Midterm Exam

15-816 Substructural Logics
Frank Pfenning

October 18, 2016

Name: **Sample Solution** Andrew ID: **fp**

Instructions

- This exam is closed-book, closed-notes.
- You have 80 minutes to complete the exam.
- There are 5 problems.

	Peg Solitaire	Contradiction	Negation	Ordered Programming	Ordered Affine Logic	
	Prob 1	Prob 2	Prob 3	Prob 4	Prob 5	Total
Score	**25**	**25**	**30**	**25**	**45**	**150**
Max	25	25	30	25	45	150

1 Peg Solitaire (25 pts)

Peg solitaire is a board game/puzzle for one player. Here, we are playing the one-dimensional version. The board consists of a linear arrangement of holes some of which are filled with pegs. For example, using o for holes and * for pegs

```
o o o o * * o * o o
```

The only legal move is a *jump* when a peg jumps over another peg right next to it and lands in an empty hole right behind it. The peg that is jumped over is removed. In the above situation there are two possible moves, resulting in one of the following two positions.

```
o o o o o o * * o o   (after leftmost peg jumps right)
o o o * o o o * o o   (after middle peg jumps left)
```

Player wins if there is only a single peg left. Player loses if no move is possible and there is more than one peg on the board. In the above two positions, the first one is winnable with either of the two possible moves, while in the second one player has lost.

We represent a position with the following ordered propositions:

peg	there is a peg filling a hole
hole	there is an empty hole
$	marks the left and right ends of the board

For example, the (winnable) initial position

```
* * * * o *
```

would be represented by

$ peg peg peg peg hole peg $

Task 1 (25 pts). Write an ordered logical specification such that for any board representation Ω, we have

$$\begin{array}{c}\Omega\\ \vdots\\ \mathsf{win}\end{array}$$

if and only if player can win. It does not matter what the final state looks like if the initial position is not winnable, or if the context is not a valid representation of a board.

$$\frac{\mathsf{peg\ peg\ hole}}{\mathsf{hole\ hole\ peg}}\ \mathsf{jr} \qquad \frac{\mathsf{hole\ peg\ peg}}{\mathsf{peg\ hole\ hole}}\ \mathsf{jl}$$

$$\frac{\$\ \mathsf{hole}}{\$}\ \$\mathsf{r} \qquad \frac{\mathsf{hole}\ \$}{\$}\ \$\mathsf{l}$$

$$\frac{\$\ \mathsf{peg}\ \$}{\mathsf{win}}\ \mathsf{w}$$

2 Contradiction (25 pts)

As we already have seen in subsingleton logic, it is possible to have an empty succedent in a sequent, $\Omega \vdash \cdot$. Logically, this represents a form of contradiction that is precise about the ordered use of the antecedents in deriving it. We can internalize this in the logic with the proposition $\bot$ and the following rules:

$$\frac{\Omega \vdash \cdot}{\Omega \vdash \bot}\ \bot R \qquad \frac{}{\bot \vdash \cdot}\ \bot L$$

All the other left rules and cut are then generalized to allow either one or no succedents.

Task 1 (5 pts). Show the cut reduction(s) for $\bot$.

$$\frac{\dfrac{\mathcal{D}}{\Omega \vdash \cdot}\ \bot R \quad \dfrac{}{\bot \vdash \cdot}\ \bot L}{\Omega \vdash \cdot}\ \mathsf{cut} \qquad \Longrightarrow_R \qquad \begin{array}{c}\mathcal{D}\\ \Omega \vdash \cdot\end{array}$$

Task 2 (10 pts). On the computational side of ordered logic, we assign a processes P to the new judgment $A_1 \dots A_n \vdash \cdot$ by writing $(x_1{:}A_1) \dots (x_n{:}A_n) \vdash P$. We think of the process P as *detached* in that it has no client. Assign process expressions to the $\bot R$ and $\bot L$ rules, choosing from the existing syntax for processes (see page 12 if you need to refresh your memory).

