Probabilistic Output Analyses for Deterministic Programs –Reusing Existing Non-probabilistic Analyses

Maja Hanne Kirkeby*
Roskilde University, Denmark
kirkebym@acm.org

We consider reusing established non-probabilistic output analyses (either forward or backwards) that yield over-approximations of a program's pre-image or image relation, e.g., interval analyses. We assume a probability measure over the program input and present two techniques (one for forward and one for backward analyses) that both derive upper and lower probability bounds for the output events. We demonstrate the most involved technique, namely the forward technique, for two examples and compare their results to a cutting-edge probabilistic output analysis.

1 Introduction

Output analyses infer information about program outputs either as main purpose, e.g., interval analysis [3] and octagon analysis [12], or as bi-product, *e.g.*, sign-analysis [16]. Output analyses are also used to construct other analyses, e.g., resource analyses where resource instrumented versions of the program are analysed using output analyses [9]. *The aim of this paper* is to reuse static (non-probabilistic) output analyses to infer information about the probabilities of a program's output when knowing the input probabilities, i.e., providing an approach to mechanically obtain probabilistic output analyses for deterministic programs.

Previously, probabilistic analyses have mainly focused on analysing probabilistic programs. Sankaranarayan *et al.* [18] and Adje *et al.* [1] present analyses optimized for each their type(s) of probability measures and both provide upper and lower probability bounds of the program output. In this paper, we focus on a subset of probabilistic programs, namely the deterministic ones, but instead of presenting a specified probability analysis, we present an approach to reuse analyses to create probabilistic analyses for any type of probability measures.

More general results exist in the form of probabilistic abstract interpretation frameworks presented by Monniaux [13] and by Cousot and Monerau [6]. Both these frameworks describe how to extend non-probabilistic abstract interpretation analyses for deterministic programs to probabilistic analyses for probabilistic programs; their resulting analyses provide upper probability bounds of output events (and not lower probability bounds). Both require a manual development to handle the randomness in the programs, e.g., random number generators and the probabilistic operations. In comparison, the techniques presented in this paper handle only deterministic programs (disallowing, e.g., random number generators) but this choice allows us to consider the existing analysis tools as black-box analyses and it does not require any manual developments. Thus, the techniques are amenable to implementation. Furthermore, they each induce not only upper probability bounds of output events, but also lower probability bounds. We will compare our results with those produced by Monniaux's experimental lifting of an interval analysis [13].

*This work is supported by The Danish Council for Independent Research, Natural Sciences, grant no. DFF 4181-00442.

A. Aldini and H. Wiklicky (Eds.): Quantitative Aspects
of Programming Languages and Systems 2019 (QAPL'19).
EPTCS 312, 2020, pp. 43–57, doi:10.4204/EPTCS.312.4

Contributions and Overview. After preliminaries (Section 2), we present two novel techniques for inducing both upper and lower probability bounds of output events of deterministic programs: one using backwards analyses (Section 3.1) and one using forward analyses (Section 3.2). We demonstrate the forward technique, the most involved of the two, by two examples (Section 4.2); one using sign analysis in combination with termination analysis, and one using interval analysis. When comparing the probability bounds we produce with the ones produced by Monniaux's experimental probabilistic analysis [13], the presented approaches infer equally good and better upper probability bounds. Furthermore, when combined with non-termination analyses, they produce novel non-trivial lower bounds, *i.e.*, bounds greater than 0.

2 Preliminaries

We let X, Y, A, B, T (sometimes indexed) denote sets. A set $A^{\complement}$ is the complement of A with respect to a set (indicated by the context), e.g., X. We denote a *countable infinite series* by $X_1, X_2, \ldots$. A *partition* T over a set X is a set of nonempty and pairwise disjoint subsets of X such that $\bigcup T = X$; when T is a finite/countable/infinite set then T is a *finite/countable/infinite partition*. For two partitions T, T' over X, we say that T is *finer* than T' if every element of T is a subset of an element in T'. When T over X consists of all singletons, *i.e.*, $T = \{\{x\} \mid x \in X\}$, it is a *singleton-partition*; the finest partition T over X is the singleton-partition.

A σ*-algebra* $\mathscr{X}$ is a non-empty subset of $\wp(X)$ that is closed under countable unions and complements, *i.e.* if $A_1, A_2, \ldots \in \mathscr{X}$, then $\bigcup_{n=0}^{\infty} A_n \in \mathscr{X}$, and if $A \in \mathscr{X}$, then $A^{\complement} \in \mathscr{X}$. Note that if $\mathscr{X}$ is a σ-algebra over X, then $X \in \mathscr{X}$ and $\emptyset \in \mathscr{X}$, and, furthermore, that a σ-algebra $\mathscr{X}$ is closed under countable intersections, *i.e.* if $A_1, A_2, \ldots \in \mathscr{X}$, then $\bigcap_{n=0}^{\infty} A_n \in \mathscr{X}$. Given a collection of sets $A \subseteq \wp(X)$, the σ-algebra *generated* by A, written $\sigma(A)$, is the intersection of all σ-algebras containing A.

A *measurable space* is a pair $(X, \mathscr{X})$ whereby *the sample space* X is a set and $\mathscr{X} \subseteq \wp(X)$ is a σ-algebra. The elements of $\mathscr{X}$ are called *events*. A *measure* μ on a measurable space $(X, \mathscr{X})$ is a function $\mu \colon \mathscr{X} \to \mathbb{R}^+$ that is countably additive, *i.e.* for every countable set of pairwise disjoint sets $A_1, A_2, \ldots \in \mathscr{X}$, $\mu(\cup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} \mu(A_i)$, and $\mu(\emptyset) = 0$. Note that a measure $\mu \colon \mathscr{X} \to \mathbb{R}^+$ is monotone, *i.e.*, $A \subseteq B \Rightarrow \mu(A) \leq \mu(B)$, whenever $A, B \in \mathscr{X}$. A *measure space* $(X, \mathscr{X}, \mu)$ is a measurable space $(X, \mathscr{X})$ with a measure μ on it. Given two measure spaces $(X, \mathscr{X}, \mu_{\mathscr{X}})$ and $(Y, \mathscr{Y}, \mu_{\mathscr{Y}})$, their *product measure space* is $(X \times Y, \sigma(\mathscr{X} \times \mathscr{Y}), \mu)$ where $\mu(A \times B) = \mu_{\mathscr{X}}(A) \cdot \mu_{\mathscr{Y}}(B)$ for all $A \in \mathscr{X}$ and $B \in \mathscr{Y}$. A measure μ on a measurable space $(X, \mathscr{X})$ is *discrete* if its weight is on at most countably many elements, *i.e.* there exists a countable set $A \in \mathscr{X}$ such that $\mu(A) = \mu(X)$, and *continuous* if the weights of all countable sets are 0, *i.e.* $\mu(A) = 0$ for all countable sets $A \in \mathscr{X}$.

A measure μ on $(X, \mathscr{X})$ is a *probability measure* if $\mu(X) = 1$; in addition, $\mu(A) = 1 - \mu(A^{\complement})$. A *probability space* $(X, \mathscr{X}, \mu)$ is a measure space wherein the measure μ is a probability measure.

Let $(X, \mathscr{X})$ and $(Y, \mathscr{Y})$ be measurable spaces: A function $f \colon X \to Y$ is *measurable* if $pre_f(B) \in \mathscr{X}$ whenever $B \in \mathscr{Y}$, where the *pre-image function of* f pre_f is defined by $pre_f(B) \triangleq \{x \in X \mid f(x) \in B\}$ and the *image function* $img_f(A) \triangleq \{f(x) \in Y \mid x \in A\}$. Often we denote a measurable function by $f \colon (X, \mathscr{X}) \to (Y, \mathscr{Y})$ and we refer to elements in $\mathscr{X}$ as *input events* and elements of $\mathscr{Y}$ as *output events* A probability space $(X, \mathscr{X}, \mu)$ and a measurable function $f \colon (X, \mathscr{X}) \to (Y, \mathscr{Y})$, defines a probability measure μ_f, called an *output probability measure*, $\mu_f(A) \triangleq \mu(pre_f(A))$ of f whenever $A \in \mathscr{Y}$, *e.g.*, [2].

3 Reusing existing analysis

A program prg may have many semantics, but in this work, we consider the semantics $|\texttt{prg}|$ to be a relation between input X and output Y, *i.e.*, a set of input-output pairs $|\texttt{prg}| \subseteq X \times Y$. For a deterministic program, the input-output relation is *functional*, *i.e.*, each input is related to at most one output. Programs that terminate for all inputs $x \in X$ define *total* relations, *i.e.*, each input relates to at least one output. Depending on the analysed language, Y could contain special elements for program results that are not per se outputs, for instance, error or nontermination. Without loss of generality, we limit the focus of this paper to the class of programs with total input-output relations. Thus, assuming the program semantics to be both measurable [11, 13], *i.e.*, $|\texttt{prg}|\colon (X,\mathscr{X}) \to (Y,\mathscr{Y})$, and total.

