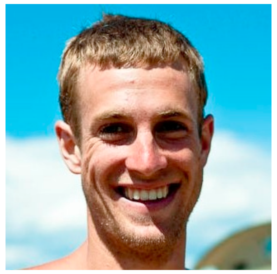
Refuting Heap Reachability

Bor-Yuh Evan Chang
University of Colorado Boulder



Sam Blackshear
CU Boulder

Manu Sridharan
Samsung

January 20, 2014
VMCAI 2014

PLV

At my first VMCAI in 2005 (Paris)

as a young PhD student

At my first VMCAI in 2005 (Paris)

as a young PhD student

“Spirit of VMCAI” introduced in my “formative academic years”

A bug that manifests spectacularly ...

A bug that manifests spectacularly …

A bug that manifests spectacularly ...

A bug that manifests spectacularly …

Wow! Android memory leaks underly rotation-based crashes.

Wow! Android memory leaks underly rotation-based crashes.

How?!?

Wow! Android memory leaks underly rotation-based crashes.

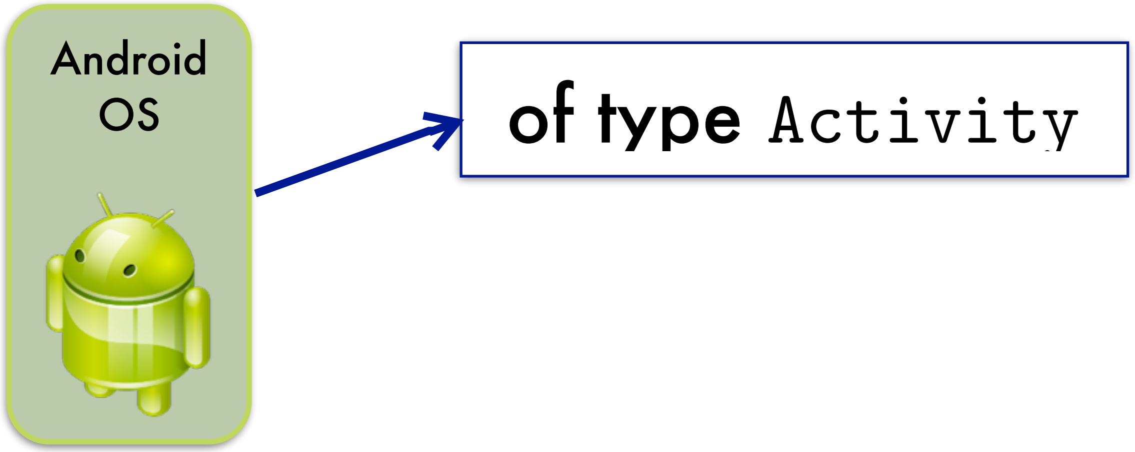
How?!?

`Activity` objects encapsulate the UI

Wow! Android memory leaks underly rotation-based crashes.

How?!?

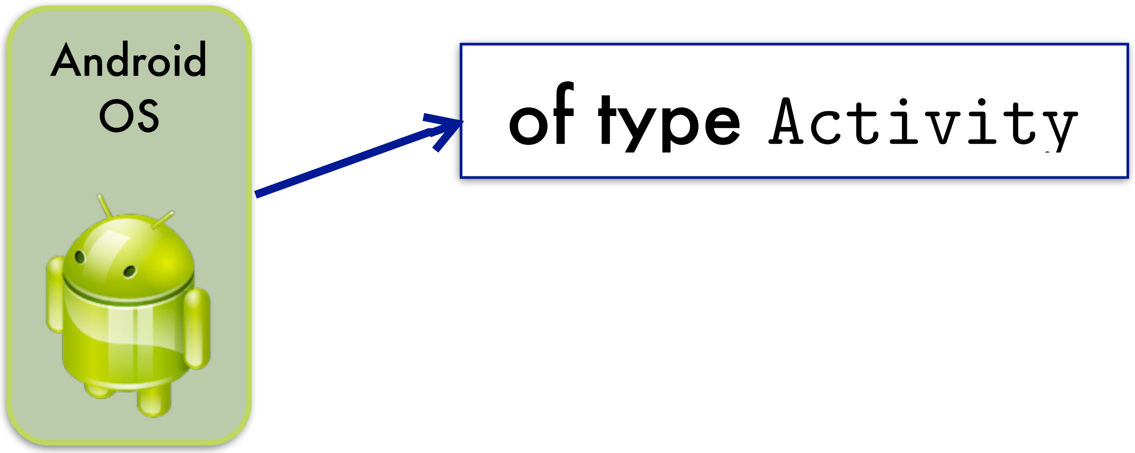
Activity objects encapsulate the UI



Wow! Android memory leaks underly rotation-based crashes.

How?!?

`Activity` objects encapsulate the UI



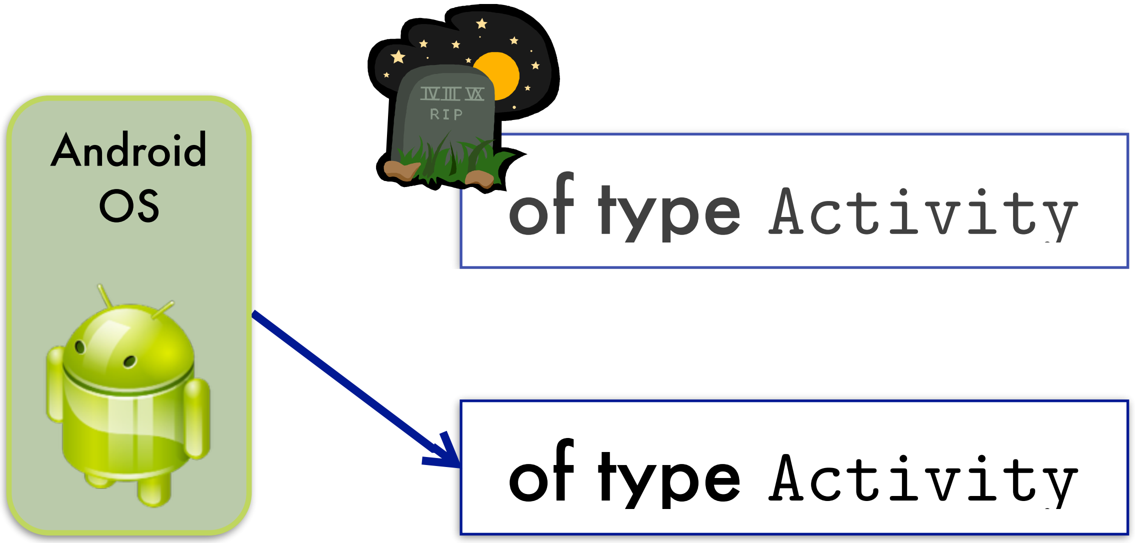
Android OS

of type `Activity`

Wow! Android memory leaks underly rotation-based crashes.

How?!?

Activity objects encapsulate the UI



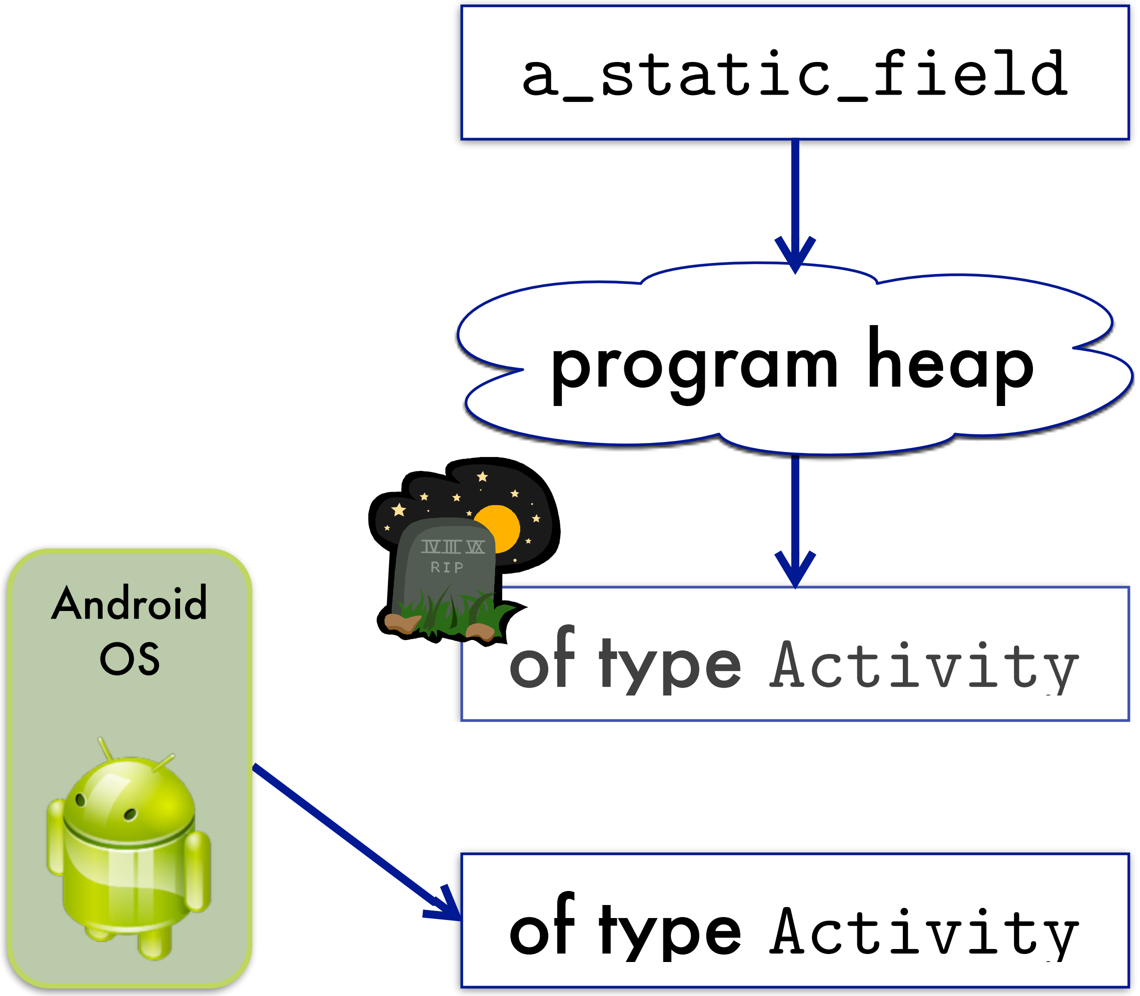
Android OS

of type Activity

of type Activity

Wow! Android memory leaks underly rotation-based crashes.

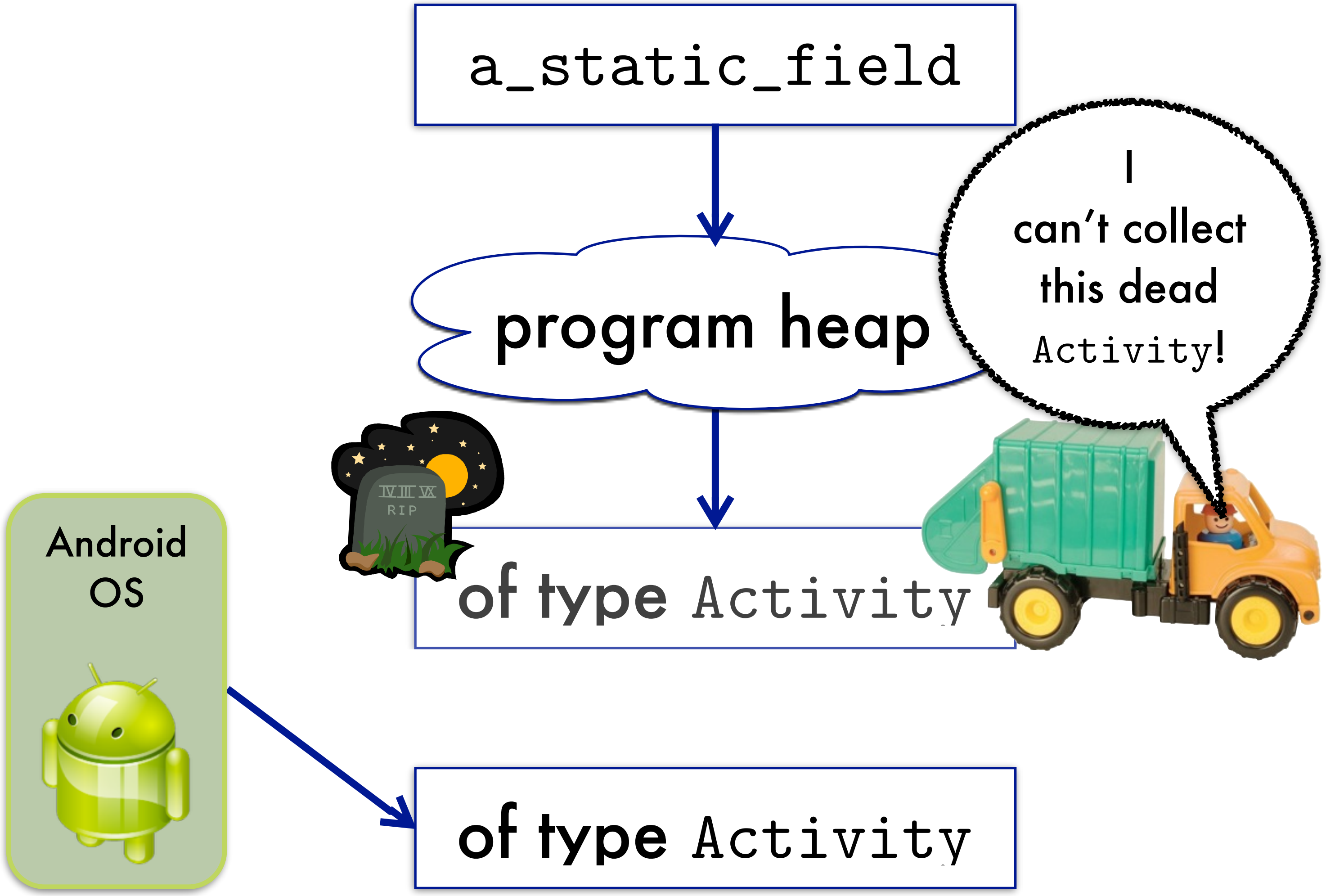
How?!?



Activity objects encapsulate the UI

Wow! Android memory leaks underly rotation-based crashes.

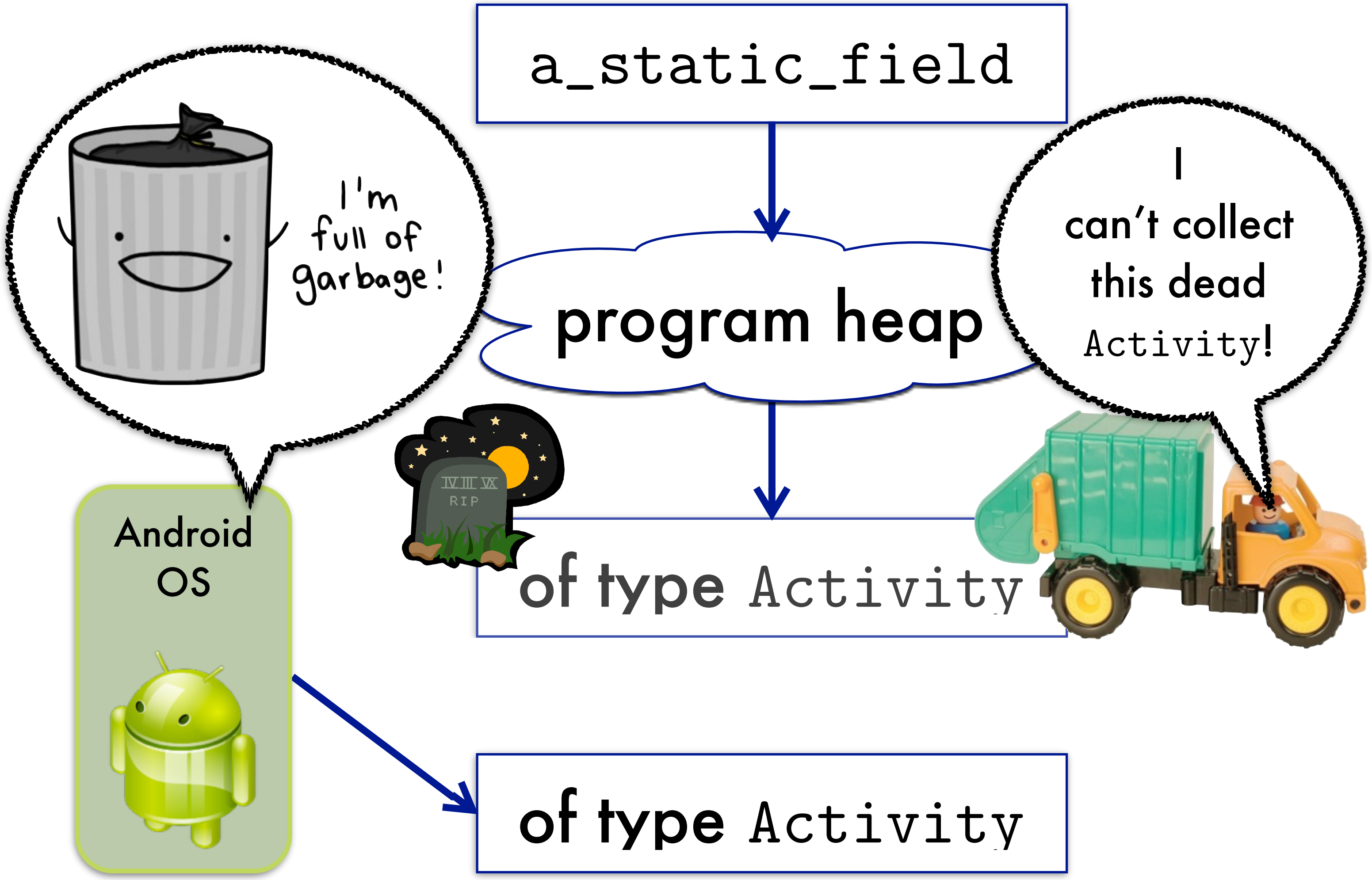
How?!?



Activity objects encapsulate the UI

Wow! Android memory leaks underly rotation-based crashes.

How?!?



a_static_field

program heap

of type Activity

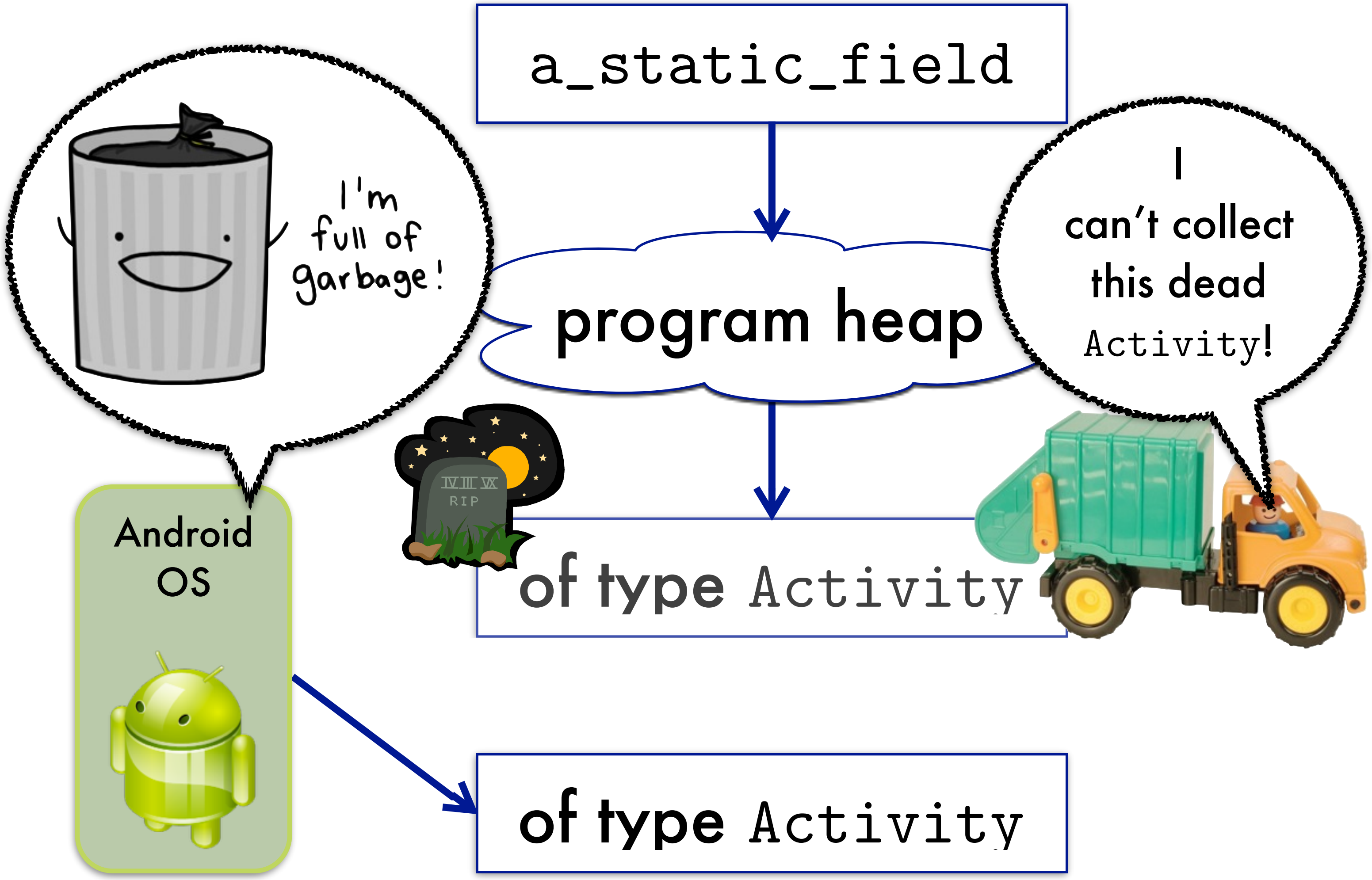
Android OS

of type Activity

Activity objects encapsulate the UI

Wow! Android memory leaks underly rotation-based crashes.

How?!?



a_static_field

program heap

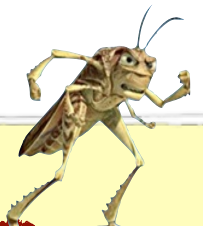
of type Activity

Android OS

of type Activity

I can't collect this dead Activity!

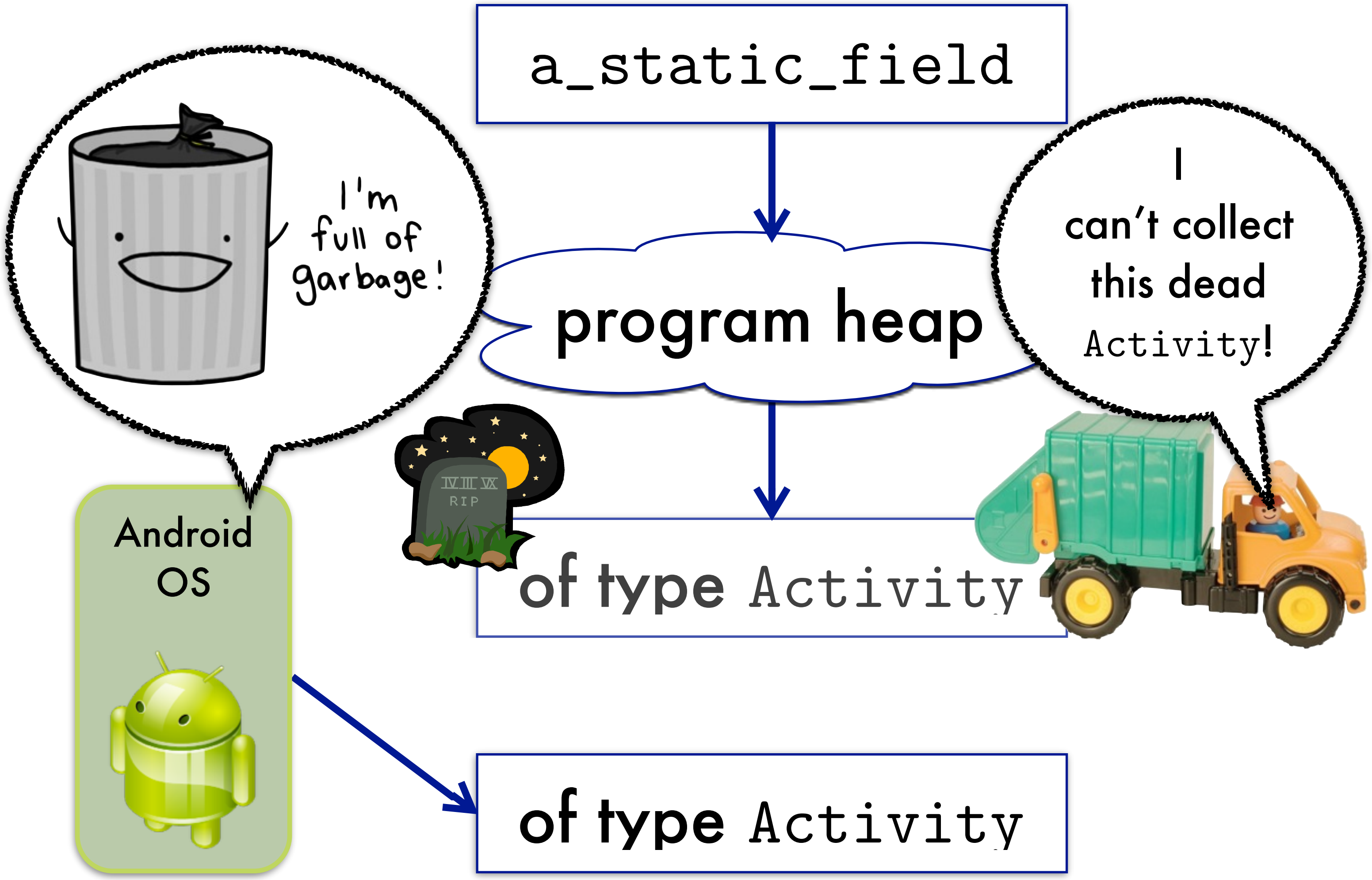
Activity objects encapsulate the UI



Bug: Holding reference to "old" Activity

Wow! Android memory leaks underly rotation-based crashes.

How?!?



a_static_field

program heap

I can't collect this dead Activity!

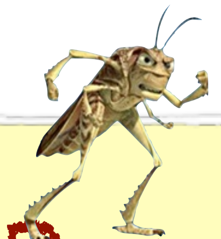
of type Activity

Android OS

of type Activity

Activity objects encapsulate the UI

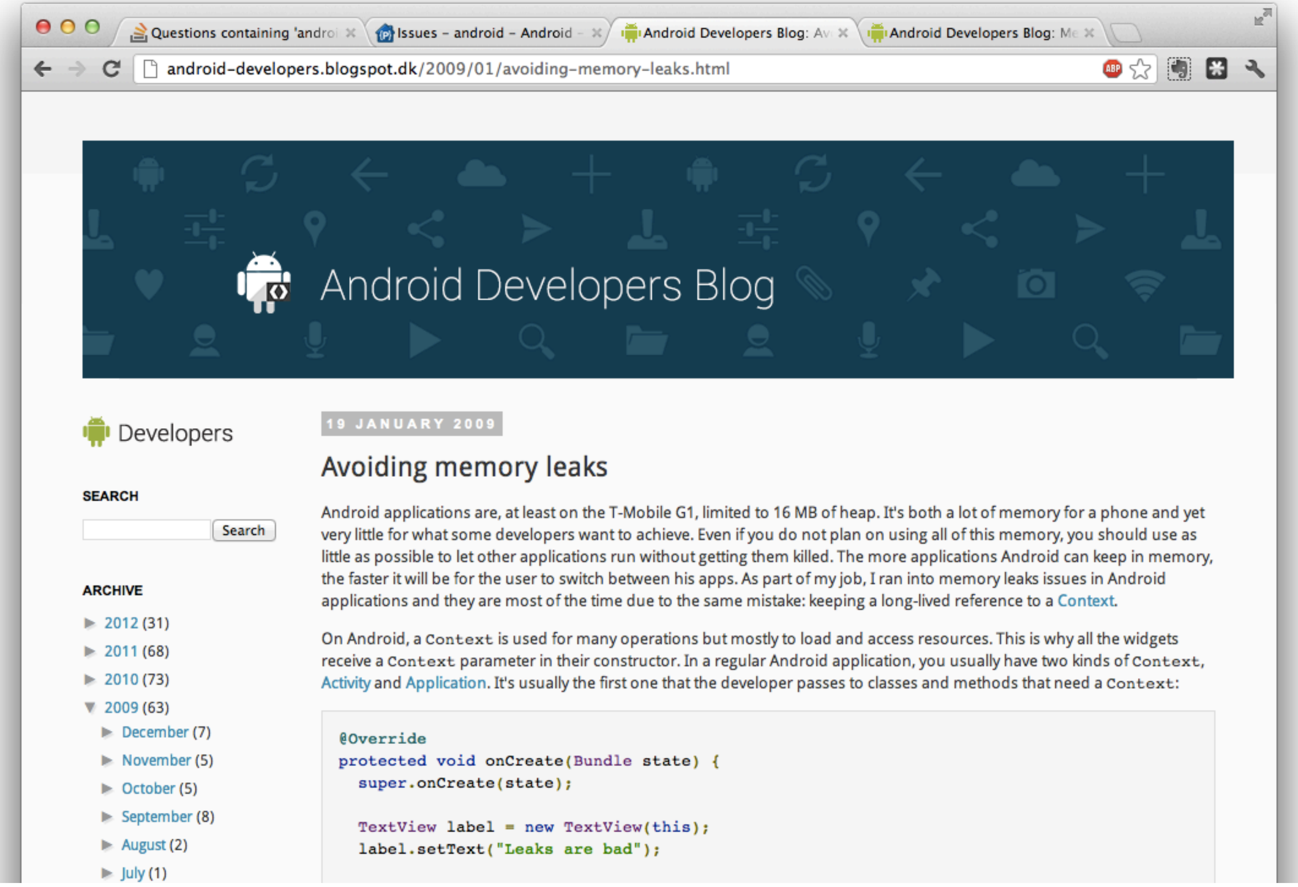
"an Activity leak"



Bug: Holding reference to "old" Activity

The expert recommendation ...

The expert recommendation ...


Questions containing 'andro
Issues - android - Android -
Android Developers Blog: Av
Android Developers Blog: Me
android-developers.blogspot.dk/2009/01/avoiding-memory-leaks.html
Android Developers Blog
Developers
SEARCH
Search
ARCHIVE
2012 (31)
2011 (68)
2010 (73)
2009 (63)
December (7)
November (5)
October (5)
September (8)
August (2)
July (1)

19 JANUARY 2009

Avoiding memory leaks

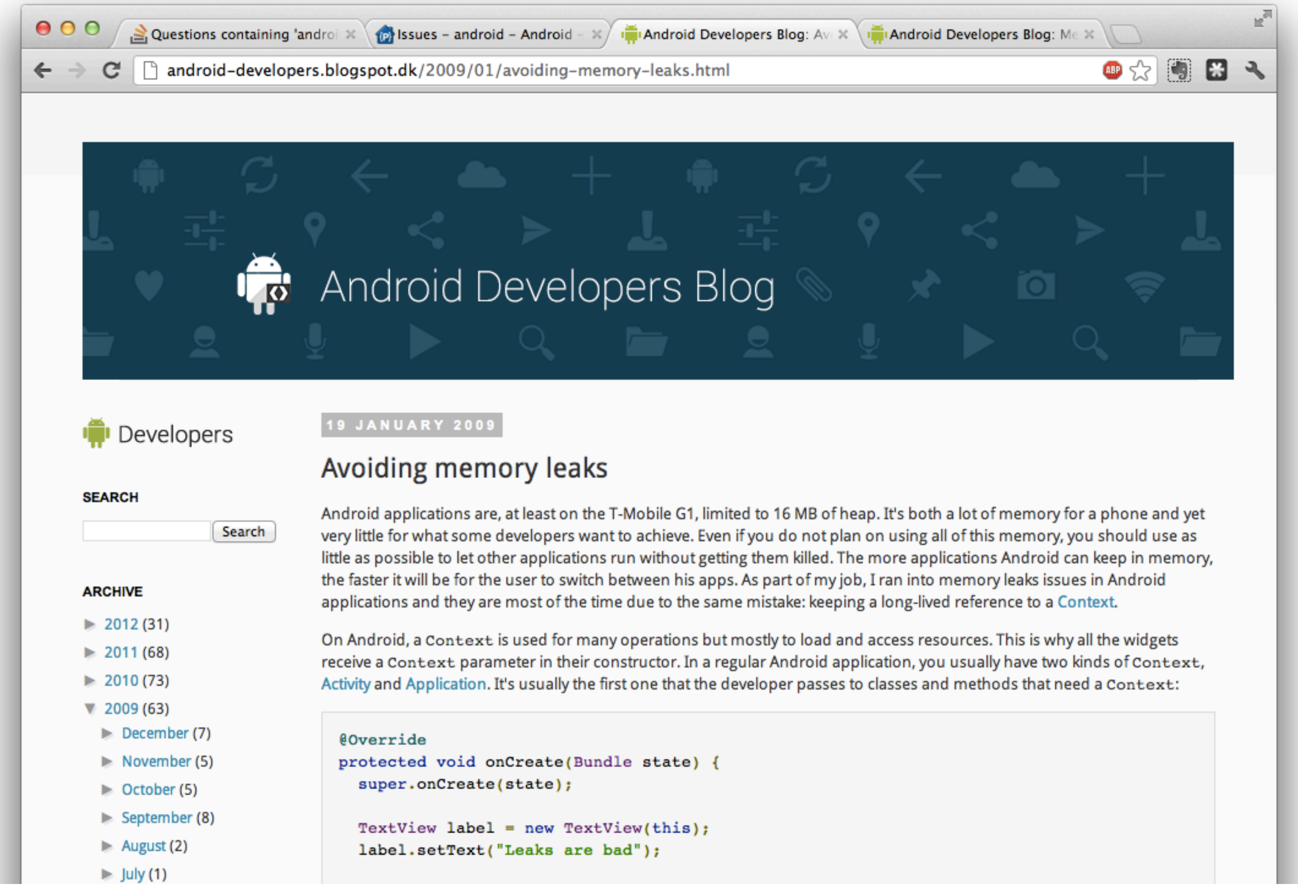
Android applications are, at least on the T-Mobile G1, limited to 16 MB of heap. It's both a lot of memory for a phone and yet very little for what some developers want to achieve. Even if you do not plan on using all of this memory, you should use as little as possible to let other applications run without getting them killed. The more applications Android can keep in memory, the faster it will be for the user to switch between his apps. As part of my job, I ran into memory leaks issues in Android applications and they are most of the time due to the same mistake: keeping a long-lived reference to a Context.

On Android, a `Context` is used for many operations but mostly to load and access resources. This is why all the widgets receive a `Context` parameter in their constructor. In a regular Android application, you usually have two kinds of `Context`, Activity and Application. It's usually the first one that the developer passes to classes and methods that need a `Context`:

```
@Override
protected void onCreate(Bundle state) {
  super.onCreate(state);

  TextView label = new TextView(this);
  label.setText("Leaks are bad");
```

The expert recommendation ...

"Do not keep long-lived references to a context-activity"


Questions containing 'andro
Issues – android – Android
Android Developers Blog: Av
Android Developers Blog: Me
android-developers.blogspot.dk/2009/01/avoiding-memory-leaks.html
Android Developers Blog
Developers
SEARCH
Search
ARCHIVE
2012 (31)
2011 (68)
2010 (73)
2009 (63)
December (7)
November (5)
October (5)
September (8)
August (2)
July (1)

19 JANUARY 2009

Avoiding memory leaks

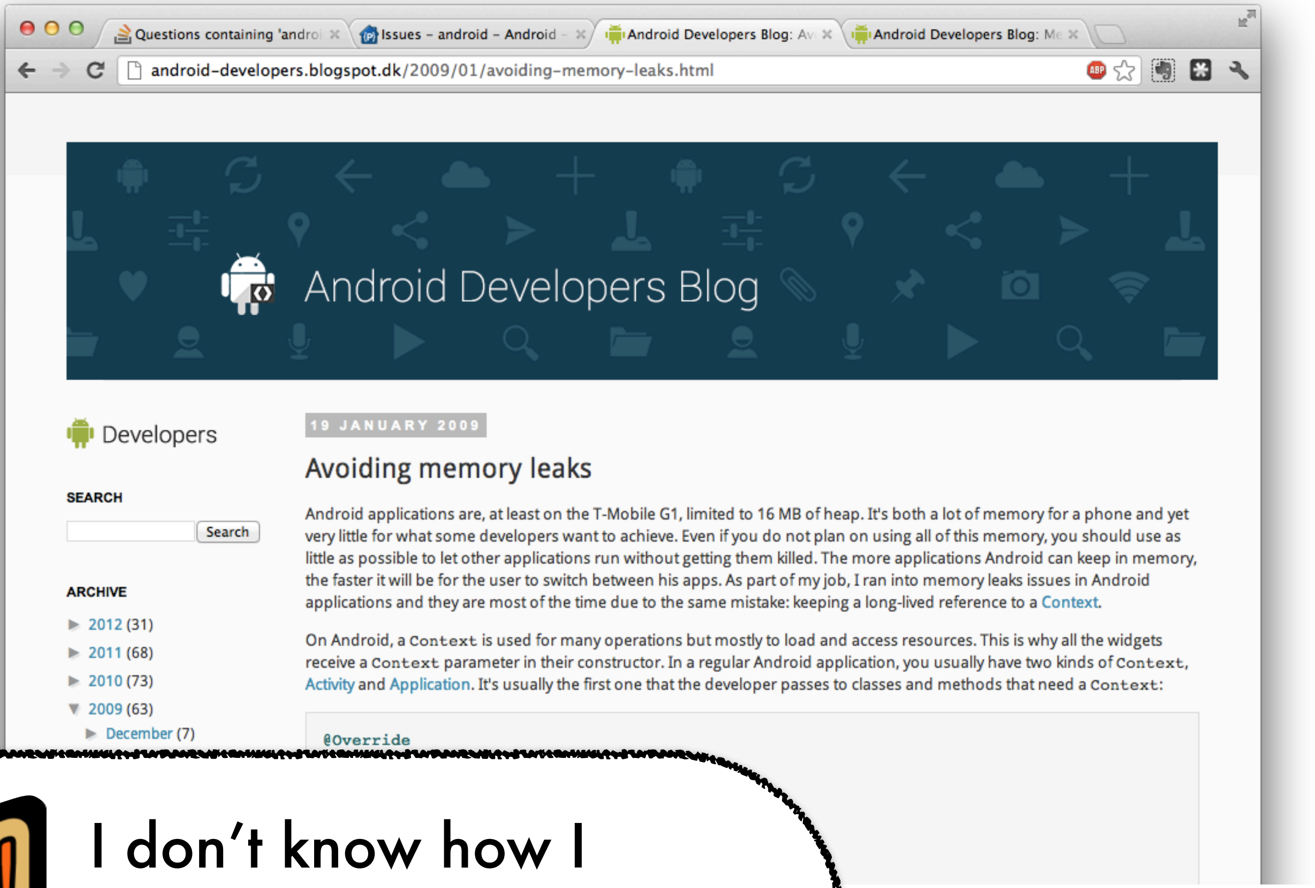
Android applications are, at least on the T-Mobile G1, limited to 16 MB of heap. It's both a lot of memory for a phone and yet very little for what some developers want to achieve. Even if you do not plan on using all of this memory, you should use as little as possible to let other applications run without getting them killed. The more applications Android can keep in memory, the faster it will be for the user to switch between his apps. As part of my job, I ran into memory leaks issues in Android applications and they are most of the time due to the same mistake: keeping a long-lived reference to a Context.

On Android, a `Context` is used for many operations but mostly to load and access resources. This is why all the widgets receive a `Context` parameter in their constructor. In a regular Android application, you usually have two kinds of `Context`, Activity and Application. It's usually the first one that the developer passes to classes and methods that need a `Context`:

```
@Override
protected void onCreate(Bundle state) {
  super.onCreate(state);

  TextView label = new TextView(this);
  label.setText("Leaks are bad");
```

The expert recommendation ...

"Do not keep long-lived references to a context-activity"



Questions containing 'andro
Issues - android - Android
Android Developers Blog: Av
Android Developers Blog: Me
android-developers.blogspot.dk/2009/01/avoiding-memory-leaks.html
Android Developers Blog
Developers
SEARCH
Search
ARCHIVE
► 2012 (31)
► 2011 (68)
► 2010 (73)
▼ 2009 (63)
► December (7)

19 JANUARY 2009

Avoiding memory leaks

Android applications are, at least on the T-Mobile G1, limited to 16 MB of heap. It's both a lot of memory for a phone and yet very little for what some developers want to achieve. Even if you do not plan on using all of this memory, you should use as little as possible to let other applications run without getting them killed. The more applications Android can keep in memory, the faster it will be for the user to switch between his apps. As part of my job, I ran into memory leaks issues in Android applications and they are most of the time due to the same mistake: keeping a long-lived reference to a Context.

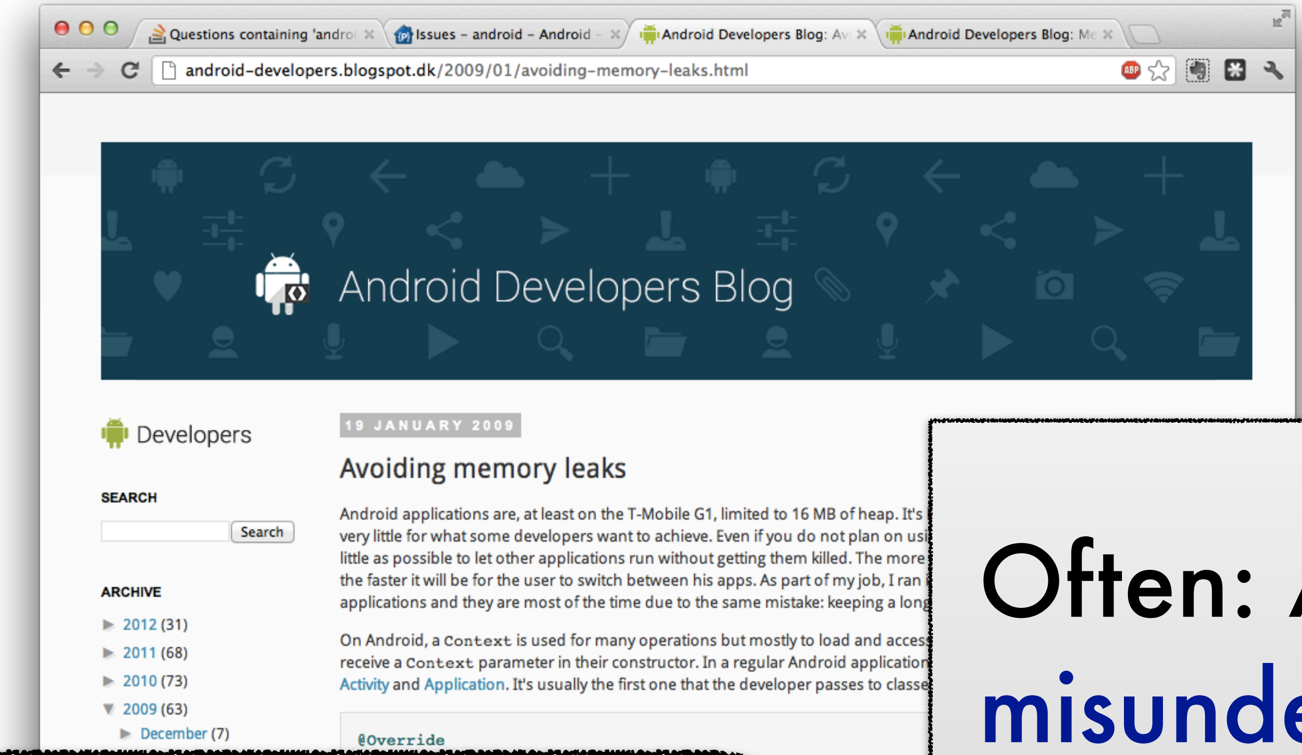
On Android, a `Context` is used for many operations but mostly to load and access resources. This is why all the widgets receive a `Context` parameter in their constructor. In a regular Android application, you usually have two kinds of `Context`, Activity and Application. It's usually the first one that the developer passes to classes and methods that need a `Context`:

```
@Override
```

I don't know how I created a long-lived reference to an `Activity`!

The expert recommendation …

"Do not keep long-lived references to a context-activity"



Questions containing 'andro
Issues - android - Android
Android Developers Blog: Av
Android Developers Blog: Me

android-developers.blogspot.dk/2009/01/avoiding-memory-leaks.html

Android Developers Blog

Developers

SEARCH

ARCHIVE

- 2012 (31)
- 2011 (68)
- 2010 (73)
- 2009 (63)
  - December (7)

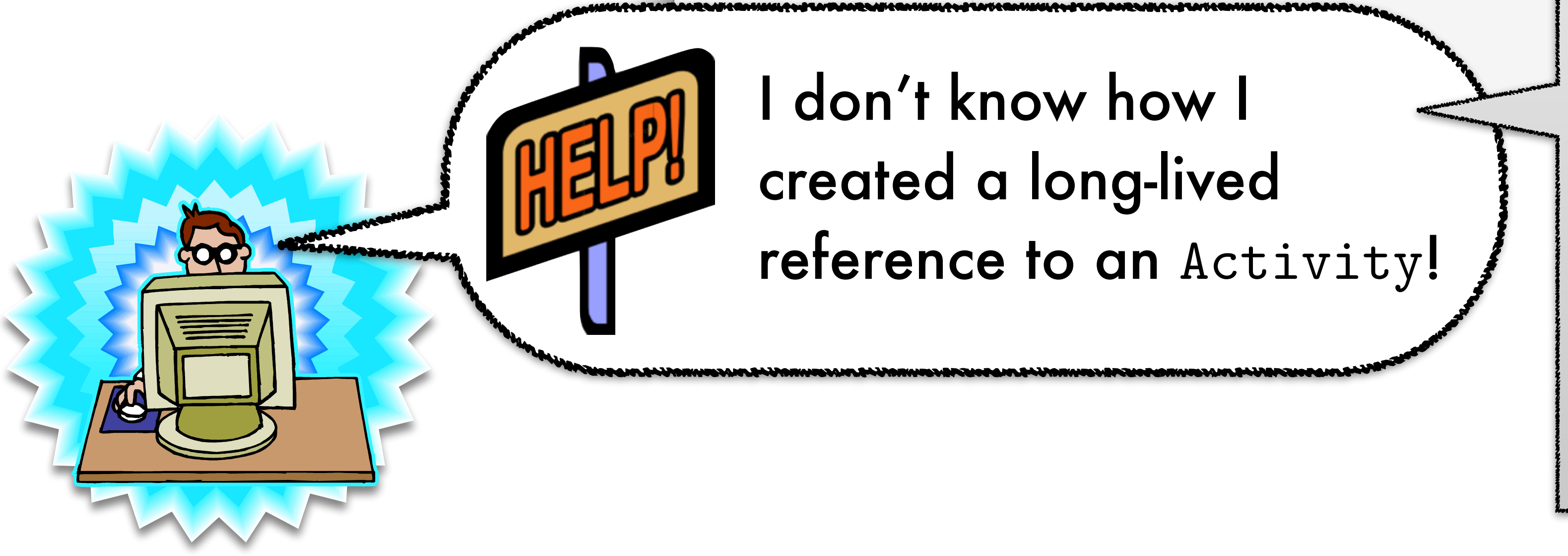
19 JANUARY 2009

Avoiding memory leaks

Android applications are, at least on the T-Mobile G1, limited to 16 MB of heap. It's very little for what some developers want to achieve. Even if you do not plan on usi little as possible to let other applications run without getting them killed. The more the faster it will be for the user to switch between his apps. As part of my job, I ran i applications and they are most of the time due to the same mistake: keeping a long

On Android, a `Context` is used for many operations but mostly to load and access receive a `Context` parameter in their constructor. In a regular Android application Activity and Application. It's usually the first one that the developer passes to classe

```
@Override
```



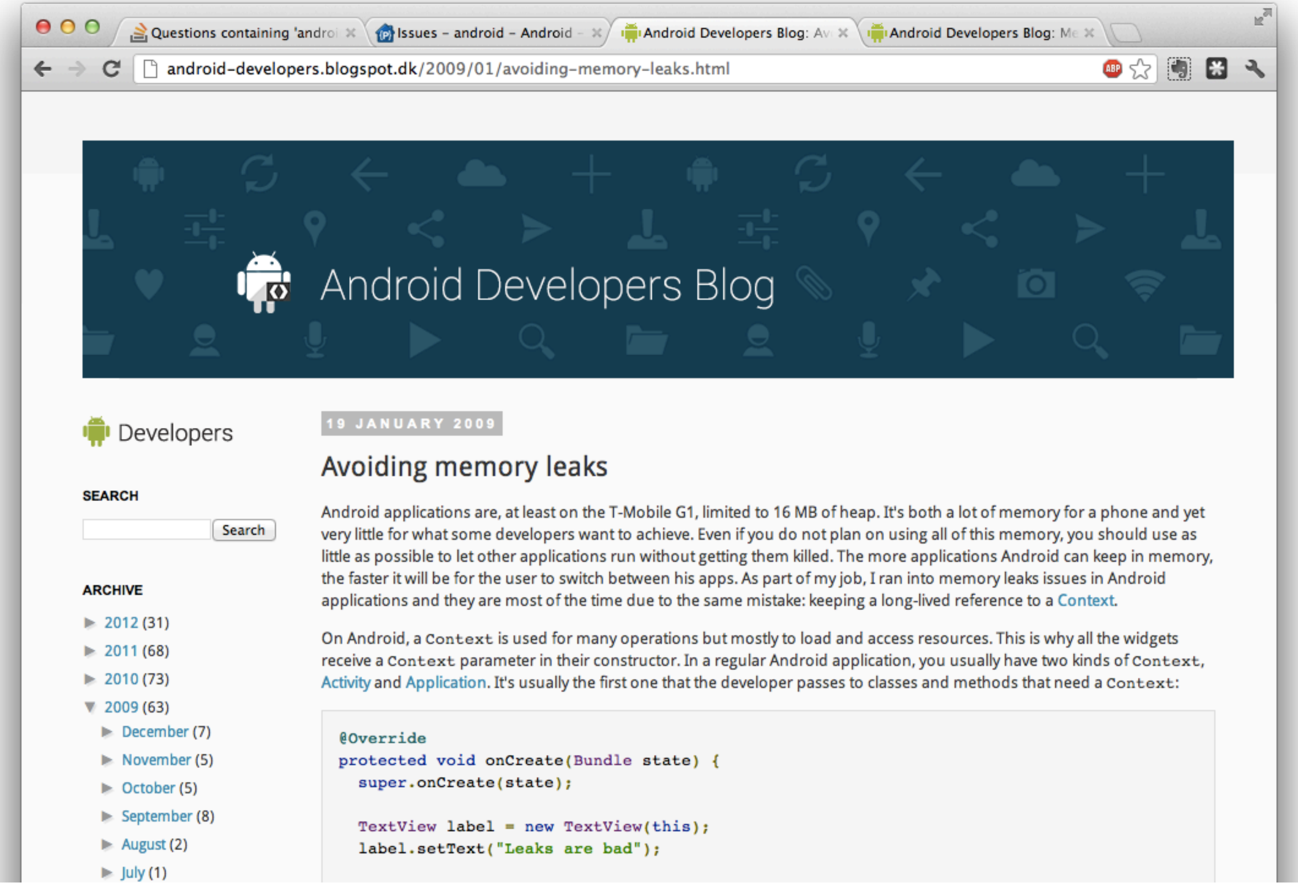
I don't know how I created a long-lived reference to an `Activity`!

Often: A **misunderstanding** of a library causes the **library** to keep the `Activity` reference.

VMCAI

The expert recommendation ...

The expert recommendation …


Questions containing 'andro
Issues - android - Android -
Android Developers Blog: Av
Android Developers Blog: Mc
android-developers.blogspot.dk/2009/01/avoiding-memory-leaks.html

Android Developers Blog

Developers

SEARCH

Search

ARCHIVE

- ► 2012 (31)
- ► 2011 (68)
- ► 2010 (73)
- ▼ 2009 (63)
  - ► December (7)
  - ► November (5)
  - ► October (5)
  - ► September (8)
  - ► August (2)
  - ► July (1)

19 JANUARY 2009

Avoiding memory leaks

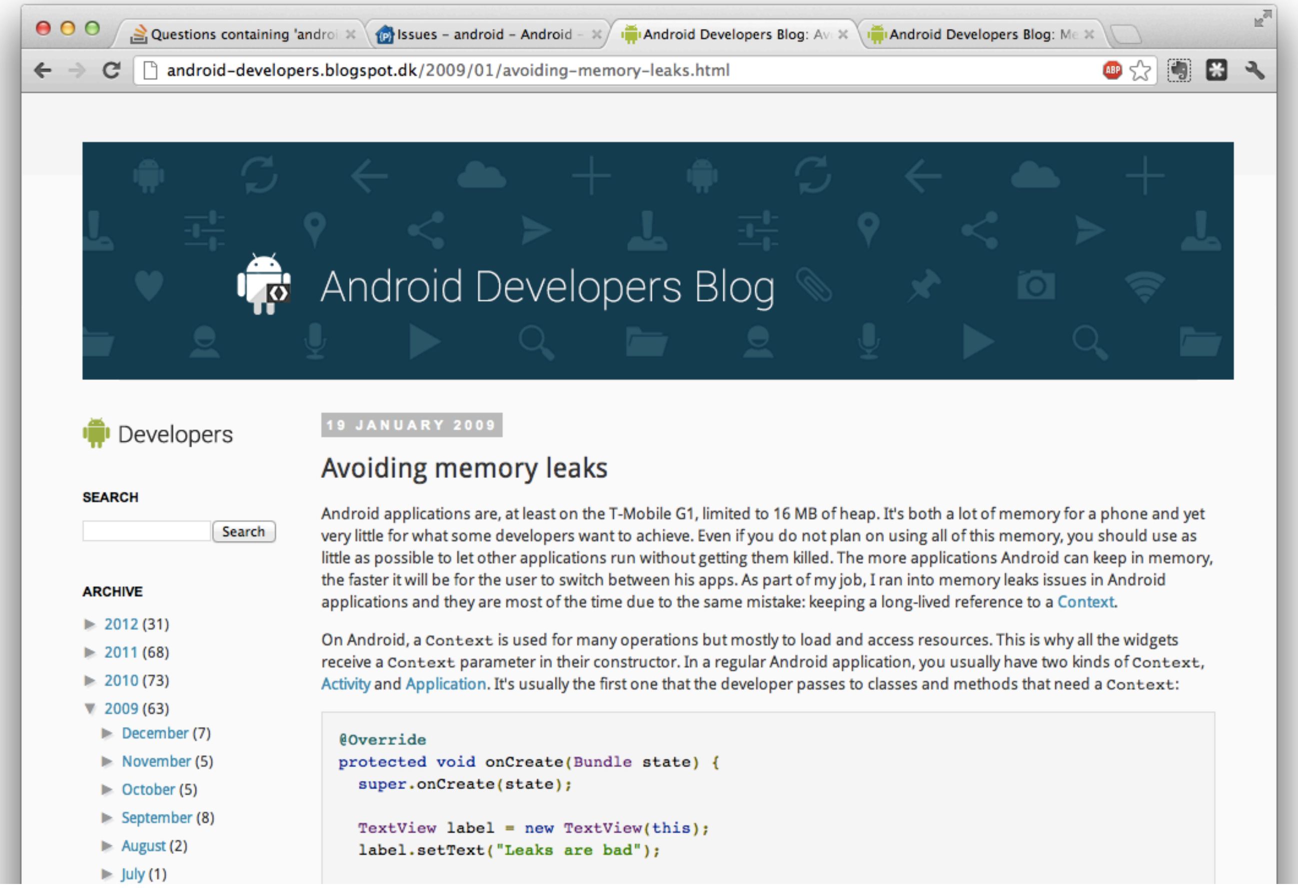
Android applications are, at least on the T-Mobile G1, limited to 16 MB of heap. It's both a lot of memory for a phone and yet very little for what some developers want to achieve. Even if you do not plan on using all of this memory, you should use as little as possible to let other applications run without getting them killed. The more applications Android can keep in memory, the faster it will be for the user to switch between his apps. As part of my job, I ran into memory leaks issues in Android applications and they are most of the time due to the same mistake: keeping a long-lived reference to a Context.

