A Fixed Point Iteration Technique for Proving Correctness of Slicing for Probabilistic Programs

Torben Amtoft[1] and Anindya Banerjee[2]

[1] Kansas State University, Manhattan KS 66506, USA tamtoft@ksu.edu
[2] IMDEA Software Institute, Pozuelo de Alarcon, Madrid, Spain
anindya.banerjee@imdea.org

Abstract. When proving the correctness of a method for slicing probabilistic programs, it was previously discovered by the authors that for a fixed point iteration to work one needs a non-standard starting point for the iteration. This paper presents and explores this technique in a general setting; it states the lemmas that must be established to use the technique to prove the correctness of a program transformation, and sketches how to apply the technique to slicing of probabilistic programs.

Keywords: Fixed Point Iteration · Program Slicing · Probabilistic Programming.

1 Setting

We wish to reason about the correctness of a program transformation of a "source semantics" ϕ into a "target semantics" γ, each semantics defined on a universe $\mathcal{U}$ of "subprograms". Typically, such correctness is established by demonstrating some "correctness relation" R such that $R\ \phi\ \gamma$ can be proved. But we shall show that for a certain application, the standard technique for such a proof cannot be adapted directly but requires modification.

We assume that for each $x \in \mathcal{U}$, $\phi(x)$ and $\gamma(x)$ are *continuous* functions from D to D where D is a *complete partial order* (cpo). Recall[3] that

- a cpo D is a set equipped with a partial order $\sqsubseteq$ that satisfies:
  if $\{d_i \mid i \geq 0\}$ is a chain (that is $d_i \sqsubseteq d_{i+1}$ for all $i \geq 0$) then it has a least upper bound, written $\sqcup\{d_i \mid i \geq 0\}$, thus:
  - $d_i \sqsubseteq \sqcup\{d_i \mid i \geq 0\}$ for all $i \geq 0$, and
  - if $d_i \sqsubseteq d'$ for all $i \geq 0$ then $\sqcup\{d_i \mid i \geq 0\} \sqsubseteq d'$.

  If in addition there is a least element $\bot$ (thus $\bot \sqsubseteq d$ for all $d \in \mathsf{D}$) we say that D is a *pointed* cpo.
- a function f, from a cpo D_1 into a cpo D_2, is continuous if for all chains $\{d_i \mid i \geq 0\}$ in D_1 it holds that
  - $\{f(d_i) \mid i \geq 0\}$ is a chain in D_2 (this implies that f is monotone)

[3] Throughout we employ elementary concepts of domain theory; see, e.g., Winskel's textbook [5].

- $f(\sqcup\{d_i \mid i \geq 0\}) = \sqcup\{f(d_i) \mid i \geq 0\}$.

We write $\mathsf{D}_1 \rightarrow_c \mathsf{D}_2$ for the space of continuous functions from D_1 to D_2.

– if D_1 and D_2 are cpos then also $\mathsf{D}_1 \rightarrow_c \mathsf{D}_2$ is a cpo, with the partial order defined pointwise, as is the least upper bound:

$$\sqcup\{f_k \mid k \geq 0\} = \lambda d \in \mathsf{D}_1.\, \sqcup\, \{f_k(d) \mid k \geq 0\}$$

and if $\bot$ is least in D_2 then $\lambda d.\bot$ is least in $\mathsf{D}_1 \rightarrow_c \mathsf{D}_2$.

2 Goal

We want ϕ and γ to be related as prescribed by a given correctness relation R over $(\mathsf{D} \rightarrow_c \mathsf{D})$; that is we shall require $R\ \phi\ \gamma$ which is an abbreviation of

$$\forall x \in \mathcal{U} : R\ (\phi\ x)\ (\gamma\ x). \tag{1}$$

We shall aim at finding a widely applicable recipe for establishing (1), but shall see that the most straightforward approach does not always work.

We shall assume that R is *admissible* in that if $\{f_i \mid i \geq 0\}$ and $\{g_i \mid i \geq 0\}$ are chains in $\mathsf{D} \rightarrow_c \mathsf{D}$ then

$$\text{if } R\ f_i\ g_i \text{ for all } i \geq 0 \text{ then } R\ (\sqcup\{f_i \mid i \geq 0\})\ (\sqcup\{g_i \mid i \geq 0\}). \tag{2}$$

3 Example Setting

We shall consider *probabilistic programming* (with [4] a seminal contribution and [3] giving a recent overview) where the effect of a program is to transform probability distributions, and shall express a program as a *probabilistic control-flow graph* [1]. In that setting,

– the subprograms in $\mathcal{U}$ are of the form (u, u') where u, u' are nodes such that u' postdominates u (that is, u' occurs on every path from u to an end node).

