Complexity of fragments of intuitionistic logic with disjunction

R. Ramanujam [*] Vaishnavi Sundararajan [†] S. P. Suresh [‡]

Abstract

In the formal study of security protocols and access control systems, fragments of intuitionistic logic play a vital role. These are required to be efficient, and are typically disjunction-free. In this paper, we study the complexity of adding disjunction to these subsystems. Our lower bound results show that very little needs to be added to disjunction to get **co-NP**-hardness, while our upper bound results show that even a system with conjunction, disjunction, and restricted forms of negation and implication is in **co-NP**. Our upper bound proofs also suggest parameters which we can bound to obtain **PTIME** algorithms.

1 Introduction

Intuitionistic logic is a subject with a rich history, with connections to fundamental aspects of mathematics, philosophy and computer science. What is perhaps surprising is that it also finds application in such concrete areas of computer science as system security and communication security in distributed protocols. Consider the question: given a finite set of formulas X, a formula a in a positive fragment of some propositional logic, and an intuitionistic proof system $\vdash$, does $X \vdash a$? This sounds arcane, but is of practical importance when X is a security policy that specifies permissions and a is the assertion of someone being permitted some action [1, 10]. Or it might be the case that X is a set of terms picked by an eavesdropper watching a channel and a is a term to be kept secret [8]. Inference in such situations is typically intuitionistic. Consider a formula A **has** t for an agent A participating in a cryptographic protocol and a term t. A different agent B might not be able to assert $(A \textbf{ has } t) \vee \neg(A \textbf{ has } t)$, since it might be that B does not have all the components that go into building the term t and the system does not allow B to assert anything about t in such a case. To consider another example, B cannot assert A **has** t by assuming

[*] The Institute of Mathematical Sciences, Chennai, India. jam@imsc.res.in
[†] Chennai Mathematical Institute, Chennai, India. vaishnavi@cmi.ac.in
[‡] Chennai Mathematical Institute, Chennai, India. spsuresh@cmi.ac.in

$\neg(A$ **has** $t)$ and then deriving a contradiction. To consider a third example, consider a formula A **can read** f, where A is a user and f is a file. An access control policy may be silent on whether A can read the file or not. Thus the formula $(A$ **can read** $f) \vee \neg(A$ **can read** $f)$ is not a validity in this system. This allows the possibility that even though A cannot read file f according to the current policy, it may be allowed that access in an extension of the policy.

In the applications mentioned above, the complexity of derivability is of prime importance, since a derivability check is often a fundamental component of more detailed security structures [6]. These systems are usually disjunction-free, with a **PTIME** derivability procedure [2, 7, 11]. But reasoning about disjunction is also important for security applications, even though it typically increases the complexity of the derivability problem (see [15], for example). In this paper, we explore the effect of disjunction on the complexity of various subsystems of intuitionistic logic.

The **PTIME** systems referred to above do not include full implication either. This is obvious, since it is well-known that the derivability problem for intuitionistic logic (and even its implication-only fragment) is **PSPACE**-complete.[1] In this context, [11] considers a restriction of full implication, the so-called *primal implication* which is defined by the following rule.

$$\frac{X \vdash \beta}{X \vdash \alpha \to \beta} \to$$

In this rule, we have the same set of antecedents (set of formulas to the left $\vdash$) both in the premise and conclusion, and this contributes to an efficient solution to the derivability problem.

We show that when we add disjunction to such efficient systems, derivability is in **co-NP**. The results are similar to those in [4], but while the results there are obtained via a translation to classical logic, we provide an explicit algorithm. Our focus is on the algorithm itself, which is a general procedure to lift a **PTIME** decision procedure for a logic to a **co-NP** procedure for the same logic with disjunction. We also provide a modification of the above procedure that runs in **PTIME** when we restrict the formulas on which disjunction elimination is applied in a proof.

We also show that we cannot do better than **co-NP** for the above logics. Subsystems involving disjunction are **co-NP**-hard with such minimal additions as the elimination rule for implication, or the introduction rule for conjunction. We also show that we get **co-NP**-hardness when we consider a system with rules for disjunction and the elimination rule for negation.

Related work As we mentioned earlier, application areas like security typically work with an intuitionistic system, and the complexity of derivability is important in such applications. In the study of cryptographic protocols, the cryptographic primitives are represented as rules in

[1] From now on, whenever we refer to the complexity of a logic, we implicitly mean the complexity of the derivability problem for it.

a proof system, following Dolev and Yao [8]. These logics are typically positive and conjunctive. The derivability problem for the basic Dolev-Yao system is in **PTIME** [16]. Other interesting non-classical conjunctions like **blind pairing** can make the problem hard when they interact distributively with the standard pairing operator [3].

The results reported in this paper are very close to work done in the realm of authorization logics, specifically primal infon logic and its extensions. It was shown that primal infon logic is in **PTIME** [2, 11] but adding disjunction makes the problem **co-NP**-complete [4]. Specifically, it was shown that a system with primal implication, conjunction, disjunction and $\bot$ is **co-NP**-hard, using a translation from classical logic. Our lower bound results can be seen as a refinement of the result in [4], as we show that disjunction with *any one* of these other connectives is already **co-NP**-hard. The upper bound results are also very similar to those in [4], but we provide an explicit algorithm while the results there are obtained via a translation to classical logic. Our procedures can be seen as a way of lifting **PTIME** decision procedures for *local theories* [7, 14] to **co-NP** procedures for the same logics with disjunction. More recently, the complexity of primal logic with disjunction was studied in further detail in [13], but the proofs are via semantic methods.

Another important area of study is the disjunction property and its effect on complexity. A system is said to have the disjunction property if it satisfies the following condition: whenever $X \vdash \alpha \vee \beta$ and X satisfies some extra conditions (for example, $\vee$ does not occur in any formula of X), then $X \vdash \alpha$ or $X \vdash \beta$. The disjunction property and its effect on decidability and complexity have been the subject of study for many years. For example, it has been proved that as long as any (propositional) logic that extends intuitionistic logic satisfies the disjunction property, derivability is **PSPACE**-hard, and otherwise it is in **co-NP** (see Chapter 18 of [5]). Various other papers also investigate extensions of intuitionistic logic with the disjunction property [9, 12, 17]. In contrast to these results, our paper considers *subsystems* of intuitionistic logic obtained by restricting implication. Further, in our paper, the focus is more on the *left disjunction property*: namely that $X, \alpha \vee \beta \vdash \delta$ iff $X, \alpha \vdash \delta$ and $X, \beta \vdash \delta$.