On Android, a `Context` is used for many operations but mostly to load and access resources. This is why all the widgets receive a `Context` parameter in their constructor. In a regular Android application, you usually have two kinds of `Context`, Activity and Application. It's usually the first one that the developer passes to classes and methods that need a `Context`:

```
@Override
protected void onCreate(Bundle state) {
  super.onCreate(state);

  TextView label = new TextView(this);
  label.setText("Leaks are bad");
```

The expert recommendation ...

"Do not keep long-lived references to a context-activity"


Questions containing 'andro
Issues – android – Android –
Android Developers Blog: Av
Android Developers Blog: Me
android-developers.blogspot.dk/2009/01/avoiding-memory-leaks.html
Developers
SEARCH
Search

Android Developers Blog

ARCHIVE

- ▶ 2012 (31)
- ▶ 2011 (68)
- ▶ 2010 (73)
- ▼ 2009 (63)
  - ▶ December (7)
  - ▶ November (5)
  - ▶ October (5)
  - ▶ September (8)
  - ▶ August (2)
  - ▶ July (1)

19 JANUARY 2009

Avoiding memory leaks

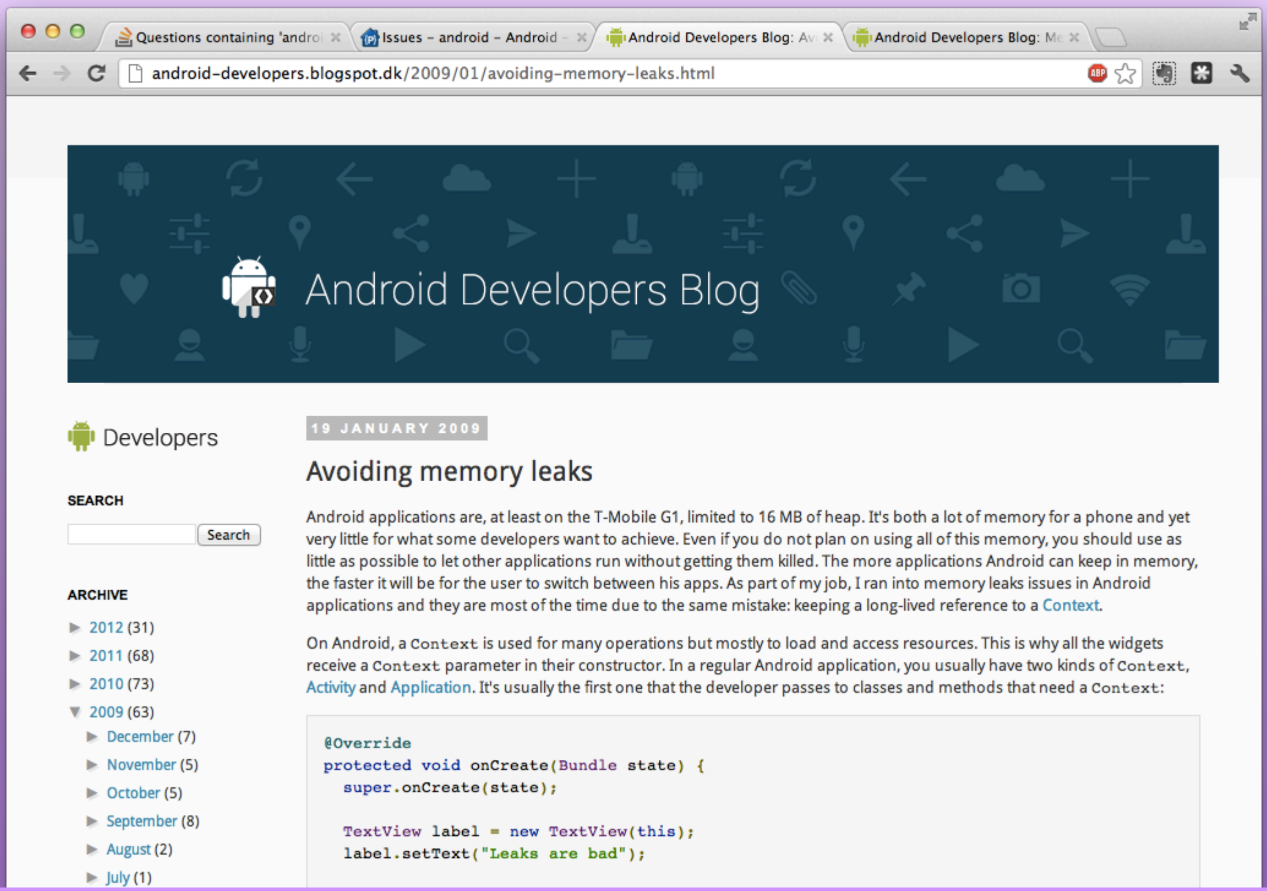
Android applications are, at least on the T-Mobile G1, limited to 16 MB of heap. It's both a lot of memory for a phone and yet very little for what some developers want to achieve. Even if you do not plan on using all of this memory, you should use as little as possible to let other applications run without getting them killed. The more applications Android can keep in memory, the faster it will be for the user to switch between his apps. As part of my job, I ran into memory leaks issues in Android applications and they are most of the time due to the same mistake: keeping a long-lived reference to a Context.

On Android, a `Context` is used for many operations but mostly to load and access resources. This is why all the widgets receive a `Context` parameter in their constructor. In a regular Android application, you usually have two kinds of `Context`, Activity and Application. It's usually the first one that the developer passes to classes and methods that need a `Context`:

```
@Override
protected void onCreate(Bundle state) {
  super.onCreate(state);

  TextView label = new TextView(this);
  label.setText("Leaks are bad");
```

The expert recommendation ...

“Do not keep long-lived references to a context-activity”



Questions containing 'androi
Issues - android - Android -
Android Developers Blog: Av
Android Developers Blog: Me
android-developers.blogspot.dk/2009/01/avoiding-memory-leaks.html
Android Developers Blog
Developers
SEARCH
Search
ARCHIVE
2012 (31)
2011 (68)
2010 (73)
2009 (63)
December (7)
November (5)
October (5)
September (8)
August (2)
July (1)

19 JANUARY 2009

Avoiding memory leaks

Android applications are, at least on the T-Mobile G1, limited to 16 MB of heap. It's both a lot of memory for a phone and yet very little for what some developers want to achieve. Even if you do not plan on using all of this memory, you should use as little as possible to let other applications run without getting them killed. The more applications Android can keep in memory, the faster it will be for the user to switch between his apps. As part of my job, I ran into memory leaks issues in Android applications and they are most of the time due to the same mistake: keeping a long-lived reference to a Context.

On Android, a `Context` is used for many operations but mostly to load and access resources. This is why all the widgets receive a `Context` parameter in their constructor. In a regular Android application, you usually have two kinds of `Context`, Activity and Application. It's usually the first one that the developer passes to classes and methods that need a `Context`:

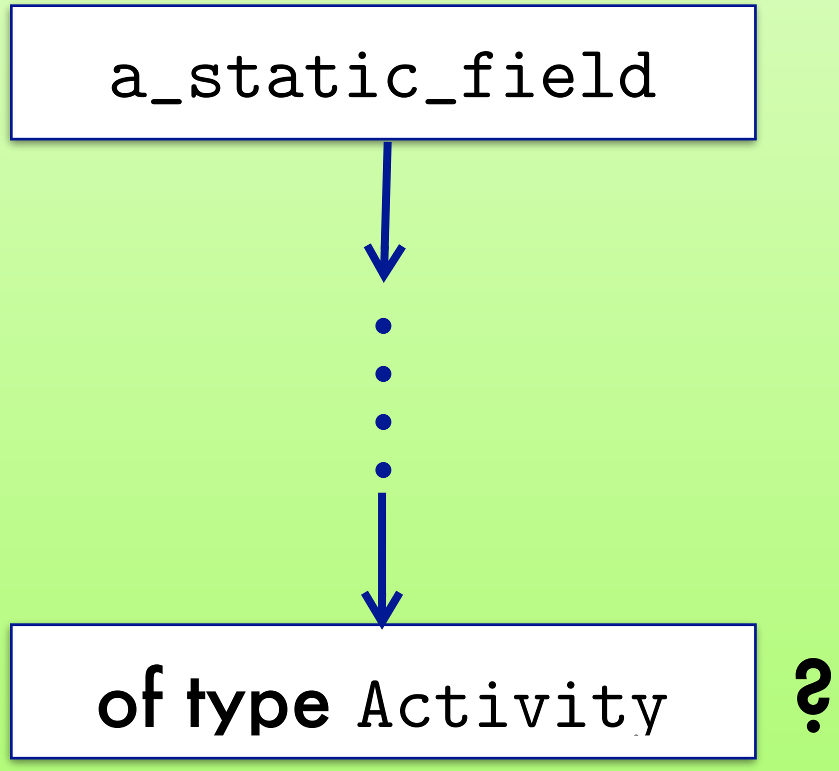
```
@Override
protected void onCreate(Bundle state) {
  super.onCreate(state);

  TextView label = new TextView(this);
  label.setText("Leaks are bad");
```

A Specific Property to Check:

No `Activity` is ever reachable from a static field.

A candidate for statically answering, “Is there an `Activity` leak?”

Is there a program execution where at some time



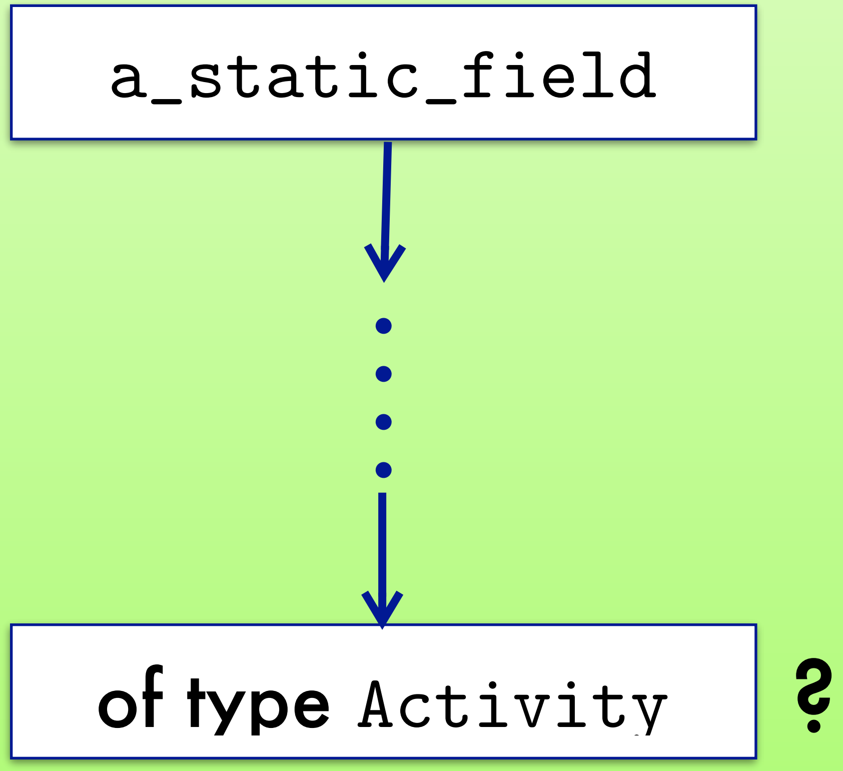
`a_static_field`

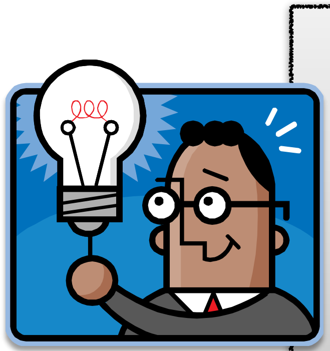
of type `Activity`

?

A candidate for statically answering, “Is there an `Activity` leak?”

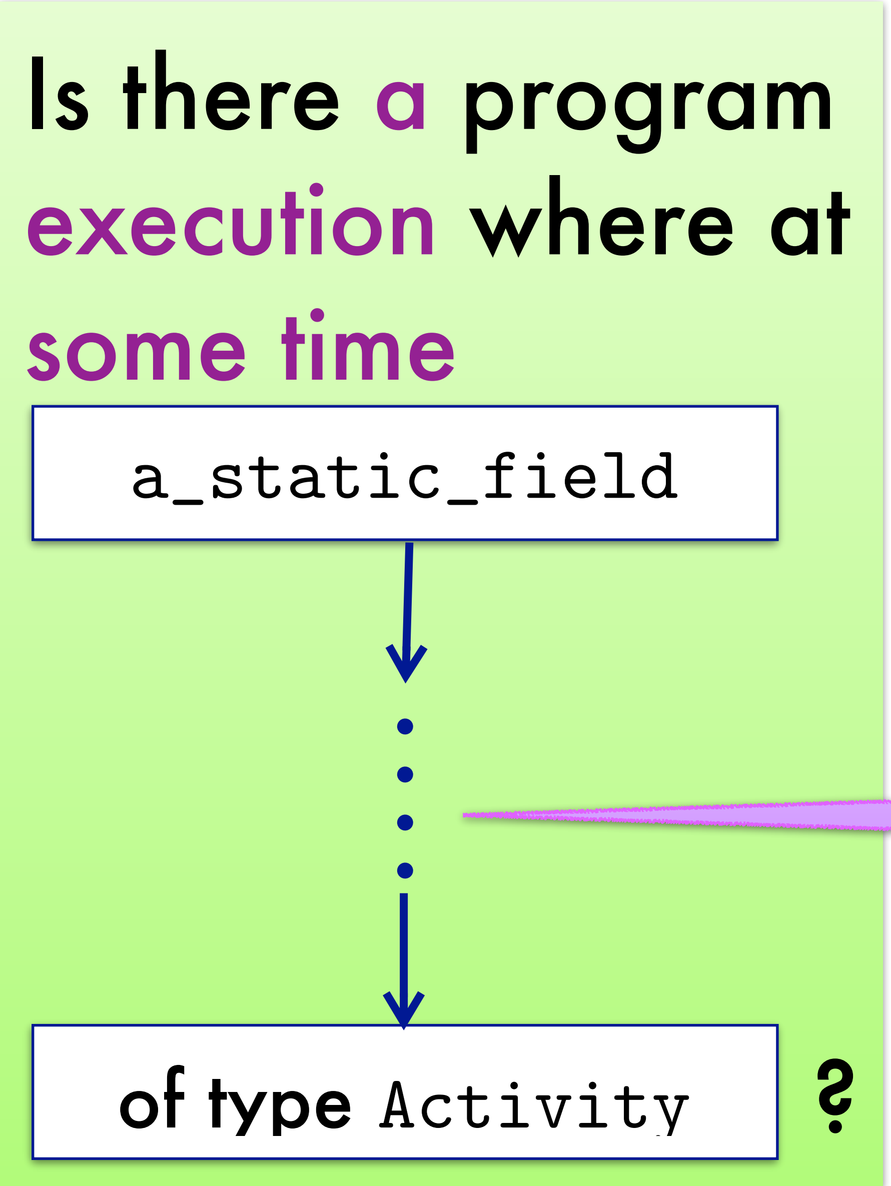
Is there a program execution where at some time





Can be answered with a **points-to analysis**

A candidate for statically answering, “Is there an `Activity` leak?”

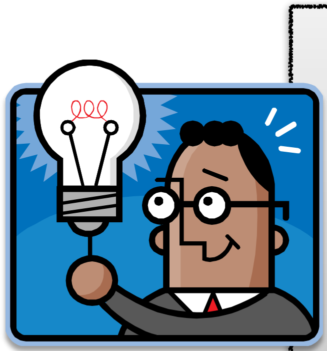


Is there a program execution where at some time

`a_static_field`

⋮

of type `Activity` ?

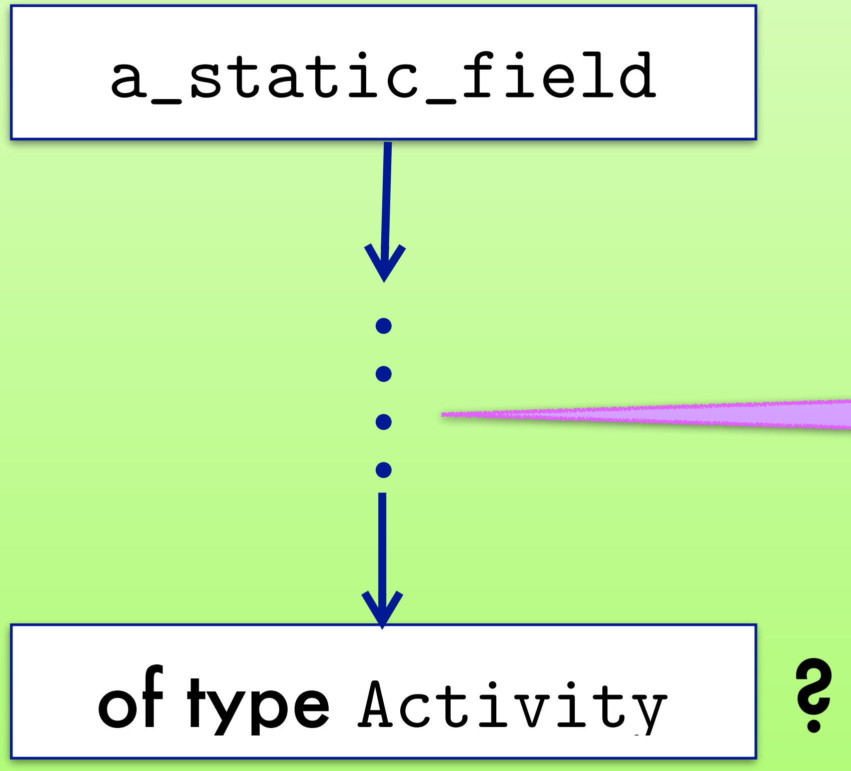


Can be answered with a points-to analysis

Compute a points-to graph and look for such points-to paths

A candidate for statically answering, “Is there an Activity leak?”

Is there a program execution where at some time

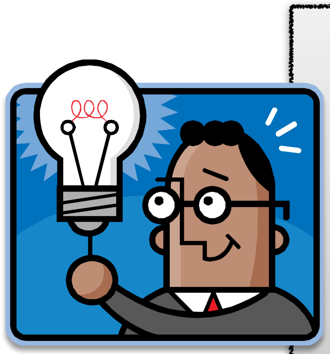


a_static_field

⋮

of type Activity

?



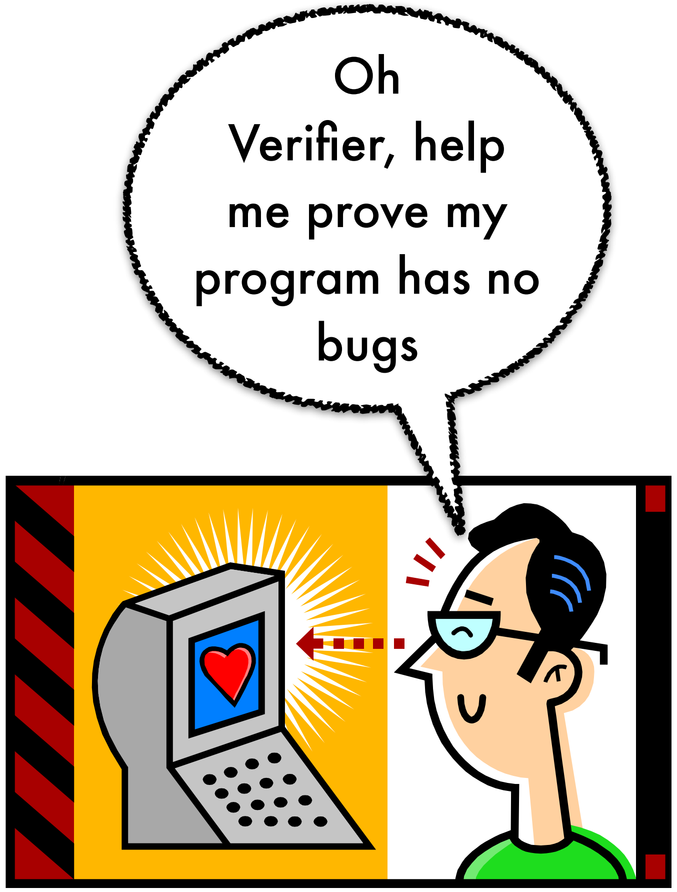
Can be answered with a points-to analysis

Compute a points-to graph and look for such points-to paths

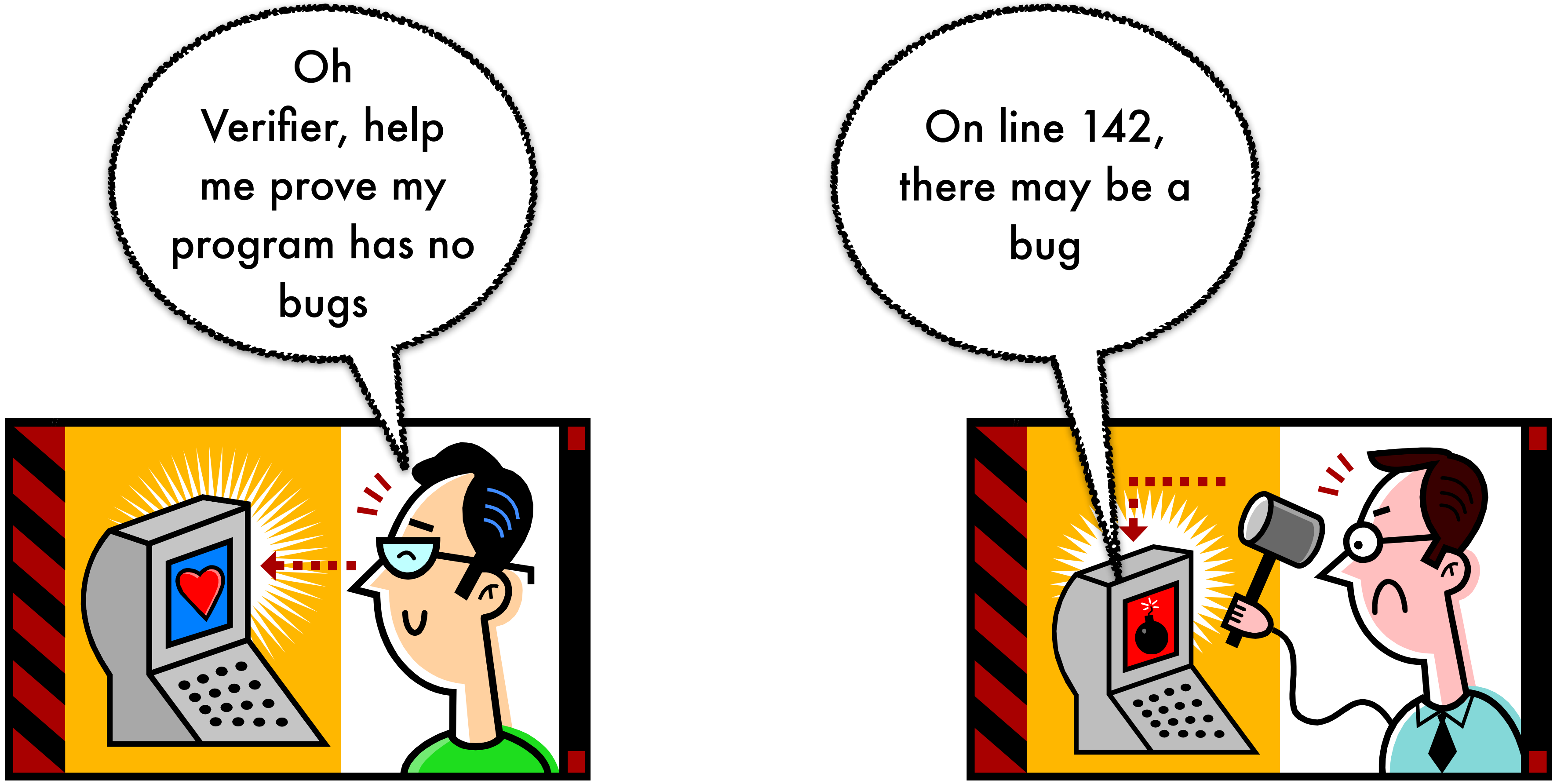
This won’t work because …

The well-known false alarm problem!

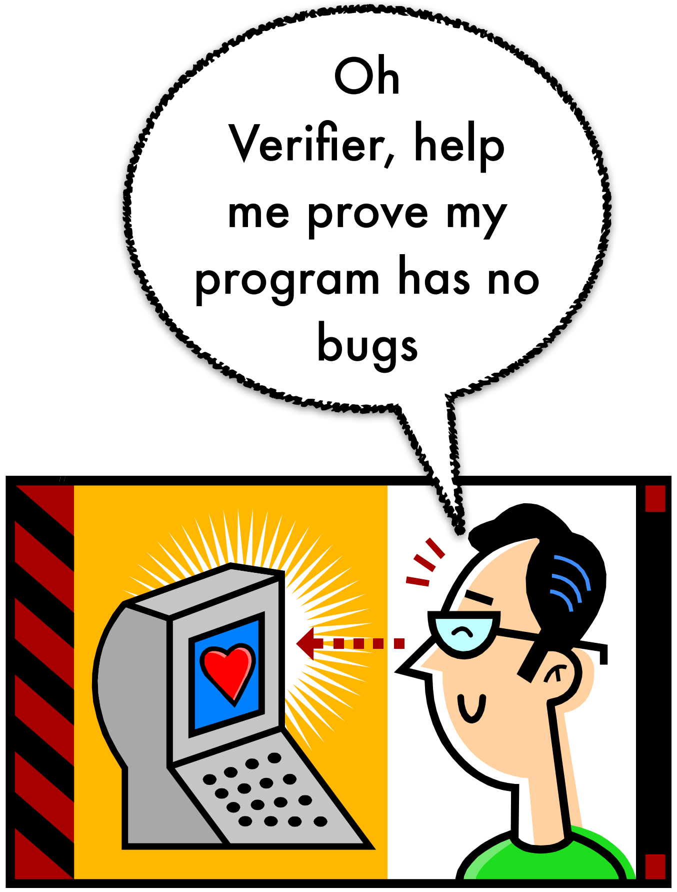
The well-known false alarm problem!

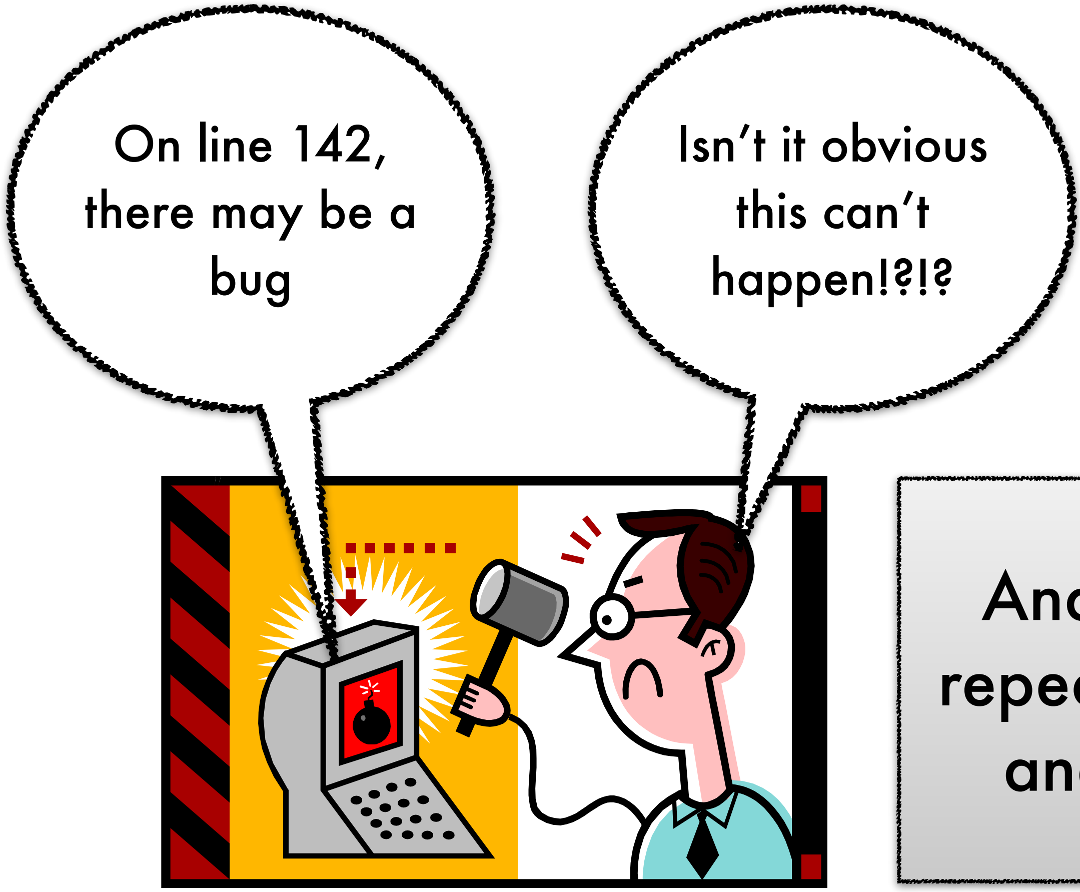


The well-known false alarm problem!


Oh Verifier, help me prove my program has no bugs
On line 142, there may be a bug

The well-known false alarm problem!



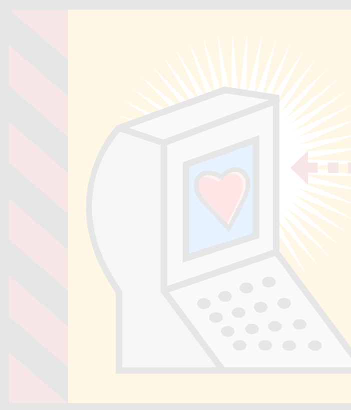

On line 142, there may be a bug
Isn’t it obvious this can’t happen!?!?

And noisily repeated over and over!

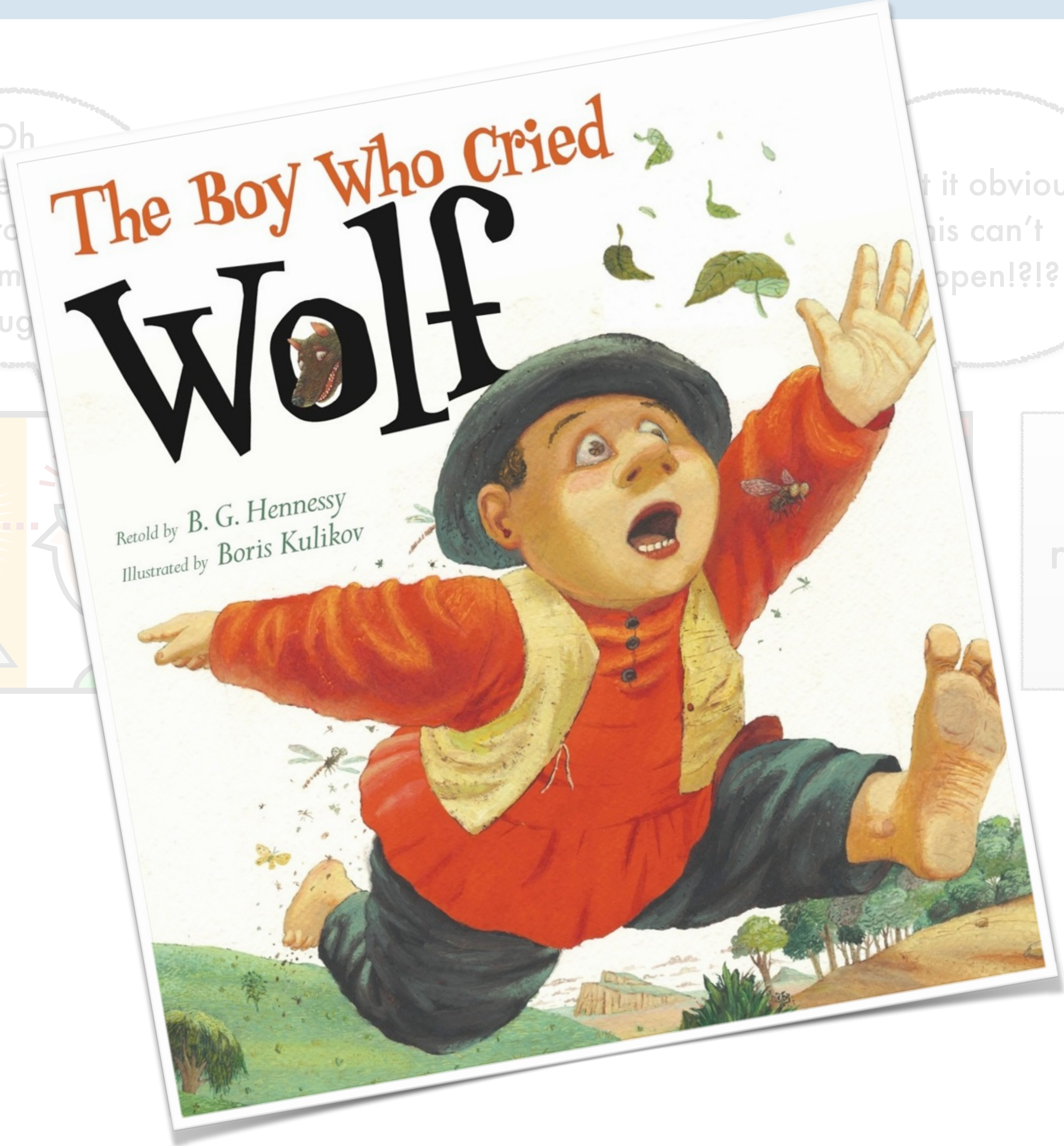
The well-known false alarm problem!

Oh
Verifie
me pro
program
bug

t it obvious
is can't
ppen!?!?



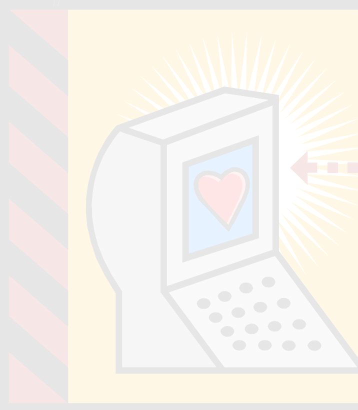
And noisily repeated over and over!

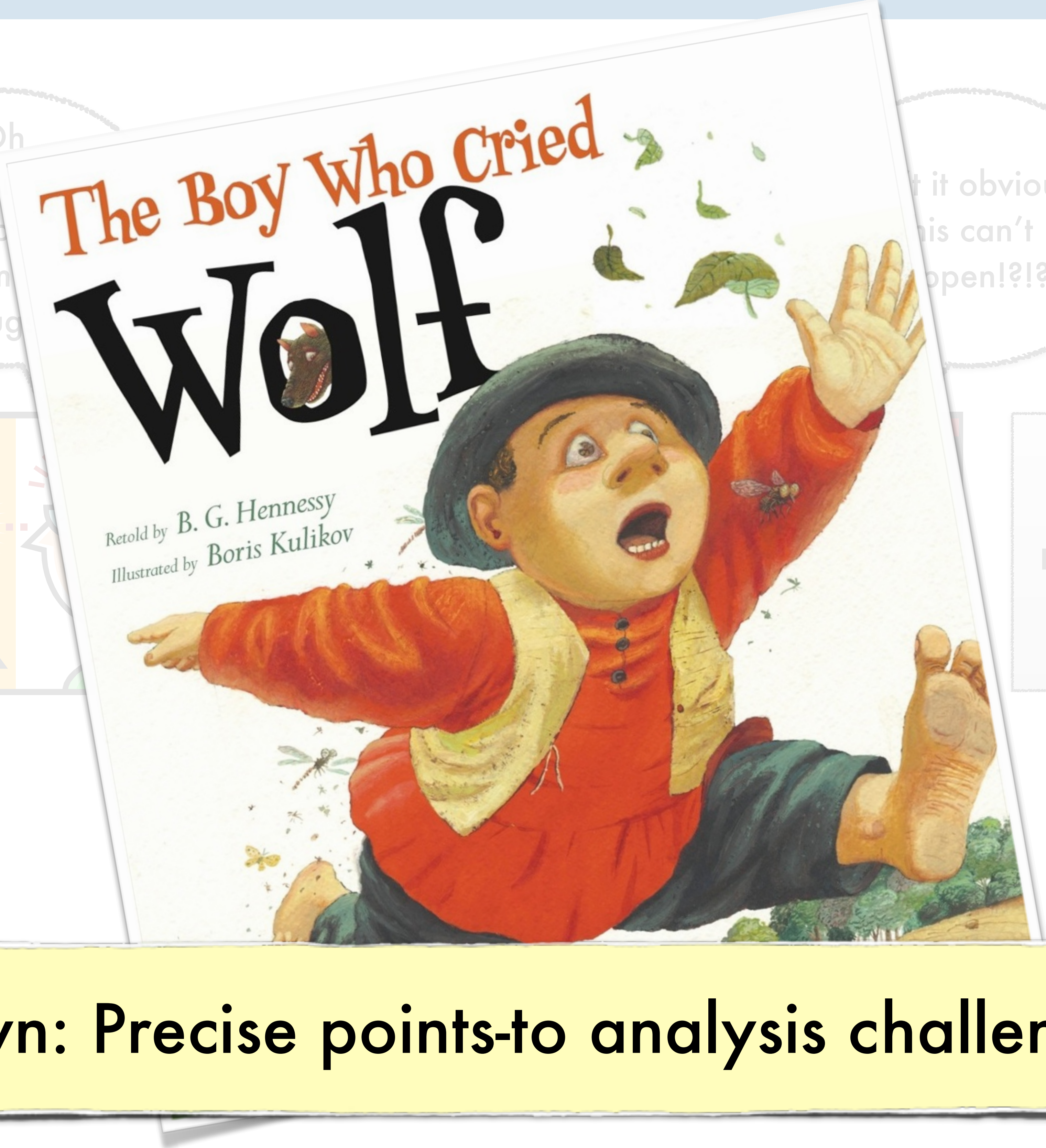


The well-known false alarm problem!

Oh
Verifie
me pro
program
bug

t it obvious
is can't
ppen!?!?





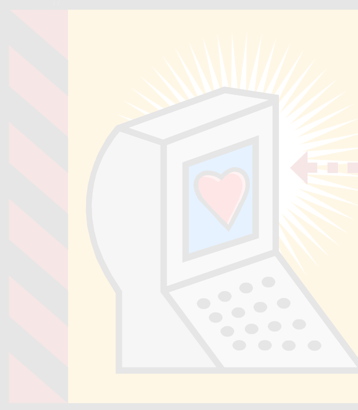
And noisily repeated over and over!

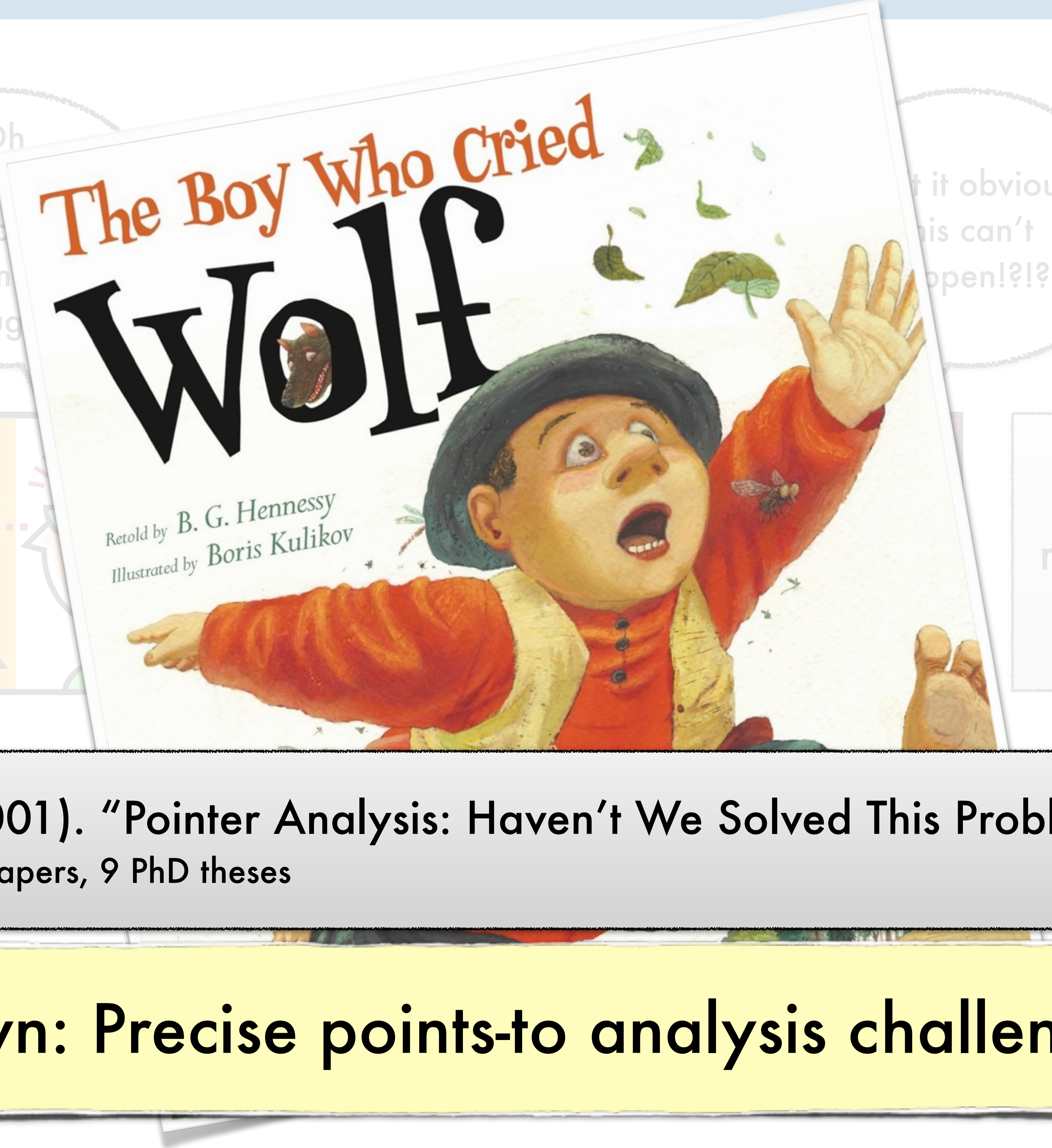
Known: Precise points-to analysis challenging

The well-known false alarm problem!

Oh
Verifie
me pro
program
bug

t it obvious
is can't
ppen!?!?





And noisily repeated over and over!

Hind (2001). "Pointer Analysis: Haven't We Solved This Problem Yet?"

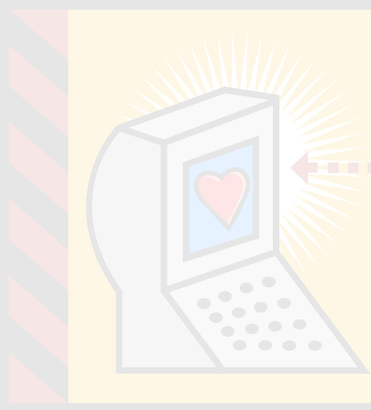
- 75 papers, 9 PhD theses

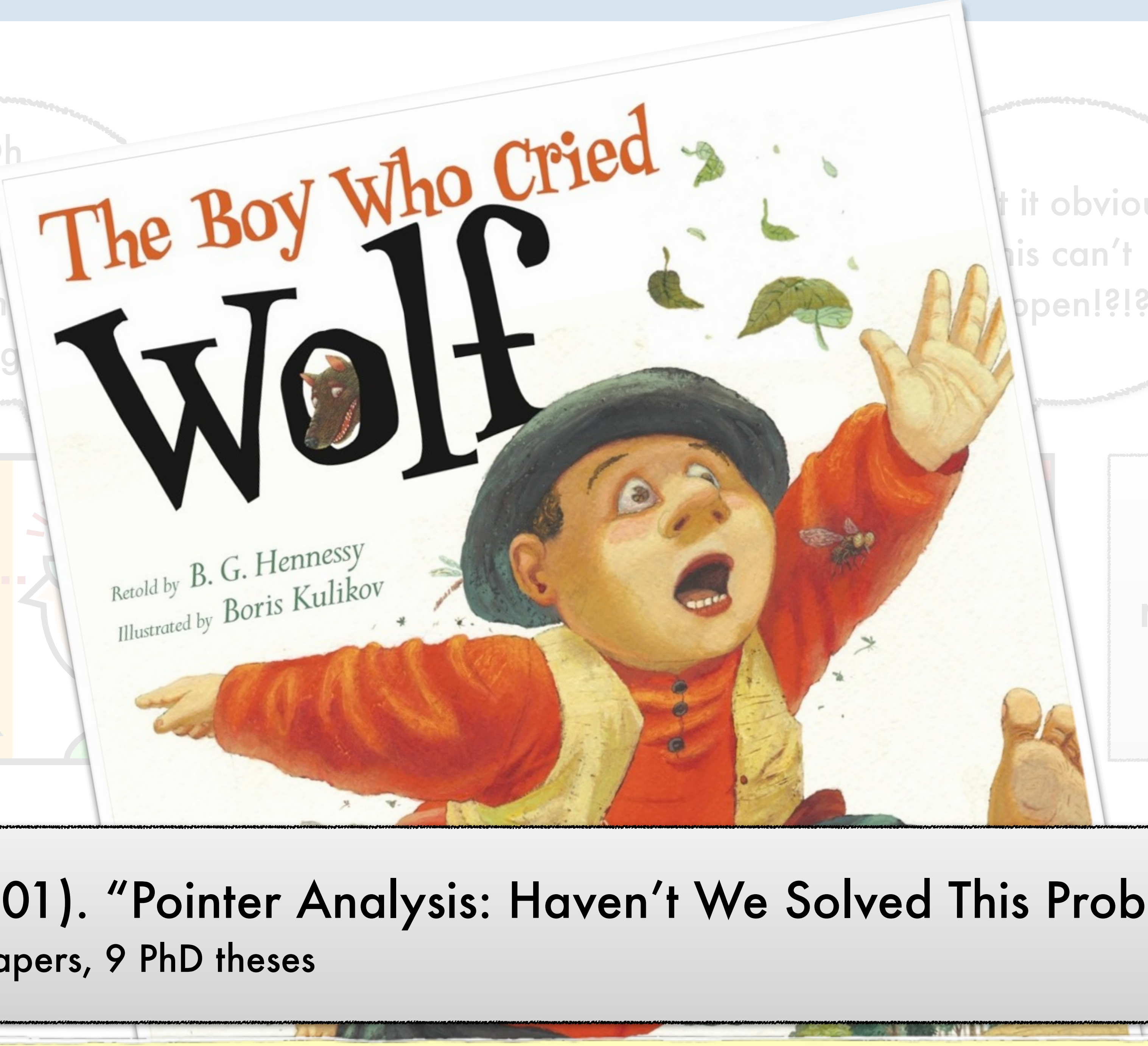
Known: Precise points-to analysis challenging

The well-known false alarm problem!

Oh
Verifie
me pro
program
bug

t it obvious
is can't
ppen!?!?





And noisily repeated over and over!

Hind (2001). “Pointer Analysis: Haven't We Solved This Problem Yet?”

- 75 papers, 9 PhD theses

Known: Precise ^enough points-to analysis ~~challenging~~ impossible?

