On Bounding Problems of Quantitative Information Flow

Hirotoshi Yasuoka
Tohoku University
yasuoka@kb.ecei.tohoku.ac.jp

Tachio Terauchi
Nagoya University
terauchi@is.nagoya-u.ac.jp

February 2, 2012

Abstract

Researchers have proposed formal definitions of quantitative information flow based on information theoretic notions such as the Shannon entropy, the min entropy, the guessing entropy, belief, and channel capacity. This paper investigates the hardness of precisely checking the quantitative information flow of a program according to such definitions. More precisely, we study the "bounding problem" of quantitative information flow, defined as follows: Given a program M and a positive real number q, decide if the quantitative information flow of M is less than or equal to q. We prove that the bounding problem is not a k-safety property for any k (even when q is fixed, for the Shannon-entropy-based definition with the uniform distribution), and therefore is not amenable to the self-composition technique that has been successfully applied to checking non-interference. We also prove complexity theoretic hardness results for the case when the program is restricted to loop-free boolean programs. Specifically, we show that the problem is PP-hard for all definitions, showing a gap with non-interference which is coNP-complete for the same class of programs. The paper also compares the results with the recently proved results on the comparison problems of quantitative information flow.

Keywords: security, quantitative information flow, program verification

1 Introduction

We consider programs containing high security inputs and low security outputs. Informally, the quantitative information flow problem concerns the amount of information that an attacker can learn about the high security input by executing the program and observing the low security output. The problem is motivated by applications in information security. We refer to the classic by Denning [12] for an overview.

In essence, quantitative information flow measures *how* secure, or insecure, a program (or a part of a program –e.g., a variable–) is. Thus, unlike non-interference [10, 13], that only tells whether a program is completely secure or not completely secure, a definition of quantitative information flow must be able to distinguish two programs that are both interferent but have different degrees of "secureness."

For example, consider the following programs.

$$\begin{array}{l} M_1 \equiv \mathsf{if}\ H = g\ \mathsf{then}\ O\ :=\ 0\ \mathsf{else}\ O\ :=\ 1 \\ M_2 \equiv O\ :=\ H \end{array}$$

In both programs, H is a high security input and O is a low security output. Viewing H as a password, M_1 is a prototypical login program that checks if the guess g matches the password.[1] By executing M_1, an attacker only learns whether H is equal to g, whereas she would be able to learn the entire content of H by executing M_2. Hence, a reasonable definition of quantitative information flow should assign a higher quantity to M_2 than to M_1, whereas non-interference would merely say that M_1 and M_2 are both interferent, assuming that there are more than one possible values of H.

Researchers have attempted to formalize the definition of quantitative information flow by appealing to information theory. This has resulted in definitions based on the Shannon entropy [12, 7, 19], the min entropy [28], the guessing entropy [17, 1], belief [8], and channel capacity [22, 20, 26]. All of these definitions map a program (or a part of a program) onto a non-negative real number, that is, they define a function $\mathcal{X}$ such that given a program M, $\mathcal{X}(M)$ is a non-negative real number. (Concretely, $\mathcal{X}$ is $\mathit{SE}[\mu]$ for the Shannon-entropy-based definition with the distribution μ, $\mathit{ME}[\mu]$ for the min-entropy-based definition with the distribution μ, $\mathit{GE}[\mu]$ for the guessing-entropy-based definition with the distribution μ, and CC for the channel-capacity-based definition.[2]) Therefore, a natural verification problem for quantitative information flow is to decide, given M and a quantity $q \geq 0$, if $\mathcal{X}(M) \leq q$. The problem is well-studied for the case $q = 0$ as it is actually equivalent to checking non-interference (cf. Section 2.1). The problem is open for $q > 0$. We call this the *bounding problem* of quantitative information flow.

The problem has a practical relevance as a user is often interested in knowing if her program leaks information within some allowed bound. That is, the bounding problem is a form of quantitative information flow *checking* problem (as opposed to *inference*). Much of the previous research has focused on information theoretic properties of quantitative information flow and approximate (i.e., incomplete and/or unsound) algorithms for checking and inferring quantitative information flow. To fill the void, in a recent work [32], we have studied the hardness and possibilities of deciding the *comparison problem* of quantitative information flow, which is the problem of precisely checking if the information

[1] Here, for simplicity, we assume that g is a program constant. See Section 2 for modeling attacker/user (i.e., low security) inputs.

[2] The belief-based definition takes additional parameters as inputs, and is discussed below.

flow of one program is larger than that of the other, that is, the problem of deciding if $\mathcal{X}(M_1) \leq \mathcal{X}(M_2)$ given programs M_1 and M_2. The study has lead to some remarkable results, summarized in Section 3 and Section 4 of this paper to contrast with the new results on the bounding problem. However, the hardness results on the comparison problem do not imply hardness of the bounding problem.[3] Thus, this paper settles the open question.

We summarize the main results of the paper below. Here, $\mathcal{X}$ is $SE[U]$, $ME[U]$, $GE[U]$ or CC, where U is the uniform distribution.

- Checking if $\mathcal{X}(M) \leq q$ is not a k-safety property [29, 9] for any k.
- Restricted to loop-free boolean programs, checking if $\mathcal{X}(M) \leq q$ is PP-hard.

Roughly, a verification problem being k-safety means that it can be reduced to a standard safety problem, such as the unreachability problem, via self composition [3, 11]. For instance, non-interference is a 2-safety property (technically, for the termination-insensitive case[4]), and this has enabled its precise checking via a reduction to a safety problem via self composition and applying automated safety verification techniques [29, 25, 31]. Also, our recent work [32] has shown that deciding the comparison problem of quantitative information flow for all distributions (i.e., checking if $\forall\mu.SE[\mu](M_1) \leq SE[\mu](M_2)$, $\forall\mu.ME[\mu](M_1) \leq ME[\mu](M_2)$, $\forall\mu.GE[\mu](M_1) \leq GE[\mu](M_2)$, and $\forall\mu.\forall h, \ell.BE[\langle\mu, h, \ell\rangle](M_1) \leq BE[\langle\mu, h, \ell\rangle](M_2)$[5]) are 2-safety problems (and in fact, all equivalent).

We also prove a complexity theoretic gap with these related problems. We have shown in the previous paper [32] that, for loop-free boolean programs, both checking non-interference and the above comparison problem with universally quantified distributions are coNP-complete. (PP is believed to be strictly harder than coNP. In particular, coNP = PP implies the collapse of the polynomial hierarchy to level 1.)

Therefore, the results suggest that the bounding problems of quantitative information flow are harder than the related problems of checking non-interference and the quantitative information flow comparison problems with universally quantified distributions, and may require different techniques to solve (i.e., not self composition).

The belief-based quantitative information flow [8] differs from the definitions above in that it focuses on the information flow from a particular execution

[3] But, they imply the hardness of the inference problem because we can compare $\mathcal{X}(M_1)$ and $\mathcal{X}(M_2)$ once we have computed them. We also note that the hardness of the bounding problems implies that of the comparison problems because we can reduce the bounding problem $\mathcal{X}(M) \leq q$ to a comparison problem that compares M with a program whose information flow is q. (But, the reverse direction does not hold.)

[4] We restrict to terminating programs in this paper. (The termination assumption is nonrestrictive because we assume safety verification as a blackbox routine.)

[5] See below for the notation $BE[\langle\mu, h, \ell\rangle](M)$ denoting the belief-based quantitative information flow of M with respect to the experiment $\langle\mu, h, \ell\rangle$. The result for the belief-based definition is proven in the extended version of the paper that is under submission [33].

of the program (called *experiment*) rather than the information flow from all executions of the program.[6] Therefore, we define and study the hardness of two types of bounding problems for the belief-based definition:

(1) $BE[\langle \mu, h, \ell \rangle](M) \leq q$

(2) $\forall h, \ell . BE[\langle \mu, h, \ell \rangle](M) \leq q$

Here, $BE[\langle \mu, h, \ell \rangle](M)$ denotes the belief-based information flow of M with the experiment $\langle \mu, h, \ell \rangle$ where h, ℓ are the particular (high-security and low-security) inputs. Note that the problem (2) checks the bound of the belief-based quantitative information flow for *all* inputs whereas (1) checks the information flow for a particular input. This paper proves that neither of these problems are k-safety for any k, and are PP-hard for loop-free boolean programs.

We note that the above results are for the case the quantity q is taken to be an input to the bounding problems. We show that when fixing the parameter q constant, some of the problems become k-safety under certain conditions for different k's (cf. Section 3.1, 3.2, and 3.3).

We also define and study the hardness of the following bounding problems that check the bound over *all* distributions.

(1) $\forall \mu . SE[\mu](M) \leq q$

(2) $\forall \mu . ME[\mu](M) \leq q$

(3) $\forall \mu . GE[\mu](M) \leq q$

(4) $\forall \mu . BE[\langle \mu, h, \ell \rangle](M) \leq q$

(5) $\forall \mu, h, \ell . BE[\langle \mu, h, \ell \rangle](M) \leq q$

We show that except for (4) and (5), these problems are also not k-safety for any k, and are PP-hard for loop-free boolean programs, when q is not a constant (but are k-safety for various k's when q is held constant). For the problems (4) and (5), we show that the problems are actually equivalent to that of checking non-interference. (1), (2), and (3) are proven by showing that the problems correspond to various "channel capacity like" definitions of quantitative information flow.

The rest of the paper is organized as follows. Section 2 reviews the existing information-theoretic definitions of quantitative information flow and formally defines the bounding problems. Section 3 proves that the bounding problems are not k-safety problems for $SE[U]$, $ME[U]$, $GE[U]$, and CC. (Section 3.1 shows that when fixing the parameter q constant, some of them become k-safety under certain conditions for different k's.) Section 3.2 shows k-safety results for the belief-based bounding problems, and Section 3.3 shows k-safety results for the

[6] Clarkson et. al. [8] also propose a definition which averages the quantitative information flow over a distribution of the inputs h and ℓ. Note that a hardness result for (1) below implies the hardness result of the bounding problem for this problem as we may take the distribution to be a point mass.

bounding problems that check the bound for all distributions. Section 4 proves complexity theoretic hardness results for the bounding problems for loop-free boolean programs for $SE[U]$, $ME[U]$, $GE[U]$, and CC, and Section 4.1 proves those for the belief-based bounding problems and the bounding problems that check the bound for all distributions. Section 5 discusses some implications of the hardness results. Section 6 discusses related work, and Section 7 concludes. All the proofs appear in Appendix A.

2 Preliminaries

We introduce the information theoretic definitions of quantitative information flow that have been proposed in literature. First, we review the notion of the *Shannon entropy* [27], $\mathcal{H}[\mu](X)$, which is the average of the information content, and intuitively, denotes the uncertainty of the random variable X.

Definition 2.1 (Shannon Entropy). *Let X be a random variable with sample space $\mathbb{X}$ and μ be a probability distribution associated with X. (We write μ explicitly for clarity.) The Shannon entropy of X is defined as*

$$\mathcal{H}[\mu](X) = \sum_{x \in \mathbb{X}} \mu(X = x) \log \frac{1}{\mu(X = x)}$$

(The logarithm is in base 2.)

Next, we define *conditional entropy*. Informally, the conditional entropy of X given Y denotes the uncertainty of X after knowing Y.

Definition 2.2 (Conditional Entropy). *Let X and Y be random variables with sample spaces $\mathbb{X}$ and $\mathbb{Y}$, respectively, and μ be a probability distribution associated with X and Y. Then, the conditional entropy of X given Y, written $\mathcal{H}[\mu](X|Y)$ is defined as*

$$\mathcal{H}[\mu](X|Y) = \sum_{y \in \mathbb{Y}} \mu(Y = y) \mathcal{H}[\mu](X|Y = y)$$

where

$$\begin{array}{l} \mathcal{H}[\mu](X|Y = y) = \sum_{x \in \mathbb{X}} \mu(X = x|Y = y) \log \frac{1}{\mu(X=x|Y=y)} \\ \mu(X = x|Y = y) = \frac{\mu(X=x,Y=y)}{\mu(Y=y)} \end{array}$$

Next, we define (conditional) mutual information. Intuitively, the conditional mutual information of X and Y given Z represents the mutual dependence of X and Y after knowing Z.

Definition 2.3 (Mutual Information). *Let X, Y and Z be random variables and μ be an associated probability distribution.*[7] *Then, the conditional mutual information of X and Y given Z is defined as*

$$\begin{array}{rcl} \mathcal{I}[\mu](X; Y|Z) & = & \mathcal{H}[\mu](X|Z) - \mathcal{H}[\mu](X|Y, Z) \\ & = & \mathcal{H}[\mu](Y|Z) - \mathcal{H}[\mu](Y|X, Z) \end{array}$$

[7] We abbreviate the sample spaces of random variables when they are clear from the context.

Let M be a program that takes a high security input H and a low security input L, and gives the low security output O. For simplicity, we restrict to programs with just one variable of each kind, but it is trivial to extend the formalism to multiple variables (e.g., by letting the variables range over tuples). Also, for the purpose of the paper, unobservable (i.e., high security) outputs are irrelevant, and so we assume that the only program output is the low security output. Let μ be a probability distribution over the values of H and L. Then, the semantics of M can be defined by the following probability equation. (We restrict to terminating deterministic programs in this paper.)

$$\mu(O = o) = \sum_{\substack{h, \ell \in \mathbb{H}, \mathbb{L} \\ M(h,\ell) = o}} \mu(H = h, L = \ell)$$

Note that we write $M(h, \ell)$ to denote the low security output of the program M given inputs h and ℓ. Now, we are ready to introduce the Shannon-entropy based definition of quantitative information flow (QIF) [12, 7, 19].

Definition 2.4 (Shannon-Entropy-based QIF)**.** *Let M be a program with a high security input H, a low security input L, and a low security output O. Let μ be a distribution over H and L. Then, the Shannon-entropy-based quantitative information flow is defined*

$$\begin{array}{rcl} \mathit{SE}[\mu](M) & = & \mathcal{I}[\mu](O; H|L) \\ & = & \mathcal{H}[\mu](H|L) - \mathcal{H}[\mu](H|O, L) \end{array}$$

Intuitively, $\mathcal{H}[\mu](H|L)$ denotes the initial uncertainty knowing the low security input and $\mathcal{H}[\mu](H|O, L)$ denotes the remaining uncertainty after knowing the low security output.

As an example, consider the programs M_1 and M_2 from Section 1. For concreteness, assume that g is the value 01 and H ranges over the space $\{00, 01, 10, 11\}$. Let U be the uniform distribution over $\{00, 01, 10, 11\}$, that is, $U(h) = 1/4$ for all $h \in \{00, 01, 10, 11\}$. Computing their Shannon-entropy based quantitative information flow, we have,

$$\begin{array}{l} \mathit{SE}[U](M_1) = \mathcal{H}[U](H) - \mathcal{H}[U](H|O) = \log 4 - \frac{3}{4} \log 3 \approx .81128 \\ \mathit{SE}[U](M_2) = \mathcal{H}[U](H) - \mathcal{H}[U](H|O) = \log 4 - \log 1 = 2 \end{array}$$

Hence, if the user was to ask if $\mathit{SE}[U](M_1) \leq 1.0$, that is, "does M_1 leak more than one bit of information (according to $\mathit{SE}[U]$)?", then the answer would be no. But, for the same query, the answer would be yes for M_2.

Next, we introduce the *min entropy*, which Smith [28] recently suggested as an alternative measure for quantitative information flow.

Definition 2.5 (Min Entropy)**.** *Let X and Y be random variables, and μ be an associated probability distribution. Then, the min entropy of X is defined*

$$\mathcal{H}_\infty[\mu](X) = \log \frac{1}{\mathcal{V}[\mu](X)}$$

and the conditional min entropy of X given Y is defined

$$\mathcal{H}_\infty[\mu](X|Y) = \log \frac{1}{\mathcal{V}[\mu](X|Y)}$$

where

$$\begin{array}{rcl} \mathcal{V}[\mu](X) & = & \max_{x \in \mathbb{X}} \mu(X = x) \\ \mathcal{V}[\mu](X|Y = y) & = & \max_{x \in \mathbb{X}} \mu(X = x|Y = y) \\ \mathcal{V}[\mu](X|Y) & = & \sum_{y \in \mathbb{Y}} \mu(Y = y)\mathcal{V}[\mu](X|Y = y) \end{array}$$

Intuitively, $\mathcal{V}[\mu](X)$ represents the highest probability that an attacker guesses X in a single try. We now define the min-entropy-based definition of quantitative information flow.

Definition 2.6 (Min-Entropy-based QIF)**.** *Let M be a program with a high security input H, a low security input L, and a low security output O. Let μ be a distribution over H and L. Then, the min-entropy-based quantitative information flow is defined*

$$ME[\mu](M) = \mathcal{H}_\infty[\mu](H|L) - \mathcal{H}_\infty[\mu](H|O, L)$$

Whereas Smith [28] focused on programs lacking low security inputs, we extend the definition to programs with low security inputs in the definition above. It is easy to see that our definition coincides with Smith's for programs without low security inputs. Also, the extension is arguably natural in the sense that we simply take the conditional entropy with respect to the distribution over the low security inputs.

Computing the min-entropy based quantitative information flow for our running example programs M_1 and M_2 from Section 1 with the uniform distribution, we obtain,

$$\begin{array}{l} ME[U](M_1) = \mathcal{H}_\infty[U](H) - \mathcal{H}_\infty[U](H|O) = \log 4 - \log 2 = 1 \\ ME[U](M_2) = \mathcal{H}_\infty[U](H) - \mathcal{H}_\infty[U](H|O) = \log 4 - \log 1 = 2 \end{array}$$

Hence, if a user is to check whether $ME[U]$ is bounded by q for $1 \leq q < 2$, then the answer would be yes for M_1, but no for M_2.

Next, we introduce the *guessing-entropy* based definition of quantitative information flow [21, 17, 1].

Definition 2.7 (Guessing Entropy)**.** *Let X and Y be random variables, and μ be an associated probability distribution. Then, the guessing entropy of X is defined*

$$\mathcal{G}[\mu](X) = \sum_{1 \leq i \leq m} i \times \mu(X = x_i)$$

where $m = |\mathbb{X}|$ and $x_1, x_2, \ldots, x_m$ satisfies $\forall i, j. i \leq j \Rightarrow \mu(X = x_i) \geq \mu(X = x_j)$.

