A Logical Analysis of an Information Filtering Architecture Based on Epistemic Trust Inference (Extended Version)

Xu Li[1], **Leendert van der Torre**[1], **Liuwen Yu**[1]
[1]University of Luxembourg
2, place de l'Université, L-4365 Esch-sur-Alzette, Luxembourg
xu.li@uni.lu, leon.vandertorre@uni.lu, liuwen.yu@uni.lu

Abstract

In agent theory, epistemic trust is used to infer beliefs, for example by filtering out the information the agent receives from untrustworthy agents. Moreover, trust itself can be inferred from other information. We introduce a simple information filtering architecture that clearly distinguishes the relation between the two kinds of inference. Moreover, we provide a logical analysis of the architecture, based on a new family of input/output logics, and we explore information filtering and belief manipulation within this formal framework. Our key finding is that due to this architecture, some of the widely debated logical rules for trust inference are redundant with respect to information-filtering mechanisms and some other are redundant with respect to belief manipulation.

1 Introduction

In agent theory, epistemic trust (Liau 2003) is a relation between two agents, and it is used to infer beliefs. We call the inverse of the trust relation belief manipulation (Hunter, Schwarzentruber, and Tsang 2017; Eiter, Hunter, and Schwarzentruber 2021). This means that if agent X trusts agent Y with respect to the belief in p, agent Y can manipulate the belief of X in p.

The trust relation can be conceptualized as two related inference processes. On the one hand, trust itself can be inferred from some information. On the other hand, the inferred trust can be used to infer beliefs by filtering out the information the agent receives from untrustworthy agents.

In this paper we are interested in the combination of these two kinds of inference from an architectural and logical point of view.

1. How to define an information filtering architecture that clearly distinguishes the relation between the two kinds of inference, and in which belief manipulation is the inverse of epistemic trust?
2. How to provide a logical analysis of the architecture, in which optional properties of the two inferences are represented by optional logical rules?
3. Which of the logical rules are redundant with respect to information-filtering mechanisms and belief manipulation, respectively?

We explore information filtering and belief manipulation within the general formal approach of input/output logics (Makinson and Van Der Torre 2000), because information filtering has the general property that the input is not included in the output. In fact, information filtering has the inverse property that the output is included in the input, and we therefore introduce a new family of input/output logics satisfying this property.

We do not consider other interpretations of the information filtering architecture (such as power or rights), which may give rise to other logical rules. Moreover, we do not consider other logics of trust in computer science, for example the logical approaches to trust management (Becker, Russo, and Sultana 2012).

Our main contributions can be summarized as follows:

1. A new two-step information-filtering architecture separating trust inference from belief acceptance.
2. A comprehensive family of trust I/O logics, including the first semantic characterizations of some derivation systems.
3. The first study of the manipulation problem in trust-based information filtering.
4. Formal redundancy results reveals that some of the widely debated logical rules for trust inference are redundant with respect to information-filtering and belief manipulation.

The structure of the paper is as follows. Section 2 introduces pre-logical transformers. Section 3 develops a family of input/output logics for inferring trust and analyzes their behavior through various inference rules. Section 4 applies these logics to define an information filtering mechanism and examine which trust inference rules are redundant in that context. Section 5 turns to belief manipulation, formally characterizing which formulas are manipulable under different filtering operations. Section 6 discusses related work, and Section 7 concludes with a summary and future research.

2 Information Filtering Architecture on a Pre-Logical Level

Makinson and van der Torre (2000) distinguish two main kinds of input/output processes. The most studied processes

are desiderative transformations, where inputs may be conditions, with outputs expressing what is deemed desirable in those conditions. In this paper we consider processes serving for information filtering, some inputs pass, others are blocked or modified.

> The box may stop some inputs, while letting others through, perhaps in modified form. Inputs may record reports of agents, of the kind 'according to source i, x is true', while the box may give as output either x itself, a qualified version of x, or nothing at all, according to the identity of i. Or it might give output x only when at least two distinct sources vouch for it, and so on.

Moreover, Makinson and van der Torre (2000) observe that classical logic serves as an inference motor, with premises as inputs and conclusions as outputs, but real-world cases call for reasoning beyond just classical inference.

> It may also be seen in another role, as 'secretarial assistant' to some other, perhaps non-logical, transformation engine. The task of logic is one of preparing inputs before they go into the machine, unpacking outputs as they emerge and, less obviously, coordinating the two. The process as a whole is one of inference only when the central transformation is so. In general, it is one of 'logically assisted' transformation. That is the general perspective underlying the present paper.

On a pre-logical level, the input/output process can be explained as follows. Consider any set L (not necessarily of formulas), and any relation $G \subseteq L \times L$, which is referred to as a *transformer* (Makinson and Van Der Torre 2000), as illustrated by Figure 1. Given an input $A \subseteq L$, the output of A under G can be understood as $G(A) = \{x : (a, x) \in G \text{ for some } a \in A\}$. The architecture operates in two steps. First, an input set A is passed into the transformer G, producing the intermediate output $G(A)$, which represents the set of elements that would be trusted or authorized given A. This stage corresponds to inferring trust. The result $G(A)$ is labeled as "Trust" in the figure, and corresponds to the output of the transformer component. In the second stage, this set is intersected with the original input A. The result $G(A) \cap A$ forms the final output of the entire architecture. This final output represents those elements that are both informed and trusted, i.e., what is ultimately accepted.

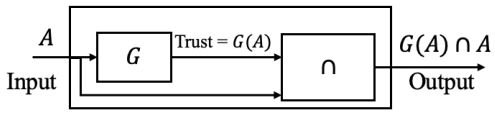

Figure 1: Pre-logical transformer

Consider a school entry policy where children may enter only if accompanied by at least one parent, while parents themselves are not admitted. A guard enforces this rule: a child without a parent is blocked, and a parent without a child has no effect. Let $P = \{p_1, p_2, p_3\}$ be the parents and $C = \{c_1, c_2, c_3\}$ the children, with parent–child relation $G = \{(p_1, c_1), (p_1, c_2), (p_2, c_1), (p_3, c_2)\}$, so c_1 has parents p_1, p_2 and c_2 has parents p_1, p_3. Given an input set $A \subseteq P \cup C$ of individuals at the gate, the authorized output is $G(A) = \{x \in C \mid \exists a \in A : (a, x) \in G\}$: a child is admitted if at least one parent is present. Thus, if $A = \{p_1\}$, both c_1 and c_2 may enter, while if $A = \{c_1\}$, nobody enters.

We distinguish *local* and *global* output. The *local output* of a transformer G is the set of all x such that $(a, x) \in G$ for some a in the input. The *global output* further restricts this by intersecting the local output with the input; if the input represents *informed information* and the local output represents *(epistemic) trust*, then the global output corresponds to *accepted belief*. In the school example, the local output consists of all children authorized to enter given the parents present, whereas the global output contains only those children who are both authorized and actually present, thus capturing a whitelist policy. We also define *manipulability*: an element $x \in L$ is manipulable if some input $A \subseteq L$ satisfies $x \in G(A) \cap A$. Since the global output represents accepted belief, manipulable elements are exactly those an agent can be made to believe. In the school example, c_3 has no parent linked in G, so for any $A \subseteq P \cup C$, we have $c_3 \notin G(A) \cap A$, and therefore c_3 is not manipulable.

3 Inferring Trust

In this section, we will present a family of I/O logics for inferring trust. We start by introducing the basics of the I/O logic framework.