Example 1. *Let $(X,\{\emptyset,X\})$ and $(Y,\wp(Y))$ be measurable spaces where $X=\{a,b\}$ and $Y=\{c,d\}$ and let $f\colon X \to Y$ be a function whereby $f(a)$=$f(b)$=c. The function f is total since it is defined for each input,* i.e.*, a and b, and it is measurable since the pre-images of every output event,* i.e.*, $A \in \{\emptyset,\{c\},\{d\},\{c,d\}\}$, is an input event,* i.e.*, $pre_f(A) \in \{\emptyset,\{a,b\}\}$; $pre_f(\emptyset) = pre_f(\{d\}) = \emptyset$ and $pre_f(\{c\}) = pre_f(\{c,d\}) = \{a,b\}$.*

According to our aim, we assume to know the input probability measure $\mu\colon \{\emptyset,X\} \to [0,1]$. Based on such an input probability measure μ, we recall that the probability of an output event $A \in \mathscr{Y}$ is defined as the input probability of A's pre-image $pre_{|\texttt{prg}|}(A)$, namely, $\mu(pre_{|\texttt{prg}|}(A))$.

Example 2 (Example 1 continued)**.** *Let input and output spaces and the total measurable function f be as in Example 1. In addition, let $\mu\colon \mathscr{X} \to [0,1]$ be a trivial input probability measure such that $\mu(\emptyset)=0$ and $\mu(\{a,b\})=0$. The probability of the output events $\{\emptyset,\{c\},\{d\},\{c,d\}\}$ are $\mu_f(\emptyset)=\mu_f(\{d\})=0$ and $\mu_f(\{c\})=\mu_f(\{c,d\})=1$, e.g., $\mu_f(\{d\})=\mu(pre_f(\{d\}))=\mu(\emptyset)=0$ or $\mu_f(\{c\})=\mu(pre_f(\{c\}))=\mu(\{a,b\})=1$.*

This paper is based on the idea of "reusing an existing analysis" and an analysis f is typically given in some abstract domain, *e.g.*, using abstract interpretation [3, 4], using an abstraction α from the concrete domain to the abstract domain. We will, in addition, assume a concretization function γ to avoid complications of the abstract domain. For instance, a forward interval analysis is actually a function between intervals $f\colon \mathscr{I} \to \mathscr{I}$ rather than a function between sets of reals $g\colon \wp(\mathbb{R}) \to \wp(\mathbb{R})$; however, we assume to compose f with the concretization $\gamma\colon \mathscr{I} \to \wp(\mathbb{R})$ and the abstraction $\alpha\colon \wp(\mathbb{R}) \to \mathscr{I}$, achieving the analysis, *i.e.*, $g = \gamma \circ f \circ \alpha$. The analyses may be either forward or backwards; we consider the analyses to be given as perhaps non-measurable functions $pre^\sharp_{|\texttt{prg}|}\colon \wp(Y) \to \wp(X)$ (backwards) or functions $img^\sharp_{|\texttt{prg}|}\colon \wp(X) \to \wp(Y)$ (forwards) that produce supersets[1] of the programs pre-image $pre_{|\texttt{prg}|}$ and image $img_{|\texttt{prg}|}$, respectively.

3.1 Backwards analysis

In this section, we assume a pre-image over-approximating backwards analysis of the program, *e.g.* [5], *i.e.*, we assume a function $pre^\sharp_{|\texttt{prg}|}\colon \wp(Y) \to \wp(X)$ such that $pre_{|\texttt{prg}|}(A) \subseteq pre^\sharp_{|\texttt{prg}|}(A)$. We want to use $pre^\sharp_{|\texttt{prg}|}$ to provide upper and lower probability bounds for all output events. We start by defining an order between the pre-image functions based on the relationship of their outputs.

Definition 3. *Let $pre, pre'\colon \wp(Y) \to \wp(X)$ be functions. We say that function pre'* over-approximates *pre,* i.e. *$pre \preceq pre'$, and that pre* under-approximates *pre', if $pre(A) \subseteq pre'(A)$*

[1] A set A is a superset of a set B if $A \supseteq B$.

The intention is to measure the over-approximated pre-images of each output event A using the assumed input probability measure $\mu\colon \mathscr{X} \to [0,1]$, *i.e.*, requiring $pre^\sharp(A) \in \mathscr{X}$. However, this is not always the case, as shown in the following example.

Example 4 (Examples 1,2 continued)**.** *An example of a backwards analysis $pre^\sharp_f$ that over-approximates pre_f could be defined so that $pre^\sharp_f(\{d\}) = \{b\}$ and $pre^\sharp_f(\{c\}) = \{a,b\}$. Here, $\{b\} = pre^\sharp_f(\{d\}) \supseteq pre_f(\{d\}) = \emptyset$ as required, however, $pre^\sharp_f(\{d\})$ is not measurable in the input space,* i.e.*, $\{b\} \notin \{\emptyset, \{a,b\}\}$.*

For these cases, we define a function $\uparrow$ that further over-approximates the pre-images.

Definition 5. *Let $(X, \mathscr{X})$ be a measurable space. A function $\uparrow\colon \wp(X) \to \mathscr{X}$ is an* abstraction *if $A \subseteq \uparrow A$.*

An abstraction $\uparrow\colon \wp(X) \to \mathscr{X}$ can always be defined using a mapping $f\colon X \to \mathscr{X}$, where $x \in f(x)$, *i.e.* $\uparrow B \triangleq \bigcup_{b \in B} f(b)$. The composition of a pre-image over-approximation $pre^\sharp$ and an abstraction $\uparrow$ over-approximates the pre-image and produces measurable input events.

Lemma 6. *Let $pre\colon \wp(Y) \to \wp(X)$ be a pre-image, let $pre \preceq pre^\sharp$, and let $\uparrow\colon \wp(X) \to \mathscr{X}$ be an abstraction; then, $pre \preceq \uparrow \circ pre^\sharp$ and $\uparrow(pre^\sharp(A)) \in \mathscr{X}$.*

We can now present the first result, namely the definition of an upper probability bound.

Theorem 7. *Let $f\colon (X, \mathscr{X}) \to (Y, \mathscr{Y})$ be a measurable function, let $(X, \mathscr{X}, \mu)$ be an input probability space, and let $\mu_f\colon \mathscr{Y} \to [0,1]$ be the output probability measure. Furthermore, let $pre^\sharp_f\colon \wp(Y) \to \wp(X)$ over-approximate pre_f and let $\uparrow\colon \wp(X) \to \mathscr{X}$ be an abstraction. We define the* upper probability bound *of μ_f as $\mu^\sharp_f \triangleq \mu \circ \uparrow \circ pre^\sharp_f$. Then, $\mu_f(A) \leq \mu^\sharp_f(A)$ and when $pre^\sharp_f$ and $\uparrow$ are monotonic, then $\mu^\sharp_f$ is monotonic.*

Proof. Lemma 6 yield that $\uparrow(pre^\sharp(A)) \in \mathscr{X}$ and $pre_f \preceq \uparrow \circ pre^\sharp_f$. Furthermore, by the monotonicity of μ, we obtain that for any $A \in \mathscr{Y}$ then $\mu(pre_f(A)) \leq \mu(\uparrow \circ pre^\sharp_f(A)$ and thus, $\mu_f(A) \leq \mu^\sharp_f(A)$. Finally, when $\uparrow$ and $pre^\sharp_f$ are monotonic then by composition $\mu \circ \uparrow \circ pre^\sharp_f$ is monotonic, *i.e.*, $\mu^\sharp_f$ is monotonic, since μ is monotonic by definition. □

Example 8 (Examples 1-4 continued)**.** *To obtain measurable input events, we create an abstraction $\uparrow_f\colon \wp(\{a,b\}) \to \{\emptyset, \{a,b,\}\}$ defined so that $\uparrow_f(\{a\}) = \uparrow_f(\{b\}) = \{a,b\}$ and $\uparrow_f(\emptyset) = \uparrow_f(\{a,b\}) = id$. According to Theorem 7, $\mu^\sharp_f = \mu \circ \uparrow_f \circ pre^\sharp_f$ provides upper probability bounds for the output events as follows –their exact probabilities μ_f are provided for comparison:*

$$
\begin{array}{llll}
\mu^\sharp_f(\emptyset) & = \mu(\uparrow_f(pre^\sharp_f(\emptyset))) & = \mu(\emptyset) & = 0 \qquad \big(\geq 0 = \mu_f(\emptyset) \qquad\big) \\
\mu^\sharp_f(\{c\}) & = \mu(\uparrow_f(pre^\sharp_f(\{c\}))) & = \mu(\{a,b\}) & = 1 \qquad \big(\geq 1 = \mu_f(\{c\}) \quad\big) \\
\mu^\sharp_f(\{d\}) & = \mu(\uparrow_f(pre^\sharp_f(\{d\}))) & = \mu(\{a,b\}) & = 1 \qquad \big(\geq 0 = \mu_f(\{d\}) \quad\big) \\
\mu^\sharp_f(\{c,d\}) & = \mu(\uparrow_f(pre^\sharp_f(\{d\}))) & = \mu(\{a,b\}) & = 1 \qquad \big(\geq 1 = \mu_f(\{c,d\})\big)
\end{array}
$$

Based on over-approximating pre-images we also want to derive lower probability bounds; to achieve this we will define under-approximating pre-images and we start by introducing the concept of a dual function.

Definition 9. *Let $f, \widetilde{f}$ be functions $f, \widetilde{f}\colon \wp(Y) \to \wp(X)$. $\widetilde{f}$ is* dual of f *if $\widetilde{f}(A) = f(A^{\complement})^{\complement}$.*

Because $|\texttt{prg}|$ is total, we can use the *dual* of $pre^\sharp$ to define a function $pre^\flat$ that under-approximates *pre*, as shown by the following lemma.

