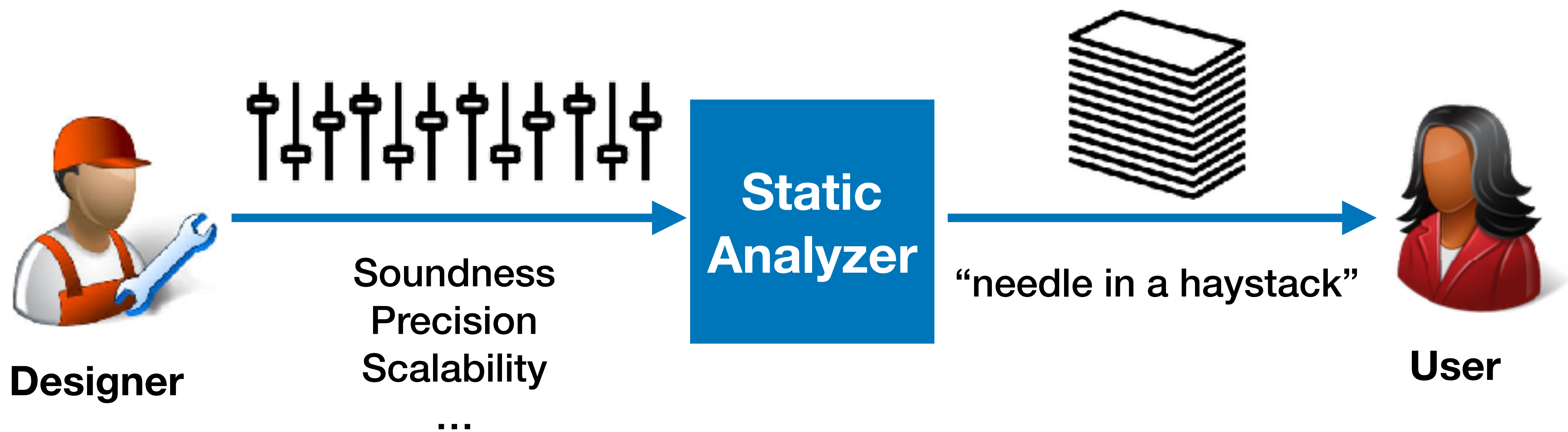
User-Guided Program Reasoning using Bayesian Inference

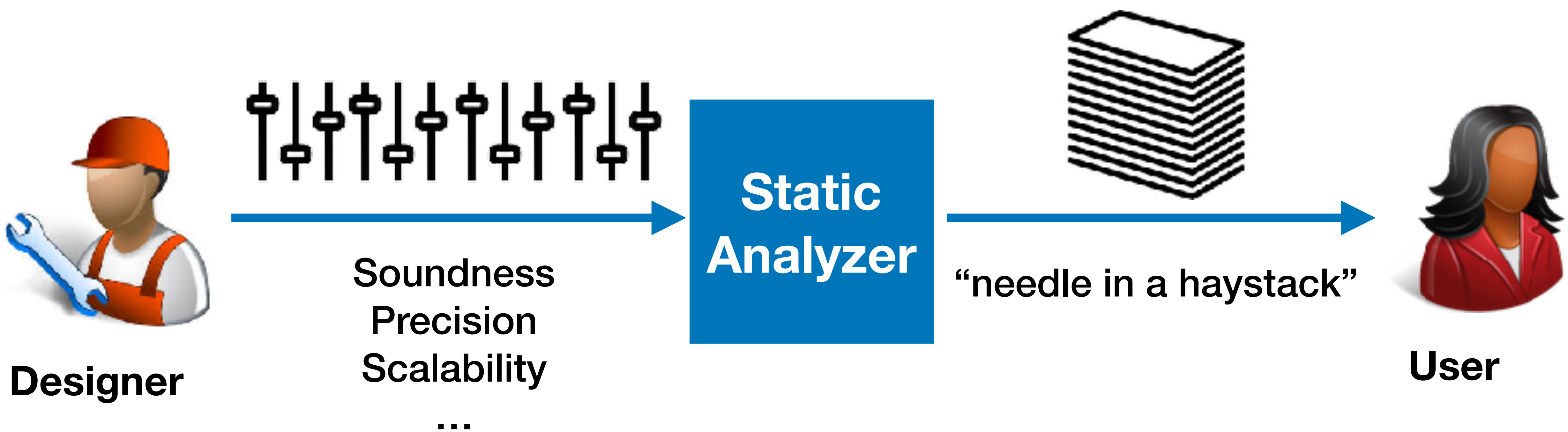
Kihong Heo
(joint work with Mukund Raghothaman, Sulekha Kulkarni, Mayur Naik)
University of Pennsylvania

Jul 6 2018 @ KAIST

Conventional Static Analysis



Why?



He does not know her:
"What is the optimal strategy regarding severity, context, idiom, etc?"

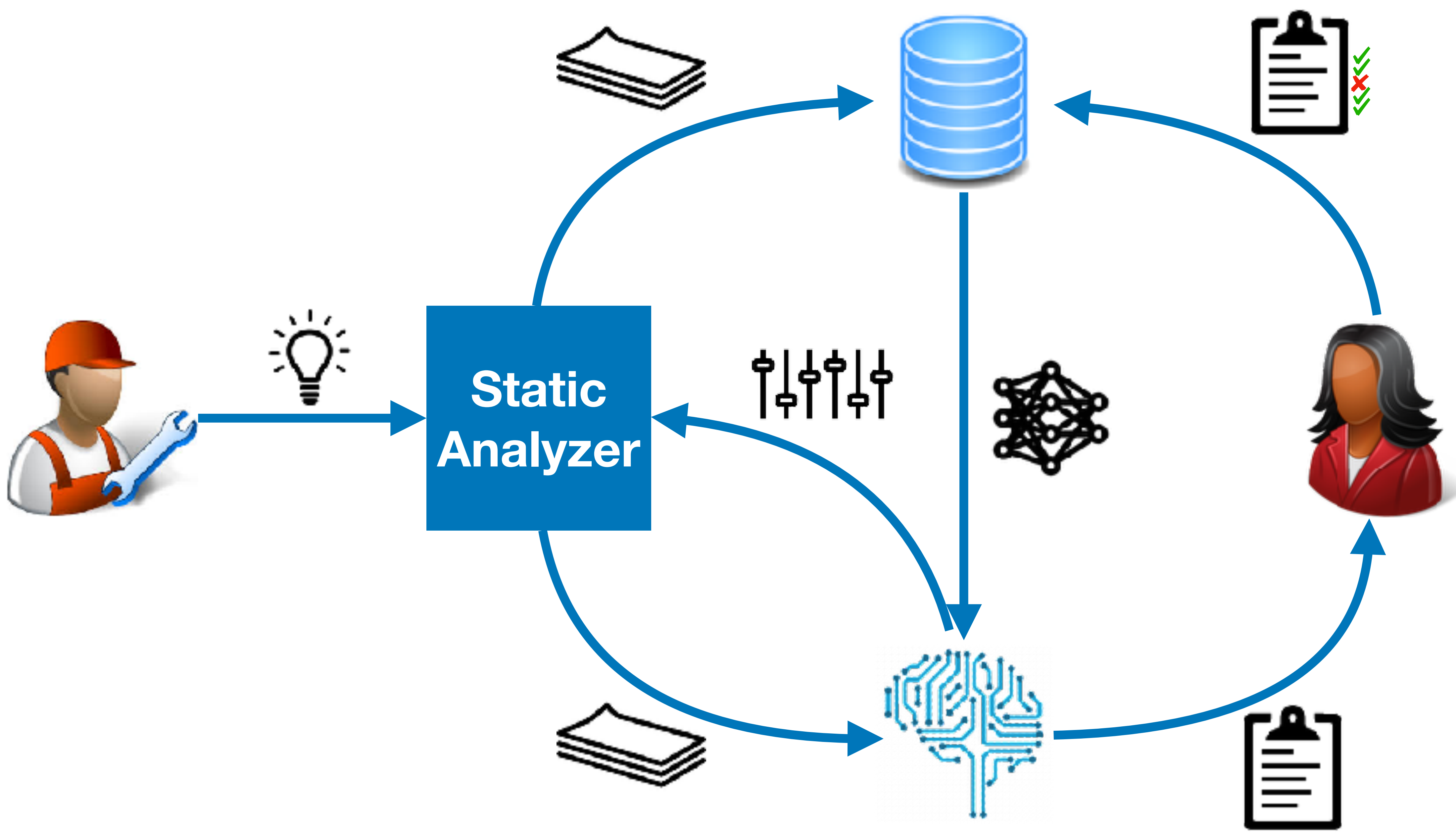
She does not know him:
"Why does this alarm occur?"
"How to avoid the similar false alarms?"

OpenSSL
Cryptography and SSL/TLS Toolkit

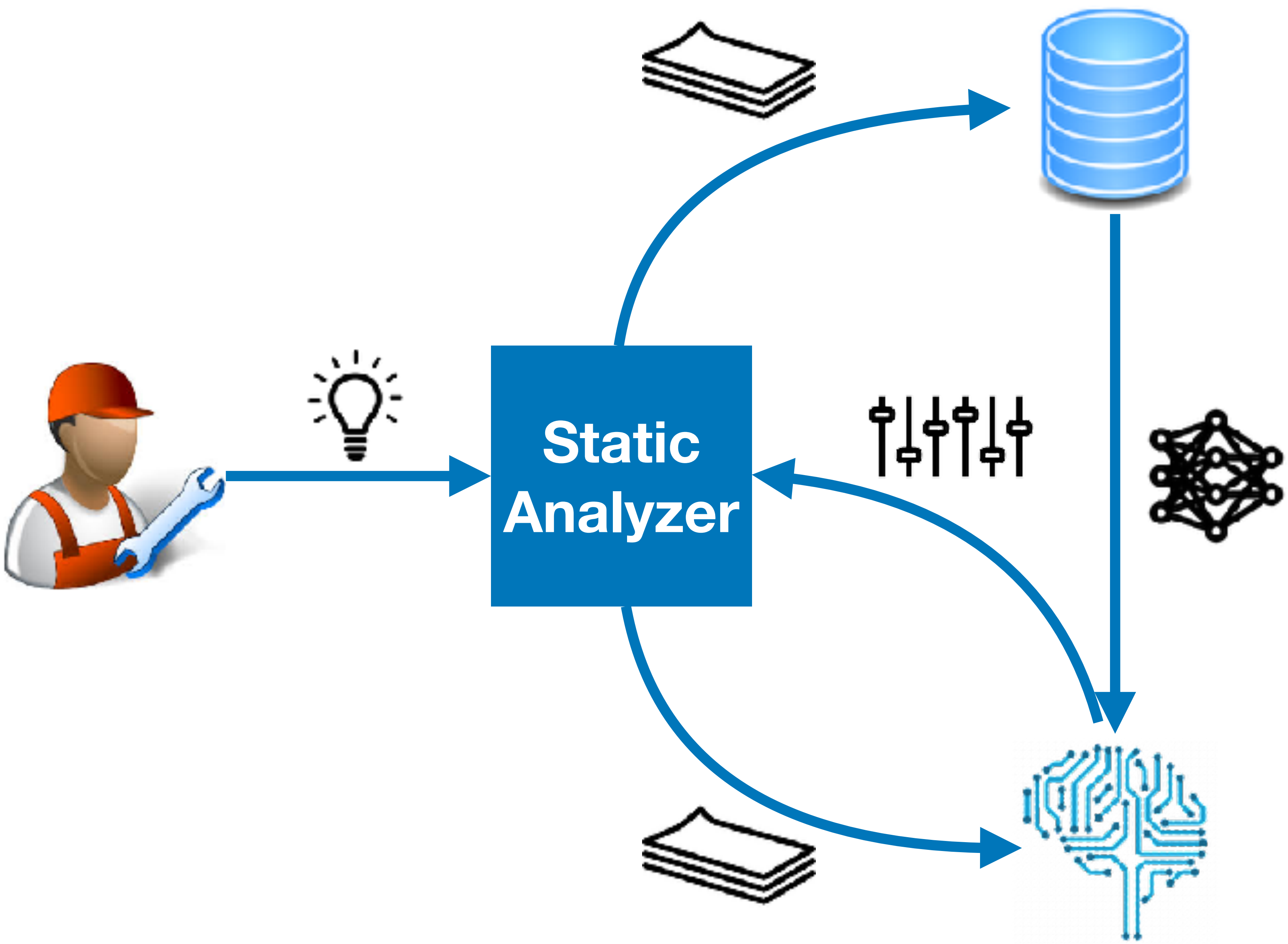
"… can be difficult to do without introducing large numbers of false positives, or scaling performance exponentially poorly. In this case, balancing these and other factors in the analysis design caused us to miss the defect."