We assume a nonempty set PROP of propositional variables or atoms. Let $\mathcal{L}$ be the propositional language generated by PROP, and elements of $\mathcal{L}$ are called formulas. For a finite set A of formulas, $\bigwedge A$ is the conjunction of all formulas in A, and likewise for $\bigvee A$. Note that $\bigwedge \emptyset := \top$ and $\bigvee \emptyset := \bot$. For all sets of formulas $A \cup \{a\}$, $A \vdash a$ denotes that a is a logical consequence of A in classical propositional logic. $Cn(A) = \{a \mid A \vdash a\}$ is the set of all the logical consequences of A. Given two formulas a and b, $a \dashv\vdash b$ abbreviates that $a \vdash b$ and $b \vdash a$.

In the I/O logic framework (Makinson and Van Der Torre 2000), a set G of pairs of formulas is called a generating set. In this paper, a pair $(a, x) \in G$ is interpreted as "if a is informed, then the judgment x is trusted". The idea behind the "conditional trust" is that some of our trusts may be triggered only if relevant information is provided. Consider the following example:

Example 1. Consider the following two statements made by an epidemiologist:

x: Vaccination reduces the spread of disease.

y: Economic lockdowns are ineffective in preventing recessions.

In this scenario, the epidemiologist's claim about vaccination falls within their domain of expertise and is thus trusted unconditionally. In our framework, this is represented by $(\top, x)$. However, their claim about economic outcomes is outside their field and is trusted only if supported by relevant evidence. Let a be "The epidemiologist has also published in

peer-reviewed economic journals." Then we may represent the conditional trust as (a, y).

An input is a set of formulas, representing the statements that have been informed. The question is how to reasonably define the set of formulas making up the output of A under G, which is intended to characterize the set of judgments that can be trusted, given the information A. The solution can be proposed in terms of either proof theory or semantics.

We first introduce the proof theory of the I/O logics in this paper. The three basic inference rules T, SI ("Strengthening of the Input"), and OEQ ("Output Equivalence") are listed below. T says that propositional tautologies are always unconditionally trusted. OEQ means that trust does not depend on the syntactic form of the sentence. SI states that trust is monotonic: if x is trusted given the information a, then it is still trusted given the more informative statement b.

Remark 1. In SI, the formula b may be inconsistent. Then it is questionable whether x can still be trusted given b. We will not address the problem of consistency in this section, but postpone it to the next where we propose an information filtering architecture with a consistency check.

T	From no premise to $(\top, \top)$.
SI	From (a, x) to (b, x) whenever $b \vdash a$.
OEQ	From (a, x) to (a, y) whenever $x \dashv\vdash y$.

In addition to the basic rules, we consider the following four optional rules. All of them have been discussed in the logical literature on trust. For example, ST abbreviates "Symmetric Trust". (The unconditional version of) ST is accepted in (Liau 2003) as an optional rule for trust inference, whereas AND is rejected in the same paper. The reason is that in AND, the conjunction $x \wedge y$ may be inconsistent (and, thus, untrustable), even if both x and y are consistent. As a weaker version of AND, DT ("Disjunctive Trust") is first introduced in (Dastani et al. 2005), where DT is argued to be a reasonable inference rule for trust that is not discussed in (Liau 2003). Finally, WO is also discussed in Liau's paper, but is considered to be problematic for trust. Recently, a normal modal logic for "trust in sincerity" is proposed in (Leturc and Bonnet 2018), in which all the following rules are valid for "trust in sincerity" except for ST.

ST	From (a, x) to $(a, \neg x)$.
DT	From (a, x) and (a, y) to $(a, x \vee y)$.
AND	From (a, x) and (a, y) to $(a, x \wedge y)$.
WO	From (a, x) to (a, y) whenever $x \vdash y$.

In this paper, we do not aim to settle the dispute over the "correct" inference rules for trust. Instead, we focus on the information filtering architecture (to be introduced) and study the effects of different inference rules on this architecture. As we shall see, some of the above rules are redundant with respect to the information filtering architecture.

In this section, we comprehensively study the I/O logics generated by all basic rules and subsets of the optional rules.

Definition 2. Let $R \subseteq \{\text{ST}, \text{DT}, \text{AND}, \text{WO}\}$. For all generating sets G, $deriv_R(G)$ is the smallest set that contains G and is closed under all the basic rules and rules in R. For all sets of formulas $A \cup \{x\}$, we put $(A, x) \in deriv_R(G)$ iff $(a, x) \in deriv_R(G)$ for some conjunction a of formulas in A. Finally, $deriv_R(G, A) = \{x \mid (A, x) \in deriv_R(G)\}$.

In what follows, we shall write $deriv_{\text{SDW}}(G)$ instead of $deriv_{\{\text{ST,DT,WO}\}}(G)$ and similarly for other derivations. That is, each rule is abbreviated by its initial. Our first observation is that ST and WO cannot both be applied for inferring trust.

Observation 3. If $R \supseteq \{\text{ST}, \text{WO}\}$, then $deriv_R(G, A) = \mathcal{L}$ for all inputs A.

Proof. By the inference rule T, $(\top, \top) \in deriv_R(G)$. By ST, $(\top, \neg\top) \in deriv_R(G)$. For all formulas x, since $\neg\top \vdash x$, $(\top, x) \in deriv_R(G)$ by WO. Since $\top = \bigwedge \emptyset$, $(A, x) \in deriv_R(G)$, i.e., $x \in deriv_R(G, A)$. Therefore, $deriv_R(G, A) = \mathcal{L}$. □

It is easy to see that DT is derivable from WO. Therefore, the next observation states that given WO, DT is redundant for inferring trust.

Observation 4. For all inputs A, we have: $deriv_{\text{W}}(G, A) = deriv_{\text{DW}}(G, A)$ and $deriv_{\text{AW}}(G, A) = deriv_{\text{DAW}}(G, A)$.

The next observation states that given ST, DT and AND are interderivable.

Observation 5. For all inputs A, we have:

$$deriv_{\text{SD}}(G, A) = deriv_{\text{SA}}(G, A) = deriv_{\text{SDA}}(G, A).$$

Proof. It suffices to show that $deriv_{\text{SD}}(G) = deriv_{\text{SDA}}(G)$ and $deriv_{\text{SA}}(G) = deriv_{\text{SDA}}(G)$. For the former, it suffices to show that $deriv_{\text{SD}}(G)$ is closed under AND. Assume that (a, x) and (a, y) are in $deriv_{\text{SD}}(G)$. The following derivation shows that $(a, x \wedge y) \in deriv_{\text{SD}}(G)$:

$$\cfrac{\cfrac{\cfrac{\cfrac{(a,x)}{(a,\neg x)}\,\text{ST} \qquad \cfrac{(a,y)}{(a,\neg y)}\,\text{ST}}{(a, \neg x \vee \neg y)}\,\text{DT}}{(a, \neg(\neg x \vee \neg y))}\,\text{ST}}{(a, x \wedge y)}\,\text{OEQ}$$

Similarly, $deriv_{\text{SA}}(G) = deriv_{\text{SDA}}(G)$. □

According to Observations 3, 4, and 5, there are at most 8 mutually different and non-trivial I/O logics generated by the optional rules: $deriv_{\emptyset}$, $deriv_{\text{S}}$, $deriv_{\text{D}}$, $deriv_{\text{A}}$, $deriv_{\text{W}}$, $deriv_{\text{SD}}$, $deriv_{\text{DA}}$, and $deriv_{\text{AW}}$. The remainder of this section focus on the semantic characterization of the 8 logics.