– the members of D are probability distributions where a probability distribution D maps a store, which is a (partial) map from program variables to integers or booleans, into a real number in [0..1] which is the "probability" of that store.

With $D_1 \sqsubseteq D_2$ defined to hold iff $D_1(s) \leq D_2(s)$ for all stores s, D will be a pointed cpo where

- the least element is $\lambda s.0$
- the least upper bound of a chain $\{D_i \mid i \geq 0\}$ is $\lambda s.S(s)$ where $S(s)$ is the supremum of the chain $\{D_i(s) \mid i \geq 0\}$.

We can think of $\phi(u, u')$ as denoting the distribution transformation done along the path(s) from u to u': if D describes how stores are distributed at point u, then $\phi(u, u')(D)$ describes how stores are distributed at point u'.

4 Example

We shall consider the probabilistic structured program in Figure 1 which

1. assigns 0 to the variables p and q
2. assigns random boolean values to the variables b and h, with the choices independent and unbiased so that each outcome (say $b = \text{true}$ and $h = \text{false}$) has probability $\frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$
3. if h is true, enters an infinite loop
4. if b is true (false), increments p (q) until a random boolean choice becomes true (which in each iteration will happen with probability $\frac{1}{2}$)

The corresponding control-flow graph is depicted in Figure 2.

We will expect that for arbitrary initial D, the distribution $\phi(\mathsf{start}, \mathsf{end})\,(D)$ is given by Table 1. To understand why say $\{b \mapsto F,\ h \mapsto F,\ p \mapsto 0,\ q \mapsto 2\}$ has probability $\frac{1}{2^{2+2}} = \frac{1}{2^4}$, observe that to obtain this store we need four fortunate binary outcomes:

1. b becomes F (so that q, rather than p, is incremented)
2. h becomes F (so the infinite loop is not taken)
3. the first random choice is F (so q will become 2)
4. the second random choice is T (so q remains 2)

Observe that the probabilities add up to $\frac{1}{2}$, since if $h = T$ we never get to end.

5 Slicing the Example

In general, if we care about only certain variables, we would like to *slice* away all parts of the program that do not affect those variables; see Danicic et al. [2] for a unifying framework for program slicing.

For our example, assume that the part of the store we care about is the value of q, whereas we do not care about the value of p. Observe that we still need to care about the value of b because it is *relevant* in that it determines whether q is incremented, but p is irrelevant, as is h as it does not impact the *relative* distribution of q (since h and b are independent). Accordingly, we can abstract the distribution from Table 1 into Table 2.

We then slice the given program so as to eliminate irrelevant operations, such as the assignments to p and h, and even the loop on h (as we consider *termination-insensitive* slicing). Figure 3 shows the resulting control-flow graph, which can be further simplified and corresponds to the structured program in Figure 4.

That control-flow graph induces the target semantics γ. It is easy to see that for arbitrary initial D, the distribution $\gamma(\mathsf{start}, \mathsf{end})\,(D)$ is given by Table 3.

By comparing Tables 2 and 3 we see:

$$\phi(\mathsf{start}, \mathsf{end})\,(D) = \frac{1}{2} \cdot \gamma(\mathsf{start}, \mathsf{end})\,(D)$$

```
p := 0; q := 0
b := random boolean value
h := random boolean value
if h
    while true
        skip
else
    skip
if b
    repeat
        p := p + 1
    until random boolean choice becomes true
else
    repeat
        q := q + 1
    until random boolean choice becomes true
```

Fig. 1. The source code of our running example.

