vdash \Gamma, z:A^{\perp}}{x\to y^{A} \mid P \vdash \Gamma, x:A^{\perp}, y:A, z:A^{\perp}}\;\textsc{Mix}}{(\boldsymbol{\nu} y^{A} z)(x\to y^{A} \mid P)\vdash \Gamma, x:A^{\perp}}\;\textsc{Cycle} \qquad\longrightarrow\qquad P\{x/z\}\vdash\Gamma, x:A^{\perp}$$

$\beta_{\mathbf{1}\perp}$

$$\dfrac{\dfrac{\dfrac{}{x[\,].\mathbf{0}\vdash x:\mathbf{1}^{\circ}}\;\mathbf{1} \qquad \dfrac{P\vdash\Gamma \qquad \mathsf{o}<\mathsf{pr}(\Gamma)}{y().P\vdash y:\perp^{\circ},\Gamma}\;\perp}{x[\,].\mathbf{0}\mid y().P\vdash\Gamma, x:\mathbf{1}^{\circ}, y:\perp^{\circ}}\;\textsc{Mix}}{(\boldsymbol{\nu} x^{A} y)(x[\,].\mathbf{0}\mid y().P)\vdash\Gamma}\;\textsc{Cycle} \qquad\longrightarrow\qquad P\vdash\Gamma$$

$\beta_{\otimes⅋}$

$$\dfrac{\dfrac{\dfrac{\mathsf{o}<\mathsf{pr}(\Gamma) \atop P\vdash\Gamma, v:A, x:B}{x[v].P\vdash\Gamma, x:A\otimes^{\circ}B}\;\otimes \qquad \dfrac{\mathsf{o}<\mathsf{pr}(\Delta) \atop Q\vdash\Delta, w:A^{\perp}, y:B^{\perp}}{y(w).Q\vdash\Delta, y:A^{\perp}⅋^{\circ}B^{\perp}}\;⅋}{x[v].P\mid y(w).Q\vdash\Gamma,\Delta, x:A\otimes^{\circ}B, y:A^{\perp}⅋^{\circ}B^{\perp}}\;\textsc{Mix}}{(\boldsymbol{\nu} x^{A\otimes^{\circ}B}y)\big(x[v].P\mid y(w).Q\big)\vdash\Gamma,\Delta}\;\textsc{Cycle}$$

$$\longrightarrow\qquad \dfrac{\dfrac{P\vdash\Gamma, v:A, x:B \qquad Q\vdash\Delta, w:A^{\perp}, y:B^{\perp}}{P\mid Q\vdash\Gamma,\Delta, v:A, x:B, w:A^{\perp}, y:B^{\perp}}\;\textsc{Mix}}{(\boldsymbol{\nu} v^{A}w)(\boldsymbol{\nu} x^{B}y)\big(P\mid Q\big)\vdash\Gamma,\Delta}\;\textsc{Cycle}^{2}$$

$\beta_{\oplus\&}$

$$\dfrac{\dfrac{\dfrac{\mathsf{o}<\mathsf{pr}(\Gamma) \atop P\vdash\Gamma, x:B_j \quad j\in I}{x\triangleleft l_j.P\vdash\Gamma, x:\oplus^{\circ}\{l_i:B_i\}_{i\in I}}\;\oplus \qquad \dfrac{\mathsf{o}<\mathsf{pr}(\Delta) \atop \forall i\in I.(Q_i\vdash\Delta, y:B_i^{\perp})}{y\triangleright\{l_i:Q_i\}_{i\in I}\vdash\Delta y:\&^{\circ}\{l_i:B_i^{\perp}\}_{i\in I}}\;\&}{x\triangleleft l_j.P\mid y\triangleright\{l_i:Q_i\}_{i\in I}\vdash\Gamma,\Delta, x:\oplus^{\circ}\{l_i:B_i\}_{i\in I}, y:\&^{\circ}\{l_i:B_i{}^{\perp}\}_{i\in I}}\;\textsc{Mix}}{(\boldsymbol{\nu} x^{\oplus^{\circ}\{l_i:B_i\}_{i\in I}}y)\big(x\triangleleft l_j.P\mid y\triangleright\{l_i:Q_i\}_{i\in I}\big)\vdash\Gamma,\Delta}\;\textsc{Cycle}$$

$$\longrightarrow\qquad \dfrac{\dfrac{P\vdash\Gamma, x:B_j \qquad Q_j\vdash\Delta, y:B_j^{\perp} \qquad j\in I}{P\mid Q_j\vdash\Gamma,\Delta, x:B_j, y:B_j^{\perp}}\;\textsc{Mix}}{(\boldsymbol{\nu} x^{B_j}y)\big(P\mid Q_j\big)\vdash\Gamma,\Delta}\;\textsc{Cycle}$$

Fig. 5. β-reductions: axiom, units, multiplicatives and additives.

$\beta_{!?}$

$$\dfrac{\dfrac{\dfrac{P \vdash ?\Gamma, v : A \quad \mathsf{o} < \mathsf{pr}(?\Gamma)}{!x(v).P \vdash ?\Gamma, x : !^{\mathsf{o}} A}\;! \qquad \dfrac{Q \vdash \Delta, w : A^{\perp} \quad \mathsf{o} < \mathsf{pr}(\Delta)}{?y[w].Q \vdash \Delta, y : ?^{\mathsf{o}} A^{\perp}}\;?}{!x(v).P \mid ?y[w].Q \vdash ?\Gamma, \Delta, x : !^{\mathsf{o}} A, y : ?^{\mathsf{o}} A^{\perp}}\;\textsc{Mix}}{(\boldsymbol{\nu} x^{!^{\mathsf{o}} A} y)\big(!x(v).P \mid ?y[w].Q\big) \vdash ?\Gamma, \Delta}\;\textsc{Cycle} \qquad \longrightarrow \qquad \dfrac{\dfrac{P \vdash ?\Gamma, v : A \quad Q \vdash \Delta, w : A^{\perp}}{P \mid Q \vdash ?\Gamma, \Delta, v : A, w : A^{\perp}}\;\textsc{Mix}}{(\boldsymbol{\nu} v^{A} w)\big(P \mid Q\big) \vdash ?\Gamma, \Delta}\;\textsc{Cycle}$$

$\beta_{!\mathrm{W}}$

$$\dfrac{\dfrac{\dfrac{P \vdash ?\Gamma, v : A \quad \mathsf{o} < \mathsf{pr}(?\Gamma)}{!x(v).P \vdash ?\Gamma, x : !^{\mathsf{o}} A}\;! \qquad \dfrac{Q \vdash \Delta}{Q \vdash \Delta, y : ?^{\mathsf{o}} A^{\perp}}\;\mathrm{W}}{!x(v).P \mid Q \vdash ?\Gamma, \Delta, x : !^{\mathsf{o}} A, y : ?^{\mathsf{o}} A^{\perp}}\;\textsc{Mix}}{(\boldsymbol{\nu} x^{!^{\mathsf{o}} A} y)\big(!x(v).P \mid Q\big) \vdash ?\Gamma, \Delta}\;\textsc{Cycle} \qquad \longrightarrow \qquad \dfrac{Q \vdash \Delta}{Q \vdash ?\Gamma, \Delta}\;\mathrm{W}$$

$\beta_{!\mathrm{C}}$

$$\dfrac{\dfrac{\dfrac{P \vdash ?\Gamma, v : A \quad \mathsf{o} < \mathsf{pr}(?\Gamma)}{!x(v).P \vdash ?\Gamma, x : !^{\mathsf{o}} A}\;! \qquad \dfrac{Q \vdash \Delta, y' : ?^{\mathsf{o}} A^{\perp}, y'' : ?^{\mathsf{o}} A^{\perp}}{Q\{y/y', y/y''\} \vdash \Delta, y : ?^{\mathsf{o}} A^{\perp}}\;\mathrm{C}}{!x(v).P \mid Q\{y/y', y/y''\} \vdash ?\Gamma, \Delta, x : !^{\mathsf{o}} A, y : ?^{\mathsf{o}} A^{\perp}}\;\textsc{Mix}}{(\boldsymbol{\nu} x^{!^{\mathsf{o}} A} y)\big(!x(v).P \mid Q\{y/y', y/y''\}\big) \vdash ?\Gamma, \Delta}\;\textsc{Cycle} \qquad \longrightarrow$$

$$\dfrac{\dfrac{\dfrac{P' \vdash ?\Gamma', v' : A \quad \mathsf{o} < \mathsf{pr}(?\Gamma')}{!x'(v').P' \vdash ?\Gamma', x' : !^{\mathsf{o}} A}\;! \qquad \dfrac{\dfrac{P'' \vdash ?\Gamma'', v'' : A \quad \mathsf{o} < \mathsf{pr}(?\Gamma'')}{!x''(v'').P'' \vdash ?\Gamma'', x'' : !^{\mathsf{o}} A}\;! \qquad Q \vdash \Delta, y' : ?^{\mathsf{o}} A^{\perp}, y'' : ?^{\mathsf{o}} A^{\perp}}{(\boldsymbol{\nu} x''^{\,!^{\mathsf{o}} A} y'')(!x''(v'').P'' \mid Q) \vdash ?\Gamma'', \Delta, y' : ?^{\mathsf{o}} A^{\perp}}\;\textsc{Mix} + \textsc{Cycle}}{(\boldsymbol{\nu} x'^{\,!^{\mathsf{o}} A} y')(!x'(v').P' \mid (\boldsymbol{\nu} x''^{\,!^{\mathsf{o}} A} y'')(!x''(v'').P'' \mid Q)) \vdash ?\Gamma', ?\Gamma'', \Delta}\;\textsc{Mix} + \textsc{Cycle}}{(\boldsymbol{\nu} x'^{\,!^{\mathsf{o}} A} y')(!x'(v').P' \mid (\boldsymbol{\nu} x''^{\,!^{\mathsf{o}} A} y'')(!x''(v'').P'' \mid Q)) \vdash ?\Gamma, \Delta}\;\mathrm{C}$$

Fig. 6. β reductions: exponentials.

