Delineating the polynomial hierarchy in a fragment of intuitionistic logic via *over*-focussing

Anupam Das

École Normale Supérieure de Lyon, France
anupam.das@ens-lyon.fr

This abstract is adapted from the article [5] which was presented at DaleFest in 2016 and is currently under submission for the associated Festschrifft.

1 Introduction and motivation

Focussed proof systems [1, 10] organise the process of bottom-up proof search into 'synchronous' and 'asynchronous' phases, distinguishing the steps where genuine choices are made from those where no information is lost. This treatment of proof search has close connections to logic programming [13] and the Curry-Howard correspondence [8]. The purpose of this article is to examine the complexity-theoretic aspects of focussed systems.

We may naturally view synchronous phases as nondeterministic computation and asynchronous phases as co-nondeterministic computation, in the proof-search-as-computation paradigm. Consequently we may use focussed systems to obtain complexity bounds for logics in terms of *alternating time*. The main advantage of this approach is the 'focussing theorem', which allows systems to significantly reduce the number of alternations between invertible and noninvertible steps in the proof search space. As an analogy, this extends the idea that, for quantified Boolean formulae (QBF), the quantifier hierarchy tightly delineates the levels of the polynomial hierarchy; from a proof-theoretic viewpoint this is exemplified by the alternation of invertible and noninvertible rule phases in Boolean Truth Trees [9].

It is not difficult to formalise complexity bounds obtained from a focussed system. Let us, for example, consider a logic L that is **PSPACE**-complete. We can encode a family 'provability predicates' as, say, QBFs parametrised by the *decide-depth* of the proof search space. Decide-depth can be calculated in polynomial time, as is implicitly shown in e.g. [14], and so give rises to an encoding of L into true QBFs (QCPL), with formulae of the same decide-depth mapped to ones of the same quantifier complexity. While the fact that this can be done is essentially folklore, we formalise this encoding in Section 2 of [5].

However, the bounds obtained by doing this directly can be far from optimal. While there can always be improvements, there is one particular aspect of focussed systems which does not adequately preserve the analogy with nondeterministic and co-nondeterministic computation. In a focussed system invertible rules must be applied during an asynchronous phase, regardless of whether they are branching or non-branching. However, a non-branching invertible rule corresponds to *deterministic* computation, and so does not contribute to alternation complexity if we are in a synchronous phase. For the sake of example, consider the following encoding in IPL of a SAT instance A over variables $\vec{x}$,

$$\begin{array}{rcl} A_0 & := & A \\ A_{i+1} & := & (A_i \supset a_i) \supset ((x_i \supset a_i) \vee (\neg x_i \supset a_i)) \end{array}$$

where the variables a_i are fresh. We have that $A \in \mathsf{SAT}$ if and only if $A_{n+1} \in \mathsf{IPL}$ (see [5]).

Let us consider a proof in the usual focussed system *LJF* for IPL, using the notation of [12], of A_{n+1}, built up inductively:[1]

$$
\dfrac{
 \dfrac{
  {\scriptstyle S_l,S_r}\dfrac{
   {\scriptstyle D_r}\dfrac{
    \dfrac{
     {\scriptstyle R_r}\dfrac{
      \dfrac{
       {\scriptstyle S_l,S_r}\dfrac{
        {\scriptstyle D_l}\dfrac{
         \dfrac{
          {\scriptstyle R_r}\dfrac{\dfrac{IH}{\pm x_n \Uparrow \cdot \vdash A_n \Uparrow \cdot}}{\pm x_n \vdash A_n \Downarrow}
          \quad
          \dfrac{}{\cdot \Downarrow a_n \vdash a_n}
         }{\pm x_n \Downarrow A_n \supset a_n \vdash a_n}
        }{A_n \supset a_n, \pm x_n \Uparrow \cdot \vdash \cdot \Uparrow a_n}
       }{A_n \supset a_n \Uparrow \pm x_n \vdash a_n \Uparrow \cdot}
      }{A_n \supset a_n \Uparrow \cdot \vdash (\pm x_n \supset a_n) \Uparrow \cdot}
     }{A_n \supset a_n \vdash (\pm x_n \supset a_n) \Downarrow}
    }{A_n \supset a_n \vdash (x_n \supset a_n) \vee (\neg x_n \supset a_n) \Downarrow}
   }{A_n \supset a_n \Uparrow \cdot \vdash \cdot \Uparrow (x_n \supset a_n) \vee (\neg x_n \supset a_n)}
  }{\cdot \Uparrow A_n \supset a_n \vdash (x_n \supset a_n) \vee (\neg x_n \supset a_n) \Uparrow \cdot}
 }{}
}{\cdot \Uparrow \cdot \vdash (A_n \supset a_n) \supset ((x_n \supset a_n) \vee (\neg x_n \supset a_n) \Uparrow \cdot}
$$