Lemma 10. *Let $pre_f, pre^\sharp_f \colon \wp(Y) \to \wp(X)$ be functions with pre_f as the pre-image of a total and measurable function $f \colon X \to Y$ and $pre_f \preceq pre^\sharp_f$. Then, the dual $pre^\flat_f \triangleq \widetilde{pre^\sharp_f}$ under-approximates pre_f,* i.e., *$pre^\flat_f \preceq pre_f$.*

Proof. Let $A \in \mathscr{Y}$. Since f is total, $pre_f(A) \cup pre_f(A^\complement) = X$. Thus, $pre^\flat_f(A) = \widetilde{pre^\sharp_f} = pre^\sharp_f(A^\complement)^\complement = X \setminus pre^\sharp_f(A^\complement) = \big(pre_f(A) \cup pre_f(A^\complement)\big) \setminus pre^\sharp_f(A^\complement) = pre_f(A) \setminus pre^\sharp_f(A^\complement) \subseteq pre_f(A)$. □

Lemma 11. *If $pre^\sharp_f$ is monotone, then $pre^\flat_f$ is monotone.*

Proof. Assume $A, B \subseteq X$, where $A \subseteq B$ and define $C = (B \setminus A)$; then, $pre^\flat_f(B) = pre^\flat_f(A \uplus C) = pre^\sharp_f(A \uplus C) \setminus pre^\sharp_f((A \uplus C)^\complement) = pre^\sharp_f(A \uplus C) \setminus pre^\sharp_f(A^\complement \cap C^\complement)) \supseteq pre^\sharp_f(A \uplus C) \setminus pre^\sharp_f(A^\complement) \supseteq pre^\sharp_f(A) \setminus pre^\sharp_f(A^\complement) = pre^\flat_f(A)$ □

When the over-approximated pre-images of the output events are measurable in the input measure space, their dual under-approximated pre-images are also measurable, as the following lemma states.

Lemma 12. *Let $f \colon (X, \mathscr{X}) \to (Y, \mathscr{Y})$ be a measurable function, $pre^\sharp_f \colon \wp(X) \to \wp(Y)$ be a function where $pre_f \preceq pre^\sharp_f$, and $pre^\flat_f$ be dual to $pre^\sharp_f$. Then, for all $A \in \mathscr{Y}$, $pre^\sharp_f(A) \in \mathscr{X}$ if and only if $pre_f{}^\flat(A) \in \mathscr{X}$*

Proof. Let $A \in \mathscr{Y}$. The following are consequences of σ-algebras being closed under complements, of the duality of $pre^\flat_f$ and $pre^\sharp$, and of the assumed measurability of A.

"$\Rightarrow$": $A \in \mathscr{Y} \Rightarrow A^\complement \in \mathscr{Y} \Rightarrow pre^\sharp_f(A^\complement) \in \mathscr{X} \Rightarrow pre^\sharp_f(A^\complement)^\complement \in \mathscr{X} \Rightarrow pre^\flat_f(A) \in \mathscr{X}$

"$\Leftarrow$": $A \in \mathscr{Y} \Rightarrow A^\complement \in \mathscr{Y} \Rightarrow pre^\flat_f(A^\complement) \in \mathscr{X} \Rightarrow pre^\sharp_f(A^{\complement\complement})^\complement \in \mathscr{X} \Rightarrow pre^\sharp_f(A)^\complement \in \mathscr{X} \Rightarrow pre^\sharp_f(A) \in \mathscr{X}$ □

We can now define a lower probability bound which is directly related to the upper probability bound.

Theorem 13. *Let $f \colon (X, \mathscr{X}) \to (Y, \mathscr{Y})$ be a measurable function, let $(X, \mathscr{X}, \mu)$ be an input probability space, and let $\mu_f \colon \mathscr{Y} \to [0,1]$ be the output probability measure. Furthermore, let $pre^\sharp_f \colon \wp(Y) \to \wp(X)$ over-approximate pre_f and let $\uparrow \colon \wp(X) \to \mathscr{X}$ be an abstraction. We let $pre'^\flat_f(A) \triangleq \big(\uparrow \circ pre^\sharp_f(A^\complement)\big)^\complement$ and define the* lower probability bound of μ_f *as $\mu^\flat \triangleq \mu \circ pre'^\flat_f$. Then, $\mu^\flat_f(A) \leq \mu_f(A)$ and $\mu^\flat_f(A) = 1 - \mu^\sharp_f(A^\complement)$. Furthermore, if $pre^\sharp_f$ and $\uparrow$ are monotonic, then $\mu^\flat_f$ is monotonic.*

Proof. By Lemmas 6 and 10, $pre'^\flat_f \preceq pre_f$, and by 6 and 12, $pre'^\flat_f(A) \in \mathscr{X}$. Furthermore, by the monotonicity of μ, we obtain $\mu(pre'^\flat_f(A)) \leq \mu(pre_f(A))$ and, thus, $\mu^\flat_f(A) \leq \mu_f(A) \leq \mu^\sharp_f(A)$. We obtain the second part by $\mu(A) = 1 - \mu(A^\complement)$ and the definitions of $\mu^\flat_f$ and $\mu^\sharp_f$, that is, $\mu^\flat_f(A) = \mu(pre'^\flat_f(A)) = 1 - \mu(pre'^\flat_f(A)^\complement) = 1 - \mu(\big(\uparrow \circ pre^\sharp_f(A^\complement)\big)^{\complement\complement}) = 1 - \mu(\uparrow \circ pre^\sharp_f(A^\complement)) = 1 - \mu^\sharp_f(A^\complement)$ Finally, since μ, $pre^\sharp_f$ and $\uparrow$ are monotonic, then by composition and Lemma 11, $\mu \circ pre'^\sharp_f$ and $\mu \circ pre'^\flat_f$ are monotonic. □

Example 14 (Examples 1-8 continued)**.** *Folowing Theorem 13, we define* $pre'^{\flat}_f(A) \triangleq \left(\uparrow \circ pre^{\sharp}_f(A^{\complement})\right)^{\complement}$ *such that* $\mu^{\flat} \triangleq \mu \circ pre'^{\flat}_f$ *provides the following lower probability bounds for the output events:*

$$\begin{array}{llll}
\mu^{\flat}_f(\emptyset) & = \mu(\uparrow_f (pre^{\sharp}_f(\emptyset^{\complement}))^{\complement}) & = \mu(\{a,b\}^{\complement}) = 0 & \left(\leq 0 = \mu_f(\emptyset)\right) \\
\mu^{\flat}_f(\{c\}) & = \mu(\uparrow_f (pre^{\sharp}_f(\{c\}^{\complement}))^{\complement}) & = \mu(\{a,b\}^{\complement}) = 0 & \left(\leq 1 = \mu_f(\{c\})\right) \\
\mu^{\flat}_f(\{d\}) & = \mu(\uparrow_f (pre^{\sharp}_f(\{d\}^{\complement}))^{\complement}) & = \mu(\{a,b\}^{\complement}) = 0 & \left(\leq 0 = \mu_f(\{d\})\right) \\
\mu^{\flat}_f(\{c,d\}) & = \mu(\uparrow_f (pre^{\sharp}_f(\{c,d\}^{\complement}))^{\complement}) & = \mu(\emptyset^{\complement}) = 1 & \left(\leq 1 = \mu_f(\{c,d\})\right)
\end{array}$$

To achieve the tightest probability bounds, the abstraction should return the least increased element in the σ-algebra. However, in general, such a least element does not exist.

Lemma 15. *Let* $(X, \mathscr{X})$ *be a measurable space, and let* $A \in \wp(X)$*; there does not always exist a least* $B \in \mathscr{X}$ *such that* $A \subseteq B$*.*

Proof. Proof by counterexample. A set $A \subseteq X$is *co-countable* if $A^{\complement}$ is countable. We define a σ-algebra $\mathscr{X}$ to be that generated by the collection of all countable and co-countable subsets of X. Note that since each singleton set is countable, they all exist in $\mathscr{X}$. Now, let $A \in \wp(X)$ be uncountable with an uncountable complement $A^{\complement}$. We will show (by contradiction) that there is no least $B \in \mathscr{X}$ such that $A \subseteq B$. Assume that there is a least set $B \in \mathscr{X}$ that contains A. Then, B would need to be uncountable, and according to the definition of $\mathscr{X}$, $B^{\complement}$ would be countable. Since $B^{\complement}$ is countable and $A^{\complement}$ is uncountable, $B^{\complement} \subset A^{\complement}$. This is equivalent to $A \subset B$, which causes $B \setminus A$ to contain at least one element; let that element be x. Because $\{x\}$ is a singleton set, $\{x\} \in \mathscr{X}$, and because $\mathscr{X}$ is closed under countable intersection, $B \setminus \{x\} \in \mathscr{X}$. This implies that there is another set, namely, $B \setminus \{x\}$, such that $A \subseteq B \setminus \{x\} \subset B$, and thus B is not the least set in $\mathscr{X}$ that contains A - this contradicts our assumption. □

When the σ-algebra is a complete lattice, such a least element does exist. For instance, a power set is both a complete lattice, *e.g.*, [15, p.394], and a σ-algebra [2, p.65]. If the σ-algebra is a complete lattice, then the abstraction is the identity function, *i.e.* $\uparrow = id$.