VMCAI

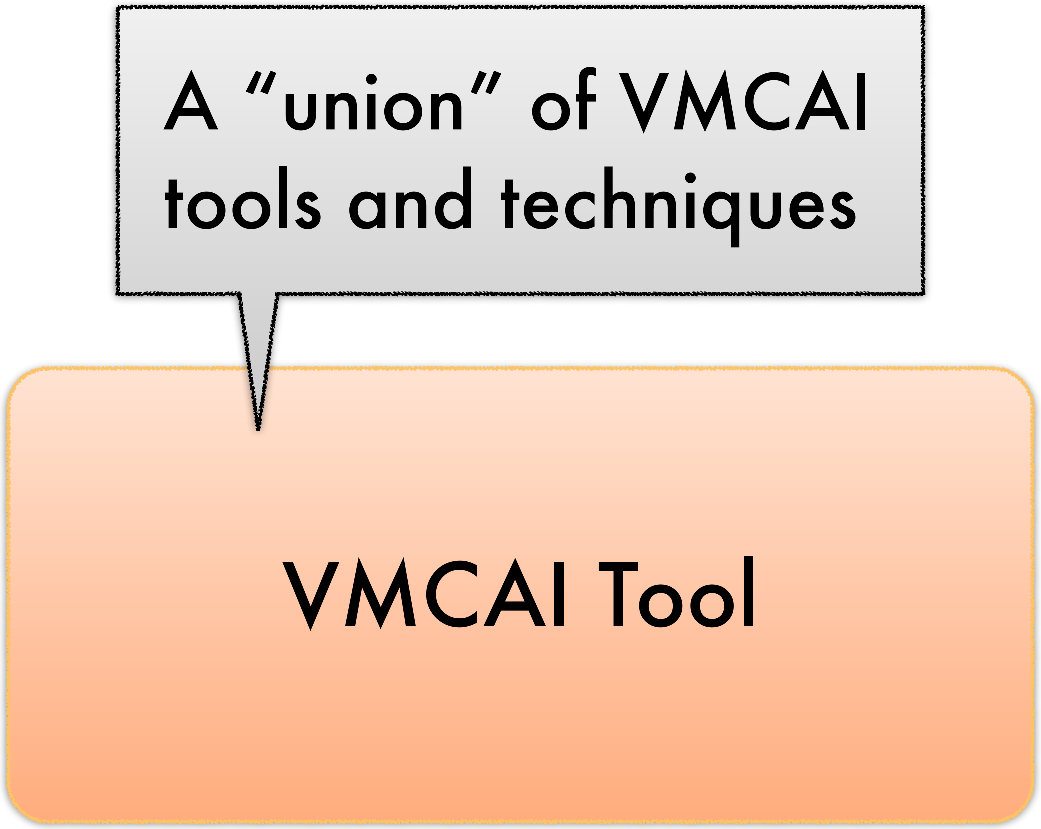
Next: A perspective on VMCAI and false alarms

Perspective: The false alarm problem and VMCAI

Perspective: The false alarm problem and VMCAI

VMCAI Tool

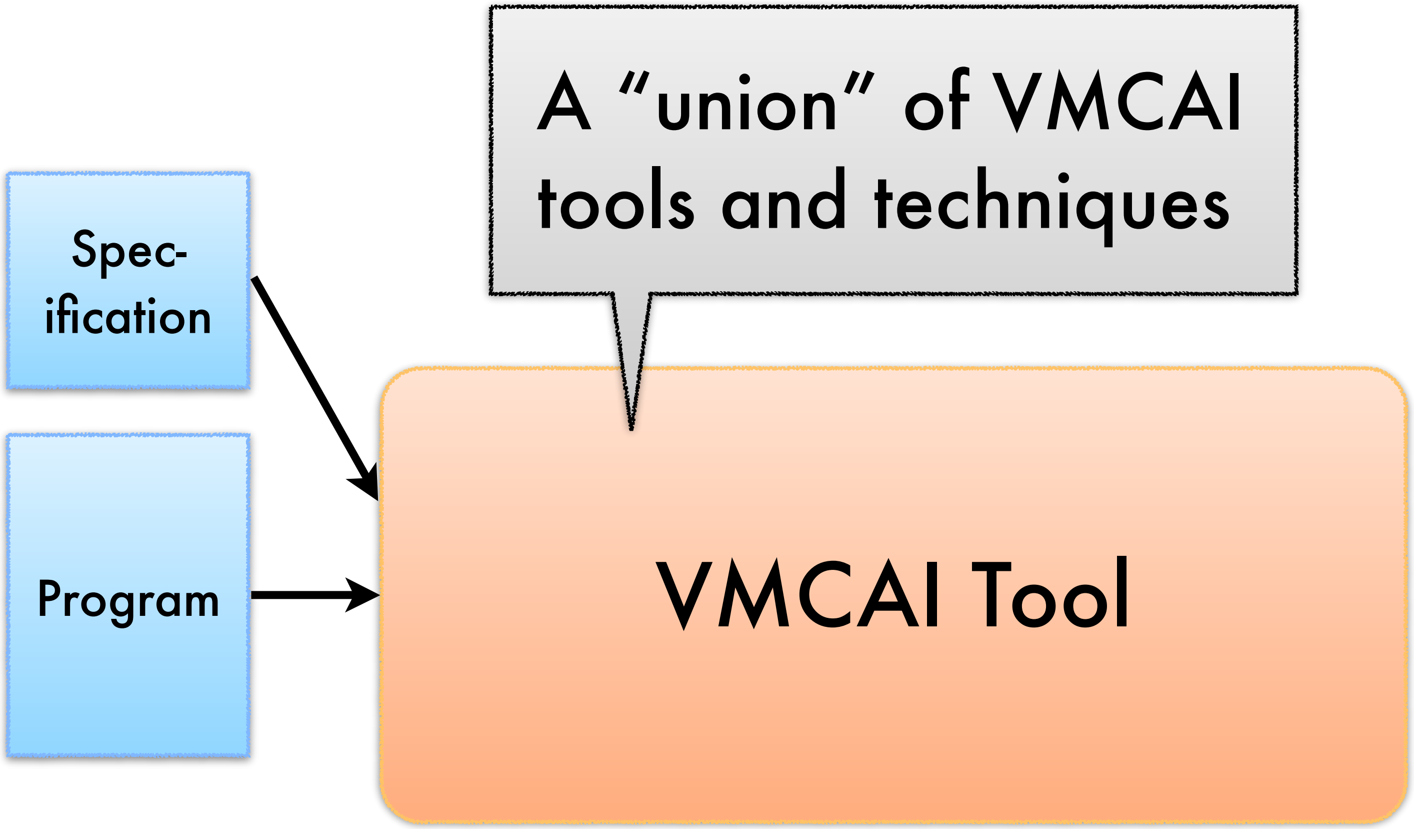
Perspective: The false alarm problem and VMCAI



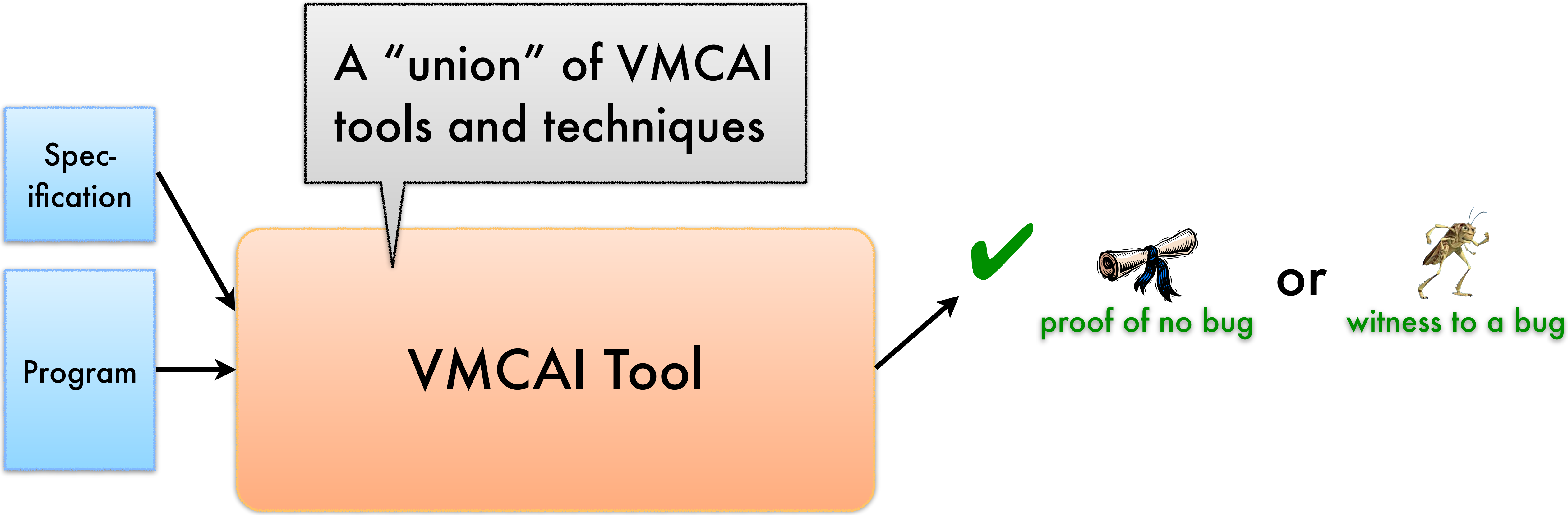
A “union” of VMCAI tools and techniques

VMCAI Tool

Perspective: The false alarm problem and VMCAI



Perspective: The false alarm problem and VMCAI



A “union” of VMCAI tools and techniques

Spec-ification

Program

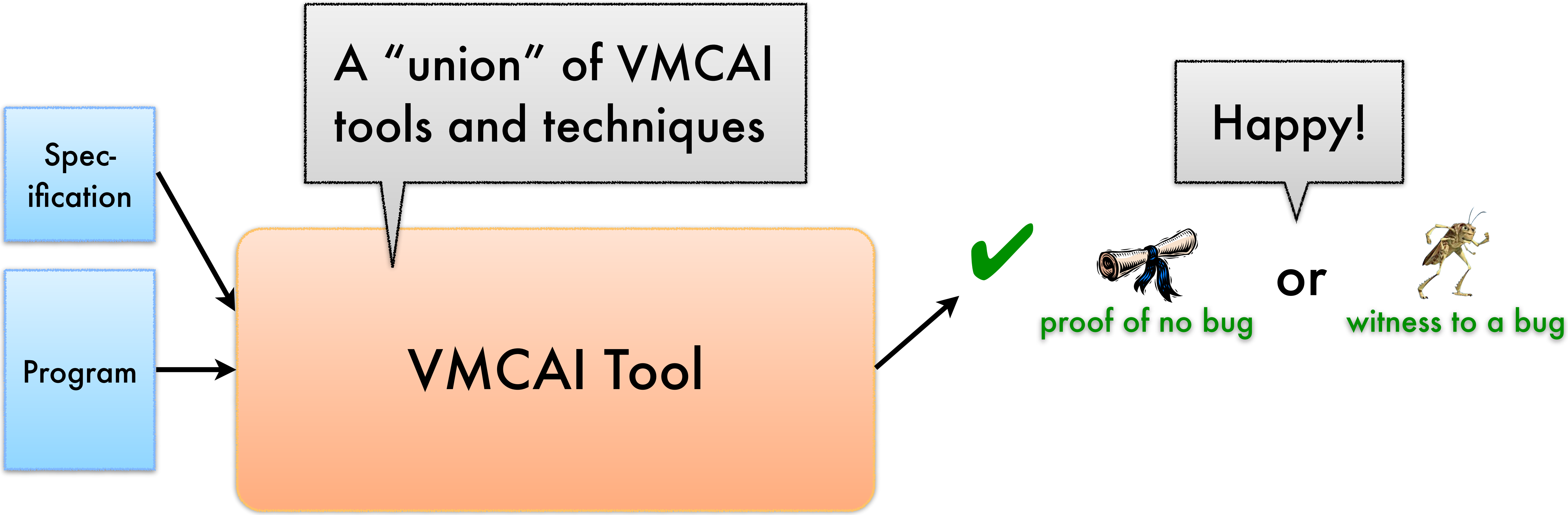
VMCAI Tool

✔ proof of no bug

or

witness to a bug

Perspective: The false alarm problem and VMCAI



Spec-ification

Program

A "union" of VMCAI tools and techniques

VMCAI Tool

✓

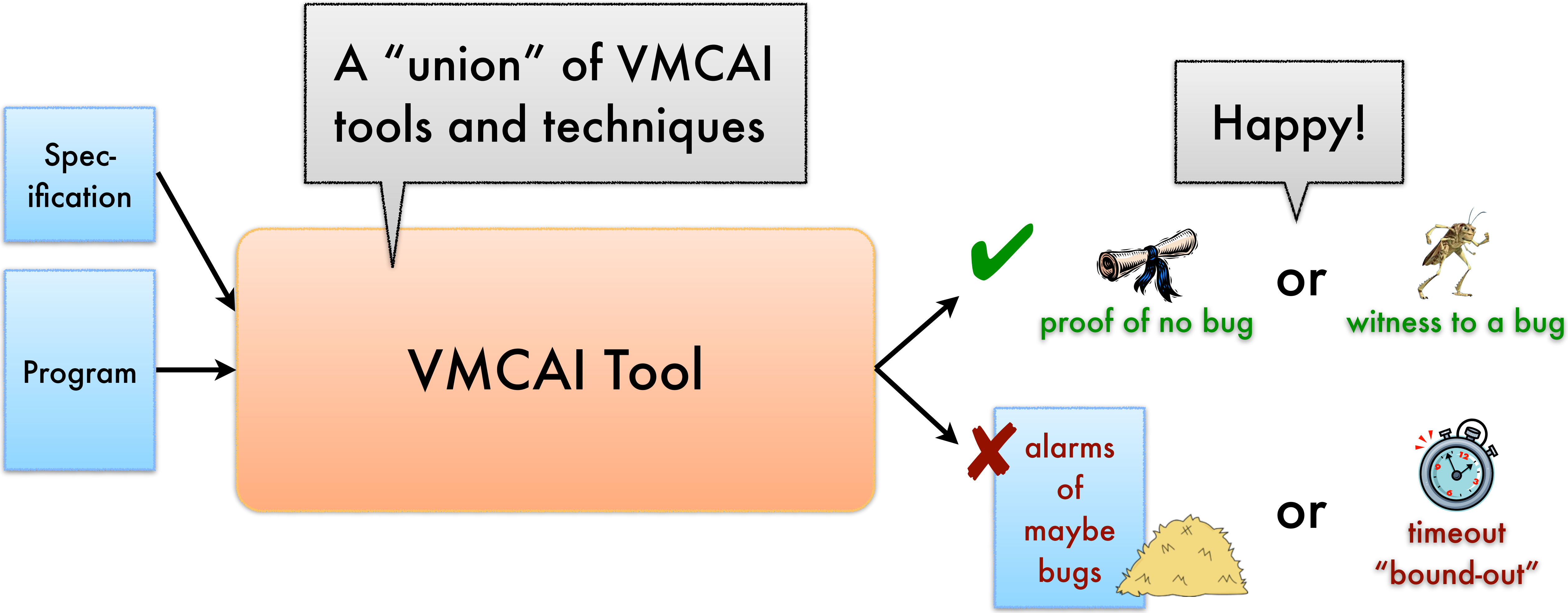
proof of no bug

or

witness to a bug

Happy!

Perspective: The false alarm problem and VMCAI



A "union" of VMCAI tools and techniques

Specification

Program

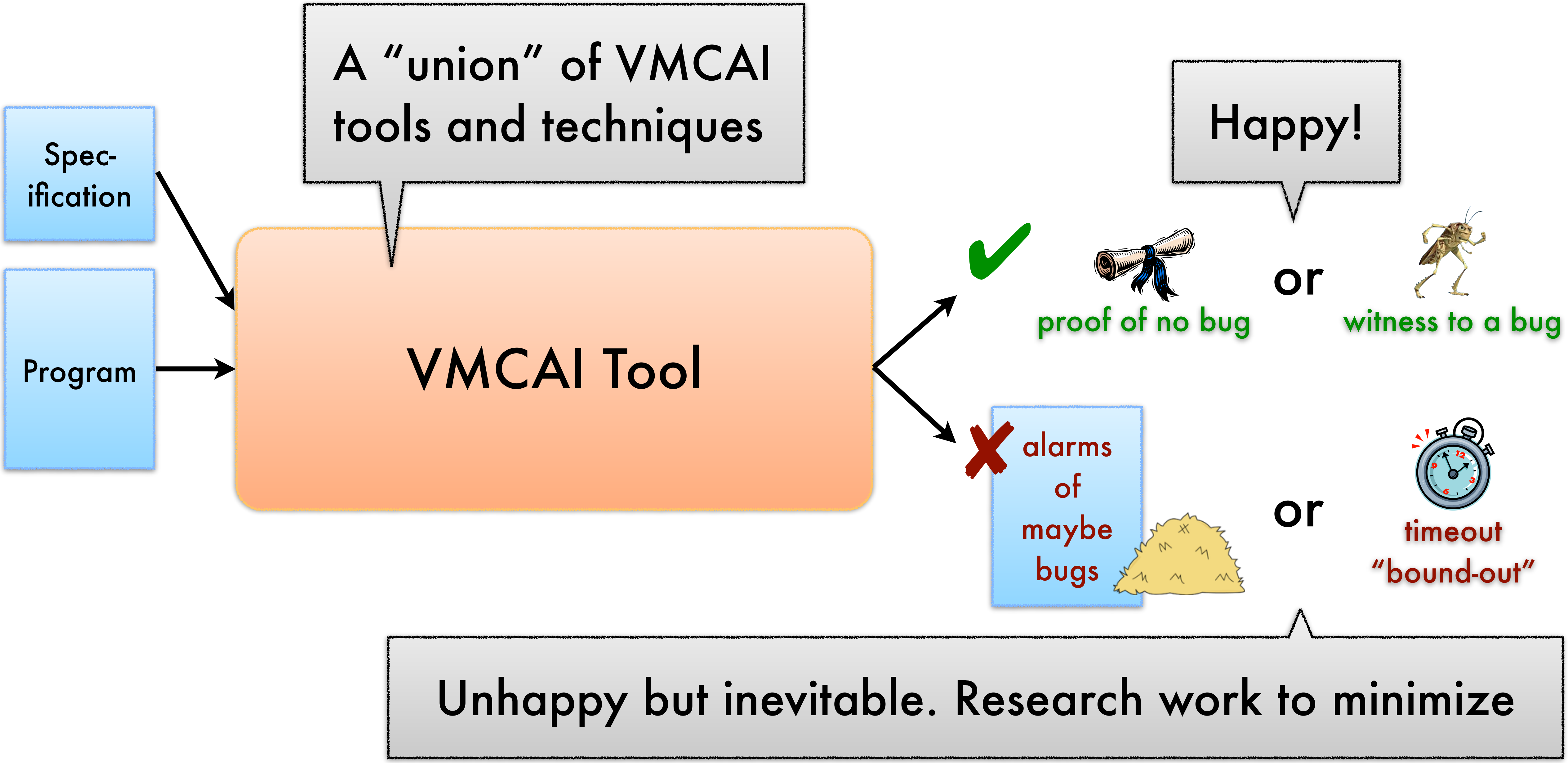
VMCAI Tool

Happy!

✓ proof of no bug or witness to a bug

✗ alarms of maybe bugs or timeout "bound-out"

Perspective: The false alarm problem and VMCAI



Spec-ification

Program

A "union" of VMCAI tools and techniques

VMCAI Tool

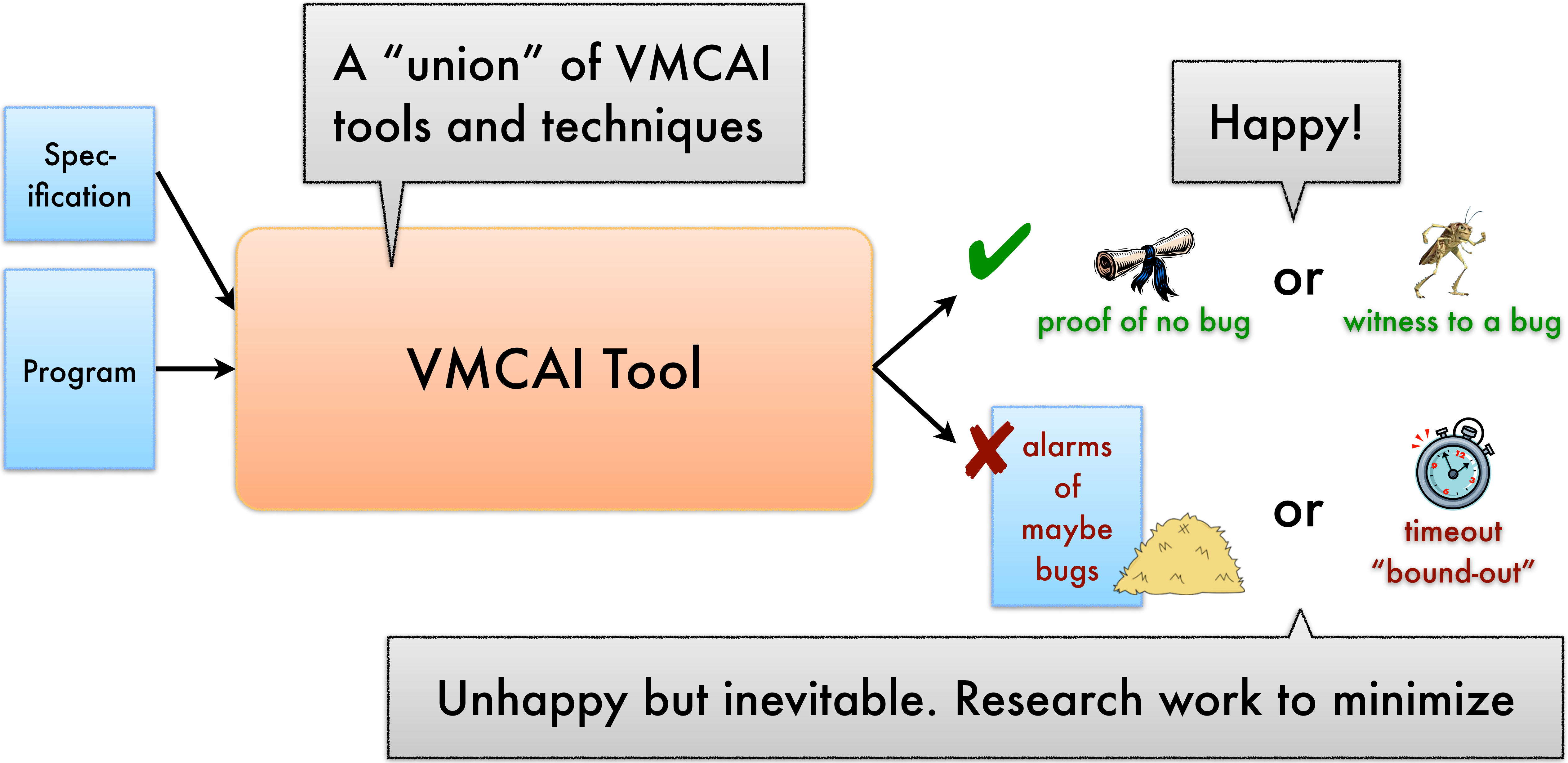
Happy!

✔ proof of no bug or witness to a bug

✘ alarms of maybe bugs or timeout "bound-out"

Unhappy but inevitable. Research work to minimize

Perspective: The false alarm problem and VMCAI



A "union" of VMCAI tools and techniques

Spec-ification

Program

VMCAI Tool

✔

proof of no bug

or

witness to a bug

Happy!

✘

alarms of maybe bugs

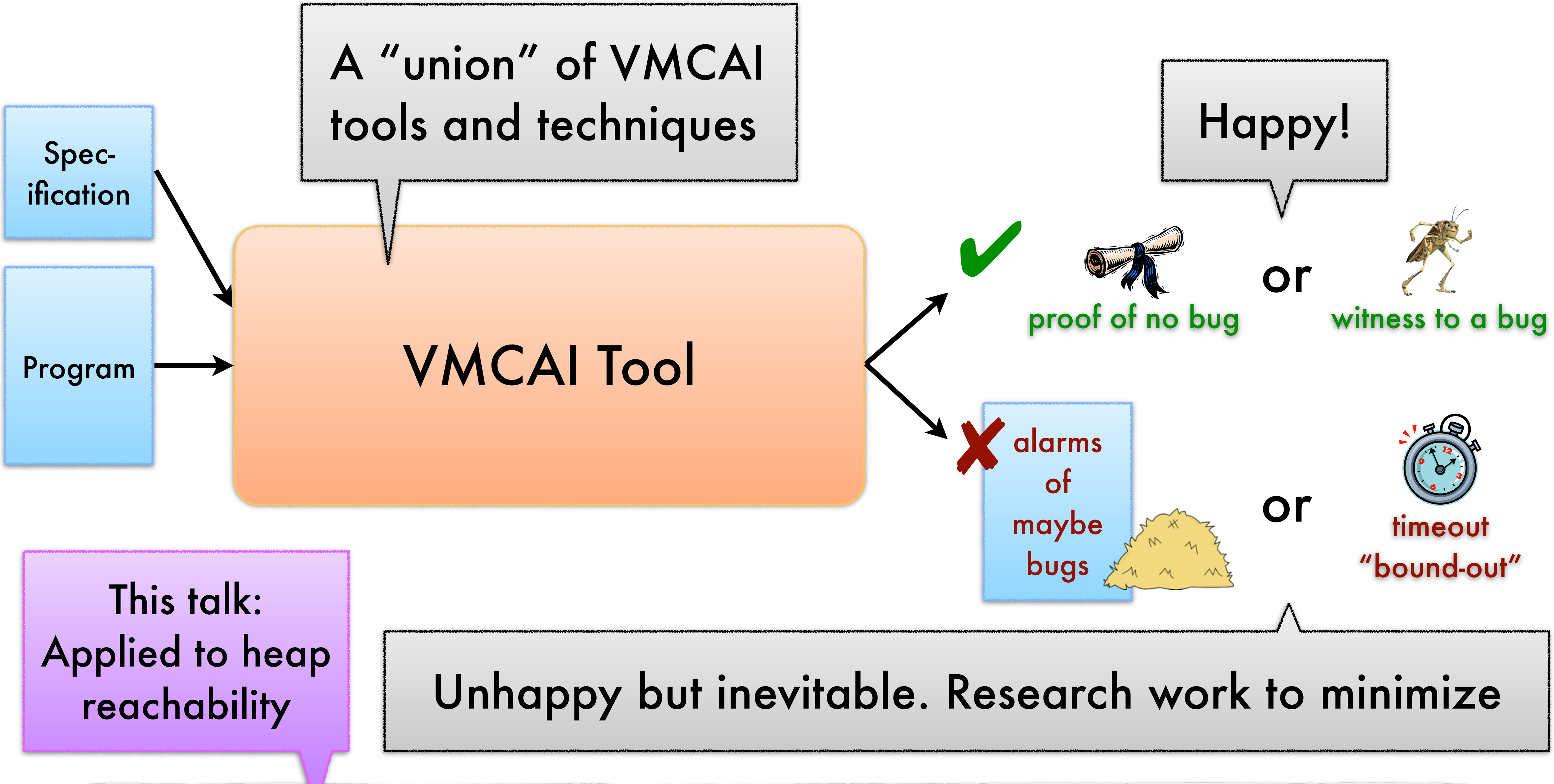
or

timeout "bound-out"

Unhappy but inevitable. Research work to minimize

Spirit of VMCAI: Recognize strength in combining V-MC-AI approaches

Perspective: The false alarm problem and VMCAI


A "union" of VMCAI tools and techniques

Spec-ification

Program

VMCAI Tool

✔ proof of no bug or witness to a bug

Happy!

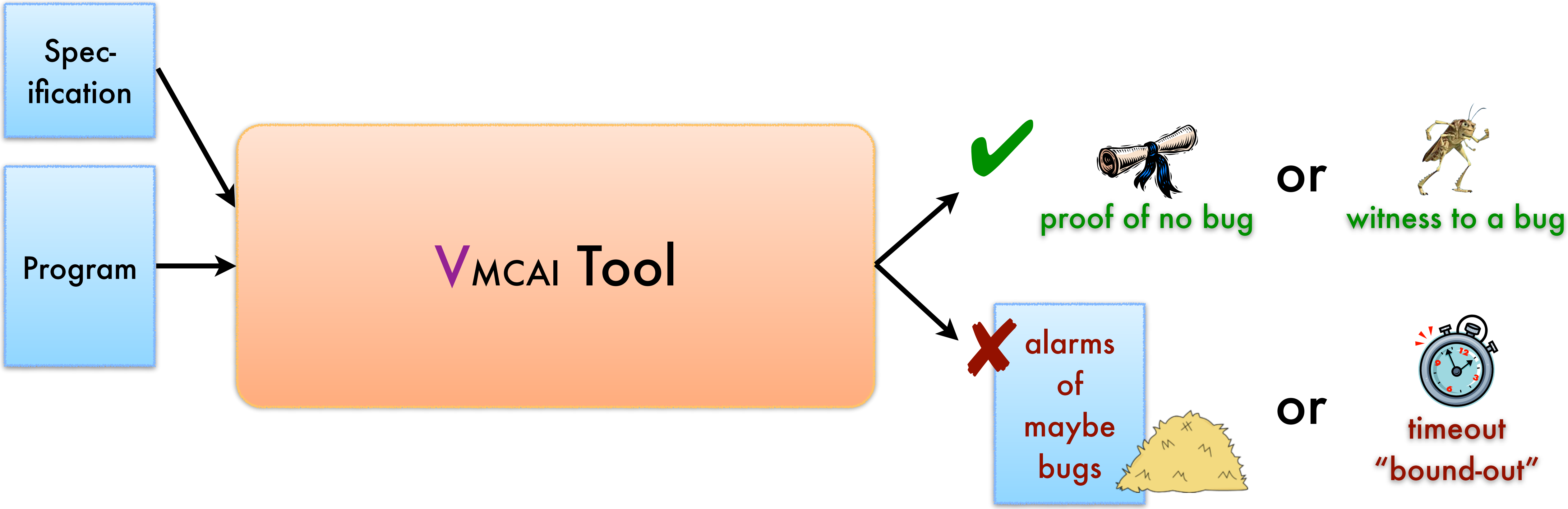
✘ alarms of maybe bugs or timeout "bound-out"

Unhappy but inevitable. Research work to minimize

This talk: Applied to heap reachability

Spirit of VMCAI: Recognize strength in combining V-MC-AI approaches

VMCAI: Make tradeoffs based on focus



Spec-ification

Program

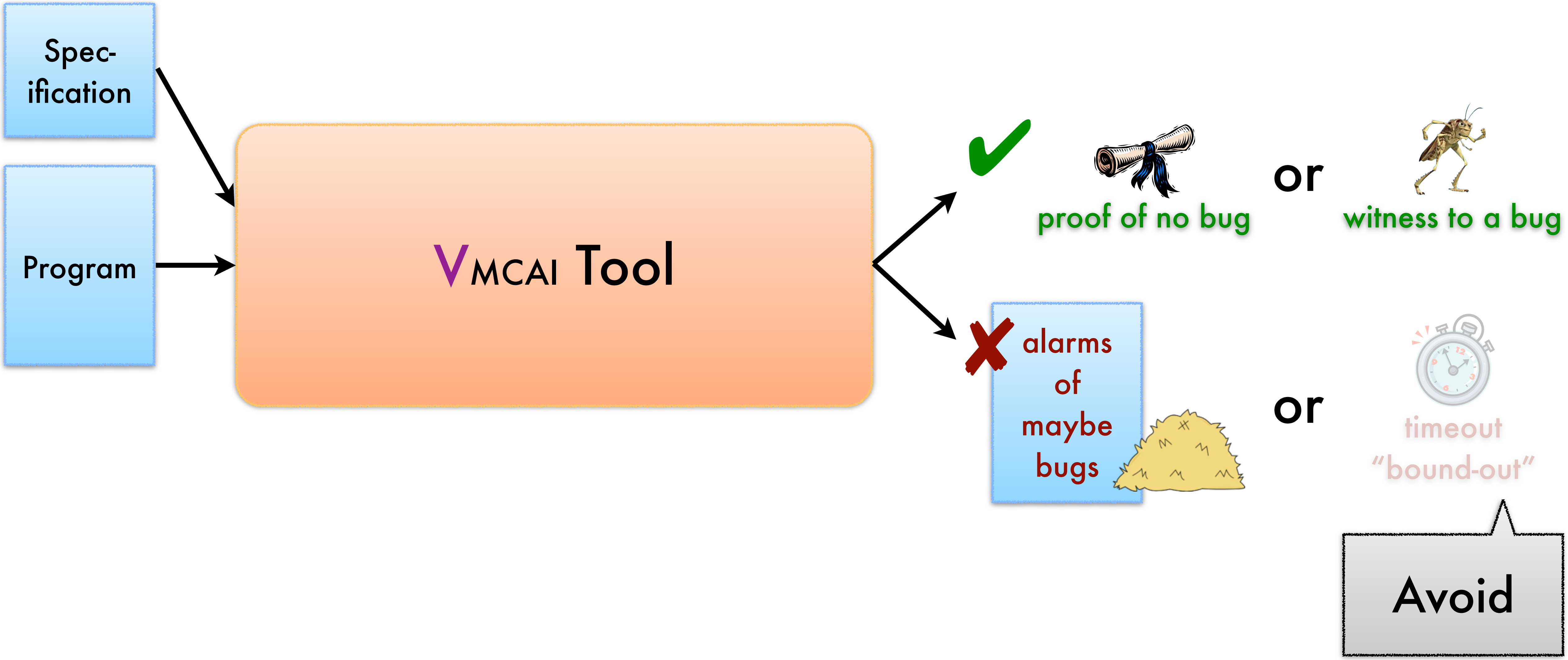
VMCAI Tool

✔ proof of no bug or witness to a bug

✘ alarms of maybe bugs or timeout "bound-out"

Dijkstra, Floyd, Hoare, …
ESC, Spec#, Boogie, Caduceus, Havoc, Calysto, Jahob, VCC, Dryad, …

VMCAI: Make tradeoffs based on focus



Spec-
ification

Program

VMCAI Tool

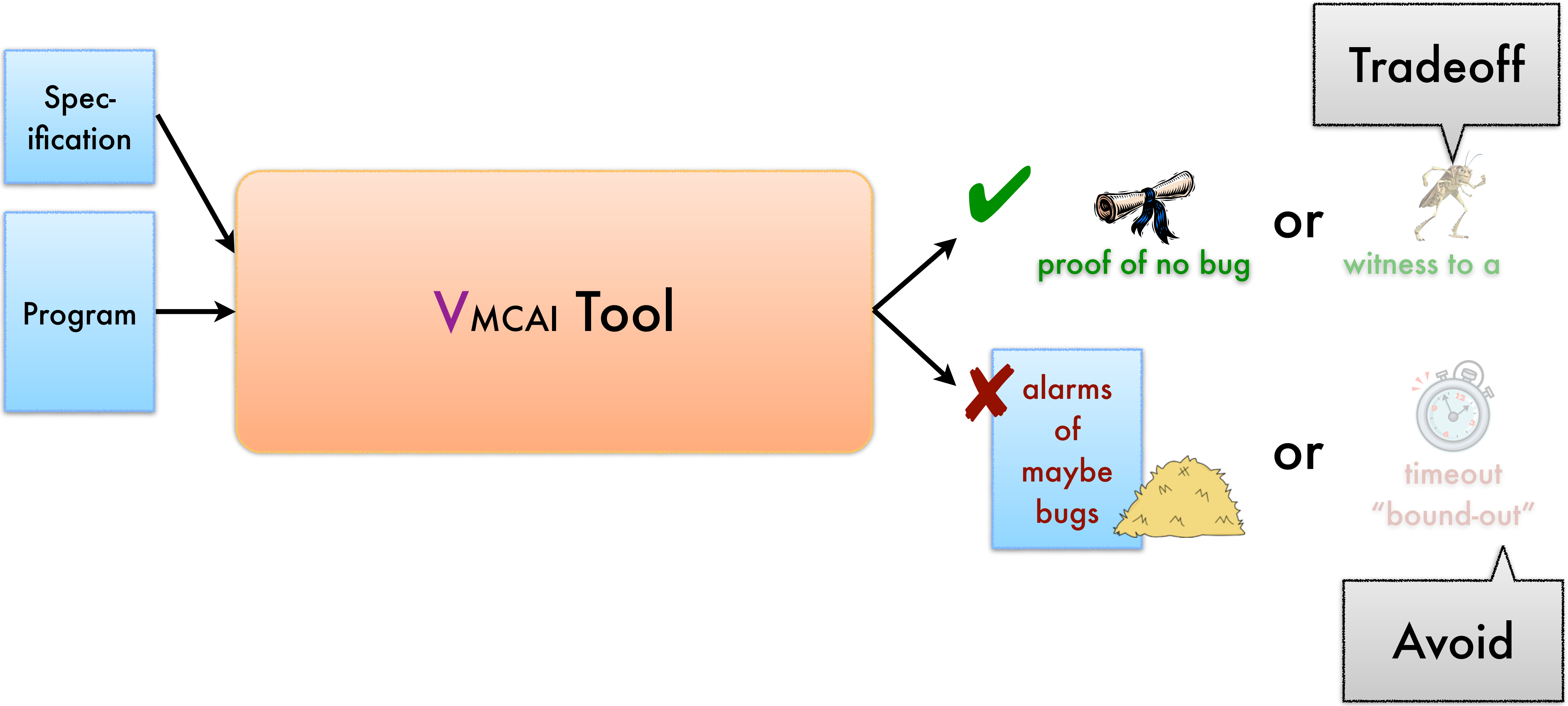
✔ proof of no bug or witness to a bug

✘ alarms of maybe bugs or timeout "bound-out"

Avoid

Dijkstra, Floyd, Hoare, …
ESC, Spec#, Boogie, Caduceus, Havoc, Calysto, Jahob, VCC, Dryad, …

VMCAI: Make tradeoffs based on focus



Spec-ification

Program

VMCAI Tool

✔ proof of no bug

or

witness to a

Tradeoff

✘ alarms of maybe bugs

or

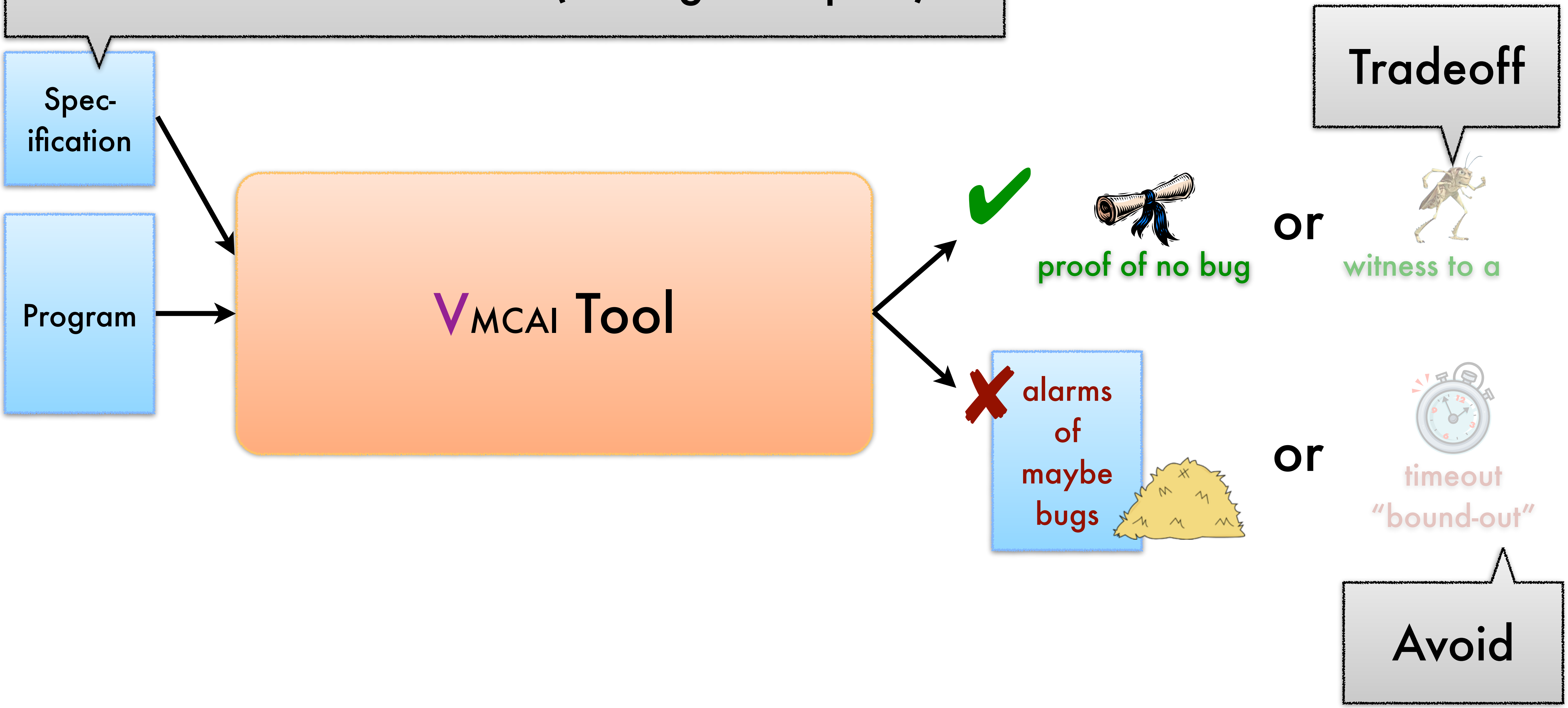
timeout "bound-out"

Avoid

Dijkstra, Floyd, Hoare, …
ESC, Spec#, Boogie, Caduceus, Havoc, Calysto, Jahob, VCC, Dryad, …

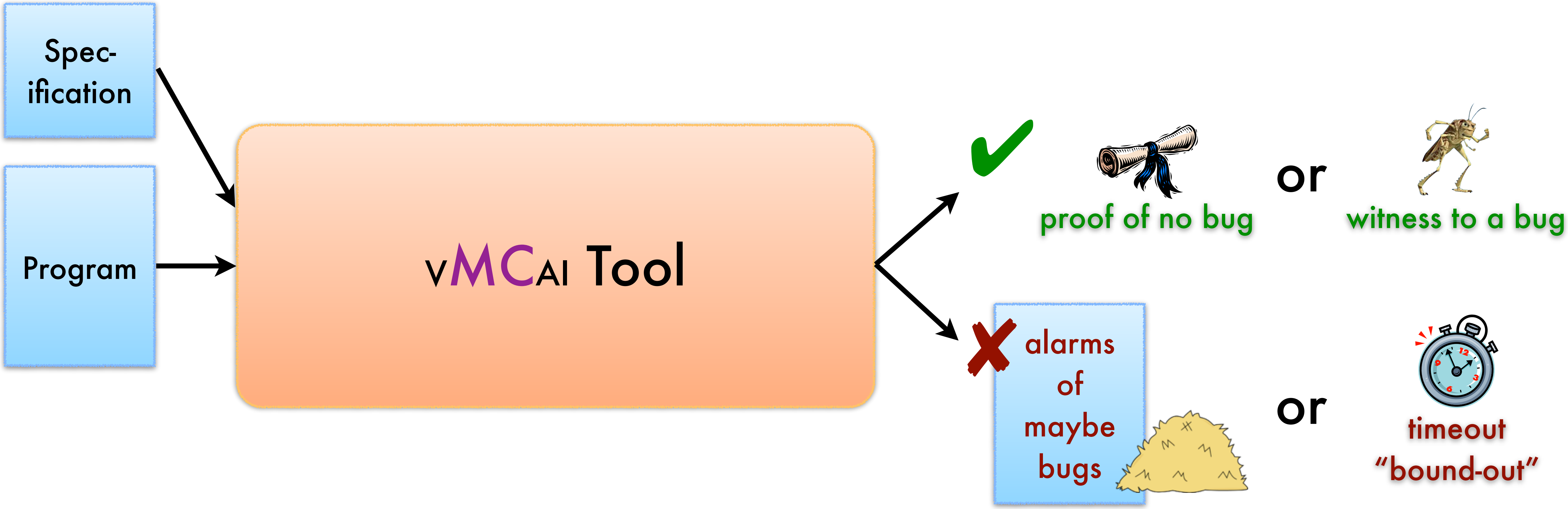
VMCAI: Make tradeoffs based on focus

Expressivity and Usability: Specification can eliminate false alarms (the right loopinv)



Dijkstra, Floyd, Hoare, …
ESC, Spec#, Boogie, Caduceus, Havoc, Calysto, Jahob, VCC, Dryad, …

VMCAI: Make tradeoffs based on focus



Specification

Program

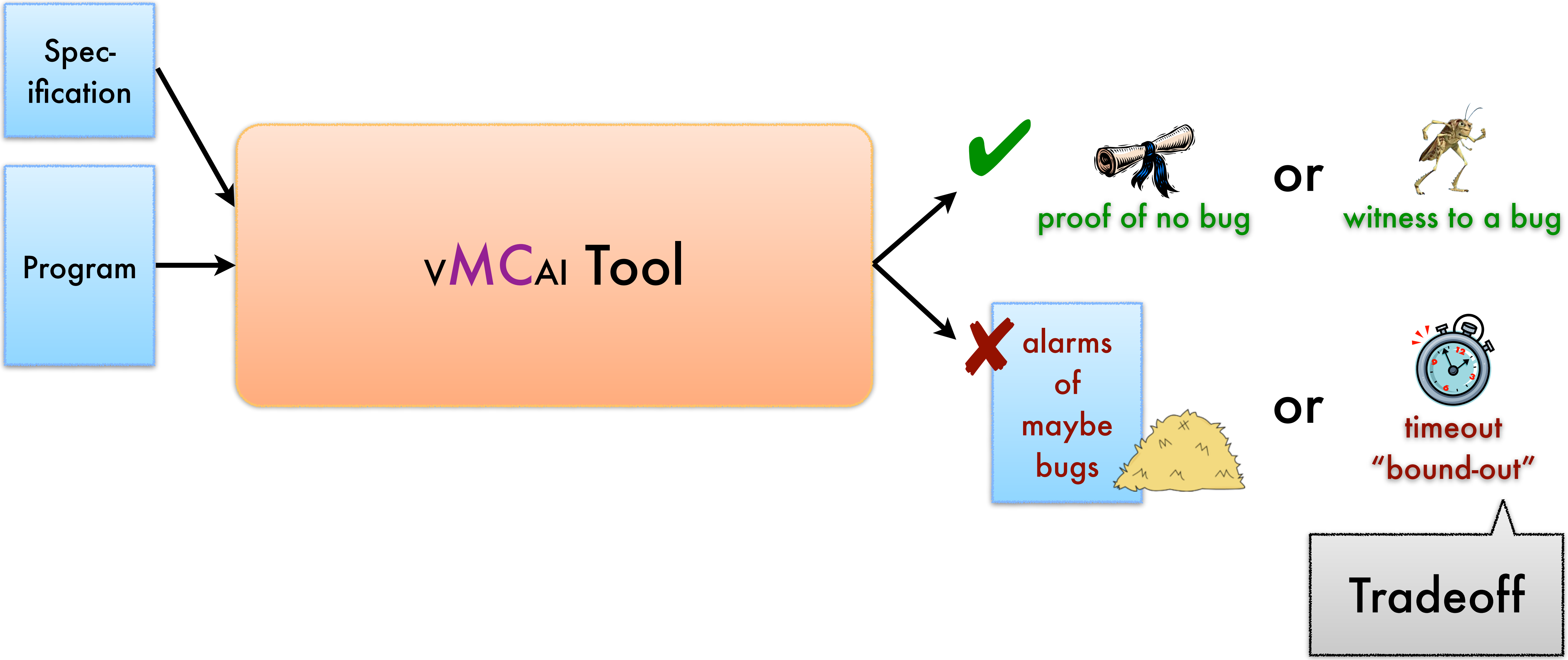
VMCAI Tool

✔ proof of no bug or witness to a bug

✘ alarms of maybe bugs or timeout "bound-out"

Clarke, Emerson, Sifakis, McMillan, ...
BDDs (SMV, ...), CEGAR (SLAM, Blast, ...), Interpolation (Impact, ...),
JPF, FSoft, CBMC, ...

VMCAI: Make tradeoffs based on focus



Spec-ification

Program

vMCAI Tool

✔ proof of no bug

or

witness to a bug

✘ alarms of maybe bugs

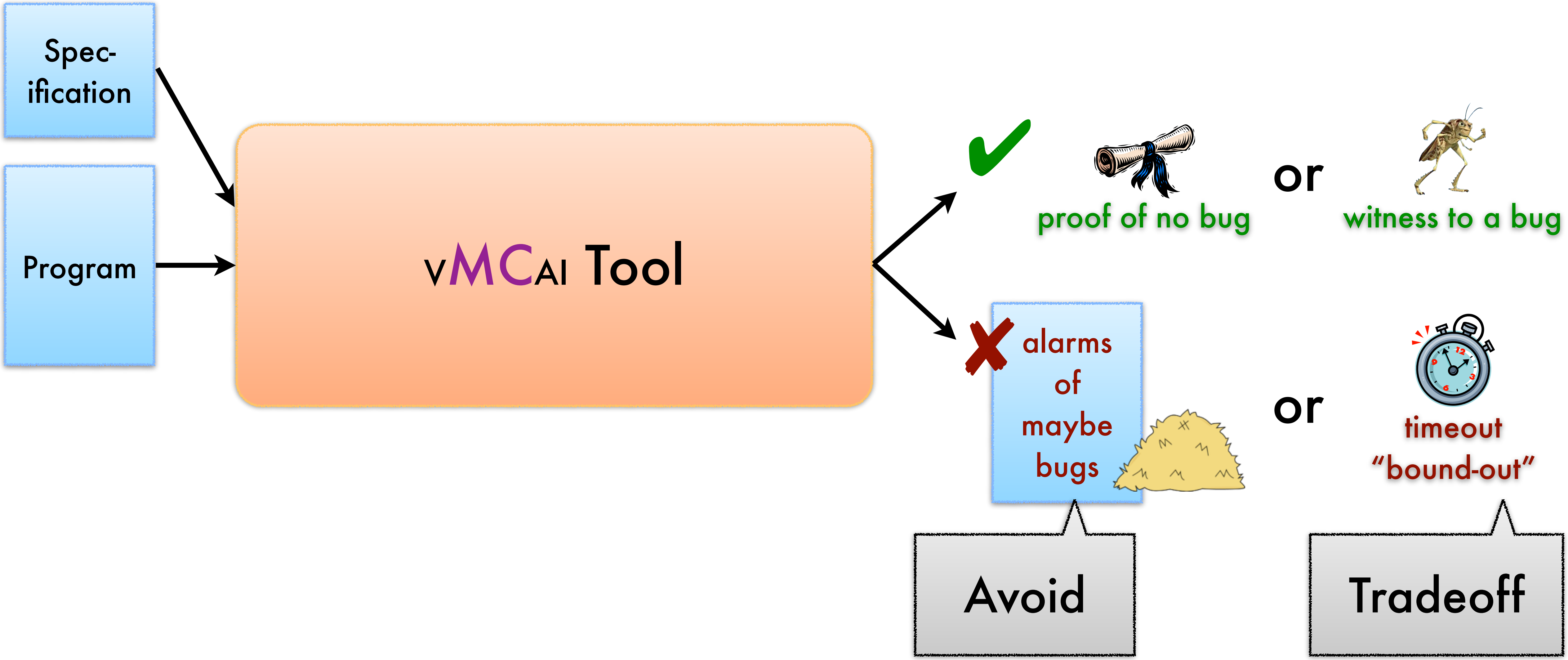
or

timeout
"bound-out"

Tradeoff

Clarke, Emerson, Sifakis, McMillan, ...
BDDs (SMV, ...), CEGAR (SLAM, Blast, ...), Interpolation (Impact, ...),
JPF, FSoft, CBMC, ...

VMCAI: Make tradeoffs based on focus



Spec-ification

Program

vMCAI Tool

✔ proof of no bug

or

witness to a bug

✗ alarms of maybe bugs

or

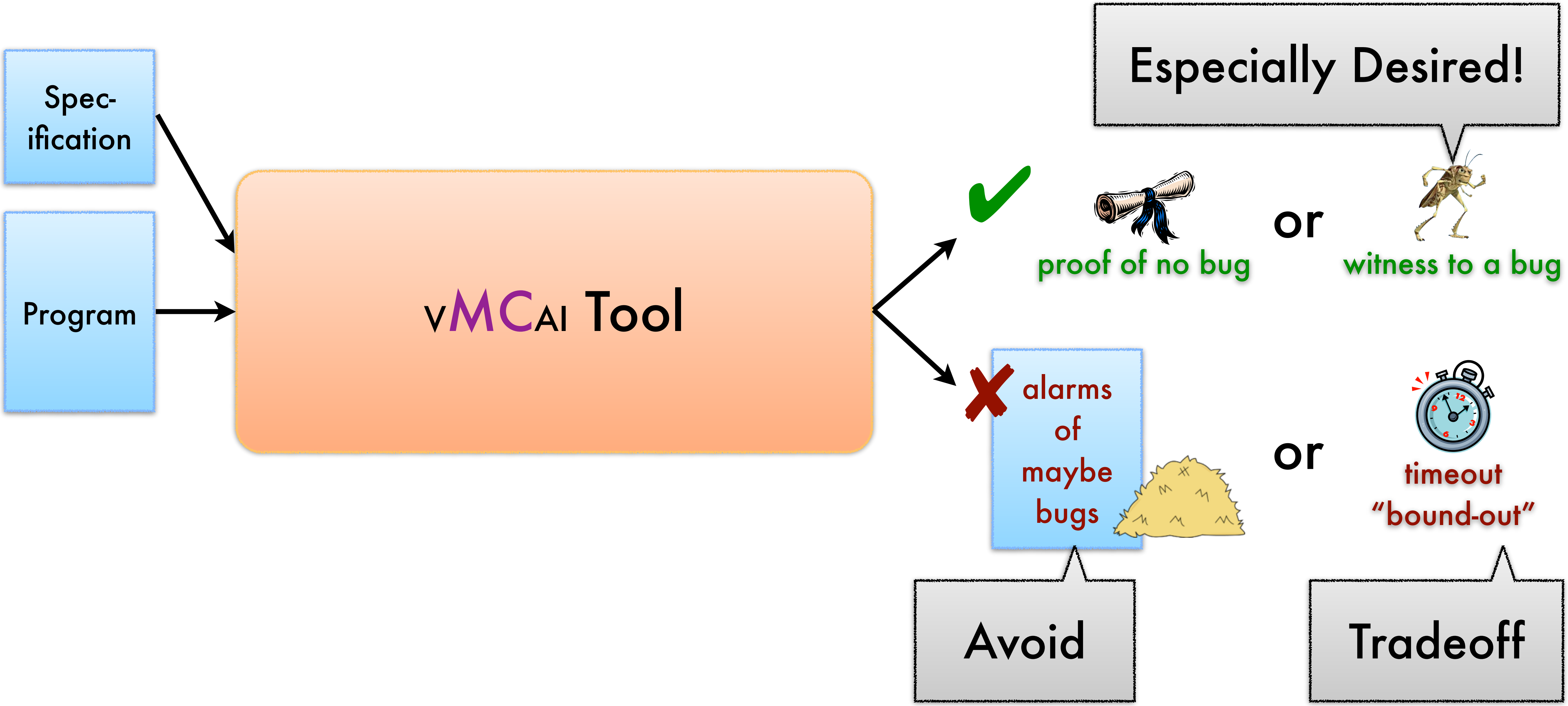
timeout "bound-out"

Avoid

Tradeoff

Clarke, Emerson, Sifakis, McMillan, …
BDDs (SMV, …), CEGAR (SLAM, Blast, …), Interpolation (Impact, …),
JPF, FSoft, CBMC, …

VMCAI: Make tradeoffs based on focus



Spec-ification

Program

VMCAI Tool

✔ proof of no bug or witness to a bug

Especially Desired!

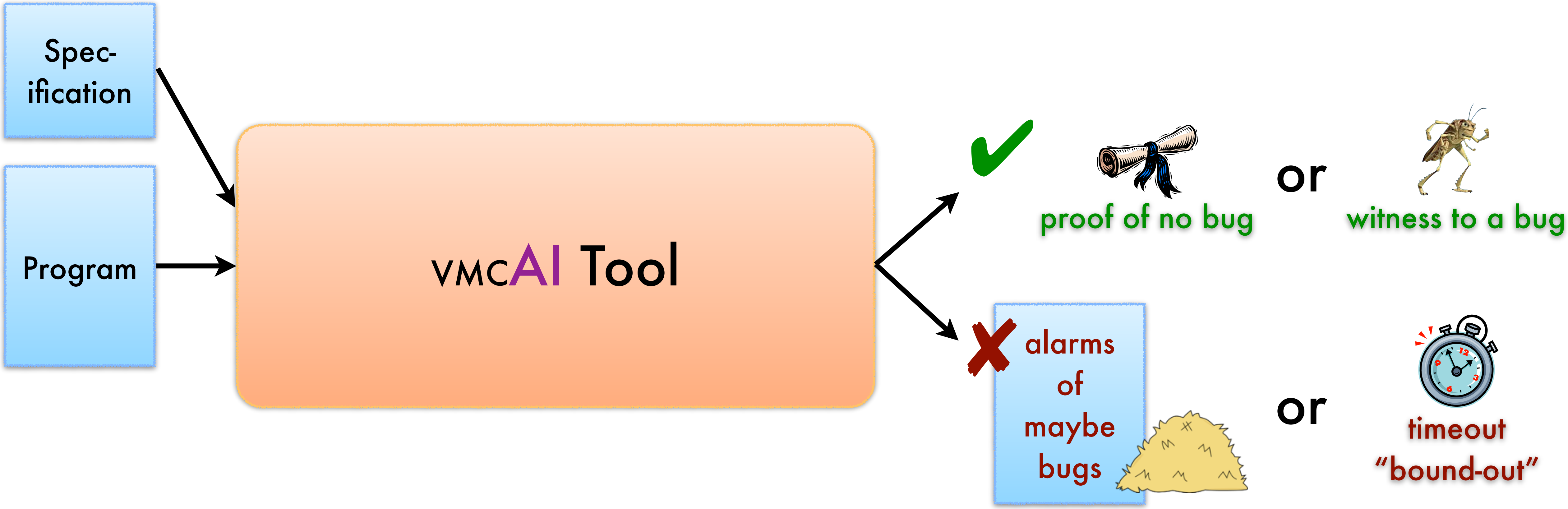
✘ alarms of maybe bugs or timeout "bound-out"

Avoid

Tradeoff

Clarke, Emerson, Sifakis, McMillan, ...
BDDs (SMV, ...), CEGAR (SLAM, Blast, ...), Interpolation (Impact, ...),
JPF, FSoft, CBMC, ...

VMCAI: Make tradeoffs based on focus



Spec-ification

Program

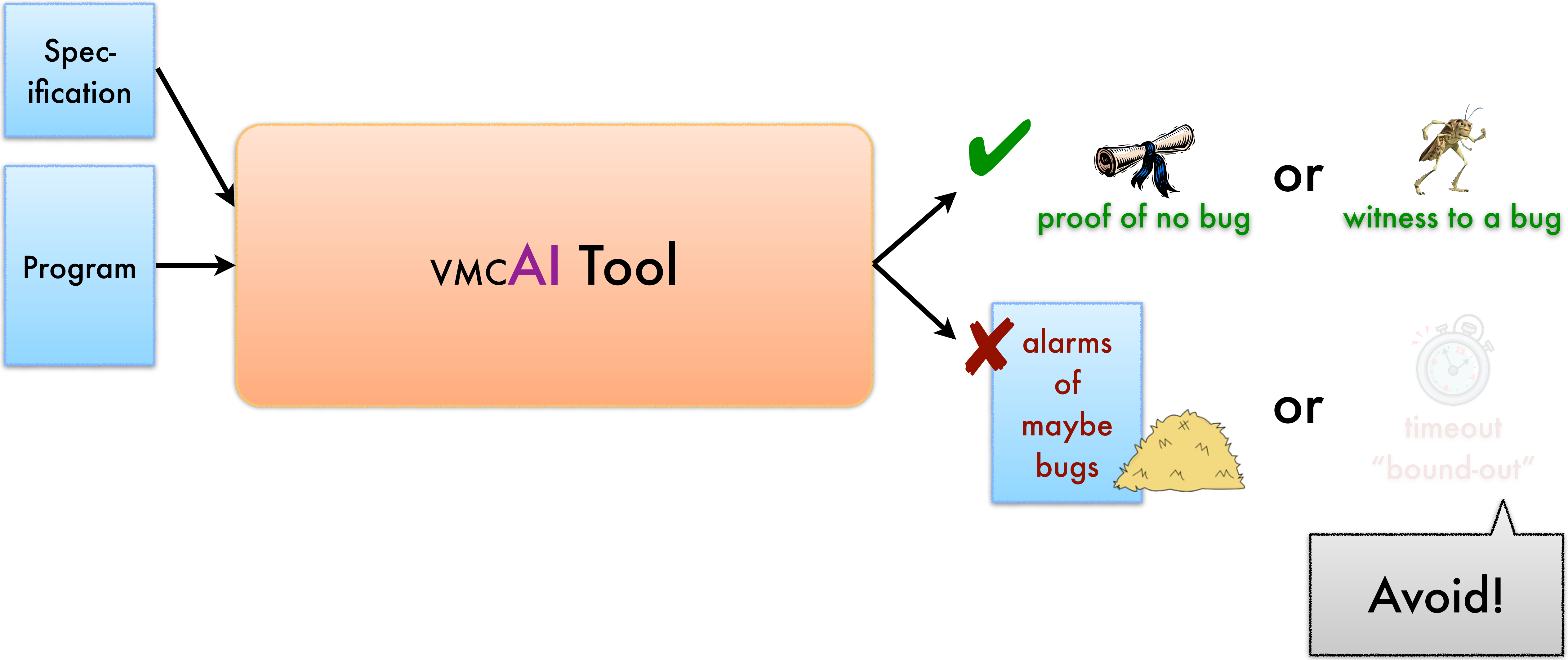
VMCAI Tool

✔ proof of no bug or witness to a bug

✘ alarms of maybe bugs or timeout "bound-out"

Cousot, Cousot, ...
Polyhedra/Octagons/... (Apron), Astrée, Polyspace, Fluctuat, Clousot, SpaceInvader, Slayer, ...