— Coverity, *On Detecting Heartbleed with Static Analysis*, 2014

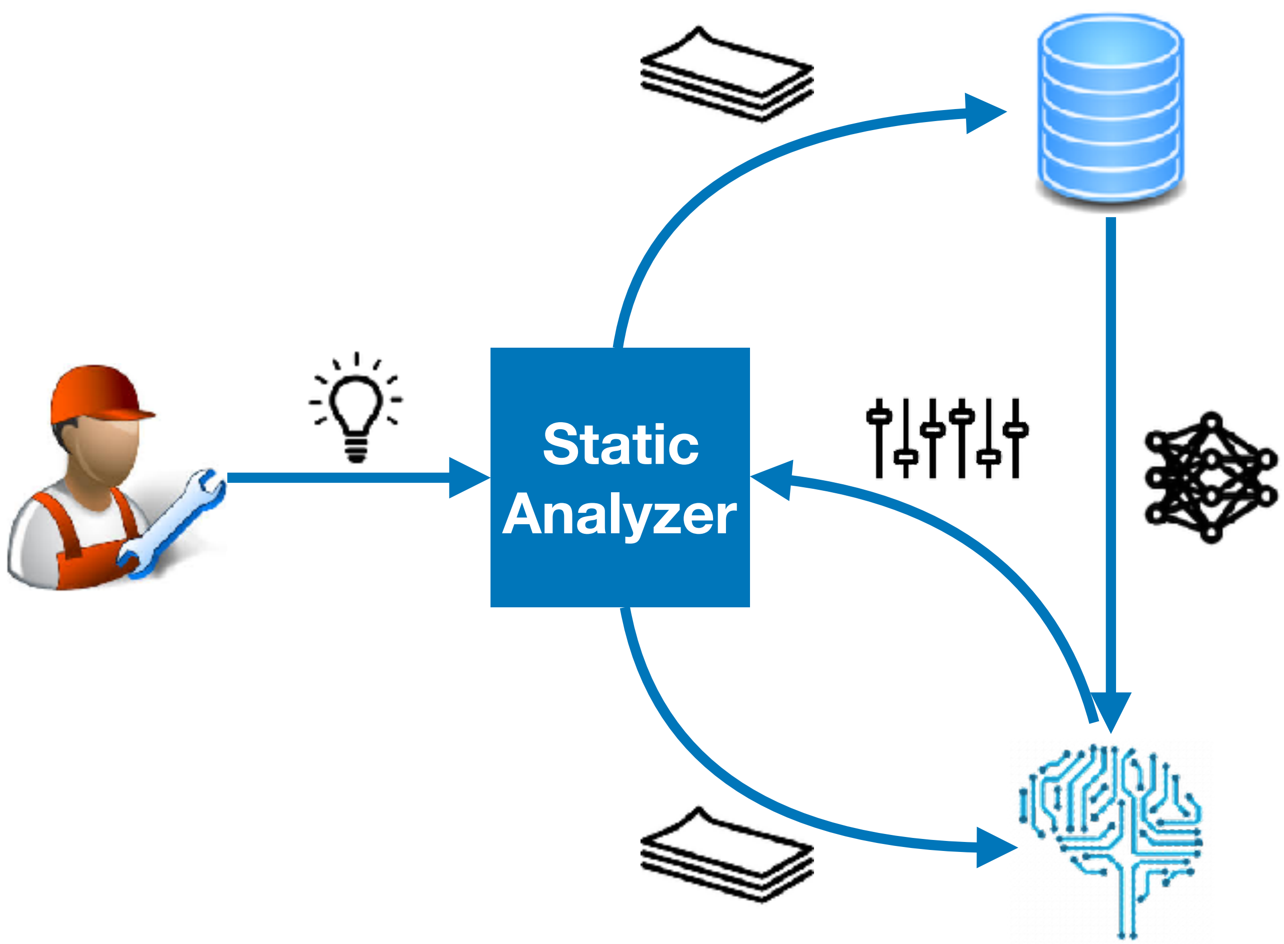
Next-generation Static Analysis



Next-generation Static Analysis



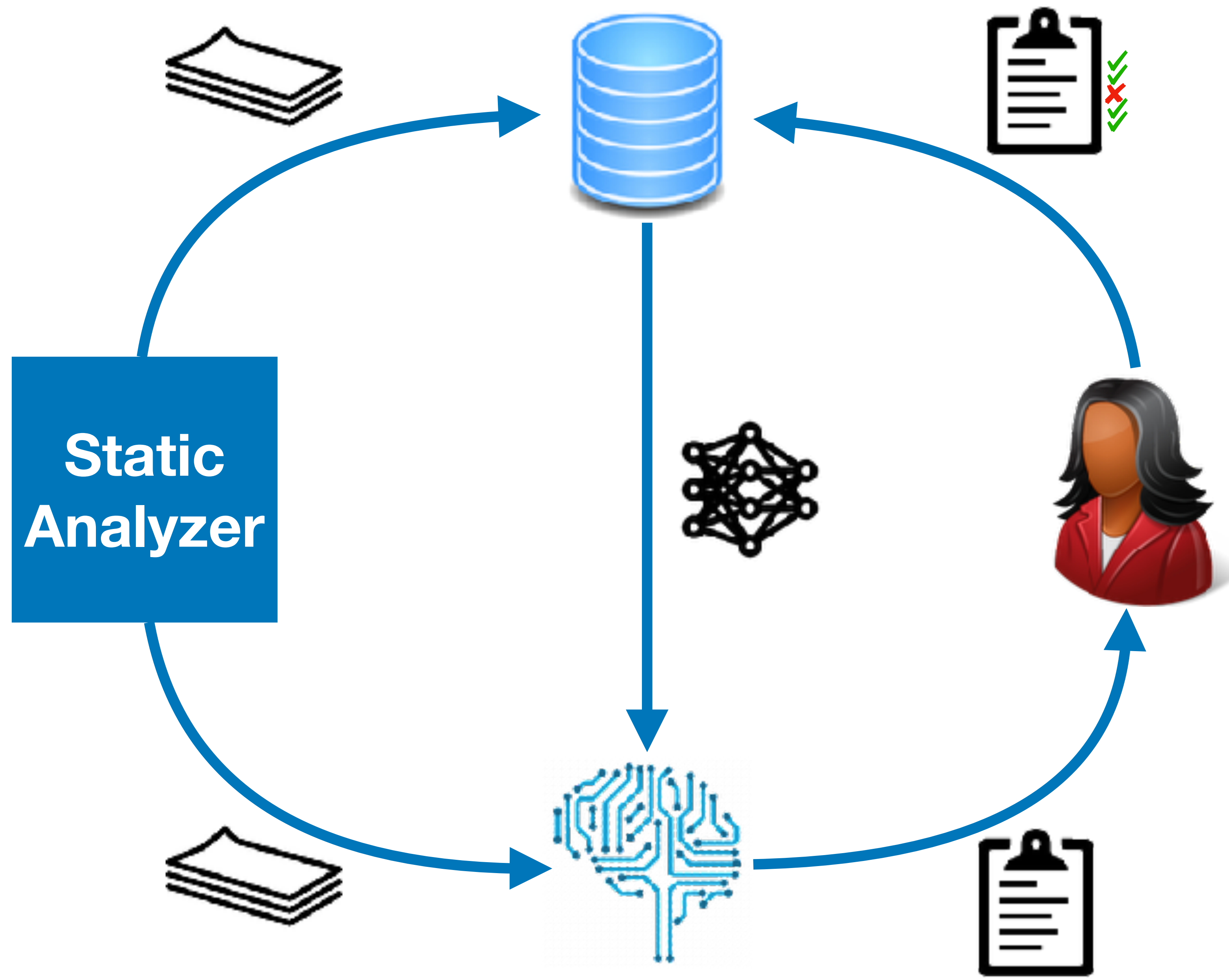
Next-generation Static Analysis



AI-based Analysis Design

- **Human** provides high-level idea
- **AI** provides detailed design choices
- **DB** accumulates performance data
- e.g.) precision [SAS'16,OOPSLA'17], soundness [ICSE'17], resource usage [in progress], rule learning [in progress]