We start with $deriv_{\text{AW}}$. In (Makinson and Van Der Torre 2000), the derivation $deriv_1$ is defined such that $deriv_1(G)$ is the smallest that contains G and is closed under T, SI, AND, and WO. The only difference between $deriv_1$ and our $deriv_{\text{AW}}$ is that OEQ does not appear in the definition of $deriv_1$. Nevertheless, it is easy to see that OEQ can be derived from WO. Thus, $deriv_1$ and $deriv_{\text{AW}}$ are the same.

Observation 6. $deriv_{\text{AW}}(G, A) = deriv_1(G, A)$ for all A.

It is established in (Makinson and Van Der Torre 2000) that $deriv_1(G, A) = Cn(G(Cn(A)))$, where $G(B) = \{x \mid (a, x) \in G \text{ for some } a \in B\}$ for all sets B of formulas. Therefore, the semantic characterization of $deriv_{\text{AW}}$ is straightforward.

Definition 7. For all inputs A, we put $out_{\mathrm{AW}}(G, A) = Cn(G(Cn(A)))$. In addition, $out_{\mathrm{AW}}(G) = \{(a, x) \mid x \in out_{\mathrm{AW}}(G, a)\}$.[1]

Proposition 8. $deriv_{\mathrm{AW}}(G, A) = out_{\mathrm{AW}}(G, A)$.

Next we present the semantic characterization for $out_{\emptyset}$, out_{S}, out_{D}, out_{A}, out_{W}. To do this, we need some notations. For all sets B of formulas, let

- $Eq(B) = \{x \mid x \dashv\vdash \top \text{ or } \exists a \in B \text{ such that } a \dashv\vdash x\}$,
- $\overline{B} = \{\neg x \mid x \in B\}$,
- $Disj(B) = \{\bigvee B' \mid \emptyset \neq B' \subseteq B \text{ and } B' \text{ is finite}\}$,
- $Conj(B) = \{\bigwedge B' \mid \emptyset \neq B' \subseteq B \text{ and } B' \text{ is finite}\}$, and
- $Cn^{-}(B) = \{x \mid x \dashv\vdash \top \text{ or } \exists a \in B \text{ such that } a \vdash x\}$.

Definition 9. For all inputs A, we put:

$$
\begin{aligned}
out_{\emptyset}(G, A) &= Eq(G(Cn(A))) \\
out_{\mathrm{S}}(G, A) &= Eq(G(Cn(A))) \cup \overline{Eq(G(Cn(A)))} \\
out_{\mathrm{D}}(G, A) &= Eq(Disj(G(Cn(A)))) \\
out_{\mathrm{A}}(G, A) &= Eq(Conj(G(Cn(A)))) \\
out_{\mathrm{W}}(G, A) &= Cn^{-}(G(Cn(A)))
\end{aligned}
$$

For each $R \in \{\emptyset, \mathrm{S}, \mathrm{D}, \mathrm{A}, \mathrm{W}\}$, $out_R(G)$ is analogously defined as before.

Proposition 10. *For each* $R \in \{\emptyset, \mathrm{S}, \mathrm{D}, \mathrm{A}, \mathrm{W}\}$,

$$deriv_R(G, A) = out_R(G, A) \textit{ for all inputs } A.$$

Proof. We distinguish five cases depending on the choices of R.

Case $R = \emptyset$. From left to right: Suppose $x \in deriv_{\emptyset}(G, A)$. Then $(a, x) \in deriv_{\emptyset}(G)$ for some conjunction of formulas in A. It is easy to verify that $out_{\emptyset}(G)$ contains G and is closed under T, SI, and OEQ. Hence, $deriv_{\emptyset}(G) \subseteq out_{\emptyset}(G)$ (since $deriv_{\emptyset}(G)$ is the smallest such set). Therefore, $(a, x) \in out_{\emptyset}(G)$, i.e., $x \in out_{\emptyset}(G, a)$. Note that $out_{\emptyset}(G, a) \subseteq out_{\emptyset}(G, A)$. Hence, $x \in out_{\emptyset}(G, A)$.

From right to left. Suppose $x \in out_{\emptyset}(G, A) = Eq(G(Cn(A)))$. We consider two cases: (1) $x \dashv\vdash \top$. Since $(\top, \top) \in deriv_{\emptyset}(G)$ by T, $(\top, x) \in deriv_{\emptyset}(G)$ by OEQ. Thus, $x \in deriv_{\emptyset}(G, A)$. (2) Otherwise, there must be $(b, y) \in G$ such that $b \in Cn(A)$ and $y \dashv\vdash x$. Since $b \in Cn(A)$, by the compactness of propositional logic, there must be a conjunction a of formulas in A such that $a \vdash b$. Since $(b, y) \in G$, $(b, y) \in deriv_{\emptyset}(G)$. Since $deriv_{\emptyset}(G)$ is closed under SI, $(a, y) \in deriv_{\emptyset}(G)$. Thus, by OEQ, $(a, x) \in deriv_{\emptyset}(G)$. Since a is a conjunction of formulas in A, $x \in deriv_{\emptyset}(G, A)$.

Case $R = \mathrm{S}$. From left to right. It suffices to show that $deriv_{\mathrm{S}}(G) \subseteq out_{\mathrm{S}}(G)$. It is straightforward to verify that $out_{\mathrm{S}}(G)$ contains G and is closed under T, SI, OEQ, and ST. Since $deriv_{\mathrm{S}}(G)$ is the smallest such set, $deriv_{\mathrm{S}}(G, A) \subseteq out_{\mathrm{S}}(G, A)$.

From right to left. Suppose $x \in out_{\mathrm{S}}(G, A) = Eq(G(Cn(A))) \cup \overline{Eq(G(Cn(A)))}$. If $x \in Eq(G(Cn(A)))$, then $x \in deriv_{\mathrm{S}}(G, A)$ since $x \in deriv_{\emptyset}(G, A)$ and $deriv_{\emptyset}(G, A) \subseteq deriv_{\mathrm{S}}(G, A)$. Otherwise, $x \in \overline{Eq(G(Cn(A)))}$. Then $x = \neg y$ for some formula $y \in Eq(G(Cn(A))) \subseteq deriv_{\mathrm{S}}(G, A)$. Since $y \in deriv_{\mathrm{S}}(G, A)$. There must be a conjunction a of formulas in A such that $(a, y) \in deriv_{\mathrm{S}}(G)$. Then $(a, x) \in deriv_{\mathrm{S}}(G)$ since $deriv_{\mathrm{S}}(G)$ is closed under ST. Thus, $x \in deriv_{\mathrm{S}}(G, A)$.

Case $R = \mathrm{D}$. From left to right. It suffices to show that $deriv_{\mathrm{D}}(G) \subseteq out_{\mathrm{D}}(G)$, which can be shown by verifying that $out_{\mathrm{D}}(G)$ contains G and is closed under the rules T, SI, OEQ, and DT.

From right to left. Suppose $x \in Eq(Disj(G(Cn(A))))$. We consider two cases. The case $x \dashv\vdash \top$ can be shown as before. Otherwise, there is a finite and nonempty subset $B = \{y_1, \ldots, y_n\} \subseteq G(Cn(A))$ such that $x \dashv\vdash \bigvee B$. For each $y_i \in B$, since $y_i \in G(Cn(A))$, there must be $a_i \in Cn(A)$ such that $(a_i, y_i) \in G \subseteq deriv_{\mathrm{D}}(G)$. Thus, $(\bigwedge_{1 \le j \le n} a_j, y_i) \in deriv_{\mathrm{D}}(G)$ for each i by SI. Hence, $(\bigwedge_{1 \le j \le n} a_j, \bigvee B) \in deriv_{\mathrm{D}}(G)$ by DT. Note that $\bigwedge_{1 \le j \le n} a_j \in Cn(A)$. Hence, by the compactness of propositional logic, there must be a conjunction a of formulas in A such that $a \vdash \bigwedge_{1 \le j \le n} a_j$. By SI, $(a, \bigvee B) \in deriv_{\mathrm{D}}(G)$. Since $x \dashv\vdash \bigvee B$, $(a, x) \in deriv_{\mathrm{D}}(G)$ by OEQ. Therefore, $x \in deriv_{\mathrm{D}}(G, A)$.