C Commuting Conversions

$\kappa_{\bot}$

$$\dfrac{\dfrac{\dfrac{\mathsf{o} < \mathsf{pr}(\Gamma, \widetilde{x}:\widetilde{A}) \qquad P \vdash \Gamma, \widetilde{x} : \widetilde{A}}{x().P \vdash \Gamma, \widetilde{x}:\widetilde{A}, x:\bot^{\mathsf{o}}}\ \bot \qquad Q \vdash \Delta, \widetilde{y}:\widetilde{A}^{\bot}}{x().P \mid Q \vdash \Gamma, \Delta, \widetilde{x}:\widetilde{A}, \widetilde{y}:\widetilde{A}^{\bot}, x:\bot^{\mathsf{o}}}\ \textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(x().P \mid Q\big) \vdash \Gamma, \Delta, x:\bot^{\mathsf{o}}}\ \textsc{Cycle} \quad \longrightarrow \quad \dfrac{\dfrac{\dfrac{\dfrac{P \vdash \Gamma, \widetilde{x}:\widetilde{A} \quad Q \vdash \Delta, \widetilde{y}:\widetilde{A}^{\bot}}{P \mid Q \vdash \Gamma, \Delta, \widetilde{x}:\widetilde{A}, \widetilde{y}:\widetilde{A}^{\bot}, v:A, x:B}\ \textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big) \vdash \Gamma, \Delta}\ \textsc{Cycle}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big) \vdash {\uparrow^{\mathsf{o}+1}}\,\Gamma, {\uparrow^{\mathsf{o}+1}}\,\Delta}\ {\uparrow^{\mathsf{o}+1}} \qquad \mathsf{o} < \mathsf{pr}({\uparrow^{\mathsf{o}+1}}\,\Gamma),\ \mathsf{o} < \mathsf{pr}({\uparrow^{\mathsf{o}+1}}\,\Delta)}{x().[(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big)] \vdash {\uparrow^{\mathsf{o}+1}}\,\Gamma, {\uparrow^{\mathsf{o}+1}}\,\Delta, x:\bot^{\mathsf{o}}}\ \bot$$

$\kappa_{\otimes}$

$$\dfrac{\dfrac{\dfrac{\mathsf{o} < \mathsf{pr}(\Gamma, \widetilde{x}:\widetilde{A}) \qquad P \vdash \Gamma, \widetilde{x}:\widetilde{A}, v:A, x:B}{x[v].P \vdash \Gamma, \widetilde{x}:\widetilde{A}, x:A \otimes^{\mathsf{o}} B}\ \otimes \qquad Q \vdash \Delta, \widetilde{y}:\widetilde{A}^{\bot}}{x[v].P \mid Q \vdash \Gamma, \Delta, \widetilde{x}:\widetilde{A}, \widetilde{y}:\widetilde{A}^{\bot}, x:A \otimes^{\mathsf{o}} B}\ \textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(x[v].P \mid Q\big) \vdash \Gamma, \Delta, x:A \otimes^{\mathsf{o}} B}\ \textsc{Cycle} \quad \longrightarrow \quad \dfrac{\dfrac{\dfrac{\dfrac{P \vdash \Gamma, \widetilde{x}:\widetilde{A}, v:A, x:B \quad Q \vdash \Delta, \widetilde{y}:\widetilde{A}^{\bot}}{P \mid Q \vdash \Gamma, \Delta, \widetilde{x}:\widetilde{A}, \widetilde{y}:\widetilde{A}^{\bot}, v:A, x:B}\ \textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big) \vdash \Gamma, \Delta, v:A, x:B}\ \textsc{Cycle}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big) \vdash {\uparrow^{\mathsf{o}+1}}\,\Gamma, {\uparrow^{\mathsf{o}+1}}\,\Delta, v:{\uparrow^{\mathsf{o}+1}}\,A, x:{\uparrow^{\mathsf{o}+1}}\,B}\ {\uparrow^{\mathsf{o}+1}} \qquad \mathsf{o} < \mathsf{pr}({\uparrow^{\mathsf{o}+1}}\Gamma),\ \mathsf{o} < \mathsf{pr}({\uparrow^{\mathsf{o}+1}}\Delta)}{x[v].\big[(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big)\big] \vdash {\uparrow^{\mathsf{o}+1}}\,\Gamma, {\uparrow^{\mathsf{o}+1}}\,\Delta, x:{\uparrow^{\mathsf{o}+1}}\,A \otimes^{\mathsf{o}} {\uparrow^{\mathsf{o}+1}}\,B}\ \otimes$$

$\kappa_{⅋}$

$$\dfrac{\dfrac{\dfrac{\mathsf{o} < \mathsf{pr}(\Gamma, \widetilde{x}:\widetilde{A}) \qquad P \vdash \Gamma, \widetilde{x}:\widetilde{A}, w:A, x:B}{x(w).P \vdash \Gamma, \widetilde{x}:\widetilde{A}, x:A ⅋^{\mathsf{o}} B}\ ⅋ \qquad Q \vdash \Delta, \widetilde{y}:\widetilde{A}^{\bot}}{x(w).P \mid Q \vdash \Gamma, \Delta, \widetilde{x}:\widetilde{A}, \widetilde{y}:\widetilde{A}^{\bot}, x:A ⅋^{\mathsf{o}} B}\ \textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(x(w).P \mid Q\big) \vdash \Gamma, \Delta, x:A ⅋^{\mathsf{o}} B}\ \textsc{Cycle} \quad \longrightarrow \quad \dfrac{\dfrac{\dfrac{\dfrac{P \vdash \Gamma, \widetilde{x}:\widetilde{A}, w:A, x:B \quad Q \vdash \Delta, \widetilde{y}:\widetilde{A}^{\bot}}{P \mid Q \vdash \Gamma, \Delta, \widetilde{x}:\widetilde{A}, \widetilde{y}:\widetilde{A}^{\bot}, w:A, x:B}\ \textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big) \vdash \Gamma, \Delta, w:A, x:B}\ \textsc{Cycle}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big) \vdash {\uparrow^{\mathsf{o}+1}}\,\Gamma, {\uparrow^{\mathsf{o}+1}}\,\Delta, w:{\uparrow^{\mathsf{o}+1}}\,A, x:{\uparrow^{\mathsf{o}+1}}\,B}\ {\uparrow^{\mathsf{o}+1}} \qquad \mathsf{o} < \mathsf{pr}({\uparrow^{\mathsf{o}+1}}\Gamma),\ \mathsf{o} < \mathsf{pr}({\uparrow^{\mathsf{o}+1}}\Delta)}{x(w).\big[(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big)\big] \vdash {\uparrow^{\mathsf{o}+1}}\,\Gamma, {\uparrow^{\mathsf{o}+1}}\,\Delta, x:{\uparrow^{\mathsf{o}+1}}\,A ⅋^{\mathsf{o}} {\uparrow^{\mathsf{o}+1}}\,B}\ ⅋$$

Fig. 7. Commuting conversions: units and multiplicatives.

$\kappa_{\oplus}$

$$\dfrac{\dfrac{\dfrac{\mathsf{o} < \mathsf{pr}(\Gamma, \widetilde{x}{:}\widetilde{A}) \quad P \vdash \Gamma, \widetilde{x}{:}\widetilde{A}, x{:}B_j \qquad j \in I}{x \triangleleft l_j.P \vdash \Gamma, \widetilde{x}{:}\widetilde{A}, x{:}\oplus^{\mathsf{o}}\{l_i : B_i\}_{i\in I}}\;\oplus \qquad Q \vdash \Delta, \widetilde{y}{:}\widetilde{A}^{\perp}}{x \triangleleft l_j.P \mid Q \vdash \Gamma, \Delta, \widetilde{x}{:}\widetilde{A}, \widetilde{y}{:}\widetilde{A}^{\perp}, x{:}\oplus^{\mathsf{o}}\{l_i : B_i\}_{i\in I}}\;\textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})(x \triangleleft l_j.P \mid Q) \vdash \Gamma, \Delta, x{:}\oplus^{\mathsf{o}}\{l_i : B_i\}_{i\in I}}\;\textsc{Cycle}$$

$$\longrightarrow \quad \dfrac{\dfrac{\dfrac{\dfrac{P \vdash \Gamma, \widetilde{x}{:}\widetilde{A}, x{:}B_j \quad Q \vdash \Delta, \widetilde{y}{:}\widetilde{A}^{\perp} \qquad j \in I}{P \mid Q \vdash \Gamma, \Delta, \widetilde{x}{:}\widetilde{A}, \widetilde{y}{:}\widetilde{A}^{\perp}, x{:}B_j}\;\textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big) \vdash \Gamma, \Delta, x{:}B_j}\;\textsc{Cycle}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big) \vdash \uparrow^{\mathsf{o}+1}\Gamma, \uparrow^{\mathsf{o}+1}\Delta, x{:}\uparrow^{\mathsf{o}+1}B_j}\;\uparrow^{\mathsf{o}+1} \quad \mathsf{o} < \mathsf{pr}(\uparrow^{\mathsf{o}+1}\Gamma),\; \mathsf{o} < \mathsf{pr}(\uparrow^{\mathsf{o}+1}\Delta)}{x \triangleleft l_j.\big[(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big)\big] \vdash \uparrow^{\mathsf{o}+1}\Gamma, \uparrow^{\mathsf{o}+1}\Delta, x{:}\oplus^{\mathsf{o}}\{l_i : \uparrow^{\mathsf{o}+1}B_i\}_{i\in I}}\;\oplus$$