$$\frac{\Omega \vdash P}{\Omega \vdash \mathsf{wait}\ x\ ;\ P :: (x : \bot)}\ \bot R \qquad \frac{}{x{:}\bot \vdash \mathsf{close}\ x}\ \bot L$$

Task 3 (10 pts). Show the synchronous computation rule $\bot C$ based on the cut reduction, writing $\mathsf{proc}(P)$ for a process with no clients and, as usual, $\mathsf{proc}(x, P)$ for a process offering along x

$$\frac{\mathsf{proc}(x, \mathsf{wait}\ x\ ;\ P) \quad \mathsf{proc}(\mathsf{close}\ x)}{\mathsf{proc}(P)}\ \bot C$$

3 Negation (30 pts)

Once we have empty succedents as introduced in Problem 2, we can define a form of ordered negation. The proposition $\neg A$ should be true if the assumption A is contradictory.

The following attempt at defining right and left rules fails.

$$\frac{\Omega\ A \vdash \cdot}{\Omega \vdash \neg A}\ \neg R?? \qquad \frac{\Omega \vdash A}{\Omega\ \neg A \vdash \cdot}\ \neg L??$$

Task 1 (5 pts). Show that the identity expansion fails.

Identity expansion fails since the only possible attempt at constructing a cut-free proof fails.

$$\frac{}{\neg A \vdash \neg A}\ \mathsf{id}_{\neg A} \quad \Longrightarrow_E?? \quad \frac{\overset{\text{no rule applicable}}{(\neg A)\ A \vdash \cdot}}{\neg A \vdash \neg A}\ \neg R?? \quad \text{(only rule applicable)}$$

Task 2 (5 pts). Show that cut reduction fails.

Cut reduction fails since the endsequents do not match without a rule of exchange.

$$\frac{\dfrac{\overset{\mathcal{D}}{\Omega'\ A \vdash \cdot}}{\Omega' \vdash \neg A}\ \neg R?? \quad \dfrac{\overset{\mathcal{E}}{\Omega \vdash A}}{\Omega\ \neg A \vdash \cdot}\ \neg L??}{\Omega\ \Omega' \vdash \cdot}\ \mathsf{cut}_{\neg A} \quad \Longrightarrow_R?? \quad \frac{\overset{\mathcal{E}}{\Omega \vdash A} \quad \overset{\mathcal{D}}{\Omega'\ A \vdash \cdot}}{\Omega'\ \Omega \vdash \cdot}\ \mathsf{cut}_A$$

Task 3 (10 pts). Analyze the above failures and fix the rule(s) for negation so that identity expansion and cut reduction hold. You do not need to show explicitly that they are now correct.

To repair the "wrong-sidedness" of the rules, we can use either

$$\frac{A\ \Omega \vdash \cdot}{\Omega \vdash \neg A}\ \neg R \qquad \frac{\Omega \vdash A}{\Omega\ \neg A \vdash \cdot}\ \neg L$$

or

$$\frac{\Omega\ A \vdash \cdot}{\Omega \vdash \neg A}\ \neg R \qquad \frac{\Omega \vdash A}{\neg A\ \Omega \vdash \cdot}\ \neg R$$

which define two different symmetric versions of negation.

Task 4 (10 pts). With the repaired rule(s), prove or refute $A \vdash \neg\neg A$ for a propositional variable A. If it is not true in general, find a more specific proposition B such that $B \vdash \neg\neg B$.