The conditional guessing entropy of X given Y is defined

$$\mathcal{G}[\mu](X|Y) = \sum_{y \in \mathbb{Y}} \mu(Y = y)\mathcal{G}[\mu](X|Y = y)$$

where

$$\begin{array}{c} \mathcal{G}[\mu](X|Y=y) = \sum_{1 \leq i \leq m} i \times \mu(X=x_i|Y=y) \\ m = |\mathbb{X}| \ \ \textit{and} \ \ \forall i,j.i \leq j \Rightarrow \mu(X=x_i|Y=y) \geq \mu(X=x_j|Y=y) \end{array}$$

Intuitively, $\mathcal{G}[\mu](X)$ represents the average number of times required for the attacker to guess the value of X. We now define the guessing-entropy-based quantitative information flow.

Definition 2.8 (Guessing-Entropy-based QIF)**.** *Let M be a program with a high security input H, a low security input L, and a low security output O. Let μ be a distribution over H and L. Then, the guessing-entropy-based quantitative information flow is defined*

$$GE[\mu](M) = \mathcal{G}[\mu](H|L) - \mathcal{G}[\mu](H|O,L)$$

Like with the min-entropy-based definition, the previous research on guessing-entropy-based quantitative information flow only considered programs without low security inputs [17, 1]. But, it is easy to see that our definition with low security inputs coincides with the previous definitions for programs without low security inputs. Also, as with the extension for the min-entropy-based definition, it simply takes the conditional entropy over the low security inputs.

We test GE on the running example from Section 1 by calculating the quantities for the programs M_1 and M_2 with the uniform distribution.

$$\begin{array}{c} GE[U](M_1) = \mathcal{G}[U](H) - \mathcal{G}[U](H|O) = \frac{5}{2} - \frac{7}{4} = 0.75 \\ GE[U](M_2) = \mathcal{G}[U](H) - \mathcal{G}[U](H|O) = \frac{5}{2} - 1 = 1.5 \end{array}$$

Hence, if a user is to check whether $GE[U]$ is bounded by q for $0.75 \leq q < 1.5$, then the answer would be yes for M_1, but no for M_2.

Next, we introduce the belief-based definition of quantitative information flow [8]. The belief-based definition computes the information leak from a single execution of the program, called an *experiment*.

Definition 2.9 (Experiment)**.** *Let μ be a distribution over a high-security input such that $\forall h.\mu(h) > 0$, $h_{\mathcal{E}}$ be a high-security input, and $\ell_{\mathcal{E}}$ be a low-security input. Then, the experiment $\mathcal{E}$ is defined to be the tuple $\langle \mu, h_{\mathcal{E}}, \ell_{\mathcal{E}} \rangle$.*[8]

Intuitively, the distribution μ represents the attacker's *belief* about the user's high security input selection, $\ell_{\mathcal{E}}$ denotes the attacker's low-security input selection, and $h_{\mathcal{E}}$ denotes the user's actual selection. Then, the belief-based quantitative information flow, which is the information flow of individual experiments, is defined as follows.

Definition 2.10 (Belief-based QIF)**.** *Let M be a program with a high security input, a low security input, and a low security output. Let $\mathcal{E}$ be an experiment*

[8] Clarkson et. al. [8] also include the output and the program itself as part of the experiment. In this paper, an experiment consists solely of the input and the distribution.

such that $\mathcal{E} = \langle \mu, h_{\mathcal{E}}, \ell_{\mathcal{E}} \rangle$. Then, the belief-based quantitative information flow is defined

$$BE[\mathcal{E}](M) = D(\mu \to \dot{h_{\mathcal{E}}}) - D(\mu|o_{\mathcal{E}} \to \dot{h_{\mathcal{E}}})$$

where

$$\begin{array}{l}
o_{\mathcal{E}} = M(h_{\mathcal{E}}, \ell_{\mathcal{E}}) \\
\dot{h} = \lambda h'.\mathsf{if}\ h = h'\ \mathsf{then}\ 1\ \mathsf{else}\ 0 \\
\mu_{\ell_{\mathcal{E}}}(o_{\mathcal{E}}) = \sum_{h \in \{h' | M(h', \ell_{\mathcal{E}}) = o_{\mathcal{E}}\}} \mu(h) \\
\mu|o_{\mathcal{E}} = \lambda h.\mathsf{if}\ M(h, \ell_{\mathcal{E}}) = o_{\mathcal{E}}\ \mathsf{then}\ \frac{\mu(h)}{\mu_{\ell_{\mathcal{E}}}(o_{\mathcal{E}})}\ \mathsf{else}\ 0 \\
D(\mu \to \mu') = \sum_h \mu'(h) \log \frac{\mu'(h)}{\mu(h)}
\end{array}$$

Here, $D(\mu \to \mu')$ is the *relative entropy* (or, *distance*) of μ and μ', and quantifies the difference between the two distributions.[9] Note that $\dot{h}$ denotes the point mass distribution at h. Intuitively, the belief-based quantitative information flow expresses the difference between the attacker's belief about the high security input and the output of the experiment. It can be shown that $BE[\mathcal{E}](M)$ is equivalent to *self-information* (for M deterministic), that is, the negative logarithm of the probability the event occurs (i.e., in this case, the output occurs).

Lemma 2.11. *Let μ be a belief, $h_{\mathcal{E}}$ be a high-security input, $\ell_{\mathcal{E}}$ be a low-security input. Then, $BE[\langle \mu, h_{\mathcal{E}}, \ell_{\mathcal{E}} \rangle](M) = -\log \Sigma_{h \in \{h' | M(h', \ell_{\mathcal{E}}) = M(h_{\mathcal{E}}, \ell_{\mathcal{E}})\}} \mu(h)$.*

Computing the belief-based quantitative information flow for our running example programs M_1 and M_2 from Section 1 with the uniform distribution, we obtain,

- $h \in \{00, 10, 11\}$

$$BE[\langle U, h \rangle](M_1) = -\log U(M_1(h)) = -\log \frac{3}{4} \approx .41503$$

- $h = 01$

$$BE[\langle U, h \rangle](M_1) = -\log U(M_1(h)) = -\log \frac{1}{4} = 2$$

And, for any $h \in \{00, 01, 10, 11\}$,

$$BE[\langle U, h \rangle](M_2) = -\log U(M_2(h)) = -\log \frac{1}{4} = 2$$

Therefore, if the user was to ask if $BE[\langle U, h \rangle]$ is bounded by 1.0 for $h = 00$, then the answer would be yes for M_1 but no for M_2. But, if the user was to ask if $BE[\langle U, h \rangle]$ is bounded by 1.0 for all h, then the answer would be no for both M_1 and M_2.

Finally, we introduce the definition of quantitative information flow based on *channel capacity* [22, 20, 26], which is defined to be the maximum of the Shannon-entropy based quantitative information flow over the distribution.

[9] Here, we follow [8] and use the notation $D(\mu \to \mu')$ over the more standard notation $D(\mu' || \mu)$.

Definition 2.12 (Channel-Capacity-based QIF). *Let M be a program with a high security input H, a low security input L, and a low security output O. Then, the channel-capacity-based quantitative information flow is defined*

$$CC(M) = \max_{\mu} \mathcal{I}[\mu](O; H|L)$$

Unlike the other definitions above, the channel-capacity based definition of quantitative information flow is not parameterized by the distribution over the inputs. As with the other definitions, let us test the definition on the running example from Section 1 by calculating the quantities for the programs M_1 and M_2:

$$\begin{array}{l} CC(M_1) = \max_{\mu} \mathcal{I}[\mu](O; H) = 1 \\ CC(M_2) = \max_{\mu} \mathcal{I}[\mu](O; H) = 2 \end{array}$$

Note that $CC(M_1)$ (resp. $CC(M_2)$) is equal to $ME[U](M_1)$ (resp. $ME[U](M_2)$). This is not a coincidence. In fact, it is known that $CC(M) = ME[U](M)$ for all programs M without low security inputs [28].

2.1 Non-interference

We recall the notion of non-interference [10, 13].

Definition 2.13 (Non-intereference). *A program M is said to be non-interferent iff for any $h, h' \in \mathbb{H}$ and $\ell \in \mathbb{L}$, $M(h, \ell) = M(h', \ell)$.*

It can be shown that for the definitions of quantitative information flow $\mathcal{X}$ introduced above, $\mathcal{X}(M) \leq 0$ iff M is non-interferent.[10] That is, the bounding problem (which we only officially define for positive bounds –see Section 2.2–) degenerates to checking non-interference when 0 is given as the bound.

Theorem 2.14. *Let μ be a distribution such that $\forall h \in \mathbb{H}, \ell \in \mathbb{L}. \mu(h, \ell) > 0$. Then,*

- *M is non-interferent if and only if $SE[\mu](M) \leq 0$.*
- *M is non-interferent if and only if $ME[\mu](M) \leq 0$.*
- *M is non-interferent if and only if $GE[\mu](M) \leq 0$.*
- *M is non-interferent if and only if $BE[\langle \mu', h, \ell \rangle](M) \leq 0$.*[11]
- *M is non-interferent if and only if $CC(M) \leq 0$.*

The equivalence result on the Shannon-entropy-based definition is proven by Clark et al. [6]. The proofs for the other four definitions are given in Appendix A.

[10] Technically, we need the non-zero-ness condition on the distribution. (See below.)

[11] Recall Definition 2.10 that μ' is a distribution over $\mathbb{H}$ such that $\mu'(h) > 0$ for all $h \in \mathbb{H}$.

2.2 Bounding Problem

We define the *bounding problem* of quantitative information flow for each definition introduced above. The bounding problem for the Shannon-entropy based definition $B_{SE}[\mu]$ is defined as follows: Given a program M and a positive real number q, decide if $SE[\mu](M) \leq q$.[12] Similarly, we define the bounding problems for the other three definitions $B_{ME}[\mu]$, $B_{GE}[\mu]$, and B_{CC} as follows.

$$\begin{array}{rcl} B_{ME}[\mu] & = & \{(M,q) \mid ME[\mu](M) \leq q\} \\ B_{GE}[\mu] & = & \{(M,q) \mid GE[\mu](M) \leq q\} \\ B_{CC} & = & \{(M,q) \mid CC(M) \leq q\} \end{array}$$

We defer the definitions of the belief-based bounding problems to Section 3.2.

3 K-Safety Property

We show that none of the bounding problems are k-safety problems for any k. Informally, a program property is said to be a *k-safety* property [29, 9] if it can be refuted by observing k number of (finite) execution traces. A k-safety problem is the problem of checking a k-safety property. Note that the standard safety property is a 1-safety property. An important property of a k-safety problem is that it can be reduced to a standard safety (i.e., 1-safety) problem, such as the unreachability problem, via a simple program transformation called *self composition* [3, 11]. This allows one to verify k-safety problems by applying powerful automated safety verification techniques [2, 14, 24, 4] that have made remarkable progress recently.

As stated earlier, we prove that no bounding problem is a k-safety property for any k. (First, we prove the result for SE, ME, GE, and CC, and defer the result for BE to Section 3.2.) To put the result in perspective, we compare it to the results of the related problems, summarized below. Here, $\mathcal{X}$ is $SE[U]$, $ME[U]$, $GE[U]$, or CC, and $\mathcal{Y}$ is SE, ME, or GE. (Recall that U denotes the uniform distribution.)

(1) Checking non-interference is a 2-safety problem, but it is not 1-safety.

(2) Checking $\mathcal{X}(M_1) \leq \mathcal{X}(M_2)$ is not a k-safety problem for any k.

(3) Checking $\forall \mu . \mathcal{Y}[\mu](M_1) \leq \mathcal{Y}[\mu](M_2)$ is a 2-safety problem.

The result (1) on non-interference is classic (see, e.g., [23, 3, 11]). The results (2) and (3) on comparison problems are proven in our recent paper [32]. Therefore, this section's results imply that the bounding problems are harder to verify (at least, via the self-composition approach) than non-interference and the quantitative information flow comparison problems with universally quantified distributions.

[12] Note that we treat μ as a parameter of the bounding problem rather than as an input.

Let *Prog* be the set of all programs, and $\mathbb{R}^+$ be the set of positive real numbers. Let $[\![M]\!]$ denote the semantics (i.e., traces) of M, represented by the set of input/output pairs, that is, $[\![M]\!] = \{((h,\ell),o) \mid h \in \mathbb{H}, \ell \in \mathbb{L}, o = M(h,\ell)\}$. Then, formally, k-safety property is defined as follows.

Definition 3.1 (k-safety property)**.** *We say that a property $P \subseteq Prog \times \mathbb{R}^+$ is a k-safety property iff $(M,q) \notin P$ implies that there exists $T \subseteq [\![M]\!]$ such that $|T| \leq k$ and $\forall M'.T \subseteq [\![M']\!] \Rightarrow (M',q) \notin P$.*

Note that the original definition of k-safety property is only defined over programs [29, 9]. However, because the bounding problems take the additional input q, we extend the notion to account for the extra parameter.

We now state the main results of this section which show that none of the bounding problems are k-safety problems for any k. Because we are interested in hardness, we focus on the case where the distribution is the uniform distribution. That is, the results we prove for the specific case applies to the general case.

Theorem 3.2. *Neither $B_{SE}[U]$, $B_{ME}[U]$, $B_{GE}[U]$, nor B_{CC} is a k-safety property for any k such that $k > 0$.*

The result follows from the fact that for each of bounding problem $B_{\mathcal{X}}$ above, for any k, there exists q such that deciding $(M,q) \in B_{\mathcal{X}}$ is not a k-safety property. In fact, as we show next, for some of the problems such as $B_{SE}[U]$, even if we fix q to an arbitrary constant, there exists no k such that the problem is k-safety. (But for other problems, for certain cases, we can find k that depends on q.) We defer the details to the next section. (See also Section 5.2.)

3.1 K-Safety Under a Constant Bound

The result above appears to suggest that the bounding problems are equally difficult for $SE[U]$, $ME[U]$, $GE[U]$, and CC. However, holding the parameter q constant (rather than having it as an input) paints a different picture. We show that the problems become k-safety for different definitions for different k's under different conditions in this case.

First, for q fixed, we show that the bounding problem for the channel-capacity based definition of quantitative information flow is k-safety for $k = \lfloor 2^q \rfloor + 1$. (Also, this bound is tight.)

Theorem 3.3. *Let q be a constant. Then, B_{CC} is $\lfloor 2^q \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor 2^q \rfloor$.*

We briefly explain the intuition behind the above result. Recall that a problem being k-safety means the existence of a *counterexample* trace set of size at most k. That is, for $(M,q) \notin B_{CC}$, we have $T \subseteq [\![M]\!]$ such that $|T| \leq \lfloor 2^q \rfloor + 1$ such that any program that also contains T as its traces also does not belong to B_{CC} (with q), that is, its channel-capacity-based quantitative information flow is greater than q. Then, the above result follows from the fact that the channel-capacity-based quantitative information flow coincides with the maximum over the low security inputs of the logarithm of the number of outputs [20], therefore, any T containing $\lfloor 2^q \rfloor + 1$ traces of the same low security input and disjoint outputs is a counterexample.

For concreteness, we show how to check B_{CC} via self composition. Suppose we are given a program M and a positive real q. We construct the self-composed program M' shown below.

$$
\begin{array}{l}
M'(H_1, H_2, \ldots, H_n, L) \equiv \\
\quad O_1 := M(H_1, L); O_2 := M(H_2, L); \ldots ; O_n := M(H_n, L); \\
\quad \mathsf{assert}(\bigvee_{i,j \in \{1,\ldots,n\}} (O_i = O_j \wedge i \neq j))
\end{array}
$$

where $n = \lfloor 2^q \rfloor + 1$. In general, a self composition involves making k copies the original program so that the resulting program would generate k traces of the original (having the desired property). By the result proven by Malacaria and Chen [20](see also Lemma A.8), it follows that M' does not cause an assertion failure iff $(M, q) \in B_{CC}$.

Next, we show that for programs without low security inputs, $B_{ME}[U]$ and $B_{GE}[U]$ are also both k-safety problems (but for different k's) when q is held constant.

Theorem 3.4. *Let q be a constant, and suppose $B_{ME}[U]$ only takes programs without low security inputs. Then, $B_{ME}[U]$ is $\lfloor 2^q \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor 2^q \rfloor$.*

Theorem 3.5. *Let q be a constant, and suppose $B_{GE}[U]$ only takes programs without low security inputs. If $q \geq \frac{1}{2}$, then, $B_{GE}[U]$ is $\lfloor \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rfloor$. Otherwise, $q < \frac{1}{2}$ and $B_{GE}[U]$ is 2-safety, but it is not 1-safety.*

The result for $ME[U]$ follows from the fact that for programs without low security inputs, the min-entropy based quantitative information flow with the uniform distribution is actually equivalent to the channel-capacity based quantitative information flow [28]. The result for $GE[U]$ may appear less intuitive, but, the key observation is that, like the channel-capacity based definition and the min-entropy based definition with the uniform distribution (for the case without low security inputs), for any set of traces $T = [\![M]\!]$, the information flow of a program containing T would be at least as large as that of M. Therefore, by holding q constant, we can always find a large enough counterexample T. The reason $B_{GE}[U]$ is 2-safety for $q < \frac{1}{2}$ is because, in the absence of low security inputs, the minimum non-zero quantity of $GE[U](M)$ is bounded (by 1/2), and so for such q, the problem $GE[U](M) \leq q$ is equivalent to checking non-interference.[13]

But, when low security inputs are allowed, neither $B_{ME}[U]$ nor $B_{GE}[U]$ are k-safety for any k, even when q is held constant.

[13] In fact, the minimum non-zero quantity property also exists for *ME*[U] without low security inputs and *CC*. There, the minimum non-zero quantity is 1, which agrees with the formulas given in the theorems.

Theorem 3.6. *Let q be a constant. (And let $B_{ME}[U]$ take programs with low security inputs.) Then, $B_{ME}[U]$ is not a k-safety property for any $k > 0$.*

Theorem 3.7. *Let q be a constant. (And let $B_{GE}[U]$ take programs with low security inputs.) Then, $B_{GE}[U]$ is not a k-safety property for any $k > 0$.*

Finally, we show that the Shannon-entropy based definition (with the uniform distribution) is the hardest of all the definitions and show that its bounding problem is not a k-safety property for any k, with or without low-security inputs, even when q is held constant.