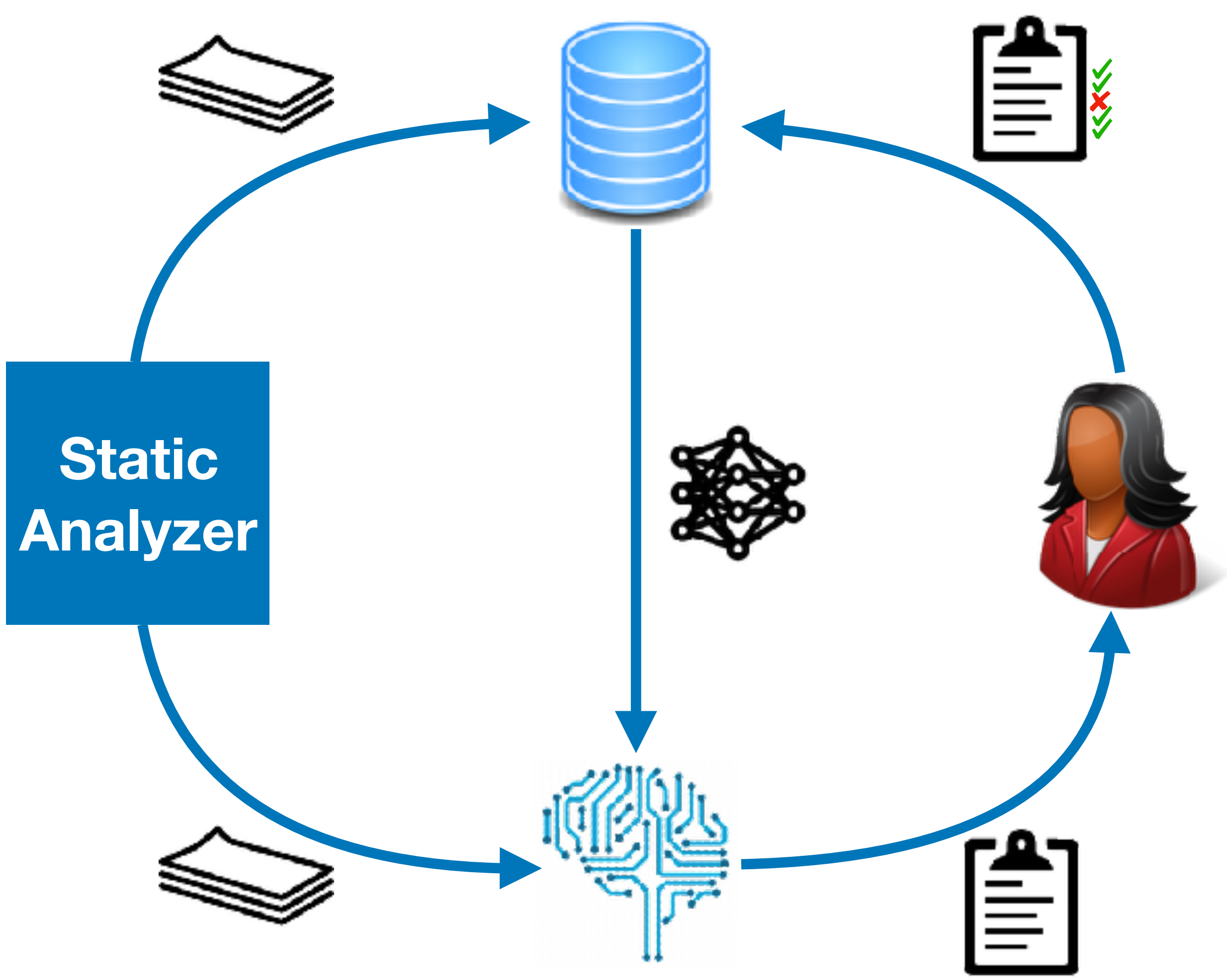
Next-generation Static Analysis



Next-generation Static Analysis

AI-based Alarm Report

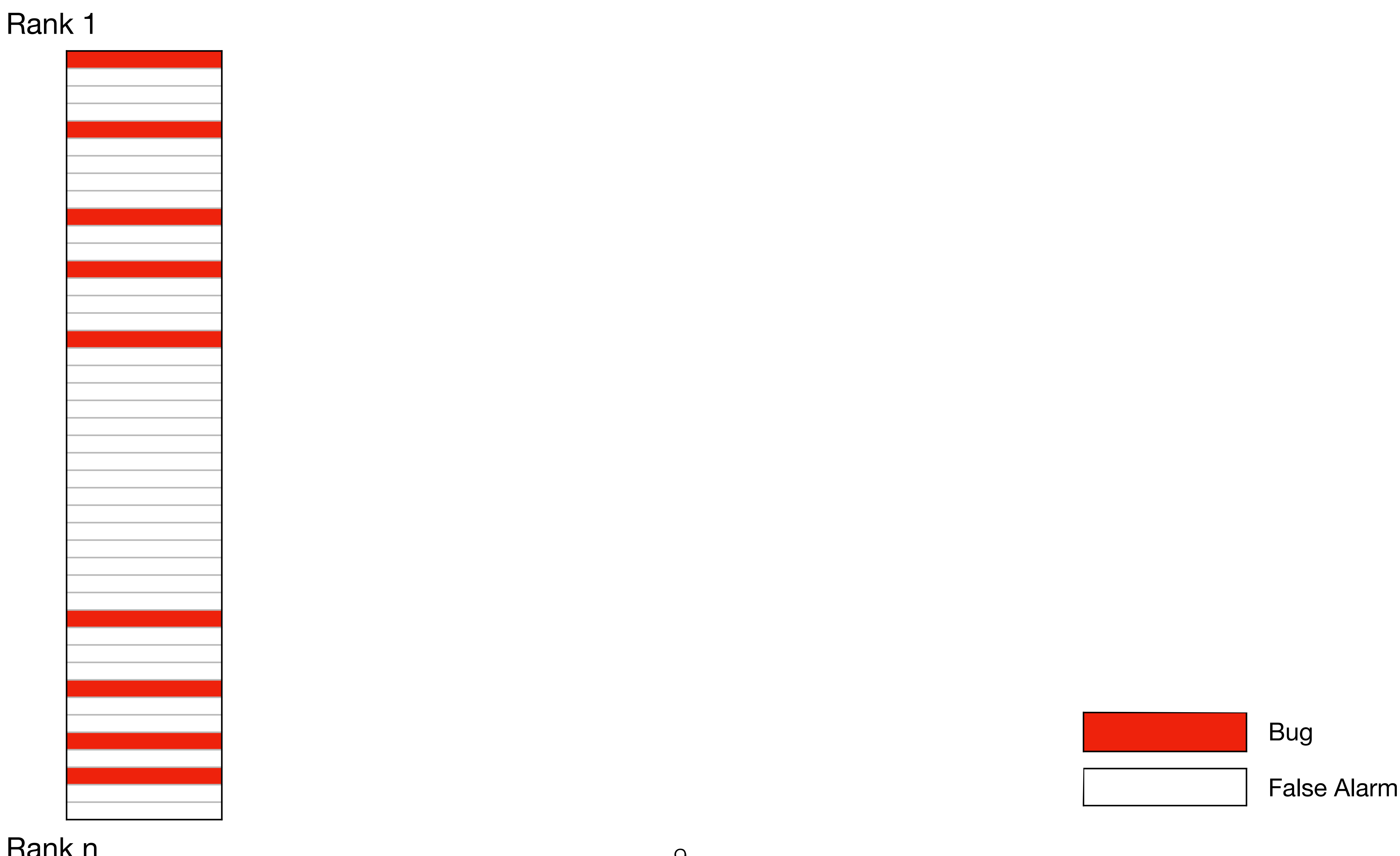
- **AI** prioritizes/classifies alarms
- **Human** inspects high confidence alarms
- **DB** accumulates human-labeled data
- e.g.) interactive alarm ranking [PLDI'18]



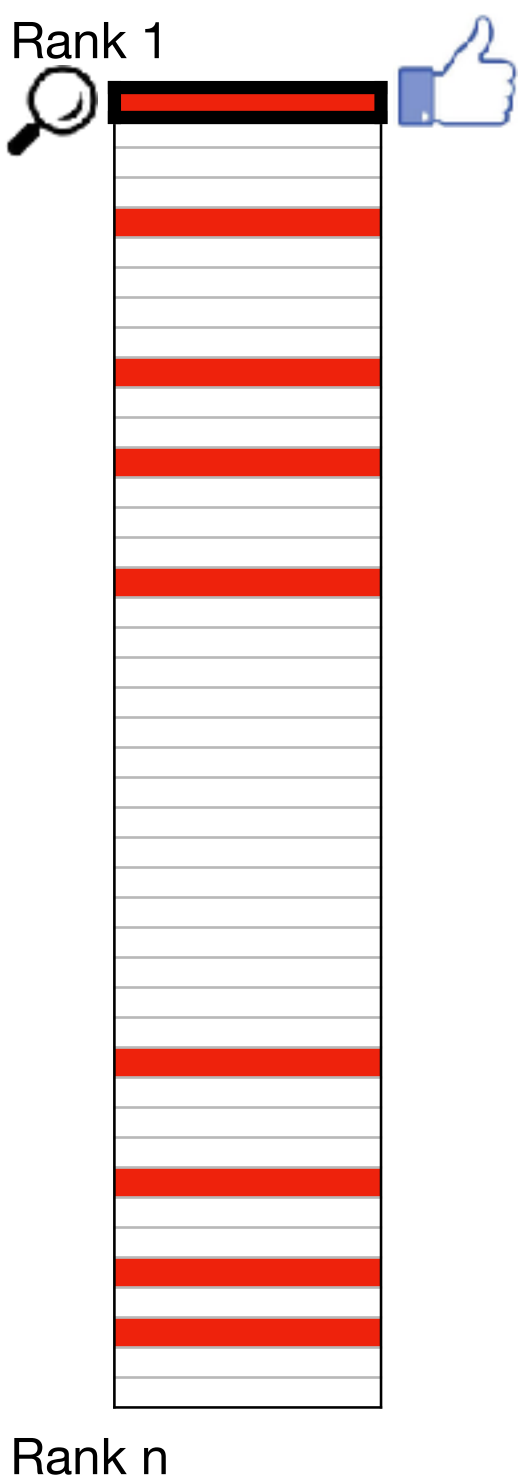
BINGO: An Interactive Alarm Ranking System

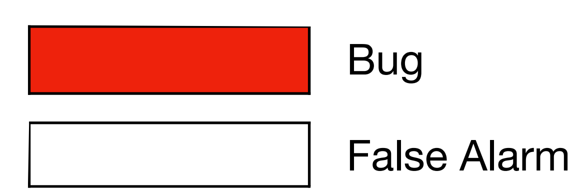
*User-Guided Program Reasoning using Bayesian Inference, PLDI'18