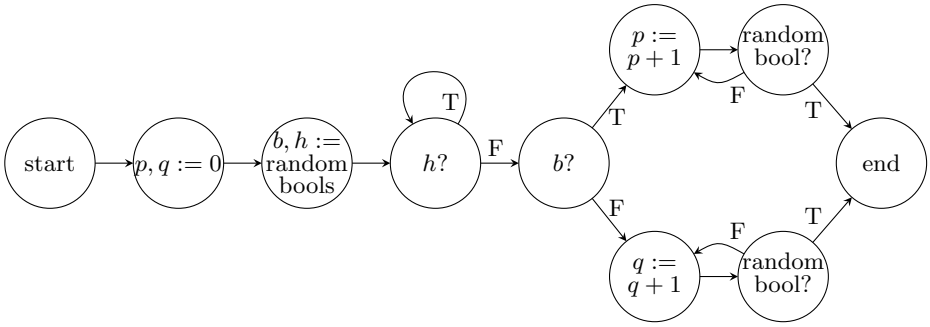

Fig. 2. The source control-flow graph of our running example.

s	$\phi(\mathsf{start}, \mathsf{end})\,(D)$	
$\{b \mapsto T,\ h \mapsto F,\ p \mapsto k,\ q \mapsto 0\}$	$\frac{1}{2^{k+2}}$	for $k \geq 1$
$\{b \mapsto F,\ h \mapsto F,\ p \mapsto 0,\ q \mapsto k\}$	$\frac{1}{2^{k+2}}$	for $k \geq 1$

Table 1. The distribution produced by the source semantics.

s	$\phi(\mathsf{start}, \mathsf{end})\,(D)\,s$	
$\{b \mapsto T,\ q \mapsto 0\}$	$\frac{1}{4}$	
$\{b \mapsto F,\ q \mapsto k\}$	$\frac{1}{2^{k+2}}$	for $k \geq 1$

Table 2. The relevant part of the distribution produced by the source semantics.

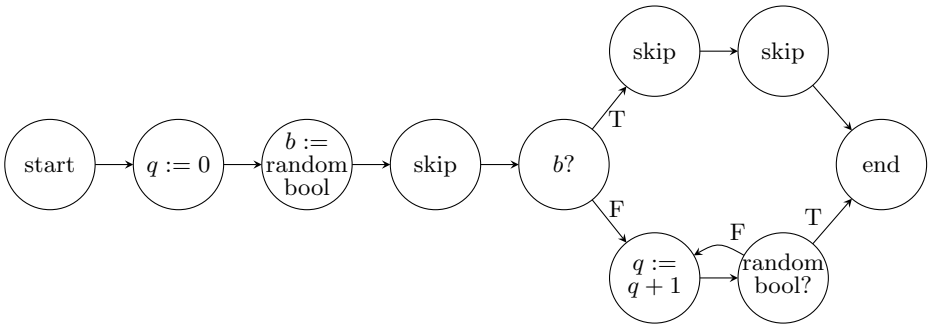

Fig. 3. The target control-flow graph of our example program.

$q := 0$; $b :=$ random boolean value
if b
    skip
else
    repeat
        $q := q + 1$
    until random boolean choice becomes true

Fig. 4. The target code of our example program.

s	$\gamma(\mathsf{start}, \mathsf{end})\,(D)\,(s)$	
$\{b \mapsto T,\ q \mapsto 0\}$	$\frac{1}{2}$	
$\{b \mapsto F,\ q \mapsto k\}$	$\frac{1}{2^{k+1}}$	for $k \geq 1$

Table 3. The distribution produced by the target semantics.

and indeed, for this application, our correctness relation R requires:

$$R\ (\phi(u,u'))\ (\gamma(u,u')) \text{ iff } \forall D \in \mathsf{D}\ \exists c : \phi(u,u')(D) = c \cdot \gamma(u,u')(D).$$

That is, the transformation must not affect the *relative* probabilities of the various stores.

We shall now study how to *prove* that this correctness relation holds.