It cannot be proven since the only possible attempt at constructing a cut-free proof fails, no matter which of the two rules we choose.

$$\dfrac{\dfrac{\text{no rule applicable}}{(\neg A)\ A \vdash \cdot}}{A \vdash \neg\neg A}\ \neg R$$

If we think of the two possible rule sets in Task 3 to define two different negations, the outer and inner negations in $\neg\neg A$ would have to be two different ones for $A \vdash \neg\neg A$ to hold in general. On the other hand, for $B = \mathbf{1}$ we have

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\cdot \vdash \mathbf{1}}\ \mathbf{1}R}{\neg\mathbf{1} \vdash \cdot}\ \neg L}{\cdot \vdash \neg\neg\mathbf{1}}\ \neg R}{\mathbf{1} \vdash \neg\neg\mathbf{1}}\ \mathbf{1}L$$

4 Ordered Programming (25 pts)

Recall the definition of lists

$$\mathsf{list}_A = \oplus\{\mathsf{cons} : A \bullet \mathsf{list}_A, \mathsf{nil} : \mathbf{1}\}$$

and the following operations on them

$$\begin{array}{rcl} \cdot & \vdash & \mathit{nil} :: (l{:}\mathsf{list}_A) \\ (x{:}A)\ (k{:}\mathsf{list}_A) & \vdash & \mathit{cons} :: (l{:}\mathsf{list}_A) \\ (l_1{:}\mathsf{list}_A)\ (l_2{:}\mathsf{list}_A) & \vdash & \mathit{append} :: (l{:}\mathsf{list}_A) \end{array}$$

We consider a new type

$$\mathsf{option}_A = \oplus\{\mathsf{some} : A, \mathsf{none} : \mathbf{1}\}$$

Task 1 (5 pts). Give a process definition

$\cdot \vdash \mathit{none} :: (o : \mathsf{option}_A)$

$o \leftarrow \mathit{none} =$

$o \leftarrow \mathit{none} =$
  $o.\mathsf{none}$; $\mathsf{close}\ o$

Task 2 (5 pts). Give a process definition

$x{:}A \vdash \mathit{some} :: (o : \mathsf{option}_A)$

$o \leftarrow \mathit{some} \leftarrow x =$

$o \leftarrow \mathit{some} \leftarrow x =$
  $o.\mathsf{some}$; $o \leftarrow x$

Task 3 (15 pts). Next we write a process that compresses a list of optional elements by keeping only the ones that contain actual elements. For this purpose we use the type $\mathsf{list}_{\mathsf{option}_A}$

$$k{:}\mathsf{list}_{\mathsf{option}_A} \vdash \mathit{compress} :: (l : \mathsf{list}_A)$$

You may freely use *none*, *some*, *nil*, *cons*, and *append* in your definition, but if you use any other auxiliary process definitions, clearly state their their types and implementations.

$l \leftarrow \mathit{compress} \leftarrow k =$

```
l ← compress ← k =
  case k  (nil ⇒ wait k ; l.nil ; close l
         | cons ⇒ o ← recv k ;
                  case o  (some ⇒ l.cons ; send l o ;
                                  l ← compress ← k
                           | none ⇒ wait o ;
                                  l ← compress ← k))
```

5 Ordered Affine Logic (45 pts)

In this problem we explore the affine version of ordered logic where antecedents need not be used, which is specified with a structural rule of weakening. We write A_{F} for affine propositions. All the connectives and rules from ordered logic carry over, and we add:

$$\frac{\Omega_1\ \Omega_2 \vdash C_{\mathsf{F}}}{\Omega_1\ A_{\mathsf{F}}\ \Omega_2 \vdash C_{\mathsf{F}}}\ \mathsf{weaken}$$

Task 1 (10 pts). Prove or refute

$$A_{\mathsf{F}}\ (B_{\mathsf{F}} \bullet A_{\mathsf{F}})\ B_{\mathsf{F}} \vdash A_{\mathsf{F}} \bullet B_{\mathsf{F}}$$

One proof below. There is also a shorter one that weakens $B \bullet A$ immediately and then proves $A\ B \vdash A \bullet B$.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A}\ \mathsf{id}}{A\ B \vdash A}\ \mathsf{weaken} \quad \dfrac{\dfrac{}{B \vdash B}\ \mathsf{id}}{A\ B \vdash B}\ \mathsf{weaken}}{A\ B\ A\ B \vdash A \bullet B}\ \bullet R}{A\ (B \bullet A)\ B \vdash A \bullet B}\ \bullet L$$