$\kappa_{\&}$

$$\dfrac{\dfrac{\dfrac{\mathsf{o} < \mathsf{pr}(\Gamma, \widetilde{x}{:}\widetilde{A}) \quad \forall i \in I.(P_i \vdash \Gamma, \widetilde{x}{:}\widetilde{A}, x : B_i)}{x \triangleright \{l_i : P_i\}_{i\in I} \vdash \Gamma, \widetilde{x}{:}\widetilde{A}, x{:}\&^{\mathsf{o}}\{l_i : B_i\}_{i\in I}}\;\& \qquad Q \vdash \Delta, \widetilde{y}{:}\widetilde{A}^{\perp}}{x \triangleright \{l_i : P_i\}_{i\in I} \mid Q \vdash \Gamma, \Delta, \widetilde{x}{:}\widetilde{A}, \widetilde{y}{:}\widetilde{A}^{\perp}, x{:}\&^{\mathsf{o}}\{l_i : B_i\}_{i\in I}}\;\textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})(x \triangleright \{l_i : P_i\}_{i\in I} \mid Q) \vdash \Gamma, \Delta, x{:}\&^{\mathsf{o}}\{l_i : B_i\}_{i\in I}}\;\textsc{Cycle}$$

$$\longrightarrow \quad \dfrac{\dfrac{\dfrac{\dfrac{P_1 \vdash \Gamma, \widetilde{x}{:}\widetilde{A}, x : B_1 \quad Q \vdash \Delta, \widetilde{y}{:}\widetilde{A}^{\perp}}{P_1 \mid Q \vdash \Gamma, \Delta, \widetilde{x}{:}\widetilde{A}, \widetilde{y}{:}\widetilde{A}^{\perp}, x{:}B_1}\;\textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P_1 \mid Q\big) \vdash \Gamma, \Delta, x{:}B_1}\;\textsc{Cycle} \quad \cdots \quad \dfrac{\dfrac{P_h \vdash \Gamma, \widetilde{x}{:}\widetilde{A}, x : B_h \quad Q \vdash \Delta, \widetilde{y}{:}\widetilde{A}^{\perp}}{P_h \mid Q \vdash \Gamma, \Delta, \widetilde{x}{:}\widetilde{A}, \widetilde{y}{:}\widetilde{A}^{\perp}, x{:}B_h}\;\textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P_h \mid Q\big) \vdash \Gamma, \Delta, x{:}B_h}\;\textsc{Cycle}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P_1 \mid Q\big) \vdash \uparrow^{\mathsf{o}+1}\Gamma, \uparrow^{\mathsf{o}+1}\Delta, x{:}\uparrow^{\mathsf{o}+1}B_1 \quad \cdots \quad (\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P_h \mid Q\big) \vdash \uparrow^{\mathsf{o}+1}\Gamma, \uparrow^{\mathsf{o}+1}\Delta, x{:}\uparrow^{\mathsf{o}+1}B_h}\;\uparrow^{\mathsf{o}+1} \quad \mathsf{o} < \mathsf{pr}(\uparrow^{\mathsf{o}+1}\Gamma),\; \mathsf{o} < \mathsf{pr}(\uparrow^{\mathsf{o}+1}\Delta)}{x \triangleright \{l_i : (\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P_i \mid Q\big)\}_{i\in I} \vdash \uparrow^{\mathsf{o}+1}\Gamma, \uparrow^{\mathsf{o}+1}\Delta, x{:}\&^{\mathsf{o}}\{l_i : \uparrow^{\mathsf{o}+1}B_i\}_{i\in I}}\;\&$$

Fig. 8. Commuting conversions: additives.

$\kappa_?$

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\mathsf{o} < \mathsf{pr}(\Gamma, \widetilde{x}\colon\widetilde{A}) \\ P \vdash \Gamma, \widetilde{x}\colon\widetilde{A}, w\colon A\end{array}}{?x[w].P \vdash \Gamma, \widetilde{x}\colon\widetilde{A}, x\colon ?^{\mathsf{o}}\, A}\;? \qquad Q \vdash \Delta, \widetilde{y}\colon\widetilde{A}^{\perp}}{?x[w].P \mid Q \vdash \Gamma, \Delta, \widetilde{x}\colon\widetilde{A}, \widetilde{y}\colon\widetilde{A}^{\perp}, x\colon ?^{\mathsf{o}}\, A}\;\textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(?x[w].P \mid Q\big) \vdash \Gamma, \Delta, x\colon ?^{\mathsf{o}}\, A}\;\textsc{Cycle}$$

$$\longrightarrow\quad \dfrac{\begin{array}{c}\dfrac{\dfrac{\dfrac{P \vdash \Gamma, \widetilde{x}\colon\widetilde{A}, w\colon A \quad Q \vdash \Delta, \widetilde{y}\colon\widetilde{A}^{\perp}}{P \mid Q \vdash \Gamma, \Delta, \widetilde{x}\colon\widetilde{A}, \widetilde{y}\colon\widetilde{A}^{\perp}, w\colon A}\;\textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big) \vdash \Gamma, \Delta, w\colon A}\;\textsc{Cycle}}{(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big) \vdash \uparrow^{\mathsf{o}+1}\Gamma, \uparrow^{\mathsf{o}+1}\Delta, w\colon\uparrow^{\mathsf{o}+1} A}\;\uparrow^{\mathsf{o}+1} \\ \mathsf{o} < \mathsf{pr}(\uparrow^{\mathsf{o}+1}\Gamma),\ \mathsf{o} < \mathsf{pr}(\uparrow^{\mathsf{o}+1}\Delta)\end{array}}{?x[w].\big[(\boldsymbol{\nu}\widetilde{x}^{\widetilde{A}}\widetilde{y})\big(P \mid Q\big)\big] \vdash \uparrow^{\mathsf{o}+1}\Gamma, \uparrow^{\mathsf{o}+1}\Delta, x\colon ?^{\mathsf{o}}\,(\uparrow^{\mathsf{o}+1} A)}\;?$$

$\kappa_!$

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\mathsf{o} < \mathsf{pr}(\Gamma, \widetilde{x}\colon\widetilde{?^{\mathsf{o}}\, A}) \\ P \vdash ?\Gamma, \widetilde{x}\colon\widetilde{?^{\mathsf{o}}\, A}, v\colon A\end{array}}{!x(v).P \vdash ?\Gamma, \widetilde{x}\colon\widetilde{?^{\mathsf{o}}\, A}, x\colon !^{\mathsf{o}}\, A}\;! \qquad Q \vdash \Delta, \widetilde{y}\colon\widetilde{!^{\mathsf{o}}\, A^{\perp}}}{!x(v).P \mid Q \vdash ?\Gamma, \Delta, \widetilde{x}\colon\widetilde{?^{\mathsf{o}}\, A}, \widetilde{y}\colon\widetilde{!^{\mathsf{o}}\, A^{\perp}} x\colon !^{\mathsf{o}}\, A}\;\textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\,\widetilde{?^{\mathsf{o}} A}}\widetilde{y})\big(!x(v).P \mid Q\big) \vdash ?\Gamma, \Delta, x\colon !^{\mathsf{o}}\, A}\;\textsc{Cycle}$$

$$\longrightarrow\quad \dfrac{\begin{array}{c}\dfrac{\dfrac{\dfrac{P \vdash ?\Gamma, \widetilde{x}\colon\widetilde{?^{\mathsf{o}}\, A}, v\colon A \quad Q \vdash \Delta, \widetilde{y}\colon\widetilde{!^{\mathsf{o}}\, A^{\perp}}}{P \mid Q \vdash ?\Gamma, \Delta, \widetilde{x}\colon\widetilde{?^{\mathsf{o}}\, A}, \widetilde{y}\colon\widetilde{!^{\mathsf{o}}\, A^{\perp}}, v\colon A}\;\textsc{Mix}}{(\boldsymbol{\nu}\widetilde{x}^{\,\widetilde{?^{\mathsf{o}} A}}\widetilde{y})\big(P \mid Q\big) \vdash ?\Gamma, \Delta, v\colon A}\;\textsc{Cycle}}{(\boldsymbol{\nu}\widetilde{x}^{\,\widetilde{?^{\mathsf{o}} A}}\widetilde{y})\big(P \mid Q\big) \vdash \uparrow^{\mathsf{o}+1} ?\Gamma, \uparrow^{\mathsf{o}+1}\Delta, v\colon\uparrow^{\mathsf{o}+1} A}\;\uparrow^{\mathsf{o}+1} \\ \mathsf{o} < \mathsf{pr}(\uparrow^{\mathsf{o}+1} ?\Gamma),\ \mathsf{o} < \mathsf{pr}(\uparrow^{\mathsf{o}+1}\Delta)\end{array}}{!x(v).\big[(\boldsymbol{\nu}\widetilde{x}^{\,\widetilde{?^{\mathsf{o}} A}}\widetilde{y})\big(P \mid Q\big)\big] \vdash \uparrow^{\mathsf{o}+1} ?\Gamma, \uparrow^{\mathsf{o}+1}\Delta, x\colon !^{\mathsf{o}}\,(\uparrow^{\mathsf{o}+1} A)}\;!$$

Fig. 9. Commuting conversions: exponentials.