VMCAI: Make tradeoffs based on focus



Spec-ification

Program

vmcAI Tool

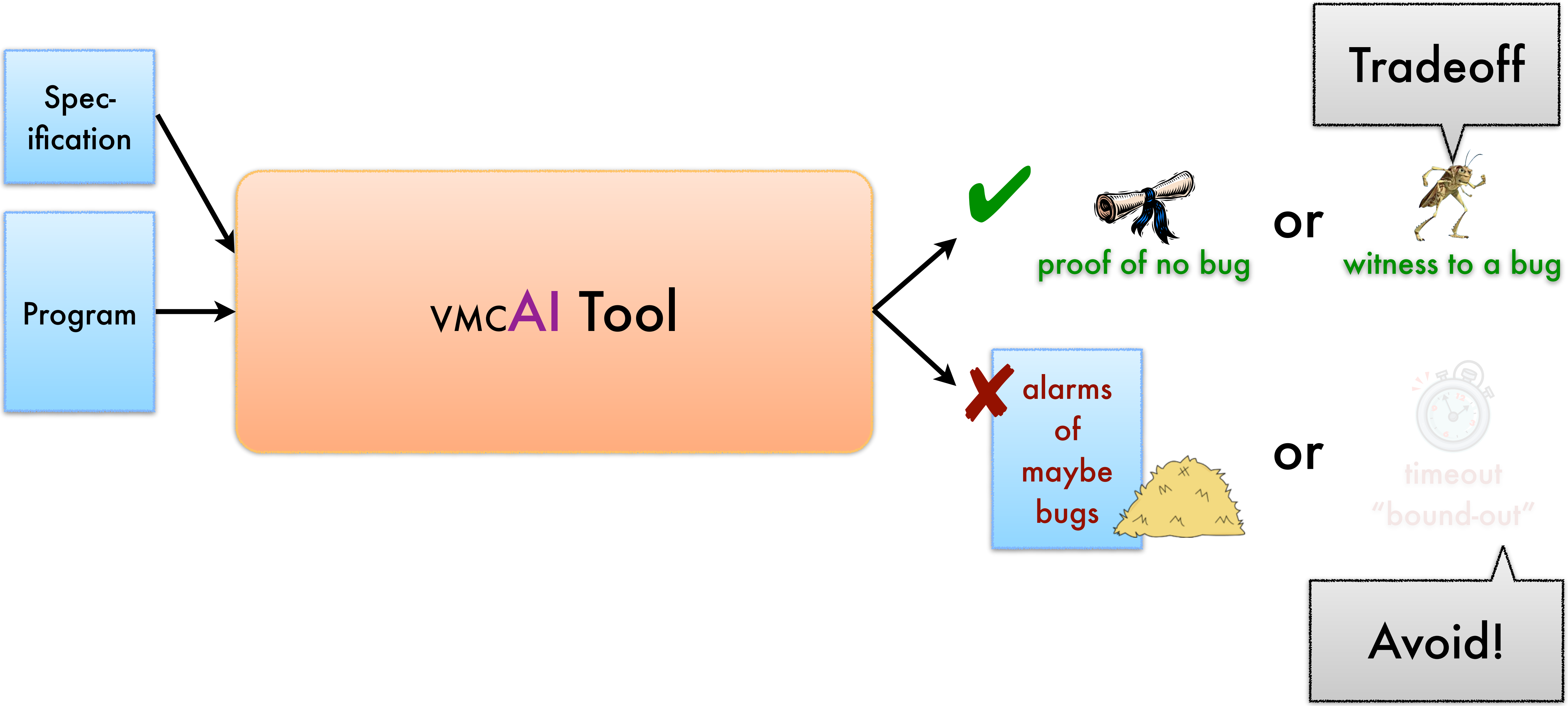
✔ proof of no bug or witness to a bug

✘ alarms of maybe bugs or timeout "bound-out"

Avoid!

Cousot, Cousot, ...
Polyhedra/Octagons/... (Apron), Astrée, Polyspace, Fluctuat, Clousot, SpaceInvader, Slayer, ...

VMCAI: Make tradeoffs based on focus



Spec-ification

Program

VMCAI Tool

✔ proof of no bug

or

witness to a bug

Tradeoff

✘ alarms of maybe bugs

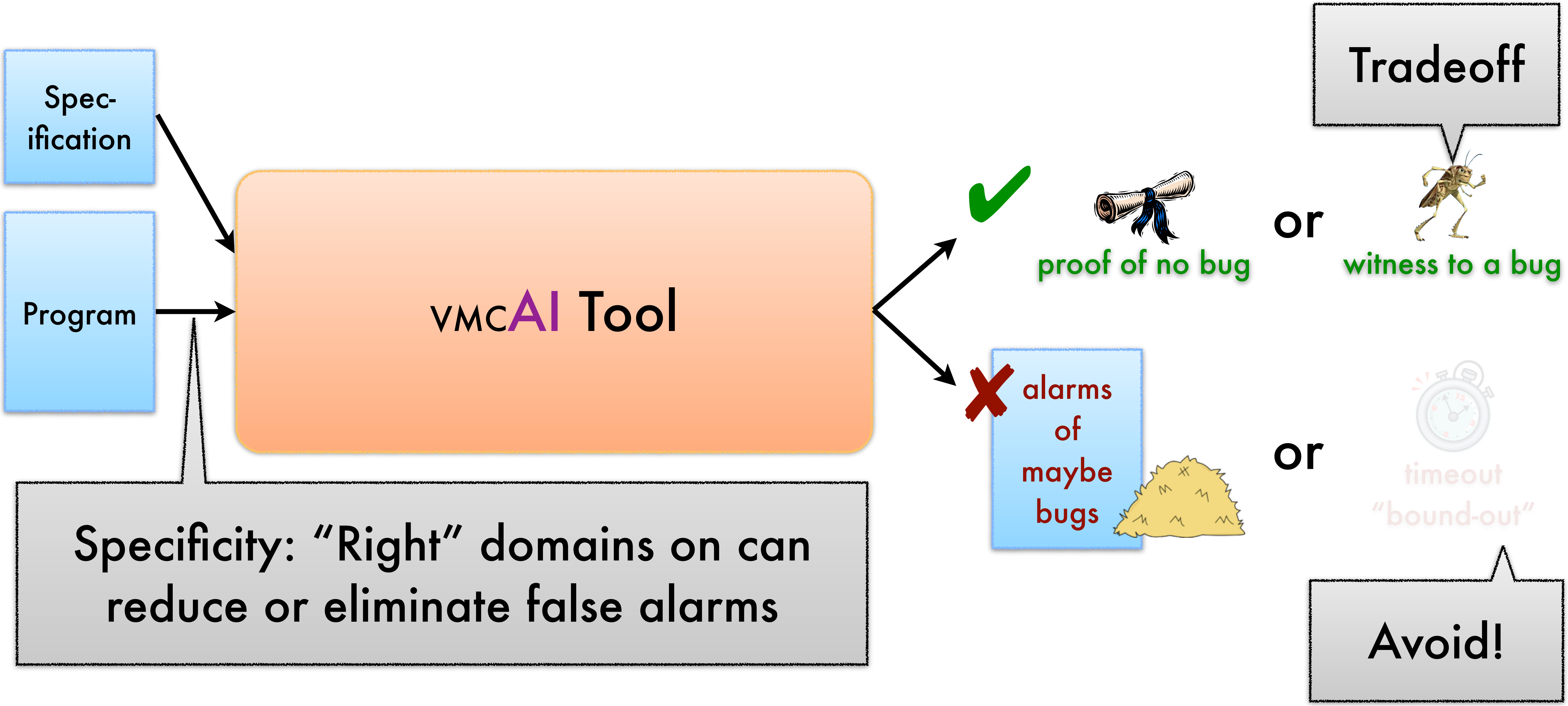
or

timeout "bound-out"

Avoid!

Cousot, Cousot, ...
Polyhedra/Octagons/... (Apron), Astrée, Polyspace, Fluctuat, Clousot, SpaceInvader, Slayer, ...

VMCAI: Make tradeoffs based on focus



Spec-ification

Program

VMCAI Tool

✓ proof of no bug

or

witness to a bug

Tradeoff

✗ alarms of maybe bugs

or

timeout "bound-out"

Avoid!

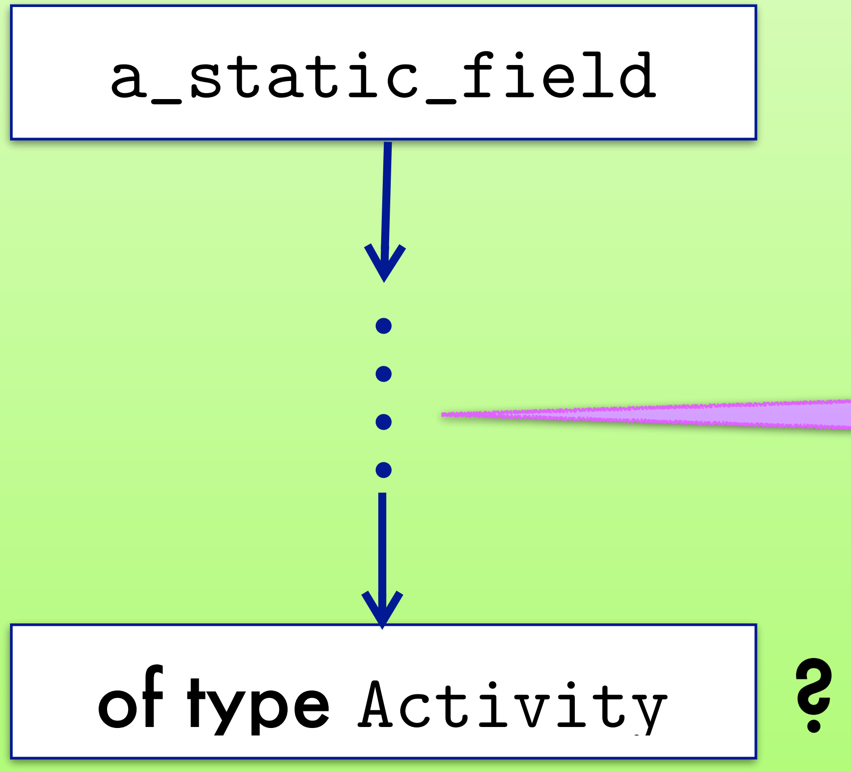
Specificity: "Right" domains on can reduce or eliminate false alarms

Cousot, Cousot, ...
Polyhedra/Octagons/... (Apron), Astrée, Polyspace, Fluctuat, Clousot, SpaceInvader, Slayer, ...

Back to …

A candidate for statically answering, “Is there an `Activity` leak?”

Is there a program execution where at some time



`a_static_field`

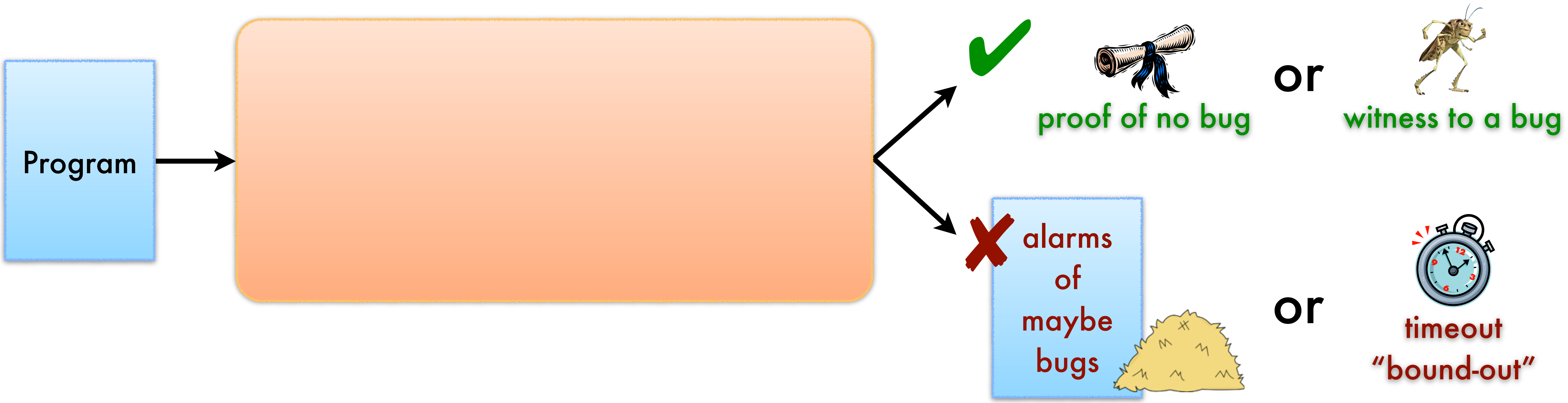
`of type Activity` ?

Can be answered with a points-to analysis

Compute a points-to graph and look for such points-to paths

This won’t work because …

Thresher [SAS'11,PLDI'13] attacks alarm triage for heap reachability properties

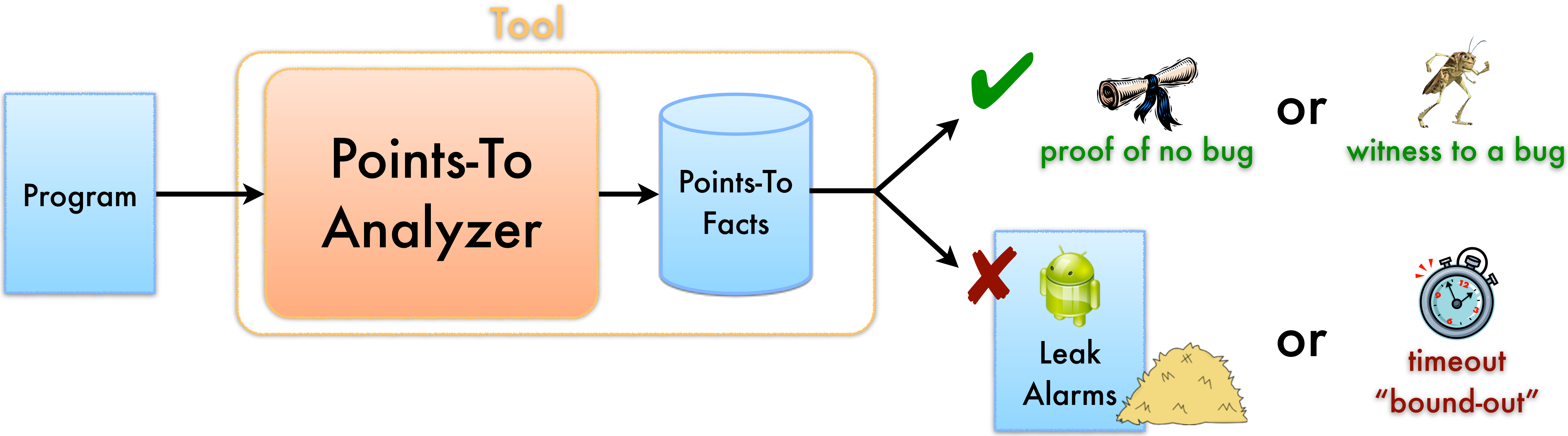


Program

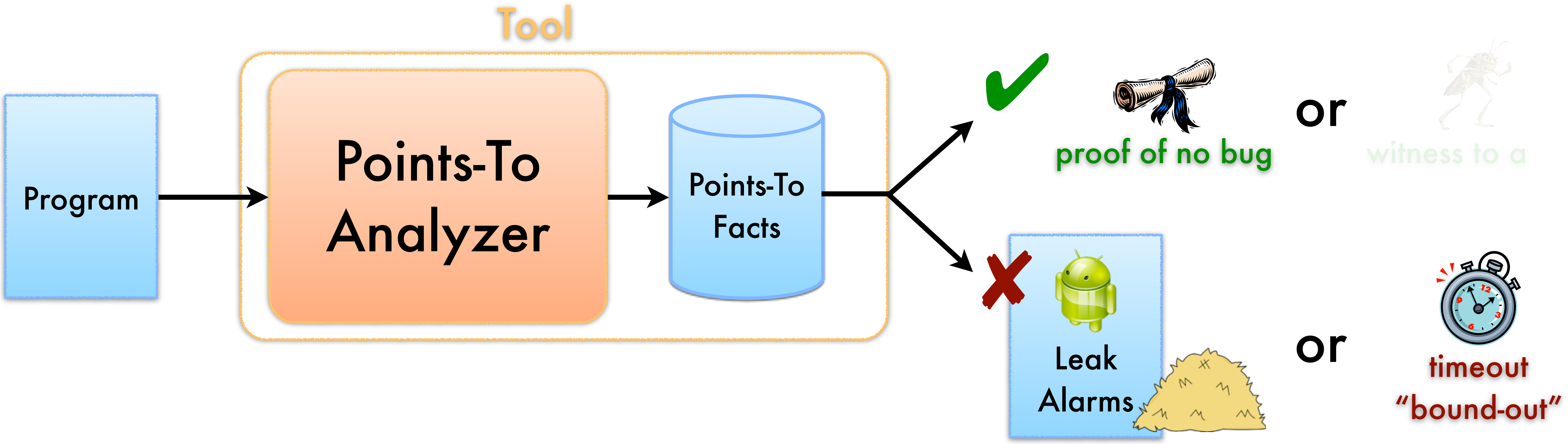
✔ proof of no bug or witness to a bug

✘ alarms of maybe bugs or timeout "bound-out"

Thresher [SAS'11,PLDI'13] attacks alarm triage for heap reachability properties



Thresher [SAS'11,PLDI'13] attacks alarm triage for heap reachability properties



Tool

Program

Points-To Analyzer

Points-To Facts

✔ proof of no bug

or

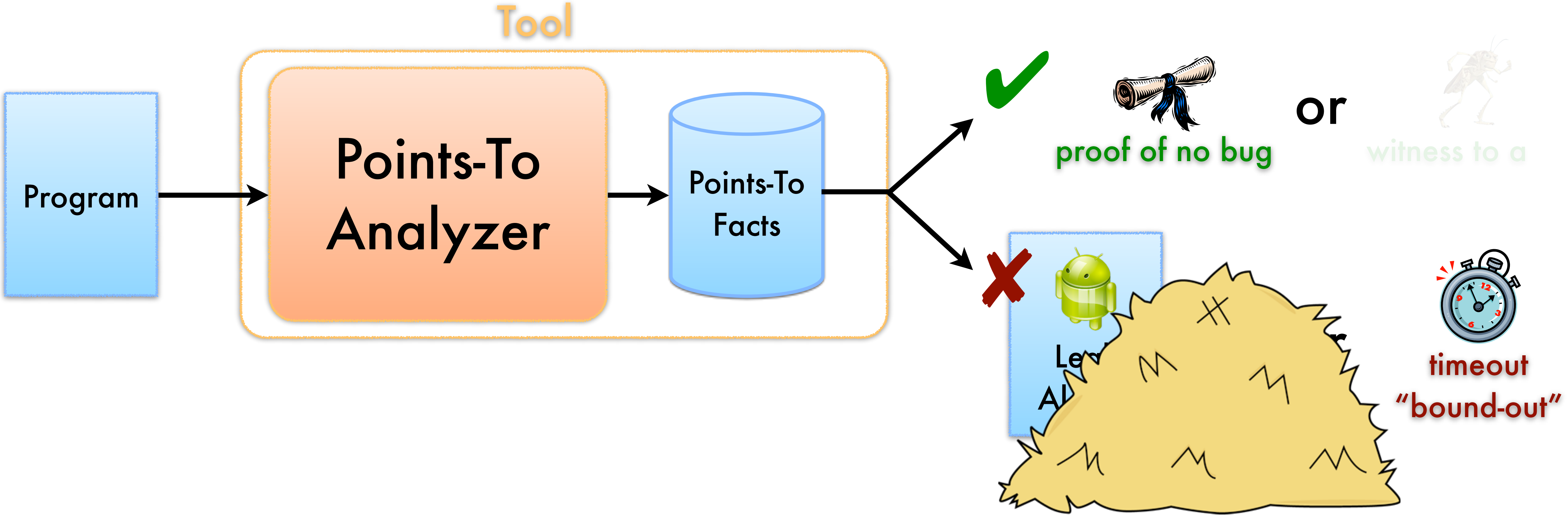
witness to a

✘ Leak Alarms

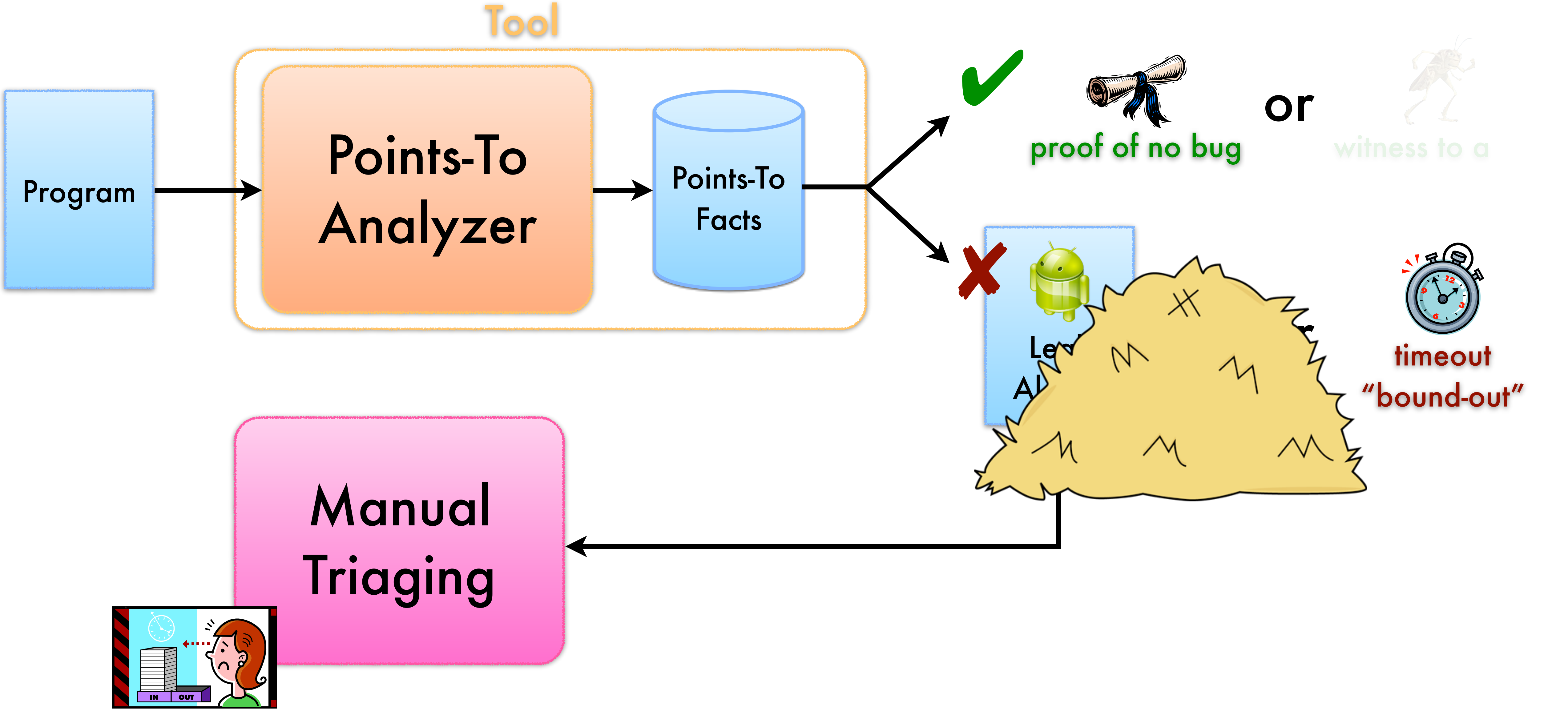
or

timeout "bound-out"

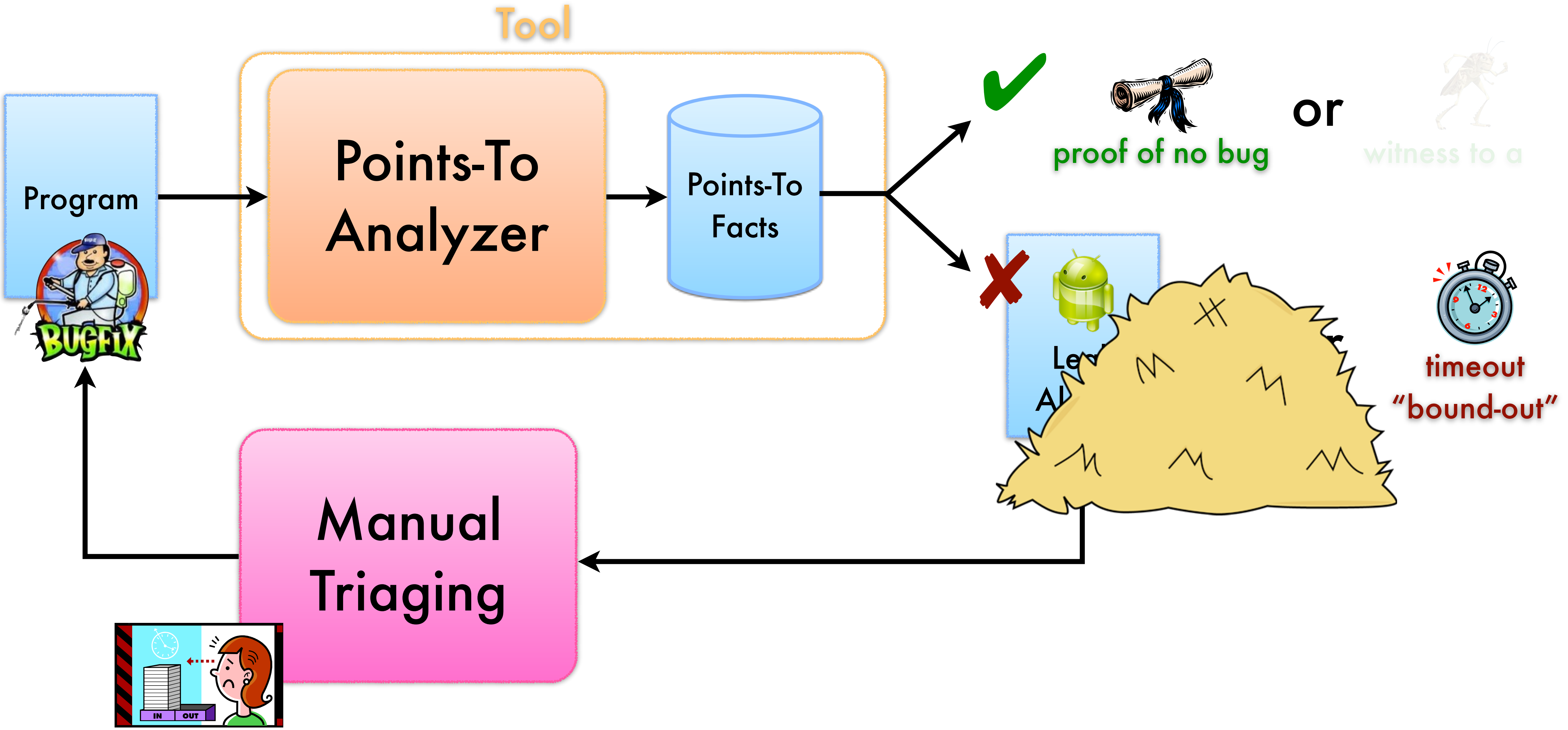
Thresher [SAS'11,PLDI'13] attacks alarm triage for heap reachability properties



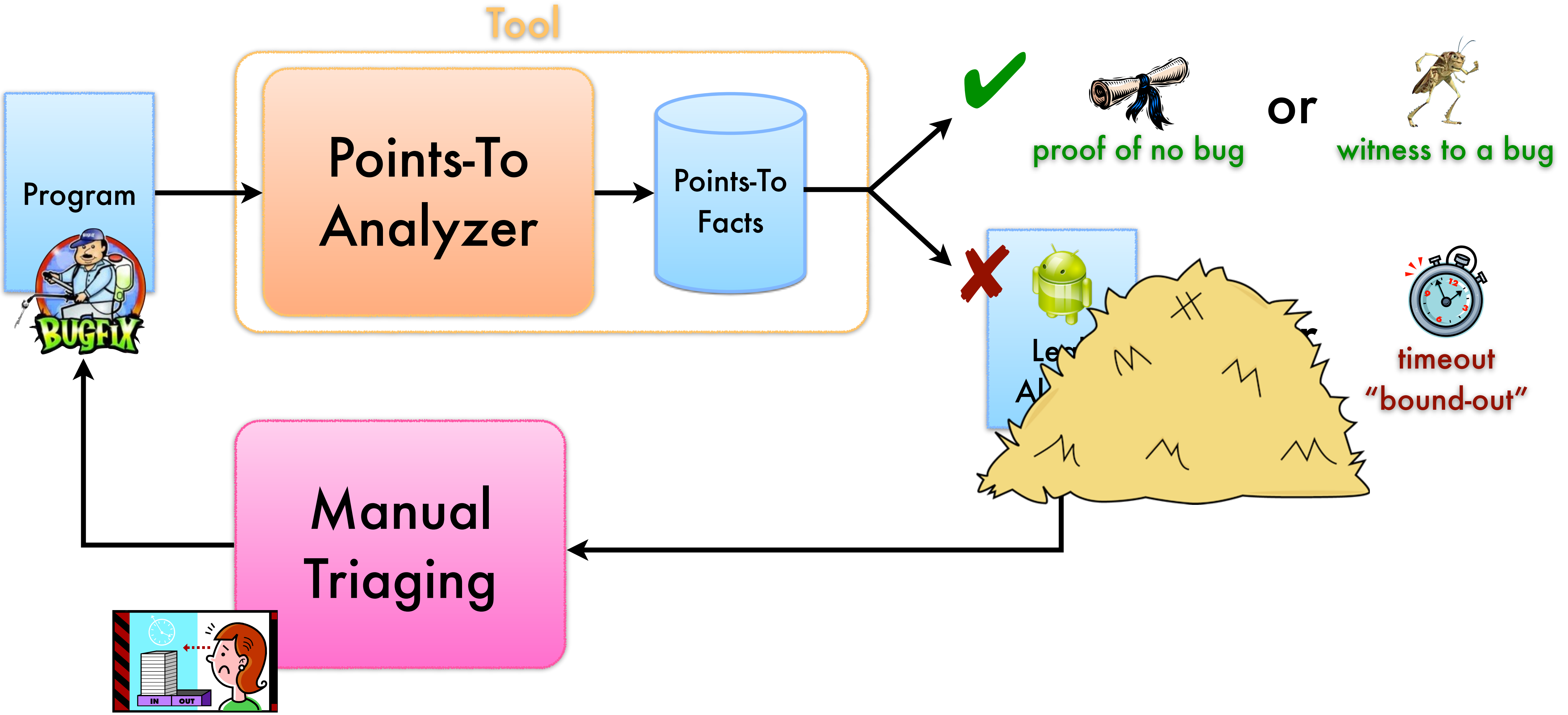
Thresher [SAS'11,PLDI'13] attacks alarm triage for heap reachability properties



Thresher [SAS'11,PLDI'13] attacks alarm triage for heap reachability properties



Thresher [SAS'11,PLDI'13] attacks alarm triage for heap reachability properties



Tool

Program

Points-To Analyzer

Points-To Facts

✔ proof of no bug or witness to a bug

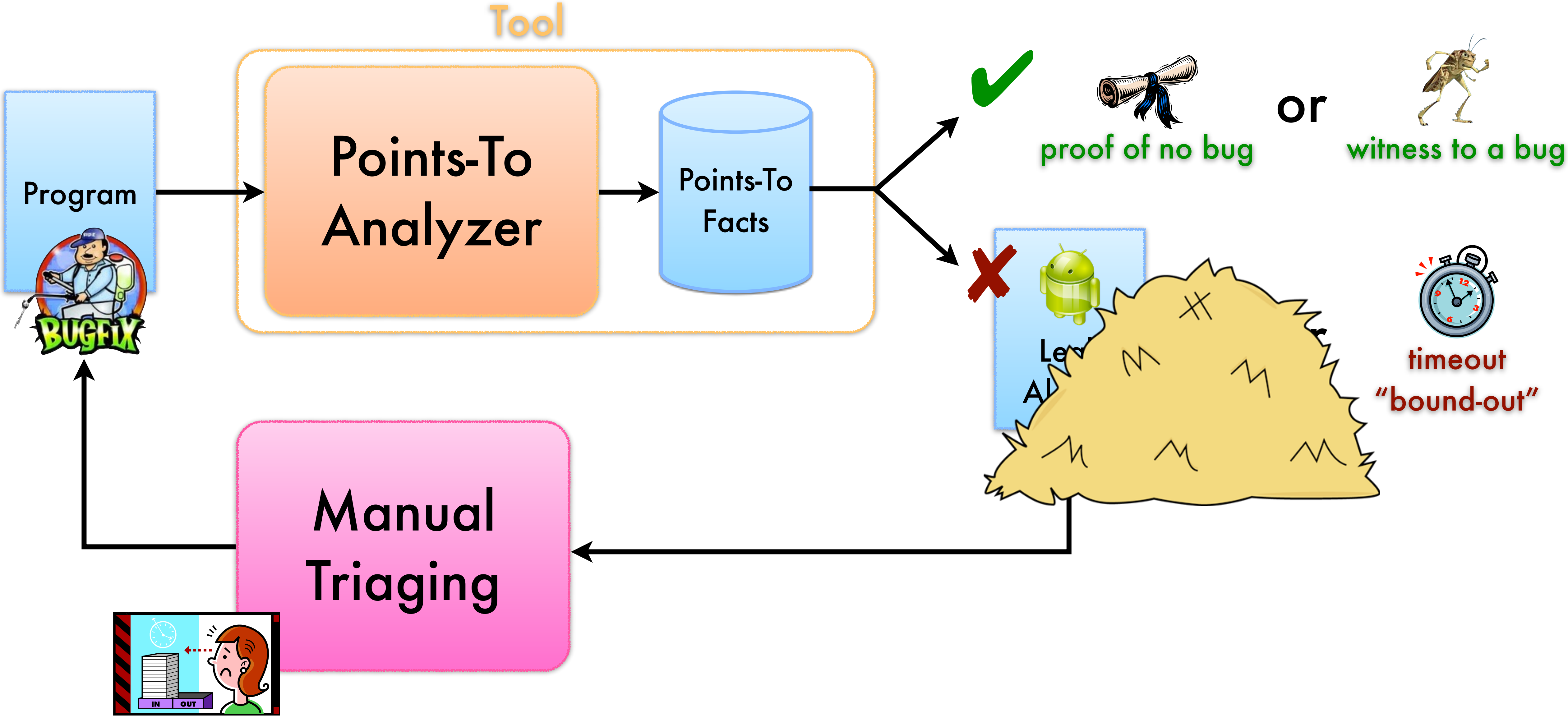
✘ Le Al

timeout

"bound-out"

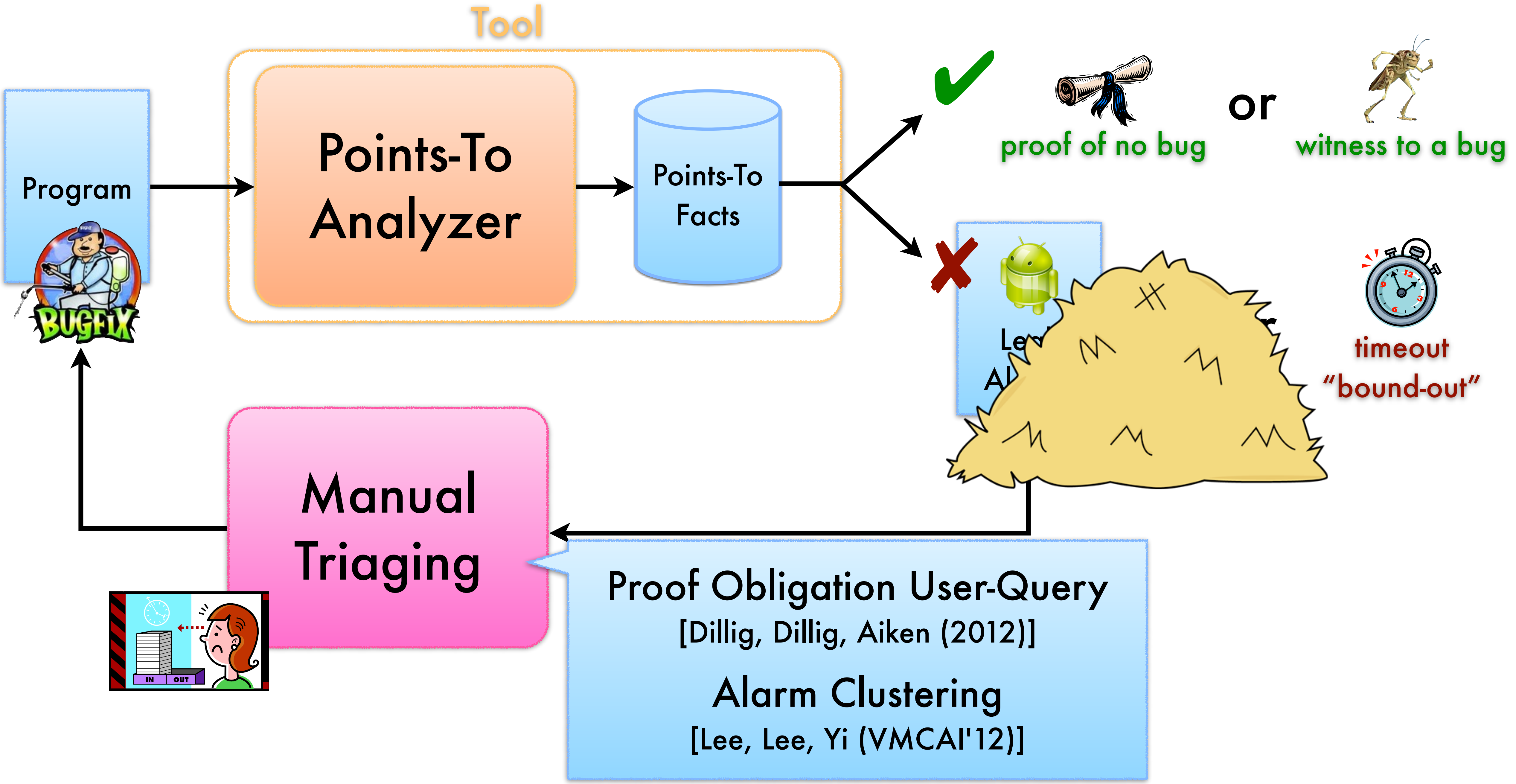
Manual Triaging

Thresher [SAS'11,PLDI'13] attacks alarm triage for heap reachability properties



Prove alarms false with a witness search

Thresher [SAS'11,PLDI'13] attacks alarm **triage** for heap reachability properties



Tool

Program

Points-To Analyzer

Points-To Facts

✔ proof of no bug or witness to a bug

✘ Le Al

timeout "bound-out"

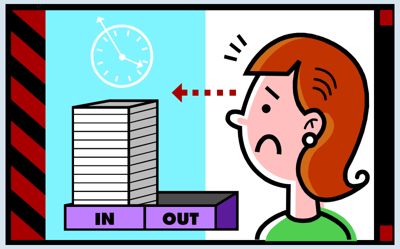
Manual Triaging

Proof Obligation User-Query [Dillig, Dillig, Aiken (2012)]

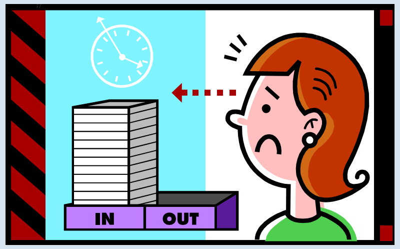
Alarm Clustering [Lee, Lee, Yi (VMCAI'12)]

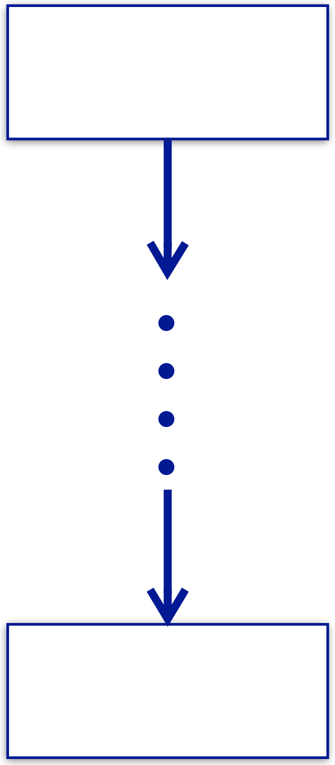
Prove alarms false with a witness search

Manual triage for heap reachability reports

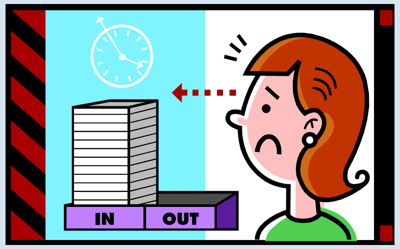


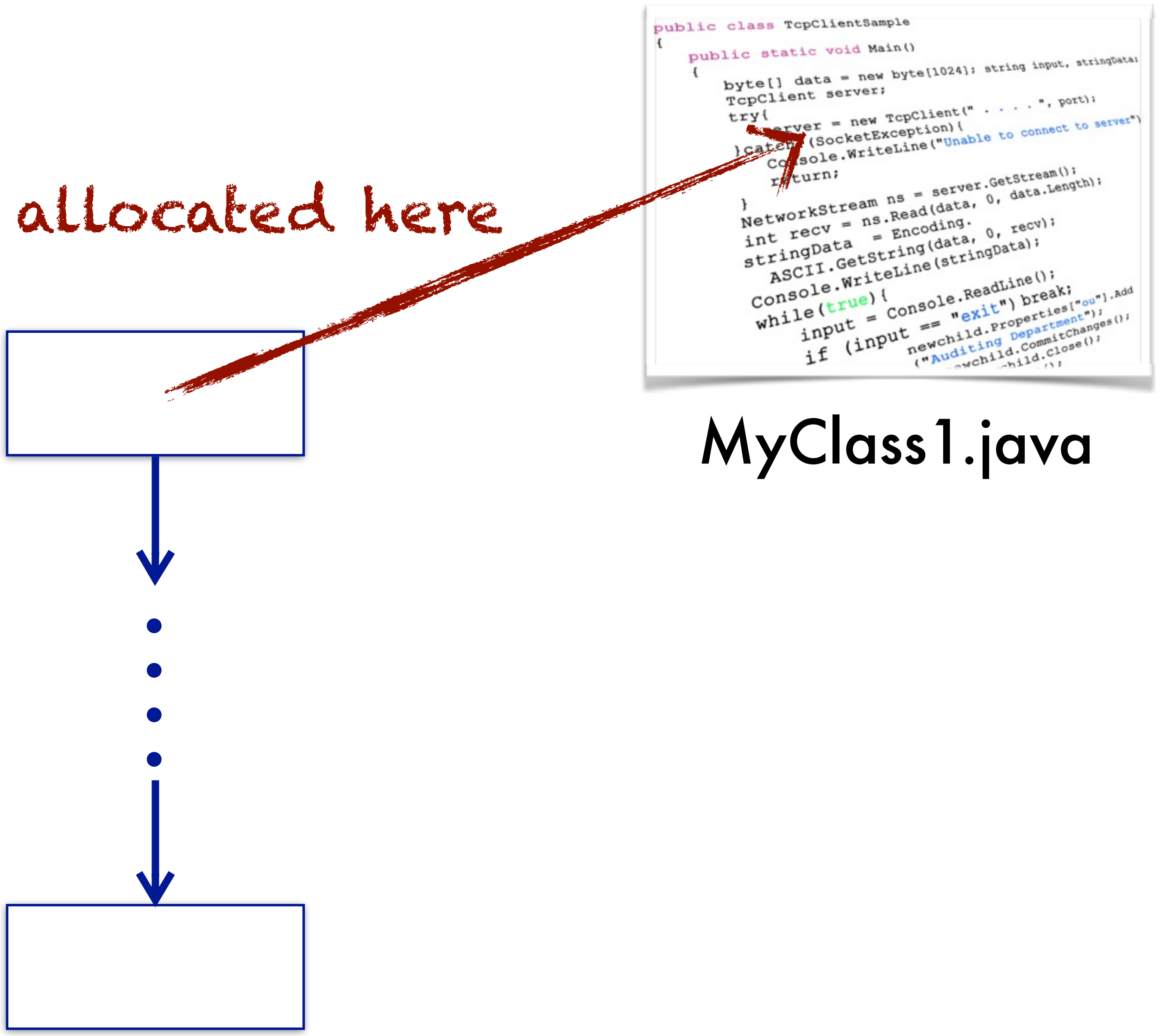
Manual triage for heap reachability reports





Manual triage for heap reachability reports



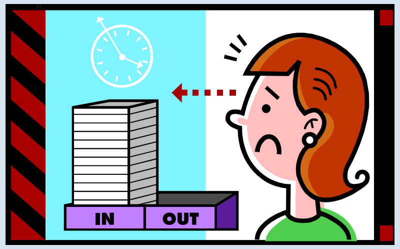


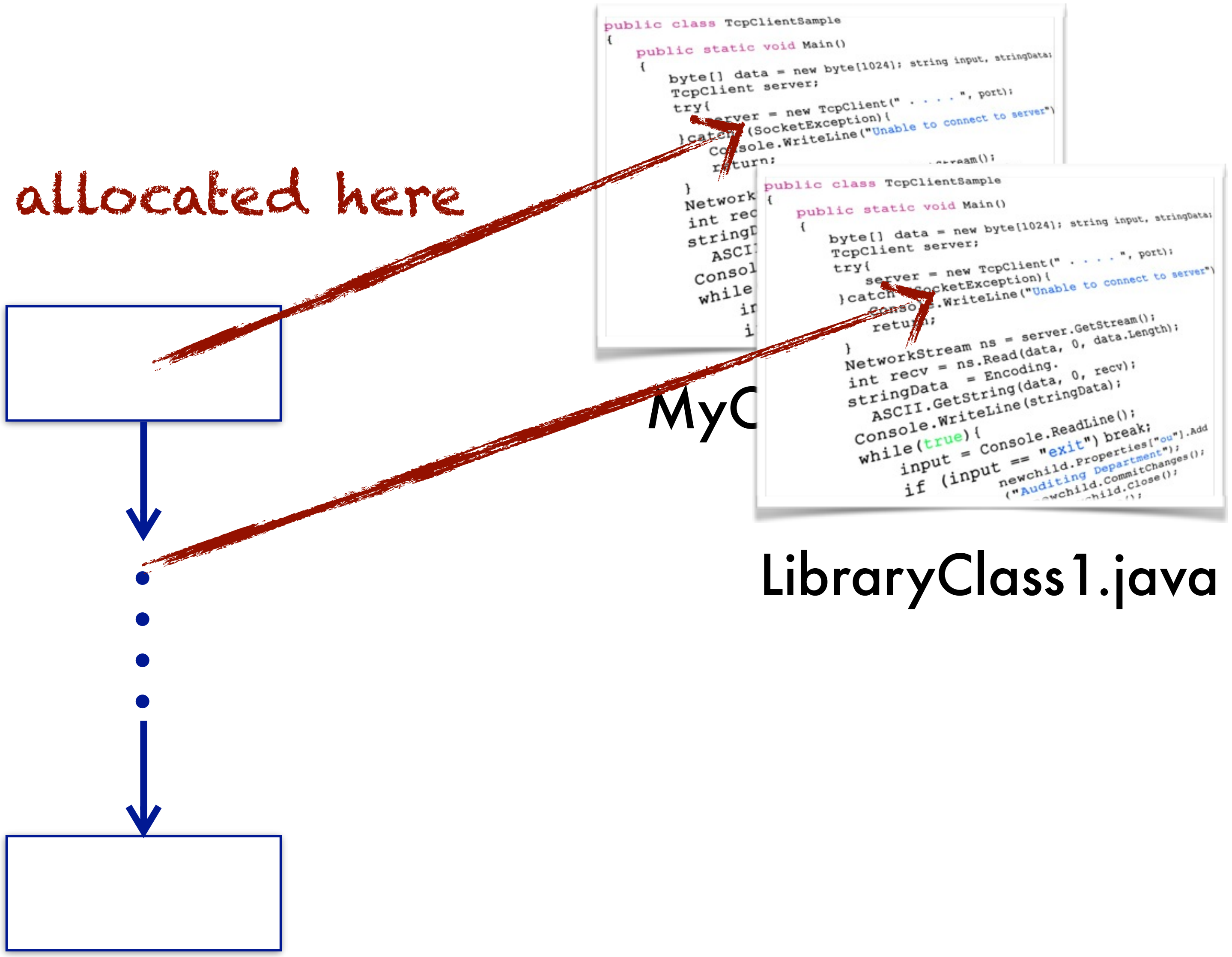
allocated here

```
public class TcpClientSample
{
    public static void Main()
    {
        byte[] data = new byte[1024]; string input, stringData;
        TcpClient server;
        try{
            server = new TcpClient(" . . . . ", port);
        }catch (SocketException){
            Console.WriteLine("Unable to connect to server")
            return;
        }
        NetworkStream ns = server.GetStream();
        int recv = ns.Read(data, 0, data.Length);
        stringData   = Encoding.
            ASCII.GetString(data, 0, recv);
        Console.WriteLine(stringData);
        while(true){
            input = Console.ReadLine();
            if (input == "exit") break;
                newchild.Properties["ou"].Add
                ("Auditing Department");
                newchild.CommitChanges();
                newchild.Close();
```

MyClass1.java

Manual triage for heap reachability reports





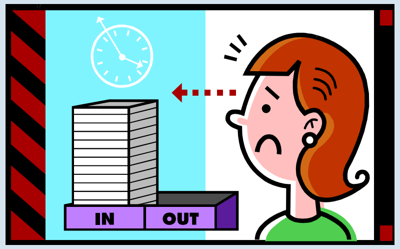
```
public class TcpClientSample
{
    public static void Main()
    {
        byte[] data = new byte[1024]; string input, stringData;
        TcpClient server;
        try{
            server = new TcpClient(" . . . . ", port);
        }catch (SocketException){
            Console.WriteLine("Unable to connect to server")
            return;
        }
        Network
        int rec
        stringD
          ASCI
        Consol
        while
            in
            i
```

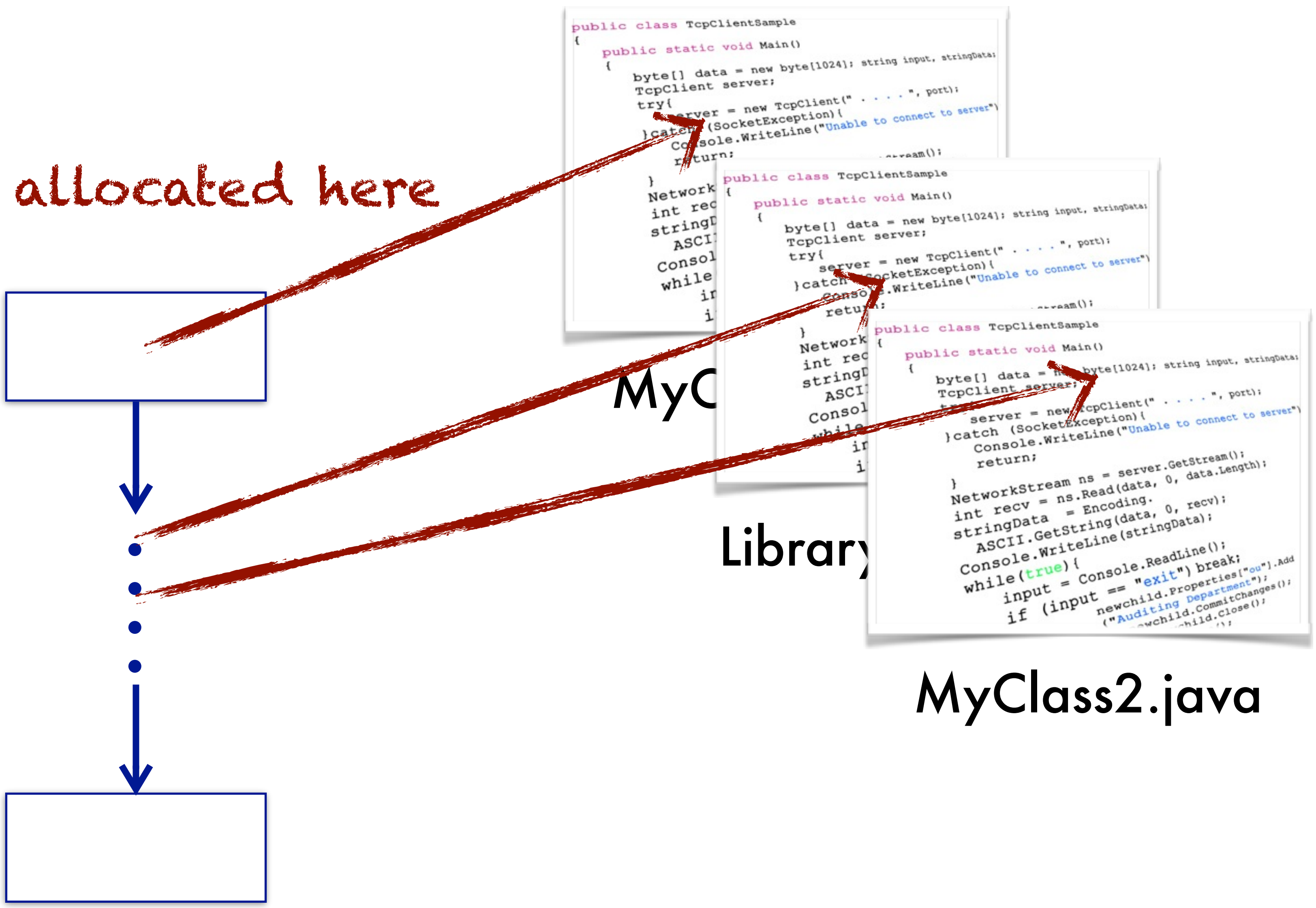
MyC

```
public class TcpClientSample
{
    public static void Main()
    {
        byte[] data = new byte[1024]; string input, stringData;
        TcpClient server;
        try{
            server = new TcpClient(" . . . . ", port);
        }catch (SocketException){
            Console.WriteLine("Unable to connect to server")
            return;
        }
        NetworkStream ns = server.GetStream();
        int recv = ns.Read(data, 0, data.Length);
        stringData  = Encoding.
          ASCII.GetString(data, 0, recv);
        Console.WriteLine(stringData);
        while(true){
            input = Console.ReadLine();
            if (input == "exit") break;
                newchild.Properties["ou"].Add
                ("Auditing Department");
                newchild.CommitChanges();
                newchild.close();
```