2 Preliminaries

Assume a countably infinite set of atomic propositions $\mathscr{P}$. The set of formulas Φ is given by

$$\alpha, \beta ::= p \mid \neg\alpha \mid \alpha \wedge \beta \mid \alpha \vee \beta \mid \alpha \to \beta$$

For a set of operators $\mathscr{O}$, we denote by $\Phi^{\mathscr{O}}$ the set of all formulas consisting only of the operators in $\mathscr{O}$. For example, $\Phi^{\{\vee\}}$ is the set of all formulas built only using the $\vee$ operator, $\Phi^{\{\vee,\wedge\}}$ is the set of all formulas built only using the $\vee$ and $\wedge$ operators, &c. For ease of notation, we ignore the braces and instead use $\Phi^{\vee}$, $\Phi^{\vee,\wedge}$, &c.

The set of **subformulas** of a formula α, denoted $\mathbf{sf}(\alpha)$, is defined to be the smallest set S such that: $\alpha \in S$; if $\neg\beta \in S$, $\beta \in S$; and if $\beta \wedge \gamma \in S$ or $\beta \vee \gamma \in S$ or $\beta \to \gamma \in S$, $\{\beta, \gamma\} \subseteq S$. For a set X of formulas, $\mathbf{sf}(X) = \bigcup_{\alpha \in X} \mathbf{sf}(\alpha)$.

$\dfrac{}{X, \alpha \vdash \alpha}\, ax$	
$\dfrac{X, \alpha \vdash \beta \quad X, \alpha \vdash \neg\beta}{X \vdash \neg\alpha}\, \neg i$	$\dfrac{X \vdash \beta \quad X \vdash \neg\beta}{X \vdash \alpha}\, \neg e$
$\dfrac{X \vdash \alpha \quad X \vdash \beta}{X \vdash \alpha \wedge \beta}\, \wedge i$	$\dfrac{X \vdash \alpha_0 \wedge \alpha_1}{X \vdash \alpha_j}\, \wedge e$
$\dfrac{X \vdash \alpha_j}{X \vdash \alpha_0 \vee \alpha_1}\, \vee i$	$\dfrac{X \vdash \alpha \vee \beta \quad X, \alpha \vdash \delta \quad X, \beta \vdash \delta}{X \vdash \delta}\, \vee e$
$\dfrac{X, \alpha \vdash \beta}{X \vdash \alpha \to \beta}\, \to i$	$\dfrac{X \vdash \alpha \to \beta \quad X \vdash \alpha}{X \vdash \beta}\, \to e$

Figure 1: The system **IL**

The logic is defined by the derivation system in Figure 1. By $X \vdash_{\mathbf{IL}} \alpha$, we mean that there is a derivation in **IL** of $X \vdash \alpha$. (For ease of notation, we drop the suffix and use $X \vdash \alpha$ to mean $X \vdash_{\mathbf{IL}} \alpha$, when there is no confusion.)

Definition 1 (Derivability problem) *Given X and α, is it the case that $X \vdash_{IL} \alpha$?*

Among the rules, ax, $\wedge e$ and $\to e$ are the *pure elimination rules*, $\neg e$, $\neg i$ and $\vee e$ are the *hybrid rules* and the rest are the *pure introduction rules*. A normal derivation is one where the major premise of every pure elimination rule and hybrid rule is the conclusion of a pure elimination rule. The following fundamental properties hold, and the proofs are standard in the proof theory literature.

Proposition 2

1. ***(Monotonicity)*** *If $X \vdash \alpha$ and $X \subseteq X'$, then $X' \vdash \alpha$.*
2. ***(Admissibility of Cut)*** *If $X \vdash \alpha$ and $X, \alpha \vdash \beta$, then $X \vdash \beta$.*

3. ***(Left Disjunction Property)*** *$X, \alpha \vee \beta \vdash \delta$ iff $X, \alpha \vdash \delta$ and $X, \beta \vdash \delta$.*

4. ***(Left Conjunction Property)*** *$X, \alpha \wedge \beta \vdash \delta$ iff $X, \alpha, \beta \vdash \delta$.*

Theorem 3 (Weak normalization) *If there is a derivation π of $X \vdash \alpha$ then there is a normal derivation $\bar{\omega}$ of $X \vdash \alpha$. Further, if a formula $\alpha \vee \beta$ occurs as the major premise of an instance of $\vee e$ in $\bar{\omega}$, it also occurs as the major premise of an instance of $\vee e$ in π.*

Theorem 4 (Subformula property) *Let π be a normal derivation with conclusion $X \vdash \alpha$ and last rule $\mathbf{r}$. Let $X' \vdash \beta$ occur in π. Then $X' \subseteq \mathit{sf}(X \cup \{\alpha\})$ and $\beta \in \mathit{sf}(X \cup \{\alpha\})$. Furthermore, if $\mathbf{r}$ is a pure elimination rule, then $X' \subseteq \mathit{sf}(X)$ and $\beta \in \mathit{sf}(X)$.*

3 The impact of disjunction: lower bounds

To gauge the effect of disjunction, we first consider disjunction in isolation, and show that the derivability problem is in **PTIME**. This indicates that the lower bound results that appear later in this section are a result of *interaction* between the various logical rules, rather than due to disjunction alone.

3.1 The disjunction-only fragment

Let $\mathbf{IL}[\vee]$ denote the fragment of **IL** consisting of the ax, $\vee i$ and $\vee e$ rules, and involving formulas of $\Phi^{\vee}$.