Notice that the only proofs have the format above and their decide and release depth increase linearly with the number of variables in A. The natural encoding of focussed proof search yields only a Σ^p_{2k} upper bound for formulae over k variables, far from the optimal bound of $\mathbf{NP} = \Sigma^p_1$. On closer inspection, observe that the only reason that so many phase alternations occur is the application of right-implication between release and decide steps, which is invertible and, in particular, non-branching. Consequently, we can actually encode the proof search procedure using only existential quantifiers, and so the dynamics of proof search 'morally' represent an **NP** procedure.

In fact we can go further than this and extend the focussing methodology itself: if we are in a synchronous phase in bottom-up proof search and reach a connective whose corresponding rule is invertible and non-branching, then we may often apply the rule but keep a focus on one of its auxiliary formulae, without losing completeness. We give an example of such a system, which we call an 'over-focussed' system, in the next section.

We point out that this approach is orthogonal from 'multi-focussed' proof systems [4], which rather address situations when *parallel* bipoles commute. Such systems allow *more* proofs, since each uni-focussed proof is also a multi-focussed proof. In particular this means that we cannot obtain better bounds for proof search from such systems, since their search space is strictly larger. On the other hand, an over-focussed system is more restrictive and admits fewer proofs, and so is better suited for complexity-theoretic analysis.

2 Case study: a fragment of IPL

Let us from now on write $\vDash_i$ for intuitionistic entailment and $\vDash_c$ for classical entailment.

Statman proved the **PSPACE**-hardness of IPL in [15] by encoding QCPL, known to be **PSPACE**-complete, into it. The fundamental observation is that,

$$
\begin{array}{lcl}
\vDash_c \forall x.A & \iff & \vDash_i (x \vee \neg x) \supset A' \\
\vDash_c \exists x.A & \iff & \vDash_i (x \supset A') \vee (\neg x \supset A')
\end{array}
\tag{1}
$$

[1] Notice that there are just as many proofs as there are satisfying assignments.

where A' is obtained by the inductive hypothesis.[2] The problem with this is that the $\exists$ case includes two copies of A', and so the size of the encoding grows exponentially with the number of $\exists$ quantifiers. One way around this is to instead map QBFs to intuitionistic *circuits*, which can share the copy of A' between the two disjuncts, yielding polynomial-size circuits.[3] From here one can simply use Tseitin extension variables to encode the local conditions of the circuit as a formula, and this is exactly what Statman does in his translation.

Theorem 1 (Statman)**.** *For each QBF A there is a (polynomially larger) propositional formula A^* such that $\vDash_c A$ if and only if $\vDash_i A^*$.*

While this indeed yields a proof of the **PSPACE**-hardness of IPL, there are many issues and questions which naturally arise.

Delineating the polynomial hierarchy in IPL There are many encodings from IPL back into QCPL, the most immediate of which arises by simply encoding the Kripke relational semantics of IPL. Other encodings can be obtained by encoding the topological semantics or game semantics [2], and of course by encoding proof search too. However, as far as we know, there is no 'natural' translation that acts as an 'inverse' of the Statman translation in the sense that quantifier complexity is preserved by the composition of the two translations. More precisely we ask the following:

Question 2. *Are there 'natural' encodings $t_1 : \mathsf{QCPL} \to \mathsf{IPL}$ and $t_2 : \mathsf{IPL} \to \mathsf{QCPL}$, such that:*

- *$\vDash_c A$ if and only if $\vDash_i t_1(A)$.*
- *$\vDash_i A$ if and only if $\vDash_c t_2(A)$.*
- *$t_2 \circ t_1$ preserves quantifier complexity.*

The purpose of this section is to identify a well-behaved variant of Statman's translation and a corresponding proof system with which we can resolve the above question via a suitable encoding of proof search.