Task 2 (10 pts). There are some new cases in the proof of admissibility of cut. Show the case where the principal proposition of the cut is the result of weakening and complete the proof for this case.

$$\overset{\mathcal{D}}{\Omega \Vdash A} \quad \text{arbitrary, and} \quad \mathcal{E} = \dfrac{\overset{\mathcal{E}'}{\Omega_1\ \Omega_2 \Vdash C}}{\Omega_1\ A\ \Omega_2 \Vdash C}\ \mathsf{weaken}$$

We need to show $\Omega_1\ \Omega\ \Omega_2 \Vdash C$. We obtain this from $\mathcal{E}'$ by repeated application of the weakening rule.

Now we combine the ordered and ordered affine logic with two modalities, $\uparrow^{\mathsf{F}}_{\mathsf{O}} A_{\mathsf{O}}$ and $\downarrow^{\mathsf{F}}_{\mathsf{O}} A_{\mathsf{F}}$.

$$\begin{array}{lrcl} \text{Ordered affine} & A_{\mathsf{F}} & ::= & \ldots \mid \uparrow^{\mathsf{F}}_{\mathsf{O}} A_{\mathsf{O}} \\ \text{Ordered} & A_{\mathsf{O}} & ::= & \ldots \mid \downarrow^{\mathsf{F}}_{\mathsf{O}} A_{\mathsf{F}} \end{array}$$

Because relative order matters, we cannot separate ordered and ordered affine propositions into disjoint contexts. Instead, we have a mixed context Ω and write "Ω aff" if all propositions in Ω have the form A_{F}, and "Ω ord" if all propositions in Ω have the form A_{O}.

Task 3 (5 pts). State the independence principle for this combination of logics.

An affine succedent cannot depend on an ordered antecedent.

Task 4 (5 pts). The independence principle allows only some sequents $\Omega \vdash A_m$ as well-formed. State which sequents should be allowed.

$\Omega \vdash A_{\mathsf{O}}$ is always well-formed.
$\Omega \vdash A_{\mathsf{F}}$ requires that Ω aff.

Task 5 (15 pts). Complete the following table of left and right rules for the shift modalities. You should always assume the conclusions are well-formed according to the criteria from Task 4. In some cases you have to add conditions to the premises to make sure they are also well-formed. The additional conditions should have the form Ω aff, Ω ord, $m = \mathsf{F}$, or $m = \mathsf{O}$. You should only add them where necessary.

$$\frac{\Omega \vdash A_{\mathsf{O}}}{\Omega \vdash \uparrow^{\mathsf{F}}_{\mathsf{O}} A_{\mathsf{O}}} \uparrow R \qquad \frac{\Omega_1\ A_{\mathsf{O}}\ \Omega_2 \vdash C_m}{\Omega_1\ (\uparrow^{\mathsf{F}}_{\mathsf{O}} A_{\mathsf{O}})\ \Omega_2 \vdash C_m} \uparrow L$$

$$\frac{\Omega \vdash A_{\mathsf{F}}}{\Omega \vdash \downarrow^{\mathsf{F}}_{\mathsf{O}} A_{\mathsf{F}}} \downarrow R \qquad \frac{\Omega_1\ A_{\mathsf{F}}\ \Omega_2 \vdash C_m}{\Omega_1\ (\downarrow^{\mathsf{F}}_{\mathsf{O}} A_{\mathsf{F}})\ \Omega_2 \vdash C_m} \downarrow L$$

$$\frac{\Omega \vdash A_{\mathsf{O}}}{\Omega \vdash \uparrow^{\mathsf{F}}_{\mathsf{O}} A_{\mathsf{O}}} \uparrow R \qquad \frac{m = \mathsf{O} \quad \Omega_1\ A_{\mathsf{O}}\ \Omega_2 \vdash C_m}{\Omega_1\ (\uparrow^{\mathsf{F}}_{\mathsf{O}} A_{\mathsf{O}})\ \Omega_2 \vdash C_m} \uparrow L$$