Theorem 3.8. *Let q be a constant, and suppose $B_{SE}[U]$ only takes programs without low security inputs. Then, $B_{SE}[U]$ is not a k-safety property for any $k > 0$.*

Intuitively, Theorems 3.6, 3.7, and 3.8 follow from the fact that, for these definitions, given any potential counterexample $T \subseteq [\![M]\!]$ to show $(M, q) \notin B_{\mathcal{X}}$, it is possible to find M' containing T whose information flow is arbitrarily close to 0 (and so $(M', q) \in B_{\mathcal{X}}$). See Section 5.2 for further discussion.

Because k tends to grow large as q grows for all the definitions and it is impossible to bound k for all q, this section's results are unlikely to lead to a practical verification of quantitative information flow. [14] Nevertheless, the results reveal interesting disparities among the different proposals for the definition of quantitative information flow.

3.2 K-Safety for Belief-based Definition

This section investigates the hardness of the bounding problems for the belief-based definition of quantitative information flow. We define two types of bounding problems.

$$\begin{array}{rcl} B_{BE1}[\langle \mu, h, \ell \rangle] & = & \{(M, q) \mid BE[\langle \mu, h, \ell \rangle](M) \le q\} \\ B_{BE2}[\mu] & = & \{(M, q) \mid \forall h, \ell . BE[\langle \mu, h, \ell \rangle](M) \le q\} \end{array}$$

B_{BE1} checks the program's information flow against the given quantity for a specific input pair h, ℓ whereas B_{BE2} checks that for all inputs.

We show that these problems are not a k-safety problems for any k, at least when q is not a constant. To put the result in perspective, we compare to the results of the comparison problem for the belief-based quantitative information flow problem [33].

(1) Checking $BE[\langle U, h, \ell \rangle](M_1) \le BE[\langle U, h, \ell \rangle](M_2)$ is not a k-safety problem for any k.

(2) Checking $\forall h, \ell . BE[\langle U, h, \ell \rangle](M_1) \le BE[\langle U, h, \ell \rangle](M_2)$ is not a k-safety problem for any k.

[14] But, a recent work [16] shows some promising results.

(3) Checking $\forall \mu, h, \ell. BE[\langle \mu, h, \ell\rangle](M_1) \leq BE[\langle \mu, h, \ell\rangle](M_2)$ is a 2-safety problem.

Note that the problem in (3) compares the two programs for *all* experiments $\langle \mu, h, \ell\rangle$. This problem also turns out to be equivalent to the comparison problems with universally quantified distributions for *SE*, *ME*, and *GE* discussed in Section 3. Hence, this section's non-k-safety results show that the bounding problems B_{BE1} and B_{BE2} are harder to verify (at least, via the self-composition approach) than non-interference and the comparison problems with universally quantified distributions and experiments.

First, we show that $B_{BE1}[\langle U, h, \ell\rangle]$ is not a k-safety property for any k, even when q is held constant, and even without low security inputs.

Theorem 3.9. *Let q be a constant, and suppose $B_{BE1}[\langle U, h\rangle]$ only takes programs without low security inputs. Then, $B_{BE1}[\langle U, h\rangle]$ is not a k-safety property for any $k > 0$.*

Next, we show that $B_{BE2}[U]$ is also not a k-safety property for any k when q is a constant and $q \geq 1$, even without low security inputs. But, when q is held constant and $q < 1$, $B_{BE2}[U]$ is a 2-safety property.

Theorem 3.10. *Let q be a constant. If $q \geq 1$, then $B_{BE2}[U]$ is not a k-safety property for any $k > 0$ even when $B_{BE2}[U]$ only takes programs without low security inputs. Otherwise, $q < 1$ and $B_{BE2}[U]$ is a 2-safety property, but it is not a 1-safety property.*

The 2-safety property for the case $q < 1$ follows because $B_{BE2}[U]$ turns out to be equivalent to non-interference for such q. The results show that the bounding problems for the belief-based definition is also quite hard, except for the case where one checks if the information flow is less than 1 for all inputs, which degenerates to checking non-interference.

3.3 K-Safety for Channel Capacity Like Definitions

In this section, we study the hardness of the bounding problems that check the bound for all distributions. We define the following problems.

$$\begin{array}{lcl}
B_{SECC} & = & \{(M, q) \mid \forall \mu. SE[\mu](M) \leq q\} \\
B_{MECC} & = & \{(M, q) \mid \forall \mu. ME[\mu](M) \leq q\} \\
B_{GECC} & = & \{(M, q) \mid \forall \mu. GE[\mu](M) \leq q\} \\
B_{BE1CC}[h, \ell] & = & \{(M, q) \mid \forall \mu. BE[\langle \mu, h, \ell\rangle](M) \leq q\} \\
B_{BE2CC} & = & \{(M, q) \mid \forall \mu. \forall h, \ell. BE[\langle \mu, h, \ell\rangle](M) \leq q\}
\end{array}$$

Note that $B_{SECC} = B_{CC}$ because $CC(M) = \max_\mu SE[\mu](M)$. For this reason, we call these bounding problems "channel capacity like." For instance, Köpf and Smith [18] call $\max_\mu ME[\mu](M)$ the *min-entropy channel capacity.* (Note that $(M, q) \in B_{MECC}$ iff $\max_\mu ME[\mu](M) \leq q$.) B_{GECC} follows the same spirit. We define two types of channel-capacity like problems for the belief-based definition corresponding to the two types of bounding problems B_{BE1} and B_{BE2}.

We prove k-safety results for each of these problems. The result below for B_{SECC} follows directly from that of B_{CC} (i.e., Theorem 3.3). But, the other results proved are new.

Theorem 3.11. *Let q be a constant. Then, B_{SECC} is $\lfloor 2^q \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor 2^q \rfloor$.*

First, we show that B_{MECC} enjoys the same property as B_{SECC}. That is, when q is held constant, it is $\lfloor 2^q \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor 2^q \rfloor$. Note that unlike $B_{ME}[U]$, this holds even for programs with low security inputs. We show this by proving the following lemma stating that $\max_\mu ME[\mu]$ is actually equivalent to $CC(M)$.

Lemma 3.12. $\max_\mu ME[\mu](M) = CC(M)$

The lemma extends the result by Braun et al. [5] that shows the equivalence for the low-security-input-free case. By the lemma, the k-safety result for B_{MECC} follows directly from that of B_{CC}.

Theorem 3.13. *Let q be a constant. Then, B_{MECC} is $\lfloor 2^q \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor 2^q \rfloor$.*

Next, we prove that, when q is held constant, B_{GECC} is k-safety for $k = \lfloor \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rfloor + 1$ when $q \geq \frac{1}{2}$ and is 2-safety for $q < \frac{1}{2}$. Recall that these k-safety bounds are equivalent to those of $B_{GE}[U]$ without low security inputs (cf. Theorem 3.5). However, unlike $B_{GE}[U]$, the k-safety result here holds even for programs with low security inputs.

Theorem 3.14. *Let q be a constant. If $q \geq \frac{1}{2}$, then, B_{GECC} is $\lfloor \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rfloor$. Otherwise, $q < \frac{1}{2}$ and B_{GECC} is 2-safety, but it is not 1-safety.*

The above is shown by proving the following lemma which states that the "guessing entropy channel capacity" $\max_\mu GE[\mu]$ is actually equivalent to $\max_\ell GE[U \otimes \dot{\ell}]$. (See below for the definition of $U \otimes \dot{\ell}$.)

Lemma 3.15. *We have $\max_\mu GE[\mu](M) = \max_{\ell'} GE[U \otimes \dot{\ell}'](M)$ where $U \otimes \dot{\ell}'$ denotes $\lambda h, \ell.\mathsf{if}\ \ell = \ell'\ \mathsf{then}\ U(h)\ \mathsf{else}\ 0$.*

Finally, we prove somewhat surprising results for $B_{BE1CC}[h, \ell]$ and B_{BE2CC} stating that they are in fact equivalent to non-interference, independent of q. It follows that these problems are 2-safety but not 1-safety.

Theorem 3.16. *$(M, q) \in B_{BE1CC}[h, \ell]$ iff $M(\ell)$ is non-interferent.*

Here, $M(\ell) = \lambda h.M(h, \ell)$. That is, the theorem states that, for any q, $(M, q) \in B_{BE1CC}[h, \ell]$ iff the program M restricted to the low security input ℓ is non-interferent. (Note that checking non-interference at a fixed low security input is also a 2-safety property and is not a 1-safety property.)

An analogous result holds for B_{BE2CC}.

Theorem 3.17. *$(M, q) \in B_{BE2CC}$ iff M is non-interferent.*

Clarkson et al. [9] also studies B_{BE2CC}, which they call QL in their paper.[15] They state that the problem is a *hypersafety* property, which is a superset of k-safety properties.[16]

4 Complexities for Loop-free Boolean Programs

In this section, we analyze the computational complexity of the bounding problems when the programs are restricted to loop-free boolean programs. We compare the complexity theoretic hardness of the bounding problems with those of the related problems for the same class of programs, as we have done with the k-safety property of the problems.

That is, we compare against the comparison problems of quantitative information flow and the problem of checking non-interference for loop-free boolean programs. The complexity results for these problems are summarized below. Here, $\mathcal{X}$ is $SE[U]$, $ME[U]$, $GE[U]$, or CC, and $\mathcal{Y}$ is SE, ME, or GE.

(1) Checking non-interference is coNP-complete

(2) Checking $\mathcal{X}(M_1) \leq \mathcal{X}(M_2)$ is PP-hard.

(3) Checking $\forall \mu . \mathcal{Y}[\mu](M_1) \leq \mathcal{Y}[\mu](M_2)$ is coNP-complete.

The results (1) and (3) are proven in our recent paper [32]. The result (2) is proven in the extended version of the paper [33] and tightens our (oracle relative) #P-hardness result from the conference version [32], which states that for each C such that C is the comparison problem for $SE[U]$, $ME[U]$, $GE[U]$, or CC, we have $\#\mathrm{P} \subseteq \mathrm{FP}^{C}$. (Recall that the notation FP^{A} means the complexity class of function problems solvable in polynomial time with an oracle for the problem A.) #P is the class of counting problems associated with NP. PP is the class of decision problems solvable in probabilistic polynomial time. PP is known to contain both coNP and NP, $\mathrm{PH} \subseteq \mathrm{P}^{\mathrm{PP}} = \mathrm{P}^{\#\mathrm{P}}$ [30], and PP is believed to be strictly larger than both coNP and NP. (In particular, PP = coNP would imply the collapse of the polynomial hierarchy (PH) to level 1.)

We show that, restricted to loop-free boolean programs, the bounding problems for the Shannon-entropy-based, the min-entropy-based, and the guessing-entropy-based definition of quantitative information flow with the uniform distribution (i.e., $SE[U]$, $ME[U]$, and $GE[U]$) and the channel-capacity based definition (i.e., CC) are all PP-hard. (The results for the belief-based definition and the channel-capacity-like definitions appear in Section 4.1.) The results strengthen the hypothesis that the bounding problems for these definitions are quite hard. Indeed, they show that they are complexity theoretically harder

[15] Technically, they allow an experiment to consist of a sequence of runs of the program whereas we restrict an experiment to a single run.

[16] Informally, a property is a hypersafety if there exists a counterexample set of traces of *any* size.

than non-interference and the comparison problems with the universally quantified distributions for loop-free boolean programs, assuming that coNP and PP are separate.

We define the syntax of loop-free boolean programs in Figure 1. We assume the usual derived formulas $\phi \Rightarrow \psi$, $\phi = \psi$, $\phi \vee \psi$, and false. We give the usual weakest precondition semantics in Figure 2.

To adapt the information flow framework to boolean programs, we make each information flow variable H, L, and O range over functions mapping boolean variables of its kind to boolean values. For example, if x and y are low security boolean variables and z is a high security boolean variable, then L ranges over the functions $\{x, y\} \rightarrow \{\mathsf{false}, \mathsf{true}\}$, and H and O range over $\{z\} \rightarrow \{\mathsf{false}, \mathsf{true}\}$.[17] (Every boolean variable is either a low security boolean variable or a high security boolean variable.) We write $M(h, \ell) = o$ for an input (h, ℓ) and an output o if $(h, \ell) \models \mathit{wp}(M, \phi)$ for a boolean formula ϕ such that $o \models \phi$ and $o' \not\models \phi$ for all output $o' \neq o$. Here, $\models$ is the usual logical satisfaction relation, using h, ℓ, o, etc. to look up the values of the boolean variables. (Note that this incurs two levels of lookup.)

As an example, consider the following program.

$$M \equiv z := x; w := y; \mathsf{if}\ x \wedge y\ \mathsf{then}\ z := \neg z\ \mathsf{else}\ w := \neg w$$

Let x, y be high security variables and z, w be low security variables. Then,

$$\begin{array}{rcl rcl} \mathit{SE}[U](M) & = & 1.5 & \mathit{GE}[U](M) & = & 1.25 \\ \mathit{ME}[U](M) & = & \log 3 \approx 1.5849625 & \mathit{CC}(M) & = & \log 3 \approx 1.5849625 \end{array}$$

We now state the main results of the section, which show that the bounding problems for $\mathit{SE}[U]$, $\mathit{ME}[U]$, $\mathit{GE}[U]$, and CC are PP-hard.

Theorem 4.1. $\mathit{PP} \subseteq B_{\mathit{SE}}[U]$

Theorem 4.2. $\mathit{PP} \subseteq B_{\mathit{ME}}[U]$

Theorem 4.3. $\mathit{PP} \subseteq B_{\mathit{GE}}[U]$

Theorem 4.4. $\mathit{PP} \subseteq B_{\mathit{CC}}$

We remind that the above results hold (even) when the bounding problems $B_{\mathit{SE}}[U]$, $B_{\mathit{ME}}[U]$, $B_{\mathit{GE}}[U]$, and B_{CC} are restricted to loop-free boolean programs. We also note that the results hold even when the programs are restricted to those without low security inputs. These results are proven by a reduction from MAJSAT, which is a PP-complete problem. MAJSAT is the problem of deciding, given a boolean formula ϕ over variables $\overrightarrow{x}$, if there are more than $2^{|\overrightarrow{x}|-1}$ satisfying assignments to ϕ (i.e., whether the majority of the assignments to ϕ are satisfying).

[17] We do not distinguish input boolean variables from output boolean variables. But, a boolean variable can be made output-only by assigning a constant to the variable at the start of the program and made input-only by assigning a constant at the end.

4.1 Complexities for Belief and Channel Capacity Like Definitions

This section investigates the complexity theoretic hardness of the bounding problems for the belief-based definition and the channel-capacity-like definition of quantitative information flow introduced in Section 3.2 and Section 3.3. As in Section 4, we focus on loop-free boolean programs.

Below shows the complexity results for the belief-based comparison problems for loop-free boolean programs [33].

(1) Checking $BE[\langle U,h,\ell\rangle](M_1) \leq BE[\langle U,h,\ell\rangle](M_2)$ is PP-hard.

(2) Checking $\forall h,\ell.BE[\langle U,h,\ell\rangle](M_1) \leq BE[\langle U,h,\ell\rangle](M_2)$ is PP-hard.

(3) Checking $\forall \mu,h,\ell.BE[\langle \mu,h,\ell\rangle](M_1) \leq BE[\langle \mu,h,\ell\rangle](M_2)$ is coNP-complete.

First, we prove that the two types of bounding problems for the belief-based definition, B_{BE1} and B_{BE2}, are both PP-hard.

Theorem 4.5. *$PP \subseteq B_{BE1}[\langle U,h,\ell\rangle]$*

Theorem 4.6. *$PP \subseteq B_{BE2}[U]$*

As in Section 4, the above theorems are proven by a reduction from MAJSAT. They show that the bounding problems for $BE[U]$ are complexity theoretically difficult.

Next, we prove the hardness results for the channel-capacity like definitions of quantitative information flow. Theorems 4.7 and 4.8 for B_{SECC} and B_{MECC} follow from the equivalence $\max_\mu SE[\mu](M) = \max_\mu ME[\mu](M) = CC(M)$ (cf. Section 3.3) and Theorem 4.4. Theorem 4.9 for B_{GECC} follows from Theorem 4.3 and the equivalence $\max_\mu GE[\mu](M) = \max_\ell GE[U \otimes \dot{\ell}](M)$ (cf. Lemma 3.15).

Theorem 4.7. *$PP \subseteq B_{SECC}$*

Theorem 4.8. *$PP \subseteq B_{MECC}$*

Theorem 4.9. *$PP \subseteq B_{GECC}$*

Finally, the following coNP-completeness results for $B_{BE1CC}[h,\ell]$ and B_{BE2CC} follow from their equivalent to non-interference and the fact that checking non-interference is coNP-complete for loop-free boolean programs (cf. Section 4).

Theorem 4.10. *$B_{BE1CC}[h,\ell]$ is coNP-complete.*

Theorem 4.11. *B_{BE2CC} is coNP-complete.*

5 Discussion

5.1 Bounding the Domains

The notion of k-safety property, like the notion of safety property from where it extends, is defined over all programs regardless of their size. (For example, non-interference is a 2-safety property for all programs and unreachability is a safety property for all programs.) But, it is easy to show that the bounding problems would become "k-safety" properties if we constrained and bounded the input domains because then the size of the semantics (i.e., the input/output pairs) of such programs would be bounded by $|\mathbb{H}|\times|\mathbb{L}|$. In this case, the problems are at most $|\mathbb{H}|\times|\mathbb{L}|$-safety. (And the complexity theoretic hardness degenerates to a constant.) But, like the k-safety bounds obtained by fixing q constant (cf. Section 3.1), these bounds are high for all but very small domains and are unlikely to lead to a practical verification method. Also, because a bound on the high security input domain puts a bound on the maximum information flow, the bounding problems become a tautology for $q \geq c$, where c is the maximum information flow for the respective definition.