D Omitted Proofs

We start with some auxiliary definitions and results.

Definition 6. *The* degree *of a proposition A, denoted by $\partial(A)$, is defined by*

- $\partial(\mathbf{1}^\circ) = \partial(\bot^\circ) = 1$
- $\partial(A \otimes^\circ B) = \partial(A \parr^\circ B) = \partial(A) + \partial(B) + 1$
- $\partial(\&^\circ\{l_i : A_i\}_{i\in I}) = \partial(\oplus^\circ\{l_i : A_i\}_{i\in I}) = \sum_{i\in I}\{\partial(A_i)\} + 1$
- $\partial(\,?^\circ A) = \partial(\,!^\circ A) = \partial(A) + 1$.

Definition 7. *A* Maxicut *is a maximal sequence of* Mix *and* Cycle *rules, ending with a* Cycle *rule.*

Because of maximality, the rules applied immediately before a Maxicut are any of the rules given in Fig. 2, except Mix and Cycle. The order in which the Mix and Cycle rules are applied within the Maxicut is irrelevant, however the following proposition holds and can be applied to simplify the structure of a Maxicut.

Proposition 1 (Canonical Maxicut). *Given an arbitrary* Maxicut*, it is always possible to obtain from it a* canonical Maxicut *consisting of a sequence of only* Mix *rules followed by a sequence of only* Cycle *rules.*

Proof. Immediately from structural equivalence given in § 3.

Proposition 2. *For all propositions A, $\partial(A) = \partial(A^\perp)$.*

Proof. Directly from Def. 6. □

Definition 8. *A* single Mix Maxicut *consists of one* Mix *rule followed by a sequence of* Cycle *rules.*

The degree of a Maxicut p*, $d(p)$, is the* sum *of the degrees of the* Cycle*s present in it.*

The degree of a proof π*, $d(\pi)$, is the* sup *of the degrees of its* Maxicut*s, meaning that $d(\pi) = 0$ if and only if proof π has no* Cycle*s.*

$A_1, \ldots, A_n, A$ are Maxicut propositions *if they are eliminated by a* Maxicut.

The height of a proof π*, $h(\pi)$, is the height of its tree, and it is defined as $h(\pi) = \sup\big(h(\pi_i)\big)_{i\in I} + 1$, where $\{\pi_i\}_{i\in I}$ are the subproofs of π.*

Proposition 3. *Let τ be a proof of $\vdash \Gamma$. For any $t \geqslant 0$ we can construct a proof τ' of $\vdash \uparrow^t \Gamma$ such that $h(\tau) = h(\tau')$ and all* Maxicut*s in τ are left unchanged, implying $d(\tau) = d(\tau')$.*

Proof. Straightforward from Def. 5 and Lem. 1. □

The Cycle-elimination theorem for our priority-based linear logic (PLL), by following the same idea of the Cut-elimination theorem for classical linear logic, eliminates a Cycle either by (i) replacing it with another Cycle on simpler propositions, or (ii) pushing it further up the proof tree. The first situation generally corresponds to (only considering the logical part) of our β-reduction rules given in § B. Usually, in the literature they are referred to as key-cases [10,25]. The second situation corresponds to (only considering the logical part) of our commuting conversion rules given in § C.

However, not all the key-cases β-reductions are the same. Let's now consider the reductions involving !, given in § B, Fig. 6. Rules $\beta_{!?}$ and $\beta_{!\mathrm{W}}$ reduce the original proofs to proofs where the propositions being Cycled are indeed simpler, as mentioned in (i). However, rule $\beta_{!\mathrm{C}}$ reduces the original proof to a proof where the proposition remains unchanged and the Cycle is not pushed upward in the tree. This reduction does not respect either (i) or (ii) stated above. This case is called a pseudo key-case [10].

In order to take care of pseudo key-cases, we present the following lemma inspired by Lemma B.1.3 in Bräuner [10] and adapted to our priority-based linear logic. This lemma allows us to reduce the degree of a proof ending with a single Mix Maxicut having the same degree as the proof and where the last rule on the left hand-side immediate subproof is !.

Notation 1 *We will use $[n]$ to denote the set of $\{1, \dots, n\}$ for $n > 0$.*

Lemma 2 (Inspired by Lemma B.1.3 in Bräuner [10]). *Let $(A)^n$ denote a list of n occurrences of A. Let τ be a proof of the following form, ending with a single* Mix Maxicut*:*

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi\\ \vdots\\ \mathsf{o} < \mathsf{pr}(?\Gamma)\\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i\\ \vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{array}}{\vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A}\ ! \qquad \dfrac{\begin{array}{c}\pi'\\ \vdots\\ \mathsf{o} < \mathsf{pr}(\Delta)\\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i\\ \forall j \in [k] : \mathsf{o} \leqslant \kappa_j\\ \vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (?^{\kappa_j} A^{\perp})^k\end{array}}{\vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \mathrm{C}^{k-1}}{\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \textsc{Mix}}{\vdash ?\Gamma, \Delta}\ \textsc{Cycle}$$

where $d(\pi) < d(\tau)$ and $d(\pi') < d(\tau)$. Then there is a proof τ' of $\vdash ?\Gamma, \Delta$ such that $d(\tau') < d(\tau)$.

Proof. We start with calculating $d(\tau) = \partial(\, ?^{\mathsf{o}_1} A_1) + \ldots + \partial(\, ?^{\mathsf{o}_n} A_n) + \partial(\, !^{\mathsf{o}} A)$.

The proof is by induction on $h(\pi')$. Let r' be the last rule applied in π'. We proceed by cases for r'.

1. Case $r' = \mathrm{Ax}$. Then, $k = 1$, and $\Delta = \ !^{\mathsf{o}} A$ and $n = 0$. We have that π' is the following proof:

$$\dfrac{}{\vdash\ !^{\mathsf{o}} A,\ ?^{\mathsf{o}} A^{\perp}}\ \mathrm{Ax}$$

Then, τ' is the following proof:

$$\dfrac{\vdash ?\Gamma, A \quad \mathsf{o} < \mathsf{pr}(?\Gamma)}{\vdash ?\Gamma,\ ?^{\mathsf{o}} A}\ !$$

and trivially $d(\tau') = d(\pi) < d(\tau)$.

2. Case $r' = ?$. One of the k occurrences of $?^{\kappa_j} A^{\perp}$ in the conclusion of π' is introduced by $?$. We distinguish two cases.
   – $k = 1$. This case corresponds to (only considering the logical part) of $\beta_{!?}$ reduction rule in § B, Fig. 6. Then, τ' is the following proof:

$$\dfrac{\dfrac{\begin{array}{c}\pi\\ \vdots\\ \vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{array} \qquad \begin{array}{c}\pi''\\ \vdots\\ \vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, A^{\perp}\end{array}}{\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, A^{\perp}}\ \textsc{Mix}}{\vdash ?\Gamma, \Delta}\ \textsc{Cycle}$$

   and

$$d(\tau') = \sup(d(\pi), d(\pi''), \partial(\, ?^{\mathsf{o}_1} A_1) + \ldots + \partial(\, ?^{\mathsf{o}_n} A_n) + \partial(A)) < d(\tau)$$

– $k > 1$, meaning that τ is the following proof:

$$
\dfrac{
\dfrac{
\begin{array}{c}\pi\\ \vdots\\ \mathsf{o} < \mathsf{pr}(?\Gamma)\\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i\\ \vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{array}
}{
\vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A
}\ !
\qquad
\dfrac{
\dfrac{
\begin{array}{c}\pi''\\ \vdots\\ \mathsf{o} < \mathsf{pr}(\Delta), \quad \forall i \in [n] : \mathsf{o} < \mathsf{o}_i, \quad \forall j \in [k] : \mathsf{o} \leqslant \kappa_j \quad (1)\\ l \in [k] \quad \forall j \in [k] \setminus l : \kappa_l < \kappa_j \quad \kappa_l < \mathsf{pr}(\Delta) \quad \forall i \in [n] : \kappa_l < \mathsf{o}_i \ (2)\\ \vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (?^{\kappa_j} A^{\perp})^{k-1}, A^{\perp}\end{array}
}{
\vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (?^{\kappa_j} A^{\perp})^{k-1},\ ?^{\kappa_l} A^{\perp}
}\ ?
}{
\vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}
}\ \mathrm{C}^{k-1}
}{
\dfrac{
\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}
}{
\vdash ?\Gamma, \Delta
}\ \textsc{Cycle}
}\ \textsc{Mix}
$$