Interactive Alarm Ranker

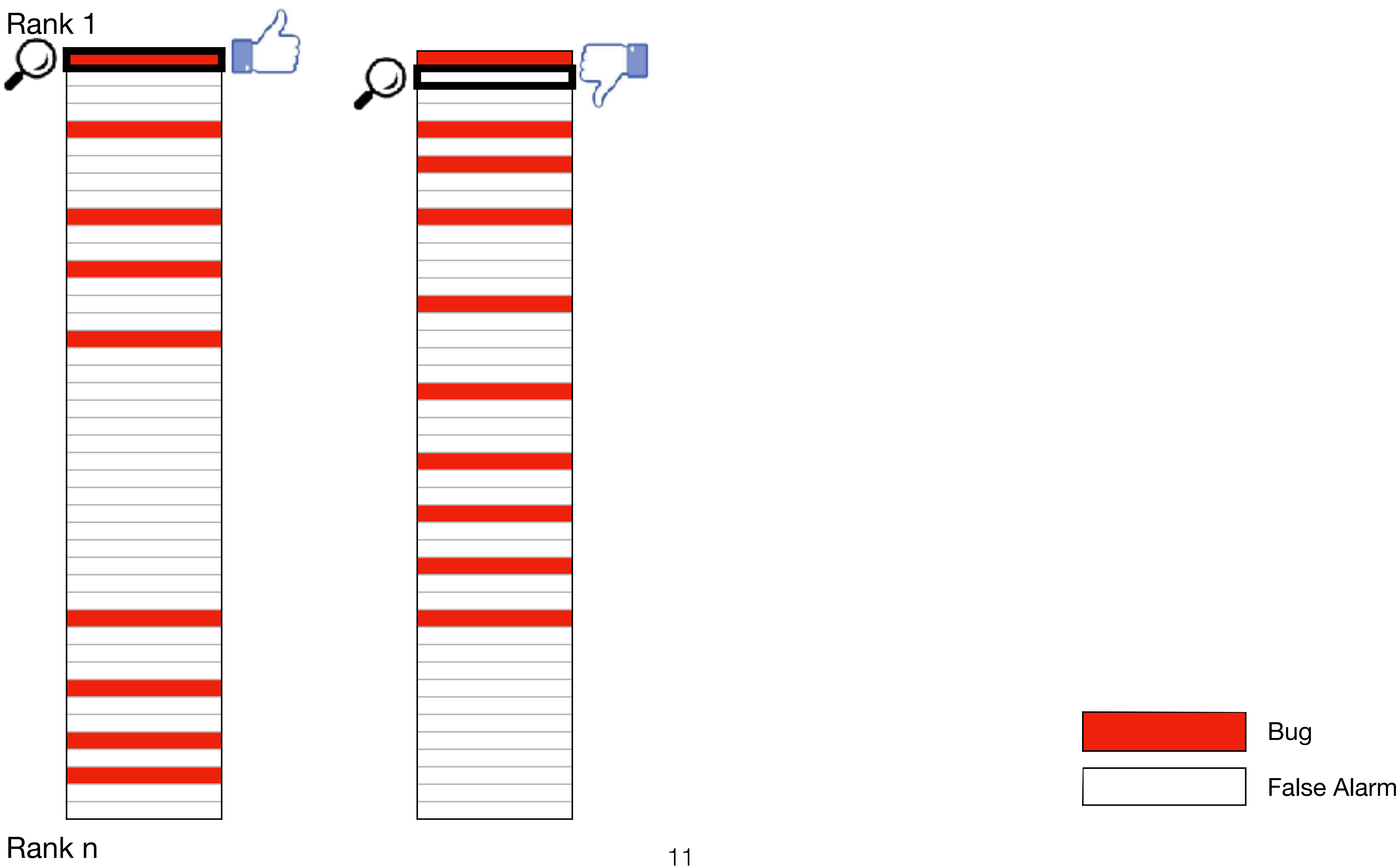


Interactive Alarm Ranker

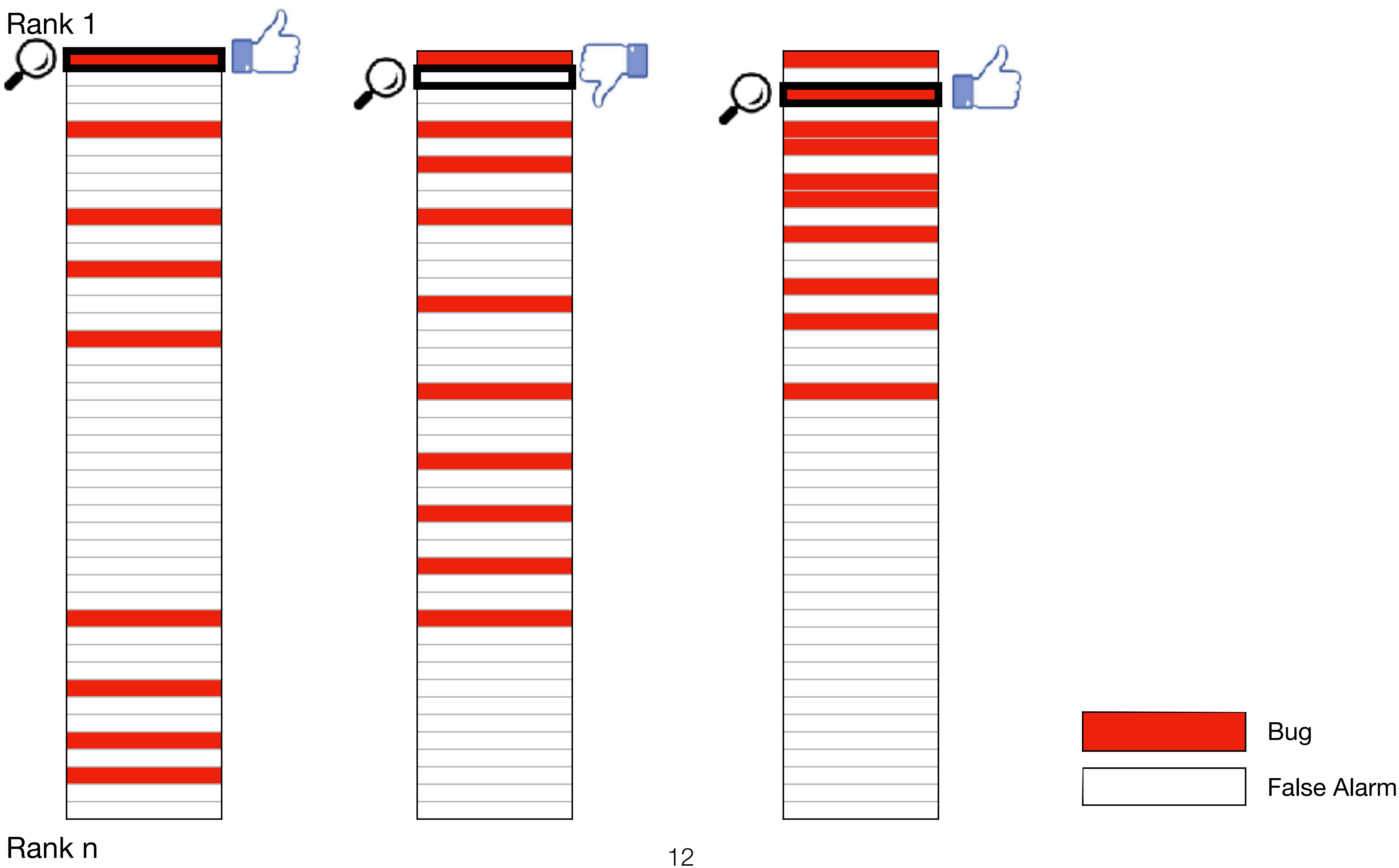




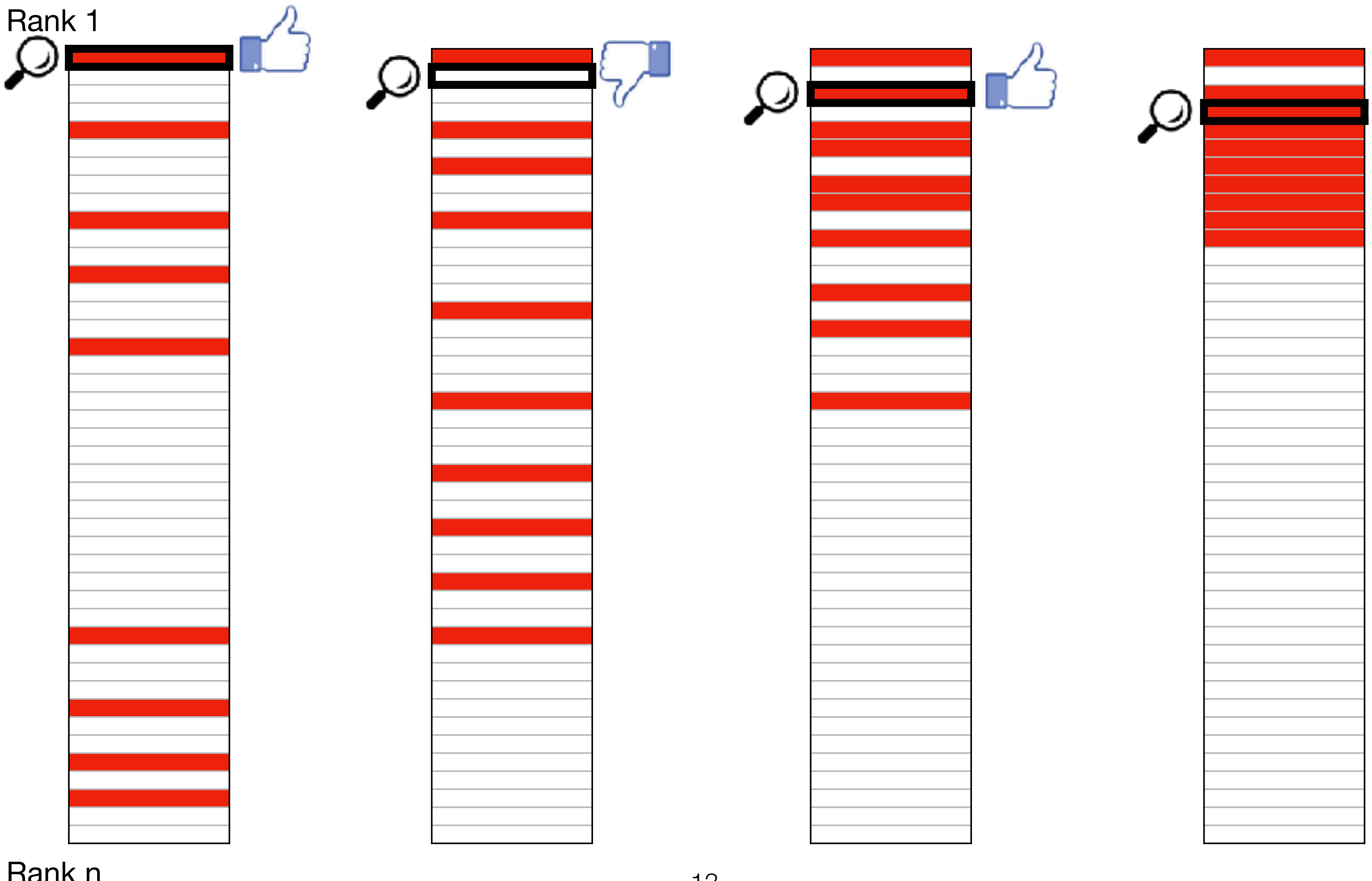
Interactive Alarm Ranker



Interactive Alarm Ranker

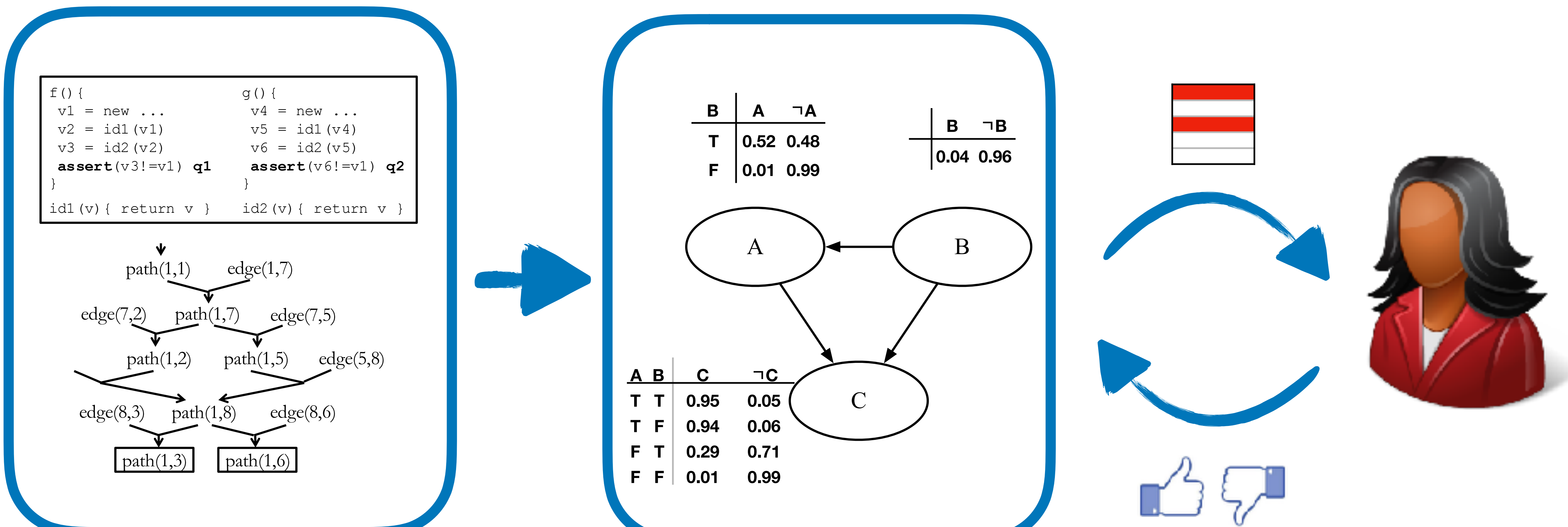


Interactive Alarm Ranker



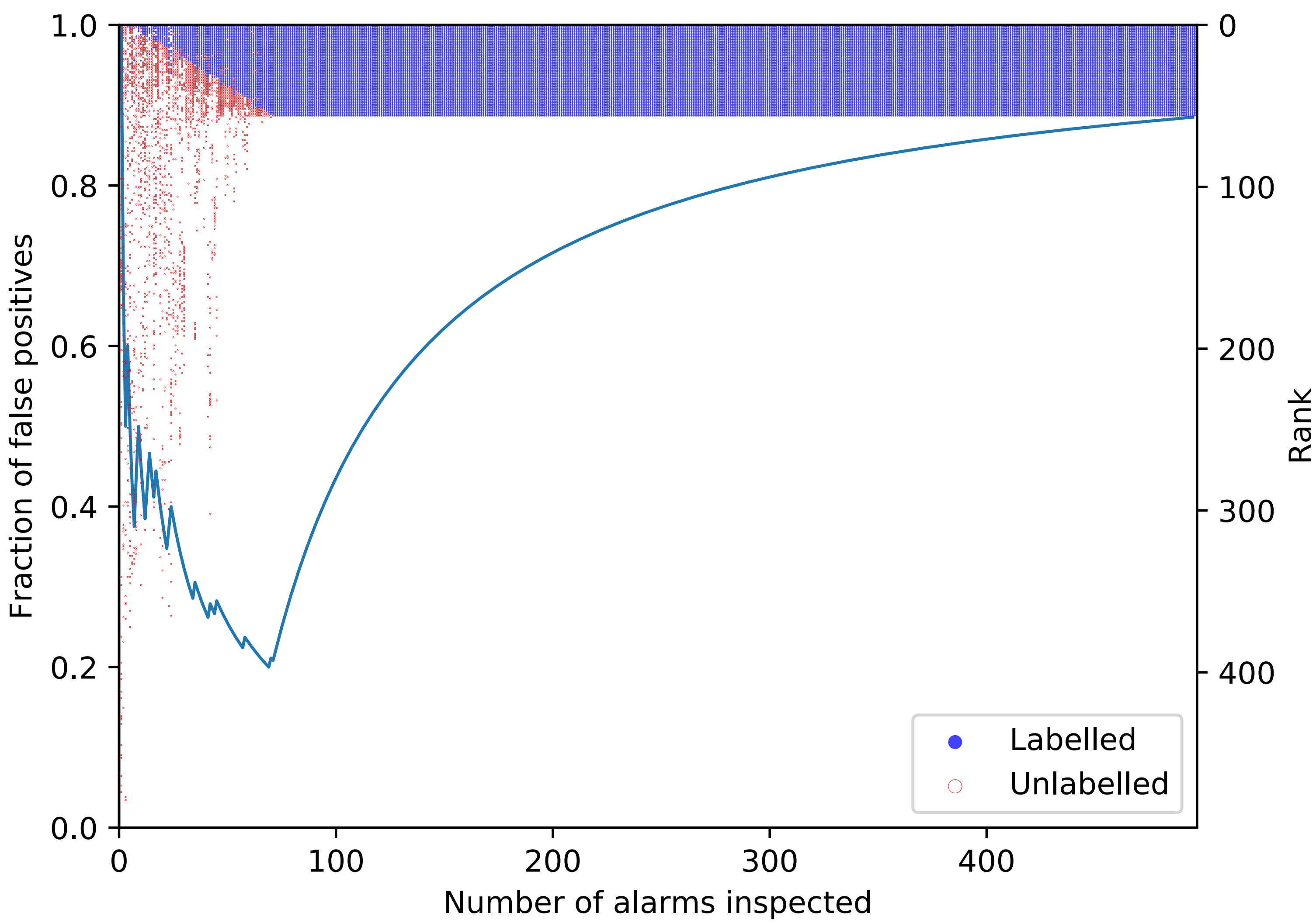
Key Idea

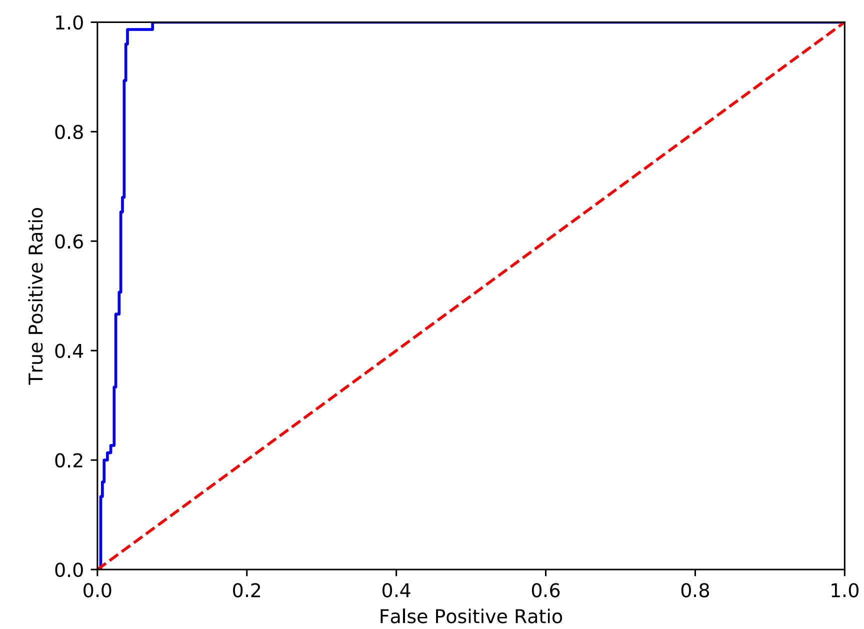
Human in the loop + Bayesian inference



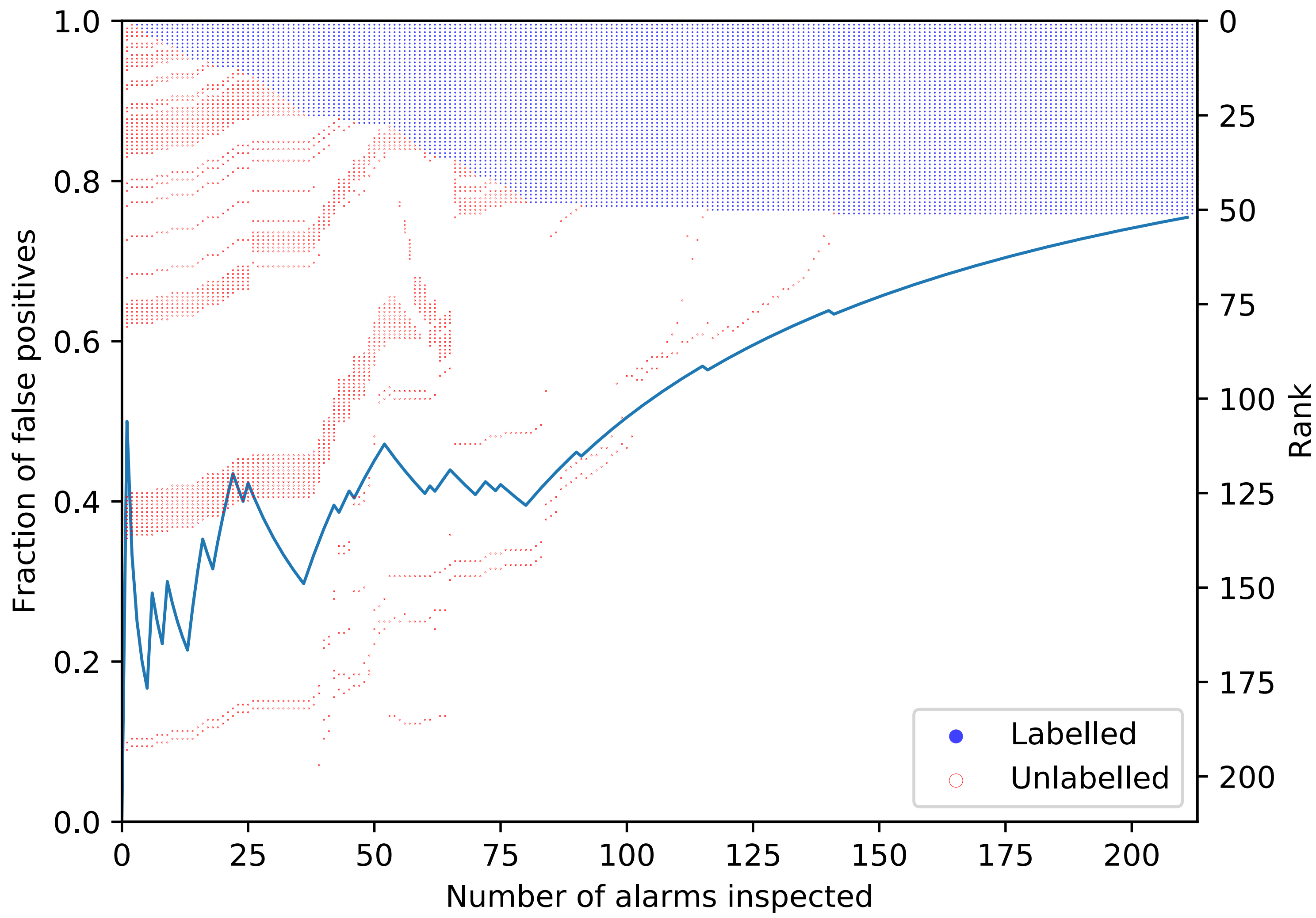
Static Analysis Result **Bayesian Network** **User**

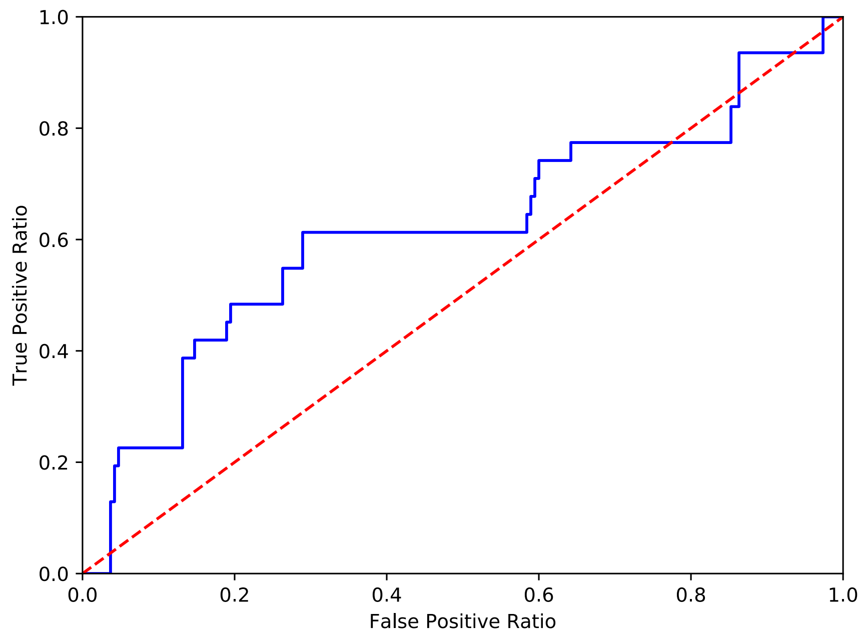
Case Study: Datarace





Case Study: Information Flow





Ex: Datarace Analysis

```java
public class RequestHandler {
  private FtpRequest request;

  public FtpRequest getRequest() {
    return request;                //L0
  }

  public void close() {
    synchronized (this) {          //L1
      if (isClosed) return;        //L2
      isClosed = true;             //L3
    }
    controlSocket.close();         //L4
    controlSocket = null;          //L5
    request.clear();               //L6
    request = null;                //L7
  }
}
```

```
Parallel(p1, p3) :- Parallel(p1, p2), Next(p2, p3),
                    Unguarded(p1, p3).
Parallel(p1, p2) :- Parallel(p2, p1).
    Race(p1, p2) :- Parallel(p1, p2), Alias(p1, p2).
```

***Apache FTP Server**

Ex: Datarace Analysis

```
public class RequestHandler {
 private FtpRequest request;

 public FtpRequest getRequest() {
  return request;                 //L0
 }

 public void close() {
  synchronized (this) {           //L1
   if (isClosed) return;          //L2
   isClosed = true;               //L3
  }
  controlSocket.close();          //L4
  controlSocket = null;           //L5