5.2 Low Security Inputs

Recall the results from Section 3.1 that, under a constant bound, the bounding problems for both the min-entropy based definition and the guessing-entropy based definition with the uniform distribution are k-safety for programs without low security inputs, but not for those with. The reason for the non-k-safety results is that the definitions of quantitative information flow ME and GE (and in fact, also SE) use the conditional entropy over the low security input distribution and are parameterized by the distribution. This means that the quantitative information flow of a program is averaged over the low security inputs according to the distribution. Therefore, by arbitrarily increasing the number of low security inputs, given any set of traces T, it becomes possible to find a program containing T whose information flow is arbitrarily close to 0 (at least under the uniform distribution). This appears to be a property intrinsic to any definition of quantitative information flow defined via conditional entropy over the low security inputs and is parameterized by the distribution of low security inputs. Note that the channel-capacity-like definitions do not share this property as it is defined to be the maximum over the distributions. The non-k-safety result for $B_{SE}[U]$ holds even in the absence of low security inputs because the Shannon entropy of a program is the average of the *surprisal* [8] of the individual observations, and so by increasing the number of high security inputs, given any set of traces T, it becomes possible to find a program containing T whose information flow is arbitrarily close to 0. The non-k-safety results for $B_{BE1}[\langle U, h\rangle]$ and $B_{BE2}[U]$ hold for similar reasons.[18]

[18] They are, respectively, the surprisal of a particular input, and the maximum surprisal over all the inputs.

6 Related Work

This work continues our recent research [32] on investigating the hardness and possibilities of verifying quantitative information flow according to the formal definitions proposed in literature [8, 12, 7, 19, 28, 17, 1, 22, 20, 26, 5, 18]. Much of the previous research has focused on information theoretic properties of the definitions and proposed approximate (i.e., incomplete and/or unsound) methods for checking and inferring quantitative information flow according to such definitions. In contrast, this paper (along with our recent paper [32]) investigates the hardness and possibilities of precisely checking and inferring quantitative information flow according to the definitions.

This paper has shown that the bounding problem, that is, the problem of checking $\mathcal{X}(M) \leq q$ given a program M and a positive real q, is quite hard (for various quantitative information flow definitions $\mathcal{X}$). This is in contrast to our previous paper that has investigated the hardness and possibilities of the comparison problem, that is, the problem of checking $\mathcal{X}(M_1) \leq \mathcal{X}(M_2)$ given programs M_1 and M_2. To the best of our knowledge, this paper is the first to investigate the hardness of the bounding problems. But, the hardness of quantitative information flow inference, a harder problem, follows from the results of our previous paper, and Backes et al. [1] and also Heusser and Malacaria [15] have proposed a precise inference method that utilizes self composition and counting algorithms. Also, independently from our work, Heusser and Malacaria [16] have recently applied the self-composition method outlined in Section 3.1 for checking the channel-capacity-based quantitative information flow.

7 Conclusion

In this paper, we have formalized and proved the hardness of the bounding problem of quantitative information flow, which is a form of (precise) checking problem of quantitative information flow. We have shown that no bounding problem is a k-safety property for any k, and therefore that it is not possible to reduce the problem to a safety problem via self composition, at least when the quantity to check against is unrestricted. The result is in contrast to non-interference and the quantitative information flow comparison problem with universally quantified distribution, which are 2-safety properties. We have also shown a complexity theoretic gap with these problems, which are coNP-complete, by proving the PP-hardness of the bounding problems, when restricted to loop-free boolean programs.

We have also shown that the bounding problems for some quantitative information flow definitions become k-safety for different k's under certain conditions when the quantity to check against is restricted to be a constant, highlighting interesting disparities among the different definitions of quantitative information flow.

It is interesting to note that, as with the comparison problems, the bounding problems become comparatively easier when the input distribution becomes universally quantified. That is, as our previous work [32] has shown that checking if $\forall\mu.\mathcal{Y}[\mu](M_1) \leq \mathcal{Y}[\mu](M_2)$ is often easier than checking if $\mathcal{Y}[U](M_1) \leq \mathcal{Y}[U](M_2)$ (for various quantitative information flow definitions $\mathcal{Y}$), we have shown that the problem of checking $\forall\mu.\mathcal{Y}[\mu](M) \leq q$ is often easier than the problem of checking $\mathcal{Y}[U](M) \leq q$.

Acknowledgments

This work was supported by MEXT KAKENHI 23700026, 22300005, and the Global COE Program "CERIES."

References

[1] M. Backes, B. Köpf, and A. Rybalchenko. Automatic discovery and quantification of information leaks. In *30th IEEE Symposium on Security and Privacy, S&P 2009*, pages 141–153. IEEE Computer Society, May 2009.

[2] T. Ball and S. K. Rajamani. The SLAM project: debugging system software via static analysis. In *Proceedings of the 29th ACM SIGPLAN-SIGACT symposium on Principles of programming languages, POPL 2002*, pages 1–3. ACM, January 2002.

[3] G. Barthe, P. R. D'Argenio, and T. Rezk. Secure information flow by self-composition. In *17th IEEE Computer Security Foundations Workshop, CSFW 2004*, pages 100–114. IEEE Computer Society, June 2004.

[4] D. Beyer, T. A. Henzinger, R. Jhala, and R. Majumdar. The software model checker Blast. *International Journal on Software Tools for Technology Transfer, STTT*, 9(5-6):505–525, 2007.

[5] C. Braun, K. Chatzikokolakis, and C. Palamidessi. Quantitative notions of leakage for one-try attacks. *Electron. Notes Theor. Comput. Sci.*, 249:75–91, August 2009.

[6] D. Clark, S. Hunt, and P. Malacaria. Quantified interference for a while language. *Electr. Notes Theor. Comput. Sci.*, 112:149–166, January 2005.

[7] D. Clark, S. Hunt, and P. Malacaria. A static analysis for quantifying information flow in a simple imperative language. *Journal of Computer Security*, 15(3):321–371, August 2007.

[8] M. R. Clarkson, A. C. Myers, and F. B. Schneider. Belief in information flow. In *18th IEEE Computer Security Foundations Workshop, CSFW 2005*, pages 31–45. IEEE Computer Society, June 2005.

[9] M. R. Clarkson and F. B. Schneider. Hyperproperties. In *Proceedings of the 21st IEEE Computer Security Foundations Symposium, CSF 2008*, pages 51–65. IEEE Computer Society, June 2008.

[10] E. S. Cohen. Information transmission in computational systems. In *Proceedings of the Sixth Symposium on Operating System Principles, SOSP 1977*, pages 133–139. ACM, November 1977.

[11] Á. Darvas, R. Hähnle, and D. Sands. A theorem proving approach to analysis of secure information flow. In *Security in Pervasive Computing, Second International Conference, SPC 2005*, volume 3450 of *Lecture Notes in Computer Science*, pages 193–209. Springer, April 2005.

[12] D. E. R. Denning. *Cryptography and data security.* Addison-Wesley Longman Publishing Co., Inc., 1982.

[13] J. A. Goguen and J. Meseguer. Security policies and security models. In *Proceedings of the IEEE Symposium on Security and Privacy, S&P 1982*, pages 11–20. IEEE Computer Society, April 1982.

[14] T. A. Henzinger, R. Jhala, R. Majumdar, and G. Sutre. Lazy abstraction. In *Proceedings of the 29th ACM SIGPLAN-SIGACT symposium on Principles of programming languages, POPL 2002*, pages 58–70. ACM, January 2002.

[15] J. Heusser and P. Malacaria. Applied quantitative information flow and statistical databases. In *Formal Aspects in Security and Trust, 6th International Workshop, FAST 2009, Revised Selected Papers*, volume 5983 of *Lecture Notes in Computer Science*, pages 96–110. Springer, November 2009.

[16] J. Heusser and P. Malacaria. Quantifying information leaks in software. In *Twenty-Sixth Annual Computer Security Applications Conference, ACSAC 2010*, pages 261–269. ACM, December 2010.

[17] B. Köpf and D. Basin. An information-theoretic model for adaptive side-channel attacks. In *Proceedings of the 14th ACM conference on Computer and communications security*, CCS 2007, pages 286–296. ACM, October 2007.

[18] B. Köpf and G. Smith. Vulnerability bounds and leakage resilience of blinded cryptography under timing attacks. In *Proceedings of the 23rd IEEE Computer Security Foundations Symposium, CSF 2010*, pages 44–56. IEEE Computer Society, July 2010.

[19] P. Malacaria. Assessing security threats of looping constructs. In *Proceedings of the 34th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2007*, pages 225–235. ACM, January 2007.

[20] P. Malacaria and H. Chen. Lagrange multipliers and maximum information leakage in different observational models. In *Proceedings of the third ACM SIGPLAN workshop on Programming languages and analysis for security*, PLAS 2008, pages 135–146. ACM, June 2008.

[21] J. L. Massey. Guessing and entropy. In *In Proceedings of the 1994 IEEE International Symposium on Information Theory*, page 204, 1994.

[22] S. McCamant and M. D. Ernst. Quantitative information flow as network flow capacity. In *Proceedings of the ACM SIGPLAN 2008 Conference on Programming Language Design and Implementation, PLDI 2008*, pages 193–205. ACM, June 2008.

[23] J. McLean. A general theory of composition for trace sets closed under selective interleaving functions. In *IEEE Symposium on Security and Privacy, S&P 1994*, pages 79–93. IEEE Computer Society, May 1994.

[24] K. L. McMillan. Lazy abstraction with interpolants. In *Computer Aided Verification, 18th International Conference, CAV 2006*, volume 4144 of *Lecture Notes in Computer Science*, pages 123–136. Springer, August 2006.

[25] D. A. Naumann. From coupling relations to mated invariants for checking information flow. In *Proceedings of the 11th European Symposium on Research in Computer Security, ESORICS 2006*, volume 4189 of *Lecture Notes in Computer Science*, pages 279–296. Springer, September 2006.

[26] J. Newsome, S. McCamant, and D. Song. Measuring channel capacity to distinguish undue influence. In *Proceedings of the ACM SIGPLAN Fourth Workshop on Programming Languages and Analysis for Security*, PLAS 2009, pages 73–85. ACM, June 2009.

[27] C. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[28] G. Smith. On the foundations of quantitative information flow. In *Proceedings of the 12th International Conference on Foundations of Software Science and Computational Structures, FOSSACS 2009*, volume 5504, pages 288–302. Springer-Verlag, March 2009.

[29] T. Terauchi and A. Aiken. Secure information flow as a safety problem. In *Proceedings of the 12th International Symposium on Static Analysis, SAS 2005*, volume 3672 of *Lecture Notes in Computer Science*, pages 352–367. Springer, September 2005.

[30] S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991.

[31] H. Unno, N. Kobayashi, and A. Yonezawa. Combining type-based analysis and model checking for finding counterexamples against non-interference. In *Proceedings of the 2006 Workshop on Programming Languages and Analysis for Security, PLAS 2006*, pages 17–26. ACM, June 2006.

[32] H. Yasuoka and T. Terauchi. Quantitative information flow - verification hardness and possibilities. In *Proceedings of the 23rd IEEE Computer Security Foundations Symposium, CSF 2010*, pages 15–27. IEEE Computer Society, July 2010.

[33] H. Yasuoka and T. Terauchi. Quantitative information flow - verification hardness and possibilities (extended version). 2010. In submission.

A Proofs

We define some abbreviations.

Definition A.1. $\mu(x) \triangleq \mu(X = x)$

We use the above notation whenever the correspondences between random variables and their values are clear.

We define some useful abbreviations for programs having low security inputs.

Definition A.2. $M[\mathbb{H}, \ell] = \{o \mid \exists h \in \mathbb{H}.o = M(h, \ell)\}$

Definition A.3. $M(\ell) = \lambda h.M(h, \ell)$

Note that $M(\ell)$ is the program M restricted to the low security input ℓ, and that $M[\mathbb{H}, \ell]$ is the set of outputs of $M(\ell)$.

We elide the parameter q from the input to the bounding problems when it is clear from the context (e.g., when q is held constant). For example, we write $B_{SE}[U](M)$ and $M \in B_{SE}[U]$ instead of $B_{SE}[U](M, q)$ or $(M, q) \in B_{SE}[U]$.

We note the following properties of deterministic programs [6].

Lemma A.4. *Let M be a program without low-security inputs, M' be a program with low-security inputs. Then, we have $SE[\mu](M) = \mathcal{I}[\mu](O; H) = \mathcal{H}[\mu](O)$ and $SE[\mu](M') = \mathcal{I}[\mu](O; H|L) = \mathcal{H}[\mu](O|L)$*

Definition A.5.

$$In(\mu, X, x) = |\{x' \in X \mid \mu(x') \geq \mu(x)\}|$$

Intuitively, $In(\mu, X, x)$ is the order of x defined in terms of μ.

Lemma A.6.

$$\mathcal{G}[\mu](X) = \Sigma_{1 \leq i \leq |X|} i\mu(x_i) = \Sigma_{x \in X} In(\mu, X, x)\mu(x)$$

Proof. Trivial. □

Lemma 2.11. *Let μ be a belief, $h_{\mathcal{E}}$ be a high-security input, $\ell_{\mathcal{E}}$ be a low-security input. Then, $BE[\langle \mu, h_{\mathcal{E}}, \ell_{\mathcal{E}} \rangle](M) = -\log \Sigma_{h \in \{h' \mid M(h', \ell_{\mathcal{E}}) = M(h_{\mathcal{E}}, \ell_{\mathcal{E}})\}} \mu(h)$.*

Proof. By definition, we have

$$\begin{aligned}
&BE[\langle \mu, h_{\mathcal{E}}, \ell_{\mathcal{E}} \rangle](M) \\
&\quad = D(\mu \to \dot{h}_{\mathcal{E}}) - D(\mu | o_{\mathcal{E}} \to \dot{h}_{\mathcal{E}}) \\
&\quad = \textstyle\sum_h \dot{h}_{\mathcal{E}}(h) \log \frac{\dot{h}_{\mathcal{E}}(h)}{\mu(h)} - \sum_h \dot{h}_{\mathcal{E}}(h) \log \frac{\dot{h}_{\mathcal{E}}(h)}{\mu | o_{\mathcal{E}}(h)} \\
&\quad = \log \frac{1}{\mu(h_{\mathcal{E}})} + \log \frac{\mu(h_{\mathcal{E}})}{\sum_{h \in \{h' | M(h', \ell_{\mathcal{E}}) = M(h_{\mathcal{E}}, \ell_{\mathcal{E}})\}} \mu(h)} \\
&\quad = -\log \textstyle\sum_{h \in \{h' | M(h', \ell_{\mathcal{E}}) = M(h_{\mathcal{E}}, \ell_{\mathcal{E}})\}} \mu(h)
\end{aligned}$$

□

Theorem 2.14. *Let μ be a distribution such that $\forall h \in \mathbb{H}, \ell \in \mathbb{L}. \mu(h, \ell) > 0$. Then,*

- *M is non-interferent if and only if $SE[\mu](M) \leq 0$.*
- *M is non-interferent if and only if $ME[\mu](M) \leq 0$.*
- *M is non-interferent if and only if $GE[\mu](M) \leq 0$.*
- *M is non-interferent if and only if $BE[\langle \mu', h, \ell \rangle](M) \leq 0$.*[19]
- *M is non-interferent if and only if $CC(M) \leq 0$.*

Proof. Let $\mathbb{O} = \{M(h, \ell) \mid h \in \mathbb{H} \wedge \ell \in \mathbb{L}\}$.

- *SE*

  (See [6].)

- *ME*

  - $\Rightarrow$

    Suppose M is non-interferent. By the definition, it suffices to show that
    $$\mathcal{V}[\mu](H|L) = \mathcal{V}[\mu](H|L, O)$$

    That is,

    $$\sum_{\ell} \mu(\ell) \max_h \mu(h|\ell) = \sum_{\ell, o} \mu(\ell, o) \max_h \mu(h|\ell, o)$$

    We have for any ℓ_x and o_x such that $\mu(\ell_x, o_x) > 0$, $\mu(\ell_x, o_x) = \mu(\ell_x)$, and for all h_y, ℓ_y, and o_y such that $\mu(h_y, \ell_y, o_y) > 0$, for any h'_y and $o' \in \mathbb{O} \setminus \{o_y\}$, $\mu(h'_y, \ell_y, o'_y) = 0$. Therefore, we have

    $$\begin{array}{rcl}
    \sum_{\ell, o} \mu(\ell, o) \max_h \mu(h|\ell, o) & = & \sum_{\ell, o} \mu(\ell, o) \max_h \frac{\mu(h, \ell, o)}{\mu(\ell, o)} \\
    & = & \sum_{\ell} \mu(\ell) \max_h \mu(h|\ell)
    \end{array}$$

[19] Recall Definition 2.10 that μ' is a distribution over $\mathbb{H}$ such that $\mu'(h) > 0$ for all $h \in \mathbb{H}$.

  - ⇐

    We prove the contraposition. Suppose M is interferent. That is, there exist h_1, h_2, and ℓ' such that $M(h_1, \ell') \neq M(h_2, \ell')$. Let $o_1 = M(h_1, \ell')$ and $o_2 = M(h_2, \ell')$. We have

    $$\sum_{\ell} \mu(\ell) \max_{h} \mu(h|\ell) = A + \max_{h} \mu(h, \ell')$$

    where $A = \sum_{\ell \in \mathbb{L} \setminus \{\ell'\}} \max_h \mu(h, \ell)$. And,

    $$\sum_{\ell, o} \mu(\ell, o) \max_{h} \mu(h|\ell, o) = B + \sum_{o} \max_{h} \mu(h, \ell', o)$$

    where $B = \sum_{(\ell, o) \in (\mathbb{L} \setminus \{\ell'\}) \times \mathbb{O}} \max_h \mu(h, \ell, o)$. Trivially, we have $A \leq B$ and

    $$\max_{h} \mu(h, \ell') < \sum_{o} \max_{h} \mu(h, \ell', o)$$

    Therefore, we have $ME[\mu](M) > 0$.

- *GE*

  - ⇒

    Suppose M is non-interferent. By the definition,

    $$\begin{aligned}
    &GE[\mu](M) \\
    &\quad = \textstyle\sum_{\ell} \sum_{h} In(\lambda h'.\mu(h', \ell), \mathbb{H}, h)\mu(h, \ell) \\
    &\qquad - \textstyle\sum_{\ell, o} \sum_{h} In(\lambda h'.\mu(h', \ell, o), \mathbb{H}, h)\mu(h, \ell, o) \\
    &\quad = \textstyle\sum_{\ell} \sum_{h} In(\lambda h'.\mu(h', \ell), \mathbb{H}, h)\mu(h, \ell) \\
    &\qquad - \textstyle\sum_{\ell} \sum_{h} In(\lambda h'.\mu(h', \ell), \mathbb{H}, h)\mu(h, \ell) \\
    &\quad = 0
    \end{aligned}$$

    since for all h_x, ℓ_x, and o_x such that $\mu(h_x, \ell_x, o_x) > 0$, for any h'_x and $o' \in \mathbb{O} \setminus \{o_x\}$, $\mu(h'_x, \ell_x, o'_x) = 0$.

