Two Approximate-Programmability Birds, One Statistical-Inference Stone

Adrian Sampson
University of Washington

APPROX 2014
EnerJ, the Language of Good-Enough Computing

submitted 8 months ago by jms_nh

23 comments  share  save  hide  give gold  report  instapaper

... 

[-] jtra  3 points 8 months ago

Good luck debugging that...

permalink  save  give gold

[-] MorePudding  10 points 8 months ago

Oh god, this sounds awful .. and it has enough potential to actually make me want to shove it down our throats by force (i.e. people recalculating things in a standard/"unofficial" hardware that tries to work around the limitations of that).
Assisted approximate programming

Statistical inference

Cheap check generation
1. Assisted approximate programming

2. Cheap check generation

3. Statistical inference

4. Next steps
Expressing quality

\[ \Pr\left[ d(f(x), f'(x)) \leq b \right] \geq p \]

original program
relaxed version
input
distance metric
bounding parameters
Assisted approximate programming

$f \rightarrow f'$
int p = 5;
int a = 7;
for (int x = 0..) {
    a += func(2);
    int z;
    z = p * 2;
    p += 4;
}
a /= 9;
func2(p);
a += func(2);
int y;
z = p * 22 + z;
p += 10;

int p = 5;
@Approx int a = 7;
for (int x = 0..) {
    a += func(2);
    @Approx int z;
    z = p * 2;
    p += 4;
}
a /= 9;
func2(p);
a += func(2);
@Approx int y;
z = p * 22 + z;
p += 10;
Assisted approximate programming

\[ f \rightarrow f' \]

\[ \Pr[|d(f(x), f'(x))| \leq b] \geq p \]

ExpAX [Esmaeilzadeh+]
Syndy [Misailovic and Rinard, WACAS]
Optimization in Rely [Misailovic+]

⋮
Assisted approximate programming

Statistical inference

Cheap check generation
Quality: the fantasy

Correctness Probability

Inputs

average probability
Quality: the reality

Correctness Probability

Inputs

average probability
Cheap checks to fall back to precise execution

Correctness Probability

Inputs

average probability
Cheap checks

\[
f \quad f'
\]
\[
\Pr[d(f(x), f'(x)) \leq b] \geq p
\]

\[
x \quad \text{s.t.}
\]
\[
\Pr[d(f(x), f'(x)) \leq b] \geq p
\]
Assisted approximate programming

Statistical inference

Cheap check generation
def dist(x1, y1, x2, y2):
    return sqrt((x1 - x2) ** 2 + (y1 - y2) ** 2)

approximate operations
def dist(x1, y1, x2, y2):
    return sqrt((x1 - x2 + error()) ** 2
                + (y1 - y2 + error()) ** 2)
                + error()
def dist(x1, y1, x2, y2):
    return sqrt((x1 - x2 + error(?)) ** 2
               + (y1 - y2 + error(?)) ** 2)
               + error(?)

S.t. \[ Pr[d(f(x), f'(x)) \leq b] \geq p \]
Cheap check generation as statistical inference

\[
x_1 = \text{dist}(?) \\
y_1 = \text{dist}(?) \\
x_2 = \text{dist}(?) \\
y_2 = \text{dist}(?)
\]

```python
def dist(x1, y1, x2, y2):
    return sqrt((x1 - x2 + error()) ** 2 + (y1 - y2 + error()) ** 2) + error()
```

\[\text{s.t. } \Pr[d(f(x), f'(x)) \leq b] \geq p\]
First steps
First steps: translate to a probability distribution

```c
int p = 5;
int a = 7;
for (int x = 0..) {
    a += func(2);
    int z;
    z = p * 2;
    p += 4;
}

a /= 9;
func2(p);
`a += func(2);
int y;
z = p * 22 + z;
p += 10;
```
First steps:
statistical inference with constraints?

\[ \Pr[d(f(x), f'(x)) \leq b] \geq p \]
First steps: statistical inference with constraints and objectives?

\[
\minimize \quad \Pr[d(f(x), f'(x)) \leq b] \geq p
\]
First steps: statistical inference with constraints and objectives scalably?

\[ \Pr[d(f(x), f'(x)) \leq b] \geq p \]