Invertible rules for IPL? Dyckhoff asks in [6] whether there exists a (terminating) system of invertible rules sound and complete for IPL. This is also noted as an open problem by Buss and Iemhoff in [3]. As we have already mentioned, such a system cannot exhibit optimal proof search complexity, but it is of natural proof theoretic interest.

In fact a positive answer to Question 2 above would partially resolve this open problem: since there is already a well-known terminating system of invertible rules for QCPL, we can just consider the translations of those rules under t_1. Invertibility and termination follows by correctness of the encoding, and the resulting system will be complete for the image of t_1.

To obtain a system complete for all of IPL we must also appeal to t_2: for each formula A of IPL we may construct a rule whose conclusion is A and whose premisses are the t_1-image of the premisses of a QCPL rule whose conclusion is $t_2(A)$. If we only consider the 'synthetic' rules, decomposing an entire block of existentials quantifiers or universal quantifiers at once, the system is terminating by the fact that $t_2 \circ t_1$ preserves quantifier complexity and the corresponding termination argument for the QCPL-calculus. The existence of such a system

[2] We ignore the base case here for simplicity.

[3] In fact, it is not hard to see that these circuits have type-level 1, meaning that the class of type 1 intuitionistic circuits is already **PSPACE**-complete.

for an expressive fragment[4] PTPL of IPL is a direct consequence of the results in this section. While this is not an ideal formulation of a proof system, since the number of premisses for a rule is unbounded, the discovery of a 'natural' such t_2 could provide useful intuitions towards finding a *bona fide* system of terminating invertible rules for IPL.

A refined translation and the positive Tseitin fragment

One problem with resolving Question 2 above is with Statman's translation itself: it is somehow too insensitive to the structure of proofs induced by the translation. At the same time, usual proof systems do not handle well Tseitin variables, which act as abbreviations: while these are essentially deterministic pieces of information, a proof system would naturally see at least one direction of the equivalence as a nondeterministic choice.

However notice that, in (1), A' only ever occurs in positive context, and so we only need one direction of the Tseitin equivalences, of the form $A \supset a$ and not $a \supset A$. We use this to construct a refined version of the translation that allows us to have better control over the proof theory of formulae in its image.

Definition 3. We mutually define the following classes of formulae:

$$L \quad ::= \quad x \quad | \quad \bot \quad | \quad R \supset a \quad | \quad L \wedge L \quad | \quad L \vee L$$

$$R \quad ::= \quad x \quad | \quad \bot \quad | \quad L \supset R \quad | \quad R \wedge R \quad | \quad R \vee R$$

An R-formula is called a *positive Tseitin* formula. The *positive Tseitin fragment* of IPL, denoted PTPL, is the set of intuitionistically valid positive Tseitin formulae.

Importantly, positive Tseitin formulae remain rich enough to encode QBFs.

Definition 4 (Refined translation)**.** We define the following translation from prenex QBFs to (unquantified) propositional formulae,

$$\begin{array}{rcl} \langle A_0 \rangle & := & A_0 \\ \langle \forall x.A \rangle & := & (x \vee \neg x) \supset \langle A \rangle \\ \langle \exists x.A \rangle & := & (\langle A \rangle \supset a) \supset ((x \supset a) \vee (\neg x \supset a)) \end{array}$$

where A_0 is quantifier-free, A is in prenex normal form and a is fresh.

Theorem 5. *If A is a closed prenex QBF, then $A \in$ QCPL if and only if $\langle A \rangle \in$ PTPL.*

Corollary 6. PTPL *is already* **PSPACE**-*complete.*

One particularly appealing property of positive Tseitin formulae is the following result:

Proposition 7. *An L-formula is either a disjunction or a Harrop formula.*[5]

This means that, in bottom-up proof search, once all disjunctions on the left have been decomposed, we can always make some choice of disjunct on the right. This also gives us access to a simple contraction-free and cut-free system for PTPL.