LibraryClass1.java

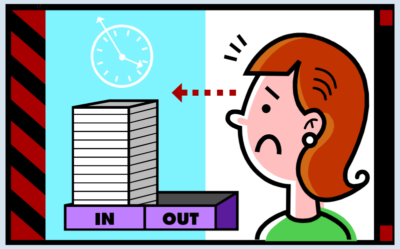
Manual triage for heap reachability reports

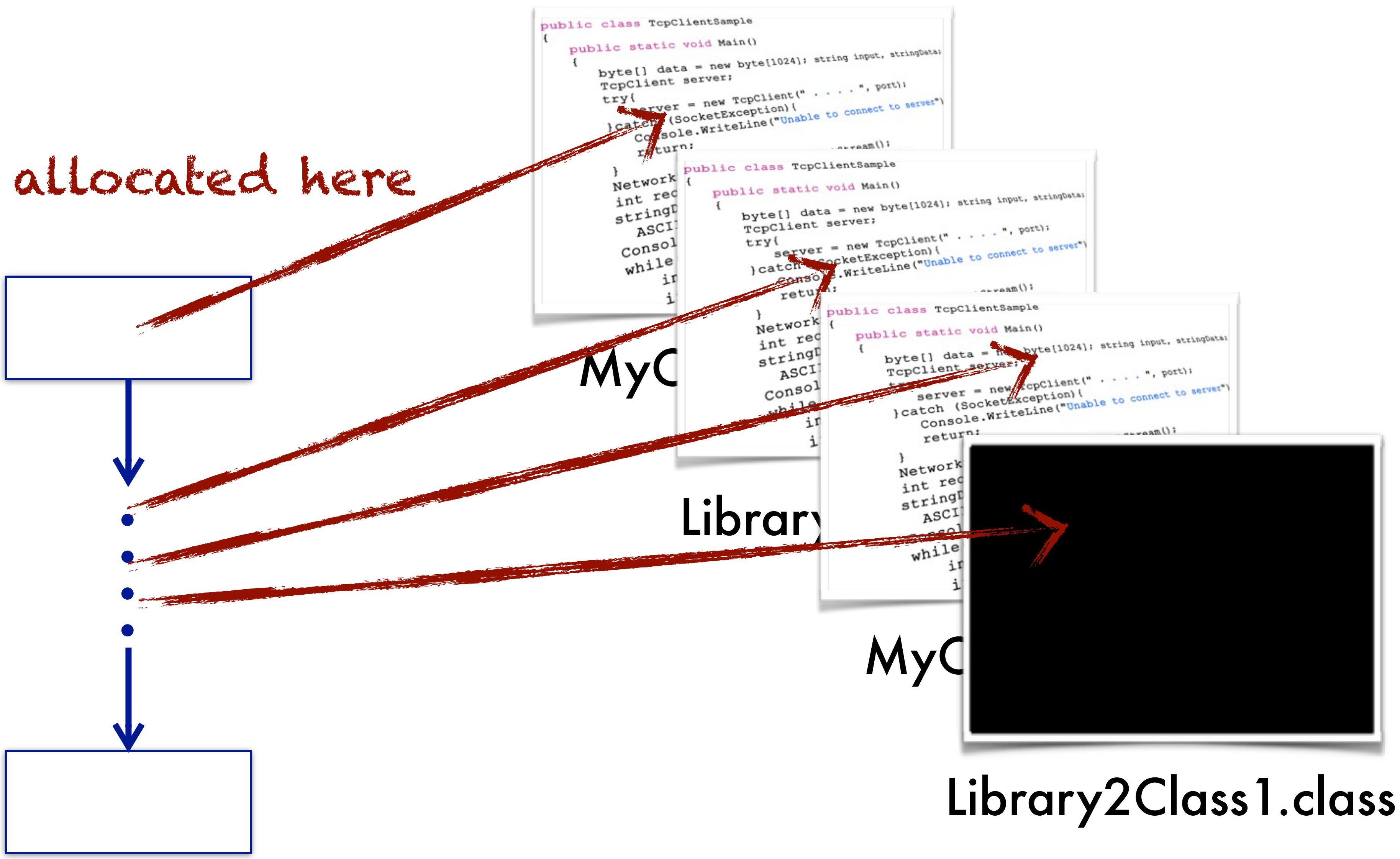




```
public class TcpClientSample
{
    public static void Main()
    {
        byte[] data = new byte[1024]; string input, stringData;
        TcpClient server;
        try{
            server = new TcpClient(" . . . .", port);
        }catch (SocketException){
            Console.WriteLine("Unable to connect to server")
            return;
        }
        NetworkStream ns = server.GetStream();
        int recv = ns.Read(data, 0, data.Length);
        stringData  = Encoding.
            ASCII.GetString(data, 0, recv);
        Console.WriteLine(stringData);
        while(true){
            input = Console.ReadLine();
            if (input == "exit") break;
                newchild.Properties["ou"].Add
                ("Auditing Department");
```

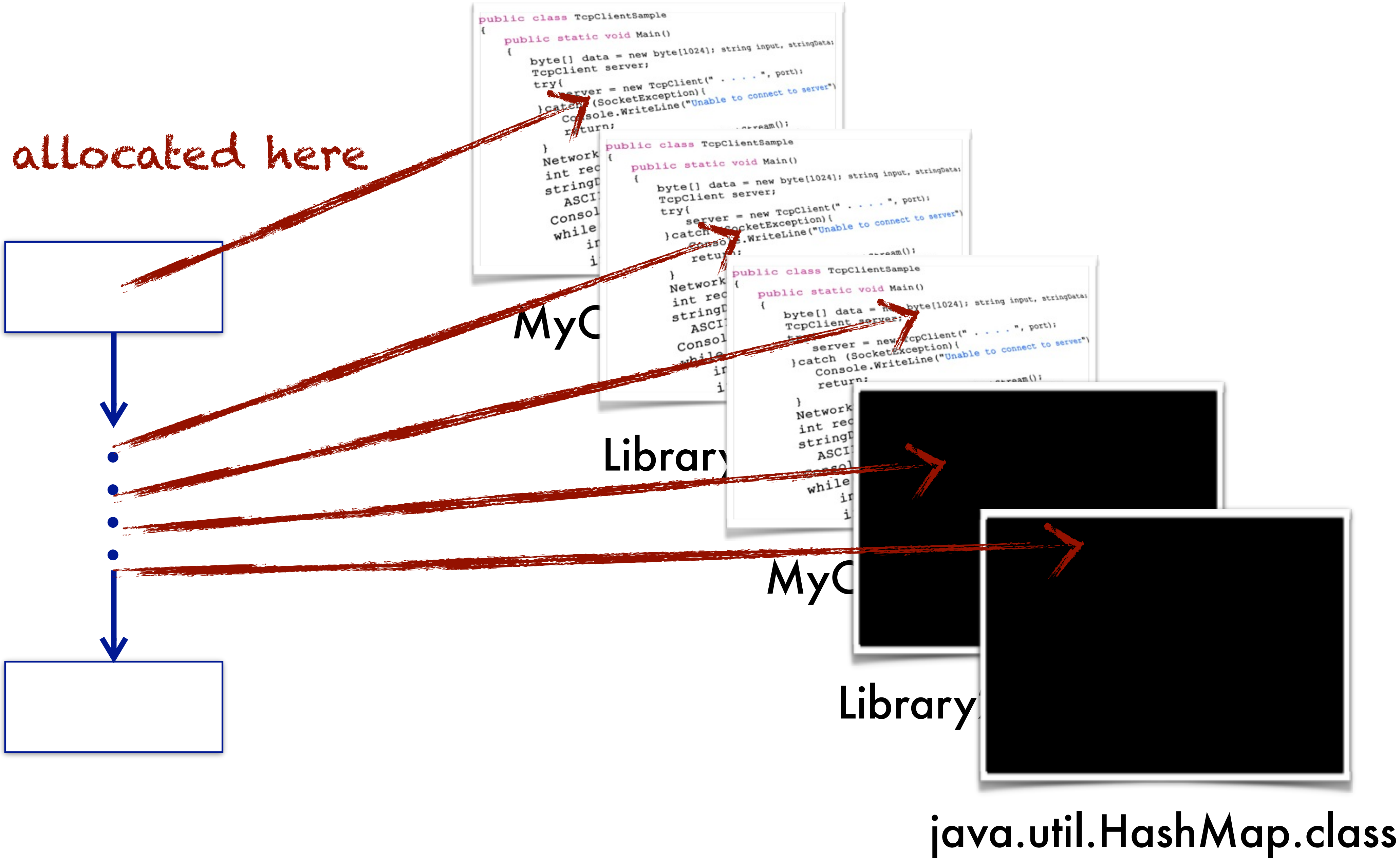
Manual triage for heap reachability reports





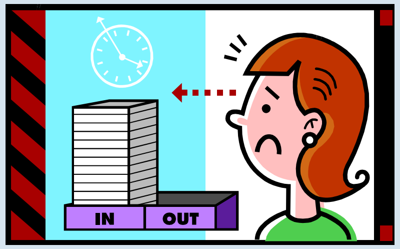
```
public class TcpClientSample
{
    public static void Main()
    {
        byte[] data = new byte[1024]; string input, stringData;
        TcpClient server;
        try{
            server = new TcpClient(" . . . . ", port);
        }catch (SocketException){
            Console.WriteLine("Unable to connect to server")
            return;
                                            ...Stream();
        }
        Network
        int rec
        stringD
          ASCI
        Consol
        while
            in
            i
```

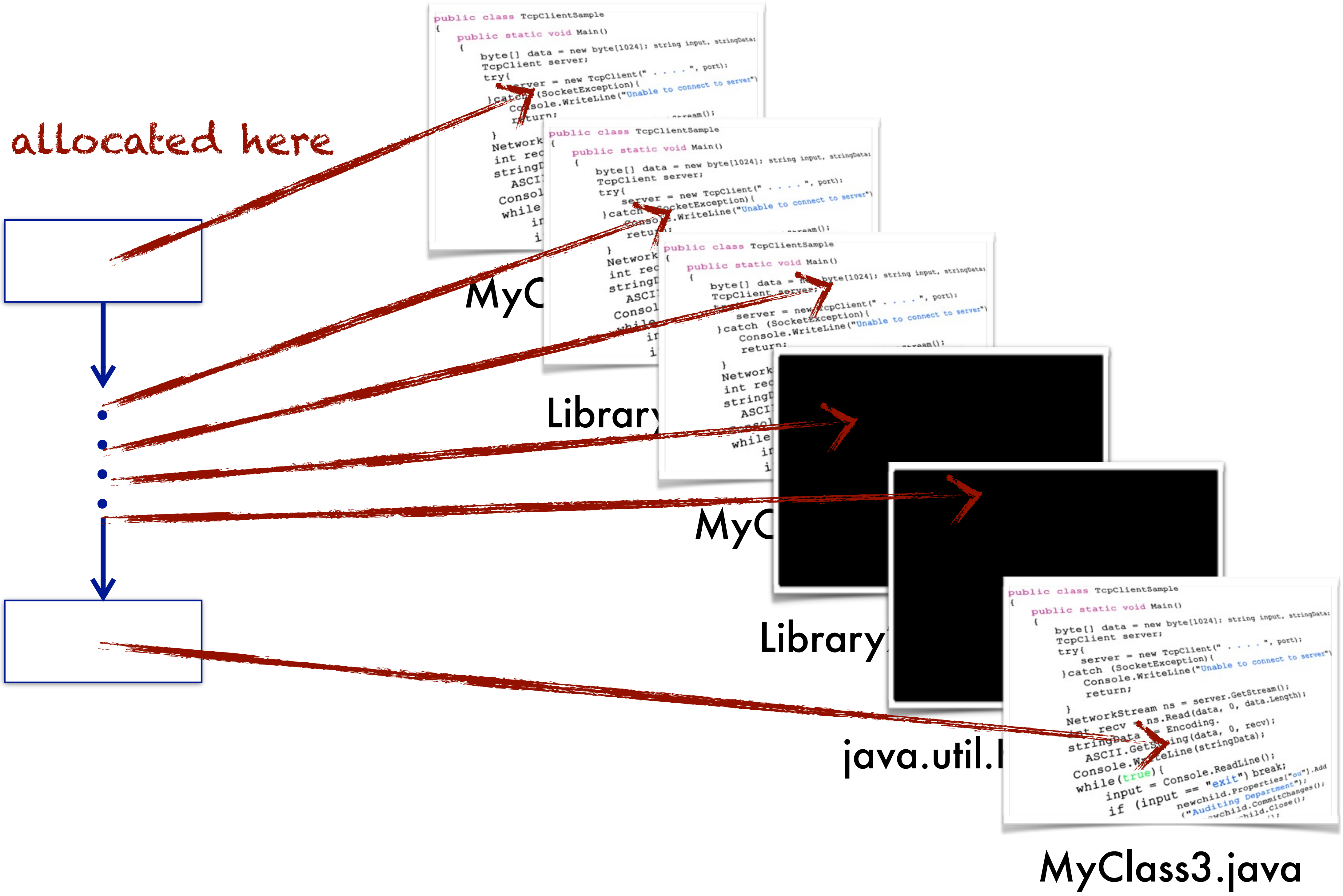
Manual triage for heap reachability reports



```
public class TcpClientSample
{
    public static void Main()
    {
        byte[] data = new byte[1024]; string input, stringData;
        TcpClient server;
        try{
            server = new TcpClient(" . . . . ", port);
        }catch (SocketException){
            Console.WriteLine("Unable to connect to server")
            return;
                                              ...Stream();
        }
        Network
        int rec
        stringD
          ASCI
        Consol
        while
            in
            i
```

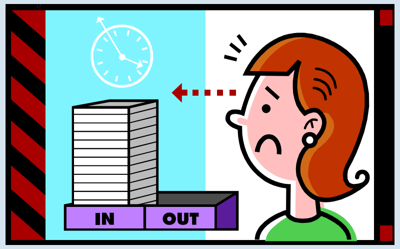
Manual triage for heap reachability reports

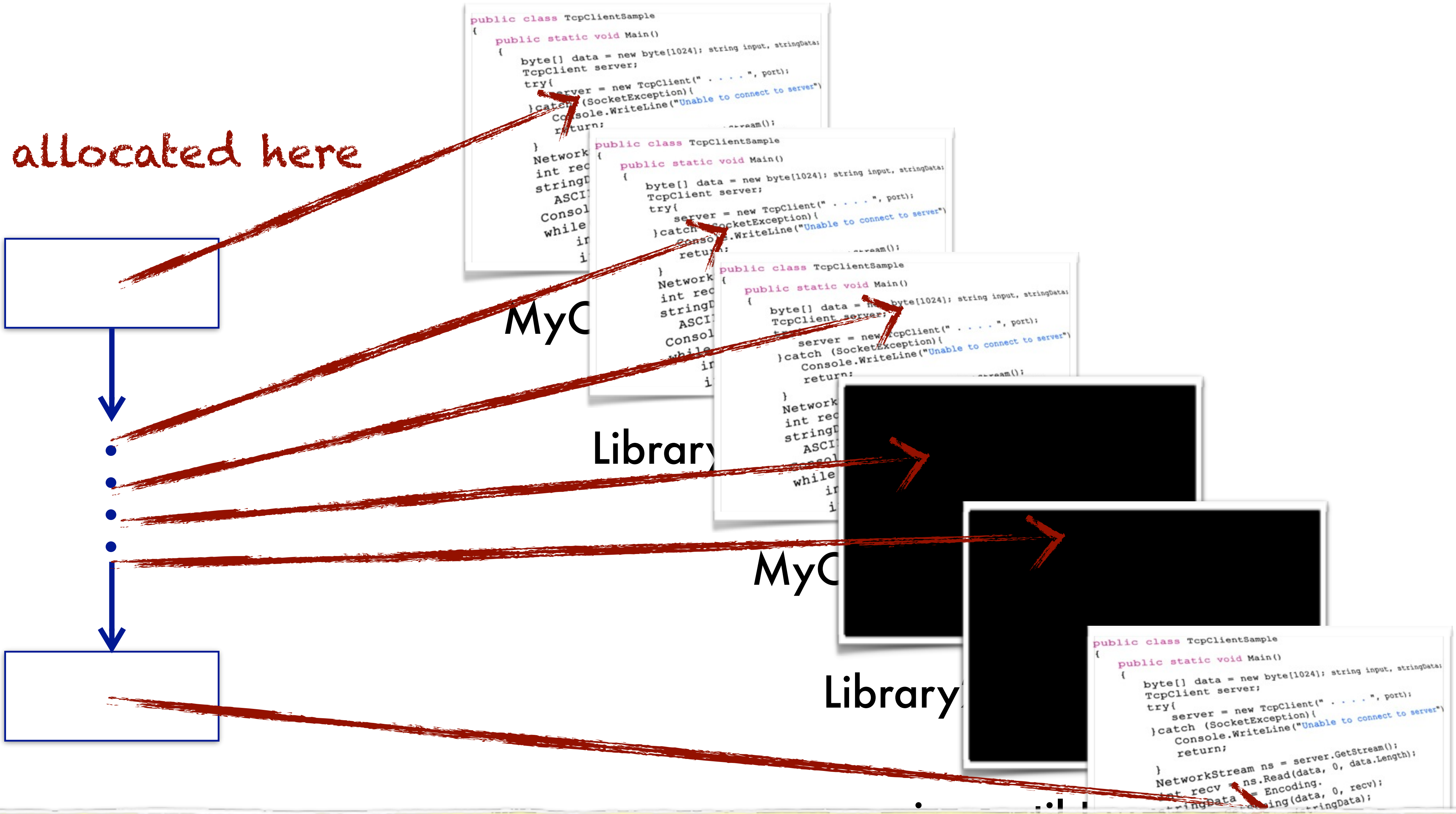




```
public class TcpClientSample
{
    public static void Main()
    {
        byte[] data = new byte[1024]; string input, stringData;
        TcpClient server;
        try{
            server = new TcpClient(" . . . . ", port);
        }catch (SocketException){
            Console.WriteLine("Unable to connect to server")
            return;
        }
        NetworkStream ns = server.GetStream();
        int recv = ns.Read(data, 0, data.Length);
        stringData = Encoding.
          ASCII.GetString(data, 0, recv);
        Console.WriteLine(stringData);
        while(true){
            input = Console.ReadLine();
            if (input == "exit") break;
                newchild.Properties["ou"].Add
                ("Auditing Department");
                newchild.CommitChanges();
                newchild.Close();
```

Manual triage for heap reachability reports

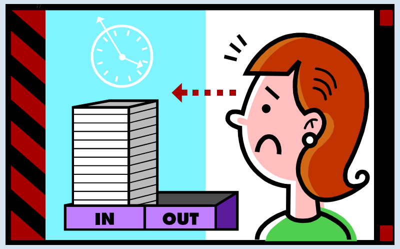




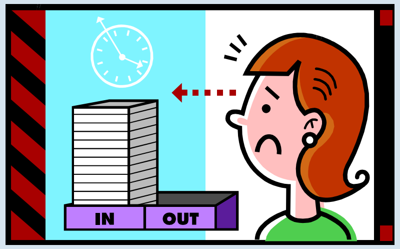
```
public class TcpClientSample
{
    public static void Main()
    {
        byte[] data = new byte[1024]; string input, stringData;
        TcpClient server;
        try{
            server = new TcpClient(" . . . . ", port);
        }catch (SocketException){
            Console.WriteLine("Unable to connect to server")
            return;
        }
        NetworkStream ns = server.GetStream();
        int recv = ns.Read(data, 0, data.Length);
        stringData = Encoding.
            ASCII.GetString(data, 0, recv);
        Console.WriteLine(stringData);
        while
```

Get abstract heap path + allocation sites

Assuming the user starts to triage an alarm ...

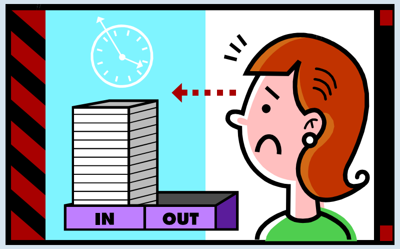


Assuming the user starts to triage an alarm …



What does the user need to do? He starts at, say, line 142 and traces back to see if a bug is possible given what's happening.

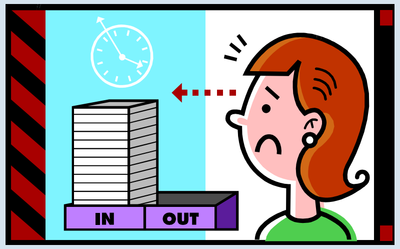
Assuming the user starts to triage an alarm ...



What does the user need to do? He starts at, say, line 142 and traces back to see if a bug is possible given what's happening.

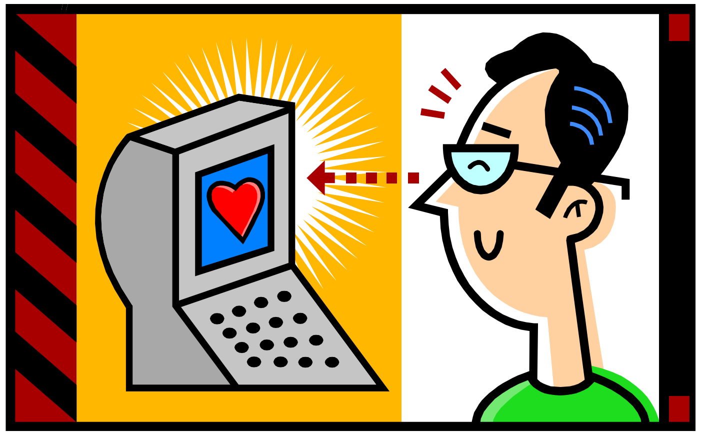
We can do this with analysis (V+MC+AI)!

Assuming the user starts to triage an alarm …



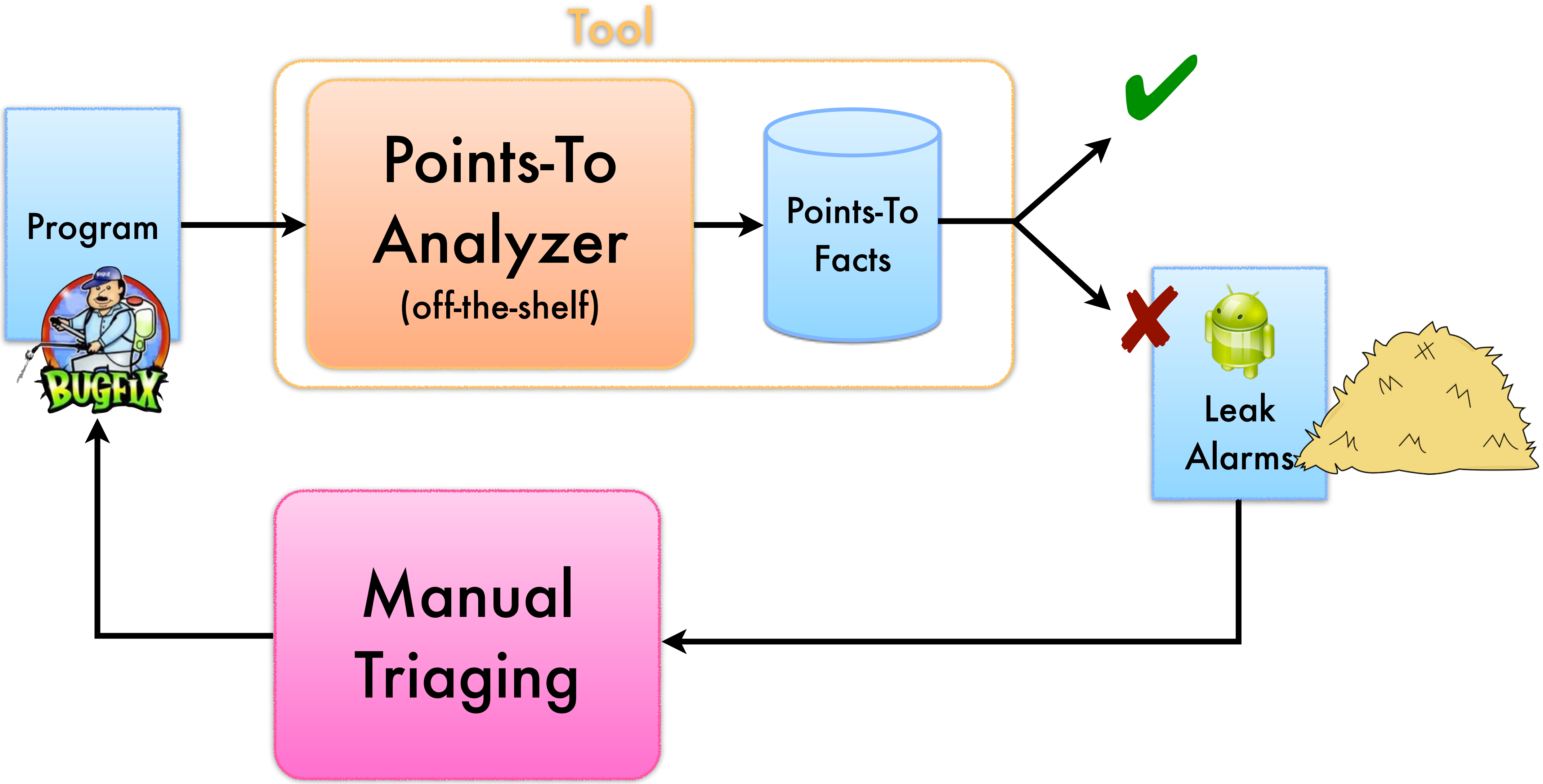
What does the user need to do? He starts at, say, line 142 and traces back to see if a bug is possible given what's happening.

We can do this with analysis (V+MC+AI)!

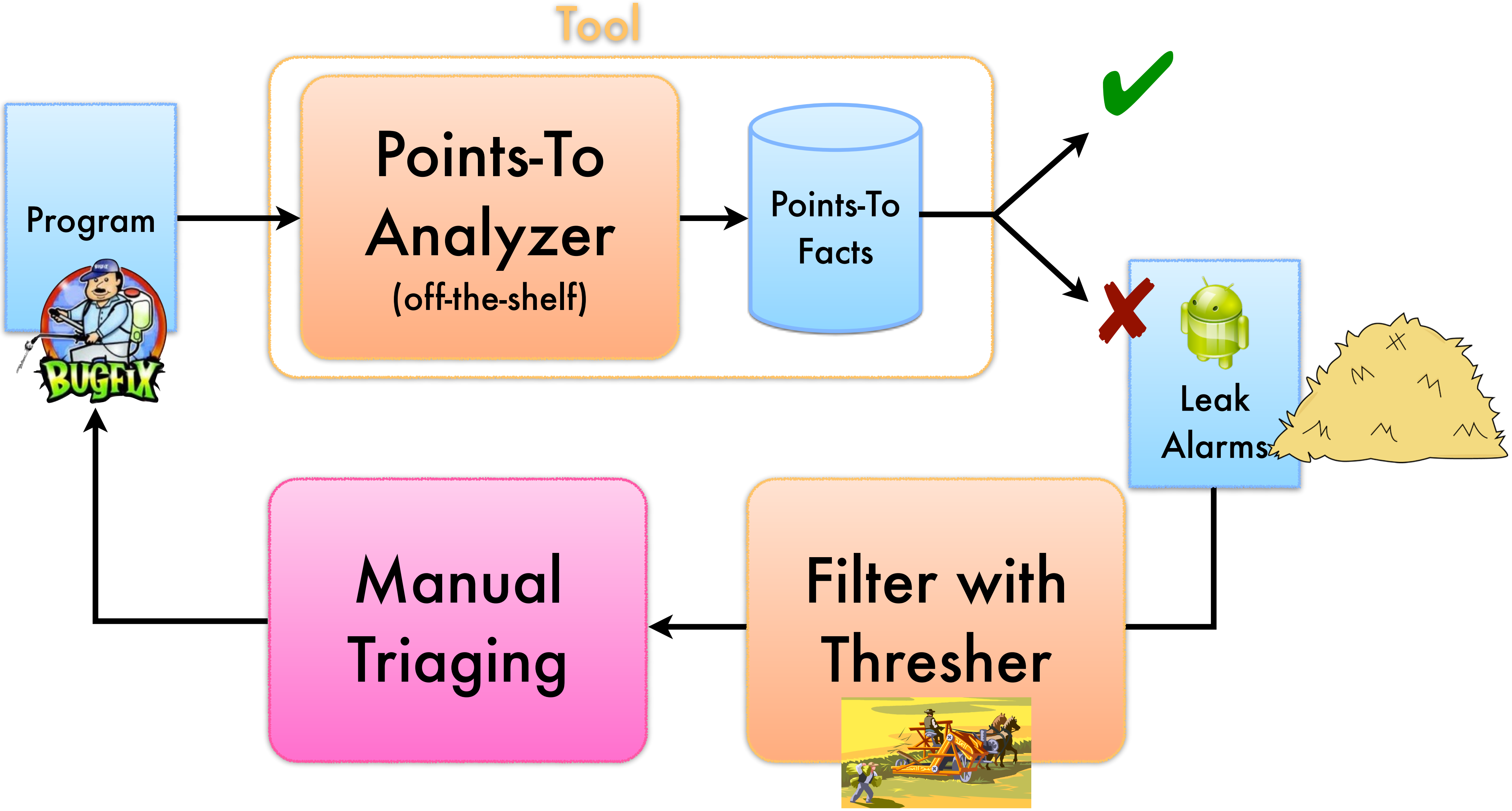


If we filter most false alarms, the user can triage more quickly and get to true bugs earlier (without frustration).

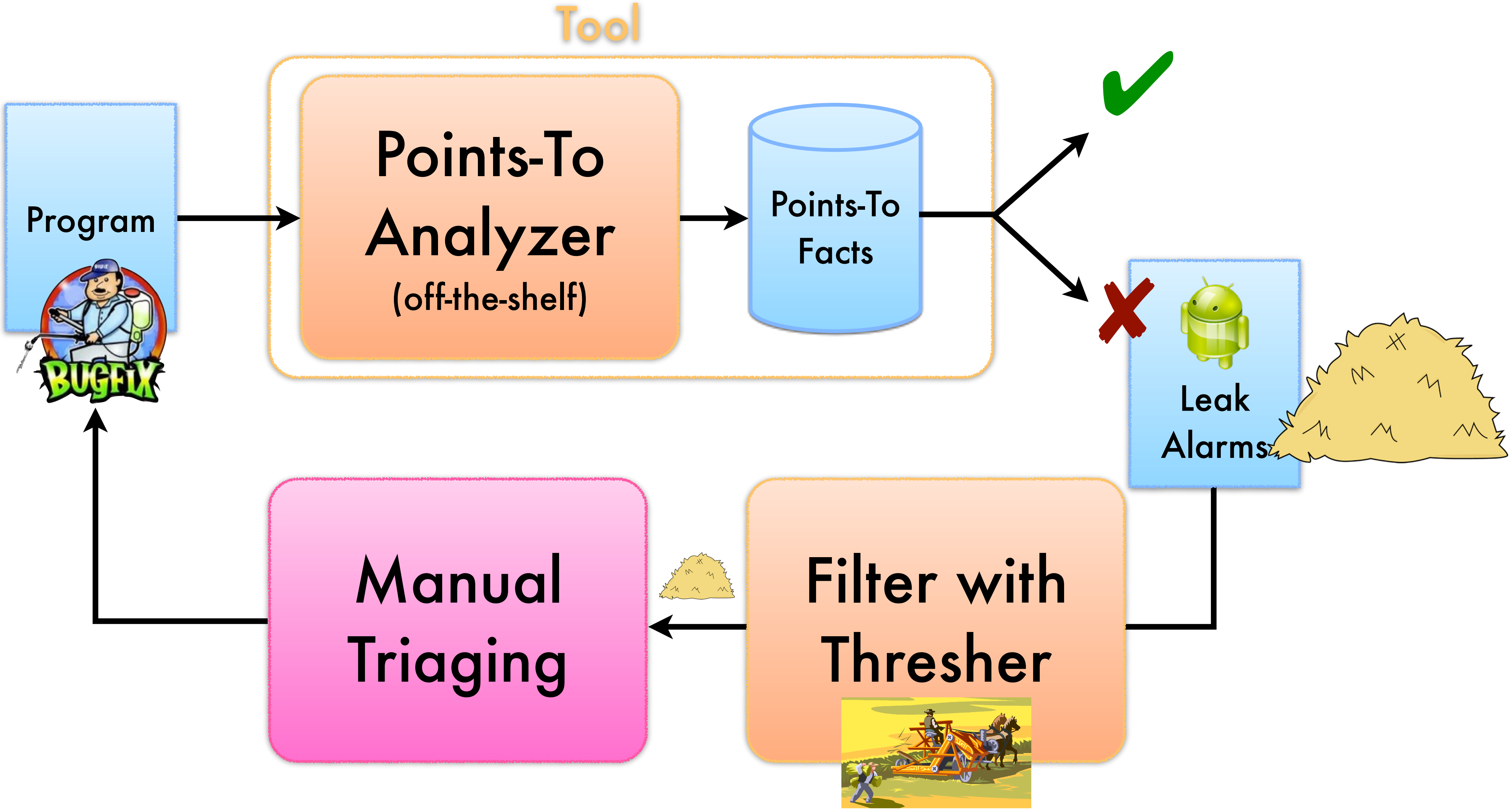
Roadmap: Thresher **filters** out false alarms by refuting them one-by-one.



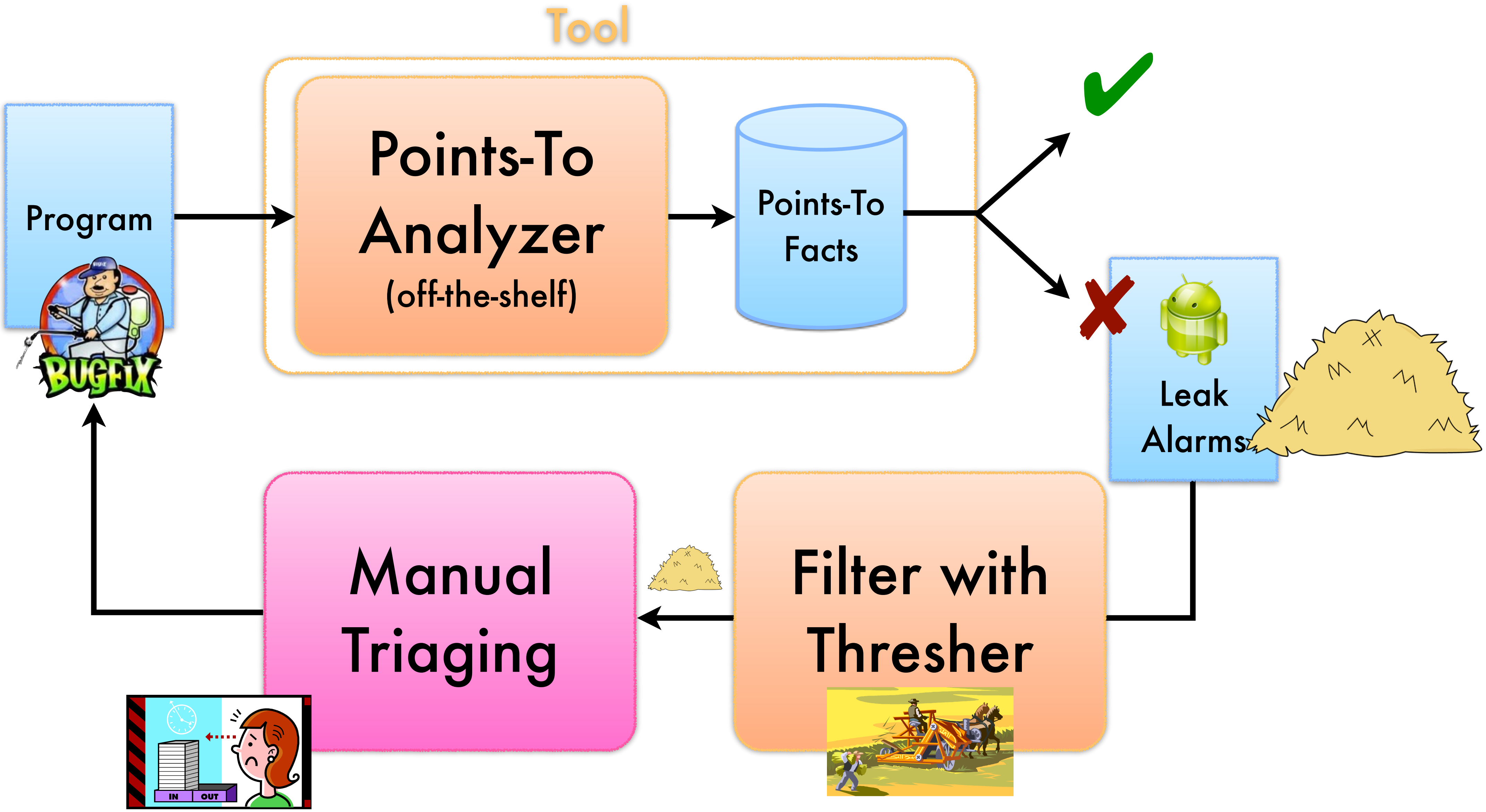
Roadmap: Thresher **filters** out false alarms by refuting them one-by-one.



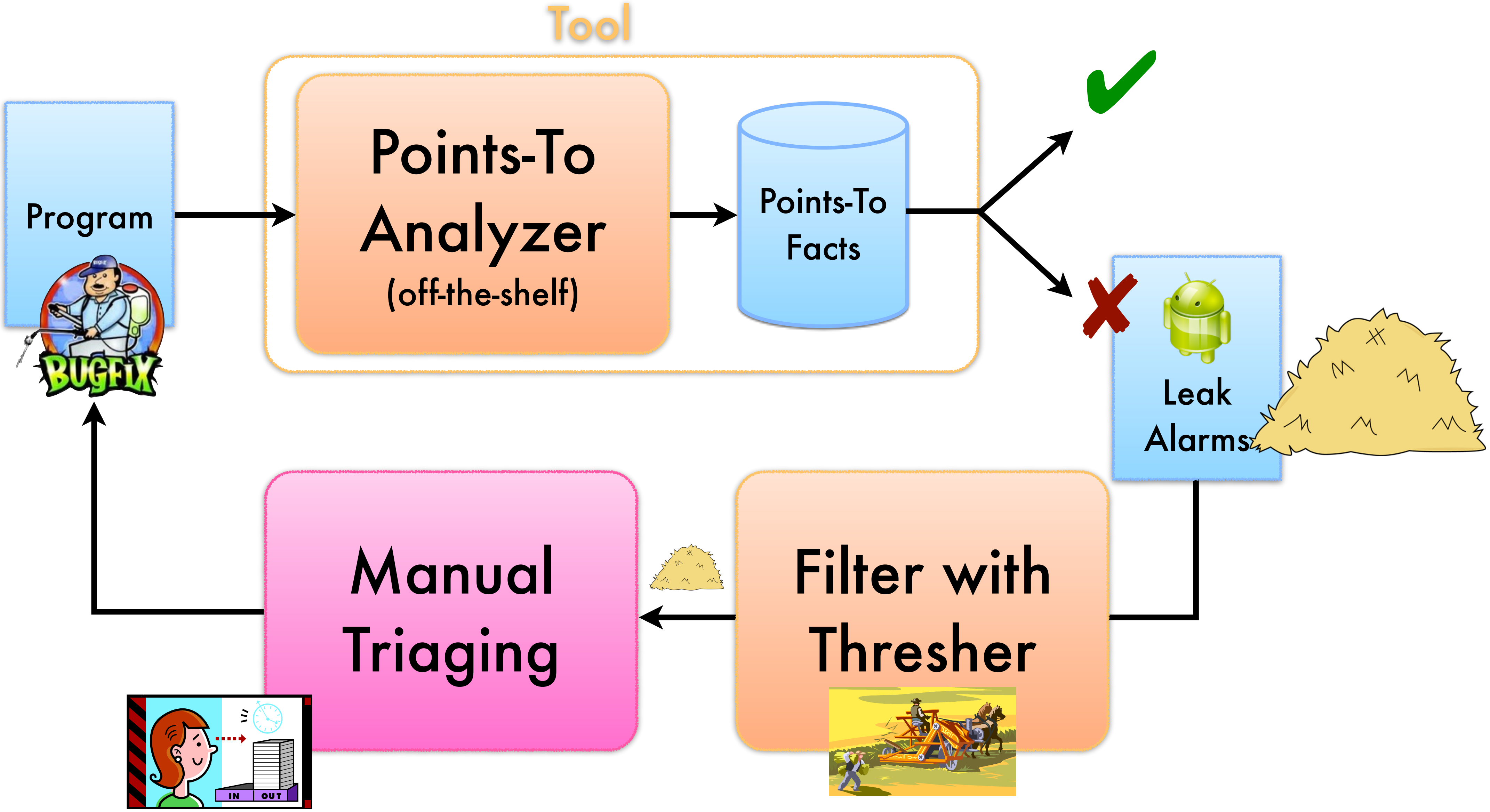
Roadmap: Thresher **filters** out false alarms by refuting them one-by-one.



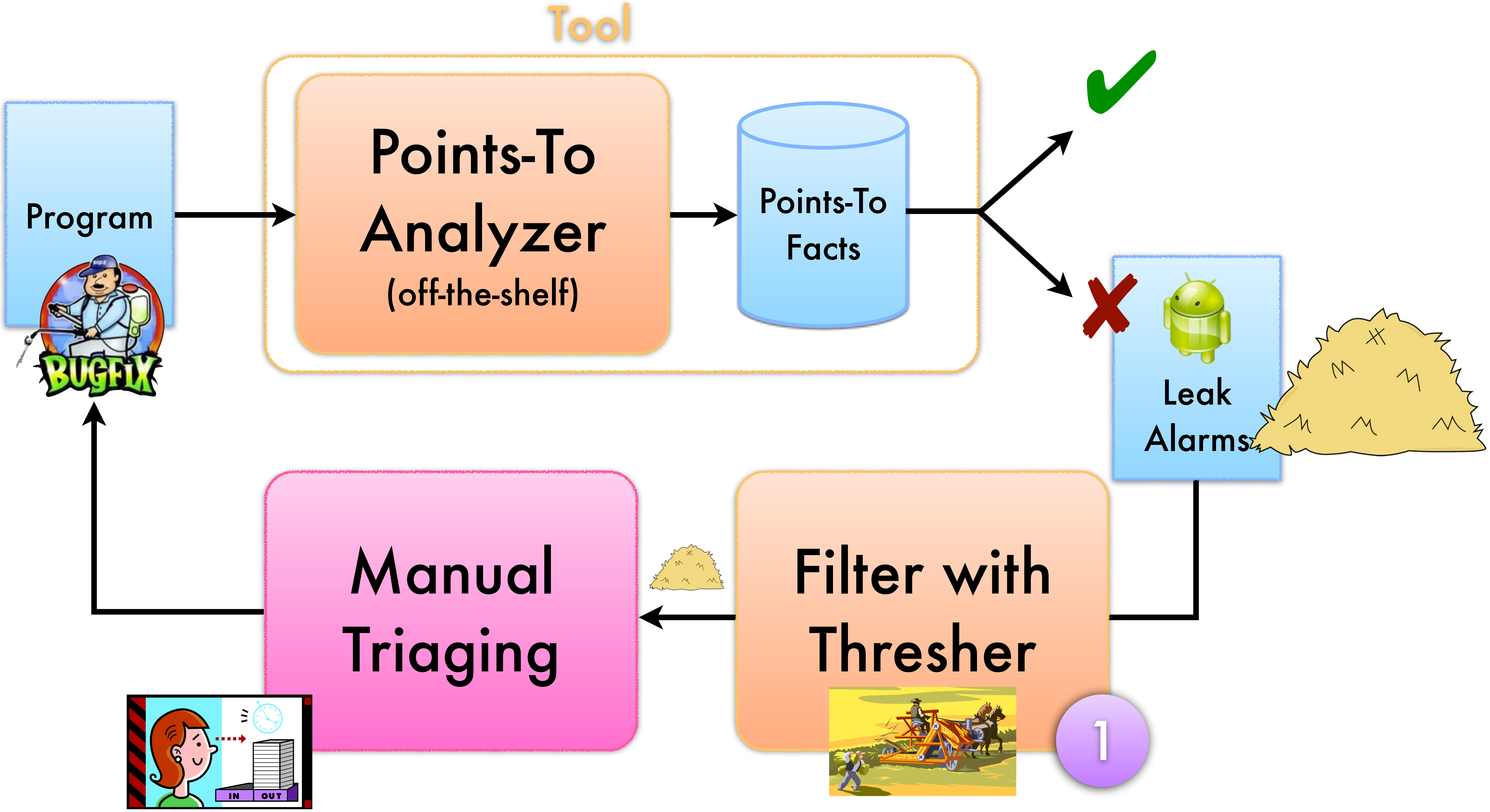
Roadmap: Thresher **filters** out false alarms by refuting them one-by-one.



Roadmap: Thresher **filters** out false alarms by refuting them one-by-one.

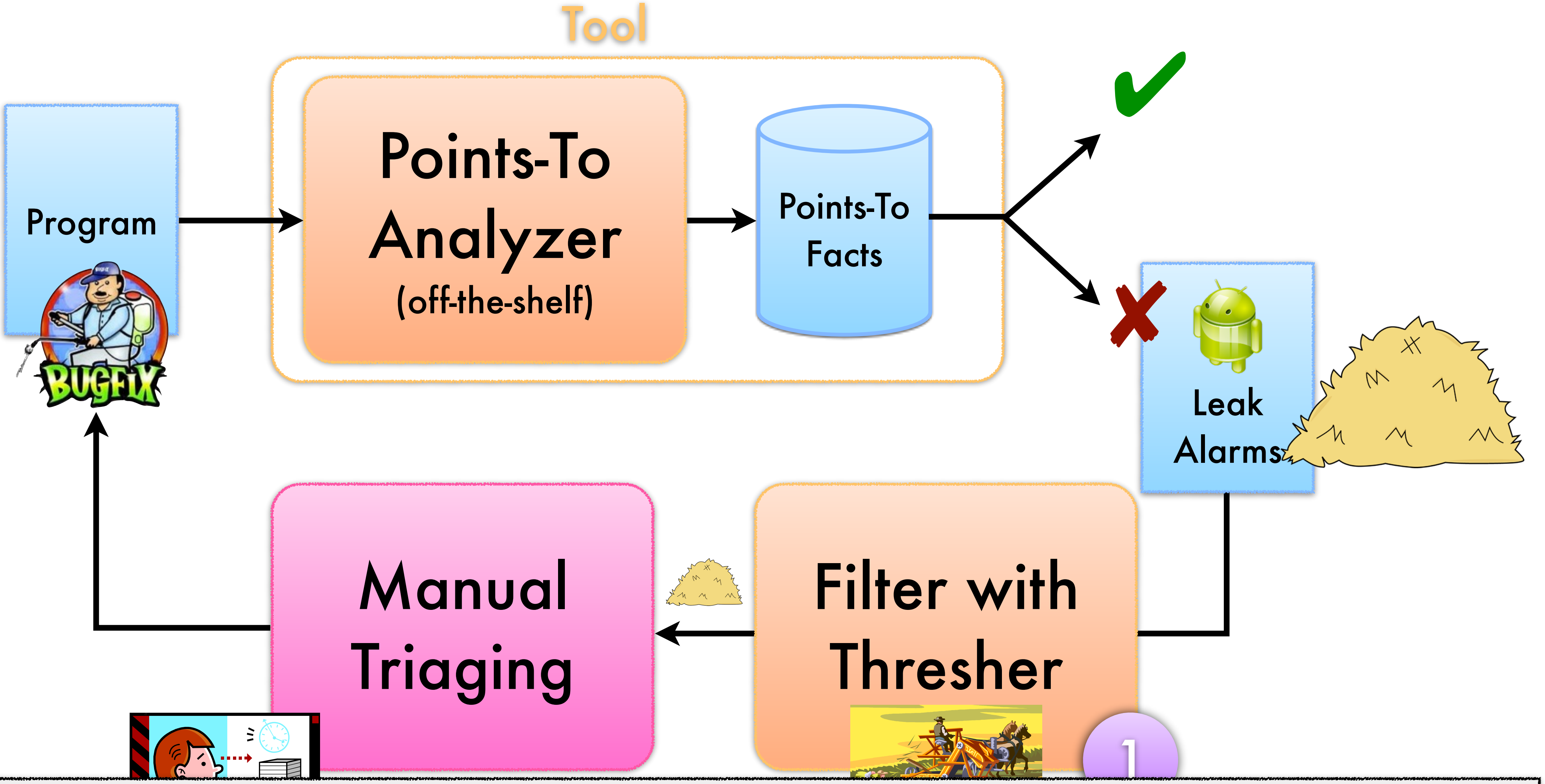


Roadmap: Thresher filters out false alarms by refuting them one-by-one.



Idea 1: Refute points-to on-demand with second precise "filter" analysis

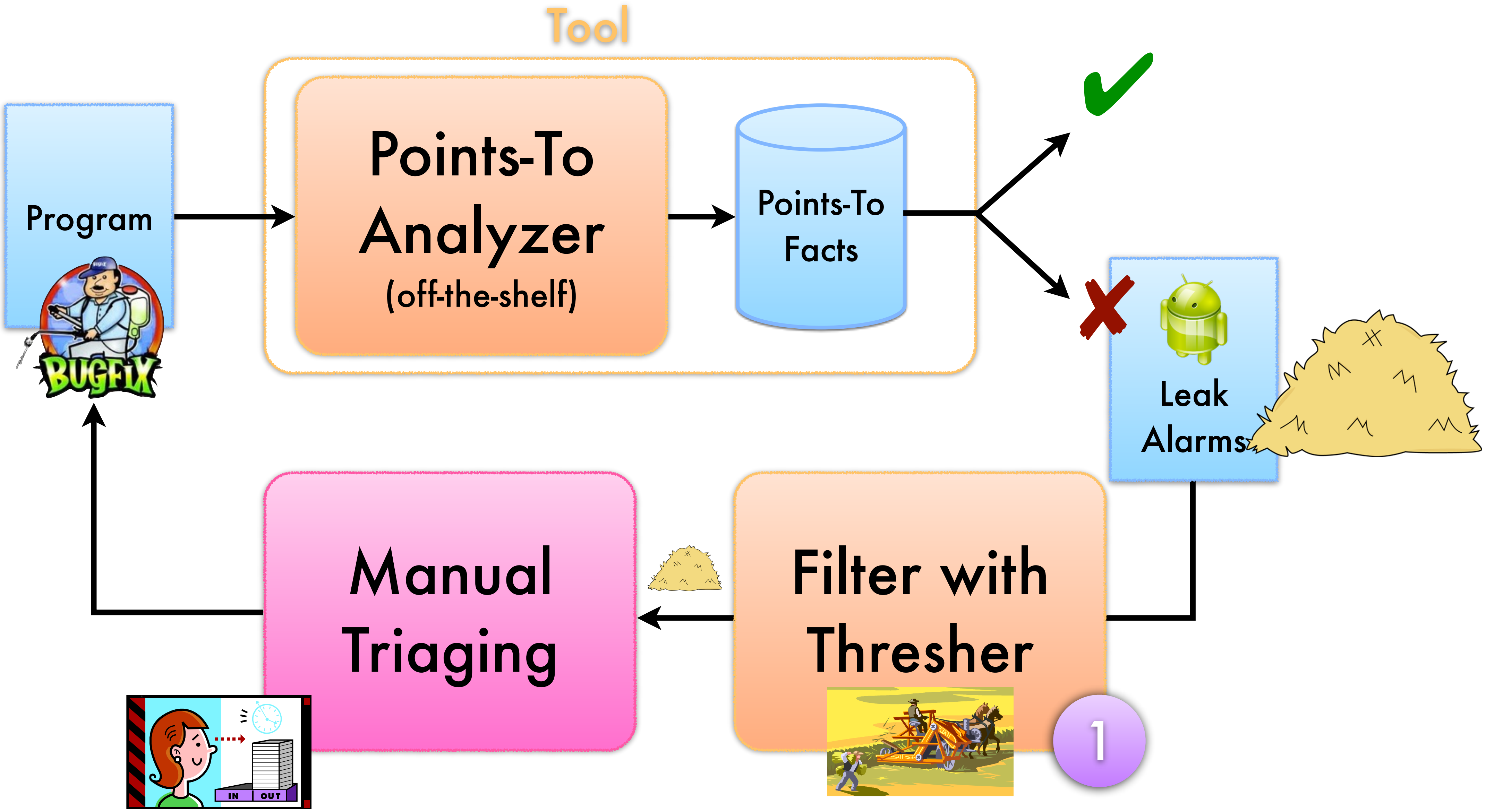
Roadmap: Thresher **filters** out false alarms by refuting them one-by-one.



*-sensitive, strong updates (separation logic) but over-approximate

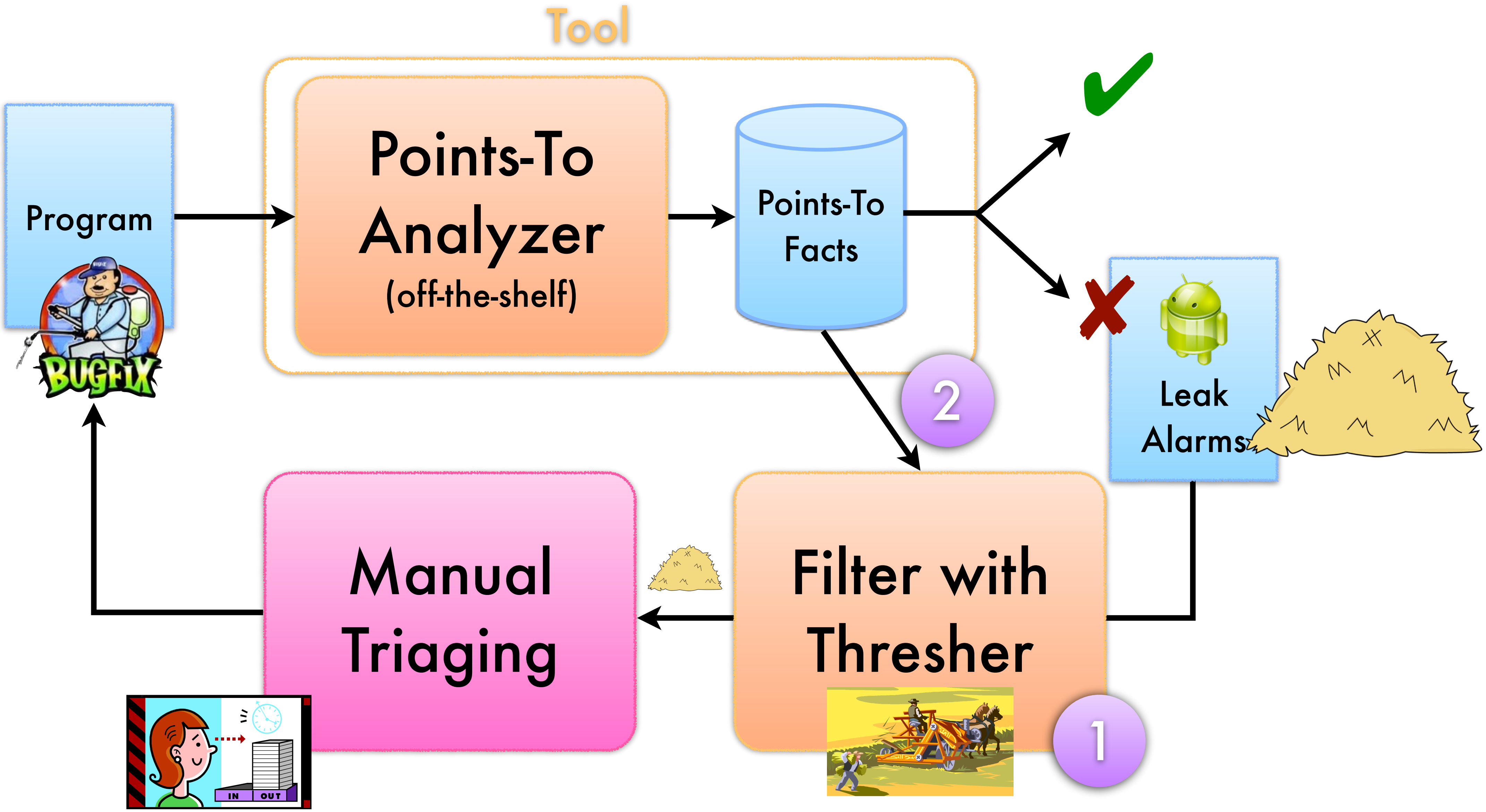
Idea 1: Refute points-to on-demand with second precise “filter” analysis

Roadmap: Thresher filters out false alarms by refuting them one-by-one.