Case $R = \mathrm{A}$. It is analogous to the case $R = \mathrm{D}$.

Case $R = \mathrm{W}$. From left to right. It suffices to show that $deriv_{\mathrm{W}}(G) \subseteq out_{\mathrm{W}}(G)$, which can be shown by verifying that $out_{\mathrm{W}}(G)$ contains G and is closed under the rules T, SI, OEQ, and WO.

From right to left. Suppose $x \in out_{\mathrm{W}}(G, A)$. We consider only the case $x \not\dashv\vdash \top$. Then there is $y \in G(Cn(A))$ such that $y \vdash x$. Since $y \in G(Cn(A))$, there is $(b, y) \in G$ such that $b \in Cn(A)$. Since $b \in Cn(A)$, by propositional logic there must be a conjunction a of formulas in A such that $a \vdash b$. Since $(b, y) \in G$ and $deriv_{\mathrm{W}}(G)$ contains G, $(b, y) \in deriv_{\mathrm{W}}(G)$. By SI, $(a, y) \in deriv_{\mathrm{W}}(G)$. By WO, $(a, x) \in deriv_{\mathrm{W}}(G)$. Hence, $x \in deriv_{\mathrm{W}}(G, A)$. □

The semantic characterization of $deriv_{\mathrm{SD}}$ and $deriv_{\mathrm{DA}}$ remains to be considered. Next we consider $deriv_{\mathrm{SD}}$.

Definition 11. For all inputs A, let $out_{\mathrm{SD}}(G, A) =$

$$\{x \mid \exists A_1, \ldots, A_n \subseteq G(Cn(A)) \cup \overline{G(Cn(A))} \text{ such that } n \ge 0, \text{ each } A_i \text{ is finite, and } x \dashv\vdash \bigvee_{1 \le i \le n} \bigwedge A_i\}.$$

Proposition 12. $deriv_{\mathrm{SD}}(G, A) = out_{\mathrm{SD}}(G, A)$.

Proof. From left to right. It suffices to show $deriv_{\mathrm{SD}}(G) \subseteq out_{\mathrm{SD}}(G)$. This can be shown by verifying that $out_{\mathrm{SD}}(G)$ contains G and is closed under the rules T, SI, OEQ, ST, and DT. We show only the case for ST. Suppose $(a, x) \in out_{\mathrm{SD}}(G)$. Then $x \in out_{\mathrm{SD}}(G, a)$. By definition, there must be finite $A_1, \ldots, A_n \subseteq G(Cn(a)) \cup \overline{G(Cn(a))}$ such that $x \dashv\vdash \bigvee_{1 \le i \le n} \bigwedge A_i$. We distinguish two cases. (1) If $n = 0$, then $x \dashv\vdash \bot$. Since $out(G)$ is closed under T and SI, $(a, \top) \in out_{\mathrm{SD}}(G)$. Thus, $(a, \neg x) \in out_{\mathrm{SD}}(G)$ by OEQ. (2)

[1] $out_{\mathrm{AW}}(G, a) = out_{\mathrm{AW}}(G, \{a\})$.

$n \neq 0$. Since $x \dashv\vdash \bigvee_{1\leq i\leq n} \bigwedge A_i$, by propositional logic,

$$\neg x \dashv\vdash \bigvee \{\neg a_1 \wedge \cdots \wedge \neg a_n \mid a_i \in A_i \text{ for each } 1 \leq i \leq n\}$$

Thus, we can let $B_1, B_2, \ldots, B_m$ be an enumeration of all sets B such that $B = \{\neg a_1, \ldots, \neg a_n\}$ with $a_i \in A_i$ for each $1 \leq i \leq n$. Then $\neg x \dashv\vdash \bigvee_{1\leq i\leq m} \bigwedge B_i$. Note that for each B_i and formula $b \in B_i$, there is $c \in G(Cn(a)) \cup \overline{G(Cn(a))}$ such that $c \dashv\vdash b$. Let us denote by B_i' the set obtained by replacing each formula in B_i with the equivalent in $G(Cn(a)) \cup \overline{G(Cn(a))}$. Then it holds that $\neg x \dashv\vdash \bigvee_{1\leq i\leq m} \bigwedge B_i'$. Since $B_i' \subseteq G(Cn(a)) \cup \overline{G(Cn(a))}$ for each $1 \leq i \leq m$, $x \in out_{\text{SD}}(G, a)$.

From right to left. Suppose $x \in out_{\text{SD}}(G, A)$. There must be finite $A_1, \ldots, A_n \subseteq G(Cn(A)) \cup \overline{G(Cn(A))}$ such that $x \dashv\vdash \bigvee_{1\leq i\leq n} \bigwedge A_i$. If $n = 0$, then $x \dashv\vdash \bot$. Since $(\top, x) \in deriv_{\text{SD}}(G)$ (by T, ST, and OEQ), $x \in deriv_{\text{SD}}(G, A)$. Otherwise, for each A_i and formula $y \in A_i$, $y \in deriv_{\text{S}}(G, A)$ by Proposition 10. Hence, $y \in deriv_{\text{SD}}(G, A)$. Thus, there must be a conjunction y^* of formulas in A such that $(y^*, y) \in deriv_{\text{SD}}(G)$. By SI and AND[2], it follows that $(\bigwedge_{y\in A_i} y^*, \bigwedge A_i) \in deriv_{\text{SD}}(G)$ for each i. By SI and DT, it follows that $(\bigwedge_{1\leq i\leq n y\in A_i} y^*, \bigvee_{1\leq i\leq n} \bigwedge A_i) \in deriv_{\text{SD}}(G)$. Thus, by OEQ, $(\bigwedge_{1\leq i\leq n y\in A_i} y^*, x) \in deriv_{\text{SD}}(G)$. Therefore, $x \in deriv_{\text{SD}}(G, A)$. □

Finally, we consider $deriv_{\text{DA}}$.

Definition 13. For all inputs A, let $out_{\text{DA}}(G, A) =$

$$\{x \mid \exists A_1, \ldots, A_n \subseteq G(Cn(A)) \text{ such that } n \geq 1, \text{ each } A_i \text{ is finite, and } x \dashv\vdash \bigvee_{1\leq i\leq n} \bigwedge A_i\}.$$

Proposition 14. $deriv_{\text{DA}}(G, A) = out_{\text{DA}}(G, A)$.

Proof. From left to right. It suffices to show $deriv_{\text{DA}}(G) \subseteq out_{\text{DA}}(G)$. This can be done by verifying that $out_{\text{DA}}(G)$ contains G and is closed under T, SI, OEQ, DT, and AND. We show only the case for AND. Suppose both (a, x) and (a, y) are in $out_{\text{DA}}(G)$. Then $x \in out_{\text{DA}}(G, a)$ and $y \in out_{\text{DA}}(G, a)$. By definition, there must be finite $A_1, \ldots, A_n, B_1, \ldots, B_m \subseteq G(Cn(a))$ such that

$$x \dashv\vdash \bigvee_{1\leq i\leq n} \bigwedge A_i \text{ and } y \dashv\vdash \bigvee_{1\leq j\leq m} \bigwedge B_j.$$

Let $\mathcal{C} = \{A_i \cup B_j \mid 1 \leq i \leq n \text{ and } 1 \leq j \leq m\}$. By propositional logic, $x \wedge y \dashv\vdash \bigvee_{C\in\mathcal{C}} \bigwedge C$. It follows that $x \wedge y \in out_{\text{DA}}(G, a)$, i.e., $(a, x \wedge y) \in out_{\text{DA}}(G)$.