[4] In particular it is **PSPACE**-complete.

[5] Note that L-formulae are not, in general, hereditary Harrop, for essentially any formulation of hereditary Harrop formulae, cf. [11].

Definition 8 (Proof system)**.** We define the system *LPT* as follows:[6]

$$\bot\text{-}l \frac{}{\Gamma, \bot \vdash A} \qquad eval \frac{l(\vec{a}) \vDash_c A(\vec{a})}{\Gamma, l(\vec{a}) \vdash A(\vec{a})}$$

$$\vee\text{-}l \frac{\Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma, A \vee B \vdash C} \qquad \supset\text{-}l \frac{\Gamma \vdash A}{\Gamma, A \supset a \vdash a} \qquad \wedge\text{-}l \frac{\Gamma, A, B \vdash C}{\Gamma, A \wedge B \vdash C}$$

$$\wedge\text{-}r \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \qquad \supset\text{-}r \frac{\Gamma, A \vdash B}{\Gamma \vdash A \supset B} \qquad \vee\text{-}r \frac{\Gamma \vdash A_i}{\Gamma \vdash A_1 \vee A_2} \; i \in \{1, 2\}$$

By inspection of these rules, we immediately have the following:

Proposition 9. *Any LPT-derivation with a positive Tseitin conclusion contains only L-formulae on the left and R-formulae on the right.*

This result, together with Proposition 7, allows us to routinely prove the completeness of *LPT* with respect to PTPL.

Theorem 10. *LPT is sound and complete for* PTPL*.*

An 'over-focussed' system for PTPL

Usually, focussed systems classify formulae as either 'negative' or 'positive'. In light of our comments in the previous section, we will be slightly more fine-grained in our classification, also distinguishing 'deterministic' formulae.

Remark 11. Notice that there is a peculiar phenomenon in *LPT* that the conjunction rules are both invertible. From the point of view of focussing, this is because we have actually used two different versions of conjunction, positive ($\wedge^+$) on the left and negative ($\wedge^-$) on the right. However, notice that *LPT* respects the polarity of a conjunction symbol in a formula (in the sense of even or odd depth to the left of implications), and so the distinction is already present at the level of formulae: *L*-formulae are composed by $\wedge^+$, while *R*-formulae are composed by $\wedge^-$. We will typically omit such annotation when it is clear from context. For similar reasons (and by the same argument), we will always assume that atoms are negative on the left of a sequent and positive on the right.

Definition 12 (Classification)**.** A *positive* formula is a $\wedge^+$ or $\vee$ formula (or a positive atom). A *negative* formula is a $\wedge^-$ or $\supset$ formula (or a negative atom). A *deterministic* formula is a $\wedge^+$ or $\supset$ formula (or an atom). We will use the following metavariables:

M	:	negative and not deterministic	$\wedge^-$
N	:	negative	$\wedge^-, \supset$
O	:	deterministic	$\wedge^+, \supset$
P	:	positive	$\wedge^+, \vee$
Q	:	positive and not deterministic	$\vee$

[6] The rule *eval* here is purely for convenience of proof analysis. It can in fact be replaced by several invertible non-branching rules (so deterministic), and this is discussed in [5].

The reason for making such distinctions is that we will admit deterministic computations in both the negative and positive phases, allowing the decomposition of positive deterministic formulae on the left and negative deterministic formulae on the right. Referring to the previous section, the corresponding such rules are invertible and non-branching. Since this is more restrictive than usual focussing, we will call this an 'over-focussed' system, and refer to the phases as 'co-nondeterministic' and 'nondeterministic' rather than 'asynchronous' and 'synchronous', respectively, which this is more compatible with our complexity-theoretic point of view.

We do not explicitly give the specification of the over-focussed system *LPTF* here due to space considerations, but it can be found in [5], or Appendix A for convenience. We do however take a moment to explain the format of lines and the dynamics of proof search in this system. There are three kinds of sequent in this system:

$$\Gamma \Uparrow \Delta \vdash \Lambda_1 \Uparrow \Lambda_2 \tag{2}$$

$$\Gamma; \Delta \Downarrow A \vdash B \tag{3}$$

$$\Gamma; \Delta \vdash A \Downarrow \tag{4}$$

where Γ and Δ are multisets of formulae and $|\Lambda_1 \sqcup \Lambda_2| = 1$. (2) is called an *unfocussed* sequent, whereas (3) and (4) are *focussed*, left and right respectively. In both the latter cases the 'focus' of the sequent is A, whereas Δ consists of the 'residues'.