Combining analyses Two black-box analyses may be combined into a tighter analyses.

Lemma 16. *Let* $pre_f, pre^{\sharp}_f, pre'^{\sharp}_f \colon \wp(Y) \to \wp(X)$ *be three functions such that* $pre_f \preceq pre^{\sharp}_f$ *and* $pre_f \preceq pre'^{\sharp}_f$*. Then* $pre_f(A) \subseteq pre^{\sharp}_f(A) \cap pre'^{\sharp}_f(A)$*.*

3.2 Forward analysis

Again, let $|\mathtt{prg}| \colon X \to Y$ be a function, and recall that $img_{|\mathtt{prg}|}(A) \triangleq \{|\mathtt{prg}|(x) \mid x \in A\}$. In this section, we present a method for computing upper and lower probability bounds for output events provided a probability measure $\mu \colon \mathscr{X} \to [0,1]$ over the input X and a forward analysis, that is, a computable over-approximation $img^{\sharp}_{|\mathtt{prg}|} \colon \wp(X) \to \wp(Y)$ of the image-function $img_{|\mathtt{prg}|} \colon \wp(X) \to \wp(Y)$, *i.e.*, $img_{|\mathtt{prg}|} \preceq img^{\sharp}_{|\mathtt{prg}|}$. To compute the probability of the events, we only need to define a computable pre-image over-approximating function $pre^{\sharp}_{|\mathtt{prg}|}$, and then, we can apply Theorems 7 and 13 and obtain Theorem 22.

We may define the pre-image function based on the image function since the image function on the singletons defines the program semantics.

Lemma 17. *For a function* f *with* $img_f \colon \wp(X) \to \wp(Y)$ *then* $pre_f(A) = \{x \in X \mid img_f(\{x\}) \cap A \neq \emptyset\}$*.*

Proof. When f is a function, $img_f(\{x\}) = \{f(x)\}$. Thus, the above is a direct consequence of the definition of pre_f, *i.e.* $pre_f(A) \triangleq \{x \in X \mid f(x) \in A\}$. □

In our case, we do not have an image function; rather, we have an image over-approximating function $img_f^\sharp$, *i.e.*, $img_f \preceq img_f^\sharp$; we may instead use that to define a pre-image over-approximation function $pre_f^\sharp$.

Lemma 18. *Let f be a function with image function img_f and pre-image function pre_f, and let $img_f^\sharp$ be a function whereby $img_f \preceq img_f^\sharp$. If we let $pre_f^\sharp(A) \triangleq \{x \in X \mid img_f^\sharp(\{x\}) \cap A \neq \emptyset\}$, then $pre_f^\sharp(A) \supseteq pre_f(A)$.*

Proof. $pre_f(A) = \{x \in X \mid img_f(\{x\}) \cap A \neq \emptyset\} \subseteq \{x \in X \mid img_f^\sharp(\{x\}) \cap A \neq \emptyset\} = pre_f^\sharp(A)$. □

For programs whereby X is finite, any output event is computable, but when X is infinite, they are not. Instead, we propose a computable $pre_f^\sharp$ based on $img_f^\sharp$ and a finite partition of input X.

Lemma 19. *Let $f\colon (X, \mathscr{X}) \to (Y, \mathscr{Y})$ be a measurable function, let the function $img_f^\sharp\colon \wp(X) \to \wp(Y)$ over-approximate img_f, and let $\mathbf{T}$ denote the set of all partitions over X. We define a function $pre_f^\sharp\colon \mathbf{T} \to (\wp(Y) \to \wp(X))$ as $pre_f^\sharp[T](B) \triangleq \bigcup\{t \in T \mid img_f^\sharp(t) \cap B \neq \emptyset\}$ Then, $pre_f \preceq pre_f^\sharp[T]$.*

Proof. $pre_f(B) = \{x \in X \mid img_f(\{x\}) \cap B \neq \emptyset\} \subseteq \{x \in X \mid t \in T \wedge x \in t \wedge img_f(\{t\}) \cap B \neq \emptyset\} \subseteq \{x \in X \mid t \in T \wedge x \in t \wedge img_f^\sharp(\{t\}) \cap B \neq \emptyset\} \subseteq \bigcup\{t \in T \mid img_f^\sharp(\{t\}) \cap B \neq \emptyset\} = pre_f^\sharp[T](B)$ □

Proposition 20. $pre_f^\sharp[T](\bigcup_{A \in \mathbf{A}} A) = \bigcup_{A \in \mathbf{A}} pre_f^\sharp[T](A)$.

Proof. $a \in pre_f^\sharp[T](A \cup B) \Leftrightarrow \exists t \in T\colon a \in t \wedge img_f^\sharp(t) \cap (A \cup B) \neq \emptyset \Leftrightarrow \exists t \in T\colon a \in t \wedge (img_f^\sharp(t) \cap A \neq \emptyset) \vee (img_f^\sharp(t) \cap B \neq \emptyset) \Leftrightarrow \exists t \in T\colon (a \in t \wedge img_f^\sharp(t) \cap A \neq \emptyset) \vee (a \in t \wedge img_f^\sharp(t) \cap B \neq \emptyset) \Leftrightarrow a \in pre_f^\sharp[T](A) \cup pre_f^\sharp[T](B)$ □

Corollary 21. *For any partition T over X, $pre_f^\sharp[T]$ is monotone.*

Applying Theorems 7 and 13 to the computable $pre^\sharp[T]$, we obtain computable upper/lower probabilities.

Theorem 22. *Let $(X, \mathscr{X}, \mu)$ be a probability space, and let $f\colon (X, \mathscr{X}) \to (Y, \mathscr{Y})$ be a measurable function that induces the output probability measure $\mu_f\colon \mathscr{Y} \to [0,1]$,* i.e. *$\mu_f = \mu \circ pre_f$. Given a function $img_f^\sharp\colon \wp(X) \to \wp(Y)$ such that $img_f \preceq img_f^\sharp$, a finite partition T over X, and a monotone abstraction $\uparrow\colon \wp(X) \to \mathscr{X}$, we let $pre'^\sharp_f[T](A) \triangleq \uparrow\bigcup\{t \mid \exists t \in T\colon img_f^\sharp(t) \cap A \neq \emptyset\}$ and $pre'^\flat_f[T](A) \triangleq pre'^\sharp_f[T](A^\complement)^\complement$, and we define $\mu_f^\sharp \triangleq \mu \circ pre'^\sharp_f[T]$ and $\mu_f^\flat \triangleq \mu \circ pre'^\flat_f[T]$. Then, (i) $\mu_f^\flat(A) \leq \mu_f(A) \leq \mu_f^\sharp(A)$ (ii) $\mu_f^\flat(A) = 1 - \mu_f^\sharp(A^\complement)$, and (iii) $\mu_f^\flat$ and $\mu_f^\sharp$ are monotone.*

Proof. The function $pre_f^\sharp[T]$ over-approximates pre_f (by Lemma 19), and it is monotone (by Corollary 21). Thus, the above is a direct consequence of Theorems 7 and 13. □

When choosing the partition T with elements measurable in $\mathscr{X}$, then $pre_f^\sharp[T](A) \in \mathscr{X}$ for every output event A. In this case we may use identity as abstraction and can unfold the $\mu_f^\sharp$ and $\mu_f^\flat$ into a simpler form.

Theorem 23. *Let $f\colon (X,\mathscr{X}) \to (Y,\mathscr{Y})$ be a measurable function with the image function img_f and the pre-image function pre_f, let $(X,\mathscr{X},\mu)$ be a probability space, let $img_f^\sharp\colon \wp(X) \to \mathscr{Y}$ be a function that over-approximates img_f, and let T be a finite partition over X such that $T \subseteq \mathscr{X}$. Then,*

$$\mu_f^\sharp(A) = \sum_{t\in T, img^\sharp(t)\cap A\neq\emptyset} \mu(t) \qquad \text{and} \qquad \mu_f^\flat(A) = \sum_{t\in T, img^\sharp(t)\subseteq A} \mu(t).$$

Proof. Both proof parts rely on Thm. 22 where $\uparrow$ is the identity function, and T's elements are non-overlapping and mesaurable and μ is additive; furthermore, the proof of $\mu_f^\flat$ relies on the unfolding $pre'^\flat_f[T](A) = pre'^\sharp_f[T](A) \setminus pre'^\sharp_f[T](A^\complement) = \bigcup\{t \in T \mid img_f^\sharp(t)\cap A \neq \emptyset \wedge \neg(img_f^\sharp(t)\cap A^\complement \neq \emptyset)\} = \bigcup\{t \in T \mid img_f^\sharp(t)\cap A \neq \emptyset \wedge img^\sharp(t) \subseteq A\} = \bigcup\{t \in T \mid img^\sharp(t) \subseteq A\}$. □

For readers familiar with Dempster-Shafer theory, the partition is a set of focal elements and Theorem 22 generalizes to any set of (overlapping) focal elements; Theorem 23 resembles the belief and plausibility functions defined based on focal elements [10]. Furthermore, the lower probability bounds defines a belief function, which have been related to inner measures [17, 8, 19].