  - ⇐

    We prove the contraposition. Suppose M is interferent. That is, there exist h_1, h_2, and ℓ' such that $M(h_1, \ell') \neq M(h_2, \ell')$. Let $o_1 = M(h_1, \ell')$ and $o_2 = M(h_2, \ell')$. By the definition,

    $$\begin{aligned}
    &GE[\mu](M) \\
    &\quad = \textstyle\sum_{\ell} \sum_{h} In(\lambda h'.\mu(h', \ell), \mathbb{H}, h)\mu(h, \ell) \\
    &\qquad - \textstyle\sum_{\ell, o} \sum_{h} In(\lambda h'.\mu(h', \ell, o), \mathbb{H}, h)\mu(h, \ell, o) \\
    &\quad = A + \textstyle\sum_{h} In(\lambda h'.\mu(h', \ell'), \mathbb{H}, h)\mu(h, \ell') \\
    &\qquad - B - \textstyle\sum_{o} \sum_{h} In(\lambda h'.\mu(h', \ell', o), \mathbb{H}, h)\mu(h, \ell', o)
    \end{aligned}$$

    where

    $$\begin{aligned}
    A &= \textstyle\sum_{\ell \in \mathbb{L} \setminus \{\ell'\}} \sum_{h} In(\lambda h'.\mu(h', \ell'), \mathbb{H}, h)\mu(h, \ell') \\
    B &= \textstyle\sum_{(\ell, o) \in (\mathbb{L} \setminus \{\ell'\}) \times \mathbb{O}} \sum_{h} In(\lambda h'.\mu(h', \ell', o), \mathbb{H}, h)\mu(h, \ell', o)
    \end{aligned}$$

Trivially, we have $A \geq B$ and

$$\begin{array}{l}\sum_h In(\lambda h'.\mu(h',\ell'),\mathbb{H},h)\mu(h,\ell') \\ \quad > \sum_o \sum_h In(\lambda h'.\mu(h',\ell',o),\mathbb{H},h)\mu(h,\ell',o)\end{array}$$

Therefore, we have $GE[\mu](M) > 0$.

- *BE*
  - $\Rightarrow$

    Suppose M is non-interferent. By Lemma 2.11, for any μ, h, and ℓ,

    $$BE[\langle\mu,h,\ell\rangle](M) = -\log \Sigma_{h' \in \{h'' | M(h'',\ell)=M(h,\ell)\}}\mu(h') = 0$$

  - $\Leftarrow$

    We prove the contraposition. Suppose M is interferent. That is, there exist h_1, h_2, and ℓ' such that $M(h_1,\ell') \neq M(h_2,\ell')$. Let μ' be a distribution such that for any h', $\mu'(h') > 0$. Then, by Lemma 2.11, we have for any h,

    $$BE[\langle\mu',h,\ell'\rangle](M) = -\log \Sigma_{h' \in \{h'' | M(h'',\ell')=M(h,\ell')\}}\mu'(h') > 0$$

- *CC*
  - $\Rightarrow$

    Suppose M is non-interferent. By Lemma A.4, for any μ,

    $$\begin{array}{rcl} SE[\mu](M) & = & \mathcal{H}[\mu](O|L) \\ & = & \sum_o \sum_\ell \mu(o,\ell)\log\frac{\mu(\ell)}{\mu(o,\ell)} \\ & = & 0 \end{array}$$

    since $\mu(o,\ell) = \mu(\ell)$. Therefore, we have $\forall\mu.SE[\mu](M) = 0$. It follows that $CC(M) = 0$.

  - $\Leftarrow$

    We prove the contraposition. Suppose M is interferent. That is, there exist h_1, h_2, and ℓ' such that $M(h_1,\ell') \neq M(h_2,\ell')$. Let $o_1 = M(h_1,\ell')$, and $o_2 = M(h_2,\ell')$. Then, there exist μ' such that

    $$\begin{array}{rcl} SE[\mu'](M) & = & \mathcal{H}[\mu'](O|L) \\ & \geq & \mu'(o_1,\ell')\log\frac{\mu'(\ell')}{\mu'(o_1,\ell')} + \mu'(o_2,\ell')\log\frac{\mu'(\ell')}{\mu'(o_2,\ell')} \\ & > & 0 \end{array}$$

    And, we have $SE[\mu'](M) \leq CC(M)$.

□

We note the following equivalence of *CC* and *ME*[U] for programs without low security inputs [28].

Lemma A.7. *Let M be a program without low security input. Then, $CC(M) = ME[U](M)$.*

Theorem 3.2. *Neither $B_{SE}[U]$, $B_{ME}[U]$, $B_{GE}[U]$, nor B_{CC} is a k-safety property for any k such that $k > 0$.*

Proof.

- $B_{SE}[U]$ is not a k-safety problem for any k such that $k > 0$.
  Trivial by Theorem 3.8.
- $B_{ME}[U]$ is not a k-safety property for any k such that $k > 0$.
  Trivial by Theorem 3.4.
- $B_{GE}[U]$ is not a k-safety property for any k such that $k > 0$.
  Trivial by Theorem 3.5.
- B_{CC} is not a k-safety property for any k such that $k > 0$.
  Trivial from Lemma A.7 and the fact that $B_{ME}[U]$ is not a k-safety property for any k.

□

Malacaria and Chen [20] have proved the following result relating the channel-capacity based quantitative information flow with the number of outputs.

Lemma A.8. *Let M be a program (with low security input). Then,*

$$CC(M) = \max_{\ell \in \mathbb{L}} \log |M[\mathbb{H}, \ell]|$$

Theorem 3.3. *Let q be a constant. Then, B_{CC} is $\lfloor 2^q \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor 2^q \rfloor$.*

Proof. We prove that B_{CC} is $\lfloor 2^q \rfloor + 1$-safety. Let M be a program such that $M \notin B_{CC}$. By Lemma A.8, it must be the case that there exists ℓ such that $|M[\mathbb{H}, \ell]| \geq \lfloor 2^q \rfloor + 1$. Then, there exists $T \subseteq [\![M]\!]$ such that $|T| \leq \lfloor 2^q \rfloor + 1$, $ran(T) \geq \lfloor 2^q \rfloor + 1$, and for all $((h, \ell'), o) \in T$, $\ell' = \ell$. Then, by Lemma A.8, it follows that for any program M' such that $T \subseteq [\![M']\!]$, $M' \notin B_{CC}$. Therefore, B_{CC} is a $\lfloor 2^q \rfloor + 1$-safety property.

Finally, we prove that $B_{CC}[U]$ is not k-safety for any $k \leq \lfloor 2^q \rfloor$. Let $k \leq \lfloor 2^q \rfloor$. For a contradiction, suppose B_{CC} is a k-safety property. Let M be a program such that $M \notin B_{CC}$. Then, there exists T such that $|T| \leq k$ and $T \subseteq [\![M]\!]$, and for any M' such that $T \subseteq [\![M']\!]$, $(M', q) \notin B_{CC}$. Let $T = \{(h_1, o_1), \ldots, (h_i, o_i)\}$. Let $\bar{M}$ be a program such that $[\![\bar{M}]\!] = T$. More formally, let $\bar{M}$ be the following program.

$$\bar{M}(h_1) = o_1, \bar{M}(h_2) = o_2, \ldots, \bar{M}(h_i) = o_i$$

Then, we have

$$CC(\bar{M}) = \log |\{o_1, o_2, \ldots, o_i\}| \leq \log k \leq q$$

It follows that $(\bar{M}, q) \in CC$, but $T \subseteq [\![\bar{M}]\!]$. Therefore, this leads to a contradiction. □

Theorem 3.4. *Let q be a constant, and suppose $B_{ME}[U]$ only takes programs without low security inputs. Then, $B_{ME}[U]$ is $\lfloor 2^q \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor 2^q \rfloor$.*

Proof. Straightforward by Theorem 3.3 and Lemma A.7. □

Lemma A.9. *Let M be a program without low security inputs. Then, we have $GE[U](M) = \frac{n}{2} - \frac{1}{2n} \sum_o |\mathbb{H}_o|^2$ where n is the number of inputs, and $\mathbb{H}_o = \{h \mid o = M(h)\}$.*

Proof. By the definition, we have

$$\begin{array}{rcl} GE[U](M) & = & \mathcal{G}[U](H) - \mathcal{G}[U](H|O) \\ & = & \sum_h In(U, \mathbb{H}, h)U(h) \\ & & \quad - \sum_o U(o) \sum_h In(\lambda h'.U(h'|o), \mathbb{H}_o, h)U(h|o) \\ & = & \frac{1}{n}\frac{1}{2}n(n+1) - \sum_o \frac{|\mathbb{H}_o|}{n} \frac{1}{2} \frac{1}{|\mathbb{H}_o|} |\mathbb{H}_o|(|\mathbb{H}_o| + 1) \\ & = & \frac{n}{2} - \frac{1}{2n} \sum_o |\mathbb{H}_o|^2 \end{array}$$

□

Lemma A.10. *Let M and M' be low-security input free programs such that $[\![M']\!] = [\![M]\!] \cup \{(h, o)\}$ and $h \notin dom([\![M]\!])$. Then, we have $GE[U](M) \leq GE[U](M')$.*

Proof. We prove $GE[U](M') - GE[U](M) \geq 0$. Let $n = |[\![M]\!]|$, $\mathbb{O} = ran([\![M]\!])$, $\mathbb{H} = dom(M)$, and $\mathbb{H}_o = \{h \in \mathbb{H} \mid o = M(h)\}$.

By Lemma A.9, we have

$$\begin{array}{l} GE[U](M') - GE[U](M) \\ \qquad = \frac{n+1}{2} - \frac{1}{2(n+1)}(B + (|\mathbb{H}_o| + 1)^2) - \frac{n}{2} + \frac{1}{2n}(B + |\mathbb{H}_o|^2) \\ \qquad = \frac{1}{2n(n+1)}((n - |\mathbb{H}_o|)^2 + B) \geq 0 \end{array}$$

where $B = \sum_{o' \in \mathbb{O} \setminus \{o\}} |\mathbb{H}_{o'}|^2$ and $\mathbb{H}_{o'} = \{h \mid o' = M(h)\}$. □

Lemma A.11. *Let $q \geq \frac{1}{2}$. Let M be a program without low security inputs such that $GE[U](M) > q$ and $\forall M'.[\![M']\!] \subsetneq [\![M]\!] \Rightarrow GE[U](M') \leq q$. Then, it must be the case that $|[\![M]\!]| \leq \lfloor \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rfloor + 1$.*

Proof. Let n be the integer such that $n = |[\![M]\!]|$. If M returns only one output, we have $GE[U](M) = 0$. Therefore, M must have more than 1 output as $GE[U](M) > q$. By Lemma A.9, we have for any o'

$$\begin{array}{rcl} GE[U](M) & = & \frac{n}{2} - \frac{1}{2n}(B + (n-i)^2) \\ & = & i - \frac{1}{2n}(B + i^2) \end{array}$$

where $i = \sum_{o \in \mathbb{O} \setminus \{o'\}} |\mathbb{H}_o|$ and $B = \sum_{o \in \mathbb{O} \setminus \{o'\}} |\mathbb{H}_o|^2$. Because $GE[U](M) > q$, we have $i > q$. Then, we have

$$\begin{array}{lll} GE[U](M) > q & \text{iff} & i - \frac{B+i^2}{2n} > q \\ & \text{iff} & n > \frac{B+i^2}{2(i-q)} \end{array}$$

By the definition of M, we have $\forall M'.[\![M']\!] \subsetneq [\![M]\!] \Rightarrow GE[U](M') \leq q$. Let $[\![\bar{M}]\!] = [\![M]\!] \setminus \{(h', o')\}$ where $M(h') = o'$. Then, we have

$$\begin{array}{lll} GE[U](\bar{M}) \leq q & \text{iff} & i - \frac{B+i^2}{2(n-1)} \leq q \\ & \text{iff} & n \leq \frac{B+i^2}{2(i-q)} + 1 \end{array}$$

Hence, we have

$$\frac{B+i^2}{2(i-q)} < n \leq \frac{B+i^2}{2(i-q)} + 1$$

Because $B = \sum_{o \in \mathbb{O} \setminus \{o'\}} |\mathbb{H}_o|^2$ and $i = \sum_{o \in \mathbb{O} \setminus \{o'\}} |\mathbb{H}_o|$, the largest n occurs when $B = i^2$. That is, when M has exactly two outputs. Therefore, it suffices to prove the lemma for just such M's.

Now, we prove $|[\![M]\!]| \leq \lfloor \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rfloor + 1$. Recall that $i = \sum_{o \in \mathbb{O} \setminus \{o'\}} |\mathbb{H}_o|$. Let $j = n - i$. We have

$$\begin{array}{lll} GE[U](M) & = & i - \frac{1}{2n}(i^2 + i^2) \\ & = & j - \frac{j^2}{n} \\ & > & q \end{array}$$

This means that $j > q$. Recall that $[\![\bar{M}]\!] = [\![M]\!] \setminus \{(h', o')\}$ where $M(h') = o'$. Then, we have

$$\begin{array}{lll} GE[U](\bar{M}) \leq q & \text{iff} & i - \frac{i^2}{n-1} \leq q \\ & \text{iff} & n \leq \frac{i^2}{i-q} + 1 \end{array}$$

Because n is an integer, we have $n \leq \lfloor \frac{i^2}{i-q} \rfloor + 1$ and $n \leq \lfloor \frac{j^2}{j-q} \rfloor + 1$. Let $f = \frac{i^2}{i-q} + 1 = \frac{j^2}{j-q} + 1$. By elementary real analysis, it can be shown that for integers i and j such that $i > q$ and $j > q$, f attains its maximum value when $i = \lfloor q \rfloor + 1$ or $j = \lfloor q \rfloor + 1$. Therefore, it follows that $|[\![M]\!]| = n \leq \lfloor \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rfloor + 1$. □

Lemma A.12. *Let $q \geq \frac{1}{2}$. Let M be a program without low-security inputs such that $GE[U](M) > q$. Then, there exists T such that*

- $T \subseteq [\![M]\!]$
- $|T| \leq \lfloor \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rfloor + 1$
- $GE[U](M') > q$ *where* $[\![M']\!] = T$.

Proof. Let $q \geq \frac{1}{2}$. Let M be a program such that $GE[U](M) > q$. By Lemma A.10 and the fact that $GE[U](M)$ is bounded by $\frac{|[\![M]\!]|}{2}$, there exists T such that

- $T \subseteq [\![M]\!]$
- $GE[U](M') > q$ where $[\![M']\!] = T$
- $\forall T' \subseteq T.GE[U](\bar{M}) \leq q$ where $[\![\bar{M}]\!] = T'$.

By Lemma A.11, we have $|T| \leq \lfloor \frac{(\lfloor q \rfloor+1)^2}{\lfloor q \rfloor+1-q} \rfloor + 1$. Therefore, we have the conclusion. □

Theorem 3.5. *Let q be a constant, and suppose $B_{GE}[U]$ only takes programs without low security inputs. If $q \geq \frac{1}{2}$, then, $B_{GE}[U]$ is $\lfloor \frac{(\lfloor q \rfloor+1)^2}{\lfloor q \rfloor+1-q} \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor \frac{(\lfloor q \rfloor+1)^2}{\lfloor q \rfloor+1-q} \rfloor$. Otherwise, $q < \frac{1}{2}$ and $B_{GE}[U]$ is 2-safety, but it is not 1-safety.*

Proof. First, we prove that $B_{GE}[U]$ for programs without low-security inputs is $\lfloor \frac{(\lfloor q \rfloor+1)^2}{\lfloor q \rfloor+1-q} \rfloor + 1$-safety for $q \geq \frac{1}{2}$. By the definition of k-safety, for any M such that $M \notin B_{GE}[U]$, there exists T such that

1. $T \subseteq [\![M]\!]$
2. $|T| \leq \lfloor \frac{(\lfloor q \rfloor+1)^2}{\lfloor q \rfloor+1-q} \rfloor + 1$
3. $\forall M'.T \subseteq [\![M']\!] \Rightarrow M' \notin B_{GE}[U]$

We show that if $M \notin B_{GE}[U]$, then there exists T such that

- $T \subseteq [\![M]\!]$
- $|T| \leq \lfloor \frac{(\lfloor q \rfloor+1)^2}{\lfloor q \rfloor+1-q} \rfloor + 1$
- $GE[U](M') > q$ where $[\![M']\!] = T$.

Note that $GE[U](M') > q$ and Lemma A.10 imply the condition 3 above. Suppose that $M \notin B_{GE}[U]$. Then, by Lemma A.12, there exists $T \subseteq [\![M]\!]$ such that $|T| \leq \lfloor \frac{(\lfloor q \rfloor+1)^2}{\lfloor q \rfloor+1-q} \rfloor + 1$, and $GE[U](M') > q$ where $[\![M']\!] = T$.

Next, we prove $B_{GE}[U]$ for programs without low-security inputs is not k-safety for any $k \leq \lfloor \frac{(\lfloor q \rfloor+1)^2}{\lfloor q \rfloor+1-q} \rfloor$. For a contradiction, suppose $B_{GE}[U]$ is a k-safety property. Let M be a program such that

$$\begin{array}{l} M(h_1) = o, M(h_2) = o, \ldots, M(h_i) = o, \\ M(h_{i+1}) = o', M(h_{i+2}) = o', \ldots, M(h_n) = o' \end{array}$$

where $h_1, h_2, \dots h_n$, and o, o' are distinct, $n = \lfloor \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rfloor + 1$, and $i = \lfloor q \rfloor + 1$. Let $\mathbb{H}_o = \{h \mid o = M(h)\}$ and and $\mathbb{H}_{o'} = \{h \mid o' = M(h)\}$. By Lemma A.9, we have

$$\begin{array}{rcl} GE[U](M) & = & \frac{n}{2} - \frac{1}{2n}(|\mathbb{H}_o|^2 + |\mathbb{H}_{o'}|^2) \\ & = & i - \frac{i^2}{n} \\ & = & \lfloor q \rfloor + 1 - \frac{(\lfloor q \rfloor + 1)^2}{\lfloor \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rfloor + 1} \end{array}$$

Let $p = \lfloor q \rfloor + 1$. If $\frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q}$ is an integer, then we have

$$\begin{array}{rcl} GE[U](M) & = & p - \frac{p^2}{\lfloor \frac{p^2}{p-q} \rfloor + 1} \\ & = & p - \frac{p^2}{\frac{p^2 + p - q}{p - q}} \\ & = & q(\frac{(p-q)^2}{p^2 q + pq - q^2} + 1) \\ & > & q \end{array}$$

The last line follows from $p^2 q + pq - q^2 = p^2 q + q(p - q) > 0$.