  request.clear();                //L6
  request = null;                 //L7
 }
}
```

```
Parallel(p1, p3) :- Parallel(p1, p2), Next(p2, p3),
                    Unguarded(p1, p3).
Parallel(p1, p2) :- Parallel(p2, p1).
  Race(p1, p2) :- Parallel(p1, p2), Alias(p1, p2).
```

Datarace

***Apache FTP Server**

Ex: Datarace Analysis

```
public class RequestHandler {
  private FtpRequest request;

  public FtpRequest getRequest() {
    return request;              //L0
  }

  public void close() {
    synchronized (this) {        //L1
      if (isClosed) return;      //L2
      isClosed = true;           //L3
    }
    controlSocket.close();       //L4
    controlSocket = null;        //L5
    request.clear();             //L6
    request = null;              //L7
  }
}
```

```
Parallel(p1, p3) :- Parallel(p1, p2), Next(p2, p3),
                    Unguarded(p1, p3).
Parallel(p1, p2) :- Parallel(p2, p1).
  Race(p1, p2) :- Parallel(p1, p2), Alias(p1, p2).
```

False alarm

False alarm

***Apache FTP Server**

Derivation Graph

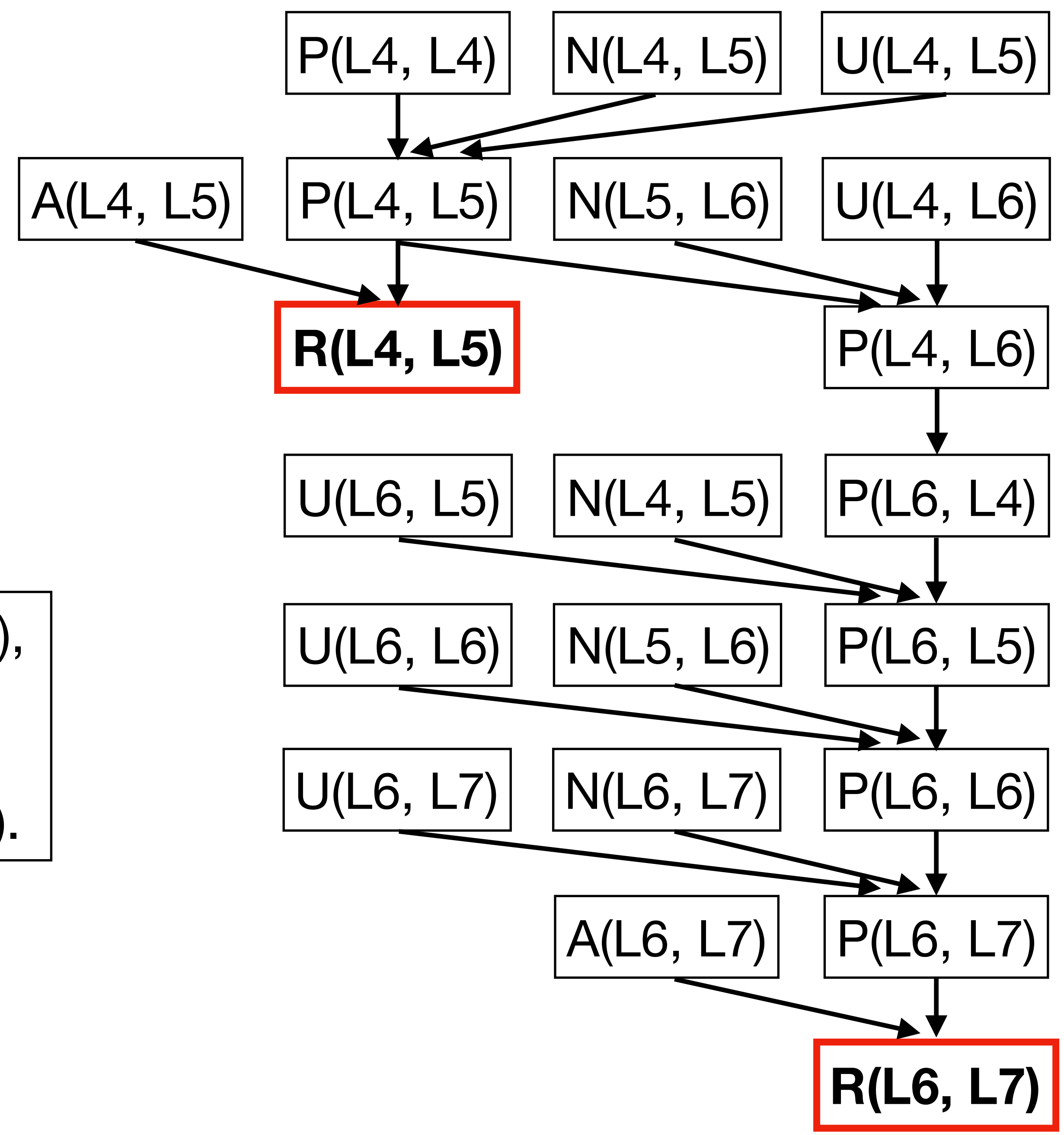
Program

```
controlSocket.close();//L4
controlSocket = null;  //L5
request.clear();       //L6
request = null;        //L7
```

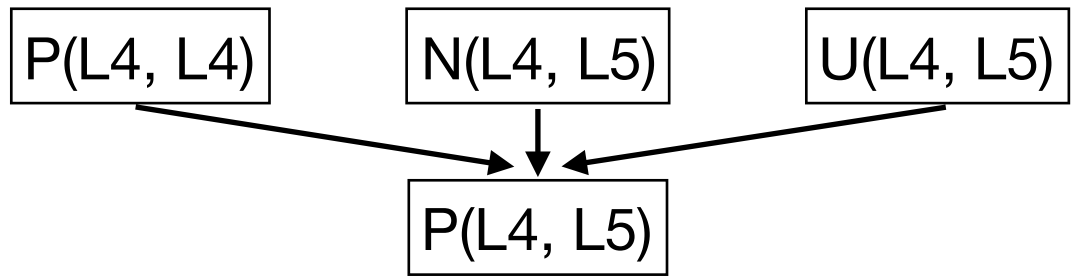
Datalog Rule

```
Parallel(p1, p3) :- Parallel(p1, p2), Next(p2, p3),
                    Unguarded(p1, p3).
Parallel(p1, p2) :- Parallel(p2, p1).
   Race(p1, p2) :- Parallel(p1, p2), Alias(p1, p2).
```

Derivation Graph



Bayesian Network



Logical Rule

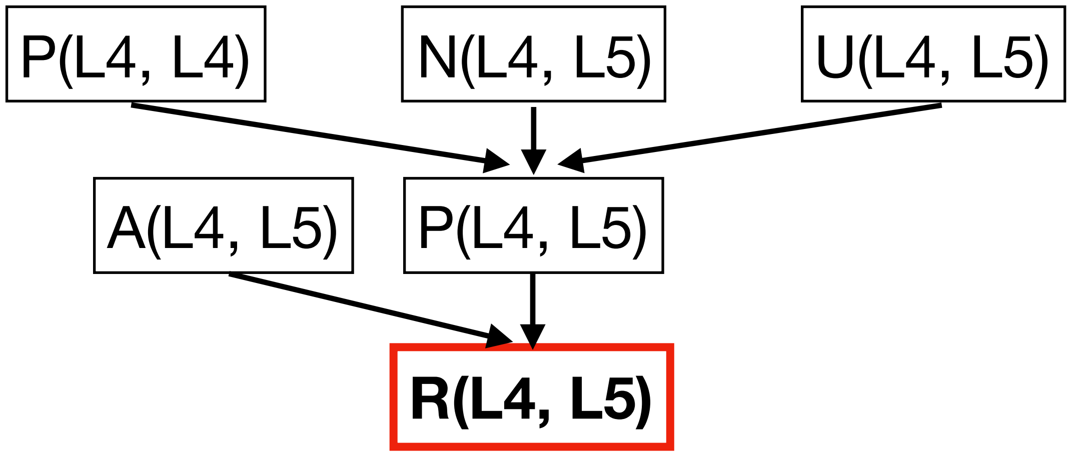
Parallel(p1, p3) :- Parallel(p1, p2), Next(p2, p3),
Unguarded(p1, p3).
Parallel(p1, p2) :- Parallel(p2, p1).
Race(p1, p2) :- Parallel(p1, p2), Alias(p1, p2).