6 Basic Approach

Recall the fixed-point theorem for cpos:

> if D is a pointed cpo, and F belongs to $\mathsf{D} \to_c \mathsf{D}$, then $\{F^i(\bot) \mid i \geq 0\}$ is a chain with
>
> $$F(\sqcup\{F^i(\bot) \mid i \geq 0\}) = \sqcup\{F^i(\bot) \mid i \geq 0\}.$$
>
> The chain property follows by induction, with the base case being $F^0(\bot) = \bot \sqsubseteq F^1(\bot)$, and the inductive step using that F is monotone:
>
> $$F^{i+1}(\bot) = F(F^i(\bot)) \sqsubseteq F(F^{i+1}(\bot)) = F^{i+2}(\bot).$$
>
> That the least upper bound is a fixed point follows from the continuity of F:
>
> $$F(\sqcup\{F^i(\bot) \mid i \geq 0\}) = \sqcup\{F(F^i(\bot)) \mid i \geq 0\} = \sqcup\{F^i(\bot) \mid i \geq 0\}$$
>
> and it is actually the least fixed point since if also f is a fixed point then for all $i \geq 0$ we have $F^i(\bot) \sqsubseteq f$, as can be proved by induction with the inductive step being $F^{i+1}(\bot) = F(F^i(\bot)) \sqsubseteq F(f) = f$.

It will typically be the case that ϕ and γ have been defined as

$$\phi = \sqcup\{F^i(\bot) \mid i \geq 0\}$$
$$\gamma = \sqcup\{G^i(\bot) \mid i \geq 0\}$$

for functionals F and G with functionality

$$F, G : (\mathcal{U} \to \mathsf{D} \to_c \mathsf{D}) \to_c (\mathcal{U} \to \mathsf{D} \to_c \mathsf{D}).$$

where $\mathcal{U} \to \mathsf{D} \to_c \mathsf{D}$ is a pointed cpo since

- $\mathsf{D} \to_c \mathsf{D}$ is a pointed cpo, since D is
- equipped with the discrete partial order, $\mathcal{U}$ is a cpo and all functions from $\mathcal{U}$ are continuous.

Thus the fixed point theorem can be applied; we shall often write ϕ_k for $F^k(\bot)$ and γ_k for $G^k(\bot)$.

To accomplish our goal (1), that is to prove that $R\ \phi\ \gamma$, due to R being admissible (2) it thus suffices to show that

$$R\ \phi_k\ \gamma_k \text{ for all } k \geq 0 \tag{3}$$

which indeed often can be proved by induction in k. But we shall now see that (3) may *not* always hold.

7 Problem

In the setting of control flow graphs, we will expect ϕ_k and γ_k to be the meaning of the program assuming we are allowed at most $k-1$ "backwards moves", that is control flowing away from the end node (as will necessarily happen each time a loop is iterated). In particular, ϕ_1 will allow only forward moves, and thus we would expect (ignoring the values of h and p) that $\phi_1(\mathsf{start}, \mathsf{end})\,(D)$ is given by

s	$\phi_1(\mathsf{start}, \mathsf{end})\,(D)$
$\{b \mapsto T,\ q \mapsto 0\}$	$\frac{1}{8}$
$\{b \mapsto F,\ q \mapsto 1\}$	$\frac{1}{8}$

whereas ϕ_2 allows for at most one backwards move and thus we would expect that $\phi_2(\mathsf{start}, \mathsf{end})\,(D)$ is given by

s	$\phi_2(\mathsf{start}, \mathsf{end})\,(D)$
$\{b \mapsto T,\ q \mapsto 0\}$	$\frac{1}{8} + \frac{1}{16}$
$\{b \mapsto F,\ q \mapsto 1\}$	$\frac{1}{8}$
$\{b \mapsto F,\ q \mapsto 2\}$	$\frac{1}{16}$

Thus the probability that execution ends with $q = 0$ is $\frac{3}{16}$, which makes sense since for this to happen we need two independent events:

- b becomes T and h becomes F; this has probability $\frac{1}{4}$
- the random choice at the upper branch becomes T either first time or second time; this has probability $\frac{3}{4}$.

In general, for $k \geq 1$, we would expect that the distribution $\phi_k(\mathsf{start}, \mathsf{end})\,(D)$ is given by Table 4, in particular we have

$$\begin{aligned}
&\phi_k(\mathsf{start}, \mathsf{end})\,(D)\ \{b \mapsto T,\ q \mapsto 0\} \\
&= \sum_{i=1}^{k} \phi_k(\mathsf{start}, \mathsf{end})\,(D)\ \{b \mapsto T,\ p \mapsto i,\ q \mapsto 0\} \\
&= \sum_{i=1}^{k} \frac{1}{2^{i+2}} = \frac{1}{8} + \frac{1}{16} + \cdots \frac{1}{2^{k+2}} = \frac{1}{4} - \frac{1}{2^{k+2}}
\end{aligned}$$

On the other hand, looking at the target control-flow graph (Figure 3), we see that no backwards moves are needed for q to end up being 0, and hence we would expect that the distribution $\gamma_k(\mathsf{start}, \mathsf{end})\,(D)$ is given by Table 5.