Notice that in (2) we have conditions on κ_l to be the *smallest* priority compared with any other priority in the remainder of the environment. (2) is required in order to apply ?. On the other hand, (1) has the same conditions for o, except for $\mathsf{o} \leqslant \kappa_j$ for all $j \in [k]$. (1) is part of the hypothesis of the theorem, and it comes from the premise of rule C. Notice that (1) and (2) are not in conflict: they are both satisfiable with a solution of $\mathsf{o} = \kappa_l$.
This case of the proof shows us why (1) is necessary. Due to ? (or even W) we have in the premise of C different priorities (here κ_j for $j \in [k]$). To deal with this situation, we require that the priority of the contracted proposition in the conclusion of C, is $\leqslant$ (here o) than any of the priorities of the propositions being contracted in the premise, which is always satisfiable.
Then, τ' is the following proof:

$$
\dfrac{
\dfrac{
\begin{array}{c}\pi\\ \vdots\\ \vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{array}
\qquad
\dfrac{
\dfrac{
\dfrac{
\begin{array}{c}\pi\\ \vdots\\ \mathsf{o} < \mathsf{pr}(?\Gamma)\\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i\\ \vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{array}
}{
\vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A
}\ !
\qquad
\dfrac{
\begin{array}{c}\pi''\\ \vdots\\ \mathsf{o} < \mathsf{pr}(\Delta) \quad \forall i \in [n] : \mathsf{o} < \mathsf{o}_i \quad \forall j \in [k] : \mathsf{o} \leqslant \kappa_j\\ \vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (?^{\kappa_j} A^{\perp})^{k-1}, A^{\perp}\end{array}
}{
\vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}, A^{\perp}
}\ \mathrm{C}^{k-2}
}{
\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}, A^{\perp}
}\ \textsc{Mix}
}{
\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, A^{\perp}
}\ \textsc{Cycle}
}{
\vdash ?\Gamma, ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, A, A^{\perp}
}\ \textsc{Mix}
}{
\dfrac{
\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, A, A^{\perp}
}{
\vdash ?\Gamma, \Delta
}\ \textsc{Cycle}
}\ \mathrm{C}
$$

We make the following remark. As in the previous case for $k = 1$, where $\beta_{!?}$ was applied, we would have wanted for the case for $k > 1$ to apply $\beta_{!\mathrm{C}}$ in order to obtain τ'. However, this is a pseudo key-case and $\beta_{!\mathrm{C}}$ does not decrease the degree of τ. This lemma deals with the fact that $\beta_{!\mathrm{C}}$ behaves differently from any other β-reduction rule.
Now we calculate the degree of τ'. We have that:

$$d(\tau') = \sup\big(d(\pi),\ d(\pi''),\ \partial(\,!^{\mathsf{o}} A),\ \partial(\,?^{\mathsf{o}_1} A_1) + \ldots + \partial(\,?^{\mathsf{o}_n} A_n) + \partial(A)\big) < d(\tau)$$

3. Case $r' = \mathrm{W}$. One of the k occurrences of $?^{\kappa_j} A^{\perp}$ in the conclusion of π' is introduced by W. We distinguish two cases.

- $k = 1$. This case corresponds to (only considering the logical part) of $\beta_{!\mathrm{W}}$ reduction rule in §B, Fig. 6.
  τ is the following proof:

$$
\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi \\ \vdots \\ \mathsf{o} < \mathsf{pr}(?\Gamma) \\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i \\ \vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{array}}{\vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A}\ ! \quad \dfrac{\begin{array}{c}\pi'' \\ \vdots \\ \vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}\end{array}}{\vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \mathrm{W}}{\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \textsc{Mix}}{\vdash ?\Gamma, \Delta}\ \textsc{Cycle}
$$

  Then, τ' is the following proof:

$$
\dfrac{\dfrac{\begin{array}{c}\pi'' \\ \vdots \\ \vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}\end{array}}{\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}}\ \mathrm{W}}{\vdash ?\Gamma, \Delta}\ \textsc{Cycle}
$$

  Calculating the degree of τ', we have that:

$$
d(\tau') = \sup\big(d(\pi''),\ \partial(\,?^{\mathsf{o}_1} A_1) + \ldots + \partial(\,?^{\mathsf{o}_n} A_n)\big) < d(\tau)
$$

- $k > 1$, meaning τ is the following proof:

$$
\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi \\ \vdots \\ \mathsf{o} < \mathsf{pr}(?\Gamma) \\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i \\ \vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{array}}{\vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A}\ ! \quad \dfrac{\dfrac{\begin{array}{c}\pi'' \\ \vdots \\ \mathsf{o} < \mathsf{pr}(\Delta), \quad \forall i \in [n] : \mathsf{o} < \mathsf{o}_i, \quad \forall j \in [k] : \mathsf{o} \leqslant \kappa_j \\ \vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (\,?^{\kappa_j} A^{\perp})^{k-1}\end{array}}{\vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (\,?^{\kappa_j} A^{\perp})^{k}}\ \mathrm{W}}{\vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \mathrm{C}^{k-1}}{\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A,\ !^{\mathsf{o}_1} {A_1}^{\perp}, ..., !^{\mathsf{o}_n} {A_n}^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \textsc{Mix}}{\vdash ?\Gamma, \Delta}\ \textsc{Cycle}
$$

  Let τ'' be the following proof:

$$
\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi \\ \vdots \\ \mathsf{o} < \mathsf{pr}(?\Gamma) \\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i \\ \vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{array}}{\vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A}\ ! \quad \dfrac{\begin{array}{c}\pi'' \\ \vdots \\ \mathsf{o} < \mathsf{pr}(\Delta), \quad \forall i \in [n] : \mathsf{o} < \mathsf{o}_i, \quad \forall j \in [k] : \mathsf{o} \leqslant \kappa_j \\ \vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (\,?^{\kappa_j} A^{\perp})^{k-1}\end{array}}{\vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \mathrm{C}^{k-2}}{\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \textsc{Mix}}{\vdash ?\Gamma, \Delta}\ \textsc{Cycle}
$$

  and use the induction hypothesis to obtain τ'.

4. Case $r' = \text{C}$. One of the k occurrences of $?^{\kappa_j} A^{\perp}$ in the conclusion of π' is introduced by C. Then τ is the following proof:

$$
\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi\\ \vdots\\ \mathsf{o} < \mathsf{pr}(?\Gamma)\\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i\\ \vdash ?\Gamma,\, ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{array}}{\vdash ?\Gamma,\, ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\, !^{\mathsf{o}} A}\,! \qquad \dfrac{\dfrac{\begin{array}{c}\pi''\\ \vdots\\ \mathsf{o} < \mathsf{pr}(\Delta), \quad \forall i \in [n] : \mathsf{o} < \mathsf{o}_i, \quad \forall j \in [k] : \mathsf{o} \leqslant \kappa_j\\ \vdash \Delta,\, !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (\,?^{\kappa_j} A^{\perp})^{k+1}\end{array}}{\vdash \Delta,\, !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (\,?^{\kappa_j} A^{\perp})^{k}}\,\text{C}}{\vdash \Delta,\, !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\, ?^{\mathsf{o}} A^{\perp}}\,\text{C}^{k-1}}{\vdash ?\Gamma, \Delta,\, ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\, !^{\mathsf{o}} A,\, !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\, ?^{\mathsf{o}} A^{\perp}}\,\textsc{Mix}}{\vdash ?\Gamma, \Delta}\,\textsc{Cycle}
$$

Since C is applied in π', then it means that the remaining propositions in $(\,?^{\kappa_j} A^{\perp})^k$ have priorities *smaller or equal* than the propositions in $(\,?^{\kappa_j} A^{\perp})^{k+1}$. Hence, by transitivity, o remains the smallest priority among κ_i for all $i \in \{1 \ldots k+1\}$. We can view τ as the proof where we have C^k applications on the premise $\vdash \Delta,\, !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (\,?^{\kappa_j} A^{\perp})^{k+1}$. Then, we apply the induction hypothesis as $h(\pi'') < h(\pi')$.

5. None of the k occurrences of $?^{\kappa_j} A^{\perp}$ in the conclusion of π' is introduced by r'. We distinguish two subcases.

   – All the k occurrences of the propositions $?^{\kappa_j} A^{\perp}$ for $j \in \{1, ..., k\}$ in the end-sequent of π' are inherited from the same immediate subproof. Let r' be a one-premise rule. The situation is analogous if r' is Mix, which is the only two premise rule in our system. If r' is introducing a logical connective among the propositions in $\Delta,\, !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}$, then the priority of the connective introduced is the *smallest* among the other ones in the judgement. This also implies that it is *strictly smaller* than κ_j. $\forall j \in [k]$. Note that Θ is the part of the typing context before r' is applied; after r' is applied Θ becomes $\Delta,\, !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}$. We have the following τ proof:

$$
\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi\\ \vdots\\ \mathsf{o} < \mathsf{pr}(?\Gamma)\\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i\\ \vdash ?\Gamma,\, ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{array}}{\vdash ?\Gamma,\, ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\, !^{\mathsf{o}} A}\,! \qquad \dfrac{\begin{array}{c}\dfrac{\begin{array}{c}\pi''\\ \vdots\\ \vdash \Theta, (\,?^{\kappa_j} A^{\perp})^{k}\end{array}}{\vdash \Delta,\, !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (\,?^{\kappa_j} A^{\perp})^{k}}\,r'\\ \mathsf{o} < \mathsf{pr}(\Delta), \quad \forall i \in [n] : \mathsf{o} < \mathsf{o}_i, \quad \forall j \in [k] : \mathsf{o} \leqslant \kappa_j\end{array}}{\vdash \Delta,\, !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\, ?^{\mathsf{o}} A^{\perp}}\,\text{C}^{k-1}}{\vdash ?\Gamma, \Delta,\, ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\, !^{\mathsf{o}} A,\, !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\, ?^{\mathsf{o}} A^{\perp}}\,\textsc{Mix}}{\vdash ?\Gamma, \Delta}\,\textsc{Cycle}
$$

which we transform into the following τ':