When $img^\sharp$ is monotone, a finer partition yields tighter probability bounds; however, for a countable infinite X, there is no finest finite partition.

Combining analyses When one or more analyses are forward analyses we can apply the above methods and combine the resulting pre-images using Lemma 16; when both are forward analyses they can be combined directly.

Lemma 24. *Let $img_f, img_f^\sharp, img'^\sharp_f\colon \wp(X) \to \wp(Y)$ be three functions such that $img_f \preceq img_f^\sharp$ and $img_f \preceq img'^\sharp_f$. Then, $img_f(A) \subseteq img_f^\sharp(A) \cap img'^\sharp_f(A)$.*

4 Examples

In the following we apply the above presented forward approach to two output analysis, namely a sign analysis and an interval analysis and compose them with termination analysis. We study three simple programs; for two of them we provide step-by-step example calculations. For the third program, we first provide the results showing an improvement compared to the essential and still cutting-edge results by Monniaux [14] and, afterwards, we provide a simple example demonstrating what causes the difference in the results.

4.1 Sign and Termination analyses

The program `sum` (Figure 1a) calculates $\sum_{i=1}^{x} i$ for an input `x`. We analyse the output properties $\wp(S)$, $S = \{\mathbb{Z}^-, \{0\}, \mathbb{Z}^+, \bot\}$ where $\bot$ represents non-terminating computations. We will derive upper and lower probability bounds for the program's output events

1. reusing a standard sign analysis, *e.g.*, [16], *i.e.*, a forward analysis $img^\sharp$,
2. reusing the online termination analysis AProVE [7], *i.e.*, a forward analysis $img'^\sharp$, and
3. by combining the image-over-approximating functions using Lemma 24 constructing a new $img''^\sharp$ that provides results that are stronger than we could from the probability bounds.

Since they are both forward analysis and the inputs are integers, we use the formulas provided in Theorem 23. We will analyse the program with respect to input partition $T = \{\mathbb{Z}^-, \{0\}, \mathbb{Z}^+\}$ with input event probabilities as follows: $\mu(\mathbb{Z}^-) = 1/3$, $\mu(\{0\}) = 1/4$, and $\mu(\mathbb{Z}^+) = 5/12$.

Sign analysis The sign analysis yields partial correctness *if the program terminates, the analysis' result contains the concrete program result* [16]. Such analyses do (obviously) not conclude anything about termination/non-termination, and we safely assume that the output of the program is that of the analysis or $\bot$, see column $img^\sharp$ in Figure 1b. Using the formulas from Theorem 23 we calculate the inferred upper and lower probabilistic bounds for each of the output properties, *e.g.*, see the following example.

Example 25. *To calculate the upper and lower probability of the output event* $\{0\}$ *the formulas from Theorem 23 require that we sum the probabilities of the input events in* T *whose image overlaps with* $\{0\}$ *and we sum the probabilities of the input events in* T *whose image is a subset of* $\{0\}$*, respectively.*

$$\mu_f^\sharp(\{0\}) = \sum_{t\in T, img^\sharp(t)\cap\{0\}\neq\emptyset}\mu(t) = \sum_{t\in\{\{0\},\mathbb{Z}^+\}}\mu(t) = \mu(\{0\})+\mu(\mathbb{Z}^+) = 1/4+5/12 = 2/3$$
$$\mu_f^\flat(\{0\}) = \sum_{t\in T, img^\sharp(t)\subseteq\{0\}}\mu(t) = \sum_{t\in\emptyset}\mu(t) = 0$$

The lower probability bound of the output event $\{0\}$ *is* 0 *and its upper probability bound is* $2/3$*; in comparison the correct probability of* $\{0\}$ *is* $1/4$.

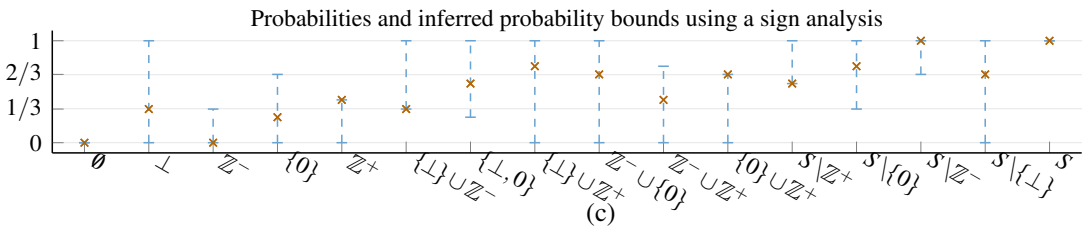
The inferred upper and lower probabilistic bounds are shown by blue dashed lines in Figure 1c; the correct probabilities are given by orange '$\times$'.

```
int (sum)(int x)
{  int y = 0;
  while (x!= 0){
     y = y + x;
     x = x - 1;}
  return y;      }
```

(a)

$t \in T$	$\mu(t)$	$img^\sharp$
$\mathbb{Z}^-$	1/3	$\{\bot\}\cup\mathbb{Z}^-$
$\{0\}$	1/4	$\{\bot,0\}$
$\mathbb{Z}^+$	5/12	$\{\bot,0\}\cup\mathbb{Z}^+$

(b)



(c)

Figure 1: The upper and lower probabilistic bounds inferred using the sign-analysis are shown by blue dashed lines and the correct probabilities are shown by the orange '$\times$'.

Termination analysis We applied the termination analyser AProVE [7] on altered versions of the program[2] for each partition element t to determine whether $\{\bot\}$ is not in t's image, *i.e.*, $img'^\sharp(t) = \{\mathbb{Z}\}$ or may be a part of t's image, *i.e.*, $img'^\sharp(t) = \{\bot,\mathbb{Z}\}$. The obtained results are displayed in column $img'^\sharp$ in Figure 2a. Again, we use the formulas from Theorem 23 to obtain upper and lower bounds, *e.g.*, see following example.

Example 26. *To calculate the upper and lower probability of the output event* $\{0\}$ *the formulas from Theorem 23 require that we sum the probabilities of the input events in* T *whose image overlaps with* $\{0\}$ *and we sum the probabilities of the input events in* T *whose image is a subset of* $\{0\}$*, respectively. Again, we use the formulas from Theorem 23 to derive upper and lower probability bounds.*

$$\mu_f^\sharp(\{0\}) = \sum_{t\in T, img'^\sharp(t)\cap\{0\}\neq\emptyset}\mu(t) = \sum_{t\in\{\mathbb{Z}^-,\{0\},\mathbb{Z}^+\}}\mu(t) = \mu(\mathbb{Z}^-)+\mu(\{0\})+\mu(\mathbb{Z}^+) = 1$$
$$\mu_f^\flat(\{0\}) = \sum_{t\in T, img'^\sharp(t)\subseteq\{0\}}\mu(t) = \sum_{t\in\emptyset}\mu(t) = 0$$

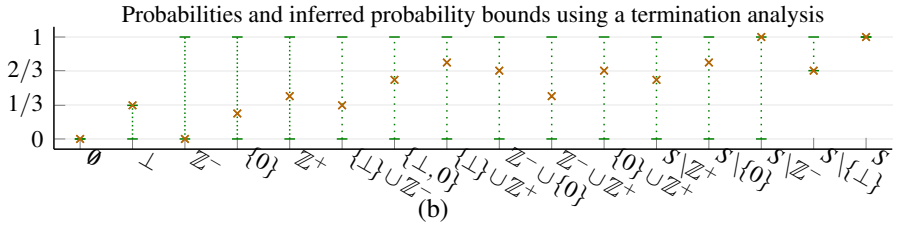
[2]For each partition element, we made sure that the alternative inputs caused the program to stop and return an integer.

The lower probability bound of the output event $\{0\}$ *is* 0 *and its upper probability bound is* 1*; in comparison the correct probability of* $\{0\}$ *is* $1/4$.

The inferred probabilistic bounds are shown in Figure 2b (green dotted) and are typically worse than those obtained by the sign-analysis.

$t \in T$	$\mu(t)$	$img'^{\sharp}$
$\mathbb{Z}^-$	$1/3$	$\{\bot\}\cup\mathbb{Z}$
$\{0\}$	$1/4$	$\mathbb{Z}$
$\mathbb{Z}^+$	$5/12$	$\mathbb{Z}$

(a)



(b)

Figure 2: The upper and lower probabilistic bounds inferred using the termination-analysis are shown by green dotted lines and the correct probabilities are shown by the orange '×'.

Combined analyses. Instead of combining the resulting probabilistic bounds from the two analyses, we have proposed to combine the analyses image-overapproximating functions using Lemma 24 and afterwards infer the probability bounds based on this combined result using, *e.g.*, the formulas of Theorem 23. Given two image-over-approximating functions $img^{\sharp}$ (see Table 1b) and $img'^{\sharp}$ (see Table 2a), we use the formula of Lemma 24 to define $img''^{\sharp}$ over the partition elements as follows.

$$\begin{array}{lllll} img''^{\sharp}(\mathbb{Z}^-) = & img^{\sharp}(\mathbb{Z}^-) \cap img'^{\sharp}(\mathbb{Z}^-) & = & (\{\bot\}\cup\mathbb{Z}^-) \cap (\{\bot\}\cup\mathbb{Z}) & = \{\bot\}\cup\mathbb{Z}^- \\ img''^{\sharp}(\{0\}) = & img^{\sharp}(\{0\}) \cap img'^{\sharp}(\{0\}) & = & \{\bot,0\} \cap \mathbb{Z} & = \{0\} \\ img''^{\sharp}(\mathbb{Z}^+) = & img^{\sharp}(\mathbb{Z}^+) \cap img'^{\sharp}(\mathbb{Z}^+) & = & (\{\bot,0\}\cup\mathbb{Z}^+) \cap \mathbb{Z} & = \{0\}\cup\mathbb{Z}^+ \end{array}$$

Again, we infer probability bounds based on the formulas of Theorem 23 using the new image-over-approximating function $img''^{\sharp}$, e.g., see the following example.