The direction from right to left can be shown similarly as in the proof of Proposition 12. □

[2]Note that $deriv_{\text{SD}}(G) = deriv_{\text{SDA}}(G)$ from Observation 5.

We conclude this section with an example illustrating the difference between the 8 I/O logics.

Example 15. Let $G = \{(\top, x), (a, y)\}$ and $A = \{a\}$.

- $out_{\emptyset}(G, A) = Eq(x, y)$.
- $out_{\text{S}}(G, A) = Eq(x, y) \cup \overline{Eq(x, y)}$.
- $out_{\text{D}}(G, A) = Eq(x, y, x \vee y)$.
- $out_{\text{A}}(G, A) = Eq(x, y, x \wedge y)$.
- $out_{\text{W}}(G, A) = Cn^{-}(x, y)$.
- $out_{\text{DA}}(G, A) = Eq(x, y, x \vee y, x \wedge y)$.
- $out_{\text{AW}}(G, A) = Cn(x, y)$.
- $\neg(x \leftrightarrow y) \in out_{\text{SD}}(G, A)$.

Note that $\neg(x \leftrightarrow y)$ is not in $out_R(G, A)$ for all $R \neq \text{SD}$.

4 Filtering Information by Trust

In the last section, we proposed a family of I/O logics for inferring trust. In this section, given a set A of statements that have been informed, we investigate which information contained in A will be believed/accepted by a rational agent. For this, we propose an information filtering architecture based on trust. Our logical analysis shows that some rules for trust inference are redundant with respect to the mechanism.

Belief change has been extensively studied in, e.g., belief revision theory (Alchourrón, Gärdenfors, and Makinson 1985) and dynamic epistemic logic (van Ditmarsch, van der Hoek, and Kooi 2008). In both approaches, the incoming information is always completely accepted without scrutiny (in AGM belief revision, the success postulate is included and in public announcement logic, the announced (propositional) formulas are always believed). However, when trust is considered in the process of belief change, not all incoming information will be believed, because "trust plays the role of filtering out noisy information" (Liau 2003).

We present a simple information filtering architecture in the next definition. The core idea is that a piece of information will be believed if (and only if) it is implied by what is both informed and trusted.

Definition 16. Let $R \in \{\emptyset, \text{S}, \text{D}, \text{A}, \text{W}, \text{SD}, \text{DA}, \text{AW}\}$. The operation out_R^f is defined as follows.

$$out_R^f(G, A) = \begin{cases} Cn(\emptyset) & \text{if } A \text{ is inconsistent,} \\ Cn(out_R(G, A) \cap Cn(A)) & \text{otherwise.} \end{cases}$$

Note that the above definition ensures that nothing (except for tautologies) will be believed if the input is inconsistent. Let us illustrate the definition by an example:

Example 17. Let $G = \{(\top, x), (a, y)\}$ and $A = \{x, y\}$. Then $out_{\emptyset}(G, A) = Eq(x)$. Thus, $out_{\emptyset}^f(G, A) = Cn(x)$.

Next, we list some obvious properties of out_R^f: the rules T, OEQ, WO, DT, and AND are valid for out_R^f. A restricted form of SI is also valid, whereas the rule ST is invalid.

Observation 18. Let out_R^f be a filtering operation. The following holds for all generating sets G and inputs A:

- $\top \in out_R^f(G, A)$.
- if $x \vdash y$, then $x \in out_R^f(G, A)$ implies $y \in out_R^f(G, A)$.

- if $x, y \in out^f_R(G, A)$, then $x \vee y, x \wedge y \in out^f_R(G, A)$.
- if $x \in out^f_R(G, A)$ then $x \in out^f_R(G, B)$, provided that $A \subseteq Cn(B)$ and B is consistent.
- $x \in out^f_R(G, A)$ implies that $\neg x \notin out^f_R(G, A)$.

The next three observations are about the redundancy of the rule AND with respect to the information filtering architecture. This fact has been informally mentioned in (Liau 2003). Our information filtering mechanism enables us to formally define the notion of "redundancy", and we obtain formal results of the redundancy of certain rules.

Observation 19. $out^f_{\text{A}}(G, A) = out^f_{\emptyset}(G, A)$.

Proof. We consider only the case when A is consistent. The inclusion $out^f_{\emptyset}(G, A) \subseteq out^f_{\text{A}}(G, A)$ follows directly from the fact that $out_{\emptyset}(G, A) \subseteq out_{\text{A}}(G, A)$. It remains to show that $out^f_{\text{A}}(G, A) \subseteq out^f_{\emptyset}(G, A)$. By definition and the properties of Cn, it suffices to show that $out_{\text{A}}(G, A) \cap Cn(A) \subseteq Cn(out_{\emptyset}(G, A) \cap Cn(A))$. Let $x \in out_{\text{A}}(G, A) \cap Cn(A)$. If $x \dashv\vdash \top$, then $x \in Cn(out_{\emptyset}(G, A) \cap Cn(A)$. Otherwise, (by the definition of out_{A}) there must be finite and non-empty $B \subseteq G(Cn(A))$ such that $x \dashv\vdash \bigwedge B$. It is easy to see that $B \subseteq out_{\emptyset}(G, A) \cap Cn(A)$. Hence, $x \in Cn(out_{\emptyset}(G, A) \cap Cn(A))$. □

Observation 20. $out^f_{\text{DA}}(G, A) = out^f_{\text{D}}(G, A)$.

Proof. We consider only the case when A is consistent. We show that $out^f_{\text{DA}}(G, A) \subseteq out^f_{\text{D}}(G, A)$ (the other direction is trivial). By definition and the properties of Cn, it suffices to show that $out_{\text{DA}}(G, A) \cap Cn(A) \subseteq Cn(out_{\text{D}}(G, A) \cap Cn(A))$. Let $x \in out_{\text{DA}}(G, A) \cap Cn(A)$. The case $x \dashv\vdash \top$ is trivial. We assume $x \not\dashv\vdash \top$. By the definition of out_{DA}, there are finite and nonempty $A_1, \ldots, A_n \subseteq G(Cn(A))$ $(n \geq 1)$ such that $x \dashv\vdash \bigvee\limits_{1 \leq i \leq n} \bigwedge A_i$. Applying (the distribution law of) propositional logic, this can be rewritten as $x \dashv\vdash \bigwedge\limits_{1 \leq j \leq m} \bigvee B_j$ where each $B_j \subseteq G(Cn(A))$. For each B_j, it is easy to see that $\bigvee B_j \in Disj(G(Cn(A))) \subseteq out_{\text{D}}(G, A)$ and $\bigvee B_j \in Cn(A)$ (since $x \in Cn(A)$). Hence, $\{\bigvee B_j \mid 1 \leq j \leq m\} \subseteq out_{\text{D}}(G, A) \cap Cn(A)$. Thus, $x \in Cn(out_{\text{D}}(G, A) \cap Cn(A))$. □

Observation 21. $out^f_{\text{AW}}(G, A) = out^f_{\text{W}}(G, A)$.