Bottom-up proof search operates as follows. First, a co-nondeterministic phase is conducted exactly analogous to an asynchronous phase in, say, *LJF*, until we reach a positive sequent. Then we enter a nondeterministic phase after 'deciding' on a focus. This is similar to a synchronous phase in *LJF*, except that if we reach a deterministic formula we must keep decomposing one of its auxiliary formulae. Other auxiliary formulae are temporarily stored as residues and are reduced until they become positive, when they are stored for real, or negative and non-deterministic. The phase is complete once only some invertible branching rules apply to the focus and the remaining residues, when we release and enter again the co-nondeterministic phase. If there is no such rule, i.e. there are no remaining residues and the focus is atomic, then we must 'redecide' a formula without releasing and remain in the nondeterministic phase.

Compatibility of proof search with the refined translation

We use the usual notions of decide depth (dd) and release depth (rd) of proofs and search spaces, counting alternations of only D_l or D_r and R_l or R_r steps respectively. For a search space this can be calculated in polynomial time from a formula by a 'worst case' analysis, similar to that in [14]. Consequently, we obtain from *LPTF* an encoding of PTPL back into QCPL.

Let us write Π^q_i (or Σ^q_i) to denote the class of prenex QBFs with i alternations of quantifiers beginning with a universal (or existential, respectively).

Theorem 13. *There is an encoding* $[\cdot] : \mathsf{PTPL} \to \mathsf{QCPL}$*, such that* $[A]$ *is in* $\Sigma^q_{dd(A)+rd(A)}$.[7]

By examining the dynamics of proof search, we may explicitly calculate the decide and release depths of formulae in the image of our refined translation. Finally, we can use these calculations to show that our encoding of proof search for *LPTF* in QCPL is well-calibrated with our refined Statman translation from QCPL to PTPL.

Theorem 14. *If* $A \in \Sigma^q_i$ *is valid then* $dd(\langle A\rangle) + rd(\langle A\rangle) = i$.

Putting these together, we conclude our partial resolution to Question 2:

Corollary 15. $[\cdot] \circ \langle\cdot\rangle : \mathsf{QCPL} \to \mathsf{PTPL} \to \mathsf{QCPL}$ *preserves quantifier complexity.*

[7] An analogous bound for co-nondeterministic complexity can be made too.

3 Conclusions

In this work we discussed how variants of focussed systems can be used to extract alternating time bounds for fragments of a logic. With this in mind, we have motivated the notion of 'over-focussing', in order to more faithfully represent the stages of nondeterministic and co-nondeterministic computation in proof search.

We considered as a case study an adequately expressive fragment PTPL of intuitionistic propositional logic and gave a sound and complete over-focussed system for it. We showed that the natural bounds extracted from this system are tight with respect to a refinement of Statman's translation from QCPL, which has various consequences for the proof theory of IPL, as discussed in the beginning of Section 2 and in more detail in [5].

As we already mentioned, our quantifier-complexity preserving pair of encodings QCPL $\to$ PTPL $\to$ QCPL allows us to infer the existence of a terminating 'macro-system' of invertible rules for PTPL, partially addressing a question of Dyckhoff [6]. It would be interesting to see if analysis of this macro-system could yield the necessary insights to construct a *bona fide* terminating system of invertible rules for IPL. However for this it seems likely that we would need to have a richer encoding from all of IPL, not just PTPL, to QCPL, that similarly behaves well with respect to the (refined) Statman translation. In future work we intend to investigate this possibility via an over-focussed version of the contraction-free system *G4ip* for IPL [6].

We repeat the point that just the existence of an encoding of t_1 : QCPL $\to$ IPL, e.g. Statman's translation or the refined version in this work, already yields a terminating system of invertible rules for the image of t_1, by direct translation of such a system for QCPL.