Probabilistic Rule

P(L4,L4)	N(L4,L5)	U(L4,L5)	Pr(P(L4,L5) | H)
TRUE	TRUE	TRUE	0.95
TRUE	TRUE	FALSE	0
	…		
FALSE	FALSE	FALSE	0

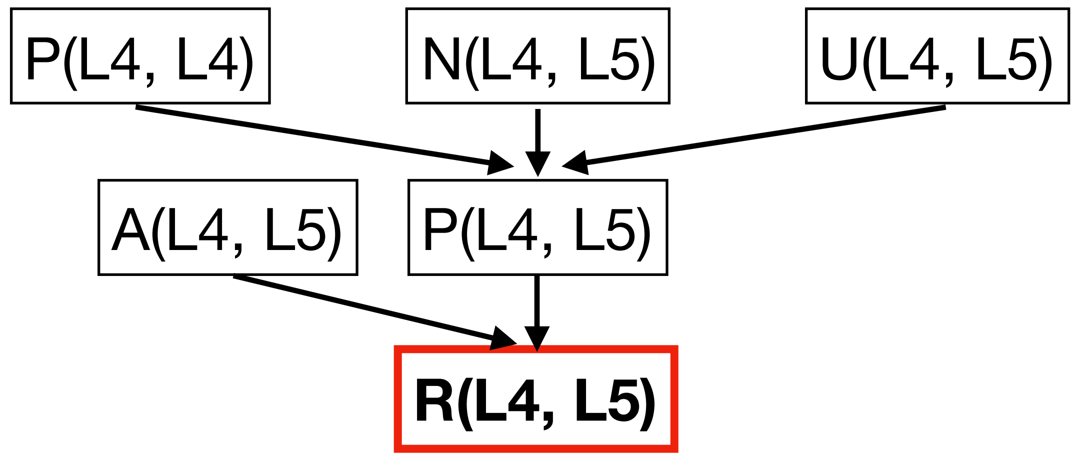
*Prior probability is computed by an offline learning

Marginal Inference



$$
\begin{aligned}
Pr(R(L4,L5)) = \; & Pr(R(L4,L5), A(L4,L5), P(L4,L5)) \\
& + Pr(R(L4,L5), \neg A(L4,L5), P(L4,L5)) \\
& + Pr(R(L4,L5), A(L4,L5), \neg P(L4,L5)) \\
& + Pr(R(L4,L5), \neg A(L4,L5), \neg P(L4,L5))
\end{aligned}
$$

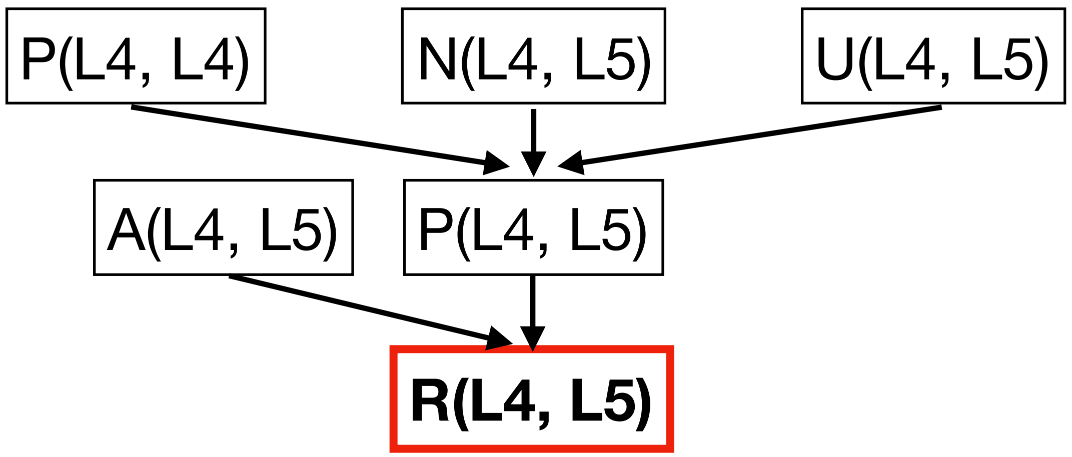
Marginal Inference



$$
\begin{aligned}
\Pr(R(L4,L5)) = {} & \Pr(R(L4,L5), A(L4,L5), P(L4,L5)) \\
& + \Pr(R(L4,L5), \neg A(L4,L5), P(L4,L5)) \\
& + \Pr(R(L4,L5), A(L4,L5), \neg P(L4,L5)) \\
& + \Pr(R(L4,L5), \neg A(L4,L5), \neg P(L4,L5))
\end{aligned}
$$

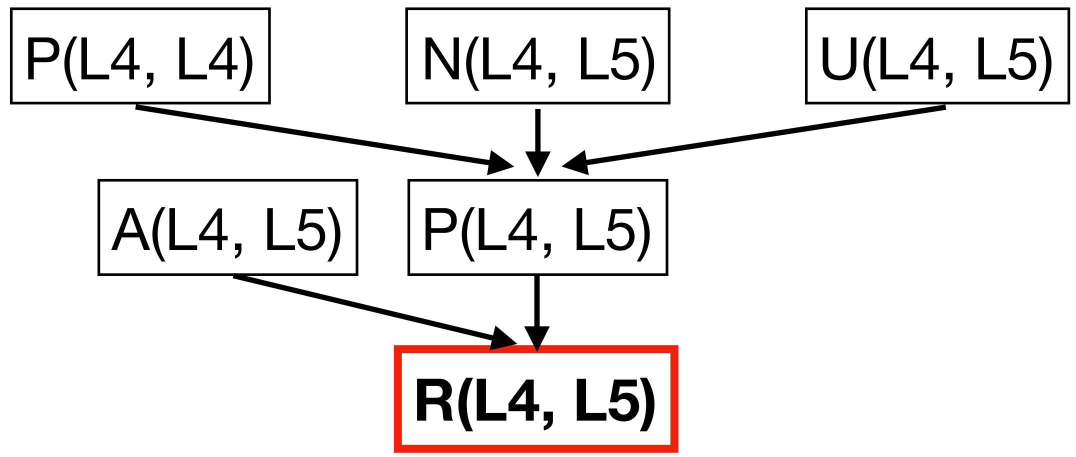
If any of the antecedents fail, then the race cannot happen.

Marginal Inference



Pr(R(L4,L5)) = Pr(R(L4,L5), A(L4,L5), P(L4,L5))

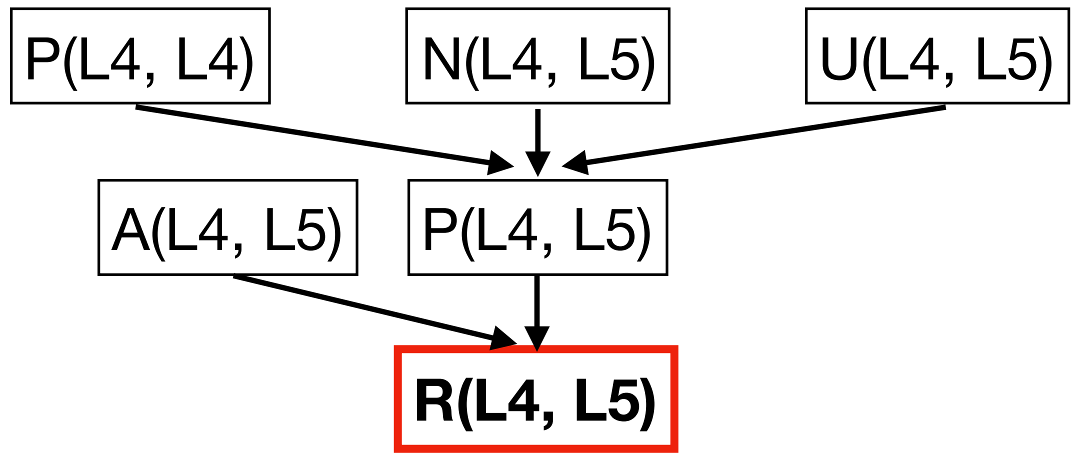
Marginal Inference



Pr(R(L4,L5)) = Pr(R(L4,L5), A(L4,L5), P(L4,L5))
= Pr(R(L4,L5) | A(L4,L5), P(L4,L5)) *
Pr(A(L4,L5)) * Pr(P(L4,L5))

By Bayes's Rule:
Pr(A,B) = Pr(A|B) * Pr(B)

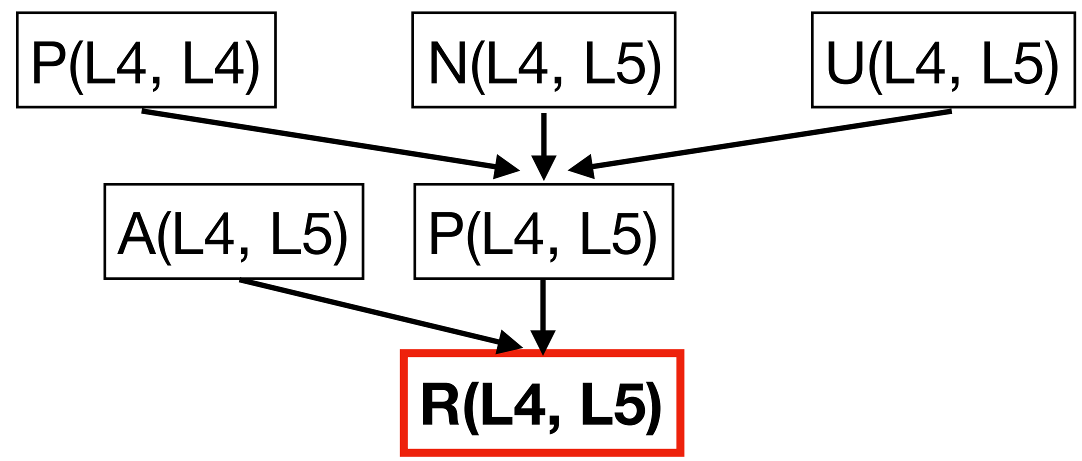
Marginal Inference



$$
\begin{aligned}
Pr(R(L4,L5)) &= Pr(R(L4,L5), A(L4,L5), P(L4,L5)) \\
&= Pr(R(L4,L5) \mid A(L4,L5), P(L4,L5)) * \\
&\quad\ Pr(A(L4,L5)) * Pr(P(L4,L5)) \\
&\boxed{\begin{aligned} &= 0.95 * 1.0 * Pr(P(L4,L5)) \\ &= 0.95 * Pr(P(L4,L5), Pr(P(L4,L4)), Pr(N(L4,L5), Pr(U(L4,L5)) \end{aligned}}
\end{aligned}
$$

Assume that the probabilities of firing each rule and input tuple are 0.95 and 1.0.