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\begin{matrix}\pi \\ \vdots \\ \mathsf{o} < \mathsf{pr}(?\Gamma) \\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i \\ \vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{matrix}}{\vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A}\ ! \qquad \dfrac{\begin{matrix}\pi'' \\ \vdots \\ \forall j \in [k] : \mathsf{o} \leqslant \kappa_j \\ \vdash \Theta, (\ ?^{\kappa_j} A^{\perp})^k\end{matrix}}{\vdash \Theta,\ ?^{\mathsf{o}} A^{\perp}}\ \mathrm{C}^{k-1}}{\vdash ?\Gamma, \Theta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A,\ ?^{\mathsf{o}} A^{\perp}}\ \textsc{Mix}}{\vdash ?\Gamma, \Theta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n}\ \textsc{Cycle}}{\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}}\ r'}{\vdash ?\Gamma, \Delta}\ \textsc{Cycle}
$$

We have that

$$
d(\tau') = \sup\big(d(\pi),\ d(\pi''),\ \partial(\ !^{\mathsf{o}} A),\ \partial(\ ?^{\mathsf{o}_1} A_1) + \ldots + \partial(\ ?^{\mathsf{o}_n} A_n)\big) < d(\tau)
$$

– Not all k occurrences of the propositions $?^{\kappa_j} A^{\perp}$ for $j \in \{1, ..., k\}$ in the end-sequent of π' are inherited from the same subproof. We have the following τ proof, where ($r' = \textsc{Mix}$) and $\Delta_1, \Delta_2 = \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}$ and let $k = p + q$:

$$
\dfrac{\dfrac{\dfrac{\begin{matrix}\pi \\ \vdots \\ \mathsf{o} < \mathsf{pr}(?\Gamma) \\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i \\ \vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{matrix}}{\vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A}\ ! \qquad \dfrac{\dfrac{\begin{matrix}\pi_1 \\ \vdots \\ \vdash \Delta_1, (\ ?^{\kappa_h} A^{\perp})^p\end{matrix} \quad \begin{matrix}\pi_2 \\ \vdots \\ \vdash \Delta_2, (\ ?^{\kappa_l} A^{\perp})^q\end{matrix}}{\vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp}, (\ ?^{\kappa_j} A^{\perp})^k}\ \textsc{Mix} \quad \mathsf{o} < \mathsf{pr}(\Delta), \quad \forall i \in [n] : \mathsf{o} < \mathsf{o}_i, \quad \forall j \in [k] : \mathsf{o} \leqslant \kappa_j}{\vdash \Delta,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \mathrm{C}^{k-1}}{\vdash ?\Gamma, \Delta,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A,\ !^{\mathsf{o}_1} A_1^{\perp}, ..., !^{\mathsf{o}_n} A_n^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \textsc{Mix}}{\vdash ?\Gamma, \Delta}\ \textsc{Cycle}
$$

which we transform into τ' as follows:

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\begin{matrix}\pi \\ \vdots \\ \mathsf{o} < \mathsf{pr}(?\Gamma) \\ \forall i \in [n] : \mathsf{o} < \mathsf{o}_i \\ \vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n, A\end{matrix}}{\vdash ?\Gamma,\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A}\ ! \qquad \dfrac{\begin{matrix}\pi_1 \\ \vdots \\ \vdash \Delta_1, (\ ?^{\kappa_h} A^{\perp})^p \\ \forall h \in \{1, ..., p\} : \mathsf{o} < \kappa_h\end{matrix}}{\vdash \Delta_1,\ ?^{\mathsf{o}} A^{\perp}}\ \mathrm{C}^{p-1}}{\vdash ?\Gamma, \Delta_1\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n,\ !^{\mathsf{o}} A,\ ?^{\mathsf{o}} A^{\perp}}\ \textsc{Mix}}{\vdash ?\Gamma, (\Delta_1 \setminus_{\textsc{Cycle}}\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n),\ !^{\mathsf{o}} A,\ ?^{\mathsf{o}} A^{\perp}}\ \textsc{Cycle} \star \Delta_1 \qquad \dfrac{\begin{matrix}\pi_2 \\ \vdots \\ \vdash \Delta_2, (\ ?^{\kappa_l} A^{\perp})^q \\ \forall l \in \{p+1, ..., k\} : \mathsf{o} < \kappa_l\end{matrix}}{\vdash \Delta_2,\ ?^{\mathsf{o}} A^{\perp}}\ \mathrm{C}^{q-1}}{\vdash ?\Gamma, \Delta_2, (\Delta_1 \setminus_{\textsc{Cycle}}\ ?^{\mathsf{o}_1} A_1, ..., ?^{\mathsf{o}_n} A_n),\ !^{\mathsf{o}} A,\ ?^{\mathsf{o}} A^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \textsc{Mix}}{\vdash ?\Gamma, \Delta,\ !^{\mathsf{o}} A,\ ?^{\mathsf{o}} A^{\perp},\ ?^{\mathsf{o}} A^{\perp}}\ \textsc{Cycle} \star \Delta_2}{\vdash ?\Gamma, \Delta,\ !^{\mathsf{o}} A,\ ?^{\mathsf{o}} A^{\perp}}\ \mathrm{C}}{\vdash ?\Gamma, \Delta}\ \textsc{Cycle}
$$

In τ' we use labels $\textsc{Cycle}\star\Delta_1$ and $\textsc{Cycle}\star\Delta_2$, respectively to denote the *cycle* between propositions $?^{\circ_1} A_1, ..., ?^{\circ_n} A_n$ and Δ_1 resulting in $\Delta_1 \setminus_{\textsc{Cycle}} ?^{\circ_1} A_1, ..., ?^{\circ_n} A_n$, where for simplicity we use $\setminus_{\textsc{Cycle}}$ to denote the judgement *after* Cycle has occurred; and the Cycle between Δ_2 and $\Delta_1 \setminus_{\textsc{Cycle}} ?^{\circ_1} A_1, ..., ?^{\circ_n} A_n$, finally resulting in Δ.
We have that

$$d(\tau') = \sup\big(d(\pi),\ d(\pi_1),\ d(\pi_2),\ \partial(\textsc{Cycle}\star\Delta_1),\ \partial(\textsc{Cycle}\star\Delta_2),\ \partial(\,!^{\circ} A)\big) < d(\tau)$$

where both $\partial(\textsc{Cycle}\star\Delta_1)$ and $\partial(\textsc{Cycle}\star\Delta_2)$ are smaller or at most equal to $\partial(\,?^{\circ_1} A_1)+\ldots+\partial(\,?^{\circ_n} A_n)$, which is strictly smaller than $d(\tau)$. $\square$

Lemma 3 (The Principal Lemma). *Let τ be a proof of $\vdash \Gamma$, ending with a canonical* Maxicut*:*

$$\dfrac{\dfrac{\pi_1 \ldots \pi_m}{\vdash \Gamma, A_1, ..., A_n, A, A_1^{\perp}, ..., A_n^{\perp}, A^{\perp}}\ \textsc{Mix}}{\vdash \Gamma}\ \textsc{Cycle}$$

such that for all $i \in [m]$, $d(\pi_i) < d(\tau)$. Then, there is a proof τ' of $\vdash \uparrow^t \Gamma$, for some $t \geqslant 0$, such that $d(\tau') < d(\tau)$.

Proof. The proof is by induction on $\sum_{i\in[m]} h(\pi_i)$.

Note that $n \geqslant 0$ and $m \geqslant 1$. Let τ end with a canonical Maxicut p.

If $m = 1$, meaning that there is only one premise, then there cannot be a Mix rule after the single premise π_1 and the Maxicut p is simply a sequence of cycles. If $m > 1$, then the Maxicut p contains also a non-empty sequence of Mix rules.

The assumption that the degrees of all π_i are *strictly smaller* than the degree of τ implies that the degree of the final Maxicut p on propositions $A_1, ..., A_n, A$ is bigger than the degree of any other Maxicut in any proof π_i.

By definition, the degree of the proof τ is the sup of the degrees of its Maxicuts, which in turn is the sum of the degrees of the cycles present in it. This implies that $d(\tau) = d(p) = \partial(A_1)+\ldots+\partial(A_n)+\partial(A)$.

Let r_i be the last rule applied in π_i, for all $i \in [m]$. Since we are considering Maxicuts, then by definition, rules r_i, for all $i \in [m]$ are any rule except Mix and Cycle. Let C_{r_i} be the proposition introduced by the rule r_i.

We now consider the set of all propositions $\{C_{r_i}\}_{i\in[m]}$ (on the processes side, these propositions type top-level prefixes or forwarders). Among these propositions we now consider the ones with the *smallest* priority. Notice that there might be more than one proposition satisfying the minimality condition on priority (on the process side, this corresponds to having different processes in parallel which are independent and can potentially communicate). If this is the case, then we just pick any proposition. Let this proposition be C_{r_k}, then π_k is the following proof:

$$\dfrac{\cdots}{\vdash \Gamma', C_{r_k}}\ r_k$$

We now proceed by cases on r_k.

1. Case $r_k = \textsc{Ax}$. Then π_k is

$$\dfrac{}{\vdash C_{r_k}^{\perp}, C_{r_k}}\ \textsc{Ax}$$

   implying that C_{r_k} is one of the Maxicut propositions, because by the definition of Maxicut we know that at least one cycle is applied. Without loss of generality, let $C_{r_k} = A$.