Proof. We consider only the case where A is consistent. We show that $out^f_{\text{AW}}(G, A) \subseteq out^f_{\text{W}}(G, A)$ (the other direction is trivial). By definition and the properties of Cn, it suffices to show that $out_{\text{AW}}(G, A) \cap Cn(A) \subseteq Cn(out_{\text{W}}(G, A) \cap Cn(A))$. Let $x \in out_{\text{AW}}(G, A) \cap Cn(A)$. If $x \dashv\vdash \top$, then $x \in Cn(out_{\text{W}}(G, A) \cap Cn(A))$. Otherwise, since $x \in out_{\text{AW}}(G, A) = Cn(G(Cn(A)))$, there must be $b_1, \ldots, b_n \in G(Cn(A))$ $(n \geq 1)$ such that $\bigwedge\limits_{1 \leq i \leq n} b_i \vdash x$. Then $x \dashv\vdash \bigwedge\limits_{1 \leq i \leq n} (x \vee b_i)$. For each i, $x \vee b_i \in out_{\text{W}}(G, A) = Cn^-(G(Cn(A)))$ and $x \vee b_i \in Cn(A)$ (since $x \in Cn(A)$). Hence, $x \in out^f_{\text{W}}(G, A) = Cn(out_{\text{W}}(G, A) \cap Cn(A))$. □

According to Observations 19, 20, and 21, there are at most 5 different filtering logics: $out^f_{\emptyset}$, out^f_{S}, out^f_{D}, out^f_{W}, and out^f_{SD}. Next, we show that they are mutually different by some examples.

Example 22. Let $G = \{(\top, \neg a \wedge b), (\top, b), (\top, a \wedge \neg b)\}$ and $A = \{a\}$. We have:

- $out_{\emptyset}(G, A) = Eq(\neg a \wedge b, b, a \wedge \neg b)$.
  Thus, $out^f_{\emptyset}(G, A) = Cn(\emptyset)$;
- $out_{\text{S}}(G, A) = Eq(\neg a \wedge b, a \vee \neg b, b, \neg b, a \wedge \neg b, \neg a \vee b, \bot)$.
  Thus, $out^f_{\text{S}}(G, A) = Cn(a \vee \neg b)$.
- $out_{\text{D}}(G, A) = Eq(Disj(\neg a \wedge b, b, a \wedge \neg b))$.
  Therefore, $out^f_{\text{D}}(G, A) = Cn(a \vee b)$.
- Since $a \in out_{\text{SD}}(G, A)$, $out^f_{\text{SD}}(G, A) = Cn(a)$.

The above example shows that $out^f_{\emptyset}$, out^f_{S}, out^f_{D}, and out^f_{SD} are different from each other. It remains to show the independence of out^f_{W}.

Example 23. Let $G = \{(\top, a), (\top, b)\}$ and $A = \{a \vee c\}$. Then $out_{\text{W}}(G, A) = Cn^-(a, b)$. Therefore, $a \vee c \in out^f_{\text{W}}(G, A)$. However, $a \vee c \notin out^f_{\text{SD}}(G, A)$ because $a \vee c \notin out_{\text{SD}}(G, A)$. Therefore, $a \vee c$ is in none of $out^f_{\emptyset}(G, A)$, $out^f_{\text{S}}(G, A)$, and $out^f_{\text{D}}(G, A)$.

5 Belief Manipulation

In the previous section, we have proposed a mechanism for the acceptance of information by rational agents based on their trust. A natural question is whether the mechanism can be manipulated. In Example 1, e.g., the epidemiologist may intend that the public believe the effects of vaccination on reducing the spread of disease. In this case, it is meaningful to know whether there exist statements by the epidemiologist that can make the public believe the effects of vaccination. In this section, we investigate the problem of belief manipulation in the information filtering architecture. We show that certain information filtering operations are equivalently manipulable, in the sense that the sets of manipulable formulas are the same. Furthermore, we characterize the set of manipulable formulas under various filtering operations.

Definition 24. Given a filtering operation out^f_R and a generating set G, a formula x is *manipulable* under G and out^f_R if there exists an input A such that $x \in out^f_R(G, A)$.

Example 25. Let $G = \{(\top, x), (a, y)\}$. The formula x is manipulable under G and $out^f_{\emptyset}$ because $x \in out^f_{\emptyset}(G, x)$. On the contrary, $\neg x$ is not manipulable under G and $out^f_{\emptyset}$.

Before characterizing the set of manipulable formulas, we first explore the relationship between the sets of manipulable formulas under different filtering operations.

Definition 26. Let two filtering operation out^f_R and $out^f_{R'}$ be given. We say out^f_R and $out^f_{R'}$ are *equivalently manipulable*, notation $out^f_R \equiv out^f_{R'}$, if for all generating sets G, the manipulable formulas under G and out^f_R are the same as that under G and $out^f_{R'}$.

In the last section, we have seen that the rule AND is redundant with respect to the information filtering architecture. Thus, the next observation is obvious.

Observation 27. $out^f_\emptyset \equiv out^f_{\text{A}}$, $out^f_{\text{D}} \equiv out^f_{\text{DA}}$, and $out^f_{\text{W}} \equiv out^f_{\text{AW}}$.

Next, we show that the rule DT is also redundant, as far as only the manipulable formulas are concerned. For this, we need the next lemma. Recall that a set V of formulas is "maximal consistent" if V is consistent, and no proper superset of V is consistent.

Lemma 28. *Let out^f_R and G be given. A formula x ($x \not\dashv\vdash \top$) is manipulable under G and out^f_R iff there is a maximal consistent set V such that $x \in Cn(out_R(G, V) \cap V)$.*

Proof. The direction from right to left is trivial. From left to right. Suppose x is manipulable and $x \not\dashv\vdash \top$. Then there exists A such that $x \in out^f_R(G, A)$. Since $x \not\dashv\vdash \top$, A must be consistent. Therefore, $x \in Cn(out_R(G, A) \cap Cn(A))$. Note that, by the Lindenbaum lemma, there exists a maximal consistent set $V \supseteq A$. Since $out_R(G, A) \subseteq out_R(G, V)$ and $Cn(A) \subseteq Cn(V) = V$, $x \in Cn(out_R(G, V) \cap V)$. □

Observation 29. $out^f_\emptyset \equiv out^f_{\text{D}}$.

Proof. We need to show that for all formulas x and generating sets G, x is manipulable under $out^f_\emptyset$ and G iff x is manipulable under out^f_{D} and G.

The case $x \dashv\vdash \top$ is trivial. We consider only the case $x \not\dashv\vdash \top$. The direction from left to right follows from the fact that $out^f_\emptyset(G, A) \subseteq out^f_{\text{D}}(G, A)$ for any A. From right to left. Suppose x is manipulable under out^f_{D} and G. By Lemma 28, there is a maximal consistent set V such that $x \in Cn(out_{\text{D}}(G, V) \cap V)$. To show that x is manipulable under $out_\emptyset$, it suffices to show that $Cn(out_{\text{D}}(G, V) \cap V) \subseteq Cn(out_\emptyset(G, V) \cap V)$. By the properties of Cn, it suffices again to show that $out_{\text{D}}(G, V) \cap V \subseteq Cn(out_\emptyset(G, V) \cap V)$. For each $y \in out_{\text{D}}(G, V) \cap V$, if $y \dashv\vdash \top$ then $y \in Cn(out_\emptyset(G, V) \cap V)$. Otherwise, $y \in V$ and there is finite and nonempty $B \subseteq G(Cn(V))$ such that $y \dashv\vdash \bigvee B$. Since $y \in V$, $\bigvee B \in V$. By the property of maximal consistent set, there is a $b \in B$ such that $b \in V$. Note that $b \in G(Cn(V)) \subseteq out_\emptyset(G, V)$. Hence, $b \in out_\emptyset(G, V) \cap V$. Hence, $y \in Cn(out_\emptyset(G, V) \cap V)$. □

Observation 30. $out^f_{\text{S}} \equiv out^f_{\text{SD}}$.