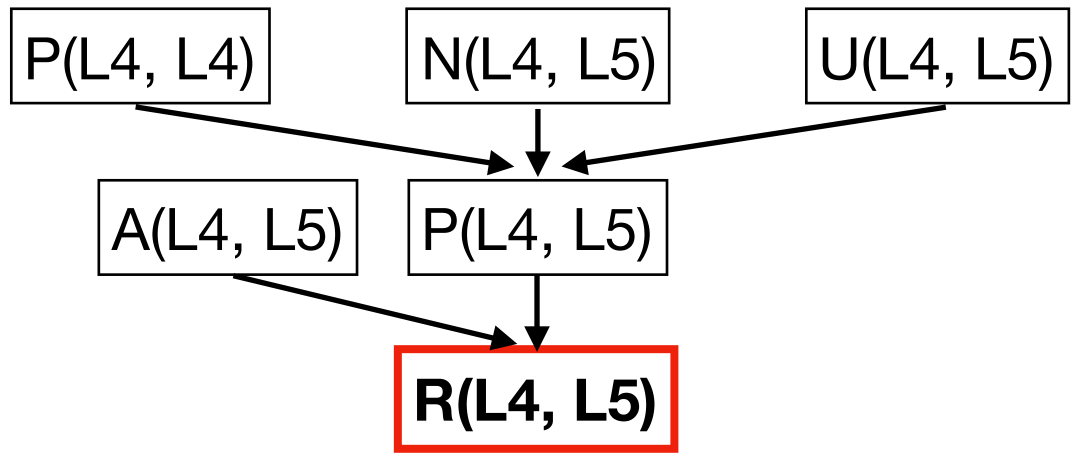
Marginal Inference



Pr(R(L4,L5)) = Pr(R(L4,L5), A(L4,L5), P(L4,L5))
= Pr(R(L4,L5) | A(L4,L5), P(L4,L5)) *
Pr(A(L4,L5)) * Pr(P(L4,L5))
= 0.95 * 1.0 * Pr(P(L4,L5))
= 0.95 * Pr(P(L4,L5), Pr(P(L4,L4)), Pr(N(L4,L5), Pr(U(L4,L5))
= 0.95 * Pr(P(L4,L5) | Pr(P(L4,L4)), Pr(N(L4,L5), Pr(U(L4,L5)) *
Pr(P(L4,L4)) * Pr(N(L4,L5)) * Pr(U(L4,L5))

By Bayes's Rule:
Pr(A,B) = Pr(A|B) * Pr(B)

Marginal Inference



Pr(R(L4,L5)) = Pr(R(L4,L5), A(L4,L5), P(L4,L5))
= Pr(R(L4,L5) | A(L4,L5), P(L4,L5)) *
Pr(A(L4,L5)) * Pr(P(L4,L5))
= 0.95 * 1.0 * Pr(P(L4,L5))
= 0.95 * 0.95 * Pr(P(L4,L4)) * Pr(N(L4,L5) * Pr(U(L4,L5))
= …
= 0.398

Alarm Ranking

```
public class RequestHandler {
  private FtpRequest request;

  public FtpRequest getRequest() {
    return request;                    //L0
  }

  public void close() {
    synchronized (this) {              //L1
      if (isClosed) return;            //L2
      isClosed = true;                 //L3
    }
    controlSocket.close();             //L4
    controlSocket = null;              //L5
    request.clear();                   //L6
    request = null;                    //L7
  }
}
```

Ranking	Alarm	Confidence
1	R(L4, L5)	0.398
2	R(L5, L5)	0.378
3	R(L6, L7)	0.324
4	R(L7, L7)	0.308
5	R(L0, L7)	0.279

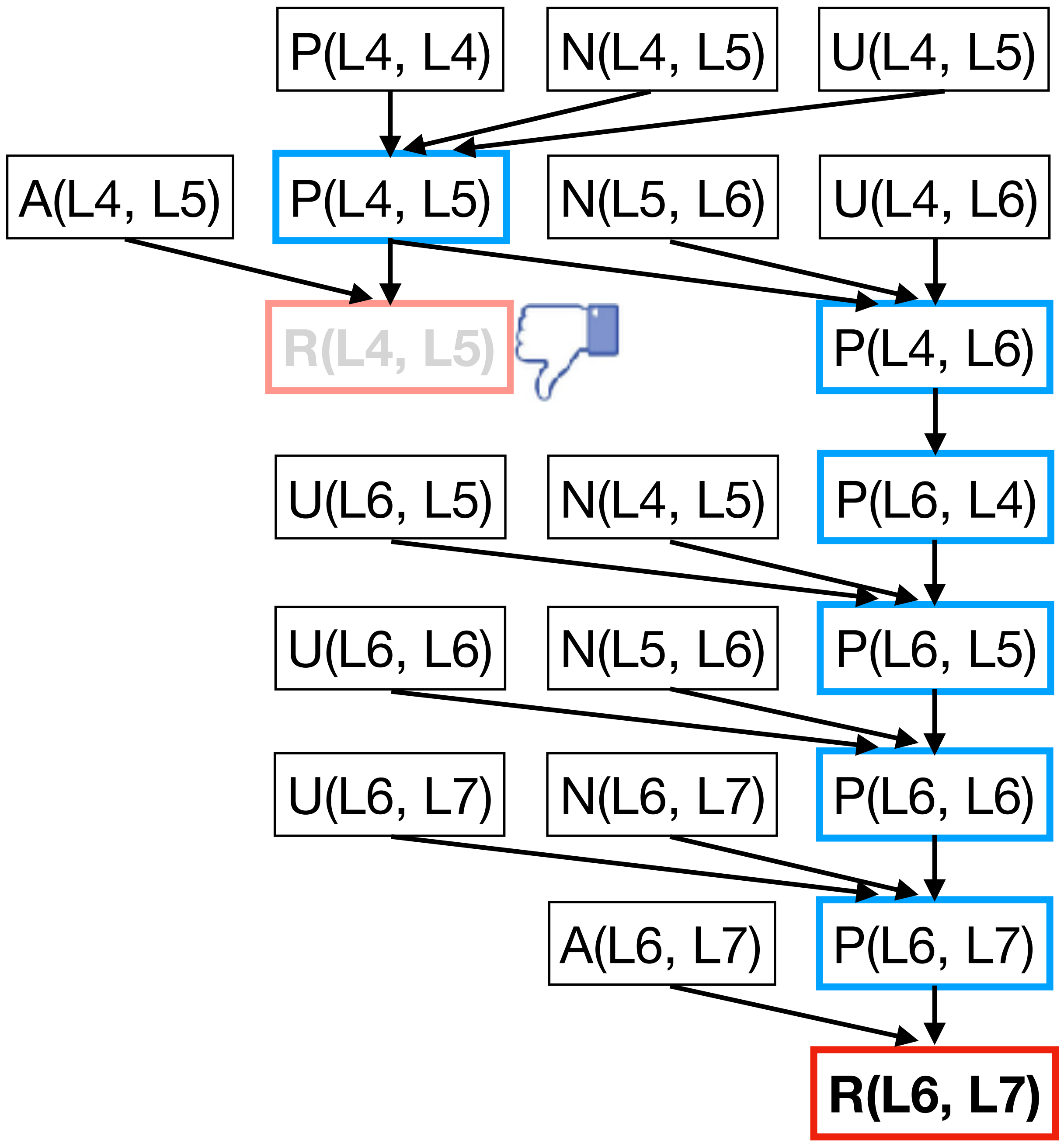
Alarm Ranking

```
public class RequestHandler {
  private FtpRequest request;

  public FtpRequest getRequest() {
    return request;                 //L0
  }

  public void close() {
    synchronized (this) {           //L1
      if (isClosed) return;         //L2
      isClosed = true;              //L3
    }
    controlSocket.close();          //L4
    controlSocket = null;           //L5
    request.clear();                //L6
    request = null;                 //L7
  }
}
```

Ranking	Alarm	Confidence
1	R(L4, L5)	0.398
2	R(L5, L5)	0.378
3	R(L6, L7)	0.324
4	R(L7, L7)	0.308
5	R(L0, L7)	0.279

Q: What are the probabilities of the other alarms when R(L4,L5) is false?

Marginal Inference



Pr(P(L4,L5) | ¬R(L4,L5))
 = Pr(¬R(L4,L5) | P(L4,L5)) *
 Pr(P(L4,L5)) / Pr(¬R(L4,L5))
 = 0.03

By Bayes's Rule:
Pr(A|B) = P(B|A) * Pr(A) / Pr(B)

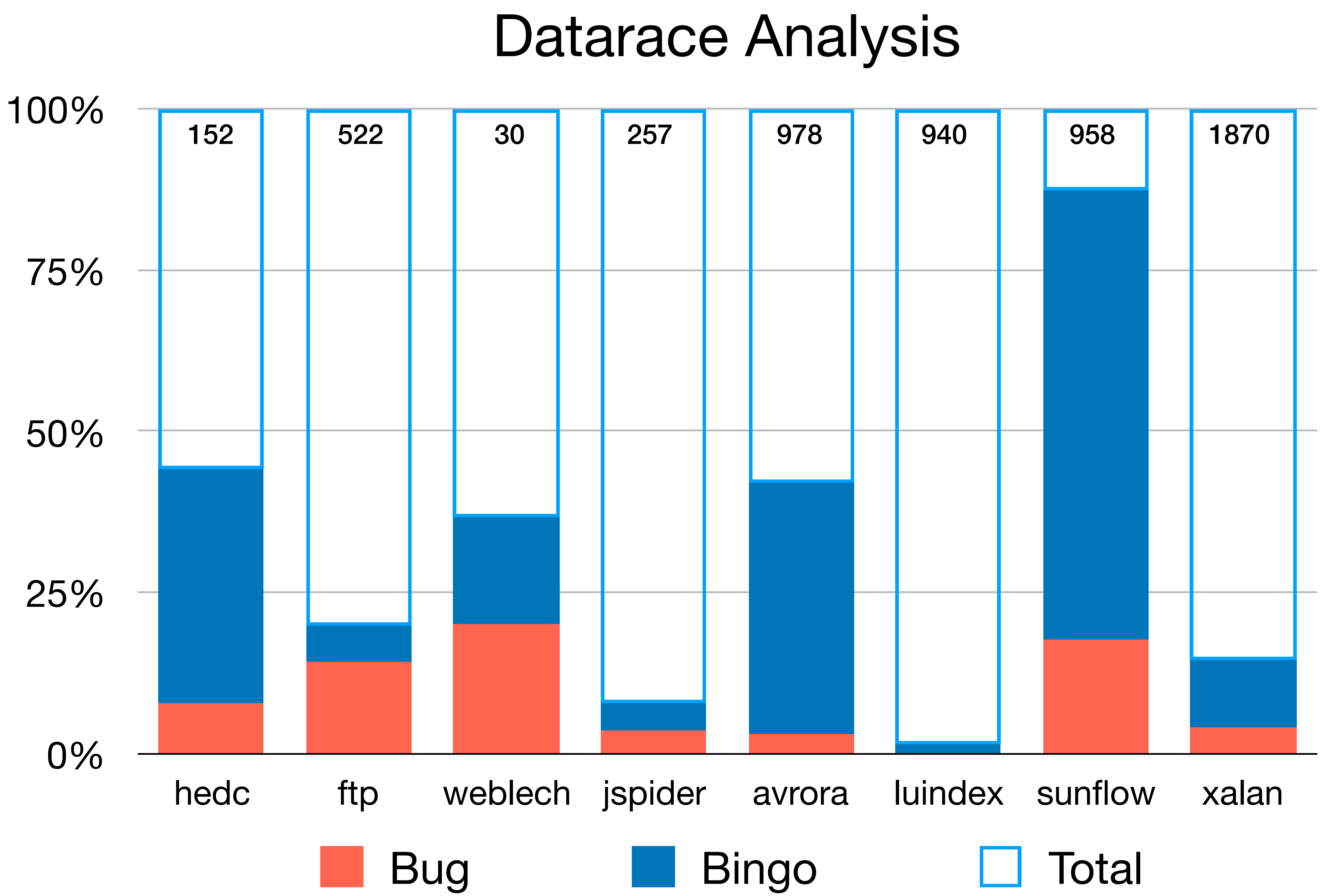
Pr(R(L6,L7) | ¬R(L4,L5))
 = Pr(R(L6,L7) | P(L4,L5)) *
 Pr(P(L4,L5)) | ¬R(L4,L5))
 = 0.03