Idea 1: Refute points-to on-demand with second precise "filter" analysis

Roadmap: Thresher **filters** out false alarms by refuting them one-by-one.



Tool

Program

Points-To Analyzer (off-the-shelf)

Points-To Facts

✓

✗

Leak Alarms

2

Manual Triaging

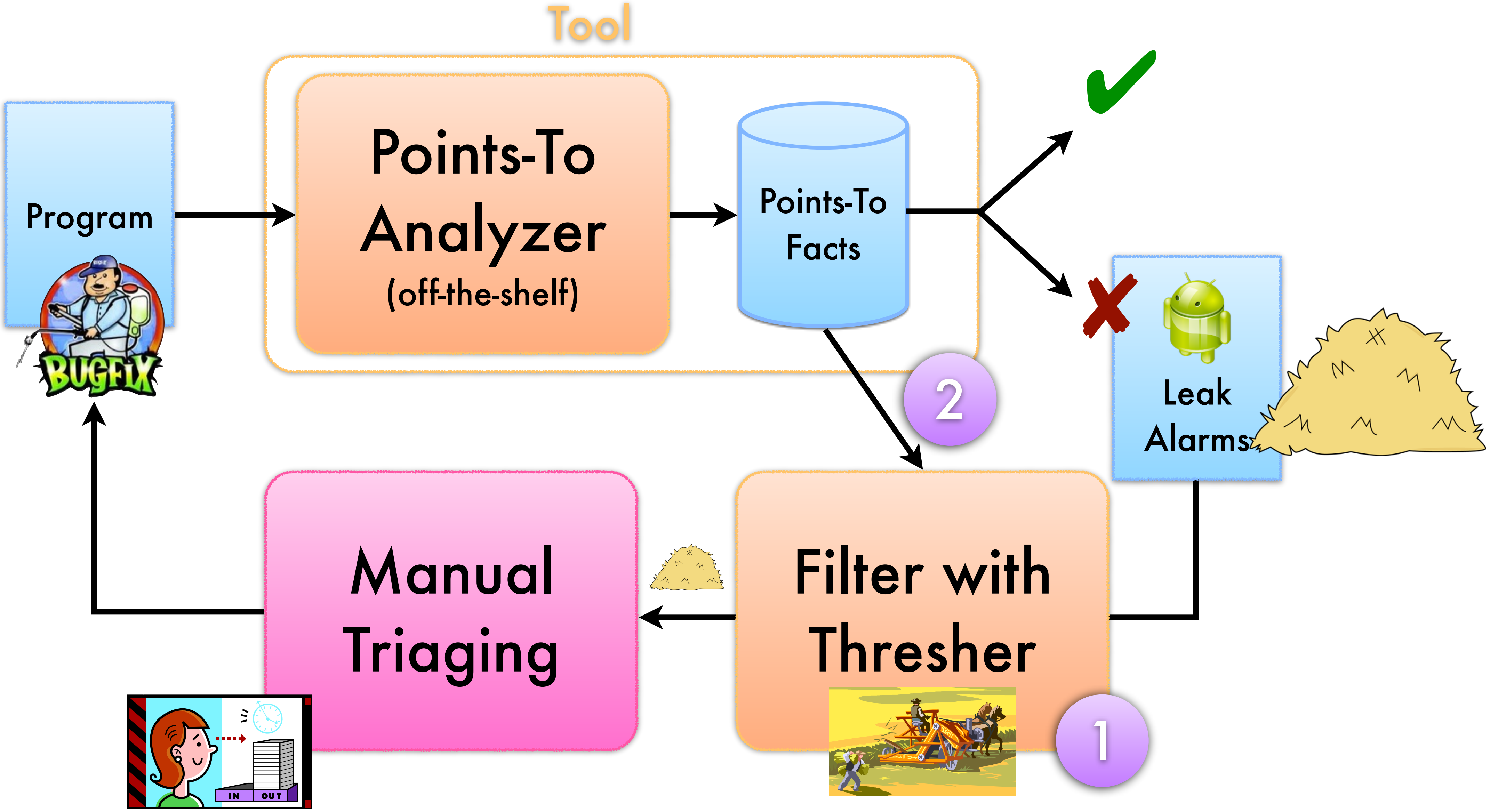
Filter with Thresher

1

Idea 1: Refute points-to on-demand with second precise “filter” analysis

Idea 2: Leverage the facts from the first analysis in the filter analysis to scale

Roadmap: Thresher **filters** out false alarms by refuting them one-by-one.

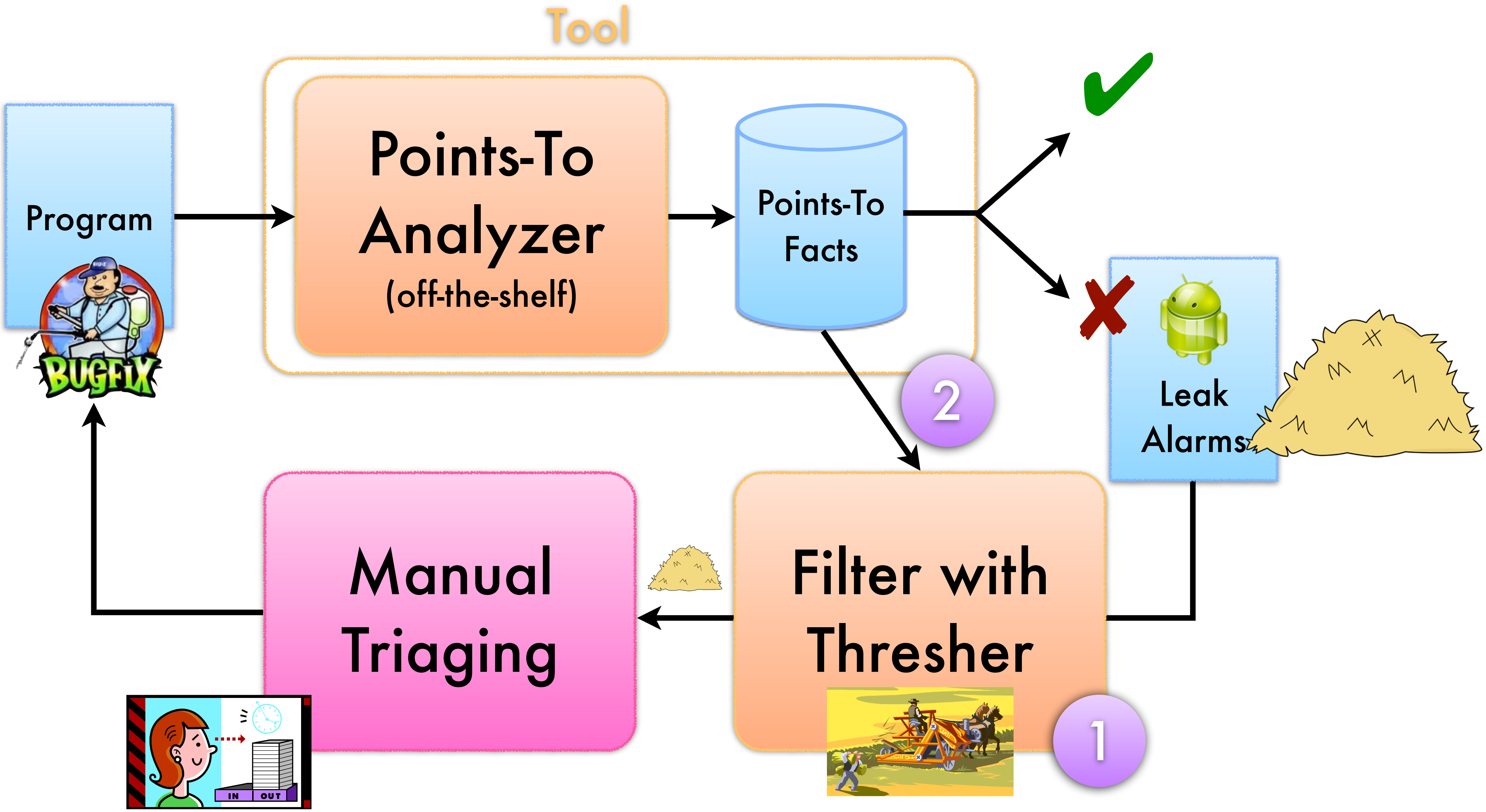


Idea 1: Refute points-to on-demand with second precise "filter" analysis

"**from** constraints" to reduce with the points-to domain

Idea 2: Leverage the facts from the first analysis in the filter analysis to scale

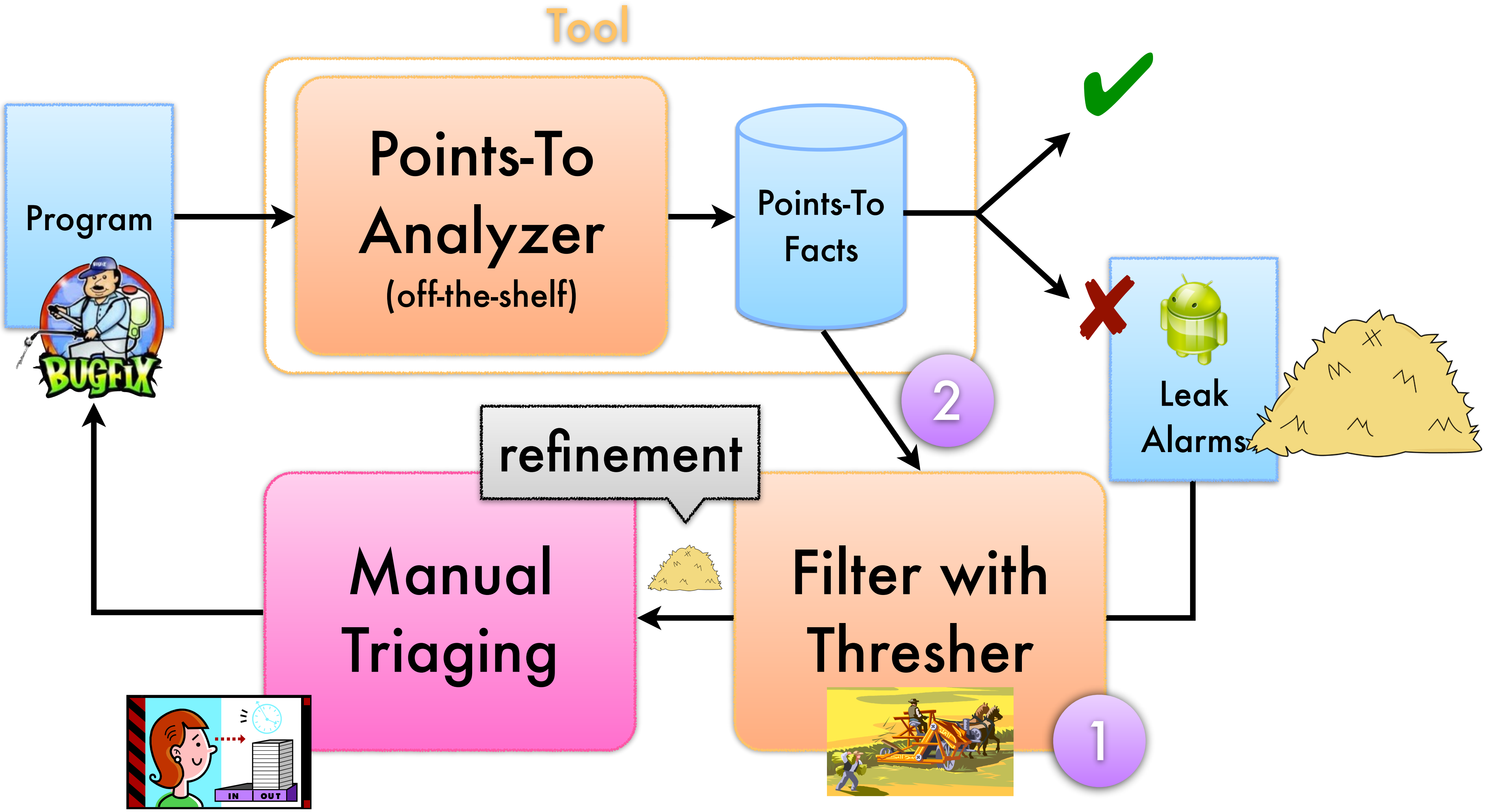
Roadmap: Thresher filters out false alarms by refuting them one-by-one.



Idea 1: Refute points-to on-demand with second precise "filter" analysis

Idea 2: Leverage the facts from the first analysis in the filter analysis to scale

Roadmap: Thresher **filters** out false alarms by refuting them one-by-one.

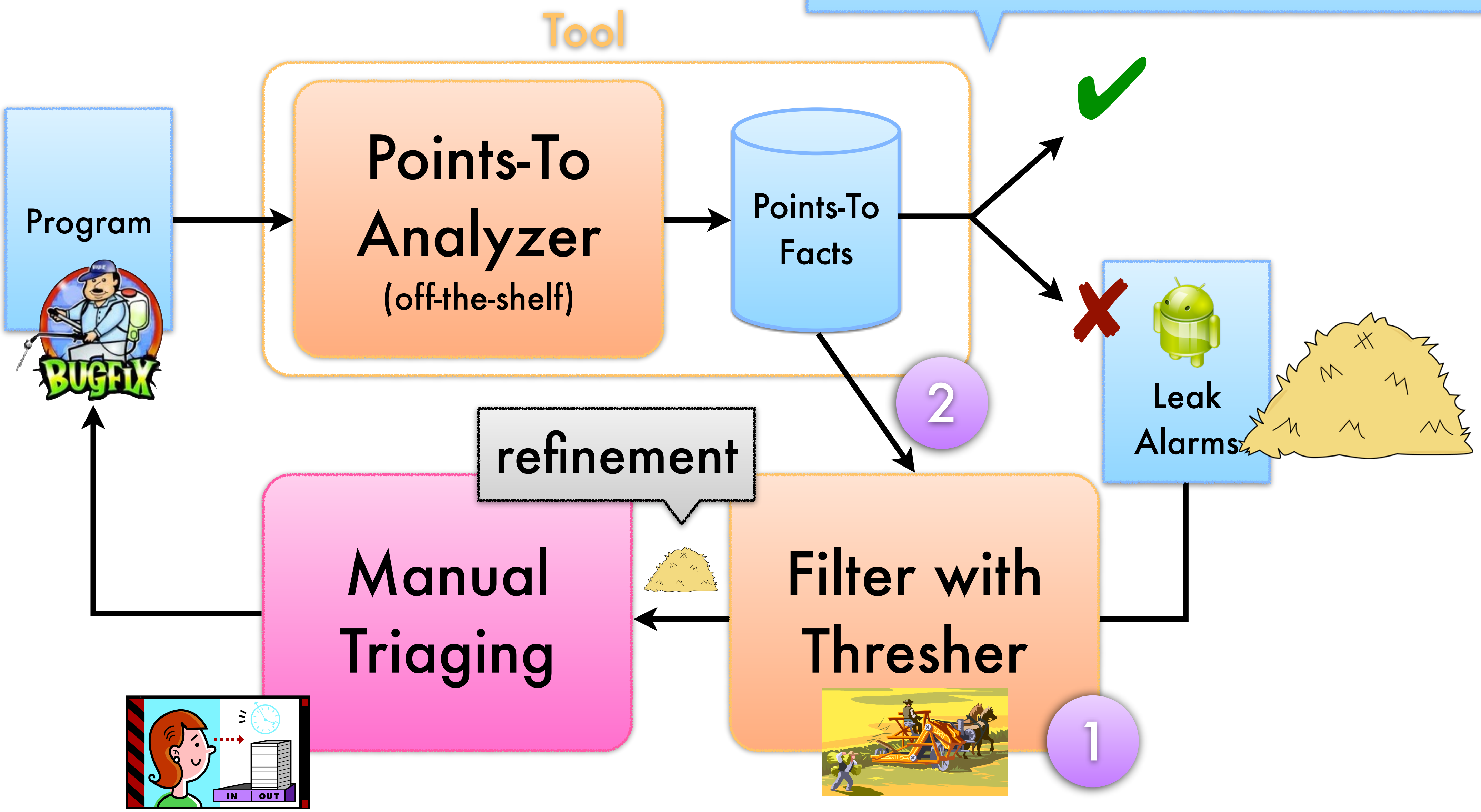


Idea 1: Refute points-to on-demand with second precise "filter" analysis

Idea 2: Leverage the facts from the first analysis in the filter analysis to scale

Roadmap: Thresher filters out false a refuting them one-by-one.

Related: Staged analysis
[Fink, Yahav, Dor, Ramalingam, Geay (2006)]



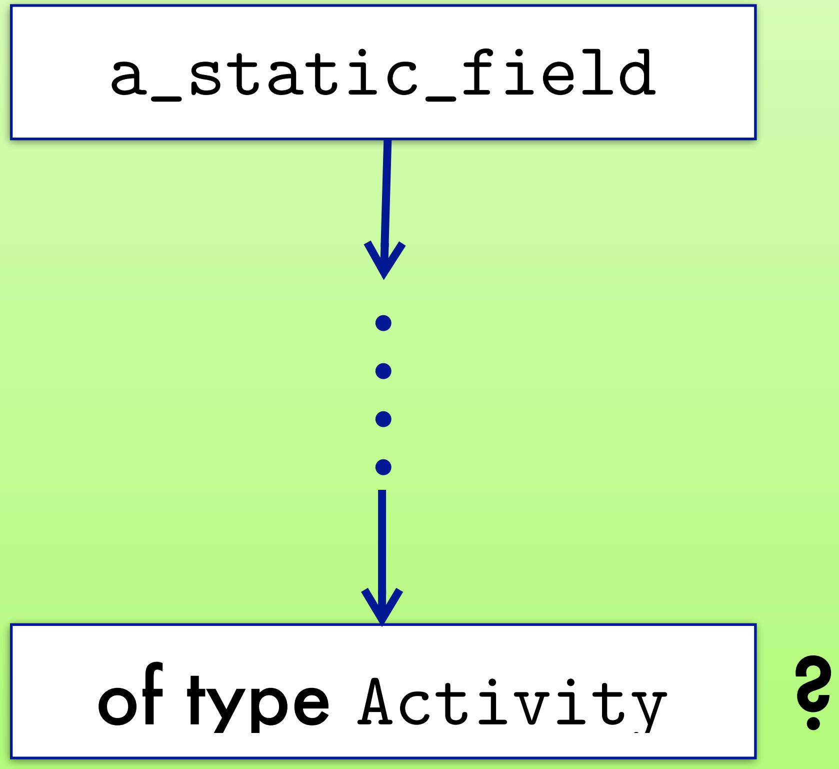
Idea 1: Refute points-to on-demand with second precise “filter” analysis

Idea 2: Leverage the facts from the first analysis in the filter analysis to scale

A top-level filter:
Filter leak alarms by refuting points-to edges

1

Is there a program execution where at some time

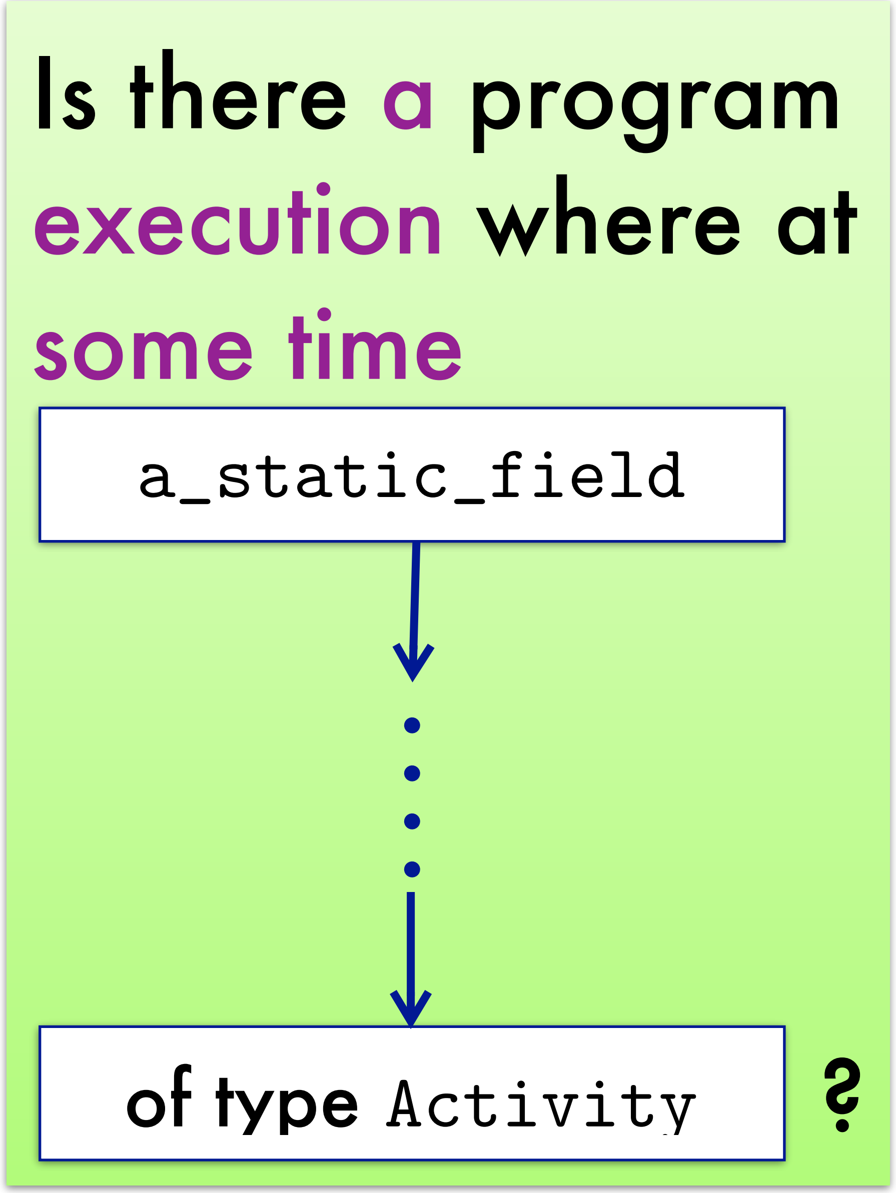


a_static_field

of type Activity ?

A top-level filter:
Filter leak alarms by refuting points-to edges

1



Is there a program execution where at some time

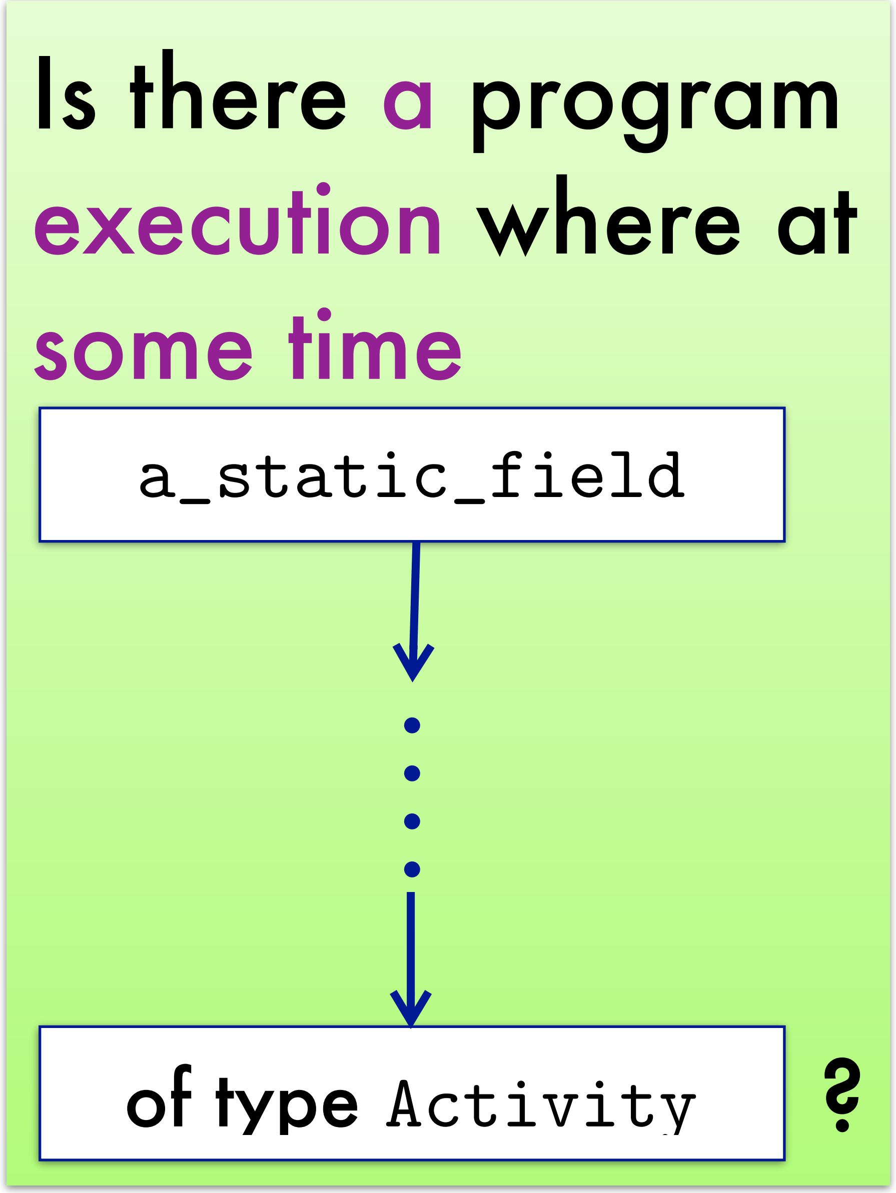
`a_static_field`

of type `Activity` ?

Select a points-to edge in the path

A top-level filter:
Filter leak alarms by refuting points-to edges

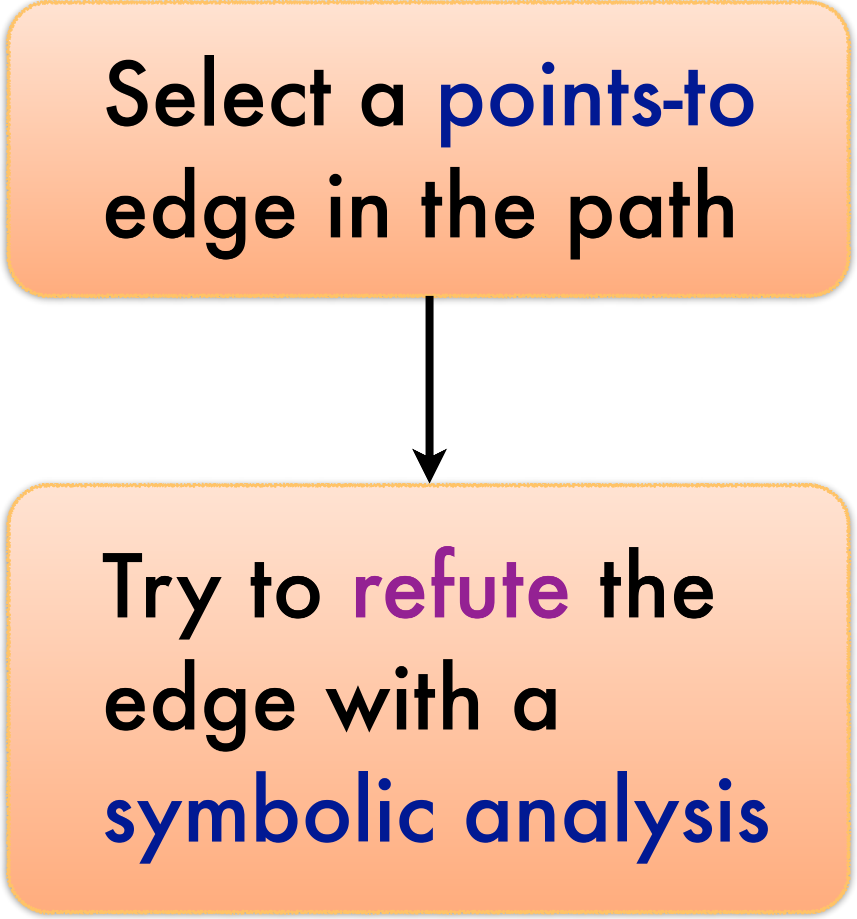
1



Is there a program execution where at some time

`a_static_field`

of type `Activity` ?



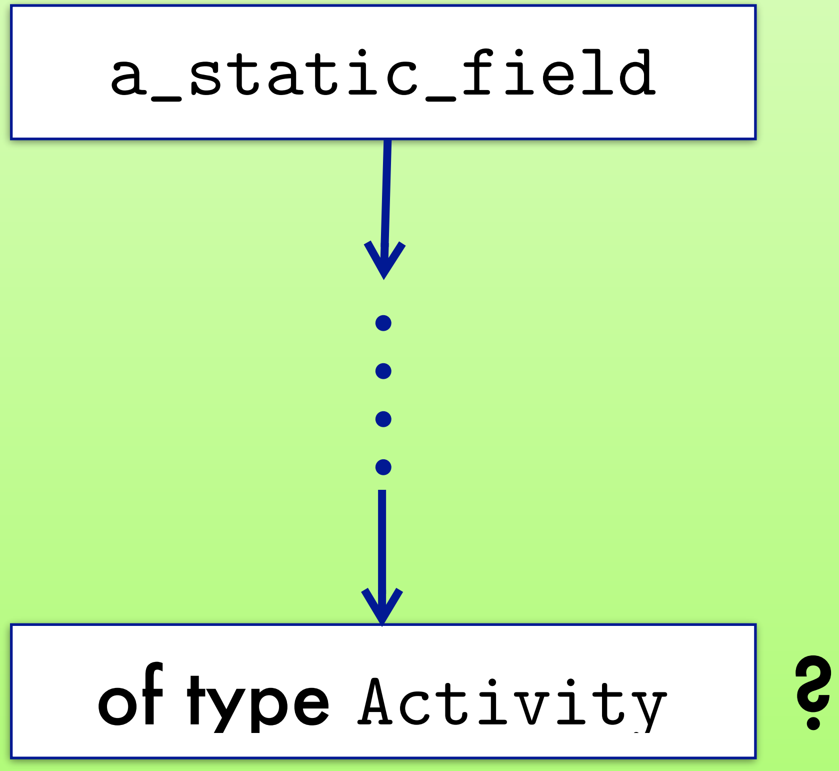
Select a points-to edge in the path

Try to refute the edge with a symbolic analysis

A top-level filter: Filter leak alarms by refuting points-to edges

1

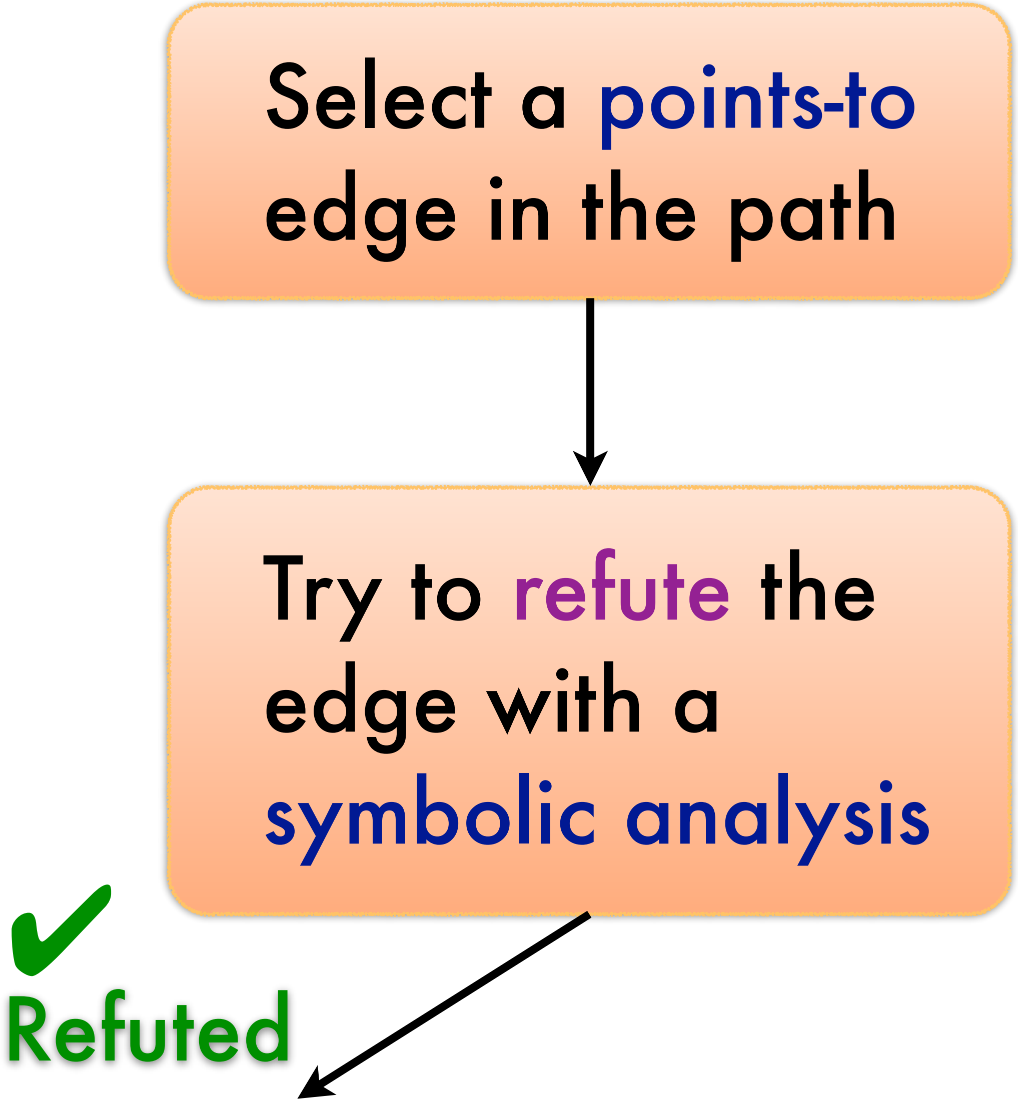
Is there a program execution where at some time



a_static_field

⋮

of type Activity ?



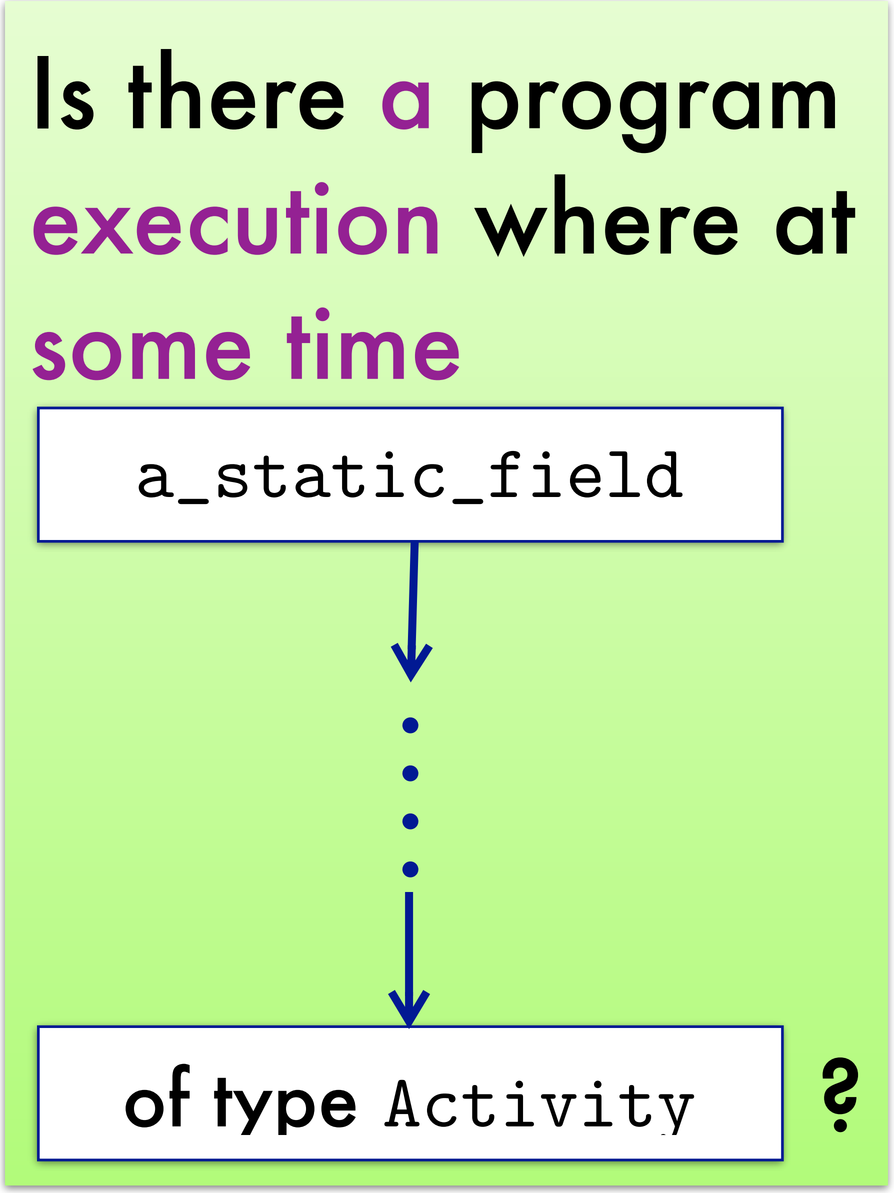
Select a points-to edge in the path

Try to refute the edge with a symbolic analysis

✔ Refuted

A top-level filter:
Filter leak alarms by refuting points-to edges

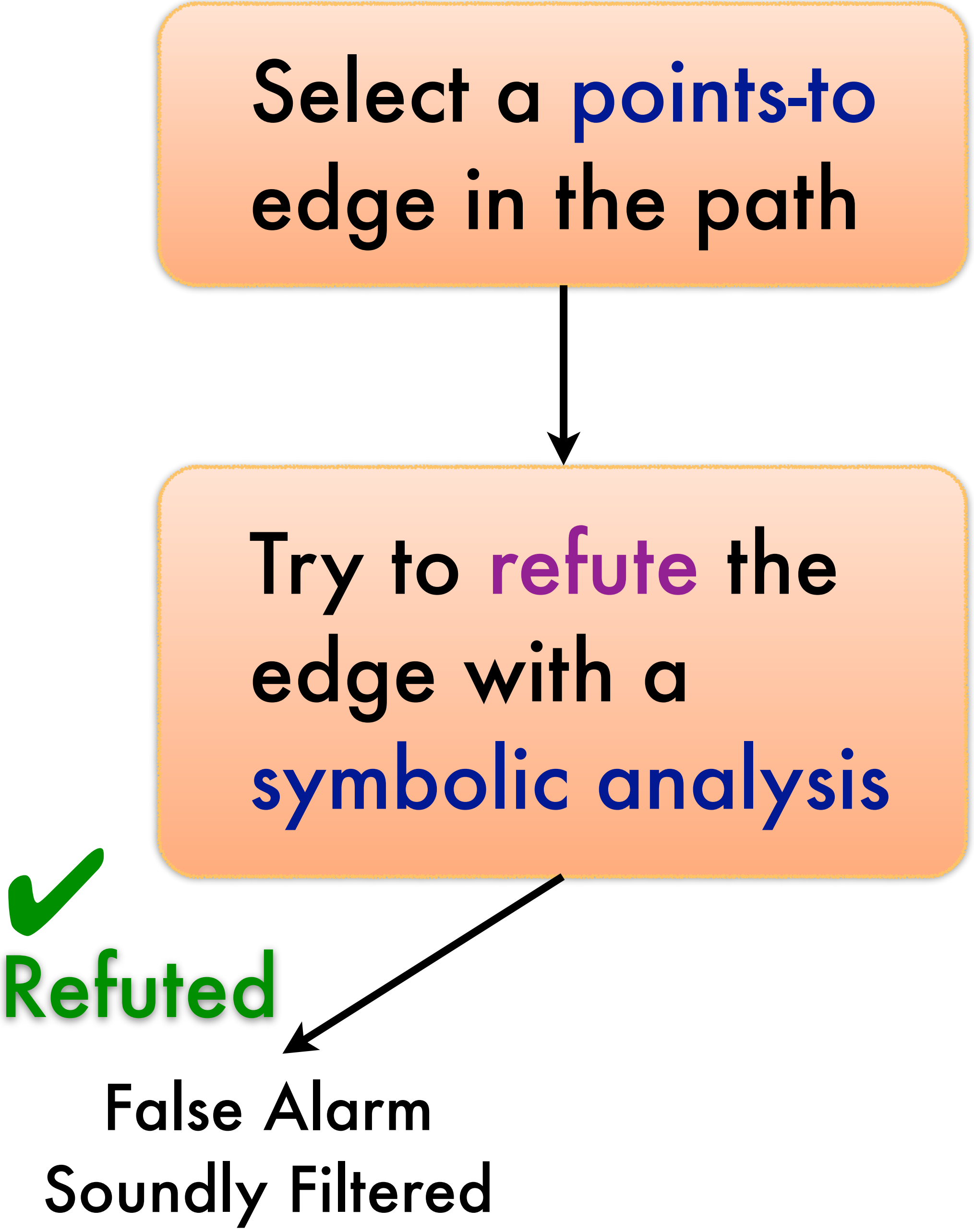
1



Is there a program execution where at some time

a_static_field

of type Activity ?



Select a points-to edge in the path

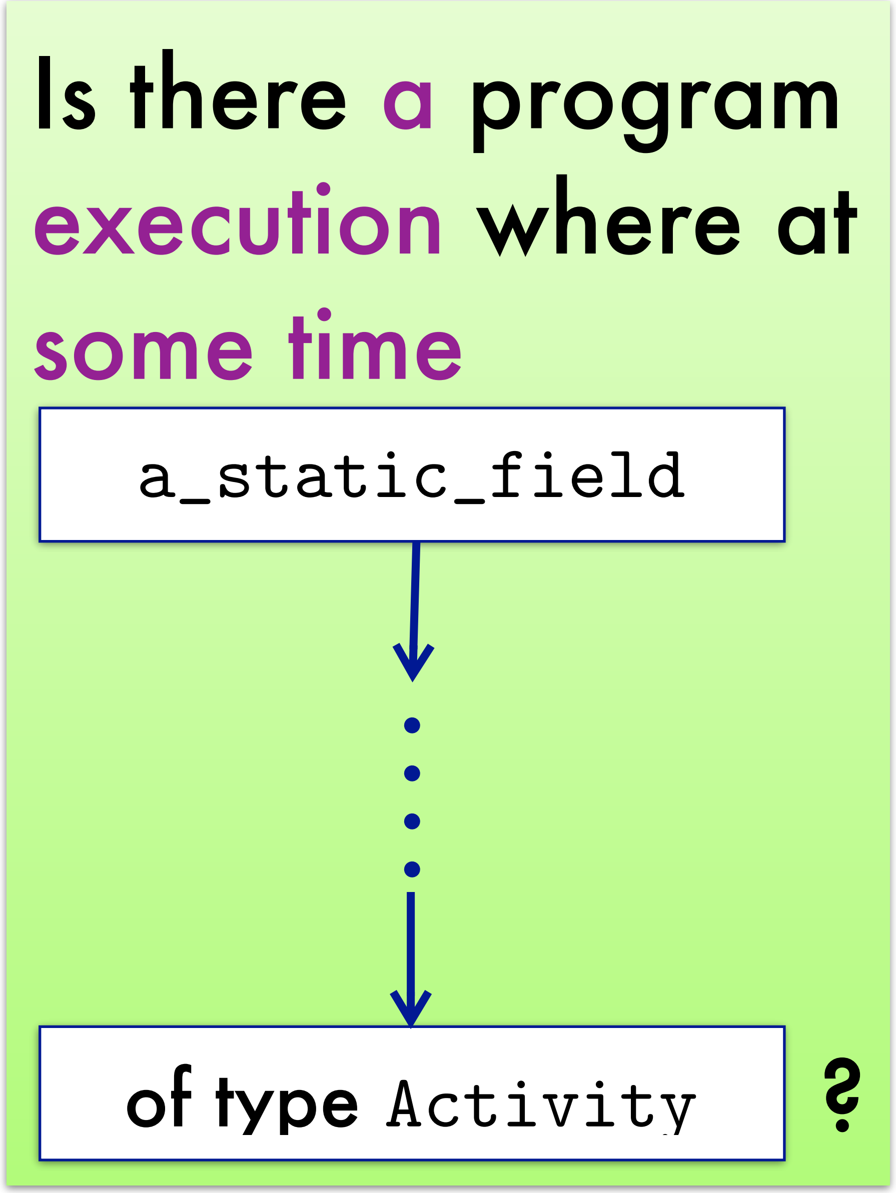
Try to refute the edge with a symbolic analysis

✓ Refuted

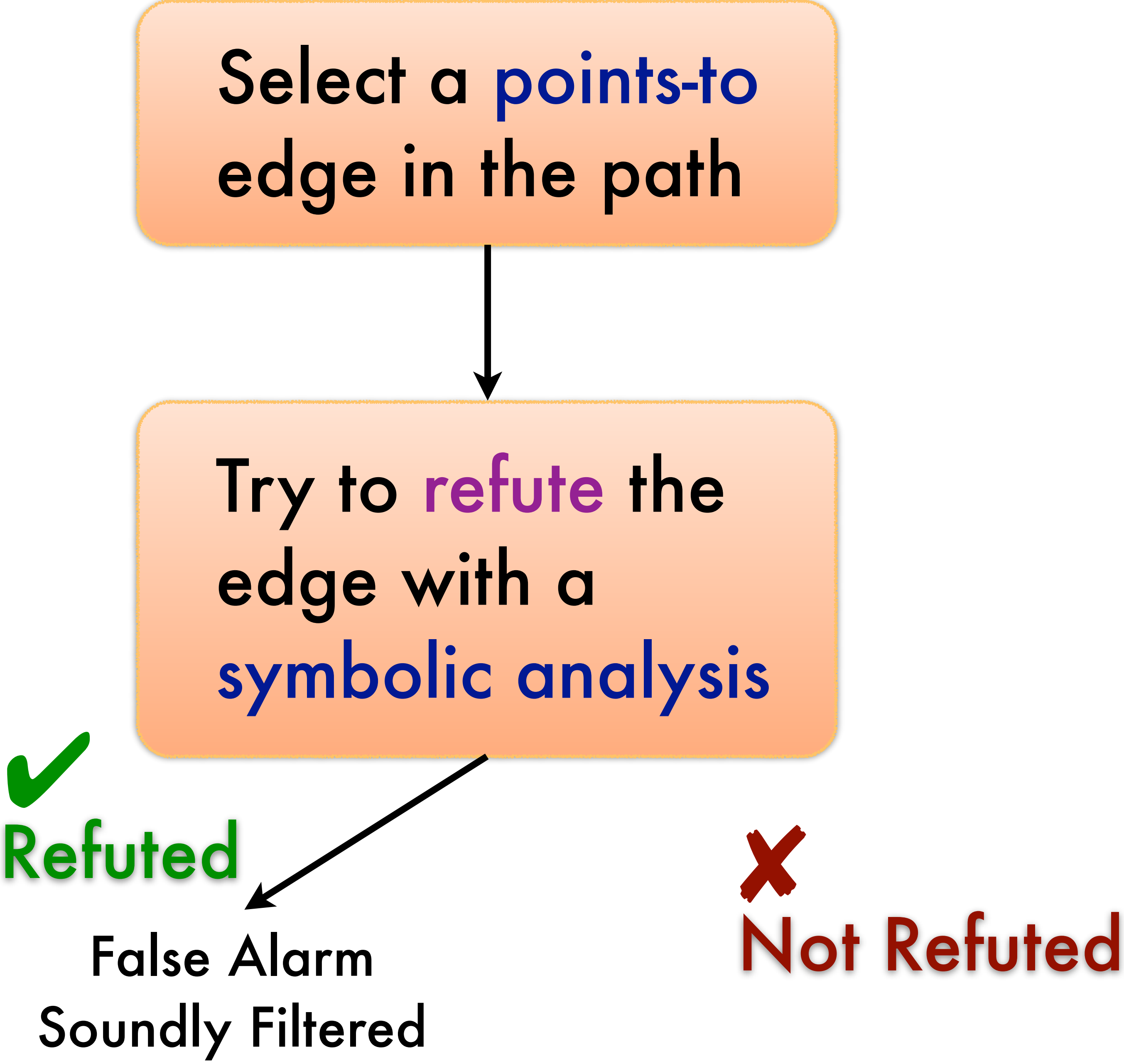
False Alarm
Soundly Filtered

A top-level filter:
Filter leak alarms by refuting points-to edges

1



Is there a program execution where at some time



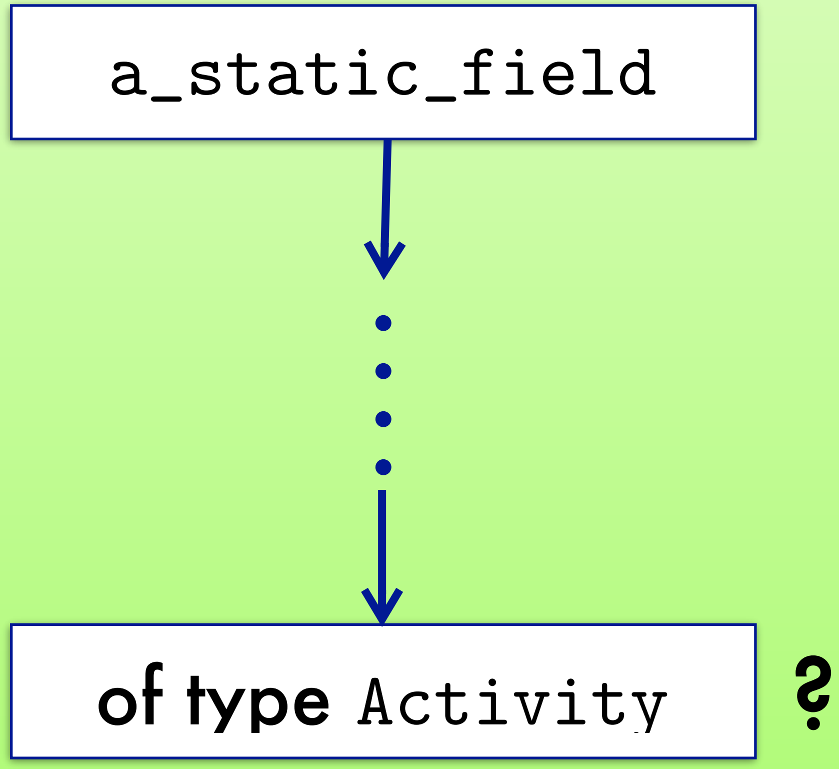
Select a points-to edge in the path

Try to refute the edge with a symbolic analysis

A top-level filter:
Filter leak alarms by refuting points-to edges

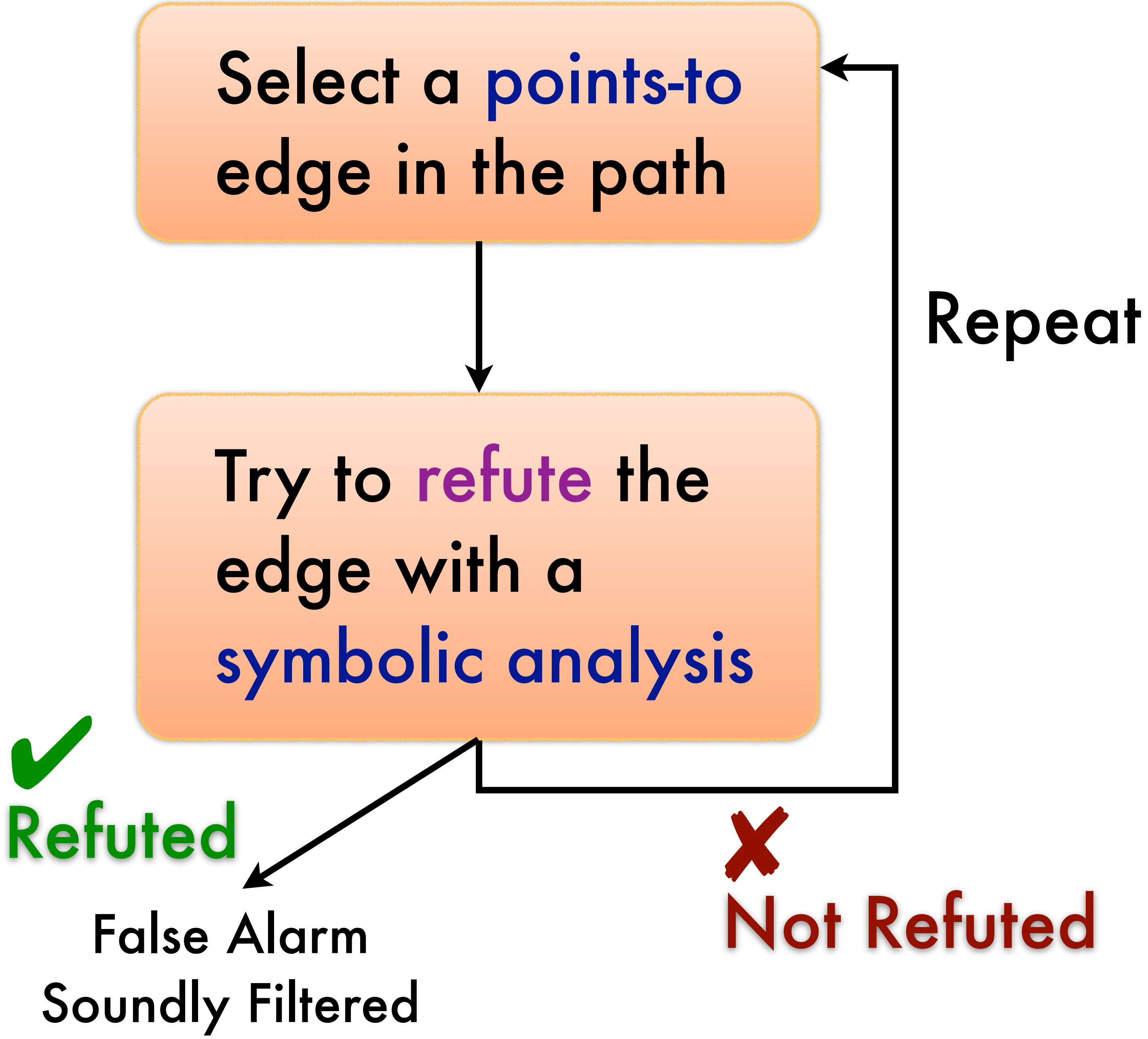
1

Is there a program execution where at some time



a_static_field

of type Activity ?