Proof. We need to show that for all formulas x and generating sets G, x is manipulable under out^f_{S} and G iff x is manipulable under out^f_{SD} and G.

By the same reasoning as in the proof of Observation 29, we need only to show that $out_{\text{SD}}(G, V) \cap V \subseteq Cn(out_{\text{S}}(G, V) \cap V)$ (for any maximal consistent set V). Let $y \in out_{\text{SD}}(G, V) \cap V$. Then there exist finite $A_1, \ldots, A_n \subseteq G(Cn(V)) \cup \overline{G(Cn(V))}$ such that $y \dashv\vdash \bigvee\limits_{1 \le i \le n} \bigwedge A_i$. Since y is consistent, $n \neq 0$. Since $y \in V$, $\bigvee\limits_{1 \le i \le n} \bigwedge A_i \in V$. By the property of maximal consistent sets, there is some $1 \le j \le n$ such that $\bigwedge A_j \in V$. Thus, $A_j \subseteq V$. On the other hand, since $A_j \subseteq G(Cn(V) \cup \overline{G(Cn(V))})$, $A_j \subseteq out_{\text{S}}(G, V)$. Hence, $A_j \subseteq out_{\text{S}}(G, V) \cap V$. Note that $A_j \vdash y$. Therefore, $y \in Cn(out_{\text{S}}(G, V) \cap V)$. □

According to Observations 27, 29, and 30, there are at most 3 filtering operations that are not equivalently manipulable: $out^f_\emptyset$, out^f_{S}, and out^f_{W}. The next two examples show that the sets of manipulable formulas under the three filtering operations are different from each other.

Example 31. To see $out^f_\emptyset \not\equiv out^f_{\text{S}}$, we employ the same G as in Example 25, i.e., $G = \{(\top, x), (a, y)\}$. We have seen that $\neg x$ is not manipulable under G and $out^f_\emptyset$. However, $\neg x \in out^f_{\text{S}}(G, A)$ where $A = \{\neg x\}$.

Example 32. Let $G = \{(p, \bot)\}$. The formula p is manipulable under G and out^f_{W} since $p \in out^f_{\text{W}}(G, p)$. However, p is not manipulable under G and any of $out^f_\emptyset$ and out^f_{S}.

Example 32 shows that the manipulable formulas under $out^f_\emptyset$ and out^f_{W} are different. But what is the exact relationship between them? The next observation answers the question. For any generating set G and formula x, let $G^x = \{(a, b \vee x) \mid (a, b) \in G\}$.

Observation 33. Let G be given. For all formulas x, x is manipulable under G and out^f_{W} iff x is manipulable under G^x and $out^f_\emptyset$.

Proof. It suffices to show that for all inputs A,

$$x \in out^f_{\text{W}}(G, A) \text{ iff } x \in out^f_\emptyset(G^x, A).$$

We consider only the case when A is consistent. From left to right. Suppose $x \in out^f_{\text{W}}(G, A)$. Then, by definition, there are $a_1, \ldots, a_n \in Cn^-(G(Cn(A))) \cap Cn(A)$ such that $\bigwedge\limits_{1 \le i \le n} a_i \vdash x$. For each i, without loss of generality, we assume that $a_i \not\dashv\vdash \top$. Since $a_i \in Cn^-(G(Cn(A)))$, there must be $b_i \in G(Cn(A))$ such that $b_i \vdash a_i$. Then $\bigwedge\limits_{1 \le i \le n} b_i \vdash x$. Therefore, $\bigwedge\limits_{1 \le i \le n} (b_i \vee x) \vdash x$. Note that for each i, $b_i \vee x \in G^x(Cn(A))$ and $b_i \vee x \in Cn(A)$. Hence, $x \in out^f_\emptyset(G^x, A)$.

From right to left. Since A is consistent, it suffices to show that $out_\emptyset(G^x, A) \subseteq out_{\text{W}}(G, A)$. Let $y \in out_\emptyset(G^x, A) = Eq(G^x(Cn(A)))$. We consider only the case $y \not\dashv\vdash \top$. Then there is $(a, b \vee x) \in G^x$ such that $a \in Cn(A)$ and $b \vee x \dashv\vdash y$. Since $(a, b) \in G$ and $b \vdash y$, $y \in Cn^-(G(Cn(A))) = out_{\text{W}}(G, A)$. □

We conclude this section by characterizing the manipulable formulas under $out^f_\emptyset$, out^f_{S}, and out^f_{W}. For all generating sets G, let $b(G) = \{a \mid (a, x) \in G\}$.

Proposition 34. *A formula x is manipulable under G and $out^f_\emptyset$ (out^f_{S}, respectively) iff there are $B \subseteq b(G)$ and $H \subseteq G(B)$ ($H \subseteq G(B) \cup \overline{G(B)}$, respectively) such that $B \cup H$ is consistent and $x \in Cn(H)$.*

Proof. We show only the case for $out^f_\emptyset$. From right to left. We show that

$$x \in out^f_\emptyset(G, B\cup H) = Cn(out_\emptyset(G, B\cup H)\cap Cn(B\cup H)).$$

Since $x \in Cn(H)$, it suffices to show that $H \subseteq out_\emptyset(G, B\cup H)$ and $H \subseteq Cn(B \cup H)$. The latter is trivial. The former holds because $H \subseteq G(B) \subseteq out_\emptyset(G, B \cup H)$.

From left to right. Suppose x is manipulable. If $x \dashv\vdash \top$, we can let $B = H = \emptyset$. Assume $x \not\dashv\vdash \top$. Then $x \in out^f_\emptyset(G, A)$ for some consistent A. Let $B = b(G) \cap Cn(A)$ and $H = G(B) \cap Cn(A)$. It is clear that $B \cup H$ is consistent. To show that $x \in Cn(H)$, it suffices to show that $(out_\emptyset(G, A) \cap Cn(A)) \subseteq Cn(H)$. Let $y \in out_\emptyset(G, A) \cap Cn(A)$ and $y \not\dashv\vdash \top$. Then there is $z \in G(Cn(A))$ such that $z \dashv\vdash y$. Since $G(Cn(A)) = G(B)$, $z \in G(B)$. Since $z \in Cn(A)$, $z \in H$. Thus, $y \dashv\vdash z \in Cn(H)$. □

Proposition 35. *A formula x is manipulable under G and out^f_{AW} ($= out^f_{\text{W}}$) iff there is $B \subseteq b(G)$ such that $B \cup \{x\}$ is consistent and $x \in Cn(G(B))$.*

Proof. From right to left. It suffices to show that $x \in out^f_{\text{AW}}(G, B \cup \{x\})$. Since $B \cup \{x\}$ is consistent, we only need to show that

$$x \in Cn(Cn(G(Cn(B \cup \{x\}))) \cap Cn(B \cup \{x\})).$$

This holds because $x \in Cn(G(B))$ and $x \in Cn(B \cup \{x\})$. From left to right. Suppose x is manipulable. Then there exists A such that $x \in out^f_{\text{AW}}(G, A)$. The case where A is inconsistent is trivial. If A is consistent, then $x \in Cn(Cn(G(Cn(A))) \cap Cn(A))$. Let $B = \{a \in b(G) \mid a \in Cn(A)\}$. It is clear that $B \cup \{x\}$ is consistent (since $B \cup \{x\} \subseteq Cn(A)$) and $G(B) = G(Cn(A))$. Therefore, $x \in Cn(G(B))$. □

6 Related Work

Logic of Trust. Epistemic trust has been studied in the logical literature, e.g., (Liau 2003; Dastani et al. 2005; Jiang and Naumov 2022). Most of them studied unconditional trust using modal logic, whereas our paper generalizes trust to a conditional setting and employs the I/O logic framework. Compared with the modal logic approach, the I/O logic framework is more flexible in combining different inference rules. Nevertheless, the I/O language is also more restrictive (only propositional trust can be expressed). It can be expected that some of our I/O logics and filtering operations can be embedded into some modal logics of trust, e.g., in (Liau 2003).