Theorem 5 *The derivability problem for $\mathbf{IL}[\vee]$ is in **PTIME**.*

Suppose $X = \{a_i^1 \vee a_i^2 \vee \cdots \vee a_i^k \mid 1 \leq i \leq n\}$ is a set of formulas from $\Phi^{\vee}$, with each $a_i^j \in \mathscr{P}$. Let $\beta = \beta^1 \vee \beta^2 \vee \cdots \vee \beta^k \in \Phi^{\vee}$, with each $\beta^j \in \mathscr{P}$. (Note that any input to the derivability problem of $\mathbf{IL}^{\vee}$ can be converted to the above form by choosing appropriate k, flattening the disjunctions, and repeating disjuncts). We now have the following claim.

Claim 6 *$X \vdash \beta$ iff there exists an $i \leq n$ such that $a_i^1 \vee a_i^2 \vee \cdots \vee a_i^k \vdash \beta$.*

Proof It is obvious that if $a_i^1 \vee a_i^2 \vee \cdots \vee a_i^k \vdash \beta$ then $X \vdash \beta$ (by Monotonicity).

For proving the other direction, suppose (towards a contradiction) $X \vdash \beta$, but there is no i such that $a_i^1 \vee a_i^2 \vee \cdots \vee a_i^k \vdash \beta$. In particular, from the Left Disjunction Property, for every i, some $a_i^{j_i} \nvdash \beta$. Without loss of generality, assume that $j_i = 1$ for every i. Thus we have that $a_1^1 \nvdash \beta$, $a_2^1 \nvdash \beta$, …, $a_n^1 \nvdash \beta$.

Now, since $X \vdash \beta$ and $a_i^1 \vdash a_i^1 \vee \cdots \vee a_i^k$ for each $i \leq n$, it follows by Admissibility of Cut that $a_1^1, \ldots, a_n^1 \vdash \beta$ (and there is a normal proof π with that conclusion). Since all the a_i^1s are atomic

propositions, the only rules that can appear in π are ax and $\vee i$. Therefore, at some point, one of the a_i^1s must have contributed to a β^j via an ax rule. However, this gives us $a_i^1 \vdash \beta$ (by deriving β^j and then applying $\vee i$), which is a contradiction. Thus we have the required claim. $\dashv$

Given this claim, we know that it is enough to see if a particular formula on the left (say a_i) derives β. In particular, from the Left Disjunction Property, we get that every disjunct in a_i needs to derive β. Therefore, the derivability problem is equivalent to checking if there is a formula in X all of whose disjuncts occur in β, and thus we obtain the required **PTIME** procedure.

3.2 Disjunction and conjunction

We have now confirmed that the $\vee$-only fragment is in **PTIME**. It is also known that some other fragments (for example the fragment consisting of primal implication, conjunction, and a restricted negation) give rise to **PTIME** logics. However, we obtain the following result for the logic with conjunction and disjunction.

Let **IL**$[\vee, \wedge]$ denote the fragment of **IL** consisting of the ax, $\vee i$, $\vee e$, $\wedge i$ and $\wedge e$ rules, and involving formulas of $\Phi^{\vee,\wedge}$.

Theorem 7 *The derivability problem for* ***IL***$[\vee, \wedge]$ *is* ***co-NP****-hard.*

The hardness result is obtained by reducing the validity problem for boolean formulas to the derivability problem for **IL**$[\vee, \wedge]$. In fact, it suffices to consider the validity problem for boolean formulas in disjunctive normal form. We show how to define for each DNF formula ϕ a set of **IL**$[\vee, \wedge]$-formulas S_ϕ and an **IL**$[\vee, \wedge]$-formula $\overline{\phi}$ such that $S_\phi \vdash \overline{\phi}$ iff ϕ is a tautology.

Let $\{x_1, x_2, \ldots\}$ be the set of all boolean variables. For each boolean variable x_i, fix two distinct atomic propositions $p_i, q_i \in \mathscr{P}$. We define $\overline{\phi}$ as follows, by induction.

- $\overline{x_i} = p_i$
- $\overline{\neg x_i} = q_i$
- $\overline{\phi \vee \psi} = \overline{\phi} \vee \overline{\psi}$
- $\overline{\phi \wedge \psi} = \overline{\phi} \wedge \overline{\psi}$

Let $\mathrm{Voc}(\phi)$, the set of all boolean variables occurring in ϕ, be $\{x_1, \ldots, x_n\}$. Then

$$S_\phi = \{p_1 \vee q_1, \ldots, p_n \vee q_n\}.$$

Lemma 8 *$S_\phi \vdash \overline{\phi}$ iff ϕ is a tautology.*

Proof Recall that a propositional valuation v over a set of variables $\mathscr{V}$ is just a subset of $\mathscr{V}$ – those variables that are set to true by v.

For a valuation $v \subseteq \{x_1, \ldots, x_n\}$, define $S_v = \{p_i \mid x_i \in v\} \cup \{q_i \mid x_i \notin v\}$.

By repeated appeal to the Left Disjunction Property, it is easy to see that $S_\phi \vdash \overline{\phi}$ iff for all valuations v over $\{x_1, \ldots, x_n\}$, $S_v \vdash \overline{\phi}$. We now show that $S_v \vdash \overline{\phi}$ iff $v \models \phi$. The statement of the lemma follows immediately from this.