Select a points-to edge in the path

Try to refute the edge with a symbolic analysis

Repeat

✔ Refuted

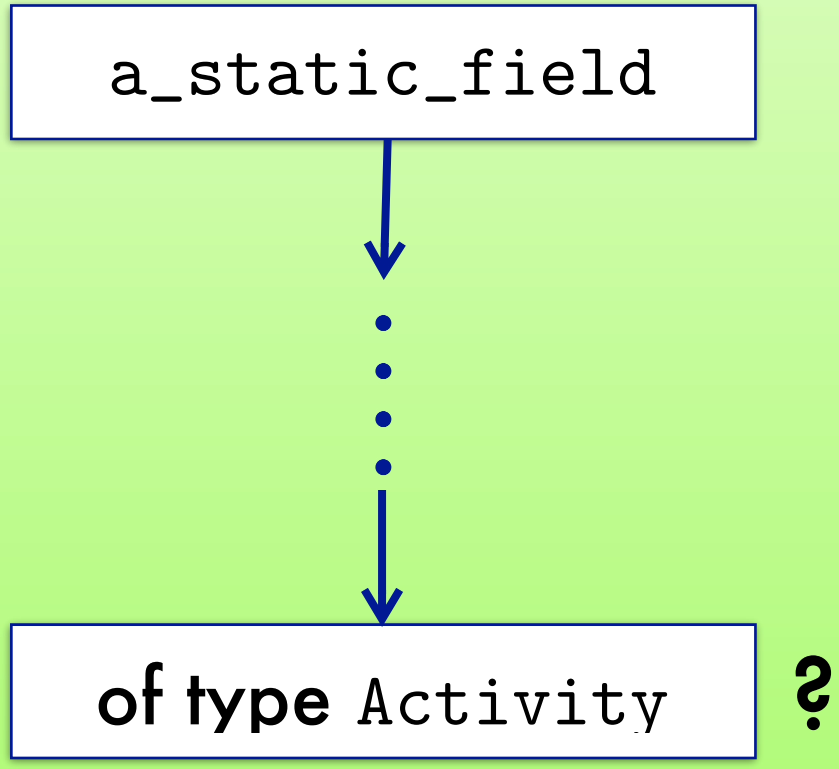
False Alarm
Soundly Filtered

✘ Not Refuted

A top-level filter:
Filter leak alarms by refuting points-to edges

1

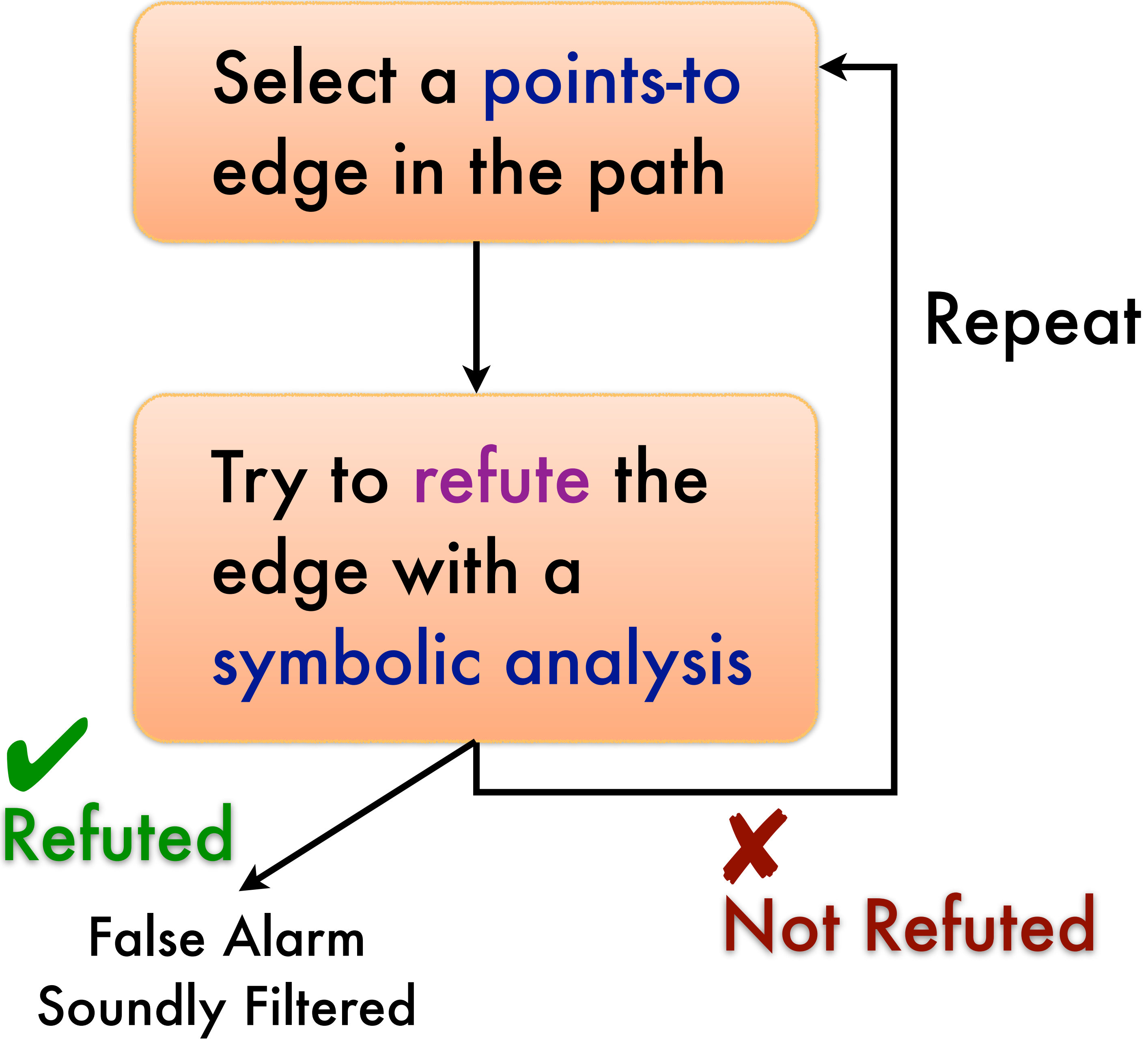
Is there a program execution where at some time



a_static_field

of type Activity

?



Select a points-to edge in the path

Try to refute the edge with a symbolic analysis

Repeat

✔ Refuted

False Alarm
Soundly Filtered

✘ Not Refuted

Refutation: Derive a contradiction, that a points-to relation can't actually hold

Refuting a points-to edge: What are we up against?

1

```
class Vec {
  static Object[] EMPTY = new$_{\mathsf{arr}_0}$ Object[1]; ...
  Vec() { this.tbl = EMPTY; capacity initially empty }









}
```

Refuting a points-to edge: What are we up against?

1

Null object pattern: Should never be written to

```
class Vec
  static Object[] EMPTY = new_{arr_0} Object[1]; ...
  Vec() { this.tbl = EMPTY; capacity initially empty }





}
```

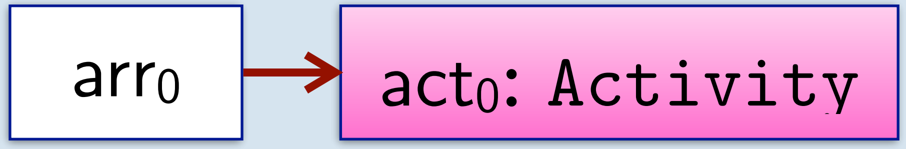
Refuting a points-to edge: What are we up against?

1

Null object pattern: Should never be written to

```
class Vec
  static Object[] EMPTY = new$_{arr_0}$ Object[1]; ...
  Vec() { this.tbl = EMPTY; capacity initially empty }

  void push(Object val) {
    if (need capacity) {
      this.tbl = new$_{arr_1}$ Object[more capacity];
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```

Refuting a points-to edge: What are we up against?


arr$_0$ → act$_0$: Activity

1

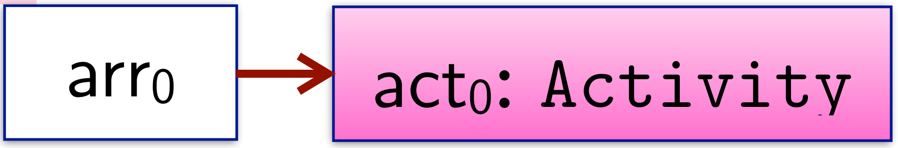
Null object pattern: Should never be written to

```
class Vec
  static Object[] EMPTY = new$_{arr_0}$ Object[1]; ...
  Vec() { this.tbl = EMPTY; capacity initially empty }

  void push(Object val) {
    if (need capacity) {
      this.tbl = new$_{arr_1}$ Object[more capacity];
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```

Refuting a points-to edge: What are we up against?

Null object pattern: Should never be written to

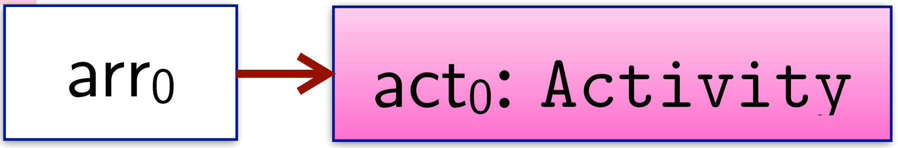
```
class Vec {
  static Object[] EMPTY = new_arr0 Object[1]; ...
  Vec() { this.tbl = EMPTY; capacity initially empty }

  void push(Object val) {
    if (need capacity) {
      this.tbl = new_arr1 Object[more capacity];
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```



Refuting a points-to edge: What are we up against?

Null object pattern: Should never be written to

```
class Vec {
  static Object[] EMPTY = new$_{arr_0}$ Object[1]; ...
  Vec() { this.tbl = EMPTY; capacity initially empty }

  void push(Object val) {
    if (need capacity) {
      this.tbl = new$_{arr_1}$ Object[more capacity];
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```



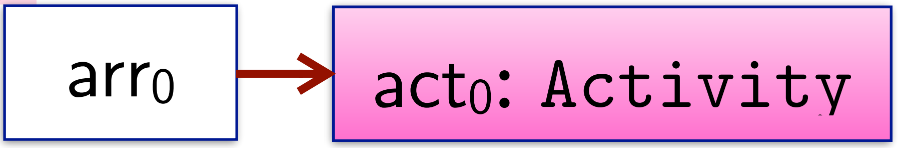
Refuting a points-to edge: What are we up against?

1

```
class Vec {
  static Object[] EMPTY = new_arr0 Object[1]; ...
  Vec() { this.tbl = EMPTY; capacity initially empty }

  void push(Object val) {
    if (need capacity) {
      this.tbl = new_arr1 Object[more capacity];
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```

Null object pattern: Should never be written to

Need interprocedural path-sensitivity



arr_0

act_0: Activity

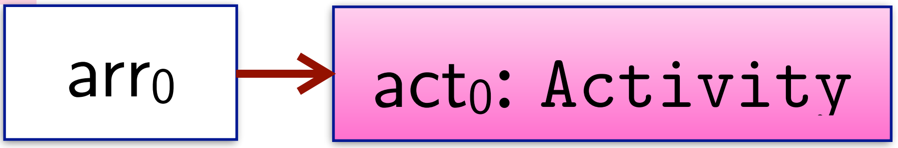
Refuting a points-to edge: What are we up against?

1

```
class Vec
  static Object[] EMPTY = new_arr0 Object[1]; ...
  Vec() { this.tbl = EMPTY; capacity initially empty }

  void push(Object
    if (need capacity) {
      this.tbl = new_arr1 Object[more capacity];
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```

Null object pattern: Should never be written to

Need interprocedural path-sensitivity



Refuting a points-to edge: What are we up against?

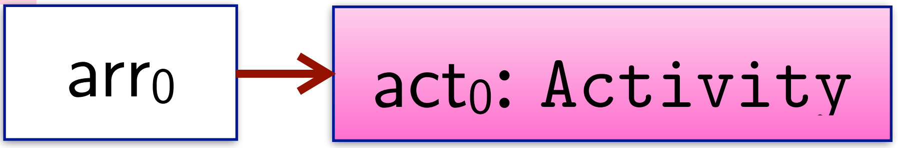
1

Null object pattern: Should never be written to

```
class Vec
  static Object[] EMPTY = new_arr0 Object[1]; ...
  Vec() { this.tbl = EMPTY; capacity initially empty }

  void push(Object
    if (need capacity) {
      this.tbl = new_arr1 Object[more capacity];
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```

Need interprocedural path-sensitivity

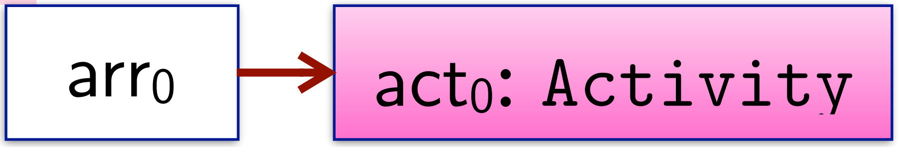
Need strong updates



A refutation analysis: Try to derive a contradiction with a backwards symbolic analysis

1

```
class Vec {
  static Object[] EMPTY = new_arr0 Object[1]; ...
  Vec() { this.tbl = EMPTY; capacity initially empty }

  void push(Object val) {
    if (need capacity) {
      this.tbl = new_arr1 Object[more capacity];
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```



A refutation analysis: Try to derive a contradiction with a backwards symbolic analysis

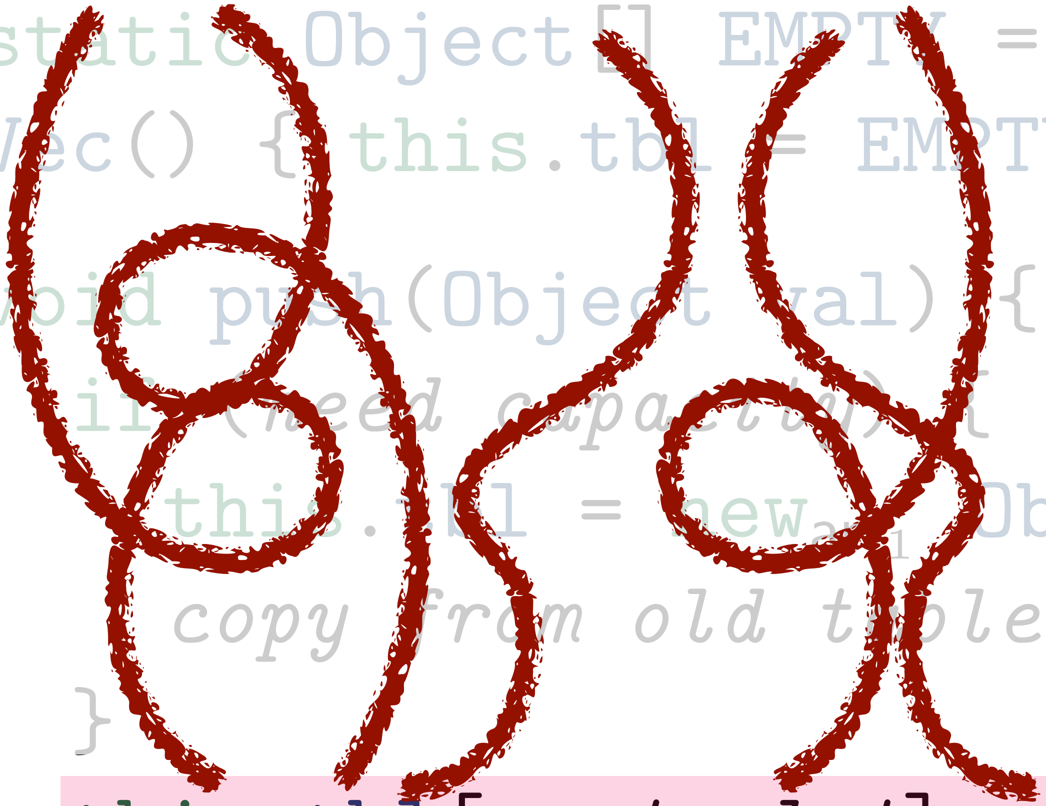
1

```
class Vec {
  static Object[] EMPTY = new_arr0 Object[1]; ...
  Vec() { this.tbl = EMPTY; capacity initially empty }

  void push(Object val) {
    if (need capacity) {
      this.tbl = new_arr1 Object[more capacity];
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```

$arr_0 \cdot$[-] $\mapsto act_0 *$ true

A refutation analysis: Try to derive a contradiction with a backwards symbolic analysis

1

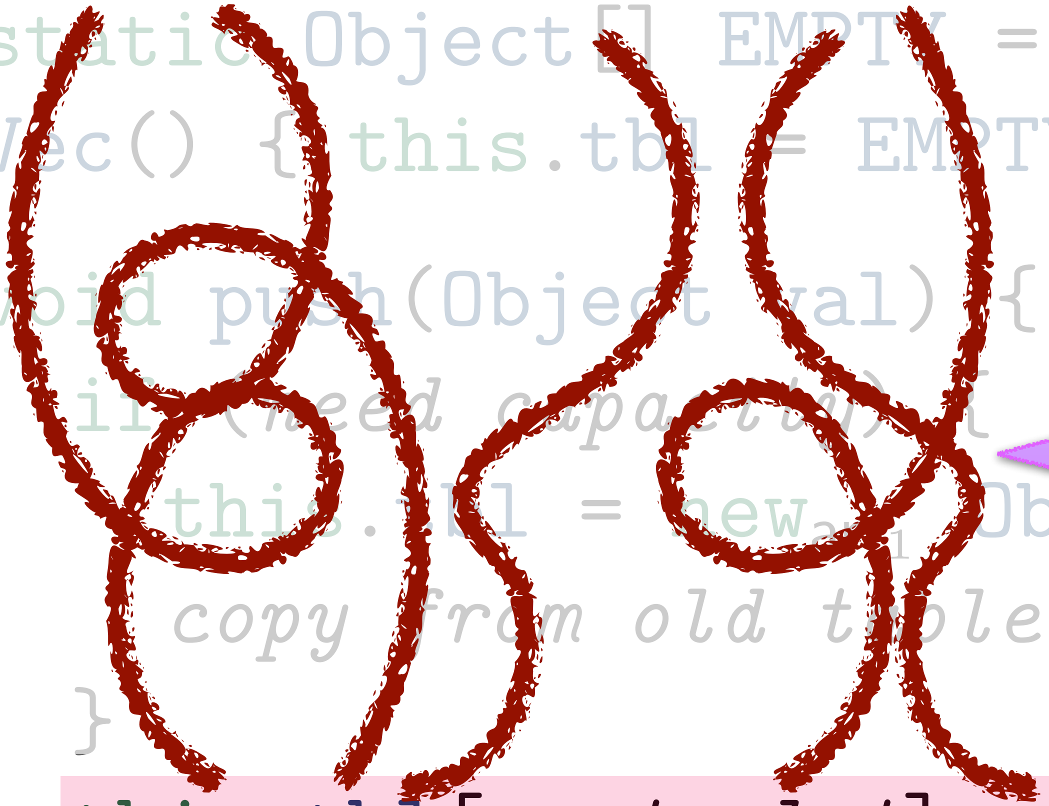


```
class Vec {
  static Object[] EMPTY = new_arr0 Object[1]; ...
  Vec() { this.tbl = EMPTY; capacity initially empty }

  void push(Object val) {
    if (need capacity) {
      this.tbl = new_arr1 Object[more capacity];
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```

$arr_0 \cdot [\text{-}] \mapsto act_0 * \mathsf{true}$

A refutation analysis: Try to derive a contradiction with a backwards symbolic analysis

1



```
class Vec {
  static Object[] EMPTY = new_arr0 Object[1]; ...
  Vec() { this.tbl = EMPTY
  
  void push(Object val) {
    if (need capacity) {
      this.tbl = new_arr1 Obj
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```

Derive a contradiction along all "backwards" path programs

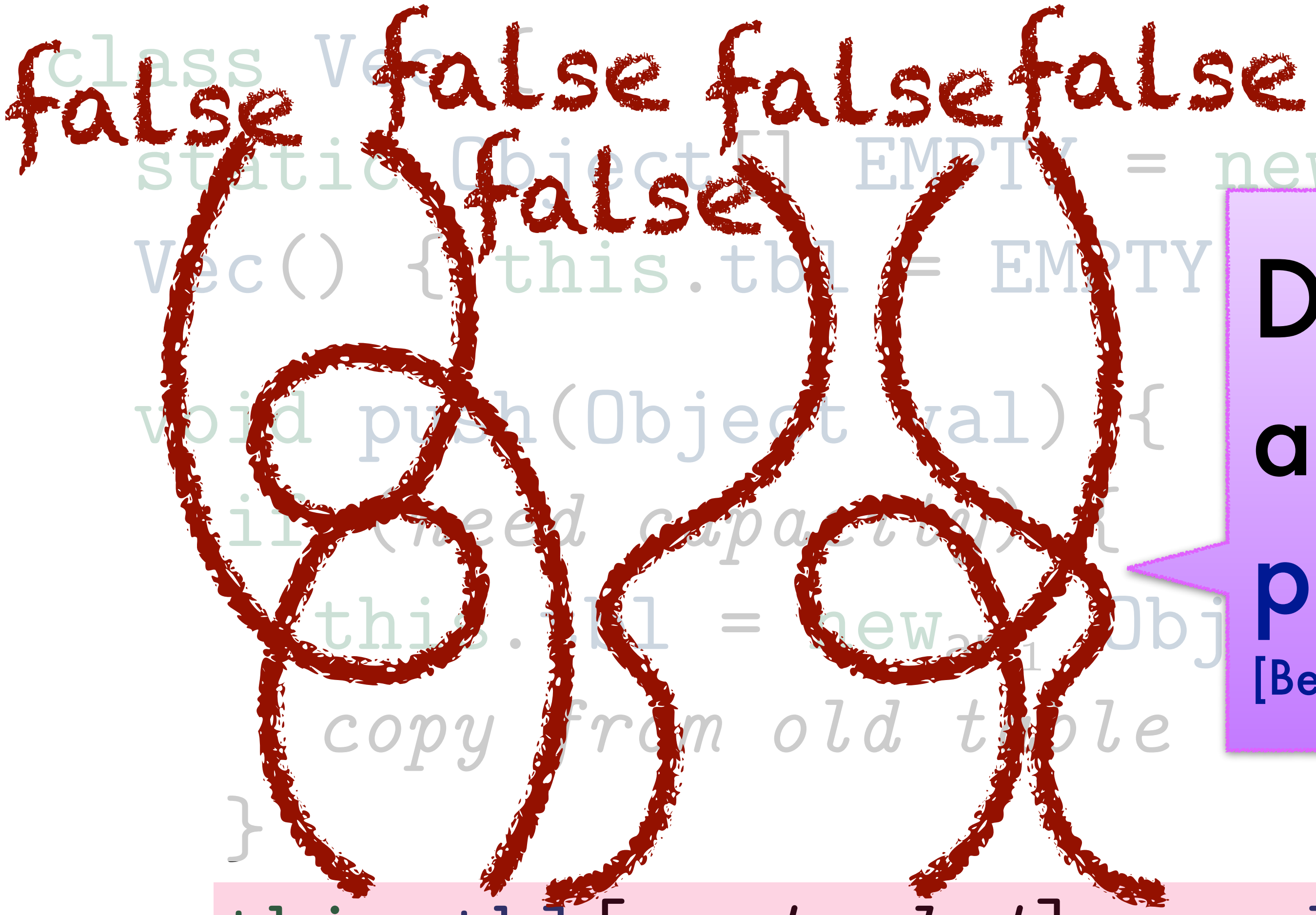
[Beyer, Henzinger, Majumdar, Rybalchenko (2007)]

$$arr_o\cdot[-] \mapsto act_o * \mathsf{true}$$

A refutation analysis: Try to derive a contradiction with a backwards symbolic analysis

1

```
class Vec {
  static Object[] EMPTY = new_arr Object[1]; ...
  Vec() { this.tbl = EMPTY

  void push(Object val) {
    if (need capacity) {
      this.tbl = new_arr Obj
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```



Derive a contradiction along all “backwards” path programs

[Beyer, Henzinger, Majumdar, Rybalchenko (2007)]

$arr_o \cdot [-] \mapsto act_o * \mathsf{true}$

A refutation analysis: Try to derive a contradiction with a backwards symbolic analysis

1

false false false false

false

```
class Vec {
  static Object[] EMPTY = new_arr0 Object[1]; ...
  Vec() { this.tbl = EMPTY
  
  void push(Object val) {
    if (need capacity) {
      this.tbl = new_arr1 Obj
      copy from old table
    }
    this.tbl[next slot] = val;
  }
}
```

Derive a contradiction along all “backwards” path programs

[Beyer, Henzinger, Majumdar, Rybalchenko (2007)]

$arr_0 \cdot [-] \mapsto act_0 * \mathsf{true}$

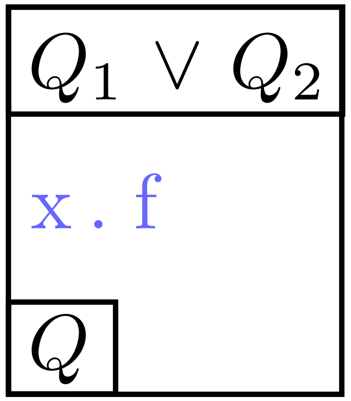
Derive refutations by trying to find witnesses

Roadmap: Precise but with scalability challenges

1

Roadmap: Precise but with scalability challenges

1



$Q_1 \vee Q_2$

x . f

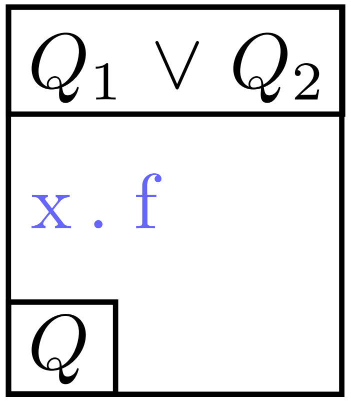
Q

Alias path explosion for strong updates

(On write, case split for each possible alias in Q to maintain separation)

Roadmap: Precise but with scalability challenges

1



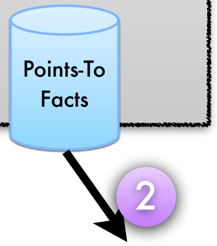
$Q_1 \vee Q_2$

x . f

Q

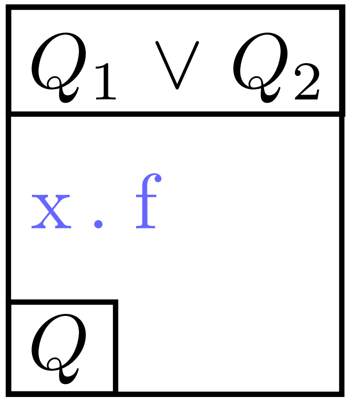
Alias path explosion for strong updates

(On write, case split for each possible alias in Q to maintain separation)



Roadmap: Precise but with scalability challenges

1



$Q_1 \vee Q_2$

x.f

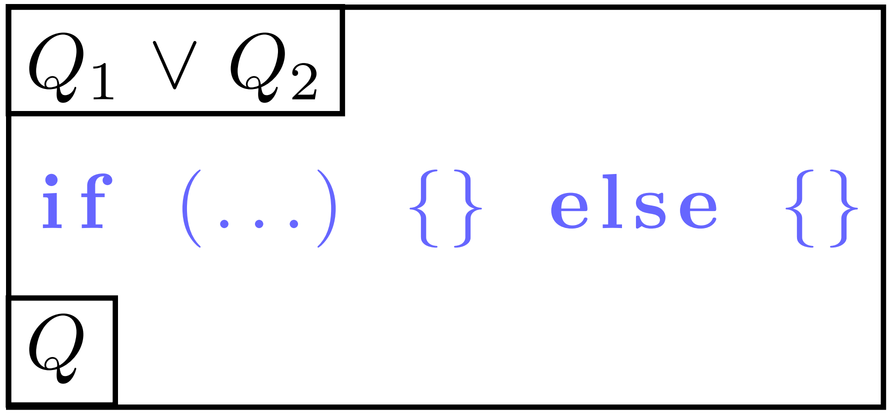
Q

Alias path explosion for strong updates

(On write, case split for each possible alias in Q to maintain separation)

Points-To Facts

2



$Q_1 \vee Q_2$

if (...) {} else {}

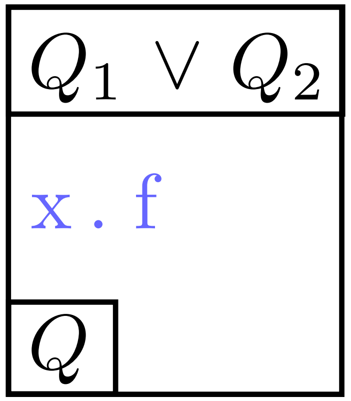
Q

Control-flow path explosion:

Ignore for now, reasonable if number of guards relevant to Q is small (e.g., [Das et al. (2002)])

Roadmap: Precise but with scalability challenges

1



$Q_1 \vee Q_2$

x.f

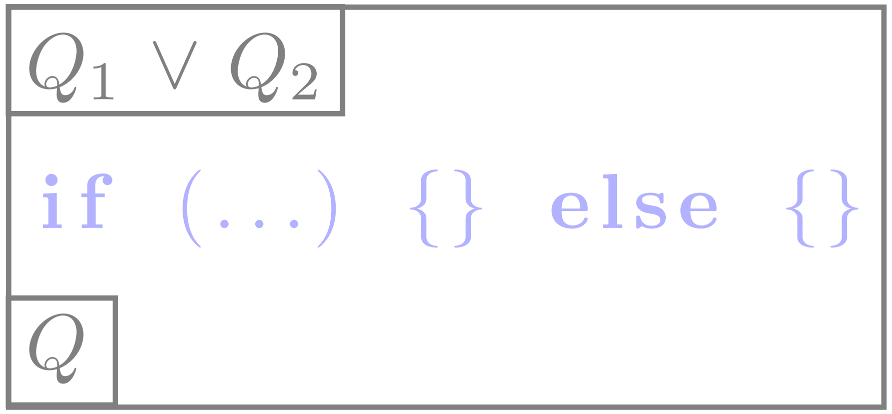
Q

Alias path explosion for strong updates

(On write, case split for each possible alias in Q to maintain separation)

Points-To Facts

2



$Q_1 \vee Q_2$

if (...) {} else {}

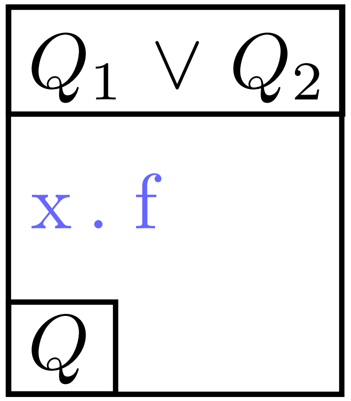
Q

Control-flow path explosion:

Ignore for now, reasonable if number of guards relevant to Q is small (e.g., [Das et al. (2002)])

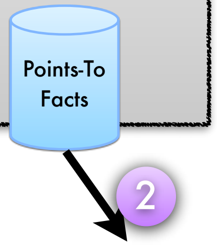
Roadmap: Precise but with scalability challenges

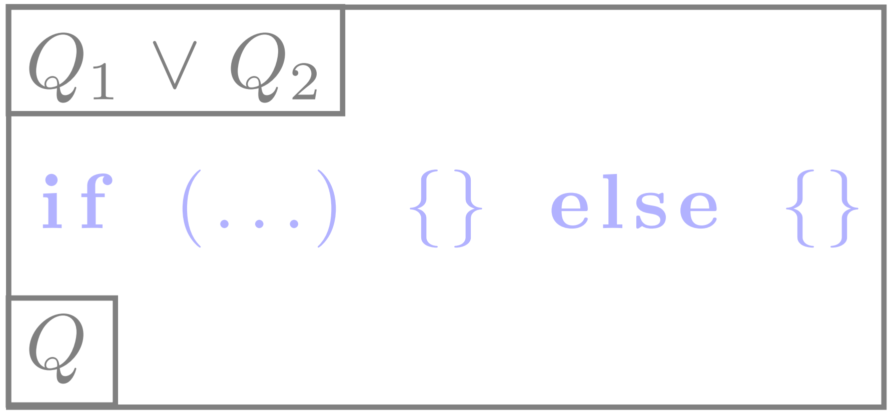
1



Alias path explosion for strong updates

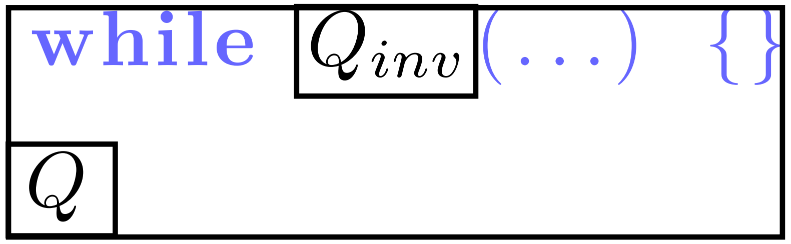
(On write, case split for each possible alias in Q to maintain separation)





Control-flow path explosion:

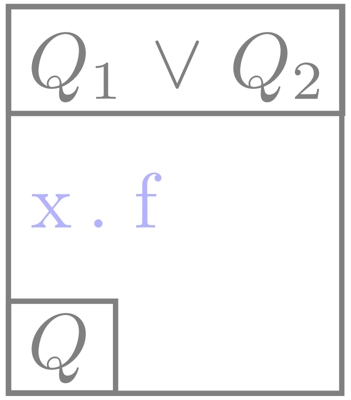
Ignore for now, reasonable if number of guards relevant to Q is small (e.g., [Das et al. (2002)])



Loops:

Simple loop invariant inference sufficient so far but more sophisticated techniques possible if needed

Roadmap: Precise but with scalability challenges



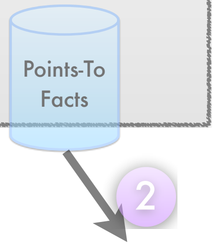
$Q_1 \vee Q_2$

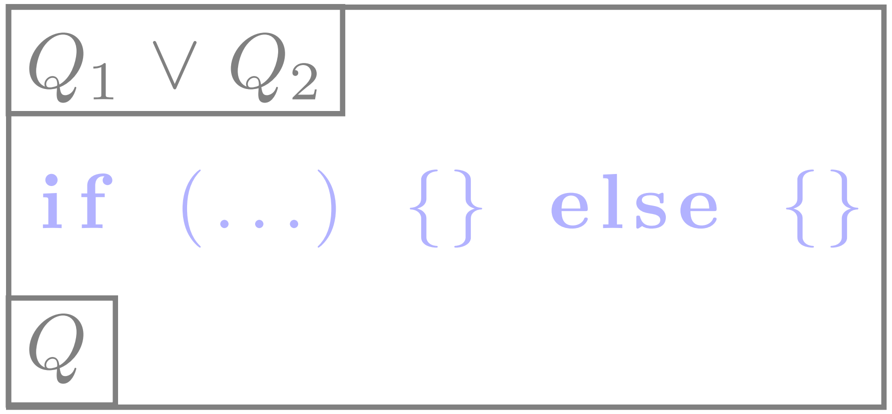
x.f

Q

Alias path explosion for strong updates

(On write, case split for each possible alias in Q to maintain separation)





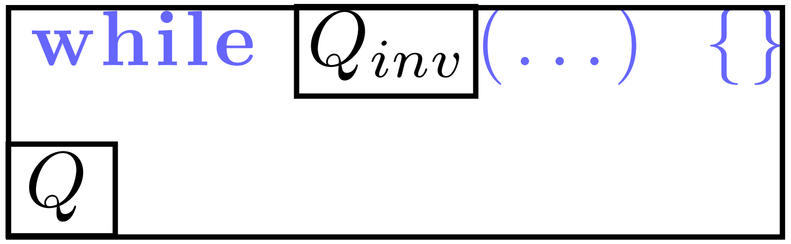
$Q_1 \vee Q_2$

if (...) {} else {}

Q

Control-flow path explosion:

Ignore for now, reasonable if number of guards relevant to Q is small (e.g., [Das et al. (2002)])



while Q_{inv} (...) {}

Q

Loops:

Simple loop invariant inference sufficient so far but more sophisticated techniques possible if needed

Roadmap: Precise but with scalability challenges

1

$Q_1 \lor Q_2$

x.f

Q

Alias path explosion for strong updates

(On write, case split for each possible alias in Q to maintain separation)

Points-To Facts

2

$Q_1 \lor Q_2$

if (...) {} else {}

Q

Control-flow path explosion:

Ignore for now, reasonable if number of guards relevant to Q is small (e.g., [Das et al. (2002)])

while $\boxed{Q_{inv}}$ (...) {}

Q

Over-approximate what?

Loops:

Simple loop invariant inference sufficient so far but more sophisticated techniques possible if needed

Soundness Criteria

1

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\quad \sigma \in \mathbf{State} \quad s \in \mathbf{Statement}$

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \textbf{State} \quad s \in \textbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \mathbf{State}$ $s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\widehat{\sigma} \in \hat{\mathbf{State}}$ $\gamma : \hat{\mathbf{State}} \to \wp(\mathbf{State})$

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \mathbf{State}$ $s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\widehat{\sigma} \in \mathbf{St\hat{a}te}$ $\gamma : \mathbf{St\hat{a}te} \to \wp(\mathbf{State})$

Standard Total Correctness Soundness Criteria

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ such that $\sigma \in \gamma(\widehat{\sigma})$,
then $\langle \sigma, s \rangle \Downarrow \sigma'$ for some $\sigma' \in \gamma(\widehat{\sigma}')$.

Soundness Criteria

1

Concrete Evaluation	$\langle \sigma, s \rangle \Downarrow \sigma'$	$\sigma \in \mathbf{State}$ $\quad s \in \mathbf{Statement}$
Abstract Analysis	$\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$	$\widehat{\sigma} \in \mathbf{St\hat{a}te}$ $\quad \gamma : \mathbf{St\hat{a}te} \to \wp(\mathbf{State})$

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ such that $\sigma \in \gamma(\widehat{\sigma})$,
then $\langle \sigma, s \rangle \Downarrow \sigma'$ for some $\sigma' \in \gamma(\widehat{\sigma}')$.

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \mathbf{State}$ $s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\widehat{\sigma} \in \widehat{\mathbf{State}}$ $\gamma : \widehat{\mathbf{State}} \to \wp(\mathbf{State})$

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ **such that** $\sigma' \in \gamma(\widehat{\sigma}')$**,**
then $\langle \sigma, s \rangle \Downarrow \sigma'$ **for some** $\sigma \in \gamma(\widehat{\sigma})$ **.**

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \mathbf{State}$ $s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\widehat{\sigma} \in \hat{\mathbf{State}}$ $\gamma : \hat{\mathbf{State}} \to \wp(\mathbf{State})$

Post: Goal

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ **such that** $\sigma' \in \gamma(\widehat{\sigma}')$**,**
then $\langle \sigma, s \rangle \Downarrow \sigma'$ **for some** $\sigma \in \gamma(\widehat{\sigma})$ **.**

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \mathbf{State}$ $s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\widehat{\sigma} \in \widehat{\mathbf{State}}$ $\gamma : \widehat{\mathbf{State}} \to \wp(\mathbf{State})$

"Total" Witness Soundness Criteria

Post: Goal

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ such that $\sigma' \in \gamma(\widehat{\sigma}')$,
then $\langle \sigma, s \rangle \Downarrow \sigma'$ for some $\sigma \in \gamma(\widehat{\sigma})$.

Soundness Criteria

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \mathbf{State}$ $s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\widehat{\sigma} \in \widehat{\mathbf{State}}$ $\gamma : \widehat{\mathbf{State}} \to \wp(\mathbf{State})$

"Total" Witness Soundness Criteria

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ such that $\sigma' \in \gamma(\widehat{\sigma}')$,
then $\langle \sigma, s \rangle \Downarrow \sigma'$ for some $\sigma \in \gamma(\widehat{\sigma})$.

Post: Goal

$\widehat{\sigma} = \bot$?

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \mathbf{State} \quad s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\widehat{\sigma} \in \widehat{\mathbf{State}} \quad \gamma : \widehat{\mathbf{State}} \to \wp(\mathbf{State})$

"Total" Witness Soundness Criteria

Post: Goal

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ such that $\sigma' \in \gamma(\widehat{\sigma}')$,
then $\langle \sigma, s \rangle \Downarrow \sigma'$ for some $\sigma \in \gamma(\widehat{\sigma})$.

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \mathbf{State}$ $s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\widehat{\sigma} \in \mathbf{St\hat{a}te}$ $\gamma : \mathbf{St\hat{a}te} \to \wp(\mathbf{State})$

"Total" Witness Soundness Criteria

Post: Goal

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ such that $\sigma' \in \gamma(\widehat{\sigma}')$,
then $\langle \sigma, s \rangle \Downarrow \sigma'$ for some $\sigma \in \gamma(\widehat{\sigma})$.

Ball, Kupferman, and Yorsh (2005)

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\quad \sigma \in \mathbf{State} \quad s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\quad \widehat{\sigma} \in \mathbf{St\hat{a}te} \quad \gamma : \mathbf{St\hat{a}te} \to \wp(\mathbf{State})$

"Total" Witness Soundness Criteria

Post: Goal

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ such that $\sigma' \in \gamma(\widehat{\sigma}')$,
then $\langle \sigma, s \rangle \Downarrow \sigma'$ for some $\sigma \in \gamma(\widehat{\sigma})$.

Ball, Kupferman, and Yorsh (2005)

Snugglebug, Alter, DART, ... are under-approximate

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \mathbf{State}$ $s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\widehat{\sigma} \in \mathbf{St\hat{a}te}$ $\gamma : \mathbf{St\hat{a}te} \to \wp(\mathbf{State})$

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ such that $\sigma' \in \gamma(\widehat{\sigma}')$,
then $\langle \sigma, s \rangle \Downarrow \sigma'$ for some $\sigma \in \gamma(\widehat{\sigma})$.

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \mathbf{State}$ $s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\widehat{\sigma} \in \hat{\mathbf{State}}$ $\gamma : \hat{\mathbf{State}} \to \wp(\mathbf{State})$

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ such that $\sigma' \in \gamma(\widehat{\sigma}')$ and $\langle \sigma, s \rangle \Downarrow \sigma'$, then $\sigma \in \gamma(\widehat{\sigma})$.

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \mathbf{State}$ $s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\widehat{\sigma} \in \mathbf{St\hat{a}te}$ $\gamma : \mathbf{St\hat{a}te} \to \wp(\mathbf{State})$

Refutation Soundness Criteria

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ **such that** $\sigma' \in \gamma(\widehat{\sigma}')$ **and** $\langle \sigma, s \rangle \Downarrow \sigma'$**,** **then** $\sigma \in \gamma(\widehat{\sigma})$**.**

Soundness Criteria

1

If a loop may "produce" a conjunct of the query, we can "assume it does" (weaken the query) only at the cost of precision.

Refutation Soundness Criteria

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ such that $\sigma' \in \gamma(\widehat{\sigma}')$ and $\langle \sigma, s \rangle \Downarrow \sigma'$, then $\sigma \in \gamma(\widehat{\sigma})$.

Soundness Criteria

1

If a loop may "produce" a conjunct of the query, we can "assume it does" (weaken the query) only at the cost of precision.

Refutation Soundness Criteria

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ such that $\sigma' \in \gamma(\widehat{\sigma}')$ and $\langle \sigma, s \rangle \Downarrow \sigma'$, then $\sigma \in \gamma(\widehat{\sigma})$.

Refutations: Prove alarms false with "partial" witnesses, an "easier condition" for loops

Soundness Criteria

1

Concrete Evaluation $\langle \sigma, s \rangle \Downarrow \sigma'$ $\sigma \in \mathbf{State}$ $s \in \mathbf{Statement}$

Abstract Analysis $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ $\widehat{\sigma} \in \hat{\mathbf{State}}$ $\gamma : \hat{\mathbf{State}} \rightarrow \wp(\mathbf{State})$

If $\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$ such that $\sigma' \in \gamma(\widehat{\sigma}')$ and $\langle \sigma, s \rangle \Downarrow \sigma'$, then $\sigma \in \gamma(\widehat{\sigma})$.

Soundness Criteria

1

Concrete Evaluation	$\langle \sigma, s \rangle \Downarrow \sigma'$	$\sigma \in \mathbf{State}$	$s \in \mathbf{Statement}$
Abstract Analysis	$\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$	$\widehat{\sigma} \in \widehat{\mathbf{State}}$	$\gamma : \widehat{\mathbf{State}} \to \wp(\mathbf{State})$

If such that and $\langle \sigma, s \rangle \Downarrow \sigma'$,
then .

Soundness Criteria

1

Concrete Evaluation	$\langle \sigma, s \rangle \Downarrow \sigma'$	$\sigma \in \mathbf{State}$ $\quad s \in \mathbf{Statement}$
Abstract Analysis	$\vdash \{\widehat{\sigma}\}\ s\ \{\widehat{\sigma}'\}$	$\widehat{\sigma} \in \mathbf{St\hat{a}te}$ $\quad \gamma : \mathbf{St\hat{a}te} \to \wp(\mathbf{State})$

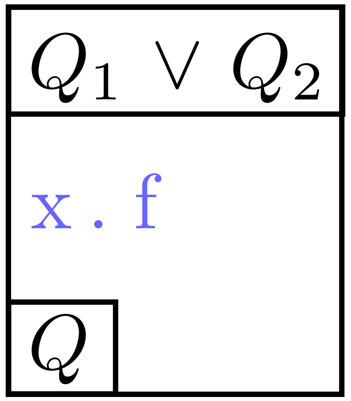
Necessary Precondition Soundness Criteria

If $\vdash \{\widehat{\sigma}\}\ s$ fault **such that** $\sigma \notin \gamma(\widehat{\sigma})$ **and** $\langle \sigma, s \rangle \Downarrow \sigma'$**,**
then $\mathrm{error}(\sigma')$.