$$\frac{\Omega\ \mathsf{aff} \quad \Omega \vdash A_{\mathsf{F}}}{\Omega \vdash \downarrow^{\mathsf{F}}_{\mathsf{O}} A_{\mathsf{F}}} \downarrow R \qquad \frac{\Omega_1\ A_{\mathsf{F}}\ \Omega_2 \vdash C_m}{\Omega_1\ (\downarrow^{\mathsf{F}}_{\mathsf{O}} A_{\mathsf{F}})\ \Omega_2 \vdash C_m} \downarrow L$$

Appendix: Some Inference Rules

$$\begin{array}{rrcl} \text{Propositions} & A, B, C & ::= & p \mid A \oplus B \mid A \,\&\, B \mid \mathbf{1} \\ & & \mid & A \,/\, B \mid B \setminus A \mid A \bullet B \mid A \circ B \end{array}$$

Judgmental rules

$$\frac{}{A \vdash A}\ \mathsf{id}_A \qquad \frac{\Omega \vdash A \quad \Omega_L\ A\ \Omega_R \vdash C}{\Omega_L\ \Omega\ \Omega_R \vdash C}\ \mathsf{cut}_A$$

Propositional rules

$$\frac{A\ \Omega \vdash B}{\Omega \vdash A \setminus B}\ \setminus R \qquad \frac{\Omega' \vdash A \quad \Omega_L\ B\ \Omega_R \vdash C}{\Omega_L\ \Omega'\ (A \setminus B)\ \Omega_R \vdash C}\ \setminus L$$

$$\frac{\Omega\ A \vdash B}{\Omega \vdash B \,/\, A}\ / R \qquad \frac{\Omega' \vdash A \quad \Omega_L\ B\ \Omega_r \vdash C}{\Omega_L\ (B \,/\, A)\ \Omega'\ \Omega_R \vdash C}\ / L$$

$$\frac{\Omega \vdash A \quad \Omega' \vdash B}{\Omega\ \Omega' \vdash A \bullet B}\ \bullet R \qquad \frac{\Omega_L\ A\ B\ \Omega_R \vdash C}{\Omega_L\ (A \bullet B)\ \Omega_R \vdash C}\ \bullet L$$

$$\frac{\Omega \vdash B \quad \Omega' \vdash A}{\Omega\ \Omega' \vdash A \circ B}\ \circ R \qquad \frac{\Omega_L\ B\ A\ \Omega_R \vdash C}{\Omega_L\ (A \circ B)\ \Omega_R \vdash C}\ \circ L$$

$$\frac{}{\cdot \vdash \mathbf{1}}\ \mathbf{1}R \qquad \frac{\Omega_L\ \Omega_R \vdash C}{\Omega_L\ \mathbf{1}\ \Omega_R \vdash C}\ \mathbf{1}L$$

$$\frac{\Omega \vdash A}{\Omega \vdash A \oplus B}\ \oplus R_1 \quad \frac{\Omega \vdash B}{\Omega \vdash A \oplus B}\ \oplus R_2 \qquad \frac{\Omega_L\ A\ \Omega_R \vdash C \quad \Omega_L\ B\ \Omega_R \vdash C}{\Omega_L\ (A \oplus B)\ \Omega_R \vdash C}\ \oplus L$$

$$\frac{\Omega \vdash A \quad \Omega \vdash B}{\Omega \vdash A \,\&\, B}\ \&R \qquad \frac{\Omega_L\ A\ \Omega_R \vdash C}{\Omega_L\ (A \,\&\, B)\ \Omega_R \vdash C}\ \&L_1 \quad \frac{\Omega_L\ B\ \Omega_R \vdash C}{\Omega_L\ (A \,\&\, B)\ \Omega_R \vdash C}\ \&L_2$$