- We first show by induction on $\psi \in \mathbf{sf}(\phi)$ that whenever $v \models \psi$, it is the case that $S_v \vdash \overline{\psi}$.
  - If $\psi = x_i$ or $\psi = \neg x_i$, then $S_v \vdash \overline{\psi}$ follows from the ax rule.
  - If $\psi = \psi_1 \wedge \psi_2$, then it is the case that $v \models \psi_1$ and $v \models \psi_2$. By induction hypothesis, $S_v \vdash \overline{\psi_1}$ and $S_v \vdash \overline{\psi_2}$. Hence, by using $\wedge i$, it follows that $S_v \vdash \overline{\psi_1 \wedge \psi_2}$.
  - If $\psi = \psi_1 \vee \psi_2$, then it is the case that either $v \models \psi_1$ or $v \models \psi_2$. By induction hypothesis, $S_v \vdash \overline{\psi_1}$ or $S_v \vdash \overline{\psi_2}$. In either case, by using $\vee i$, it follows that $S_v \vdash \overline{\psi_1 \vee \psi_2}$.
- We now show that if $S_v \vdash \overline{\phi}$, then $v \models \phi$. Suppose π is a normal proof of $S_v \vdash \phi$, and that there is an occurrence of the $\wedge e$ rule or $\vee e$ rule in π with major premise $S' \vdash \gamma$. We denote by ϖ this subproof with conclusion $S' \vdash \gamma$. Note that ϖ ends in a pure elimination rule, since π is normal and every pure elimination rule and hybrid rule has as its major premise the conclusion of a pure elimination rule. By Theorem 4, we see that $S' \subseteq \mathbf{sf}(S_v) = S_v$, and $\gamma \in \mathbf{sf}(S')$. But γ is of the form $\alpha \vee \beta$ or $\alpha \wedge \beta$, and this contradicts the fact that $S_v \subseteq \mathscr{P}$. Thus π consists of only the ax, $\wedge i$ and $\vee i$ rules. We now show by induction that for all subproofs π' of π with conclusion $S_v \vdash \overline{\psi}$, $v \models \psi$.
  - Suppose the last rule of π' is ax. Then $\overline{\psi} \in S_v$, and for some $i \leq n$, $\psi = x_i$ or $\psi = \neg x_i$. It can be easily seen that $v \models \psi$ (by the definition of S_v).
  - Suppose the last rule of π' is $\wedge i$. Then $\overline{\psi} = \overline{\psi_1} \wedge \overline{\psi_2}$, and $S_v \vdash \overline{\psi_1}$ and $S_v \vdash \overline{\psi_2}$. Thus, by induction hypothesis, $v \models \psi_1$ and $v \models \psi_2$. Therefore $v \models \psi$.
  - Suppose the last rule of π' is $\vee i$. Then $\overline{\psi} = \overline{\psi_1} \vee \overline{\psi_2}$, and either $S_v \vdash \overline{\psi_1}$ or $S_v \vdash \overline{\psi_2}$. Thus, by induction hypothesis, either $v \models \psi_1$ or $v \models \psi_2$. Therefore $v \models \psi$. ⊣

3.3 Disjunction and implication elimination

We now consider another minimal system, $\mathbf{IL}[\vee, \rightarrow e]$, consisting of the rules ax, $\vee i$, $\vee e$ and $\rightarrow e$ and involving formulas from $\Phi^{\vee,\rightarrow}$, and prove the following result.

Theorem 9 *The derivability problem for* $\mathbf{IL}[\vee, \rightarrow e]$ *is* ***co-NP****-hard.*

The proof is by reduction from the validity problem for 3-DNF, as detailed below.

Let ϕ be a 3-DNF formula with each clause having exactly three literals. Let $\mathrm{Voc}(\phi)$, the set of all boolean variables occurring in ϕ, be $\{x_1, \ldots, x_n\}$. We define $indx(\phi)$ to be the set $\{1, 1', \ldots, n, n'\}$, where $(i')' = i$ for any $i \in indx(\phi)$. For $i \leq n$, we define $\ell(i) = x_i$ and $\ell(i') = \neg x_i$.

We define the following sets.

$$S_\phi := \{p_a \vee p_{a'} \mid a \in indx(\phi)\}.$$

$$T_\phi := \{p_a \to p_b \to p_c \to p_{abc} \mid a, b, c \in indx(\phi)\}.$$

We define $\overline{\phi}$ as follows:

$$\overline{\phi} := \bigvee \{p_{abc} \mid \ell(a) \wedge \ell(b) \wedge \ell(c) \text{ is a disjunct of } \phi\}.$$

For each valuation $v \subseteq \{x_1, \ldots, x_n\}$, define S_v to be

$$\{p_i \mid x_i \in v\} \cup \{p_{i'} \mid x_i \notin v\}.$$

Lemma 10 *$S_\phi, T_\phi \vdash \overline{\phi}$ iff ϕ is a tautology.*

Proof By repeated appeal to the Left Disjunction Property, it is easy to see that $S_\phi, T_\phi \vdash \overline{\phi}$ iff $S_v, T_\phi \vdash \overline{\phi}$ for all valuations v over $\{x_1, \ldots, x_n\}$. We now show that for all such valuations, $v \models \phi$ iff $S_v, T_\phi \vdash \overline{\phi}$.

Let π be a normal proof of $S_v, T_\phi \vdash \overline{\phi}$. The last rule of π has to be $\vee$i, since if π ends in an elimination rule, from the Subformula Property it follows that a disjunction is a subformula of $S_v \cup T_\phi$, which is not the case. Repeating this argument, we see that there is a subproof of π with conclusion $S_v, T_\phi \vdash p_{abc}$ for some disjunct $\ell(a) \wedge \ell(b) \wedge \ell(c)$ of ϕ. We now show that for any valuation v, $S_v, T_\phi \vdash p_{abc}$ iff $v \models \ell(a) \wedge \ell(b) \wedge \ell(c)$.

If $v \models \ell(a) \wedge \ell(b) \wedge \ell(c)$, then we have $p_a, p_b, p_c \in S_v$ (from the definition of S_v), and therefore by applying the $\to$e rule to $p_a \to p_b \to p_c \to p_{abc}$ in T_ϕ, we have $S_v, T_\phi \vdash p_{abc}$. In the other direction, suppose we have a normal proof π of $S_v, T_\phi \vdash p_{abc}$. By examining S_v and T_ϕ, we see that only $p_a \to p_b \to p_c \to p_{abc}$ mentions p_{abc}. So it is clear that p_c must be derivable from S_v, T_ϕ, and the last rule of π must be $\to$e, applied to $p_c \to p_{abc}$. Now in order for this formula to be derivable, p_b must be derivable, and similarly p_a must be derivable. Since p_a, p_b and p_c can only be obtained by ax, it must be that $p_a, p_b, p_c \in S_v$ and therefore $v \models \ell(a) \wedge \ell(b) \wedge \ell(c)$.