Cousot, Cousot, Fähndrich, Logozzo (VMCAI'13)

Roadmap: Precise but with scalability challenges

1



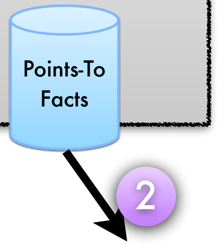
$Q_1 \vee Q_2$

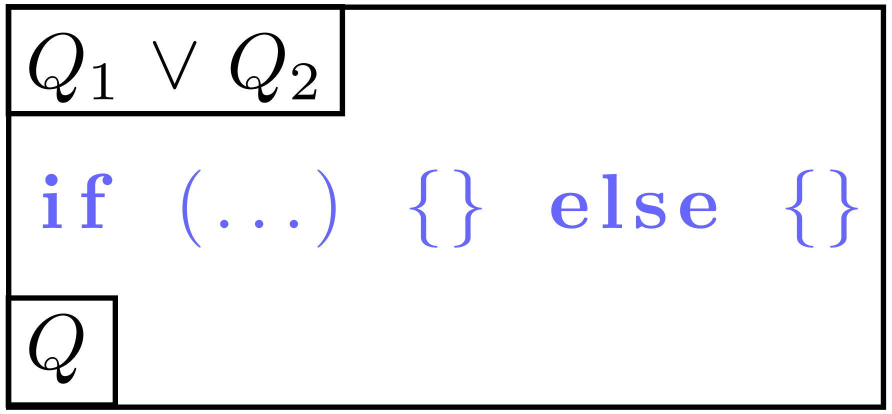
x.f

Q

Alias path explosion for strong updates

(On write, case split for each possible alias in Q to maintain separation)





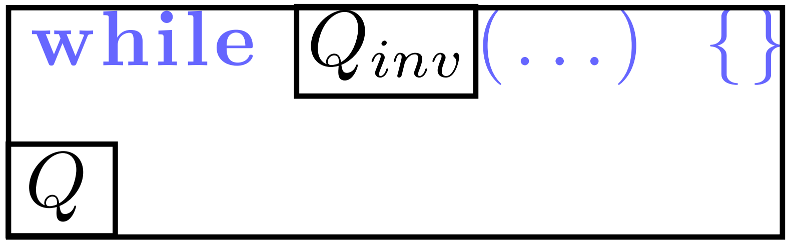
$Q_1 \vee Q_2$

if (...) {} else {}

Q

Control-flow path explosion:

Ignore for now, reasonable if number of guards relevant to Q is small (e.g., [Das et al. (2002)])



while Q_{inv} (...) {}

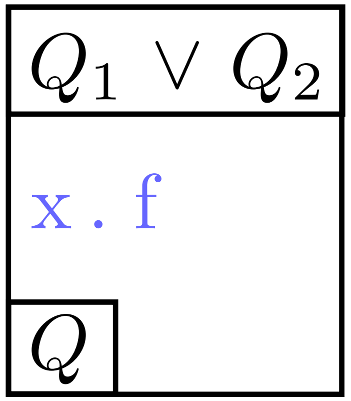
Q

Loops:

Simple loop invariant inference sufficient so far but more sophisticated techniques possible if needed

Roadmap: Precise but with scalability challenges

1



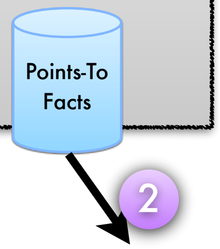
$Q_1 \vee Q_2$

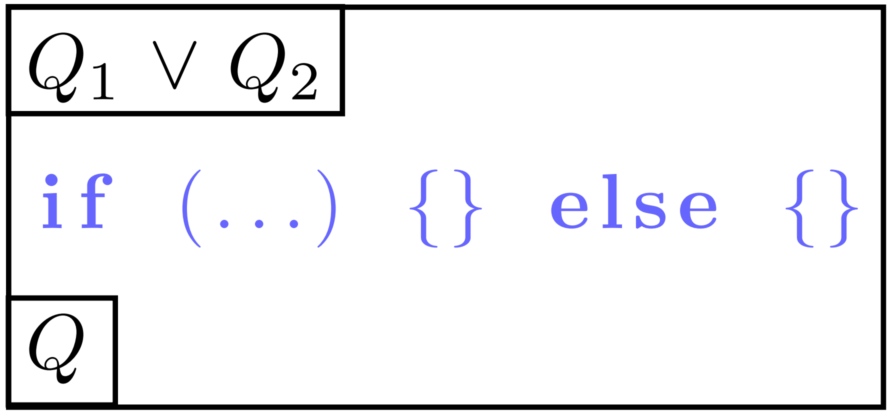
`x.f`

Q

Alias path explosion for strong updates

(On write, case split for each possible alias in Q to maintain separation)





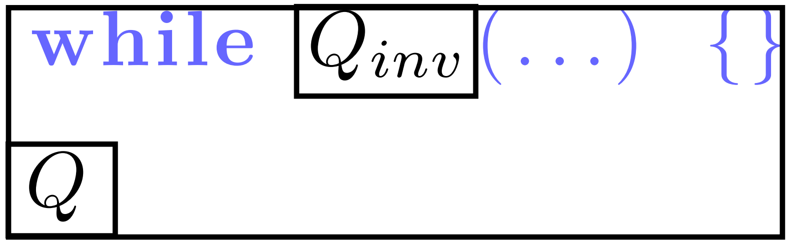
$Q_1 \vee Q_2$

`if (...) {} else {}`

Q

Control-flow path explosion:

Ignore for now, reasonable if number of guards relevant to Q is small (e.g., [Das et al. (2002)])



`while` Q_{inv} `(...) {}`

Q

Loops:

Simple loop invariant inference sufficient so far but more sophisticated techniques possible if needed

Roadmap: Precise but with scalability challenges

1

$Q_1 \vee Q_2$

```
x.f
```

Q

Alias path explosion for strong updates

(On write, case split for each possible alias in Q to maintain separation)

Points-To Facts

2

$Q_1 \vee Q_2$

```
if (...) {} else {}
```

Q

Control-flow path explosion:

Ignore for now, reasonable if number of guards relevant to Q is small (e.g., [Das et al. (2002)])

```
while
```

Q_{inv}

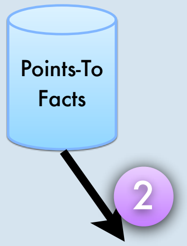
```
(...) {}
```

Q

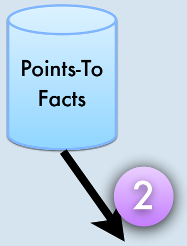
Loops:

Simple loop invariant inference sufficient so far but more sophisticated techniques possible if needed

from constraints: Reducing separation constraints with points-to facts

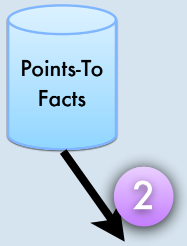


from constraints: Reducing separation constraints with points-to facts



o from { ..., $\boxed{a_i}$,... }

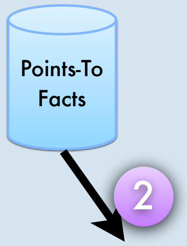
from constraints: Reducing separation constraints with points-to facts



o from { ..., $\boxed{a_i}$,... }

symbolic object instance (an address)

from constraints: Reducing separation constraints with points-to facts



o from { ..., $\boxed{a_i}$,... }

symbolic object instance (an address)

abstract loc in points-to (set of addresses)

from constraints: Reducing separation constraints with points-to facts

Points-To Facts

2

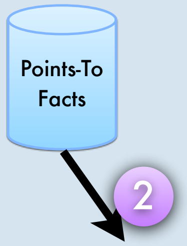
o from { ..., $\boxed{a_i}$,... }

symbolic object instance (an address)

abstract loc in points-to (set of addresses)

Refute (derive false) if:

from constraints: Reducing separation constraints with points-to facts



o from { ..., $\boxed{a_i}$,... }

symbolic object instance (an address)

abstract loc in points-to (set of addresses)

Refute (derive false) if:

`i > j ∧ i < j`

from constraints: Reducing separation constraints with points-to facts

Points-To Facts

2

o from { ..., $\boxed{a_i}$,... }

symbolic object instance (an address)

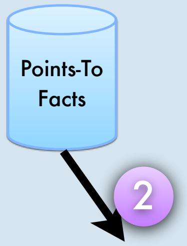
abstract loc in points-to (set of addresses)

Refute (derive false) if:

$i > j \land i < j$

or $o \cdot f \mapsto p \ast o \cdot f \mapsto q \land p \neq q$

from constraints: Reducing separation constraints with points-to facts



o from { ..., $\boxed{a_i}$,... }

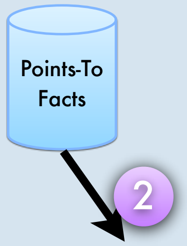
symbolic object instance (an address)

abstract loc in points-to (set of addresses)

Refute (derive false) if:

- `i > j ∧ i < j`
- **or** `o·f↦p * o·f↦q ∧ p≠q`
- **or** `o from ∅`

from constraints: Reducing separation constraints with points-to facts



o from { ..., $\boxed{a_i}$,... }

symbolic object instance (an address)

abstract loc in points-to (set of addresses)

Refute (derive false) if:

$\mathtt{i} > \mathtt{j} \wedge \mathtt{i} < \mathtt{j}$

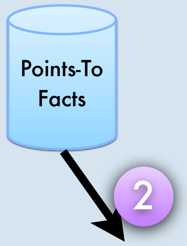
or $\mathtt{o} \cdot \mathtt{f} \mapsto \mathtt{p} * \mathtt{o} \cdot \mathtt{f} \mapsto \mathtt{q} \wedge \mathtt{p} \neq \mathtt{q}$

or $\mathtt{o}$ from $\varnothing$

```
x.f = p
```

$\boxed{\mathtt{y} \cdot \mathtt{f} \mapsto \mathtt{p}}$

from constraints: Reducing separation constraints with points-to facts



o from { ..., $\boxed{a_i}$,... }

symbolic object instance (an address)

abstract loc in points-to (set of addresses)

Refute (derive false) if:

$i > j \wedge i < j$

or $o \cdot f \mapsto p \ast o \cdot f \mapsto q \wedge p \neq q$

or o from $\varnothing$

$$\boxed{\mathrm{y} \text{ from } \mathrm{pt}(\mathrm{x}) \cap \mathrm{pt}(\mathrm{y}) \wedge \mathrm{x} = \mathrm{y}}$$

$$\vee \boxed{\mathrm{y} \cdot \mathrm{f} \mapsto \mathrm{p} \wedge \mathrm{x} \neq \mathrm{y}}$$

```
x.f = p
```

$$\boxed{\mathrm{y} \cdot \mathrm{f} \mapsto \mathrm{p}}$$

from constraints: Reducing separation constraints with points-to facts

Points-To Facts

2

o from { ..., $\boxed{a_i}$,... }

symbolic object instance (an address)

abstract loc in points-to (set of addresses)

Refute (derive false) if:

`i > j ∧ i < j`

or `o·f↦p * o·f↦q ∧ p≠q`

or `o from ∅`

$$\boxed{\mathrm{y\ from\ pt(x) \cap pt(y) \land x = y}}$$

$$\lor\ \boxed{\mathtt{y} \cdot \mathtt{f} \mapsto \mathtt{p} \land \mathtt{x} \neq \mathtt{y}}$$

```
x.f = p
```

$$\boxed{\mathtt{y} \cdot \mathtt{f} \mapsto \mathtt{p}}$$

from constraints: Reducing separation constraints with points-to facts

Points-To Facts

2

o from { ..., $\boxed{a_i}$,... }

symbolic object instance (an address)

abstract loc in points-to (set of addresses)

Refute (derive false) if:

$\texttt{i > j} \wedge \texttt{i < j}$

or $\texttt{o}\cdot\texttt{f}\mapsto\texttt{p} * \texttt{o}\cdot\texttt{f}\mapsto\texttt{q} \wedge \texttt{p}\neq\texttt{q}$

or $\texttt{o}$ from $\varnothing$

$$\boxed{\mathrm{y}\ \text{from}\ \mathrm{pt}(\mathrm{x}) \cap \mathrm{pt}(\mathrm{y}) \wedge \mathrm{x} = \mathrm{y}}$$

$$\vee\ \boxed{\mathrm{y}\cdot\mathrm{f}\mapsto\mathrm{p}\wedge\mathrm{x}\neq\mathrm{y}}$$

```
x.f = p
```

$$\boxed{\mathrm{y}\cdot\mathrm{f}\mapsto\mathrm{p}}$$

Generalized disalias check:

$\mathrm{pt}(\mathrm{x}) \cap \mathrm{pt}(\mathrm{y}) = \varnothing$

from constraints: Reducing separation constraints with points-to facts

Points-To Facts

2

o from { …, $\boxed{a_i}$,… }

symbolic object instance (an address)

abstract loc in points-to (set of addresses)

Restriction on possible abstract locations based on flow in the backwards analysis

Refute (derive false) if:

$\mathtt{i > j \land i < j}$

or $\mathtt{o \cdot f \mapsto p \ast o \cdot f \mapsto q \land p \neq q}$

or o from $\varnothing$

$$\boxed{\mathrm{y} \text{ from } \mathrm{pt(x) \cap pt(y) \land x = y}}$$

$$\lor \boxed{\mathtt{y \cdot f \mapsto p} \land \mathrm{x \neq y}}$$

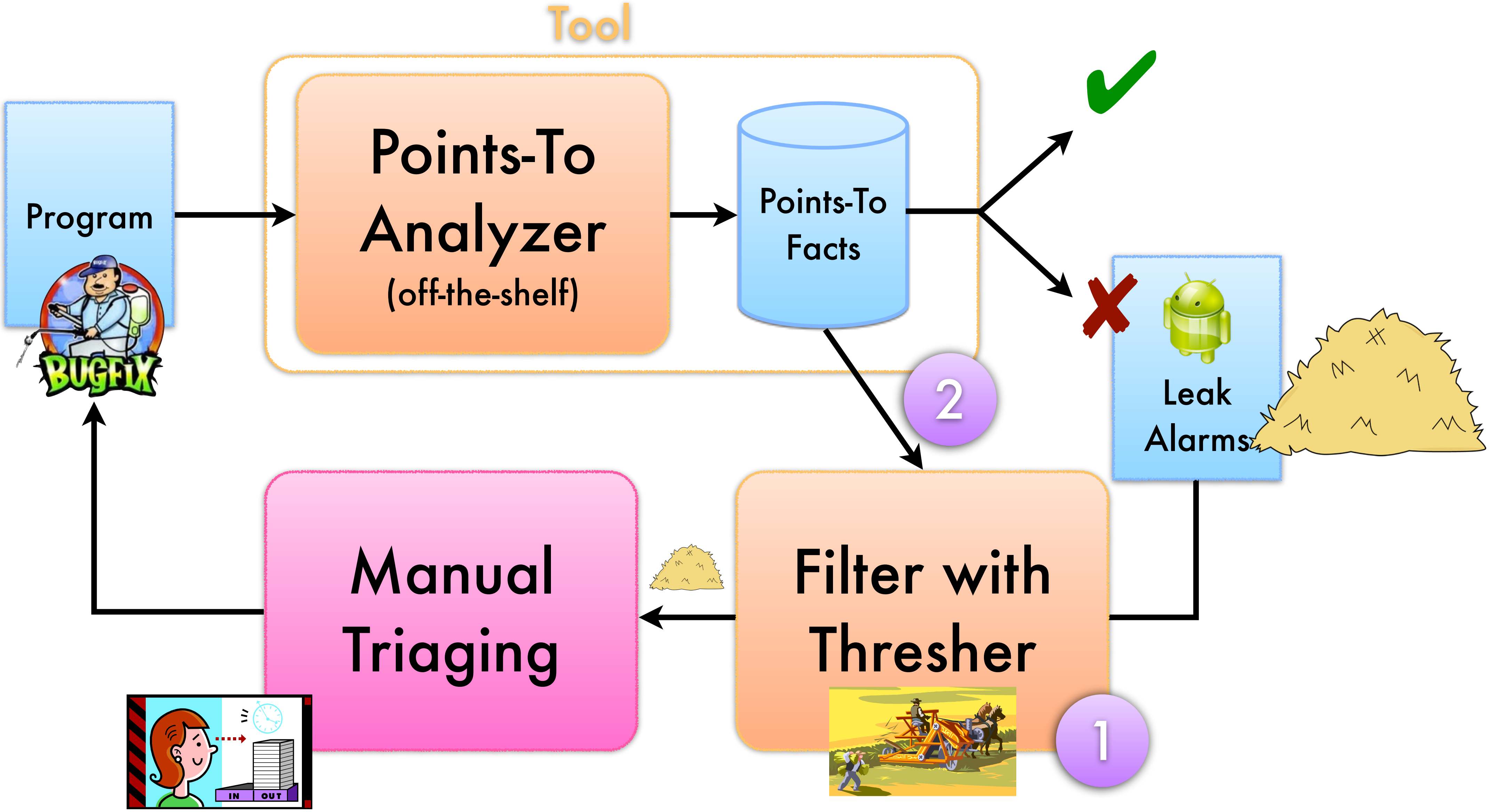
```
x.f = p
```

$$\boxed{\mathtt{y \cdot f \mapsto p}}$$

Generalized disalias check:

$\mathrm{pt(x) \cap pt(y) = \varnothing}$

Roadmap: Thresher filters out false alarms by refuting them one-by-one.



Idea 1: Refute points-to on-demand with second precise "filter" analysis

Idea 2: Leverage the facts from the first analysis in the filter analysis to scale

Is Thresher effective at filtering?

1

Thresher analyzes Java VM bytecode

7 Android app benchmarks

2,000 to 40,000 source lines of code

+ 880,000 sources lines of Android framework code

Off-the-shelf, state-of-the-art points-to analysis from WALA

Is Thresher effective at filtering?

1

Program	LOC	Points-To Alarms
PulsePoint	unknown	16
StandupTimer	2K	25
DroidLife	3K	3
SMSPopUp	7K	5
aMetro	20K	54
K9Mail	40K	208
Total	**72K**	**311**

Is Thresher effective at filtering?

1

Program	LOC	Points-To Alarms
PulsePoint	unknown	16
StandupTimer	2K	25
DroidLife	3K	3
SMSPopUp	7K	5
aMetro	20K	54
K9Mail	40K	208
Total	**72K**	**311**

`staticfield-Activity` pairs

Is Thresher effective at filtering?

1

Program	LOC	Points-To Alarms	Thresher Refuted
PulsePoint	unknown	16	8
StandupTimer	2K	25	15
DroidLife	3K	3	0
SMSPopUp	7K	5	1
aMetro	20K	54	18
K9Mail	40K	208	130
Total	**72K**	**311**	**172**

`staticfield-Activity` pairs

Filtered

Is Thresher effective at filtering?

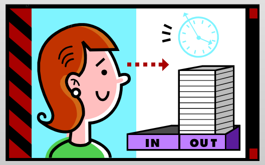
1

Program	LOC	Points-To Alarms	Thresher Refuted	True Bugs
PulsePoint	unknown	16	8	8
StandupTimer	2K	25	15	0
DroidLife	3K	3	0	3
SMSPopUp	7K	5	1	4
aMetro	20K	54	18	36
K9Mail	40K	208	130	64
Total	**72K**	**311**	**172**	**115**

staticfield-Activity pairs

Filtered

Manual



Is Thresher effective at filtering?

1

Program	LOC	Points-To Alarms	Thresher Refuted	True Bugs
PulsePoint	unknown	16	8	8
StandupTimer	2K	25	15	0
DroidLife	3K	3	0	3
SMSPopUp	7K	5	1	4
aMetro	20K	54	18	36
K9Mail	40K	208	130	64
Total	**72K**	**311**	**172**	**115**

Is Thresher effective at filtering?

1

Program	LOC	Points-To Alarms	Thresher Refuted	True Bugs	Thresher Time (s)
PulsePoint	unknown	16	8	8	95
StandupTimer	2K	25	15	0	1068
DroidLife	3K	3	0	3	1
SMSPopUp	7K	5	1	4	46
aMetro	20K	54	18	36	18
K9Mail	40K	208	130	64	374
Total	**72K**	**311**	**172**	**115**	**1602**

Is Thresher effective at filtering?

1

Program	LOC	Points-To Alarms	Thresher Refuted	True Bugs	Thresher Time (s)
PulsePoint	unknown	16	8	8	95
StandupTimer	2K	25	15	0	1068
DroidLife	3K	3	0	3	1
SMSPopUp	7K	5	1	4	46
aMetro	20K	54	18	36	18
K9Mail	40K	208	130	64	374
Total	**72K**	**311**	**172**	**115**	**1602**

< ~coffee to lunch break

Is Thresher effective at filtering?

1

Program	LOC	Points-To Alarms	Thresher Refuted	True Bugs	Thresher Time (s)
PulsePoint	unknown	16	8	8	95
StandupTimer	2K	25	15	0	1068
DroidLife	3K	3	0	3	1
SMSPopUp	7K	5	1	4	46
aMetro	20K	54	18	36	18
K9Mail	40K	208	130	64	374
Total	**72K**	**311**	**172**	**115**	**1602**

Is Thresher effective at filtering?

1

Program	LOC	Points-To Alarms	Thresher Refuted	True Bugs	Thresher Time (s)	False Alarm %
PulsePoint	unknown	16	8	8	95	0
StandupTimer	2K	25	15	0	1068	100
DroidLife	3K	3	0	3	1	0
SMSPopUp	7K	5	1	4	46	0
aMetro	20K	54	18	36	18	0
K9Mail	40K	208	130	64	374	18
Total	**72K**	**311**	**172**	**115**	**1602**	**17**

% after filtering

Is Thresher effective at filtering?

1

Program	LOC	Points-To Alarms	Thresher Refuted	True Bugs	Thresher Time (s)	False Alarm %	Filtered %
PulsePoint	unknown	16	8	8	95	0	100
StandupTimer	2K	25	15	0	1068	100	60
DroidLife	3K	3	0	3	1	0	-
SMSPopUp	7K	5	1	4	46	0	100
aMetro	20K	54	18	36	18	0	100
K9Mail	40K	208	130	64	374	18	90
Total	**72K**	**311**	**172**	**115**	**1602**	**17**	**88**

% after filtering

Is Thresher effective at filtering?

1

Program	LOC	Points-To Alarms	Thresher Refuted	True Bugs	Thresher Time (s)	False Alarm %	Filtered %
PulsePoint	unknown	16	8	8	95	0	100
StandupTimer	2K	25	15	0	1068	100	60
DroidLife	3K	3	0	3	1	0	-
SMSPopUp	7K	5	1	4	46	0	100
aMetro	20K	54	18	36	18	0	100
K9Mail	40K	208	130	64	374	18	90
Total	**72K**	**311**	**172**	**115**	**1602**	**17**	**88**

Is Thresher effective at filtering?

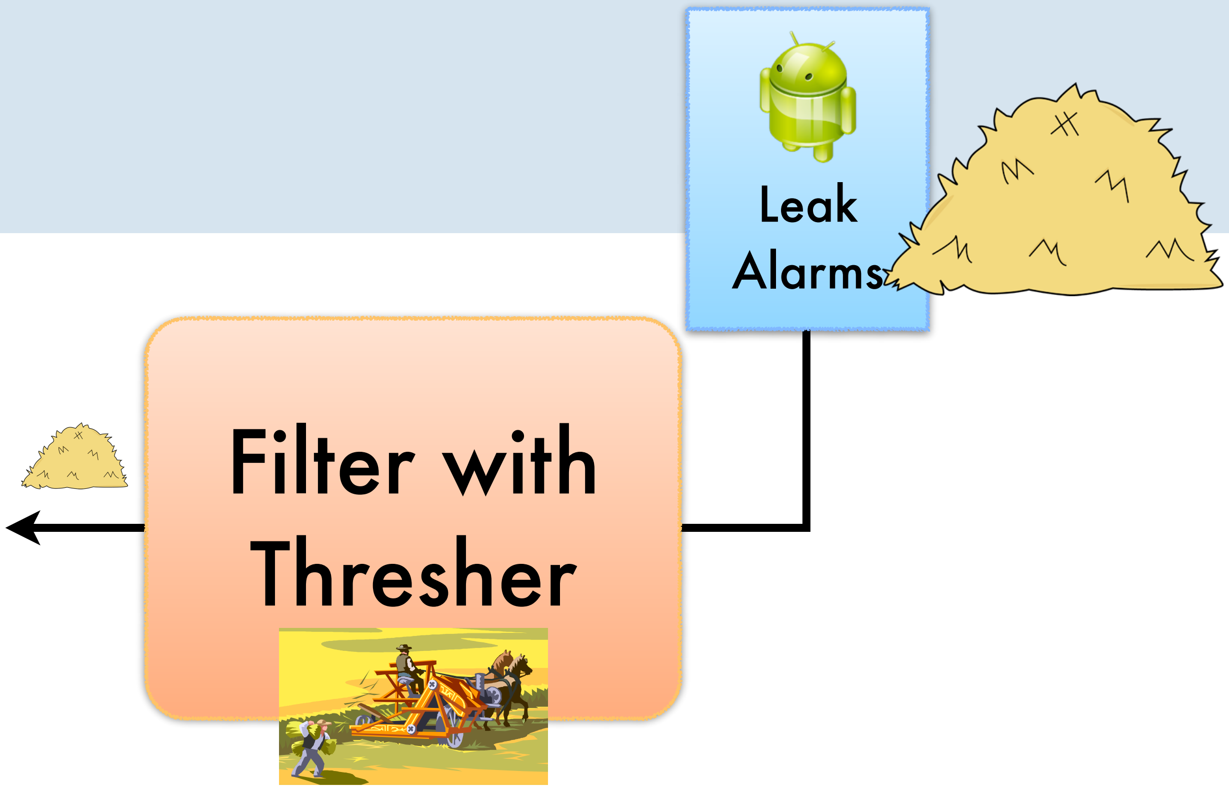
1

Program	LOC	Points-To Alarms	Thresher Refuted	True Bugs	Thresher Time (s)	False Alarm %	Filtered %
PulsePoint	unknown	16	8	8	95	0	100
StandupTimer	2K	25	15	0	1068	100	60
DroidLife	3K	3	0	3	1	0	-
SMSPopUp	7K	5	1	4	46	0	100
aMetro	20K	54	18	36	18	0	100
K9Mail	40K	208	130	64	374	18	90
Total	**72K**	**311**	**172**	**115**	**1602**	**17**	**88**

False alarms down to 17% from 63% (points-to analysis only)

Thresher filters 88% of false alarms from points-to analysis

Some Highlights

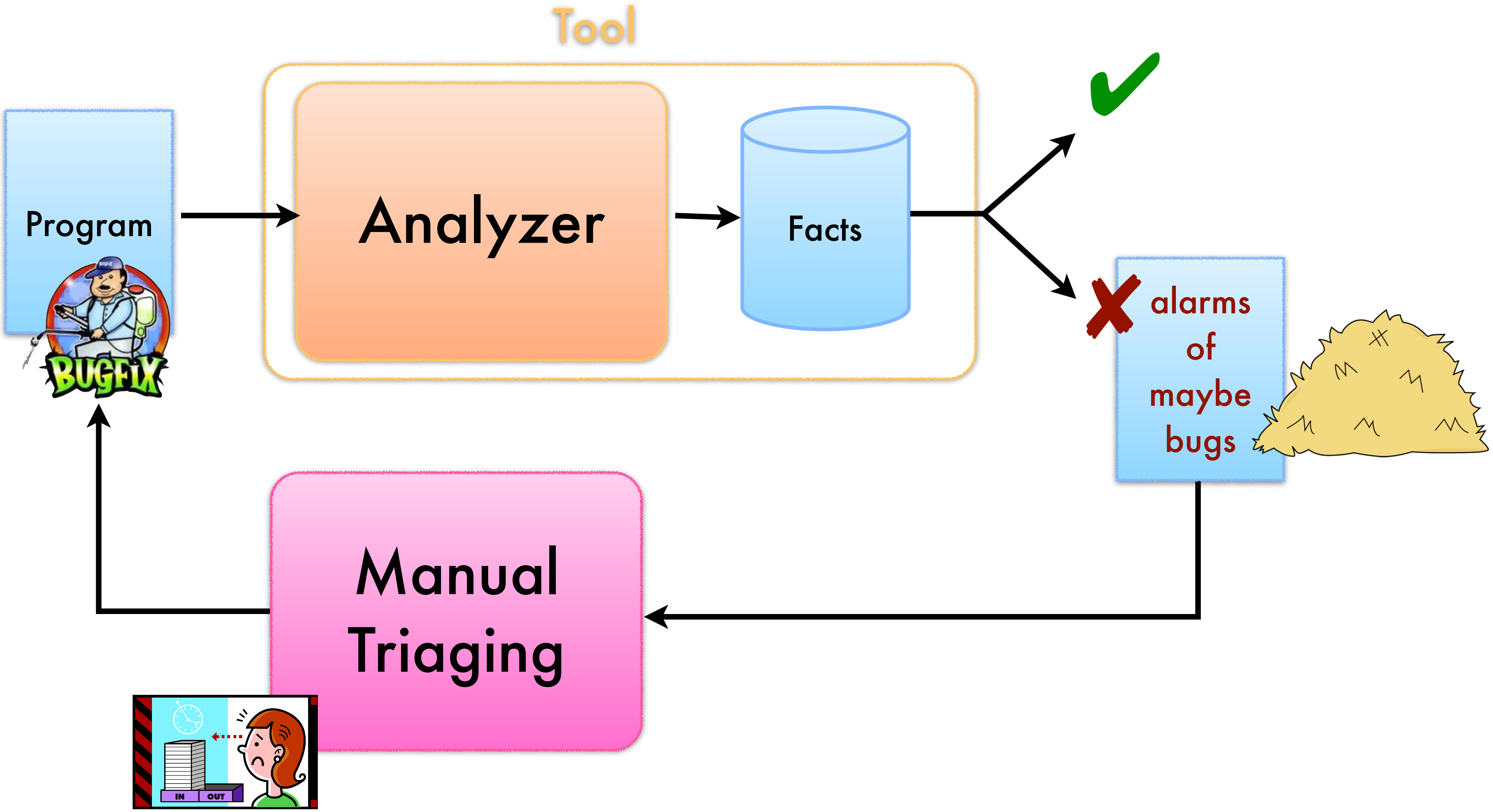


Thresher: Precise Refutations for Heap Reachability

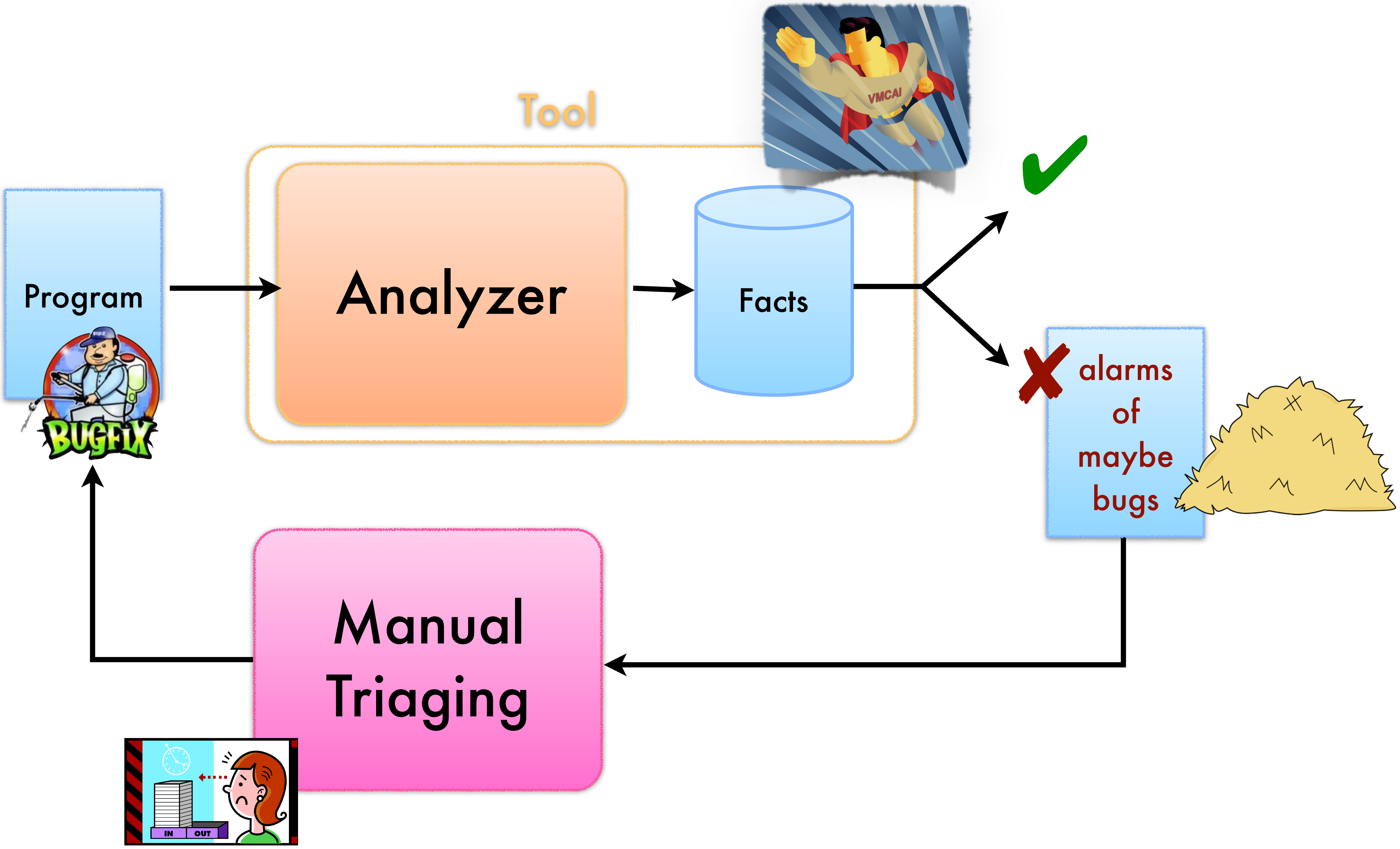
Assist in triage of queries about heap relations

- Assume alarms false, prove them so (refute) automatically with a “partial” witness search
- Reduced separation constraints with points-to facts
- Filters out ~90% of false alarms to expose true bugs
- Application: Find memory leaks and eliminate crashes in Android

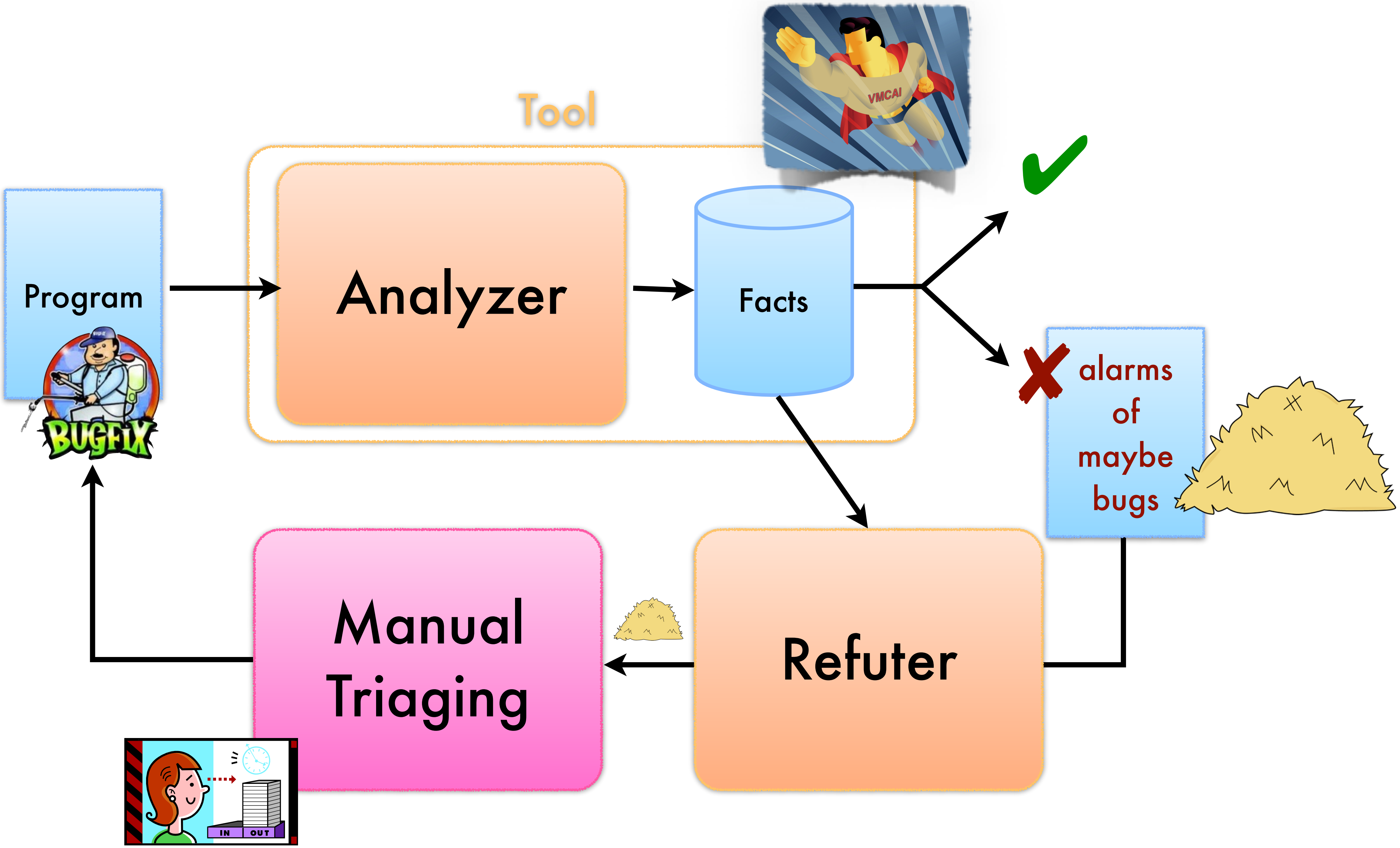
Final Commentary: Design and apply analyses to the whole bug mitigation process!



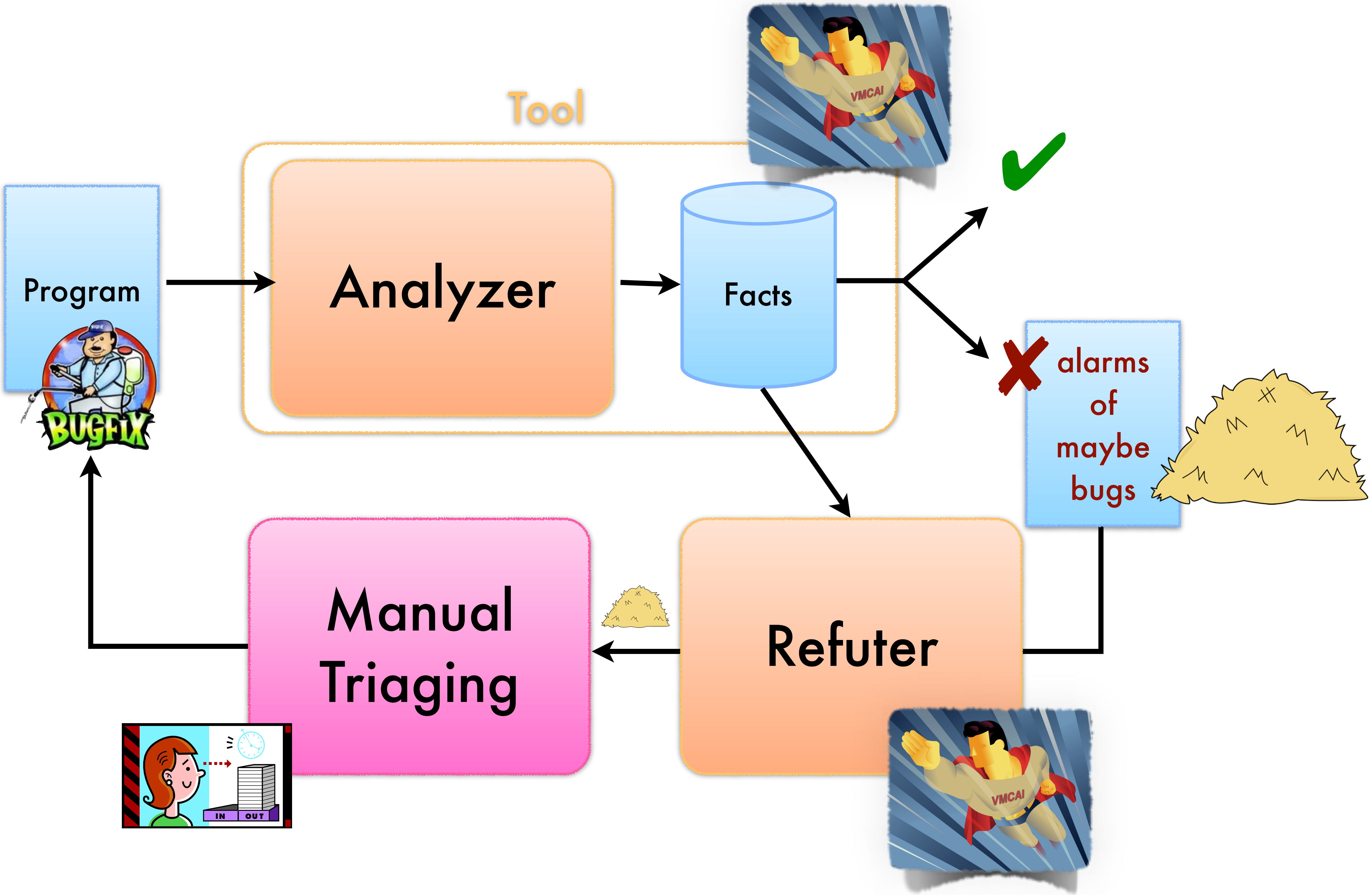
Final Commentary: Design and apply analyses to the whole bug mitigation process!



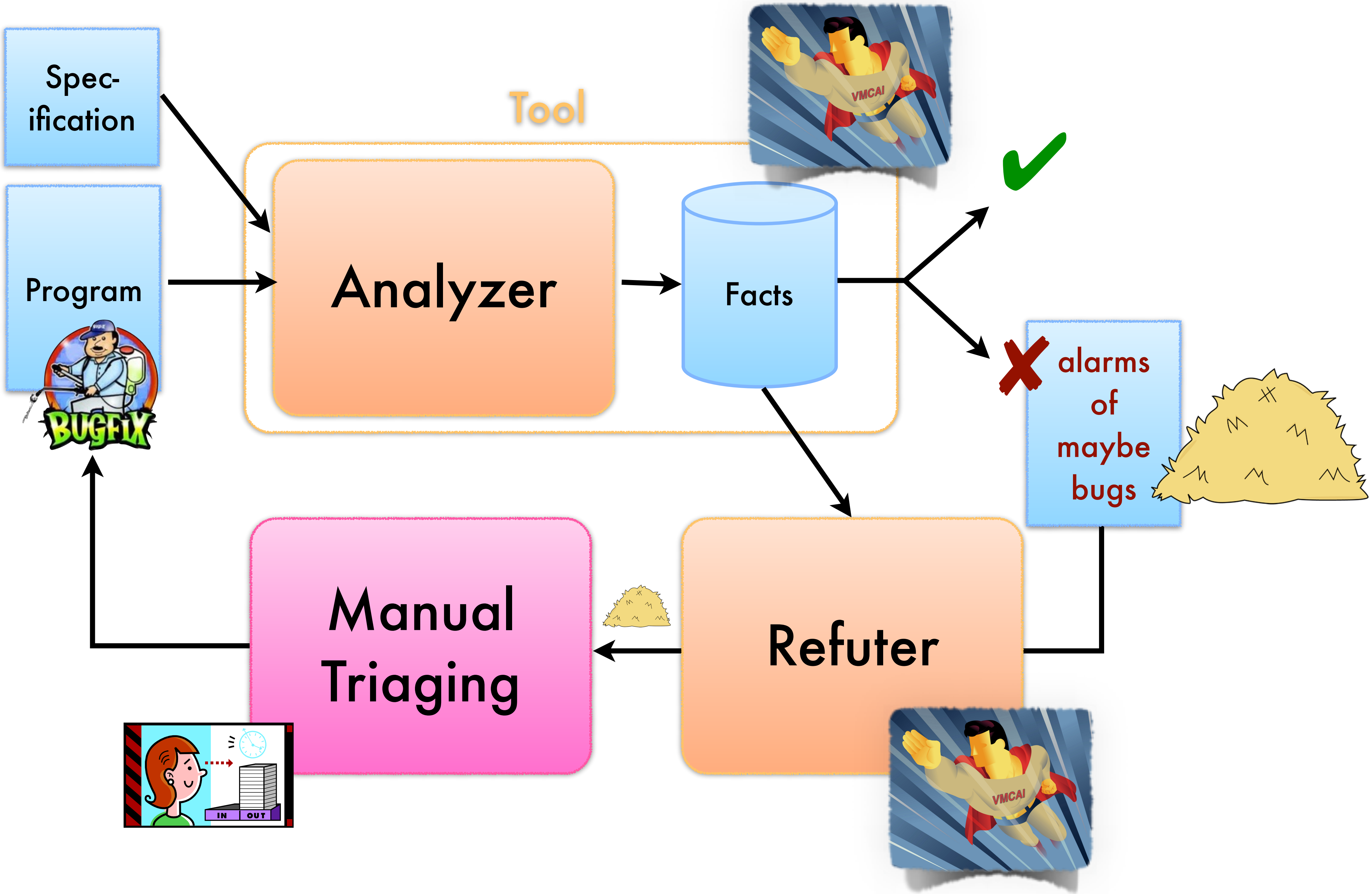
Final Commentary: Design and apply analyses to the whole bug mitigation process!



Final Commentary: Design and apply analyses to the whole bug mitigation process!



Final Commentary: Design and apply analyses to the whole bug mitigation process!



Spec-ification

Program

Tool

Analyzer

Facts

✔

✘ alarms of maybe bugs

Manual Triaging

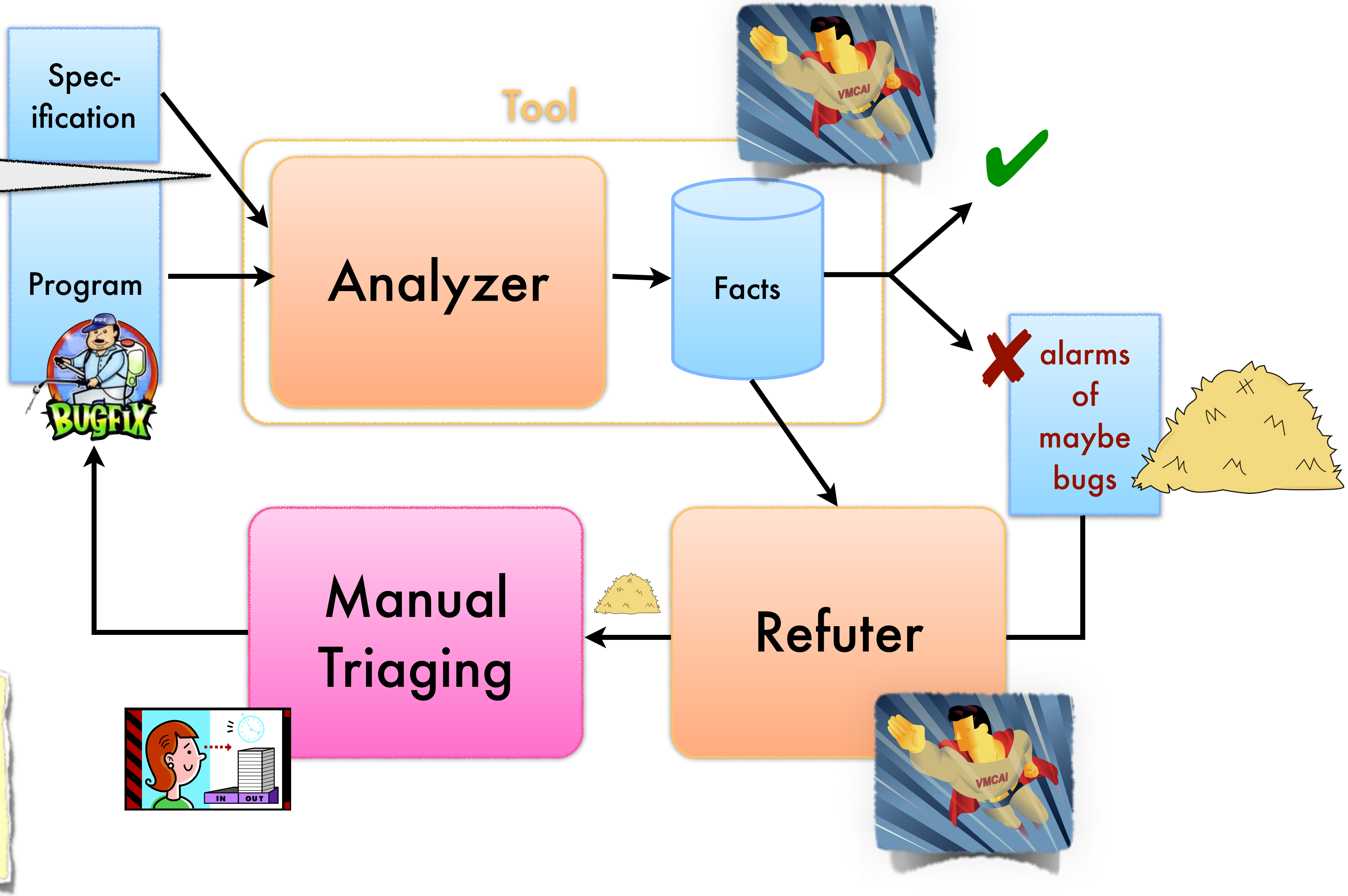
Refuter

Final Commentary: Design and apply analyses to the whole bug mitigation process!

Fissile Types: Checking Almost Everywhere Invariants

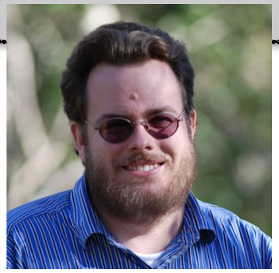
[Coughlin+ POPL'14]

Wed
11:45am

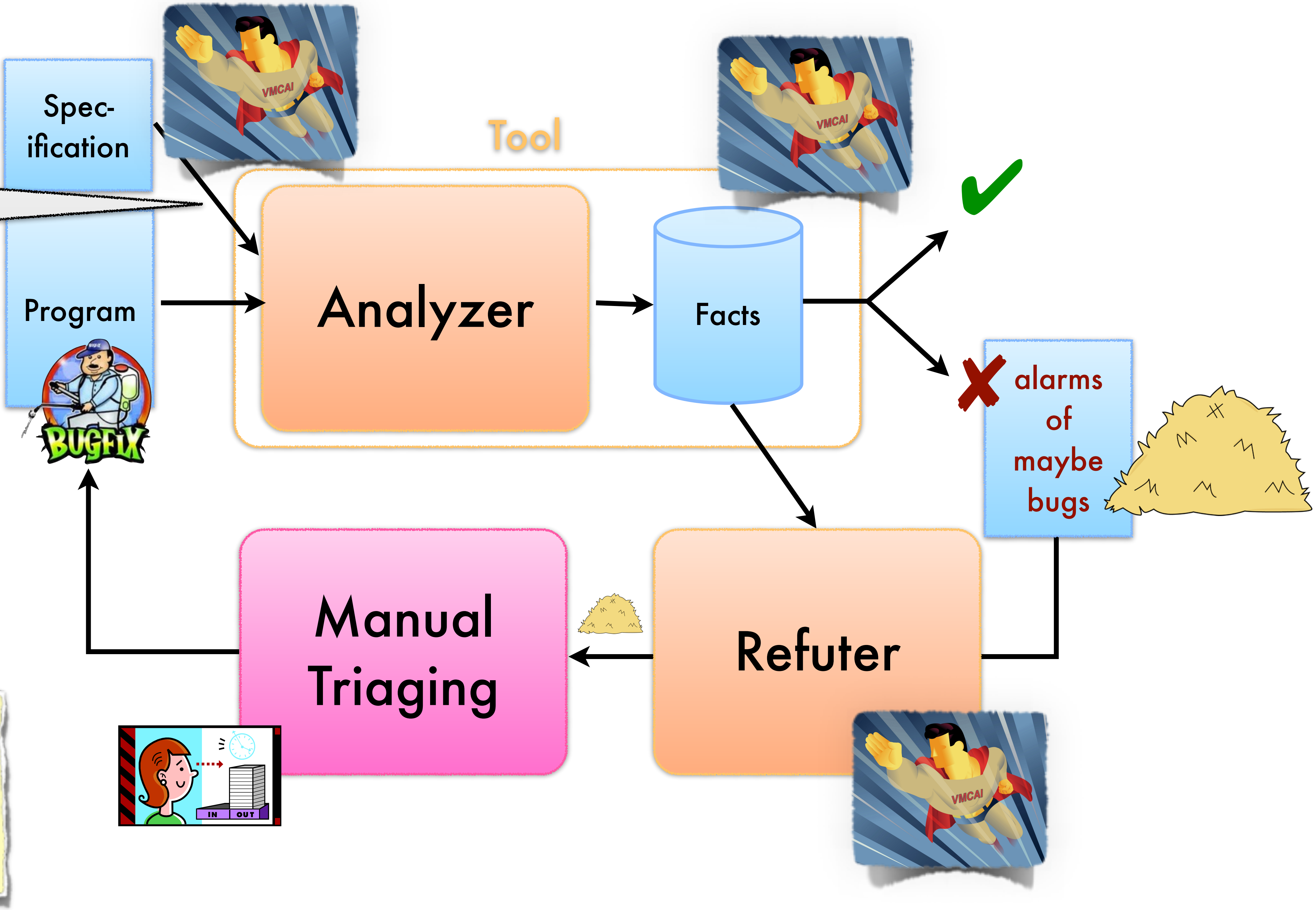


Final Commentary: Design and apply analyses to the whole bug mitigation process!

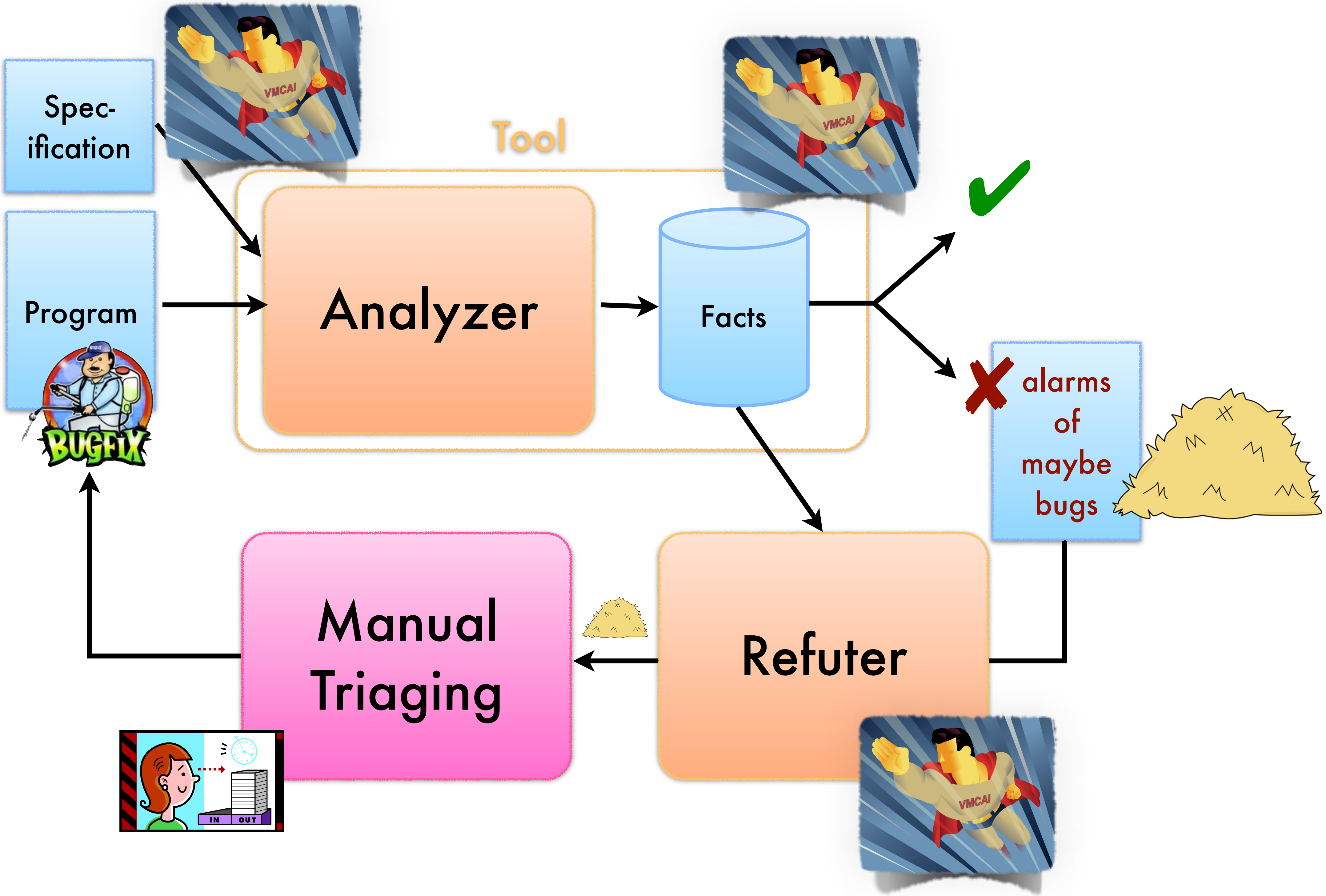
Fissile Types: Checking Almost Everywhere Invariants
[Coughlin+ POPL'14]



Wed
11:45am



Final Commentary: Design and apply analyses to the whole bug mitigation process!



Spec-ification

Program

Tool

Analyzer

Facts

✔

✘ alarms of maybe bugs

Manual Triaging

Refuter

Thank You!

PLV

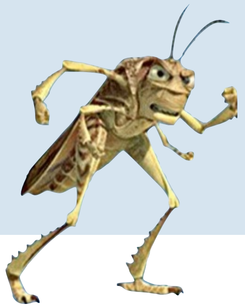
www.cs.colorado.edu/~bec
pl.cs.colorado.edu

Android
OS

Android
OS

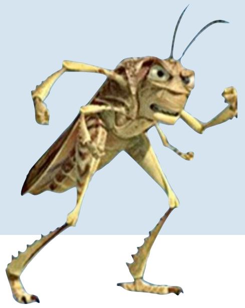
... in the process of finding leaks in apps

Find Android's HashMap bug ...



```
class HashMap {
  static Object[] EMPTY = new Object[2]; ...
  HashMap() { this.tbl = EMPTY; capacity initially empty }

  void put(Object key, Object val) {
    if (need capacity) {
      this.tbl = new Object[more capacity];
      copy from old table
    }
    this.tbl[bucket using hash of key] = val;
  }

  HashMap(Map m) {
    if (m.size() < 1) { this.tbl = EMPTY; }
    else { this.tbl = new Object[at least m.size()]; }
    copy from m
  }
}
```

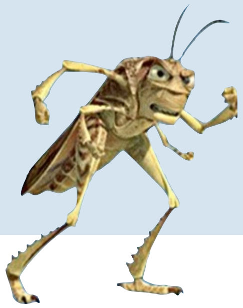
Find Android's HashMap bug ...



Null object pattern: Should never be written to

```
class HashM
  static Object[] EMPTY = new Object[2]; ...
  HashMap() { this.tbl = EMPTY; capacity initially empty }

  void put(Object key, Object val) {
    if (need capacity) {
      this.tbl = new Object[more capacity];
      copy from old table
    }
    this.tbl[bucket using hash of key] = val;
  }

  HashMap(Map m) {
    if (m.size() < 1) { this.tbl = EMPTY; }
    else { this.tbl = new Object[at least m.size()]; }
    copy from m
  }
}
```

Find Android's HashMap bug ...



Null object pattern: Should never be written to

```
class HashM
  static Object[] EMPTY = new Object[2]; ...
  HashMap() { this.tbl = EMPTY; capacity initially empty }

  void put(Object key, Object val) {
    if (need capacity) {
      this.tbl = new Object[more capacity];
      copy from old table
    }
    this.tbl[bucket using hash of key] = val;
  }

  HashMap(Map m) {
    if (m.size() < 1) { this.tbl = EMPTY; }
    else { this.tbl = new Object[at least m.size()]; }
    copy from m
  }
}
```

allocate new backing array on first write

Find Android's HashMap bug ...

Null object pattern: Should never be written to

```
class HashM
  static Object[] EMPTY = new Object[2]; ...
  HashMap() { this.tbl = EMPTY; capacity initially empty }

  void put(Object key, Object val) {
    if (need capacity) {
      this.tbl = new Object[more capacity];
      copy from old table
    }
    this.tbl[bucket using hash of key] = val;
  }

  HashMap(Map m) {
    if (m.size() < 1) { this.tbl = EMPTY; }
    else { this.tbl = new Object[at least m.size()]; }
    copy from m
  }
}
```

allocate new backing array on first write

Find Android’s HashMap bug …

Null object pattern: Should never be written to

```
class HashM
  static Object[] EMPTY = new Object[2]; ...
  HashMap() { this.tbl = EMPTY; capacity initially empty }

  void put(Object key, Object val) {
    if (need capacity) {
      this.tbl = new Object[more capacity];
      copy from old table
    }
    this.tbl[bucket using hash of key] = val;
  }

  HashMap(Map m) {
    if (m.size() < 1) { this.tbl = EMPTY; }
    else { this.tbl = new Object[at least m.size()]; }
    copy from m
  }
}
```

allocate new backing array on first write

An “evil” implementation of the `Map` interface can corrupt `EMPTY`. Then, all `HashMap`s created in the future will be corrupted.

Find Android’s HashMap bug …

Null object pattern: Should never be written to

```
class HashM
  static Object[] EMPTY = new Object[2]; ...
  HashMap() { this.tbl = EMPTY; capacity initially empty }

  void put(Object key, Object val) {
    if (need capacity) {
      this.tbl = new Object[more capacity];
      copy from old table
    }
    this.tbl[bucket using hash of key] = val;
  }

  HashMap(Map m) {
    if (m.size() < 1) { this.tbl = EMPTY; }
    else { this.tbl = new Object[at least m.size()]; }
    copy from m
  }
}
```

allocate new backing array on first write

return 0

return “evil” content

An “evil” implementation of the `Map` interface can corrupt `EMPTY`. Then, all `HashMap`s created in the future will be corrupted.

Find Android’s HashMap bug …

Null object pattern: Should never be written to

```
class HashM
 static Object[] EMPTY = new Object[2]; ...
 HashMap() { this.tbl = EMPTY; capacity initially empty }

 void put(Object key, Object val) {
  if (need capacity) {
   this.tbl = new Object[more capacity];
   copy from old table
  }
  this.
 }

 HashMap(Map m) {
  if (m.size() < 1) { this.tbl = EMPTY; }
  else { this.tbl = new Object[at least m.size()]; }
  copy from m
 }
}
```

allocate new backing array on first write

We reported this, Google fixed it

https://android-review.googlesource.com/#/c/52183/

return 0

return “evil” content

An “evil” implementation of the Map interface can corrupt EMPTY. Then, all HashMaps created in the future will be corrupted.