Alarm Ranking

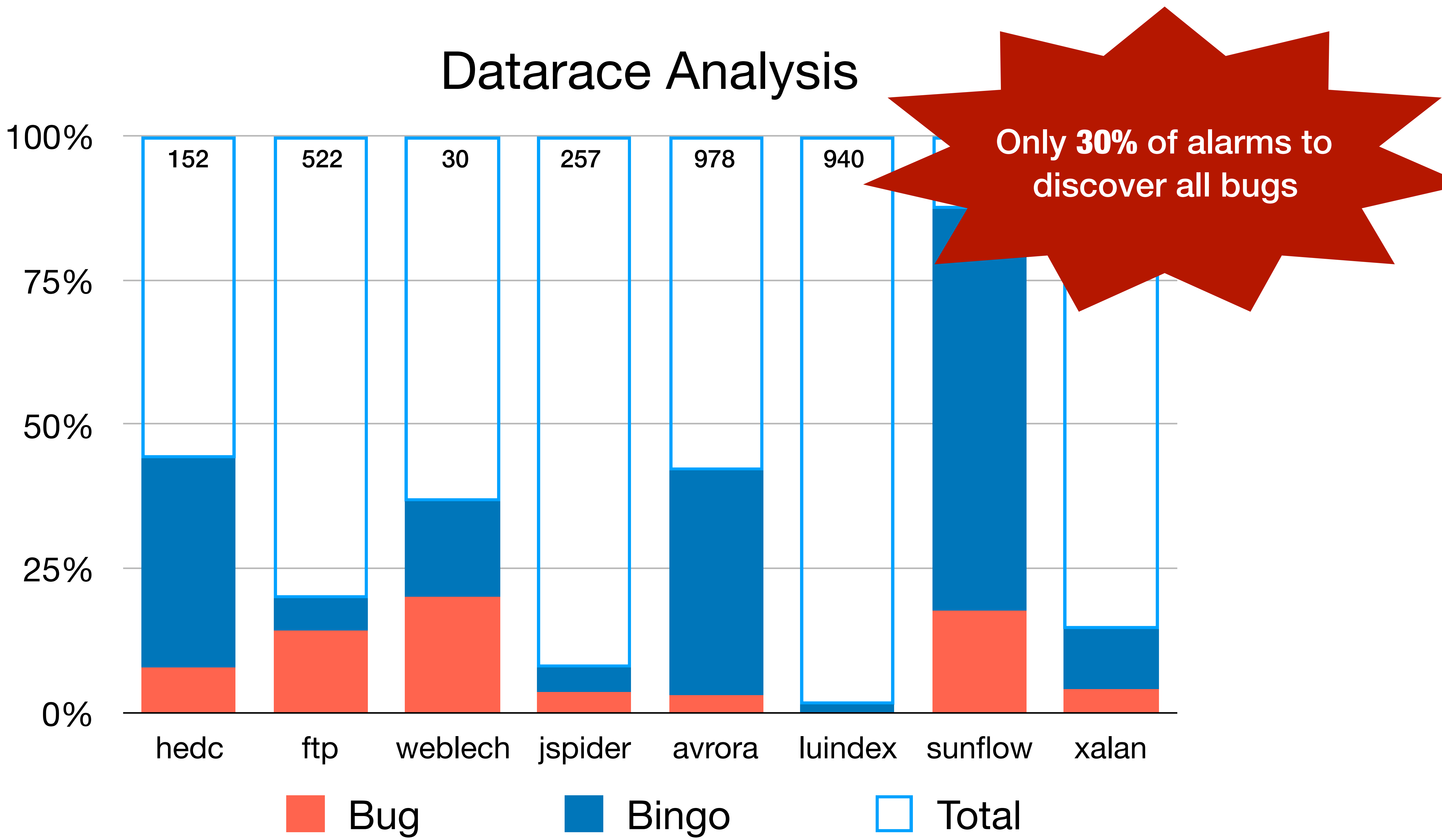
Ranking	Alarm	Confidence
1	R(L4, L5)	0.398
2	R(L5, L5)	0.378
3	R(L6, L7)	0.324
4	R(L7, L7)	0.308
5	R(L0, L7)	0.279

Ranking	Alarm	Confidence
1	R(L0, L7)	0.279
2	R(L5, L5)	0.035
3	R(L6, L7)	0.030
4	R(L7, L7)	0.028
5	R(L4, L5)	0

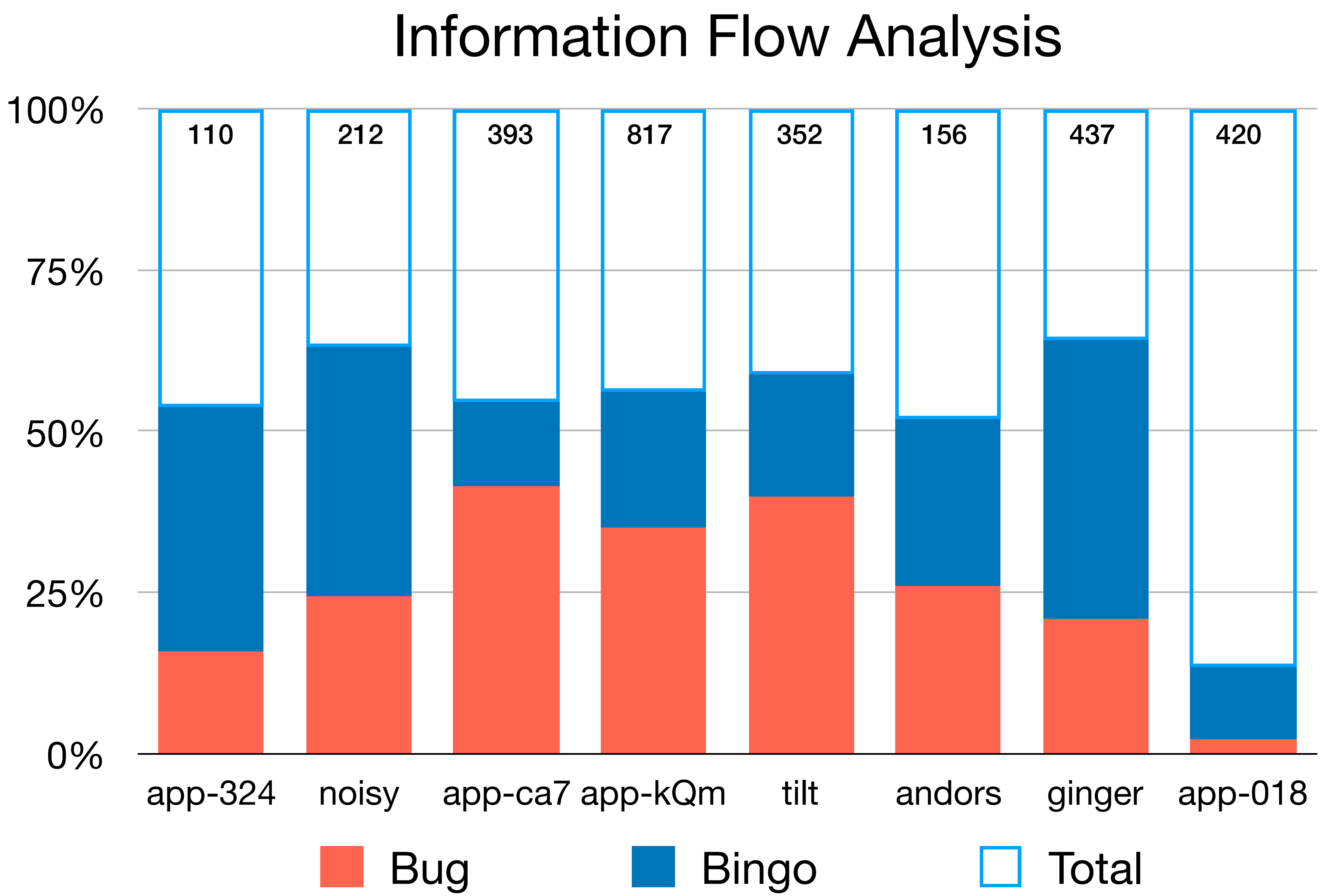
Experimental Results



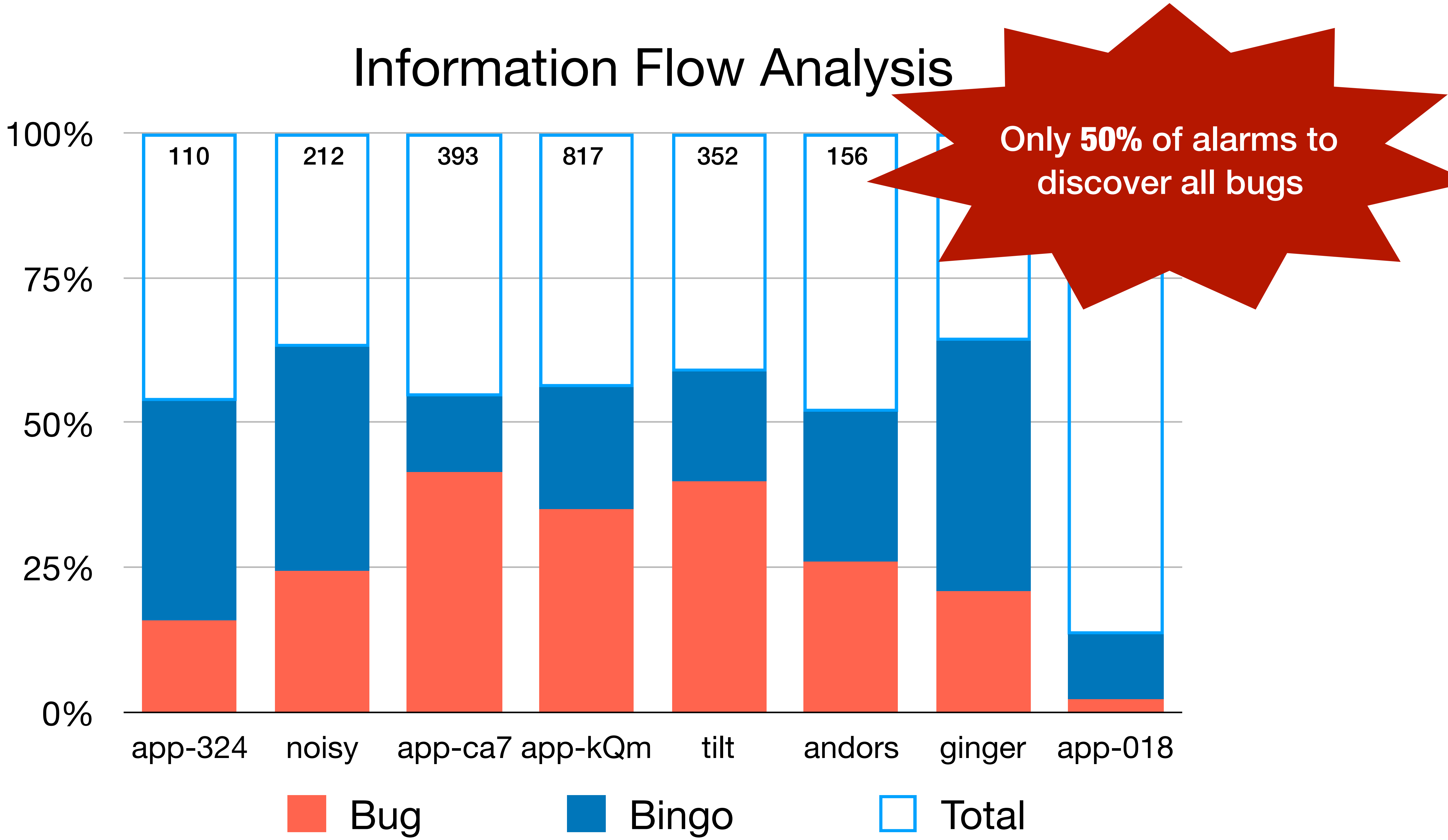
Experimental Results



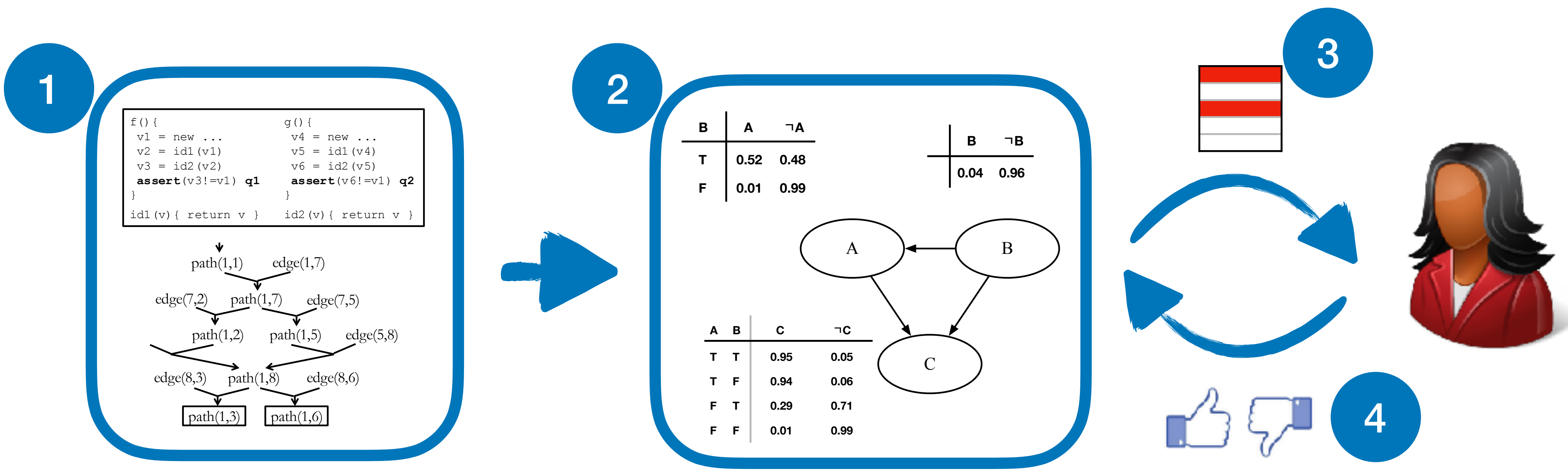
Experimental Results



Experimental Results



Future Work



1. Generalizing to non-datalog static analyses
2. Transferring the learned knowledge to other programs
3. Optimizing the marginal inference solver
4. Designing more fine-grained interaction models

Conclusion

- First interactive alarm ranking system
- Logical + probabilistic reasoning using Bayesian network
- Hope to build AI-guided static analysis system

Conclusion

- First interactive alarm ranking system
- Logical + probabilistic reasoning using Bayesian network
- Hope to build AI-guided static analysis system

Thank You