Thus we have that $S_v, T_\phi \vdash p_{abc}$ iff $v \models \ell(a) \wedge \ell(b) \wedge \ell(c)$, and the required claim follows. $\dashv$

4 Upper bounds

We now show that a system with conjunction, disjunction, primal implication, and a restricted version of negation (allowing only negation elimination, but not negation introduction) is in **co-NP**. We first give a **PTIME** procedure for the logic without disjunction elimination and then lift it to a **co-NP** procedure which accounts for disjunction elimination.[2]

Fix a set of formulas X_0 and a formula a_0 for the rest of the section. Let $\mathbf{sf} = \mathbf{sf}(X_0 \cup \{a_0\})$. Let $N = |\mathbf{sf}|$.

Definition 11 *For any $X \subseteq \mathbf{sf}$:*

- $\mathit{derive}(X) = \{a \in \mathbf{sf} \mid X \vdash a\}$.
- $\mathit{derive}'(X) = \{a \in \mathbf{sf} \mid$ *there is a proof of* $X \vdash a$ *not using the* $\vee e$ *rule*$\}$.

The following properties of *derive* and *derive′* are immediate.

- $X \subseteq \mathit{derive}'(X) \subseteq \mathit{derive}(X)$.
- $\mathit{derive}(X) = \mathit{derive}'(\mathit{derive}(X)) = \mathit{derive}(\mathit{derive}(X))$ (by Admissibility of Cut).
- $\mathit{derive}'(X) = \mathit{derive}'(\mathit{derive}'(X))$ (by Admissibility of Cut).
- If X is of the form $\mathit{derive}'(Y)$, then $\mathit{derive}'(X) = X$. If $X = \mathit{derive}(Y)$, then $\mathit{derive}(X) = X$.

4.1 A PTIME procedure for *derive′*

In the absence of $\vee e$, there is no branching during proof search. Hence we can compute $\mathit{derive}'(Y)$ bottom-up in **PTIME**, as detailed below in Algorithm 1.

For $Y \subseteq \mathbf{sf}$, we define $\mathit{onestep}(Y) \subseteq \mathbf{sf}$ to be the set

$$\{a \in \mathbf{sf} \mid a \text{ is the conclusion of a rule } \mathbf{r} \text{ (other than } \vee e) \text{ with premises } Z \subseteq Y\}.$$

Two important observations about $\mathit{onestep}(Y)$.

- $Y \subseteq \mathit{onestep}(Y)$, because of the rule *ax*.
- $\mathit{onestep}(Y)$ is computable in time $O(N^2)$, where $N = |\mathbf{sf}|$. This is because in all the rules other than $\vee e$, the antecedents (formulas occurring to the left of $\vdash$) in the premises are the same as the antecedents in the conclusion. Thus we need to consider only consequents (the formulas to the right of $\vdash$) in a proof. This means that we only need to consider all pairs of formulas in Y to compute $\mathit{onestep}(Y)$.

Algorithm 1 Algorithm to compute $\mathit{derive}'(X)$, for $X \subseteq \mathbf{sf}$

$Y \leftarrow \emptyset$;
$Y' \leftarrow X$;
while $(Y \neq Y')$ do
    $Y \leftarrow Y'$;
    $Y' \leftarrow \mathit{onestep}(Y)$;
end while
return Y.

Since $|\mathbf{sf}| = N$ and Y increases monotonically, the while loop runs only for N iterations. Thus $\mathit{derive}'(X)$ is computable in time $O(N^3)$.

4.2 A co-NP procedure for *derive*

Algorithm 2 checks if $X_0 \nvdash \alpha_0$. It uses the notion of a *down-closed set*. A set X of formulas is *down-closed* if it satisfies the following two conditions:

- $\mathit{derive}'(X) \subseteq X$.
- whenever $\alpha \vee \beta \in X$, then either $\alpha \in X$ or $\beta \in X$.

Y is said to be a *down-closure* of X if Y is down-closed and $X \subseteq Y$.

Algorithm 2 Algorithm to check if $X_0 \nvdash \alpha_0$

$Y \leftarrow \mathit{derive}'(X_0)$;
while (Y is not down-closed) do
    guess a formula $\beta_0 \vee \beta_1 \in Y$ such that $\beta_0 \notin Y$ and $\beta_1 \notin Y$;
    guess $i \in \{0,1\}$;
    $Y \leftarrow \mathit{derive}'(Y \cup \{\beta_i\})$;
end while
Return "Yes" if $\alpha_0 \notin Y$, and "No" otherwise.

In Algorithm 2, it is an invariant that $Y = \mathit{derive}'(Z)$ for some Z and hence $\mathit{derive}'(Y) \subseteq Y$. Thus when Y is not down-closed, there exists $\beta_0 \vee \beta_1 \in Y$ such that neither β_0 nor β_1 is in Y.

[2] It is important to note that we consider only the negation elimination rule. The algorithms in this section do not work in the presence of the $\neg i$ rule. Nor do we know of a straightforward modification to handle the $\neg i$ rule. It is not easy to say without further study whether the complexity stays the same or increases, either.

The algorithm guesses a down-closure Y of X_0 such that $\alpha_0 \notin Y$. The following theorem guarantees that one can successfully guess such a Y iff $X_0 \nvdash \alpha_0$. This ensures the correctness of the algorithm.

Theorem 12 *For any X and α (with $X \cup \{\alpha\} \subseteq \mathbf{sf}$), $X \vdash \alpha$ iff $\alpha \in Y$ for every down-closure Y of X.*

This theorem is a consequence of the following three lemmas. But first we need a general claim related to the Left Disjunction Property.

Claim 13 *Suppose $\phi_0 \vee \phi_1 \in Z$ and $i \in \{0,1\}$. Then $Z \setminus \{\phi_0 \vee \phi_1\}, \phi_i \vdash \theta$ iff $Z, \phi_i \vdash \theta$.*

Lemma 14 *For any X and α (with $X \cup \{\alpha\} \subseteq \mathbf{sf}$), $X \vdash \alpha$ iff $Y \vdash \alpha$ for every down-closure Y of X.*

Proof Suppose $X \vdash \alpha$ and Y is a down-closure of X. Then $X \subseteq Y$ and hence it is immediate that $Y \vdash \alpha$.