– In the sequence of CYCLEs in the MAXICUT there is a CYCLE on A. Then, by structural equivalence, we can transform the proof τ and apply CYCLE immediately after AX, and obtain the following proof $\pi_{\text{AX}-\text{CYCLE}}$:

$$\cfrac{\cfrac{}{\vdash A^{\perp}, A}\ \text{AX}}{\vdash \emptyset}\ \text{CYCLE}$$

We have $d(\pi_{\text{AX}-\text{CYCLE}}) = \partial(A)$. Then, we can remove the above CYCLE and AX, by replacing $\pi_{\text{AX}-\text{CYCLE}}$ with the following π'_k, which corresponds to (only considering the logical part) of SC-AX-CYCLE rule given in § A, Fig. 4:

$$\cfrac{}{\mathbf{0} \vdash \emptyset}\ \emptyset$$

where $d(\pi'_k) = 0$ because π'_k has no CYCLEs. Then, we construct τ' as follows:

$$\cfrac{\cfrac{\{\pi_1 \dots \pi_m\} \setminus \pi_{\text{AX}-\text{CYCLE}}}{\vdash \Gamma, A_1, ..., A_n, A_1^{\perp}, ..., A_n^{\perp}}\ \text{MIX}}{\vdash \Gamma}\ \text{CYCLE}$$

whose degree is strictly smaller than the degree of τ. Notice that in order to build τ' we have removed π_k, together with one MIX and one CYCLE from the MAXICUT. Moreover, using SC-MIX-NIL rule given in § A, Fig. 4, we have eliminated π'_k from τ'.

– We consider the case where CYCLE is applied on A in π_k and $A^{\perp}$ in one of the proofs in $\{\pi_1 \dots \pi_m\} \setminus \pi_k$. This implies that $m > 1$ and one of the ${A_i}^{\perp}$ is $A^{\perp}$, and there is CYCLE on A from π_k and ${A_i}^{\perp}$ in a different proof than π_k. Then, τ is the following proof:

$$\cfrac{\cfrac{\text{AX}\ \cfrac{}{\vdash A^{\perp}, A} \qquad \{\pi_1 \dots \pi_m\} \setminus \pi_k}{\vdash \Gamma, A_1, ..., A_{i-1}, A, A_{i+1}, ..., A_n, A, A_1^{\perp}, ..., {A_{i-1}}^{\perp}, A^{\perp}, {A_{i+1}}^{\perp}, ..., A_n^{\perp}, A^{\perp}}\ \text{MIX}}{\vdash \Gamma}\ \text{CYCLE}^{n+1}$$

Then, we construct τ' as follows, which corresponds to (only considering the logical part) of β_{AxCycle} given in § B, Fig. 5:

$$\cfrac{\cfrac{\{\pi_1 \dots \pi_m\} \setminus \pi_k}{\vdash \Gamma, A_1, ..., A_{i-1}, A, A_{i+1}, ..., A_n, A_1^{\perp}, ..., {A_{i-1}}^{\perp}, A^{\perp}, {A_{i+1}}^{\perp}, ..., A_n^{\perp}}\ \text{MIX}}{\vdash \Gamma}\ \text{CYCLE}^{n}$$

whose degree is strictly smaller than the degree of τ.

The case where CYCLE is applied on $A^{\perp}$ in π_k, is completely symmetrical.

2. Case $r_k = \otimes$ introducing one of the MAXICUT propositions $A_1, \dots, A_n, A$. Without loss of generality, suppose r_k is applied on proposition A. This implies that $A = E \otimes^{\mathsf{o}} F$ for some propositions E and F and a natural number o. By the condition of the $\otimes$ rule, we have that $\mathsf{o} < \mathsf{pr}(\Gamma')$. Since A is a MAXICUT proposition, by duality we have that $A^{\perp} = E^{\perp} \parr^{\mathsf{o}} F^{\perp}$ (at this point we know that $A^{\perp}$ is a $\parr$). Since $\mathsf{o} < \mathsf{pr}(\Gamma')$ and $\mathsf{pr}(A^{\perp}) = \mathsf{o}$, then it must be the case that $A^{\perp}$ is in another proof, which implies that $m > 1$. Let the proof containing $A^{\perp}$ be π_h for $h \in [m]$ and $h \neq k$ (at this point we know that A and $A^{\perp}$ are in parallel).

Let π_h be the following proof:

$$\dfrac{\dots}{\vdash \Gamma'', E^\perp \parr^{\circ} F^\perp}\, r_h$$

If r_h is the Ax rule, then this falls in the previous case. We now consider the case where r_h is a logical rule, hence $E \otimes^{\circ} F$ is in another sequent.
We will prove that r_h is applied on $A^\perp$, and as a consequence it is a $\parr$ rule. (This is equivalent to proving that $A^\perp$ is a top prefix).
Suppose r_h is applied on $C_{r_h} \neq A^\perp$. For any logical rule to be applied, it is required that $\mathsf{pr}(C_{r_h}) < \mathsf{pr}(\Gamma'', E^\perp \parr^{\circ} F^\perp \setminus C_{r_h})$, which implies that

$$\mathsf{pr}(C_{r_h}) < \mathsf{pr}(E^\perp \parr^{\circ} F^\perp) = \mathsf{pr}(E \otimes^{\circ} F) = \circ$$

But this contradicts the fact that $\mathsf{pr}(E \otimes^{\circ} F)$ is the smallest. Hence, r_h must be a $\parr$ on $A^\perp$ (at this point we know that also $A^\perp$ is on top).
We now construct the following proof τ_A which ends with a single Mix Maxicut p_A applied on *at least* A:

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_\otimes\\ \vdots\\ \vdash \Gamma', E, F\end{array}\quad \circ < \mathsf{pr}(\Gamma', E)}{\vdash \Gamma', E \otimes^{\circ} F}\,\otimes \qquad \dfrac{\begin{array}{c}\pi_\parr\\ \vdots\\ \vdash \Gamma'', E^\perp, F^\perp\end{array}\quad \circ < \mathsf{pr}(\Gamma'')}{\vdash \Gamma'', E^\perp \parr^{\circ} F^\perp}\,\parr}{\vdash \Gamma', \Gamma'', E \otimes^{\circ} F, E^\perp \parr^{\circ} F^\perp}\,\textsc{Mix}}{\vdash \Gamma'''}\,\textsc{Cycle}$$

Suppose that p_A is applied on $s \geqslant 0$ propositions in Γ', Γ'', in addition to A. Then,

$$d(p_A) = \partial(A_1) + \ldots + \partial(A_s) + \partial(A) = \partial(A_1) + \ldots + \partial(A_s) + \partial(E) + \partial(F) + 1$$

We can rewrite τ as follows:

$$\dfrac{\dfrac{\big(\{\pi_1 \ldots \pi_m\} \setminus \{\pi_k, \pi_h\}\big), \tau_A}{\vdash \Gamma, A_1, ..., A_{n-s}, A_1^\perp, ..., A_{n-s}^\perp}\,\textsc{Mix}}{\vdash \Gamma}\,\textsc{Cycle}$$

Then, we can rewrite the degree of τ as

$$d(\tau) = d(p) = \partial(A_1) + \ldots + \partial(A_{n-s}) + d(p_A)$$

We now construct τ'_A as follows, which corresponds to (only considering the logical part) of $\beta_{\otimes\parr}$ given in §B, Fig. 5:

$$\dfrac{\dfrac{\begin{array}{c}\pi_\otimes\\ \vdots\\ \vdash \Gamma', E, F\end{array} \quad \begin{array}{c}\pi_\parr\\ \vdots\\ \vdash \Gamma'', E^\perp, F^\perp\end{array}}{\vdash \Gamma', \Gamma'', E, F, E^\perp, F^\perp}\,\textsc{Mix}}{\vdash \Gamma'''}\,\textsc{Cycle}$$

Let the last $\textsc{Maxicut}$ of τ'_A be p'_A. Then,

$$d(p'_A) = \partial(A_1) + \ldots + \partial(A_s) + \partial(E) + \partial(F) < d(p_A) \leqslant d(\tau)$$

We know that $m > 1$. If $m = 2$, meaning that the only premises of τ are π_k and π_h, then we let τ' be τ'_A and this concludes the case.
Otherwise, if $m > 2$ we construct τ' as follows:

$$\dfrac{\dfrac{\big(\{\pi_1 \ldots \pi_m\} \setminus \{\pi_k, \pi_h\}\big), \tau'_A}{\vdash \Gamma, A_{s+1}, ..., A_n, A_{s+1}^{\perp}, ..., A_n^{\perp}}\ \textsc{Mix}}{\vdash \Gamma}\ \textsc{Cycle}$$

Let p' be the final $\textsc{Maxicut}$ of τ'. We observe that p', which is not a single $\textsc{Mix}$ $\textsc{Maxicut}$, is composed by the final $\textsc{Maxicut}$ p'_A of τ'_A and by the sequence of $\textsc{Mix}$ and $\textsc{Cycle}$ rules on propositions $A_1, ..., A_{n-s}$ at the bottom of τ'. We have that

$$d(\tau') = d(p') = \partial(A_{s+1}) + \ldots + \partial(A_n) + d(p'_A) < d(p) = d(\tau)$$