Example 27. *The procedure is similar to those in Examples 25 and 26; here, we use the function* $img''^{\sharp}$ *to derive upper and lower probability bounds of the output event* $\{0\}$.

$$\begin{array}{lll} \mu^{\sharp}_{|sum|}(\{0\}) = & \sum_{t\in T, img''^{\sharp}(t)\cap\{0\}\neq\emptyset} \mu(t) & = \sum_{t\in\{\{0\},\mathbb{Z}^+\}} \mu(t) = \mu(\{0\}) + \mu(\mathbb{Z}^+) = 1/4 + 5/12 = 2/3 \\ \mu^{\flat}_{|sum|}(\{0\}) = & \sum_{t\in T, img''^{\sharp}(t)\subseteq\{0\}} \mu(t) & = \sum_{t\in\{\{0\}\}} \mu(t) = \mu(\{0\}) = 1/4 \end{array}$$

The lower probability bound of the output event $\{0\}$ *is* $1/4$ *and its upper probability bound is* $2/3$*; in comparison the correct probability of* $\{0\}$ *is* $1/4$. *Note that both the previous probabilistic analyses produced 0 as the lower probability bound.*

The improved probability bounds are displayed in Figure 3 (solid black) together with the previous results. These combined results are more precise than if we had simply used the minimum and maximum of the bounds of the individual analyses; for instance the upper and lower probability bounds of $\{0\}$ would have been $2/3$ and 0 and with the suggested method the lower bound is improved to $1/3$. As expected from part ii of Theorem 22, *i.e.*, $\mu^{\flat}_f(A) = 1 - \mu^{\sharp}_f(A^{\complement})$, the improvement of the lower probability bound of $\{0\}$ influences the upper probability bound of its complement set, *i.e.*, $S\setminus\{0\}$, *i.e.*, $\mu_{|sum|}(S\setminus\{0\})$ is reduced from 1 to $2/3$.

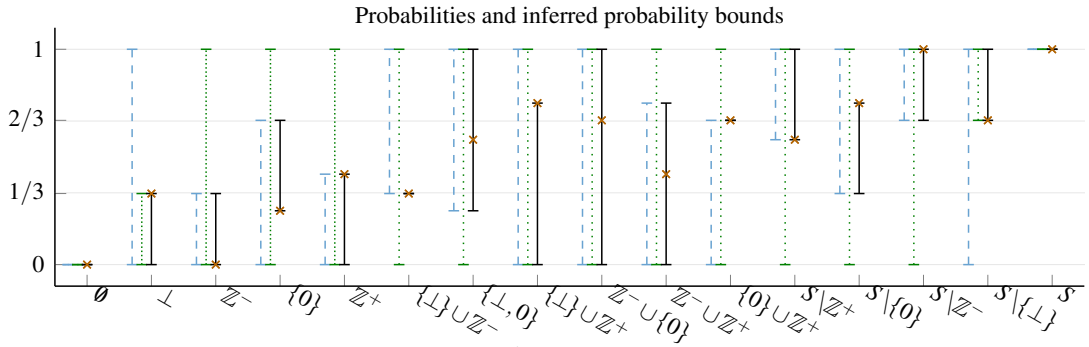


Figure 3

Figure 4: Probability bounds inferred by $img^\sharp$ (using the sign-analysis) are shown as blue dashed lines, those inferred by $img'^\sharp$ (using the termination-analysis) are shown as green dotted lines, and those inferred by their combined image-overapproximating function $img''^\sharp$ are shown as solid black lines. For comparison, the correct probability distribution is indicated by orange '×'.

4.2 Interval analyses

We study the following simple programs `f` and `g` and compare our results with those by Monniaux; the first program `f` has no branching points, but the second program `g` do.

```
double f(double x1, x2, x3, x4){
  double x; x = 0.0;
  x = x+ x1*2.0-1.0;
  x = x+ x2*2.0-1.0;
  x = x+ x3*2.0-1.0;
  x = x+ x4*2.0-1.0;
  return x; }
```

```
double g(double x1, x2, x3, x4, x5){
  double x; x = 0.0;   (*)
  if (x5 >= 0.5) x = x+ x1*2.0-1.0; (**)
  x = x+ x2*2.0-1.0;
  x = x+ x3*2.0-1.0;
  x = x+ x4*2.0-1.0;   (***)
  return x;}
```

These programs are deterministic versions of two programs similar to those analysed by Monniaux's experimental analysis[3] [13]; however, instead of having random generators in the program, here, the random generated inputs are given as program input. We analyse the programs using a black-box interval analysis corresponding to that lifted by Monniaux [13] to handle probabilistic programs; furthermore, to ensure a fair comparison we use an input-partition that corresponds to the abstraction used by Monniaux.

For both programs we infer probability bounds for the output events $\{[-4,-3],[-3,-2],\ldots,[3,4]\}$, given an input probability measure where the input arguments are independent and each is uniformly distributed between 0 and 1. Since the interval analysis yields partially correct results, stating nothing about non-termination, we safely assume to include non-termination as part of its output.

Program `f`. The program `f` return the sum four input variables. Because the interval analysis $img^\sharp_{|\mathtt{f}|}$ is a forward analysis, we use the formulas of Theorem 23 and let the input partition T be the cartesian product $T = I_{1/10} \times I_{1/10} \times I_{1/10} \times I_{1/10}$ where $I_{1/10} = \{[0, \frac{1}{10}], \ldots, [\frac{9}{10}, 1]\}$, *e.g.*, see following example.

Example 28. *The interval analysis provides an image-overapproximating function* $img^\sharp_{|\mathtt{f}|}$ *where the*

[3] Personal communication with D. Monniaux.

following are three examples.

$$\begin{aligned} img^{\sharp}_{|\mathtt{f}|}([0,\tfrac{1}{10}],[0,\tfrac{1}{10}],[0,\tfrac{1}{10}],[0,\tfrac{1}{10}]\,) &= \{[-4,-\tfrac{32}{10}],\bot\} \\ img^{\sharp}_{|\mathtt{f}|}([0,\tfrac{1}{10}],[0,\tfrac{1}{10}],[0,\tfrac{1}{10}],[\tfrac{1}{10},\tfrac{2}{10}]) &= \{[-\tfrac{38}{10},-\tfrac{30}{10}],\bot\} \\ img^{\sharp}_{|\mathtt{f}|}([0,\tfrac{1}{10}],[0,\tfrac{1}{10}],[0,\tfrac{1}{10}],[\tfrac{2}{10},\tfrac{3}{10}]) &= \{[-\tfrac{36}{10},-\tfrac{28}{10}],\bot\} \end{aligned}$$

There are $10'000$ partition elements, and due to the uniform probability measure each of the partition elements has probability $1/10'000$. We use the formulas from Theorem 23 to derive upper and lower probability bounds of the output events, *e.g.*, see following example. The inferred probability bounds are shown as black solid lines in Figure 5a and the similar results obtained using Monniaux's experimental analysis [13] are depicted by the blue dotted lines.

Example 29. *We use the formulas from Theorem 23 to calculate the upper and lower probability bounds of the output event* $[-4,-3]$*. There are 70 input partition elements whose over-approximated images,* i.e.*,* $img^{\sharp}_{|\mathtt{f}|}$*, overlap with the output event* $[-4,-3]$*; we will refrain from specifying them. Since all the over-approximated images contain the element* $\bot$ *then there are 0 input partition elements whose over-approximated images are subsets of* $[-4,-3]$*. The input probability of each element is* $1/10'000$ *which simplifies the calculations.*

$$\begin{aligned} \mu^{\sharp}_{|\mathtt{f}|}([-4,-3]) = &\quad \textstyle\sum_{t\in T, img^{\sharp}_{|\mathtt{f}|}(t)\cap[-4,-3]\neq\emptyset}\mu(t) &&= 70\cdot\tfrac{1}{10000} = \tfrac{7}{1000} \approx 0.007 \\ \mu^{\flat}_{|\mathtt{f}|}([-4,-3]) = &\quad \textstyle\sum_{t\in T, img'^{\sharp}(t)\subseteq[-4,-3]}\mu(t) &&= \textstyle\sum_{t\in\emptyset}\mu(t) = 0 \end{aligned}$$

In comparison the correct probability of the output event $[-4,-3]$ *is* $1/384 \approx 0.00260417$ *(verified in Mathematica).*