Suppose now that $X \nvdash \alpha$. We show that there is a sequence $Y_0 \subsetneq Y_1 \subsetneq \cdots \subsetneq Y_n \subseteq \mathbf{sf}$ of sets such that

- $X \subseteq Y_0$,
- Y_n is down-closed,
- for all $i \leq n$, $\mathit{derive}'(Y_i) \subseteq Y_i$, and
- for all $i \leq n$, $Y_i \nvdash \alpha$.

The sequence is constructed by induction. Y_0 is defined to be $\mathit{derive}'(X)$. Since $X \nvdash \alpha$, it follows that $Y_0 \nvdash \alpha$. Suppose Y_k has been defined for some $k \geq 0$ such that $Y_k \nvdash \alpha$. If Y_k is down-closed, we are done. Otherwise, since $\mathit{derive}'(Y_k) \subseteq Y_k$, there is a $\beta_0 \vee \beta_1 \in Y_k$ such that $\beta_0 \notin Y_k$ and $\beta_1 \notin Y_k$. Since $Y_k \nvdash \alpha$, it follows by the Left Disjunction property that $Y_k \setminus \{\beta_0 \vee, \beta_1\}, \beta_i \nvdash \alpha$ for some $i \in \{0,1\}$. By Claim 13 it follows that $Y_k, \beta_i \nvdash \alpha$ for some $i \in \{0,1\}$.

$$Y_{k+1} = \begin{cases} \mathit{derive}'(Y_k \cup \{\beta_0\}) & \text{if } Y_k, \beta_0 \nvdash \alpha \\ \mathit{derive}'(Y_k \cup \{\beta_1\}) & \text{otherwise} \end{cases}$$

Clearly $Y_k \subsetneq Y_{k+1}$ and $\mathit{derive}'(Y_{k+1}) = Y_{k+1}$. Assume that $Y_{k+1} = \mathit{derive}'(Y_k \cup \{\beta_0\})$, without loss of generality. By construction, $Y_k \cup \{\beta_0\} \nvdash \alpha$. Now suppose $Y_{k+1} \vdash \alpha$. Then, since $Y_k \cup \{\beta_0\} \vdash \phi$ for every $\phi \in Y_{k+1}$, it would follow by Admissibilty of Cut that $Y_k \cup \{\beta_0\} \vdash \alpha$, which is a contradiction. Thus $Y_{k+1} \nvdash \alpha$. Thus we can always extend the sequence as desired.

Further, the Y_i's are strictly increasing, and are all subsets of $\mathbf{sf}$. Thus $n \leq |\mathbf{sf}|$ and the above construction terminates. Y_n is a down-closure of X that does not derive α. ⊣

Lemma 15 *Let π be a proof of $X \vdash a$ with at least one occurrence of the $\vee e$ rule. Then there is an occurrence of $\vee e$ in π with major premise $X \vdash \phi \vee \psi$ such that $\phi \vee \psi \in derive'(X)$.*

Proof In any proof of the form

$$\dfrac{\begin{array}{ccc}\pi_1 & \pi_2 & \pi_3 \\ \vdots & \vdots & \vdots \\ X_1 \vdash a_1 & X_2 \vdash a_2 & X_3 \vdash a_3\end{array}}{Y \vdash \delta}\ \mathbf{r}$$

we say that any rule in π_1 is to the left of **r**, **r** is to the left of any rule in π_2, and any rule in π_2 is to the left of any rule in π_3.

Now consider the leftmost occurrence of $\vee e$ in π. It is the last rule of a subproof π' of π which looks as follows.

$$\dfrac{\begin{array}{ccc}\pi'_1 & \pi'_2 & \pi'_3 \\ \vdots & \vdots & \vdots \\ X' \vdash \phi \vee \psi & X', \phi \vdash \theta & X', \psi \vdash \theta\end{array}}{X' \vdash \theta}\ \vee e$$

Since this is the leftmost occurrence of $\vee e$, there is no occurrence of $\vee e$ in π'_1. Further, if $X' \neq X$, it means that π' is part of the proof of a minor premise of some other $\vee e$ rule in π. But that contradicts the fact that π' ends in the leftmost $\vee e$ in π. Thus $X' = X$, and π'_1 witnesses the fact that $\phi \vee \psi \in derive'(X)$. ⊣

Lemma 16 *For a down-closed Y, $Y \vdash a$ iff $a \in Y$.*

Proof If $a \in Y$, then it is obvious that $Y \vdash a$.

In the other direction, suppose $Y \vdash a$ via a proof π with k instances of $\vee e$. We prove the required claim by induction on k.

In the base case, $k = 0$, and $a \in derive'(Y)$. Since Y is down-closed, $derive'(Y) \subseteq Y$ and we have $a \in Y$.

In the induction step, suppose there is an instance of $\vee e$ in the proof of $Y \vdash a$. By Lemma 15, we know that there is at least one occurrence of $\vee e$ (say $Y \vdash \delta$) with major premise $Y \vdash \phi \vee \psi$ such that $\phi \vee \psi \in derive'(Y) \subseteq Y$, which looks as follows.

$$\dfrac{\begin{array}{ccc}\pi_1 & \pi_2 & \pi_3 \\ \vdots & \vdots & \vdots \\ Y \vdash \phi \vee \psi & Y, \phi \vdash \delta & Y, \psi \vdash \delta\end{array}}{Y \vdash \delta}\ \vee e$$

Thus we have $\phi \vee \psi \in Y$. Since Y is down-closed either $\phi \in Y$ or $\psi \in Y$. Suppose, without loss of generality, that $\phi \in Y$. Now consider π_2. Since $\phi \in Y$, we know that $Y \cup \{\phi\} = Y$, and we can replace the big proof of $Y \vdash \delta$ by π_2, thereby reducing the number of instances of $\vee e$ in the proof of $Y \vdash a$. By induction hypothesis, $a \in Y$, and the lemma follows. ⊣

Running time We now analyze the running time of Algorithm 2. Since Y strictly increases with each iteration of the loop, there are at most $N = |\mathbf{sf}|$ iterations of the loop. In each iteration, we test whether Y is down-closed, which amounts to checking whether there is some $\beta_0 \vee \beta_1 \in Y$ such that neither β_0 nor β_1 is in Y. This check takes $O(N)$ time. We also compute $\mathit{derive}'(Y)$ in each iteration, which takes time $O(N^3)$. Thus the overall running time is $O(N^4)$. This can be improved to $O(N^2)$ by using a linear-time algorithm for derive' like the one given in [11].