By comparing Tables 4 and 5 we see that (3) does **not** hold, in particular

$$\phi_k(\mathsf{start}, \mathsf{end})\,(D) \neq \frac{1}{2} \cdot \gamma_k(\mathsf{start}, \mathsf{end})\,(D).$$

We shall now show how to repair this, by replacing $\{\phi_k \mid k \geq 0\}$ by a more appropriate chain.

8 Refined Approach

To prepare for the correctness proof, the idea is to first transform the source control-flow graph. This involves replacing loops that are irrelevant, and eventually terminate, by skip.

In our setting, with p considered irrelevant, this will result in the control-flow graph depicted in Figure 5.

With that control-flow graph as our starting point, we may define a chain $\{\phi'_k \mid k \geq 0\}$ where for $k \geq 1$, we would expect that the distribution $\phi'_k(\mathsf{start}, \mathsf{end})\,(D)$ is given by Table 6.

By comparing Tables 6 and 5 we see that for $k \geq 1$,

$$\phi'_k(\mathsf{start}, \mathsf{end})\,(D) = \frac{1}{2} \cdot \gamma_k(\mathsf{start}, \mathsf{end})\,(D)$$

which suggests that we do indeed have $R\ \phi'_k\ \gamma_k$ when $k \geq 1$.

This is a special case, of the general idea that we shall now present.

9 General Result

Recall the setting: with D a pointed cpo, $\mathcal{U}$ a set of subprograms, F and G functionals of type $(\mathcal{U} \to \mathsf{D} \to_c \mathsf{D}) \to_c (\mathcal{U} \to \mathsf{D} \to_c \mathsf{D})$, and with R an admissible correctness relation, we want to prove $R\ \phi\ \gamma$ where

$$\phi = \sqcup\{F^i(\bot) \mid i \geq 0\}$$
$$\gamma = \sqcup\{G^i(\bot) \mid i \geq 0\}.$$

For that purpose, we might be able to use the following result:

Theorem 1. *With $X \subseteq \mathcal{U}$ given, for $k \geq 0$ define $\phi'_k = F^k(\phi'_0)$ with ϕ'_0 given by*

$$\phi'_0(x) = \phi(x) \quad \textit{if } x \in X$$
$$\phi'_0(x) = \bot \quad \textit{otherwise}$$

(thus the starting point for the iteration may be above *the standard starting point $\bot$), and similarly for $k \geq 0$ define $\gamma'_k = G^k(\gamma'_0)$ where γ'_0 is given by:*

$$\gamma'_0(x) = \gamma(x) \quad \textit{if } x \in X$$
$$\gamma'_0(x) = \bot \quad \textit{otherwise.}$$

Now assume that the below 3 properties all hold:

$$\forall k \geq 0 : R\ \phi'_k\ \gamma'_k \tag{4}$$

$$\forall x \in X : \phi(x) \sqsubseteq \phi'_1(x) \tag{5}$$

$$\forall x \in X : \gamma(x) \sqsubseteq \gamma'_1(x). \tag{6}$$

Then we do have $R\ \phi\ \gamma$.

s	$\phi_k(\mathsf{start}, \mathsf{end})\,(D)$	
$\{b \mapsto T,\ q \mapsto 0\}$	$\frac{1}{4} - \frac{1}{2^{k+2}}$	
$\{b \mapsto F,\ q \mapsto i\}$	$\frac{1}{2^{i+2}}$	when $1 \leq i \leq k$

Table 4. The relevant part of the source distribution after $k \geq 1$ iterations.

s	$\gamma_k(\mathsf{start}, \mathsf{end})\,(D)$	
$\{b \mapsto T,\ q \mapsto 0\}$	$\frac{1}{2}$	
$\{b \mapsto F,\ q \mapsto i\}$	$\frac{1}{2^{i+1}}$	when $1 \leq i \leq k$

Table 5. The relevant part of the target distribution after $k \geq 1$ iterations.