$$
\begin{array}{llll}
\text{Types} & A, B, C & ::= & \oplus\{l_i : A_i\}_{i\in I} \mid \&\{l_i : A_i\}_{i\in I} \mid \mathbf{1} \\
& & \mid & A \mathbin{/} B \mid B \setminus A \mid A \bullet B \mid A \circ B \\
\\
\text{Processes} & P, Q & ::= & x \leftarrow y & \text{identity/forward} \\
& & \mid & x \leftarrow P_x \mathbin{;} Q_x & \text{cut/spawn} \\
& & \mid & x.l_k \mathbin{;} P \mid \mathsf{case}\ x\ (l_i \Rightarrow Q_i)_{i\in I} & \oplus, \& \\
& & \mid & \mathsf{close}\ x \mid \mathsf{wait}\ x \mathbin{;} Q & \mathbf{1} \\
& & \mid & \mathsf{send}\ x\ y \mathbin{;} P \mid y \leftarrow \mathsf{recv}\ x \mathbin{;} Q_x & /, \setminus, \bullet, \circ
\end{array}
$$

Judgmental Rules

$$
\frac{\Omega \vdash P_x :: (x{:}A) \quad \Omega_L\ (x{:}A)\ \Omega_R \vdash Q_x :: (z{:}C)}{\Omega_L\ \Omega\ \Omega_R \vdash (x \leftarrow P_x \mathbin{;} Q_x) :: (z{:}C)}\ \mathsf{cut} \qquad \frac{}{y{:}A \vdash x \leftarrow y :: (x{:}A)}\ \mathsf{id}
$$

Propositional Rules

$$
\frac{\Omega \vdash P :: (x{:}A_k) \quad (k \in I)}{\Omega \vdash (x.l_k \mathbin{;} P) :: (x : \oplus\{l_i{:}A_i\}_{i\in I})}\ \oplus R_k \qquad \frac{\Omega_L\ (x{:}A_i)\ \Omega_R \vdash Q_i :: (z{:}C) \quad (\forall i \in I)}{\Omega_L\ (x{:}\oplus\{l_i{:}A_i\}_{i\in I})\ \Omega_R \vdash \mathsf{case}\ x\ (l_i \Rightarrow Q_i)_{i\in I} :: (z{:}C)}\ \oplus L
$$

$$
\frac{\Omega \vdash P_i :: (x{:}A_i) \quad (\forall i \in I)}{\Omega \vdash \mathsf{case}\ x\ (l_i \Rightarrow P_i)_{i\in I} :: (x{:}\&\{l_i{:}A_i\}_{i\in I}))}\ \&R \qquad \frac{\Omega_L\ (x{:}A_k)\ \Omega_R \vdash P :: (z{:}C) \quad (k \in I)}{\Omega_L\ (x : \&\{l_i{:}A_i\}_{i\in I})\ \Omega_R \vdash (x.l_k \mathbin{;} Q) :: (z{:}C)}\ \&L_k
$$

$$
\frac{}{\cdot \vdash \mathsf{close}\ x :: (x{:}\mathbf{1})}\ \mathbf{1}R \qquad \frac{\Omega_L\ \Omega_R \vdash Q :: (z{:}C)}{\Omega_L\ (x{:}\mathbf{1})\ \Omega_R \vdash (\mathsf{wait}\ x \mathbin{;} Q) :: (z{:}C)}\ \mathbf{1}L
$$

$$
\frac{\Omega\ (y{:}A) \vdash P_y :: (x{:}B)}{\Omega \vdash (y \leftarrow \mathsf{recv}\ x \mathbin{;} P_y) :: (x{:}B \mathbin{/} A)}\ /R \qquad \frac{\Omega_L\ (x{:}B)\ \Omega_R \vdash Q :: (z{:}C)}{\Omega_L\ (x{:}B \mathbin{/} A)\ (w{:}A)\ \Omega_R \vdash (\mathsf{send}\ x\ w \mathbin{;} Q) :: (z{:}C)}\ /L^*
$$