4.3 Bounding resources

As is evident from the lower bound proofs, disjunction elimination contributes heavily to the complexity of the derivation problem. Thus the use of the $\vee e$ rule is an important resource. It makes sense to bound the use of this resource and explore its effect on complexity. In particular, we show that if we bound the set of formulas on which to perform disjunction elimination, we get a procedure whose running time is polynomial in the input size, though exponential in the number of disjunction eliminations allowed. The following definition makes this notion precise.

Definition 17 *Let A be a set of disjunctive formulas. We define a proof of α from X using A (denoted $X \vdash_A \alpha$) as a proof where any $\vee e$ rules are applied only to formulas which appear in A.*

Recall that we have fixed a set $\mathbf{sf}$ of size N, and that we consider the derivability of $X \vdash \alpha$ where $\mathbf{sf}(X \cup \{\alpha\}) \subseteq \mathbf{sf}$. We define $\mathit{derive}_A(X)$ to be $\{\beta \in \mathbf{sf} \mid X \vdash_A \beta\}$. Note that $\mathit{derive}_\varnothing(X)$ is $\mathit{derive}'(X)$. The check for $X \vdash_A \alpha$ is done by using Algorithm 3 to compute $\mathit{derive}_A(X)$ and then testing whether $\alpha \in \mathit{derive}_A(X)$. (For the purposes of the algorithm, we assume that the set A is equipped with a linear order, so we can refer to the least formula in any subset of A.)

Algorithm 3 Algorithm to compute $\mathit{derive}_A(X)$

```
1: function f(A, X)
2:     Y ← derive'(X);
3:     if A ∩ Y = ∅ then
4:         return Y;
5:     else
6:         A' ← A \ {α ∨ β}, where α ∨ β is the least formula in A ∩ Y;
7:         return f(A', Y ∪ {α}) ∩ f(A', Y ∪ {β});
8:     end if
9: end function
```

In order to prove the correctness of the above algorithm, we require the following claim.

Claim 18 *Suppose A is a set of disjunctions and $\alpha \vee \beta \in A$. Let $A' = A \setminus \{\alpha \vee \beta\}$. Then the following hold:*

- *If $X \vdash_A \gamma$ then $X, \alpha \vdash_{A'} \gamma$ and $X, \beta \vdash_{A'} \gamma$.*
- *If $X \vdash_A \alpha \vee \beta$, $X, \alpha \vdash_{A'} \gamma$ and $X, \beta \vdash_{A'} \gamma$, then $X \vdash_A \gamma$.*

Proof

- Suppose $X \vdash_A \gamma$. Then by monotonicity, we obtain a proof π of $X, \alpha \vdash \gamma$, such that the major premise of every instance of the $\vee e$ rule in π is in A. Note that for every sequent $X' \vdash \delta$ in π, $\alpha \in X'$. Consider any subproof π' of π whose conclusion is $X' \vdash \delta$ and last rule is $\vee e$ with major premise $\alpha \vee \beta$ (if there is no such subproof, then π witnesses the fact that $X, \alpha \vdash_{A'} \gamma$). π' has the following form.

$$\dfrac{\begin{array}{ccc} \vdots\,\pi'_1 & \vdots\,\pi'_2 & \vdots\,\pi'_3 \\ X' \vdash \alpha \vee \beta & X', \alpha \vdash \delta & X', \beta \vdash \delta \end{array}}{X' \vdash \delta} \vee e$$

  But observe that since $\alpha \in X'$, $X' \cup \{\alpha\} = X'$. Thus π'_2 is itself a proof of $X' \vdash \delta$. We can replace π' by π'_2, thereby removing at least one instance of the $\vee e$ rule involving $\alpha \vee \beta$ in π. Repeating this, we obtain that $X, \alpha \vdash_{A'} \gamma$. A similar reasoning gives us the result for $X, \beta \vdash_{A'} \gamma$.

- Performing an or-elimination on $\alpha \vee \beta$ using the given proofs of $X, \alpha \vdash_{A'} \gamma$ and $X, \beta \vdash_{A'} \gamma$ and $X \vdash_A \alpha \vee \beta$ for premises gives us the required result of $X \vdash_A \gamma$. ⊣

Lemma 19 (Correctness of Algorithm 3) *For all X and A,*

$$\mathit{derive}_A(X) = f(A, X).$$

Proof The proof is by induction on the size of A. The base case is when $A = \varnothing$, when clearly the procedure f returns $\mathit{derive}'(X)$.

For the induction case, suppose $X \vdash_A \delta$, and let $Y = \mathit{derive}'(X)$. Consider a normal proof π witnessing $X \vdash_A \delta$ and assume without loss of generality that there is at least one instance of $\vee e$ in π. From Lemma 15, we see that there is an instance of $\vee e$ in π with major premise $X \vdash \phi \vee \psi$, where $\phi \vee \psi \in \mathit{derive}'(X)$. Thus $A \cap Y \neq \varnothing$. Let $\alpha \vee \beta$ be the least formula in $A \cap Y$. Now since $X \subseteq Y$, $Y \vdash_A \delta$. Furthermore, $\alpha \vee \beta \in Y$. Hence, by Claim 18, $Y, \alpha \vdash_{A'} \delta$ and $Y, \beta \vdash_{A'} \delta$, where $A' = A \setminus \{\alpha \vee \beta\}$. Since A' is of smaller size than A, by the induction hypothesis, $\mathit{derive}_{A'}(Z) = f(A', Z)$ for any Z.

Thus $\delta \in f(A', Y \cup \{\alpha\}) \cap f(A', Y \cup \{\beta\})$. It follows from the definition of f that $\delta \in f(A, X)$. Thus $derive_A(X) \subseteq f(A, X)$.