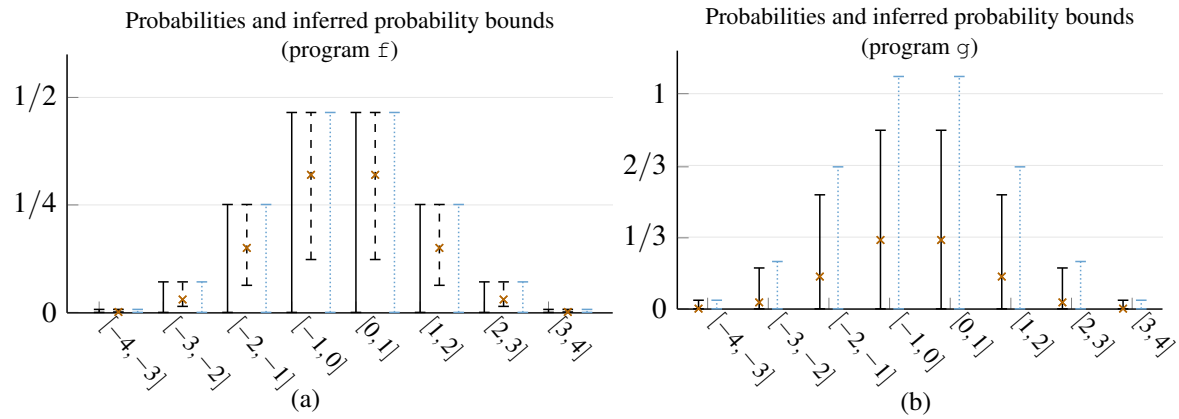


Figure 5: The black solid lines are the inferred probability bounds obtained using an interval analysis. In (a), the dashed black lines show the improved inferred probability bounds based on combining the interval-analysis results with termination-analysis results. The blue dotted lines indicates the upper probability bounds for the output events derived using Monniaux's example analysis [13].

Combined analyses. As we saw in the sign-analysis example (Section 4.1), combining the analysis with the results of a termination analysis may improve the results. This is also the case in this example as shown by the black dotted lines in Figure 5a, e.g., see following example.

Example 30. *As for* $img^{\sharp}_{|\mathtt{f}|}$ *the same 70 partition elements overlaps with the output event* $[-4,-3]$*. However, there are 5 input partition elements for which* $img'^{\sharp}_{|\mathtt{f}|}$ *are subsets of* $[-4,-3]$*, namely* $T' =$

$\{(i,i,i,i),(j,i,i,i),(i,j,i,i),(i,i,j,i),(i,i,i,j)\}$ *where* $i=[0,\frac{1}{10}]$ *and* $j=[\frac{1}{10},\frac{2}{10}]\}$*, as the first two calculations in Example 28 indicate. The input probability of each element is* $1/10'000$ *which simplifies the calculations.*

$$\mu^{\sharp}_{|\mathcal{f}|}([-4,-3]) = \sum_{t\in T, img^{\sharp}_{|\mathcal{f}|}(t)\cap[-4,-3]\neq\emptyset}\mu(t) = 70\cdot\tfrac{1}{10000} = \tfrac{7}{1000}\approx 0.007$$
$$\mu^{\flat}_{|\mathcal{f}|}([-4,-3]) = \sum_{t\in T, img''^{\sharp}(t)\subseteq[-4,-3]}\mu(t) = \sum_{t\in T'}\tfrac{1}{10000} = 5\cdot\tfrac{1}{10000} = \tfrac{5}{10000}\approx 0.0005$$

In comparison the correct probability of the output event $[-4,-3]$ *is* $1/384\approx 0.00260417$ *(verified in Mathematica).*

The non-trivial lower probability bounds is a novel development, and Monniaux's lifting framework is created with the purpose of deriving only upper probability bounds.

Program $\mathcal{g}$. The program $\mathcal{g}$ (53) returns the sum three or four input variables, depending on the value of the fifth input variable. Due to the branching point in $\mathcal{g}$, *i.e.*, the if-expression, we see a difference in the upper probability bounds derived using the presented approach and using Monniaux's experimental analysis.

Again, the interval analysis $img^{\sharp}_{|\mathcal{g}|}$ is a forward analysis and we have used the formulas of Theorem 23. The difference in the bounds stems from undecidability occurring in the branching point and to expose this difference we have chosen a partition causing such undecidability. The input partition is the cartesian product $I_{1/3}\times I_{1/3}\times I_{1/3}\times I_{1/3}\times I_{1/3}$ where $I_{1/3}=\{[0,\frac{1}{3}],[\frac{1}{3},\frac{2}{3}],[\frac{2}{3},1]\}$. There are 243 partition elements, and due to the uniform probability measure each of the partition elements has probability $1/243$. When comparing the results of the methods presented in this paper, shown by the solid black lines in Figure 5b, and the results obtained using Monniaux's experimental analysis [13], shown by the blue dotted lines in Figure 5b, the methods presented here produce better results.

The difference. In the following we demonstrate via an example what causes the difference between our results and those derived using Monniaux's experimental analysis [13]. Instead of studying the above partition with 243 elements, we study a smaller input partition with 6 elements, namely $T=\{[0,1]\}\times\{[0,1]\}\times\{[0,1]\}\times\{[0,\frac{1}{2}],[\frac{1}{2},1]\}\times\{[0,\frac{1}{3}],[\frac{1}{3},\frac{2}{3}],[\frac{2}{3},1]\}$ where each partition element has input probability $1/6$ and we will derive the upper probability bound for the output event $[2\frac{1}{2},3\frac{1}{2}]$.

Example 31. *The interval analysis provides an image-overapproximating function* $img^{\sharp}_{|\mathcal{g}|}$ *over partition* T*:*

$$\begin{aligned}
img^{\sharp}_{|\mathcal{g}|}([0,1],[0,1],[0,1],[0,\tfrac{1}{2}],[0,\tfrac{1}{3}]) &= \{[-3,2],\bot\}\\
img^{\sharp}_{|\mathcal{g}|}([0,1],[0,1],[0,1],[0,\tfrac{1}{2}],[\tfrac{1}{3},\tfrac{2}{3}]) &= \{[-4,3],\bot\}\\
img^{\sharp}_{|\mathcal{g}|}([0,1],[0,1],[0,1],[0,\tfrac{1}{2}],[\tfrac{2}{3},1]) &= \{[-4,3],\bot\}\\
img^{\sharp}_{|\mathcal{g}|}([0,1],[0,1],[0,1],[\tfrac{1}{2},1],[0,\tfrac{1}{3}]) &= \{[-2,3],\bot\}\\
img^{\sharp}_{|\mathcal{g}|}([0,1],[0,1],[0,1],[\tfrac{1}{2},1],[\tfrac{1}{3},\tfrac{2}{3}]) &= \{[-3,4],\bot\}\\
img^{\sharp}_{|\mathcal{g}|}([0,1],[0,1],[0,1],[\tfrac{1}{2},1],[\tfrac{1}{2},1]) &= \{[-3,4],\bot\}
\end{aligned}$$

Similar to the previous examples, we use Theorem 23 to infer an upper probability bound for output event $[2\frac{1}{2},3\frac{1}{2}]$.

$$\mu^{\sharp}_{|\mathcal{f}|}([2\frac{1}{2},3\frac{1}{2}]) = \sum_{t\in T, img^{\sharp}_{|\mathcal{g}|}(t)\cap[2\frac{1}{2},3\frac{1}{2}]\neq\emptyset}\mu(t) = \sum_{t\in T'}\mu(t) = 5/6.$$

where $T'=T\setminus\{([0,1],[0,1],[0,1],[0,\frac{1}{2}],[0,\frac{1}{3}])\}$.

Monniaux's experimental abstracts the input measure to a set containing pairs of interval environments (similar to the interval analysis) and a weight, *i.e.*, the input probability of that interval environment, *e.g.*, $\langle E, w\rangle$,. Afterwards, it propagates the pairs through the program according to the specifications of the interval analysis; at the branching points the lifted interval analysis duplicates the pair, *e.g.*, $\langle E, w\rangle$ and $\langle E, w\rangle$, and adjust the environments according to the condition and its negation to avoid infeasible environments, *e.g.*, $\langle E_1, w\rangle$ and $\langle E_2, w\rangle$, *e.g.*, see following example. When the set is propagated all the way through the program, the probability of an output event is the sum of the weights of those environments yielding output which overlaps with the output event.

In the following we propagate the pairs through the program and calculates the upper probability bound; we provide the set of pairs at the program points at $\star$, $\star\star$, and $\star\star\star$.

Example 32. *At program point $\star$ in $\mathcal{G}$ (p. 53) the environments are still recognizable from the input partition elements with the addition of variable* x*. The interval of* x5 *in the second and fifth pair may lead to the condition being true or false (undecidable); this causes a split of each the those pairs into two whose* x5 *intervals are updated according to the branch condition. Afterwards,* x *is updated as expected; according to the branch (or skip) before reaching program point $\star\star$), and again upon reaching program point $\star\star\star$. In the following, we have only included the essential parts of the environments essential,* e.g.*, at every program point we have left out* $\mathtt{x1} \mapsto [0,1], \mathtt{x2} \mapsto [0,1], \mathtt{x3} \mapsto [0,1]$.