The logics of trust are also studied in the belief revision context, e.g., (Booth and Hunter 2018) and (Singleton and Booth 2022). Booth and Hunter suggest representing agents' trust as a partition Π over the set of all valuations (states) on a given finite set of atoms. The semantic intuition is that the agent trusts the source to distinguish between states belonging to different cells in the partition Π. Given an informed statement φ, the filtered information by Π is semantically represented by $\Pi(\varphi) = \bigcup\{\Pi(s) \mid s \models \varphi\}$, where s is a state, $\Pi(s)$ is the element of Π containing s, and $\models$ is the satisfaction in classical propositional logic. We remark that this approach can be simulated using the operation out^f_{SD}. But, due to the space limitation, we cannot include the proof.

Belief Manipulation. Belief manipulation is concerned with the ability of one agent to convince another to hold certain beliefs (Hunter, Schwarzentruber, and Tsang 2017; Eiter, Hunter, and Schwarzentruber 2021). Similar notions, such as "knowability", are also studied in dynamic epistemic logic (Balbiani et al. 2008). Strictly speaking, our definition of "manipulable formulas" does not correspond to the exact notion of "belief manipulation", because in our framework, agents' belief states are not represented and no mechanism for belief change is included. Our emphasis is rather on manipulation in the process of trust-based information filtering. It can be imagined that, even if certain formulas cannot be manipulated in our information filtering architecture, one agent (the sender) is still able to make another (the receiver) believe the formulas. To model this, we need to extend our framework with, e.g., belief revision theory.

Input/Output Logic. I/O logic is originally introduced in (Makinson and Van Der Torre 2000). Although I/O logic is an influential framework for normative reasoning, it can also be applied to other domains, such as causal reasoning and nonmonotonic logic (Makinson and van der Torre 2001; Bochman 2005). In the literature, I/O logics without WO have been studied, see (Stolpe 2008) and (Parent and van der Torre 2019). But, in both of (Stolpe 2008) and (Parent and van der Torre 2019), the rule AND is included. I/O logics without WO and AND is considered in (Farjami 2020), but in an algebraic setting.

7 Conclusion and Future Work

Trust can be inferred from other information and plays the role of filtering out noisy information. In this paper, we first developed a family of new I/O logics for inferring trust and, then, presented an information filtering architecture based on trust inference. We also investigated the manipulation problem and we characterized the manipulable formulas in the information filtering architecture. Our main results (see Table 1) are about the redundancy of inference rules for trust: AND is redundant with respect to information-filtering mechanism and, furthermore, both AND and DT are redundant with respect to belief manipulation.

For future work, we plan to study the redundancy problem for other inference rules in I/O logic. We can also study a conditional notion of manipulability, because we may want to convince others to believe something while not revealing secret information. Finally, we can extend our architecture with belief revision operations.

$out_\emptyset$	out_{A}	out_{D}	out_{DA}	out_{S}	out_{SD}	out_{W}	out_{AW}
$out_\emptyset = out^f_{\text{A}}$		$out^f_{\text{D}} = out^f_{\text{DA}}$		out^f_{S}	out^f_{SD}	$out^f_{\text{W}} = out^f_{\text{AW}}$	
$out_\emptyset \equiv out^f_{\text{A}} \equiv out^f_{\text{D}} \equiv out^f_{\text{DA}}$				$out^f_{\text{S}} \equiv out^f_{\text{SD}}$		$out^f_{\text{W}} \equiv out^f_{\text{AW}}$	

Table 1: Summary of main results.

References

Alchourrón, C. E.; Gärdenfors, P.; and Makinson, D. 1985. On the Logic of Theory Change: Partial Meet Contraction and Revision Functions. *The Journal of Symbolic Logic*, 50(2): 510–530.

Balbiani, P.; Baltag, A.; van Ditmarsch, H.; Herzig, A.; Hoshi, T.; and de Lima, T. 2008. 'Knowable' as 'Known after an Announcement'. *The Review of Symbolic Logic*, 1(3): 305–334.

Becker, M. Y.; Russo, A.; and Sultana, N. 2012. Foundations of Logic-Based Trust Management. In *2012 IEEE Symposium on Security and Privacy*, 161–175.

Bochman, A. 2005. *Explanatory Nonmonotonic Reasoning*. WORLD SCIENTIFIC.

Booth, R.; and Hunter, A. 2018. Trust as a precursor to belief revision. *Journal of Artificial Intelligence Research*, 61: 699–722.

Dastani, M.; Herzig, A.; Hulstijn, J.; and van der Torre, L. 2005. Inferring Trust. In Leite, J.; and Torroni, P., eds., *Computational Logic in Multi-Agent Systems*, 144–160. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN 978-3-540-31857-6.

Eiter, T.; Hunter, A.; and Schwarzentruber, F. 2021. How Hard to Tell? Complexity of Belief Manipulation Through Propositional Announcements. In Zhou, Z.-H., ed., *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 1866–1872. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Farjami, A. 2020. *Discursive Input/output Logic: Deontic Modals, and Computation*. Ph.D. thesis, University of Luxembourg.

Hunter, A.; Schwarzentruber, F.; and Tsang, E. 2017. Belief Manipulation Through Propositional Announcements. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 1109–1115.

Jiang, J.; and Naumov, P. 2022. In Data We Trust: The Logic of Trust-Based Beliefs. In Raedt, L. D., ed., *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, 2683–2689. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Leturc, C.; and Bonnet, G. 2018. A Normal Modal Logic for Trust in the Sincerity. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '18, 175–183. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

Liau, C.-J. 2003. Belief, information acquisition, and trust in multi-agent systems – A modal logic formulation. *Artificial Intelligence*, 149(1): 31–60.

Makinson, D.; and Van Der Torre, L. 2000. Input/output logics. *Journal of philosophical logic*, 29: 383–408.

Makinson, D.; and van der Torre, L. 2001. Constraints for Input/Output Logics. *Journal of Philosophical Logic*, 30(2): 155–185.

Parent, X.; and van der Torre, L. 2019. Input/output logics without weakening. *Filosofiska Notise*, 6(1): 189–209.

Singleton, J.; and Booth, R. 2022. Who's the Expert? On Multi-source Belief Change. In *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning*, 331–340.

Stolpe, A. 2008. Normative Consequence: The Problem of Keeping It Whilst Giving It up. In van der Meyden, R.; and van der Torre, L., eds., *Deontic Logic in Computer Science*, 174–188. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN 978-3-540-70525-3.

van Ditmarsch, H.; van der Hoek, W.; and Kooi, B. 2008. *Dynamic Epistemic Logic*. Springer, Dordrecht.