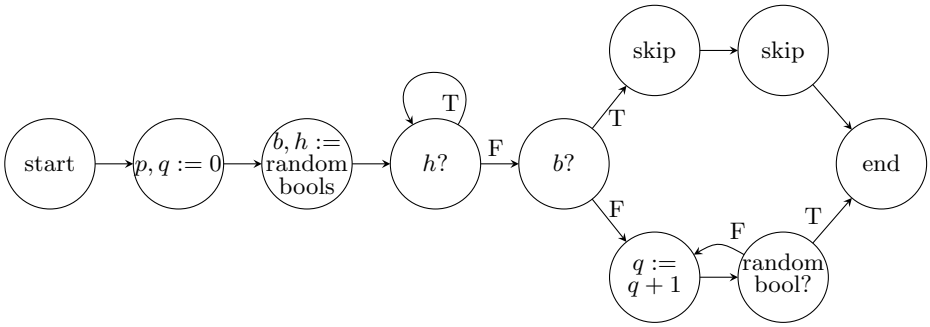

Fig. 5. The modified source control-flow graph of our example program.

s	$\phi'_k(\mathsf{start}, \mathsf{end})\,(D)$	
$\{b \mapsto T,\ q \mapsto 0\}$	$\frac{1}{4}$	
$\{b \mapsto F,\ q \mapsto i\}$	$\frac{1}{2^{i+2}}$	when $1 \leq i \leq k$

Table 6. The relevant part of the revised source distribution after $k \geq 1$ iterations.

Proof. Observe that due to (4) and (2), our goal $R\ \phi\ \gamma$ will follow if we can prove that

$\{\phi'_k \mid k \geq 0\}$ is a chain with $\sqcup\{\phi'_k \mid k \geq 0\} = \phi$

$\{\gamma'_k \mid k \geq 0\}$ is a chain with $\sqcup\{\gamma'_k \mid k \geq 0\} = \gamma$

and by symmetry it suffices to prove the first property. To do so, first observe that

$$\forall x \in \mathcal{U} : \phi'_0(x) \sqsubseteq \phi'_1(x)$$

which is trivial if $x \notin X$ and otherwise follows from (5). By monotonicity of F, this implies that $\{\phi'_k \mid k \geq 0\}$ is a chain. Writing ϕ_i for $F^i(\bot)$ (with $i \geq 0$), we obviously have

$$\phi_0 \sqsubseteq \phi'_0 \sqsubseteq \phi$$

which implies (since F is monotone and has ϕ as a fixed point)

$$\phi_1 \sqsubseteq \phi'_1 \sqsubseteq F(\phi) = \phi$$

and in general (by induction in k)

$$\phi_k \sqsubseteq \phi'_k \sqsubseteq \phi \text{ for all } k \geq 0$$

Because $\sqcup\{\phi_k \mid k \geq 0\} = \phi$, we obtain the desired $\sqcup\{\phi'_k \mid k \geq 0\} = \phi$.

To summarize, we see that the successful application of our approach requires choosing a suitable set X, to be called an *exclusion set* as it is excluded from the iteration process, and then establishing (4), (5), (6).

In the next section, we shall see that our example setting does allow a suitable X to be found; we shall not address whether there are heuristics for that in a general setting.

10 An Application

We have presented a recipe for proving the correctness of an algorithm for slicing probabilistic programs. An elaborate, albeit unmotivated, proof of correctness was furnished in the authors' earlier work [1]. In contrast, the technique proposed in Section 9 abstracts the essence of the proof structure, and clarifies the key lemmas that are needed.

In [1], slicing involves finding suitable node sets Q and Q_0, where Q are those nodes that impact the variable(s) of interest, whereas Q_0 are those of the remaining nodes that may cause non-termination. The exclusion set X (cf. Section 9) is then defined to contain the pairs $(v, v') \in \mathcal{U}$ where (using the terminology of [1]) "v stays outside $Q \cup Q_0$ until v'", a property which (as proved in [1][Lemma 5.8]) ensures that from v, control will eventually (without risk of non-termination) reach v', without affecting relevant variables.