Otherwise, we have $\lfloor \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rfloor + 1 = \lceil \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q} \rceil > \frac{(\lfloor q \rfloor + 1)^2}{\lfloor q \rfloor + 1 - q}$. And,

$$\begin{array}{rcl} GE[U](M) & = & p - \frac{p^2}{\lceil \frac{p^2}{p-q} \rceil} \\ & > & p - \frac{p^2}{\frac{p^2}{p-q}} \\ & = & q \end{array}$$

Hence, we have $GE[U](M) > q$. Therefore, $M \notin B_{GE}[U]$. Then, there exists T such that $|T| \leq k$, $T \subseteq [\![M]\!]$, and for any M' such that $T \subseteq [\![M']\!]$, $M' \notin B_{GE}[U]$. Let $\bar{M}$ be a program such that $[\![\bar{M}]\!] = T$. Then, by Lemma A.9 and Lemma A.10, we have

$$\begin{array}{rcl} GE[U](\bar{M}) & \leq & \frac{n-1}{2} - \frac{1}{2(n-1)}(i^2 + (n - 1 - i)^2) \\ & = & i - \frac{i^2}{\lfloor \frac{i^2}{i-q} \rfloor} \\ & \leq & i - \frac{i^2}{\frac{i^2}{i-q}} \\ & = & q \end{array}$$

It follows that $\bar{M} \in B_{GE}[U]$. Recall that $T \subseteq [\![\bar{M}]\!]$. Therefore, this leads to a contradiction.

Next, we prove that $B_{GE}[U]$ is 2-safety for any $q < \frac{1}{2}$. It suffices to show that $GE[U](M) \leq q$ iff M is non-interferent, because non-interference is a 2-safety property and not a 1-safety property [23, 3, 11]. We prove that if $GE[U](M) \leq q$ then M is non-interferent. The other direction follows from Theorem 2.14. We prove the contraposition. Suppose M is interferent. It must be the case that there exist h and h' such that $M(h) \neq M(h')$. Let $o = M(h)$, and $o' = M(h')$. Let M' be a program such that $[\![M']\!] = \{(h, o), (h', o')\}$. Note that we have $[\![M']\!] \subseteq [\![M]\!]$. By Lemma A.10, we have

$$GE[U](M') = \frac{1}{2} \leq GE[U](M)$$

It follows that $GE[U](M) > q$. □

Lemma A.13. *Let M be a program that has a low-security input, a high-security input, and a low-security output. Then, we have*

$$ME[U](M) = \log \frac{|\mathbb{O}_{\mathbb{L}}|}{|\mathbb{L}|}$$

where $\mathbb{O}_{\mathbb{L}} = \{(o, \ell) \mid \exists h. o = M(h, \ell)\}$, and $\mathbb{L}$ is sample space of the low-security input.

Proof. By the definition of ME, we have

$$ME[U](M) = \log \frac{1}{\mathcal{V}[U](H|L)} - \log \frac{1}{\mathcal{V}[U](H|O, L)}$$

where

$$\begin{array}{r} \mathcal{V}[U](H|L) = \frac{1}{|\mathbb{H}|} \\ \mathcal{V}[U](H|O, L) = \frac{|\mathbb{O}_{\mathbb{L}}|}{|\mathbb{H}||\mathbb{L}|} \end{array}$$

It follows that

$$ME[U](M) = \log \frac{|\mathbb{O}_{\mathbb{L}}|}{|\mathbb{L}|}$$

□

Theorem 3.6. *Let q be a constant. (And let $B_{ME}[U]$ take programs with low security inputs.) Then, $B_{ME}[U]$ is not a k-safety property for any $k > 0$.*

Proof. For a contradiction, suppose $B_{ME}[U]$ is a k-safety property. Let M be a program such that $M \notin B_{ME}[U]$. Then, there exists T such that $|T| \leq k$, $T \subseteq [\![M]\!]$, and for any M' such that $T \subseteq [\![M']\!]$, $M' \notin B_{ME}[U]$. Let $T = \{((h_1, \ell_1), o_1), \ldots, ((h_i, \ell_i), o_i)\}$. Let $\bar{M}$ be the following program.

$$\begin{array}{l} \bar{M}(h_1, \ell_1) = o_1, \bar{M}(h_2, \ell_2) = o_2, \ldots, \bar{M}(h_i, \ell_i) = o_i, \\ \bar{M}(h_{i+1}, \ell_{i+1}) = o_i, \bar{M}(h_{i+2}, \ell_{i+2}) = o_i, \ldots, \bar{M}(h_n, \ell_n) = o_i \end{array}$$

where $n = |\bar{\mathbb{H}}||\bar{\mathbb{L}}|$, and $\bar{\mathbb{H}}, \bar{\mathbb{L}}$ are the high security inputs and the low security inputs of $\bar{M}$. Then, by Lemma A.13, we have

$$\begin{array}{rcl} ME[U](\bar{M}) & = & \log \frac{|\mathbb{O}_{\bar{\mathbb{L}}}|}{|\bar{\mathbb{L}}|} \\ & \leq & \log \frac{i + |\bar{\mathbb{L}}|}{|\bar{\mathbb{L}}|} \end{array}$$

Therefore, for any $q > 0$, there exists $\bar{\mathbb{L}}$ such that $ME[U](\bar{M}) \leq q$ and $T \subseteq [\![\bar{M}]\!]$. Therefore, this leads to a contradiction. □

Lemma A.14. *Let M be a program that has a high-security input with sample space $\mathbb{H}$, a low-security input with sample space $\mathbb{L}$, and a low-security output. Then, we have*

$$GE[U](M) = \frac{|\mathbb{H}|}{2} - \frac{1}{2|\mathbb{H}||\mathbb{L}|} \sum_{o,\ell} |\mathbb{H}_{o,\ell}|^2$$

where $\mathbb{H}_{o,\ell} = \{h \mid o = M(h, \ell)\}$.

Proof. By the definition, we have

$$\begin{array}{rcl} GE[U](M) & = & \mathcal{G}[U](H|L) - \mathcal{G}[U](H|O,L) \\ & = & \sum_{\ell} U(\ell) \sum_{h} In(\lambda h'.U(h'|\ell), \mathbb{H}, h) U(h|\ell) \\ & & \quad - \sum_{o,\ell} U(o,\ell) \sum_{h} In(\lambda h'.U(h'|o,\ell), \mathbb{H}_{o,\ell}, h) U(h|o,\ell) \\ & = & \frac{|\mathbb{H}|+1}{2} - \sum_{o,\ell} \frac{|\mathbb{H}_{o,\ell}|}{|\mathbb{H}||\mathbb{L}|} \frac{1}{|\mathbb{H}_{o,\ell}|} \frac{1}{2} |\mathbb{H}_{o,\ell}|(|\mathbb{H}_{o,\ell}|+1) \\ & = & \frac{|\mathbb{H}|}{2} - \frac{1}{2|\mathbb{H}||\mathbb{L}|} \sum_{o,\ell} |\mathbb{H}_{o,\ell}|^2 \end{array}$$

□

Theorem 3.7. *Let q be a constant. (And let $B_{GE}[U]$ take programs with low security inputs.) Then, $B_{GE}[U]$ is not a k-safety property for any $k > 0$.*

Proof. For a contradiction, suppose $B_{GE}[U]$ is a k-safety property. Let M be a program such that $M \notin B_{GE}[U]$. Then, there exists T such that $|T| \leq k$, $T \subseteq [\![M]\!]$, and for any M' such that $T \subseteq [\![M']\!]$, $M' \notin B_{GE}[U]$. Let $T = \{((h_1, \ell_1), o_1), \dots, ((h_i, \ell_i), o_i)\}$. Let $\bar{M}$ be the following program.

$$\begin{array}{l} \bar{M}(h_1, \ell_1) = o_1, \bar{M}(h_2, \ell_2) = o_2, \dots, \bar{M}(h_i, \ell_i) = o_i, \\ \bar{M}(h_{i+1}, \ell_{i+1}) = o_i, \bar{M}(h_{i+2}, \ell_{i+2}) = o_i, \dots \bar{M}(h_{mn}, \ell_{mn}) = o_i \end{array}$$

where $n = |\bar{\mathbb{H}}|$ and $m = |\bar{\mathbb{L}}|$, and $\bar{\mathbb{H}}, \bar{\mathbb{L}}$ are the high security inputs and the low security inputs of $\bar{M}$. Then, by Lemma A.14, we have

$$\begin{array}{rcl} GE[U](\bar{M}) & = & \frac{n}{2} - \frac{1}{2mn} \sum_{o,\ell} |\mathbb{H}_{o,\ell}|^2 \\ & \leq & \frac{n}{2} - \frac{1}{2mn}(in + (m-i)n^2) \\ & = & \frac{1}{2mn}(-in + in^2) \end{array}$$

Therefore, for any $q > 0$, there exists $\bar{\mathbb{L}}$ such that $GE[U](\bar{M}) \leq q$ and $T \subseteq [\![\bar{M}]\!]$. Therefore, this leads to a contradiction. □

Theorem 3.8. *Let q be a constant and suppose $B_{SE}[U]$ only takes programs without low security inputs. Then, $B_{SE}[U]$ is not a k-safety property for any $k > 0$.*

Proof. For a contradiction, suppose $B_{SE}[U]$ is a k-safety property. Let M be a program such that $M \notin B_{SE}[U]$. Then, there exists T such that $|T| \leq k$, $T \subseteq [\![M]\!]$, and for any M' such that $T \subseteq [\![M']\!]$, $M' \notin B_{SE}[U]$. Let $T = \{(h_1, o_1), \dots, (h_i, o_i)\}$. Let $\bar{M}$ and $\bar{M}'$ be the following programs.

$$\begin{array}{l} \bar{M}(h_1) = o_1, \bar{M}(h_2) = o_2, \dots, \bar{M}(h_i) = o_i, \bar{M}(h_{i+1}) = o, \dots, \bar{M}(h_n) = o \\ \bar{M}'(h_1) = o'_1, \bar{M}'(h_2) = o'_2, \dots, \bar{M}'(h_i) = o'_i, \bar{M}'(h_{i+1}) = o', \dots, \bar{M}'(h_n) = o' \end{array}$$

where h_1, h_2, $\dots$, h_n are distinct, and o'_1, o'_2, $\dots$, o'_i, and o' are distinct. Then, we have

$$\begin{array}{rcl} SE[U](\bar{M}) & \leq & SE[U](\bar{M}') \\ & = & \frac{i}{n} \log n + \frac{n-i}{n} \log \frac{n}{n-i} \\ & = & \log \frac{n}{n-i} + \frac{i}{n} \log(n-i) \end{array}$$

Therefore, for any $q > 0$, there exists $\bar{M}$ such that $SE[U](\bar{M}) \leq q$ and $T \subseteq [\![\bar{M}]\!]$. Therefore, this leads to a contradiction. □

Theorem 3.9. *Let q be a constant, and suppose $B_{BE1}[\langle U, h\rangle]$ only takes programs without low security inputs. Then, $B_{BE1}[\langle U, h\rangle]$ is not a k-safety property for any $k > 0$.*

Proof. For a contradiction, suppose $B_{BE1}[\langle U, h\rangle]$ is a k-safety property. Let M be a program such that

$$M(h_1) = o, \dots, M(h_m) = o, M(h) = o'$$

where $m = \lfloor 2^q \rfloor$, and $h, h_1, \dots, h_m$ and o, o' are distinct. Then, we have $BE[\langle U, h\rangle](M) = \log(m+1) > \log 2^q = q$. That is, $(M, q) \notin B_{BE1}[\langle U, h\rangle]$. Then, it must be the case that there is T such that $|T| \leq k$, $T \subseteq [\![M]\!]$, and for any $\bar{M}$ such that $T \subseteq [\![\bar{M}]\!]$, $(\bar{M}, q) \notin B_{BE1}[\langle U, h\rangle]$. Let $T = \{(h'_1, o'_1), \dots, (h'_i, o'_i)\}$. Let $\bar{M}$ be the following program.

$$\begin{array}{l}\bar{M}(h'_1) = o'_1, \bar{M}(h'_2) = o'_2, \dots, \bar{M}(h'_i) = o'_i, \\ \bar{M}(h'_{i+1}) = o', \bar{M}(h'_{i+2}) = o', \dots, \bar{M}(h'_n) = o'\end{array}$$

where

- h'_1, h'_2, $\dots$, h'_n are distinct,
- $h \in \{h'_1, \dots, h'_n\}$,
- $\{o'_1, o'_2, \dots, o'_i\} = \{o, o'\}$, and
- $\bar{M}(h) = o'$.

Then, we have

$$BE[\langle U, h\rangle](\bar{M}) \leq -\log \tfrac{n-i}{n}$$

It follows that there exists n such that $BE[\langle U, h\rangle](\bar{M}) \leq q$. This leads to a contradiction. □

Lemma A.15. *Let T be a trace such that $T = \{((h_1, \ell'), o_1), \dots, ((h_i, \ell'), o_i)\}$ where $o_1, \dots, o_i$ are distinct. Let M be the program such that $[\![M]\!] = T$ and M' be a program such that $[\![M']\!] \supseteq T$. Then, we have $\max_{h,\ell} BE[\langle U, h, \ell\rangle](M') \geq \max_{h,\ell} BE[\langle U, h, \ell\rangle](M)$.*

Proof. By definition, we have

$$\begin{array}{rcl}\max_{h,\ell} BE[\langle U, h, \ell\rangle](M) & = & \log i \\ \\ \max_{h,\ell} BE[\langle U, h, \ell\rangle](M') & = & \max_{h,\ell} -\log \Sigma_{h_0 \in \{h' | M'(h',\ell) = M'(h,\ell)\}} U(h_0) \\ & \geq & \max_{h} -\log \Sigma_{h_0 \in \{h' | M'(h',\ell') = M'(h,\ell')\}} U(h_0) \\ & = & \log \frac{|\{h' | \exists o. M'(h',\ell') = o\}|}{\min_o |\{h' | M'(h',\ell') = o\}|}\end{array}$$

Therefore, it suffices to show that

$$|\{h' \mid \exists o. M'(h', \ell') = o\}| \geq i \min_o \{h' \mid M'(h', \ell') = o\}$$

Then,

$$\begin{array}{l}|\{h' \mid \exists o.M'(h',\ell') = o\}| - i\min_o\{h' \mid M'(h',\ell') = o\} \\ \quad \geq (m-i)\min_o\{h' \mid M'(h',\ell') = o\} \\ \quad \geq 0\end{array}$$

where $m = |\{o \mid \exists h.M'(h,\ell') = o\}|$. □

Theorem 3.10. *Let q be a constant. If $q \geq 1$, then $B_{BE2}[U]$ is not a k-safety property for any $k > 0$ even when $B_{BE2}[U]$ only takes programs without low security inputs. Otherwise, $q < 1$ and $B_{BE2}[U]$ is a 2-safety property, but it is not a 1-safety property.*

Proof. First, we show for the case $q \geq 1$, $B_{BE2}[U]$ is not a k-safety property for any $k > 0$. For a contradiction, suppose $B_{BE2}[U]$ is a k-safety property. Let M be the program such that

$$M = \{h_1 \mapsto o, \ldots, h_m \mapsto o, h \mapsto o'\}$$

where $m = \lfloor 2^q \rfloor$. Then, we have $BE[\langle U,h\rangle](M) = \log(m+1) > \log 2^q = q$. That is, $(M,q) \notin B_{BE2}[U]$. Then, it must be the case that there exists T such that $|T| \leq k$, $T \subseteq [\![M]\!]$, and for any M' such that $T \subseteq [\![M']\!]$, $(M',q) \notin B_{BE2}[U]$. Note that for any M' such that $[\![M']\!] \subsetneq [\![M]\!]$, $\forall h.BE[\langle U,h\rangle](M') \leq q$, and therefore, it must be the case that such T must be equal to $[\![M]\!]$.

Let $\bar{M}$ be the following program.

$$\begin{array}{l}\bar{M}(h_1) = o, \bar{M}(h_2) = o, \ldots, \bar{M}(h_m) = o, \\ \bar{M}(h) = o', \bar{M}(h_{m+1}) = o', \bar{M}(h_{m+2}) = o', \ldots, \bar{M}(h_{2m-1}) = o'\end{array}$$

where h, h_1, …, h_{2m-1} are distinct.

Then, we have $|\{h' \mid \bar{M}(h') = o\}| = |\{h' \mid \bar{M}(h') = o'\}| = m$. Therefore, for any h',

$$BE[\langle U,h'\rangle](\bar{M}) = -\log\frac{m}{2m} = 1 \leq q$$

This leads to a contradiction.