$$
\frac{(y{:}A)\ \Omega \vdash P_y :: (x{:}B)}{\Omega \vdash (y \leftarrow \mathsf{recv}\ x \mathbin{;} P_y) :: (x{:}A \setminus B)}\ \setminus R \qquad \frac{\Omega_L\ (x{:}B)\ \Omega_R \vdash Q :: (z{:}C)}{\Omega_L\ (w{:}A)\ (x{:}A \setminus B)\ \Omega_R \vdash (\mathsf{send}\ x\ w \mathbin{;} Q) :: (z{:}C)}\ \setminus L^*
$$

$$
\frac{\Omega \vdash P :: (x{:}B)}{(w{:}A)\ \Omega \vdash (\mathsf{send}\ x\ w \mathbin{;} P) :: (x{:}A \bullet B)}\ \bullet R^* \qquad \frac{\Omega_L\ (y{:}A)\ (x{:}B)\ \Omega_R \vdash Q_y :: (z{:}C)}{\Omega_L\ (x{:}A \bullet B)\ \Omega_R \vdash (y \leftarrow \mathsf{recv}\ x \mathbin{;} Q_y) :: (z{:}C)}\ \bullet L
$$

$$
\frac{\Omega \vdash P :: (x{:}B)}{\Omega\ (w{:}A) \vdash (\mathsf{send}\ x\ w \mathbin{;} P) :: (x{:}A \circ B)}\ \circ R^* \qquad \frac{\Omega_L\ (x{:}B)\ (y{:}A)\ \Omega_R \vdash Q_y :: (z{:}C)}{\Omega_L\ (x{:}A \bullet B)\ \Omega_R \vdash (y \leftarrow \mathsf{recv}\ x \mathbin{;} Q_y) :: (z{:}C)}\ \circ L
$$

Computation Rules

$$
\frac{\mathsf{proc}(z, x \leftarrow P_x \mathbin{;} Q_x)}{\mathsf{proc}(w, P_w) \quad \mathsf{proc}(z, Q_w)}\ \mathsf{cmp}^w \qquad \frac{\mathsf{proc}(x, x \leftarrow y)}{x = y}\ \mathsf{fwd} \qquad \frac{\mathsf{proc}(x, \mathsf{close}\ x) \quad \mathsf{proc}(z, \mathsf{wait}\ x \mathbin{;} Q)}{\mathsf{proc}(z, Q)}\ \mathbf{1}C
$$

$$
\frac{\mathsf{proc}(x, x.l_k \mathbin{;} P) \quad \mathsf{proc}(z, \mathsf{case}\ x\ (l_i \Rightarrow Q_i)_{i\in I})}{\mathsf{proc}(x, P) \quad \mathsf{proc}(z, Q_k)}\ \oplus C \qquad \frac{\mathsf{proc}(x, \mathsf{case}\ x\ (l_i \Rightarrow P_i)_{i\in I}) \quad \mathsf{proc}(z, x.l_k \mathbin{;} Q)}{\mathsf{proc}(x, Q) \quad \mathsf{proc}(z, P_k)}\ \&C
$$

$$
\frac{\mathsf{proc}(x, y \leftarrow \mathsf{recv}\ x \mathbin{;} P_y) \quad \mathsf{proc}(z, \mathsf{send}\ x\ w \mathbin{;} Q)}{\mathsf{proc}(x, P_w) \quad \mathsf{proc}(z, Q)}\ /C, \setminus C \qquad \frac{\mathsf{proc}(x, \mathsf{send}\ x\ w \mathbin{;} P) \quad \mathsf{proc}(z, y \leftarrow \mathsf{recv}\ x \mathbin{;} Q_y)}{\mathsf{proc}(P) \quad \mathsf{proc}(Q_w)}\ \bullet C, \circ C
$$