For the control-flow graph in Figure 2, two nodes are not in $Q \cup Q_0$: the node that increments p, and its successor (the node that tests h may cause non-termination, and all other nodes impact q). With u_p the node that increments p we thus have

$$(u_p, \mathsf{end}) \in X \tag{7}$$

In Section 9 we listed the properties sufficient for correctness: (4), (5), (6) and admissibility (2). They are all stated in [1] (though with quite different naming conventions[4]):

- property (4) is stated as [1][Lemma 6.9] (proved by induction in k, and with a case analysis on $(v, v') \in \mathcal{U}$)
- property (5) is a special case of [1][Lemma 6.2] which (renamed) states that $\phi'_k(x) = \phi(x)$ for all $x \in X$ and all $k \geq 0$
- property (6) is a special case of [1][Lemma 4.31] which (renamed) entails that $G(g_1)(x) = G(g_2)(x)$ for all $x \in X$ and all functions g_1, g_2, from which we for $x \in X$ infer $\gamma(x) = G(\gamma)(x) = G(\gamma'_0)(x) = \gamma'_1(x)$
- property (2) follows from the calculation in [1][Proof of Theorem 6.6], except that we need to *rectify* [1][Lemma 6.13]. That lemma claims that a certain sequence of reals is a chain; this is not necessarily the case, but (as shown in Appendix A) still the sequence will have a limit, and that is sufficient to establish property (2).

11 The Intuition Behind the Exclusion Set

In our application, where the source control-flow graph of Figure 2 is transformed into the target control-flow graph of Figure 3. consider paths from start to end along which b becomes true, thus causing the value of q at end to be 0.

- For the target program (Figure 3), the only such path is the one that takes the upper branch. That path has no loops and hence the fixed point (which assigns $q = 0$ the probability $\frac{1}{2}$) is reached already in the first iteration, as can be seen by comparing the first row of Table 5 with the first row of Table 3.
- On the other hand, for the source program (Figure 2), there are denumerable many such paths; all will follow the upper branch but one path will not loop, another path will loop once, another path will loop twice, etc. Hence the fixed point (which assigns $q = 0$ the probability $\frac{1}{4}$) is reached only in the limit of the iterations, as can be seen by comparing the first row of Table 4 with the first row of Table 2.

We conclude that source iteration and target iteration get out of lockstep, which is why we cannot establish the correctness relation between the iterands.

Our approach is to let the iteration on the source program start with the parts in the exclusion set X already at their fixed point. This is accomplished

[4] [1] uses "ω" for what the current paper calls "ϕ", uses "γ" for "ϕ'", uses "ϕ" for "γ", uses "Φ" for "γ'", uses "H_V" for "F", and uses "H_Q" for "G".

by ensuring that if (u, u') in X then no path from u to u' has loops. Recall (7) that with u_p the node that increments p we have $(u_p, \mathsf{end}) \in X$. And indeed, no path from u_p to end has loops in the control-flow graph in Figure 5, as we put skip at node u_p and at its successor.

As a consequence, the revised iterands of the source program (Table 6) are in lockstep with the iterands of the target program (Table 5), with each iterand of the source being $\frac{1}{2}$ of the corresponding iterand of the target.

12 Perspectives

We have distilled a proof technique that in at least one (recently published) situation [1] comes to the rescue when the standard technique falls short. We conjecture that other applications exist, and encourage the discovery of such.

More generally, while we were surprised that we had to begin the fixed point iteration above the bottom element, we believe it likely that other researchers have encountered and reported similar situations, though so far we have not come across it in our literature search.

We offer this paper as a tribute to Alan Mycroft who has inspired the authors since early in their careers, by his kindness and enthusiasm, and his work on program analysis where the computation of fixed points is ubiquitous.

Acknowledgments. We thank the anonymous reviewers for their comments which in various ways helped improve the presentation of the paper, and also thank Patrick Cousot for looking into our findings. Banerjee's research is based on work supported by the National Science Foundation (NSF), while working at the Foundation. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the NSF.