Next, we prove that $B_{BE2}[U]$ is a 2-safety property for any $q < 1$. It suffices to show that $\forall h,\ell.BE[\langle U,h,\ell\rangle](M) \leq q$ iff M is non-interferent, because non-interference is a 2-safety property and is not a 1-safety property [23, 3, 11]. We prove that if $\forall h,\ell.BE[\langle U,h,\ell\rangle](M) \leq q$ then M is non-interferent. The other direction follows from Theorem 2.14. We prove the contraposition. Suppose M is interferent. It must be the case that there exist h_0, h_1, and ℓ' such that $M(h_0,\ell') \neq M(h_1,\ell')$. Let $o = M(h_0,\ell')$, and $o' = M(h_1,\ell')$. Let M' be a program such that $[\![M']\!] = \{((h_0,\ell'),o),((h_1,\ell'),o')\}$. Note that we have $[\![M']\!] \subseteq [\![M]\!]$. By Lemma A.15, we have

$$\max_{h,\ell} BE[\langle U,h,\ell\rangle](M') = 1 \leq \max_{h,\ell} BE[\langle U,h,\ell\rangle](M)$$

It follows that $\neg(\forall h,\ell.BE[\langle U,h,\ell\rangle] \leq q)$. □

Theorem 3.11. *Let q be a constant. Then, B_{SECC} is $\lfloor 2^q \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor 2^q \rfloor$.*

Proof. Trivial from Theorem 3.3 and the fact that B_{SECC} is equivalent to B_{CC}. □

Lemma A.16. *Let μ be a distribution. Then, for any low-security input ℓ, we have $m_\ell \max_h \mu(h, \ell) \geq \sum_o \max_h \mu(h, \ell, o)$ where $m_\ell = |M[\mathbb{H}, \ell]|$*

Proof.

$$\begin{aligned} & m_\ell \max_h \mu(h, \ell) - \textstyle\sum_o \max_h \mu(h, \ell, o) \\ & \quad = \textstyle\sum_o (\max_h \mu(h, \ell) - \max_h \mu(h, \ell, o)) \\ & \quad \geq 0 \end{aligned}$$

since we have $\forall o.\, \max_h \mu(h, \ell) \geq \max_h \mu(h, \ell, o)$. □

Lemma 3.12. $\max_\mu \mathit{ME}[\mu](M) = \mathit{CC}(M)$

Proof. The statement was proved for programs without low security inputs by Braun et al. [5]. We show that the same result holds for programs with low security inputs.

Let ℓ' be a low-security input such that for any ℓ, $m_{\ell'} \geq m_\ell$ where $m_{\ell_0} = |M[\mathbb{H}, \ell_0]|$. Let μ' be a distribution such that $\forall h.\mu'(h, \ell') = \frac{1}{n}$ where n is the number of high-security inputs. We have $\mathit{CC}(M) = \mathit{ME}[\mu'](M) = \log m_{\ell'}$. Therefore, it suffices to show that for any μ, $\mathit{ME}[\mu'](M) \geq \mathit{ME}[\mu](M)$. By definition,

$$\begin{aligned} \mathit{ME}[\mu'](M) &= \log \frac{\sum_o \max_h \mu'(h, \ell', o)}{\max_h \mu'(h, \ell')} \\ \mathit{ME}[\mu](M) &= \log \frac{\sum_\ell \sum_o \max_h \mu(h, \ell, o)}{\sum_\ell \max_h \mu(h, \ell)} \end{aligned}$$

Therefore, it suffices to show that

$$\begin{aligned} & (\textstyle\sum_o \max_h \mu'(h, \ell', o))(\sum_\ell \max_h \mu(h, \ell)) \\ & \qquad -(\max_h \mu'(h, \ell'))(\textstyle\sum_\ell \sum_o \max_h \mu(h, \ell, o)) \geq 0 \end{aligned}$$

By Lemma A.16,

$$\begin{aligned} & (\textstyle\sum_o \max_h \mu'(h, \ell', o))(\sum_\ell \max_h \mu(h, \ell)) \\ & \qquad -(\max_h \mu'(h, \ell'))(\textstyle\sum_\ell \sum_o \max_h \mu(h, \ell, o)) \\ & \quad = \frac{m_{\ell'}}{n} \textstyle\sum_\ell \max_h \mu(h, \ell) - \frac{1}{n}(\sum_\ell \sum_o \max_h \mu(h, \ell, o)) \\ & \quad \geq \frac{m_{\ell'}}{n} (\textstyle\sum_\ell \max_h \mu(h, \ell) - \sum_\ell \frac{m_\ell}{m_{\ell'}} \max_h \mu(h, \ell)) \\ & \quad \geq 0 \end{aligned}$$

Therefore, we have $\mathit{ME}[\mu'](M) \geq \mathit{ME}[\mu](M)$. □

Theorem 3.13. *Let q be a constant. Then, B_{MECC} is $\lfloor 2^q \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor 2^q \rfloor$.*

Proof. Trivial by Theorem 3.3 and Lemma 3.12. □

We define the "normal form" of the guessing-entropy-based quantitative information flow expression.

Definition A.17 (Guessing entropy QIF Normal Form)**.** *Let M be a program without low-security input. The guessing-entropy based quantitative information flow $GE[\mu](M)$ can be written as the linear expression (over $\mu(h_1), \dots, \mu(h_n)$) $\sum_i a_i\mu(h_i)$ where $\mu(h_1) \geq \mu(h_2) \geq \cdots \geq \mu(h_n)$, and each a_i is a non-negative integer. We call this expression $\sum_i a_i\mu(h_i)$ the* normal form *of $GE[\mu](M)$.*

Lemma A.18. *Let M be a program without low-security input. Let $\sum_i a_i\mu(h_i)$ be the normal form of $GE[\mu](M)$. Then, for any x such that $x < |\mathbb{H}|$, we have*

$$\sum_{i\leq x} a_i \leq \frac{1}{2}(x-1)x - \frac{1}{2}(j-2)(j-1)$$

where $j = |\{h \in \{h_1, \dots, h_{x+1}\} \mid M(h) = M(h_{x+1})\}|$.

Proof. By the definition of guessing-entropy-based quantitative information flow, we have

$$a_i = i - |\{h \in \{h_1, \dots, h_i\} \mid M(h) = M(h_i)\}|$$

Therefore, we have

$$\begin{aligned}
&\textstyle\sum_{i\leq x} a_i \\
&\quad = \textstyle\sum_{i\leq x}(i - |\{h \in \{h_1, \dots, h_i\} \mid M(h) = M(h_i)\}|) \\
&\quad = \tfrac{1}{2}x(x+1) - \tfrac{1}{2}(j-1)j \\
&\qquad - \textstyle\sum_{i\in\{i'\leq x \mid M(h_{i'})\neq M(h_{x+1})\}} |\{h \in \{h_1, \dots, h_i\} \mid M(h) = M(h_i)\}| \\
&\quad \leq \tfrac{1}{2}(x-1)x - \tfrac{1}{2}(j-2)(j-1)
\end{aligned}$$

where $j = |\{h \in \{h_1, \dots, h_{x+1}\} \mid M(h) = M(h_{x+1})\}|$ □

Lemma A.19. *Let M be a program without low-security input. Let $\sum_i a_i\mu(h_i)$ be the normal form of $GE[\mu](M)$. Then, for any x such that $x < |\mathbb{H}|$, we have $\sum_{i\leq x} a_i \leq x a_{x+1}$.*

Proof. By Lemma A.18, we have

$$\sum_{i\leq x} a_i \leq \frac{1}{2}(x-1)x - \frac{1}{2}(j-2)(j-1)$$

where $j = |\{h \in \{h_1, \dots, h_{x+1}\} \mid M(h) = M(h_{x+1})\}|$, that is, $j = x + 1 - a_{x+1}$. Therefore, it suffices to show that $\frac{1}{2}(x-1)x - \frac{1}{2}(j-2)(j-1) \leq x a_{x+1}$. Then,

$$x a_{x+1} - \frac{1}{2}(x-1)x + \frac{1}{2}(j-2)(j-1) = \frac{1}{2}((x + \frac{3-2j}{2})^2 - \frac{1}{4})$$

By elementary numerical analysis, it can be shown that for integers x and j such that $x + 1 \geq j$, $\frac{1}{2}((x + \frac{(3-2j)}{2})^2 - \frac{1}{4})$ attains its minimum value 0 when $x = j - 1$. Therefore, we have $\sum_{i\leq x} a_i \leq x a_{x+1}$. □

Lemma A.20. *Let M be a program without low-security input. Let μ be a distribution. Let $h_1, \dots, h_n$ be such that $\mu(h_1) = \mu(h_2) = \cdots = \mu(h_{i-1}) > \mu(h_i) \geq \cdots \geq \mu(h_n)$. Let μ' be a distribution such that $\frac{(i-1)\mu(h_1)+\mu(h_i)}{i} = \mu'(h_1) = \cdots = \mu'(h_i)$, and $\forall x.x > i \Rightarrow \mu'(h_x) = \mu(h_x)$. Then, we have $GE[\mu](M) \leq GE[\mu'](M)$.*

Proof. Let $\sum_j a_j\mu(h_j)$ be the normal form of $GE[\mu](M)$. By the construction of μ', $\sum_j a_j\mu'(h_j)$ is the normal form of $GE[\mu'](M)$. Therefore,

$$\begin{aligned}
&GE[\mu'](M) - GE[\mu](M)\\
&\quad = \textstyle\sum_j a_j\mu'(h_j) - \sum_j a_j\mu(h_j)\\
&\quad = (a_1 + \cdots + a_i)\tfrac{(i-1)\mu(h_1)+\mu(h_i)}{i} - (a_1 + \cdots + a_{i-1})\mu(h_1) - a_i\mu(h_i)\\
&\quad = \tfrac{1}{i}((i-1)a_i - A)(\mu(h_1) - \mu(h_i))
\end{aligned}$$

where $A = a_1 + \cdots + a_{i-1}$. Since we have $(i-1)a_i - (a_1 + \cdots + a_{i-1}) \geq 0$ by Lemma A.19, and $\mu(h_1) - \mu(h_i) > 0$, we have

$$\frac{1}{i}((i-1)a_i - A)(\mu(h_1) - \mu(h_i)) \geq 0$$

Therefore, we have $GE[\mu'](M) \geq GE[\mu](M)$. □

Lemma 3.15. *We have $\max_\mu GE[\mu](M) = \max_{\ell'} GE[U \otimes \dot{\ell}'](M)$ where $U \otimes \dot{\ell}'$ denotes $\lambda h, \ell.\mathsf{if}\ \ell = \ell'\ \mathsf{then}\ U(h)\ \mathsf{else}\ 0$.*

Proof.

$$\begin{aligned}
GE[\mu](M) &= \textstyle\sum_\ell \mu(\ell)\sum_i i\mu(h_i|\ell) - \sum_\ell\sum_o \mu(\ell, o)\sum_i i\mu(h_i|\ell, o)\\
&= \textstyle\sum_\ell \mu(\ell)(\sum_i i\mu(h_i|\ell) - \sum_o\sum_i i\mu(h_i, o|\ell))\\
&= \textstyle\sum_\ell \mu(\ell) GE[\lambda h.\mu(h|\ell)](M(\ell))
\end{aligned}$$

By Lemma A.20, we have $\max_\mu GE[\mu](M(\ell)) = GE[U](M(\ell))$. Therefore, we have $\max_\mu GE[\mu](M) = (\max_{\ell'} GE[U \otimes \dot{\ell}'](M))$. □

Lemma A.21. *Let M and M' be programs such that $[\![M']\!] = [\![M]\!] \cup \{((h', \ell'), o)\}$ and $(h', \ell') \notin dom([\![M]\!])$. Then, we have $\max_\ell GE[U \otimes \ell](M) \leq \max_\ell GE[U \otimes \ell](M')$.*

Proof. By Lemma A.10, for any ℓ, we have $GE[U \otimes \ell](M) \leq GE[U \otimes \ell](M')$. Therefore, $\max_\ell GE[U \otimes \ell](M) \leq \max_\ell GE[U \otimes \ell](M')$. □

Theorem 3.14. *Let q be a constant. If $q \geq \frac{1}{2}$, then, B_{GECC} is $\lfloor \frac{(\lfloor q \rfloor+1)^2}{\lfloor q \rfloor+1-q} \rfloor + 1$-safety, but it is not k-safety for any $k \leq \lfloor \frac{(\lfloor q \rfloor+1)^2}{\lfloor q \rfloor+1-q} \rfloor$. Otherwise, $q < \frac{1}{2}$ and B_{GECC} is 2-safety, but it is not 1-safety.*

Proof. By Lemma 3.15, $(M, q) \in B_{GECC}$ iff $\max_{\ell'} GE[U \otimes \dot{\ell}'](M) \leq q$.[20] We prove for the case $q \geq \frac{1}{2}$ by a "reduction" to the result of Theorem 3.5. The case for $q < \frac{1}{2}$ follows by essentially the same argument.

[20] Therefore, for programs without low security inputs, this theorem follows from Theorem 3.5. But, we show that the theorem holds also for programs with low security inputs.

First, we show that B_{GECC} is $\lfloor\frac{(\lfloor q\rfloor+1)^2}{\lfloor q\rfloor+1-q}\rfloor + 1$-safety in this case. By the definition of k-safety, for any M such that $M \notin B_{GECC}$, there exists T such that

1. $T \subseteq [\![M]\!]$
2. $|T| \leq \lfloor\frac{(\lfloor q\rfloor+1)^2}{\lfloor q\rfloor+1-q}\rfloor + 1$
3. $\forall M'.T \subseteq [\![M']\!] \Rightarrow M' \notin B_{GECC}$

Suppose that $M \notin B_{GECC}$. By Lemma 3.15, it must be the case that there exists ℓ' such that $\max_\mu GE[\mu](M) = GE[U](M(\ell'))$. Then, by Lemma A.12, there exists $T \subseteq [\![M(\ell')]\!]$ such that $|T| \leq \lfloor\frac{(\lfloor q\rfloor+1)^2}{\lfloor q\rfloor+1-q}\rfloor + 1$, and $GE[U](M') > q$ where $[\![M']\!] = T$. Let $T' = \{((h, \ell'), o) \mid (h, o) \in T\}$. Then, we have $GE[U](M'') > q$ where $[\![M'']\!] = T'$. Finally, by Lemma A.21, we have that for any M' such that $T' \subseteq [\![M']\!]$, $M' \notin B_{GECC}$, and so B_{GECC} is $\lfloor\frac{(\lfloor q\rfloor+1)^2}{\lfloor q\rfloor+1-q}\rfloor + 1$-safety.

To see that B_{GECC} is not k-safety for any $k \leq \lfloor\frac{(\lfloor q\rfloor+1)^2}{\lfloor q\rfloor+1-q}\rfloor$, recall Theorem 3.5 that $B_{GE}[U]$ is not k-safety for such k (even) for low-security-input-free programs. Therefore, the result follows by Lemma 3.15. □

Theorem 3.16. *$(M, q) \in B_{BE1CC}[h, \ell]$ iff $M(\ell)$ is non-interferent.*

Proof. We prove that if $\forall\mu.BE[\langle\mu, h, \ell\rangle](M) \leq q$ then $M(\ell)$ is non-interferent. The other direction follows from Theorem 2.14. We prove the contraposition. Suppose $M(\ell)$ is interferent, that is, there exist h_0 and h_1 such that $M(h_0, \ell) \neq M(h_1, \ell)$. If $M(h, \ell) \neq M(h_1, \ell)$, then let μ' be a distribution such that $\mu'(h_1) = 1 - \frac{1}{\lfloor 2^q\rfloor+1}$. Otherwise, let μ' be a distribution such that $\mu'(h_0) = 1 - \frac{1}{\lfloor 2^q\rfloor+1}$. Then, we have

$$BE[\langle\mu', h, \ell\rangle](M) \geq \log(\lfloor 2^q\rfloor + 1) > q$$

□

Theorem 3.17. *$(M, q) \in B_{BE2CC}$ iff M is non-interferent.*

Proof. Straightforward from Theorem 3.16 and the fact that a program M is non-interferent iff for all ℓ, $M(\ell)$ is non-interferent. □

Notation In the proofs below, for convenience, we sometimes use large letters H, L, O, etc. to range over boolean variables as well as generic random variables. Also, we assume that variables H, H', H_1, etc. are high security boolean variables and L, L', L_i, O, O_1, O_i, etc. are low security boolean variables.

Majority SAT The following PP-hardness results (Theorems 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.8, and 4.9) are proven by a reduction from MAJSAT, which is a PP-complete problem. MAJSAT is defined as follows.

$$\text{MAJSAT} = \{\phi \mid \#SAT(\phi) > 2^{n-1}\}$$

where n is the number of variables in the boolean formula ϕ, and $\#SAT(\phi)$ is the number of satisfying assignments of ϕ.

Lemma A.22. *Let $\overrightarrow{H}$ and H' be distinct boolean random variables. Let n and m be any non-negative integers such that $n \leq 2^{|\overrightarrow{H}|}$ and $m \leq 2^{|\overrightarrow{H}|}$. Let ϕ_m (resp. ϕ_n) be a formula over $\overrightarrow{H}$ having m (resp. n) satisfying assignments. Then, $n \leq m$ iff $SE[U](M_m) \leq SE[U](M_n)$. where $M_n \equiv S(\phi_n)$, $M_m \equiv S(\phi_m)$, and S is defined in Figure 3.*[21]

Proof. First, we explain the construction $S(\psi)$ of Figure 3. Here, we use ML-like case statements (i.e., earlier cases have the precedence). It is easy to see that the case statements can be written as nested if-then-else statements. Note that $\overrightarrow{O} = \overrightarrow{\mathsf{true}}$, $O' = \mathsf{true}$, and $O'' = \mathsf{true}$ iff either $H' \wedge \psi$, or $H' \wedge H_1$ and at least one of $H_2, \ldots, H_n$ is false. For other inputs, $S(\psi)$ returns disjoint outputs. Therefore, the number of inputs h such that $S(\psi)(h) = \overrightarrow{\mathsf{true}}$ is $\#SAT(\psi) + 2^{|\overrightarrow{H}|-1} - 1$, and for the rest of the $2^{|\overrightarrow{H}|+1} - (\#SAT(\psi) + 2^{|\overrightarrow{H}|-1} - 1)$ inputs, $S(\psi)$ returns disjoint outputs different from $\overrightarrow{\mathsf{true}}$.

Therefore,

$$\begin{array}{rcl} SE[U](M_n) & = & \frac{n+2^{x-1}-1}{2^{x+1}} \log \frac{2^{x+1}}{n+2^{x-1}-1} + \frac{2^x-n+2^{x-1}+1}{2^{x+1}} \log 2^{x+1} \\ SE[U](M_m) & = & \frac{m+2^{x-1}-1}{2^{x+1}} \log \frac{2^{x+1}}{m+2^{x-1}-1} + \frac{2^x-m+2^{x-1}+1}{2^{x+1}} \log 2^{x+1} \end{array}$$

where $x = |\overrightarrow{H}|$.