Program point $\star$:	*Program point $\star\star$:*	*Program point $\star\star\star$:*
$\{\langle \ldots, \mathtt{x4} \mapsto [0,\frac{1}{2}], \mathtt{x5} \mapsto [0,\frac{1}{3}], \mathtt{x} \mapsto [0,0]; \frac{1}{6}\rangle,$	$\{\langle \ldots, \mathtt{x5} \mapsto [0,\frac{1}{3}], \mathtt{x} \mapsto [0,0]; \frac{1}{6}\rangle,$	$\{\langle \ldots, \mathtt{x} \mapsto [-3,2]; \frac{1}{6}\rangle,$
$\langle \ldots, \mathtt{x4} \mapsto [0,\frac{1}{2}], \mathtt{x5} \mapsto [\frac{1}{3},\frac{2}{3}], \mathtt{x} \mapsto [0,0]; \frac{1}{6}\rangle,$	$\langle \ldots, \mathtt{x5} \mapsto [\frac{1}{3},\frac{1}{2}], \mathtt{x} \mapsto [0,0]; \frac{1}{6}\rangle,$	$\langle \ldots, \mathtt{x} \mapsto [-3,2]; \frac{1}{6}\rangle,$
	$\langle \ldots, \mathtt{x5} \mapsto [\frac{1}{2},\frac{2}{3}], \mathtt{x} \mapsto [-1,1]; \frac{1}{6}\rangle,$	$\langle \ldots, \mathtt{x} \mapsto [-4,3]; \frac{1}{6}\rangle,$
$\langle \ldots, \mathtt{x4} \mapsto [0,\frac{1}{2}], \mathtt{x5} \mapsto [\frac{2}{3},1], \mathtt{x} \mapsto [0,0]; \frac{1}{6}\rangle,$	$\langle \ldots, \mathtt{x5} \mapsto [\frac{2}{3},1], \mathtt{x} \mapsto [-1,1]; \frac{1}{6}\rangle,$	$\langle \ldots, \mathtt{x} \mapsto [-4,3]; \frac{1}{6}\rangle,$
$\langle \ldots, \mathtt{x4} \mapsto [\frac{1}{2},1], \mathtt{x5} \mapsto [0,\frac{1}{3}], \mathtt{x} \mapsto [0,0]; \frac{1}{6}\rangle,$	$\langle \ldots, \mathtt{x5} \mapsto [0,\frac{1}{3}], \mathtt{x} \mapsto [0,0]; \frac{1}{6}\rangle,$	$\langle \ldots, \mathtt{x} \mapsto [-2,3]; \frac{1}{6}\rangle,$
$\langle \ldots, \mathtt{x4} \mapsto [\frac{1}{2},1], \mathtt{x5} \mapsto [\frac{1}{3},\frac{2}{3}], \mathtt{x} \mapsto [0,0]; \frac{1}{6}\rangle,$	$\langle \ldots, \mathtt{x5} \mapsto [\frac{1}{3},\frac{1}{2}], \mathtt{x} \mapsto [0,0]; \frac{1}{6}\rangle,$	$\langle \ldots, \mathtt{x} \mapsto [-2,3]; \frac{1}{6}\rangle,$
	$\langle \ldots, \mathtt{x5} \mapsto [\frac{1}{2},\frac{2}{3}], \mathtt{x} \mapsto [-1,1]; \frac{1}{6}\rangle,$	$\langle \ldots, \mathtt{x} \mapsto [-3,4]; \frac{1}{6}\rangle,$
$\langle \ldots, \mathtt{x4} \mapsto [\frac{1}{2},1], \mathtt{x5} \mapsto [\frac{2}{3},1], \mathtt{x} \mapsto [0,0]; \frac{1}{6}\rangle\}$	$\langle \ldots, \mathtt{x5} \mapsto [\frac{2}{3},1], \mathtt{x} \mapsto [-1,1]; \frac{1}{6}\rangle\}$	$\langle \ldots, \mathtt{x} \mapsto [-3,4]; \frac{1}{6}\rangle\}$

The upper probability bound of output event $[2\frac{1}{2}, 3\frac{1}{2}]$ is the sum of the weights of the pairs for which the x*'s value overlaps with $[2\frac{1}{2}, 3\frac{1}{2}]$ at program point $\star\star\star$. The last six pairs are the only ones where* x *interval overlap with $[2\frac{1}{2}, 3\frac{1}{2}]$, thus, Monniaux's experimental analysis [13] derives $6 \cdot \frac{1}{6} = 1$ as the upper probability bound for $[2\frac{1}{2}, 3\frac{1}{2}]$. Thus, the bound $(5/6)$ derived by the presented technique is tighter.*

5 Conclusion

We have presented two simple techniques for reusing existing (non-probabilistic) analyses to derive upper and lower probability bounds of output events; introducing abstraction when the pre-image is non-measurable. We demonstrated forward technique and the initial results are powerful compared to more complex analyses.

References

[1] Assale Adje, Olivier Bouissou, Jean Goubault-Larrecq, Eric Goubault & Sylvie Putot (2014): *Static Analysis of Programs with Imprecise Probabilistic Inputs*, pp. 22–47. Springer Berlin Heidelberg, Berlin, Heidelberg, doi:10.1007/978-3-642-54108-7_2.

[2] Troy Butler, Don Estep & Nishant Panda (2018): *A Ramble Through the Foundations of Probability [unpublished]*.

[3] Patrick Cousot & Radhia Cousot (1977): *Abstract Interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints*, doi:10.1145/512950.512973.

[4] Patrick Cousot & Radhia Cousot (1979): *Systematic design of program analysis frameworks*, doi:10.1145/567752.567778.

[5] Patrick Cousot, Radhia Cousot, Manuel Fähndrich & Francesco Logozzo (2013): *Automatic inference of necessary preconditions*. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7737 LNCS, pp. 128–148, doi:10.1007/978-3-642-35873-9_10.

[6] Patrick Cousot & Michael Monerau (2012): *Probabilistic Abstract Interpretation*, pp. 169–193. Springer Berlin Heidelberg, Berlin, Heidelberg, doi:10.1007/978-3-642-28869-2_9.

[7] Jürgen Giesl, Cornelius Aschermann, Marc Brockschmidt, Fabian Emmes, Florian Frohn, Carsten Fuhs, Jera Hensel, Carsten Otto, Martin Plücker, Peter Schneider-Kamp, Thomas Ströder, Stephanie Swiderski & René Thiemann (2017): *Analyzing Program Termination and Complexity Automatically with AProVE*. *Journal of Automated Reasoning* 58(1), pp. 3–31, doi:10.1007/s10817-016-9388-y.

[8] Joseph Y. Halpern & Ronald Fagin (1989): *Modelling knowledge and action in distributed systems*. *Distributed Computing* 3(4), pp. 159–177, doi:10.1007/BF01784885.

[9] Maja H. Kirkeby & Mads Rosendahl (2016): *Probabilistic Resource Analysis by Program Transformation*. In Marko van Eekelen & Ugo Dal Lago, editors: *Foundational and Practical Aspects of Resource Analysis: 4th International Workshop, FOPARA 2015, London, UK, April 11, 2015. Revised Selected Papers*, Springer International Publishing, Cham, pp. 60–80, doi:10.1007/978-3-319-46559-3_4.

[10] George J Klir (2007): *Uncertainty and Information: Foundations of Generalized Information Theory (Klir, G.J.; 2006)*. 18, doi:10.1109/TNN.2007.906888.

[11] Dexter Kozen (1985): *A probabilistic PDL*. *Journal of Computer and System Sciences* 30(2), pp. 162–178, doi:10.1016/0022-0000(85)90012-1.

[12] Antoine Miné (2006): *The octagon abstract domain*. *Higher-Order and Symbolic Computation* 19(1), pp. 31–100, doi:10.1007/s10990-006-8609-1.

[13] David Monniaux (2000): *Abstract Interpretation of Probabilistic Semantics*. In Jens Palsberg, editor: *Static Analysis, 7th International Symposium, SAS 2000, Santa Barbara, CA, USA, June 29 - July 1, 2000, Proceedings*, *Lecture Notes in Computer Science* 1824, Springer, pp. 322–339, doi:10.1007/978-3-540-45099-3_17.

[14] David Monniaux (2001): *Backwards Abstract Interpretation of Probabilistic Programs*. In David Sands, editor: *Programming Languages and Systems, 10th European Symposium on Programming, ESOP 2001 Genova, Italy, April 2-6, 2001, Proceedings*, *Lecture Notes in Computer Science* 2028, Springer, pp. 367–382, doi:10.1007/3-540-45309-1_24.

[15] Flemming Nielson, Hanne Riis Nielson & Chris Hankin (1999): *Principles of Program Analysis*. Springer Berlin Heidelberg, Berlin, Heidelberg, doi:10.1007/978-3-662-03811-6.

[16] Hanne Riis Nielson & Flemming Nielson (2007): *Semantics with Applications: an Appetizer*. doi:10.1159/000321363.

[17] Enrique H. Ruspini: *Epistemic Logics, Probability, and the Calculus of Evidence*. In: *Classic Works of the Dempster-Shafer Theory of Belief Functions*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 435–448, doi:10.1007/978-3-540-44792-4_17.

[18] Sriram Sankaranarayanan, Aleksandar Chakarov & Sumit Gulwani (2013): *Static analysis for probabilistic programs*. *ACM SIGPLAN Notices* 48(6), p. 447, doi:10.1145/2499370.2462179.

[19] Glenn Shafer (1990): *Perspectives on the theory and practice of belief functions*. *International Journal of Approximate Reasoning* 4(5-6), pp. 323–362, doi:10.1016/0888-613X(90)90012-Q.