References

1. Amtoft, T., Banerjee, A.: A theory of slicing for imperative probabilistic programs. ACM Trans. Program. Lang. Syst. **42**(2) (Apr 2020). https://doi.org/10.1145/3372895
2. Danicic, S., Barraclough, R.W., Harman, M., Howroyd, J.D., Kiss, Á., Laurence, M.R.: A unifying theory of control dependence and its application to arbitrary program structures. Theoretical Computer Science **412**(49), 6809–6842 (Nov 2011). https://doi.org/10.1016/j.tcs.2011.08.033
3. Gordon, A.D., Henzinger, T.A., Nori, A.V., Rajamani, S.K.: Probabilistic programming. In: Future of Software Engineering Proceedings. pp. 167—-181. FOSE 2014, Association for Computing Machinery, New York, NY, USA (2014). https://doi.org/10.1145/2593882.2593900
4. Kozen, D.: Semantics of probabilistic programs. Journal of Computer and System Sciences **22**(3), 328–350 (1981), https://doi.org/10.1016/0022-0000(81)90036-2
5. Winskel, G.: The Formal Semantics of Programming Languages. MIT Press, Cambridge, MA, USA (1993), https://doi.org/10.7551/mitpress/3054.001.0001

A Rectifying Lemma 6.13 in [1]

We shall refer to the setting of [1], where $\mathcal{U}$ is the set of pairs (v, v') such that v' postdominates v, and D is the set of probability distributions. With $c_{k,D}^{v,v'}$ a real number as defined in Definition 6.10, we shall replace Lemma 6.13 by the following result:

> **Lemma 6.13, restated**. Assume $\{D_k \mid k \geq 0\}$ is a chain in D, and that $\{f_k \mid k \geq 0\}$ is a chain of functions in $\mathsf{D} \to_c \mathsf{D}$.
> Then for all $(v, v') \in \mathcal{U}$, the limit $lim_{k\to\infty}\, c_{k,f_k(D_k)}^{v,v'}$ does exist.

Observe that as a special case, with each f_k the identity and each D_k equal D, we have

> For all $(v, v') \in \mathcal{U}$ and $D \in \mathsf{D}$, there does exist a real number c such that
>
> $$lim_{k\to\infty}\, c_{k,D}^{v,v'} = c$$

which suffices to prove Theorem 6.6, stating the correctness of slicing in [1].

We shall prove the restated Lemma 6.13 by induction in the maximum length of an acyclic path from v to v', where two cases are non-trivial; in each we let v'' denote the first proper postdominator of v:

- Assume $v' \neq v$ and $v' \neq v''$. We can inductively assume that there exists c'', c' such that

  $$lim_{k\to\infty}\, c_{k,f_k(D_k)}^{v,v''} = c''$$
  $$lim_{k\to\infty}\, c_{k,\gamma_k^{(v,v'')}(f_k(D_k))}^{v'',v'} = c'$$

  but then

  $$lim_{k\to\infty}\, c_{k,f_k(D_k)}^{v,v'} = lim_{k\to\infty}\, (c_{k,f_k(D_k)}^{v,v''} \cdot c_{k,\gamma_k^{(v,v'')}(f_k(D_k))}^{v'',v'}) = c'' \cdot c'.$$

- Assume $v' = v''$ but v does not stay outside Q_0 until v'.
  Since $\{f_k(D_k) \mid k \geq 0\}$ and $\{\gamma_k^{(v,v')}(f_k(D_k)) \mid k \geq 0\}$ are chains in the cpo D, there exists D_1 and D_2 such that

  $$lim_{k\to\infty}\, f_k(D_k) = D_1$$
  $$lim_{k\to\infty}\, \gamma_k^{(v,v')}(f_k(D_k)) = D_2.$$

  By Lemma 4.1 we see that

  $$lim_{k\to\infty}\, (\sum f_k(D_k)) = \sum D_1$$
  $$lim_{k\to\infty}\, (\sum \gamma_k^{(v,v')}(f_k(D_k))) = \sum D_2.$$

If $D_1 = 0$ we for all k have $f_k(D_k) = 0$ and thus $c^{v,v'}_{k,f_k(D_k)} = 1$ making the claim trivial. We can thus assume that $D_1 \neq 0$, in which case we see that

$$lim_{k\to\infty}\, c^{v,v'}_{k,f_k(D_k)} = lim_{k\to\infty}\, \frac{\sum \gamma^{(v,v')}_k(f_k(D_k))}{\sum f_k(D_k)} = \frac{\sum D_2}{\sum D_1}$$

which yields the claim.

But observe that while $\{c^{v,v'}_{k,f_k(D_k)} \mid k \geq 0\}$ has a limit, we apparently have no assurance it is a chain (it is rather a *ratio* between two chains) and thus the original Lemma 6.13 makes an unwarranted claim.