Acknowledgments

I would like to thank David Baelde and Roy Dyckhoff for several enlightening discussions on various aspects of this work.

References

[1] Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *J. Log. Comput.*, 2(3):297–347, 1992.

[2] Nick Bezhanishvili and Dick de Jongh. *Intuitionistic logic.*

[3] Samuel R Buss and Rosalie Iemhoff. The depth of intuitionistic cut free proofs, 2003.

[4] Kaustuv Chaudhuri, Stefan Hetzl, and Dale Miller. A multi-focused proof system isomorphic to expansion proofs. *Journal of Logic and Computation*, page exu030, 2014.

[5] Anupam Das. From focussed proof systems to complexity bounds, 2016. http://anupamdas.com/items/focussing-complexity/focussed-systems-complexity-bounds.pdf. *DaleFest.*

[6] Roy Dyckhoff. Intuitionistic decision procedures since gentzen. In *Advances in Proof Theory*, pages 245–267. Springer, 2016.

[7] Gilda Ferreira and Paulo Oliva. On various negative translations. *arXiv preprint arXiv:1101.5442*, 2011.

[8] Stéphane Graham-Lengrand. Polarities & focussing: a journey from realisability to automated reasoning. *CoRR*, abs/1412.6781, 2014.

[9] Alexander Hertel and Alasdair Urquhart. Proof complexity of intuitionistic propositional logic. 2006.

[10] Chuck Liang and Dale Miller. Focusing and polarization in linear, intuitionistic, and classical logics. *Theoretical Computer Science*, 410(46):4747–4768, 2009.

[11] Dale Miller. Logic programming based on higher-order hereditary harrop formulas. 1988.

[12] Dale Miller. Focused LJ. *Encyclopaedia of Proof Systems*, 2015.

[13] Dale Miller and Gopalan Nadathur. *Programming with Higher-Order Logic*. Cambridge University Press, 2012.

[14] Vivek Nigam. Investigating the use of lemmas.

[15] Richard Statman. Intuitionistic propositional logic is polynomial-space complete. *Theor. Comput. Sci.*, 9:67–72, 1979.

[16] Vitezslav Svejdar. On the polynomial-space completeness of intuitionistic propositional logic. *Arch. Math. Log.*, 42(7):711–716, 2003.

A The over-focussed system *LPTF*

Co-nondeterministic phase

$$\frac{}{\Gamma \Uparrow \bot, \Delta \vdash \Uparrow P} \qquad {\scriptstyle eval\Uparrow}\,\frac{}{\Gamma, l(\vec{a}) \Uparrow \cdot \vdash \Uparrow P(\vec{a})}\; l(\vec{a}) \vDash_c P(\vec{a})$$

$$\frac{\Gamma \Uparrow A, \Delta \vdash C \Uparrow \quad \Gamma \Uparrow B, \Delta \vdash C \Uparrow}{\Gamma \Uparrow A \vee B, \Delta \vdash C \Uparrow} \qquad \frac{\Gamma \Uparrow A, B, \Delta \vdash C \Uparrow}{\Gamma \Uparrow A \wedge^{+} B, \Delta \vdash C \Uparrow} \qquad {\scriptstyle S_l}\,\frac{\Gamma, N \Uparrow \Delta \vdash C \Uparrow}{\Gamma \Uparrow N, \Delta \vdash C \Uparrow}$$

$$\frac{\Gamma \Uparrow \cdot \vdash A \Uparrow \quad \Gamma \Uparrow \cdot \vdash B \Uparrow}{\Gamma \Uparrow \cdot \vdash A \wedge^{-} B \Uparrow} \qquad \frac{\Gamma \Uparrow A \vdash B \Uparrow}{\Gamma \Uparrow \cdot \vdash A \supset B \Uparrow} \qquad {\scriptstyle S_r}\,\frac{\Gamma \Uparrow \cdot \vdash \Uparrow P}{\Gamma \Uparrow \cdot \vdash P \Uparrow}$$

where $l(\vec{a})$ is a list containing precisely one of a or $\neg a$ for each a in $\vec{a}$.