3. Case $r_k = \otimes$ introducing a proposition different than any propositions $A_1, \ldots, A_n, A$ involved in the $\textsc{Maxicut}$. Then τ is the following proof:

$$\dfrac{\dfrac{\dfrac{\begin{matrix}\pi_{k_{EF}} \\ \vdots \\ \vdash \Gamma', E, F\end{matrix} \quad \mathsf{o} < \mathsf{pr}(\Gamma', E)}{\vdash \Gamma', E \otimes^{\mathsf{o}} F}\ \otimes \qquad \{\pi_1 \ldots \pi_m\} \setminus \pi_k}{\vdash \Gamma'', E \otimes^{\mathsf{o}} F, A_1, ..., A_n, A, A_1^{\perp}, ..., A_n^{\perp}, A^{\perp}}\ \textsc{Mix}}{\vdash \Gamma'', E \otimes^{\mathsf{o}} F}\ \textsc{Cycle}$$

where $\Gamma = \Gamma'', E \otimes^{\mathsf{o}} F$ for some propositions E, F and priority o.
We construct proof τ_2 as follows, where the last $\textsc{Maxicut}$ is p_2:

$$\dfrac{\dfrac{\begin{matrix}\pi_{k_{EF}} \\ \vdots \\ \vdash \Gamma', E, F\end{matrix} \quad \{\pi_1 \ldots \pi_m\} \setminus \pi_k}{\vdash \Gamma'', E, F, A_1, ..., A_n, A, A_1^{\perp}, ..., A_n^{\perp}, A^{\perp}}\ \textsc{Mix}}{\vdash \Gamma'', E, F}\ \textsc{Cycle}$$

We have that $d(p_2) = \partial(A_1) + \ldots + \partial(A_n) + \partial(A) = d(\tau)$. This means that p_2 is the biggest $\textsc{Maxicut}$ in τ_2 and hence $d(\tau_2) = d(p_2) = d(\tau)$.
Also

$$h(\pi_{k_{EF}}) + \sum_{i \in [m] \setminus \{k\}} h(\pi_i) < \sum_{i \in [m]} h(\pi_i)$$

and by assumption $d(\pi_i)$, for all $i \in [m]$ (including $d(\pi_k)$ and $d(\pi_{k_{EF}})$) are strictly smaller than $d(\tau_2)$. In particular, for $m = 1$ the above inequation becomes $h(\pi_{k_{EF}}) < h(\pi_k)$. Then, by induction hypothesis there is a proof τ'_2 of $\vdash \uparrow^t \big(\Gamma'', E, F\big)$, for some $t \geqslant 0$, whose degree is strictly smaller than $d(\tau_2)$.

By applying the auxiliary Prop. 3 on the proof τ_2', where priority is increased by t already, we can obtain a proof τ_2'' for $\vdash \uparrow^{\max(t,\mathsf{o}+1)}\big(\Gamma'', E, F\big)$, such that $h(\tau_2'') = h(\tau_2')$ and $d(\tau_2'') = d(\tau_2') < d(\tau)$. We now obtain τ' as follows:

$$\dfrac{\begin{array}{c}\tau_2''\\ \vdots\\ \vdash \uparrow^{\max(t,\mathsf{o}+1)}\big(\Gamma'', E, F\big) \quad \mathsf{o} < \mathsf{pr}(\uparrow^{\max(t,\mathsf{o}+1)}\Gamma'', E)\end{array}}{\vdash \uparrow^{\max(t,\mathsf{o}+1)}\Gamma'', \uparrow^{\max(t,\mathsf{o}+1)} E \otimes^{\mathsf{o}} \uparrow^{\max(t,\mathsf{o}+1)} F}\ \otimes$$

This corresponds to applying the logical part of $\kappa\kappa_{\otimes}$ given in §C, Fig. 7 to the induction hypothesis of this case.

Then, we have
$$d(\tau') = d(\tau_2'') < d(\tau)$$
and this concludes the case.

4. Case $r_k = !$ introducing one of the Maxicut propositions $A_1, \ldots, A_n, A$. Without loss of generality, suppose r_k introduces proposition A, implying that $A = !^{\mathsf{o}} A'$ for some A' and natural number o. Then, π_k is the following proof:

$$\dfrac{\begin{array}{c}\pi_!\\ \vdots\\ \vdash ?\Theta, A' \quad \mathsf{o} < \mathsf{pr}(?\Theta)\end{array}}{\vdash ?\Theta,\ !^{\mathsf{o}} A'}\ !$$

where $\Gamma' = ?\Theta$.

Since $A = !^{\mathsf{o}} A'$ is a Maxicut proposition, by duality we have that $A^{\perp} = ?^{\mathsf{o}} A'^{\perp}$ (at this point we know that $A^{\perp}$ is a ?). Since $\mathsf{o} < \mathsf{pr}(\Gamma')$ and $\mathsf{pr}(A^{\perp}) = \mathsf{o}$, then it must be the case that $A^{\perp}$ is in another proof (which implies that $m > 1$). Let the proof containing $A^{\perp}$ be π_h for $h \in [m]$ and $h \neq k$ (at this point we know that A and $A^{\perp}$ are in parallel). Then, τ is the following proof:

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_!\\ \vdots\\ \vdash ?\Theta, A' \quad \mathsf{o} < \mathsf{pr}(?\Theta)\end{array}}{\vdash ?\Theta,\ !^{\mathsf{o}} A'}\ ! \qquad \{\pi_1 \ldots \pi_m\} \setminus \pi_k}{\vdash \Gamma'', A_1, ..., A_n,\ !^{\mathsf{o}} A', A_1^{\perp}, ..., A_n^{\perp},\ ?^{\mathsf{o}} A'^{\perp}}\ \textsc{Mix}}{\vdash \Gamma}\ \textsc{Cycle}$$

Then, we apply Lem. 2 on π_k and π_h, obtaining a proof which we use to construct τ', as we did in Item 2 of this lemma.

5. Case $r_k = !$ introducing a proposition different from any of the Maxicut propositions $A_1, \ldots, A_n, A$. Thus, all Maxicut propositions are query propositions, namely $A_1 = ?^{\mathsf{o}_1} A_1', \ldots, A_n = ?^{\mathsf{o}_n} A_n'$, and $A = ?^{\mathsf{o}'} A'$. Then τ is the following proof:

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_!\\ \vdots\\ \vdash ?\Theta, E \quad \mathsf{o} < \mathsf{pr}(?\Theta)\end{array}}{\vdash ?\Theta,\ !^{\mathsf{o}} E}\ ! \qquad \{\pi_1 \ldots \pi_m\} \setminus \pi_k}{\vdash \Gamma'',\ !^{\mathsf{o}} E, A_1, ..., A_n, A, A_1^{\perp}, ..., A_n^{\perp}, A^{\perp}}\ \textsc{Mix}}{\vdash \Gamma'',\ !^{\mathsf{o}} E}\ \textsc{Cycle}$$

where $\Gamma' = ?\Theta$ and $\Gamma = \Gamma'', !^{\circ} E$ for some proposition E and priority $\circ$. We construct proof τ_2 as follows, where the last Maxicut is p_2:

$$\dfrac{\dfrac{\begin{matrix}\pi_! \\ \vdots \\ \vdash \Gamma', E\end{matrix} \qquad \{\pi_1 \dots \pi_m\} \setminus \pi_k}{\vdash \Gamma'', E, A_1, ..., A_n, A, A_1^{\perp}, ..., A_n^{\perp}, A^{\perp}}\ \textsc{Mix}}{\vdash \Gamma'', E}\ \textsc{Cycle}$$

and proceed as in Item 3.

6. Case $r_k = \parr$ introducing one of the Maxicut propositions, then we proceed as in Item 2; otherwise, if $\parr$ introduces a proposition different than any of the Maxicut propositions, we proceed as in Item 3.
7. Case $r_k = \&$ (or $r_k = \oplus$) introducing one of the Maxicut propositions, then we proceed as in Item 2; otherwise, if $\&$ (or $\oplus$) introduces a proposition different than any of the Maxicut propositions, we proceed as in Item 3. ☐

Lemma 4. *Given a proof τ of $\vdash \Gamma$, such that $d(\tau) > 0$, then for some $t \geqslant 0$ there is a proof τ' of $\vdash \uparrow^t \Gamma$ such that $d(\tau') < d(\tau)$.*

Proof. By induction on $h(\tau)$. We have the following cases.

– If τ ends in a Maxicut whose degree is *the same as* the degree of τ:

$$\dfrac{\dfrac{\pi_1 \dots \pi_m}{\vdash \Gamma, A_1, ..., A_n, A, A_1^{\perp}, ..., A_n^{\perp}, A^{\perp}}\ \textsc{Mix}^m}{\vdash \Gamma}\ \textsc{Cycle}^{n+1}$$

then we can apply the induction hypothesis on the subproofs of τ right before the last Mix preceding the sequence of Cycles, and this lets us reduce their degrees to smaller than $d(\tau)$. Then we use Lem. 3. Notice that $\uparrow^t$ is performed twice on Γ: once by using the induction hypothesis and once by applying Lem. 3.

– Otherwise, by using the inductive hypothesis on the immediate subproofs to reduce their degree, we also reduce the degree of the whole proof. ☐

Theorem 1 (Cycle-elimination). *Given any proof of $\vdash \Gamma$, we can construct a cycle-free proof of $\vdash\ \uparrow^t \Gamma$, for some $t \geqslant 0$.*

Proof. Iteration on Lem. 4. ☐