On the other hand suppose $\delta \in f(A, X)$, and assume without loss of generality that $A \cap Y \neq \emptyset$, where $Y = derive'(X)$. Letting $\alpha \vee \beta$ be the least formula in $A \cap Y$ and $A' = A \setminus \{\alpha \vee \beta\}$, it is clear that $\delta \in f(A', Y \cup \{\alpha\}) \cap f(A', Y \cup \{\beta\})$ from the definition of f. Since A' is of smaller size than A, it follows from the induction hypothesis that $Y, \alpha \vdash_{A'} \delta$ and $Y, \beta \vdash_{A'} \delta$. Since $Y = derive'(X)$, it is the case that $X \vdash' \gamma$ for every $\gamma \in Y$. Thus we can appeal to the admissibility of cut to conclude that $X, \alpha \vdash_{A'} \delta$ and $X, \beta \vdash_{A'} \delta$. It follows from Claim 18 that $X \vdash_A \delta$. Thus $f(A, X) \subseteq derive_A(X)$. ⊣

Theorem 20 *If $|A| = k$, then $derive_A(X)$ is computable in time $O(2^k \cdot N)$.*

Proof There are at most 2^k recursive calls to f, and in each invocation we make one call to $derive'$, which takes $O(N)$ time. Thus the overall running time is $O(2^k \cdot N)$. ⊣

5 Discussion

To summarize our results, we have proved that **IL**$[\vee]$ is in **PTIME**, while even minimal extensions like $[\vee, \wedge]$, $[\vee, \to e]$ and $[\vee, \bot]$ are **co-NP**-hard. On the other hand, even the system with conjunction, disjunction, primal implication and negation elimination is in **co-NP**.

Of the two rules for negation, $\neg e$ does not modify the assumptions in the sequents, whereas $\neg i$ discharges the assumption α while concluding $\neg\alpha$. There does not appear to be a straightforward adaptation of either Algorithm 1 or Algorithm 2 to handle $\neg i$. As we mentioned earlier, it is not clear whether the complexity of the logic changes either. Note that [4] considers a fragment with rules for primal implication, disjunction, and a $\bot$ operator. While full implication and $\bot$ can express full negation, primal implication and $\bot$ can only capture the effect of the $\neg e$ rule, not the $\neg i$ rule. So the complexity of the fragment involving primal implication, conjunction, disjunction and "full" negation is still open. We leave this for future study.

We can also consider adding $\Box$-like modalities to the $[\wedge, \vee]$ fragment of our logic. This system is in **co-NP**, and the algorithm proceeds along similar lines to the one in [15]. On the other hand, if we add modalities to a logic with implication (even primal implication), the system is **PSPACE**-complete [4].

There are several interesting ways in which to take this work forward. It is worthwhile to look for logics with restricted forms of disjunction that are efficiently solvable. We also need to identify scenarios in which it suffices to consider a bounded number of disjunction eliminations, wherein our **PTIME** algorithm in Section 4.3 is applicable.

References

[1] Martin Abadi, Michael Burrows, Butler Lampson, and Gordon Plotkin. A calculus for access control in distributed systems. *ACM Transactions on Programming Languages and Systems*, 15(4):706–734, 1993.

[2] A. Baskar, Prasad Naldurg, K. R. Raghavendra, and S. P. Suresh. Primal Infon Logic: Derivability in Polynomial Time. In *FSTTCS 2013*, volume 24 of *LIPIcs*, pages 163–174, 2013.

[3] A. Baskar, R. Ramanujam, and S.P. Suresh. A DEXPTIME-complete Dolev-Yao theory with distributive encryption. In *Proceedings of MFCS 2010*, volume 6281 of *LNCS*, pages 102–113. 2010.

[4] Lev D. Beklemishev and Yuri Gurevich. Propositional primal logic with disjunction. *Journal of Logic and Computation*, 24(1):257–282, 2014.

[5] Alexander Chagrov and Michael Zakharyaschev. *Modal Logic, volume 35 of Oxford Logic Guides*. Clarendon Press, Oxford, 1997.

[6] Hubert Comon-Lundh and Vitaly Shmatikov. Intruder Deductions, Constraint Solving and Insecurity Decisions in Presence of Exclusive or. In *Proceedings of LICS 2003*, pages 271–280, 2003.

[7] Hubert Comon-Lundh and Ralf Treinen. Easy intruder deductions. In *Verification: Theory and Practice*, volume 2772 of *LNCS*, pages 225–242, 2003.

[8] Danny Dolev and Andrew Yao. On the Security of public-key protocols. *IEEE Transactions on Information Theory*, 29:198–208, 1983.

[9] Dov M. Gabbay and Dick H J de Jongh. A sequence of decidable finitely axiomatizable intermediate logics with the disjunction property. *The Journal of Symbolic Logic*, 39(01):67–78, 1974.

[10] Yuri Gurevich and Itay Neeman. DKAL: Distributed-knowledge authorization language. In *Proceedings of 21st IEEE CSF Symposium*, pages 149–162, 2008.

[11] Yuri Gurevich and Itay Neeman. Logic of infons: The propositional case. *ACM Transactions of Computational Logic*, 12(2):9, 2011.

[12] Hidenori Kurokawa. Hypersequent calculi for intuitionistic logic with classical atoms. *Annals of Pure and Applied Logic*, 161(3):427–446, 2009.

[13] Marco Magirius, Martin Mundhenk, and Raphaela Palenta. The complexity of primal logic with disjunction. *Information Processing Letters*, 115(5):536–542, 2015.

[14] David A McAllester. Automatic recognition of tractability in inference relations. *Journal of the ACM (JACM)*, 40(2):284–303, 1993.

[15] R. Ramanujam, Vaishnavi Sundararajan, and S.P. Suresh. Extending Dolev-Yao with Assertions. In *Proceedings of ICISS 2014*, volume 8880 of *LNCS*, pages 50–68, 2014.

[16] Michaël Rusinowitch and Mathieu Turuani. Protocol Insecurity with Finite Number of Sessions and Composed Keys is NP-complete. *Theoretical Computer Science*, 299:451–475, 2003.

[17] Alexander Sakharov. Median logic. Technical report, St. Petersberg Mathematical Society. `http://www.mathsoc.spb.ru/preprint/2004/index.html`, 2004.