Nondeterministic phase

$${\scriptstyle D_l}\,\frac{\Gamma;\ \Downarrow N \vdash P}{\Gamma, N \Uparrow \cdot \vdash \Uparrow P} \qquad {\scriptstyle D_r}\,\frac{\Gamma;\ \vdash P^{\not a} \Downarrow}{\Gamma \Uparrow \cdot \vdash \Uparrow P^{\not a}} \qquad \frac{\Gamma; \Delta \vdash A \Downarrow}{\Gamma; \Delta \Downarrow A \supset a \vdash a} \qquad \frac{\Gamma; \Delta \vdash A_i \Downarrow}{\Gamma; \Delta \vdash A_1 \vee A_2 \Downarrow}\; i \in \{1, 2\}$$

where D_l and D_r apply only if the conclusion is not an instance of *eval* $\Uparrow$, and $P^{\not a}$ is not atomic.

$$\frac{\Gamma; \Delta, A_i \Downarrow A_j \vdash B}{\Gamma; \Delta \Downarrow A_1 \wedge^{+} A_2 \vdash B}\; \{i, j\} = \{1, 2\} \qquad \frac{\Gamma; \Delta \Downarrow A \vdash B}{\Gamma; \Delta \vdash A \supset B \Downarrow} \qquad \frac{\Gamma; \Delta, A \vdash B \Downarrow}{\Gamma; \Delta \vdash A \supset B \Downarrow}$$

$$\frac{\Gamma; A, B, \Delta\ \mathcal{L} \vdash \mathcal{R}}{\Gamma; A \wedge^{+} B, \Delta\ \mathcal{L} \vdash \mathcal{R}} \qquad {\scriptstyle s_l^N}\,\frac{\Gamma, N; \Delta\ \mathcal{L} \vdash \mathcal{R}}{\Gamma; N, \Delta\ \mathcal{L} \vdash \mathcal{R}} \qquad \frac{\Gamma; \Delta, B \Downarrow A \vdash C}{\Gamma; \Delta \Downarrow A \vdash B \supset C}$$

where either $\mathcal{L}$ is $\Downarrow C$ and $\mathcal{R}$ is D or $\mathcal{L}$ is empty and $\mathcal{R}$ is $D \Downarrow$.

$${\scriptstyle s_l^Q}\,\frac{\Gamma; \Delta, Q \Downarrow \cdot \vdash A}{\Gamma; \Delta \Downarrow Q \vdash A} \qquad {\scriptstyle s_l^a}\,\frac{\Gamma, a; \Delta \Downarrow \cdot \vdash A}{\Gamma; \Delta \Downarrow a \vdash A} \qquad {\scriptstyle d_l}\,\frac{\Gamma;\ \Downarrow N \vdash P}{\Gamma, N;\ \Downarrow \cdot \vdash P} \qquad {\scriptstyle d_l^a}\,\frac{\Gamma;\ \Downarrow N \vdash a}{\Gamma, N;\ \vdash a \Downarrow} \qquad {\scriptstyle d_r}\,\frac{\Gamma;\ \vdash P \Downarrow}{\Gamma;\ \Downarrow \cdot \vdash P}$$

$$\frac{}{\Gamma; \Delta \Downarrow \bot \vdash R} \qquad {\scriptstyle eval\Downarrow}\,\frac{}{\Gamma, l(\vec{a}); \Delta \vdash M^a(\vec{a}) \Downarrow}\; l(\vec{a}) \vDash_c M^a(\vec{a}) \qquad {\scriptstyle R_l}\,\frac{\Gamma \Uparrow \vec{Q} \vdash M^a \Uparrow}{\Gamma; \vec{Q} \vdash M^a \Downarrow} \qquad {\scriptstyle R_r}\,\frac{\Gamma \Uparrow \vec{Q} \vdash R \Uparrow}{\Gamma; \vec{Q} \Downarrow \cdot \vdash R}$$

where M^a is either M or an atom and R is either M or, as long as $\vec{Q}$ is nonempty, P. Furthermore, the rules R_l and R_r apply only if the conclusion is not an instance of *eval* $\Downarrow$. Again, $l(\vec{a})$ is a list containing precisely one of a or $\neg a$ for each a in $\vec{a}$.