- $\Rightarrow$

  Suppose $n \leq m \leq 2^{|\overrightarrow{H}|}$. Let $x = |\overrightarrow{H}|$, and let p and q be positive real numbers such that $p = \frac{n+2^{x-1}-1}{2^{x+1}}$ and $q = \frac{m+2^{x-1}-1}{2^{x+1}}$. We have $0 \leq p \leq q \leq \frac{1}{2}$. Therefore,

  $$\begin{array}{l} SE[U](M_n) - SE[U](M_m) \\ \quad = p \log \frac{1}{p} + (1-p) \log 2^{x+1} - q \log \frac{1}{q} - (1-q) \log 2^{x+1} \\ \quad \geq p \log(\frac{q}{p}) + (q-p) \log 2^{x+1} \\ \quad \geq 0 \end{array}$$

- $\Leftarrow$

  We prove the contraposition. Suppose $m < n \leq 2^{|\overrightarrow{H}|}$. Let $x = |\overrightarrow{H}|$, and let p and q be positive real numbers such that $p = \frac{n+2^{x-1}-1}{2^{x+1}}$ and $q = \frac{m+2^{x-1}-1}{2^{x+1}}$. We have $0 \leq q < p \leq \frac{1}{2}$. Therefore,

  $$\begin{array}{l} SE[U](M_m) - SE[U](M_n) \\ \quad = \log(\frac{1}{q})^q + \log p^p + ((1-q) - (1-p)) \log 2^{x+1} \\ \quad \geq \log(\frac{1}{q})^q + \log p^q + (p-q) \log 2^{x+1} \\ \quad \geq (p-q) \log 2^{x+1} \\ \quad > 0 \end{array}$$

[21] The encoding S is defined so that MAJSAT is reduced to a bounding problem with a rational upper-bound q in Theorem 4.1 below. A simpler encoding is possible if we were to do a reduction with a non-rational q.

□

Theorem 4.1. *$PP \subseteq B_{SE}[U]$*

Proof. Let ϕ be a boolean formula. Let ψ be a boolean formula such that $\#SAT(\psi) = 2^{n-1} + 1$ where n is the number of variables in ϕ. Let q be the number such that

$$\begin{array}{rcl} q & = & SE[U](S(\psi)) \\ & = & \frac{2^{n-1}+1+2^{n-1}-1}{2^{n+1}} \log \frac{2^{n+1}}{2^{n-1}+1+2^{n-1}-1} + \frac{2^{n}-(2^{n-1}+1)+2^{n-1}+1}{2^{n+1}} \log 2^{n+1} \\ & = & \frac{1}{2} + \frac{n+1}{2} \end{array}$$

where S is defined in Figure 3. Then,

$$\begin{array}{rll} (S(\phi), q) \in B_{SE}[U](S(\phi)) & \text{iff} & SE[U](S(\phi)) \leq SE[U](S(\psi)) \\ & \text{iff} & \#SAT(\phi) \geq \#SAT(\psi) \\ & \text{iff} & \phi \in \text{MAJSAT} \end{array}$$

by Lemma A.22. Therefore, we can decide if $\phi \in$ MAJSAT by deciding if $SE[U](S(\phi)) \leq q$. Note that the boolean program $S(\phi)$ and q can be constructed in time polynomial in the size of ϕ. Therefore, this is a reduction from MAJSAT to $B_{SE}[U]$. □

Lemma A.23. *Let $\overrightarrow{H}$ and H' be distinct boolean variables. Let ϕ be a boolean formula. Then, we have $ME[U](T(\phi)) = \log(\#SAT(\neg\phi)+1)$ where T is defined in Figure 4.*

Proof. It is easy to see that the number of outputs of $T(\phi)$ is equal to the number of satisfying assignment to $\neg\phi$ plus 1. Therefore, it follows from Lemma A.13 that $ME[U](T(\phi)) = \log(\#SAT(\neg\phi) + 1)$. □

Lemma A.24. *Let $\overrightarrow{H}$ and H' be distinct boolean random variables. Let m and n be any non-negative integers such that $m \leq 2^{|\overrightarrow{H}|}$ and $n \leq 2^{|\overrightarrow{H}|}$. Let ϕ_m (resp. ϕ_n) be a formula over $\overrightarrow{H}$ having m (resp. n) satisfying assignments. Then, $n \leq m$ iff $ME[U](M_m) \leq ME[U](M_n)$. where $M_n \equiv T(\phi_n)$, $M_m \equiv T(\phi_m)$, and T is defined in Figure 4.*

Proof. By Lemma A.7, Lemma A.8, and Lemma A.23, we have $ME[U](T(\phi_m)) \leq ME[U](T(\phi_n)))$ iff $\log(2^{|\overrightarrow{H}|} - m + 1) \leq \log(2^{|\overrightarrow{H}|} - n + 1)$ iff $n \leq m$. □

Theorem 4.2. *$PP \subseteq B_{ME}[U]$*

Proof. Let ϕ be a boolean formula. Let ψ be a boolean formula such that $\#SAT(\psi) = 2^{n-1} + 1$ where n is the number of variables in ϕ. Let q be the number such that

$$q = ME[U](T(\psi)) = \log(2^n - (2^{n-1} + 1) + 1) = n - 1$$

where T is defined in Figure 4. Then, we have

$$\begin{array}{rll} ME[U](T(\phi)) \leq q & \text{iff} & ME[U](T(\phi)) \leq ME[U](T(\psi)) \\ & \text{iff} & \phi \in \text{MAJSAT} \end{array}$$

by Lemma A.24. Therefore, we can decide if $\phi \in$ MAJSAT by deciding if $ME[U](T(\phi)) \leq q$. Note that $T(\phi)$ and q can be constructed in time polynomial in the size of ϕ. Therefore, this is a reduction from MAJSAT to $B_{ME}[U]$. □

Definition A.25. *Let M be a function such that $M : \mathbb{A} \to \mathbb{B}$. For any $o \in \mathbb{B}$, we write $M^{-1}(o)$ to mean*

$$M^{-1}(o) = \{i \in \mathbb{A} \mid o = M(i)\}$$

Lemma A.26. *Let $\overrightarrow{H}$ and H' be distinct boolean random variables. Let n and m be non-negative integers such that $n \leq 2^{|\overrightarrow{H}|}$ and $m \leq 2^{|\overrightarrow{H}|}$. Let ϕ_m (resp. ϕ_n) be a formula over $\overrightarrow{H}$ having m (resp. n) satisfying assignments. Then, $m \leq n$ iff $GE[U](M_n) \leq GE[U](M_m)$. where $M_n \equiv O := \phi_n \vee H'$ and $M_m \equiv O := \phi_m \vee H'$.*

Proof. By the definition,

$$\begin{array}{rcl} GE[U](M) & = & \mathcal{G}(H) - \mathcal{G}(H|O) \\ & = & \frac{1}{2}(2^{|H|+1}) + \frac{1}{2} - \sum_o \sum_{1 \leq i \leq |H|} iU(h_i, o) \\ & = & 2^{|H|} - \frac{1}{2^{|H|+2}}(|M^{-1}(\mathsf{true})|^2 + |M^{-1}(\mathsf{false})|^2) \end{array}$$

Therefore, we have

$$GE[U](M_n) \leq GE[U](M_m)$$

iff

$$|M_m^{-1}(\mathsf{true})|^2 + |M_m^{-1}(\mathsf{false})|^2 \leq |M_n^{-1}(\mathsf{true})|^2 + |M_n^{-1}(\mathsf{false})|^2$$

iff $m \leq n$. □

Theorem 4.3. *$PP \subseteq B_{GE}[U]$*

Proof. Let ϕ be a boolean formula. Let ψ be a boolean formula such that $\#SAT(\psi) = 2^{n-1} + 1$ where n is the number of variables in ϕ. Let q be the number such that

$$\begin{array}{rcl} q & = & GE(O := \psi \vee H) \\ & = & \frac{2^{n+1}}{2} - \frac{1}{2^{n+2}}(|M^{-1}(\mathsf{true})|^2 + |M^{-1}(\mathsf{false})|^2) \\ & = & 2^n - \frac{1}{2^{n+2}}((2^{n-1} + 1)^2 + (2^{n-1} - 1)^2) \end{array}$$

where H is a boolean variable that does not appear in ψ and ϕ. Then, we have

$$\begin{array}{rll} GE[U](O := \phi \vee H) \leq q & \text{iff} & GE[U](O := \phi \vee H) \leq GE[U](O := \psi \vee H) \\ & \text{iff} & GE[U](O := \phi \vee H) \leq q \\ & \text{iff} & \#SAT(\phi) \geq \#SAT(\psi) \\ & \text{iff} & \phi \in \text{MAJSAT} \end{array}$$

by Lemma A.26. Therefore, we can decide if $\phi \in$ MAJSAT by deciding if $GE[U](O := \phi \vee H) \leq q$. Note that $O := \phi \vee H$ and q can be constructed in time polynomial in the size of ϕ. Therefore, this is a reduction from MAJSAT to $B_{GE}[U]$. □

Theorem 4.4. *$PP \subseteq B_{CC}$*

Proof. Straightforward from Lemma A.7 and Theorem 4.2. □

Lemma A.27. *Let $\overrightarrow{H}$, H', and H'' be distinct boolean random variables. Let n and m be any non-negative integers such that $n \leq 2^{|\overrightarrow{H}|}$ and $m \leq 2^{|\overrightarrow{H}|}$. Let ϕ_m (resp. ϕ_n) be a formula over $\overrightarrow{H}$ having m (resp. n) satisfying assignments. Then, $n \leq m$ iff $\max_h BE[\langle U, h\rangle](M_m) \leq \max_h BE[\langle U, h\rangle](M_n)$, where $M_n \equiv V(\psi_n)$, $M_m \equiv V(\phi_m)$, and V is defined in Figure 5.* [22]

Proof. First, we explain the construction $V(\psi)$ of Figure 5. Note that $V(\psi) =$ true iff either $H' \wedge H'' \wedge \psi$, or $H' \wedge \neg H'' \wedge H_1$ and at least one of $H_2, \ldots, H_n$ is false. Therefore, there are strictly more inputs h such that $V(\psi)(h) =$ false than inputs h such that $V(\psi)(h) =$ true. Hence, $\max_h BE[\langle U, h\rangle](V(\psi)) = BE[\langle U, h'\rangle](V(\psi))$ where h' is any input such that $V(\psi)(h') =$ true.

Now, let $x = |\overrightarrow{H}|$. Then,

$$\begin{array}{rcl} \max_h BE[\langle U, h\rangle](M_n) & = & \log \frac{2^{x+2}}{n+2^{x-1}-1} \\ \max_h BE[\langle U, h\rangle](M_m) & = & \log \frac{2^{x+2}}{m+2^{x-1}-1} \end{array}$$

Therefore, $n \leq m$ iff $\max_h BE[\langle U, h\rangle](M_m) \leq \max_h BE[\langle U, h\rangle](M_n)$. □

Theorem 4.5. *$PP \subseteq B_{BE1}[\langle U, h, \ell\rangle]$*

Proof. Let ϕ be a boolean formula. Let ψ be a boolean formula such that $\#SAT(\psi) = 2^{n-1} + 1$ where n is the number of variables in ϕ. Let q be the number such that

$$q = BE[\langle U, h\rangle](V(\psi)) = \log \frac{2^{n+2}}{2^{n-1} + 1 + 2^{n-1} - 1} = 2$$

where V is defined in Figure 5, h is a high security input such that $h(H') =$ true, $h(H'') =$ false, $h(H_1) =$ true, and $h(H_2) =$ false. Note that $V(\psi)(h) = V(\phi)(h) =$ true. Then, we have

$$\begin{array}{rll} (V(\phi), q) \in B_{BE1}[\langle U, h\rangle] & \text{iff} & \max_{h'} BE[\langle U, h'\rangle](V(\phi)) \leq q \\ & \text{iff} & \max_{h'} BE[\langle U, h'\rangle](V(\phi)) \\ & & \qquad \leq \max_{h'} BE[\langle U, h'\rangle](V(\psi)) \\ & \text{iff} & \#SAT(\phi) \geq \#SAT(\psi) \\ & \text{iff} & \phi \in \text{MAJSAT} \end{array}$$

[22] As in Lemma A.22, the encoding is chosen so as to reduce MAJSAT to the bounding problem with a rational upper-bound.

by Lemma A.27, and the fact that $\max_{h'} BE[\langle U, h'\rangle](V(\phi)) = BE[\langle U, h\rangle](V(\phi))$ and $\max_{h'} BE[\langle U, h'\rangle](V(\psi)) = BE[\langle U, h\rangle](V(\psi))$. Therefore, we can decide if $\phi \in$ MAJSAT by deciding if $BE[\langle U, h\rangle](V(\phi)) \leq q$. Note that $V(\phi)$ and q can be constructed in time polynomial in the size of ϕ (in fact, q is just the constant 2). Therefore, this is a reduction from MAJSAT to $B_{BE1}[\langle U, h\rangle]$. □

Theorem 4.6. *$PP \subseteq B_{BE2}[U]$*

Proof. Let ϕ be a boolean formula. Let ψ be a boolean formula such that $\#SAT(\psi) = 2^{n-1} + 1$ where n is the number of variables in ϕ. Let q be the number such that

$$q = \max_h BE[\langle U, h\rangle](V(\psi)) = \log \frac{2^{n+2}}{2^{n-1} + 1 + 2^{n-1} - 1} = 2$$

where V is defined in Figure 5. We have

$$\begin{array}{lll} (V(\phi), q) \in B_{BE2}[U] & \text{iff} & \max_h BE[\langle U, h\rangle](V(\phi)) \leq q \\ & \text{iff} & \max_h BE[\langle U, h\rangle](V(\phi)) \leq \max_h BE[\langle U, h\rangle](V(\psi)) \\ & \text{iff} & \#SAT(\phi) \geq \#SAT(\psi) \\ & \text{iff} & \phi \in \text{MAJSAT} \end{array}$$

by Lemma A.27. Therefore, we can decide if $\phi \in$ MAJSAT by deciding if $\max_h BE[\langle U, h\rangle](V(\phi)) \leq q$. Note that $V(\phi)$ and q can be constructed in time polynomial in the size of ϕ (in fact, q is just the constant 2). Therefore, this is a reduction from MAJSAT to $B_{BE2}[U]$. □

Theorem 4.7. *$PP \subseteq B_{SECC}$*

Proof. Trivial from Theorem 4.4 and the fact that B_{SECC} is equivalent to B_{CC}. □

Theorem 4.8. *$PP \subseteq B_{MECC}$*

Proof. Straightforward from Lemma 3.12 and Theorem 4.4. □

Theorem 4.9. *$PP \subseteq B_{GECC}$*

Proof. Straightforward from Lemma 3.15 and Theorem 4.3. □

We have shown in a previous work [32] that checking non-interference for loop-free boolean programs is coNP-complete.

Lemma A.28. *Checking non-interference is coNP-complete for loop-free boolean programs.*

Theorem 4.10. *$B_{BE1CC}[h, \ell]$ is coNP-complete.*

Proof. Straightforward from Lemma A.28 and Theorem 3.16. □

Theorem 4.11. *B_{BE2CC} is coNP-complete.*

Proof. Straightforward from Lemma A.28 and Theorem 3.17. □

$$\begin{array}{rcl} M & ::= & x := \psi \mid M_0; M_1 \\ & \mid & \mathsf{if}\ \psi\ \mathsf{then}\ M_0\ \mathsf{else}\ M_1 \\ \phi, \psi & ::= & \mathsf{true} \mid x \mid \phi \wedge \psi \mid \neg\phi \end{array}$$

Figure 1: The syntax of loop-free boolean programs

$$
\begin{array}{l}
wp(x := \psi, \phi) = \phi[\psi/x] \\
wp(\mathsf{if}\ \psi\ \mathsf{then}\ M_0\ \mathsf{else}\ M_1, \phi) \\
\qquad = (\psi \Rightarrow wp(M_0, \phi)) \wedge (\neg\psi \Rightarrow wp(M_1, \phi)) \\
wp(M_0; M_1, \phi) = wp(M_0, wp(M_1, \phi))
\end{array}
$$

Figure 2: The weakest precondition for loop-free boolean programs

```
$S(\psi) \equiv$
  case $(H', \psi, \overrightarrow{H})$
    when $(\mathsf{true}, \mathsf{true}, \_)$ then $\overrightarrow{O} := \overrightarrow{\mathsf{true}}; O' := \mathsf{true}; O'' := \mathsf{true}$
    when $(\mathsf{true}, \mathsf{false}, \_)$ then $\overrightarrow{O} := \overrightarrow{H}; O' := \mathsf{true}; O'' := \mathsf{false}$
    when $(\mathsf{false}, \_, \overrightarrow{\mathsf{true}})$ then $\overrightarrow{O} := \overrightarrow{\mathsf{true}}; O' := \mathsf{false}; O'' := \mathsf{false}$
    else
      if $H_1$
        then $\overrightarrow{O} := \overrightarrow{\mathsf{true}}; O' := \mathsf{true}; O'' := \mathsf{true}$
        else $\overrightarrow{O} := \overrightarrow{H}; O' := \mathsf{false}; O'' := \mathsf{false}$
```

where H', $\overrightarrow{H} = H_1, \ldots, H_n$, and O', O'', $\vec{O}$ are distinct.

Figure 3: The Boolean Program for Lemma A.22 and Theorem 4.1.

```
$T(\phi) =$
  if $\phi \vee H'$
    then $O_f$ := true; $\overrightarrow{O} := \overrightarrow{\mathsf{false}}$
    else $O_f$ := false; $\overrightarrow{O} := \overrightarrow{H}$
```

where $\overrightarrow{H}$ and H' are distinct, and O_f and $\overrightarrow{O}$ are distinct.

Figure 4: The Boolean Program for Lemma A.23, Lemma A.24, and Theorem 4.2

```
$V(\psi) \equiv$
  case $(H', H'', \overrightarrow{H})$
    when (true, true, _) then
        if $\psi$ then $O :=$ true else $O :=$ false
    when (true, false, $\overrightarrow{\text{true}}$) then $O :=$ false
    when (true, false, _) then
        if $H_1$ then $O :=$ true else $O :=$ false
    else $O :=$ false
```

where $\overrightarrow{H} = H_1, \ldots, H_h$ is the vector of variables appearing in ψ, and $\overrightarrow{H}$, H', and H'' are distinct.

Figure 5: The Boolean Program for Lemma A.27, Theorem 4.5, and Theorem 4.6.