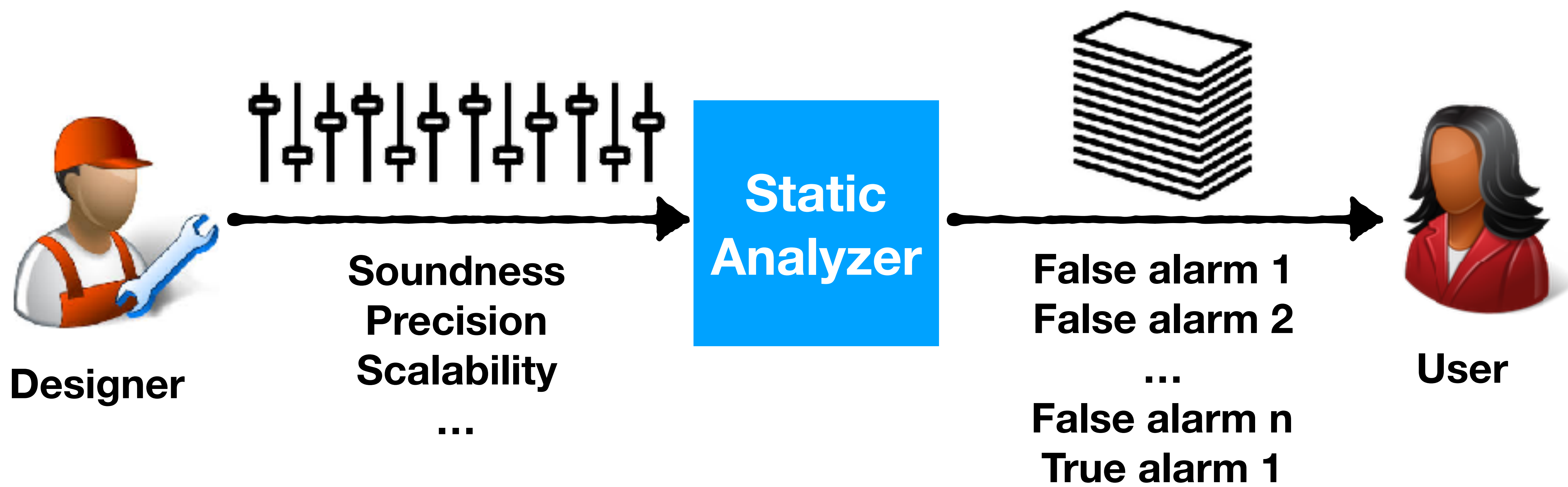
Interactive Alarm Ranking System using Bayesian Inference

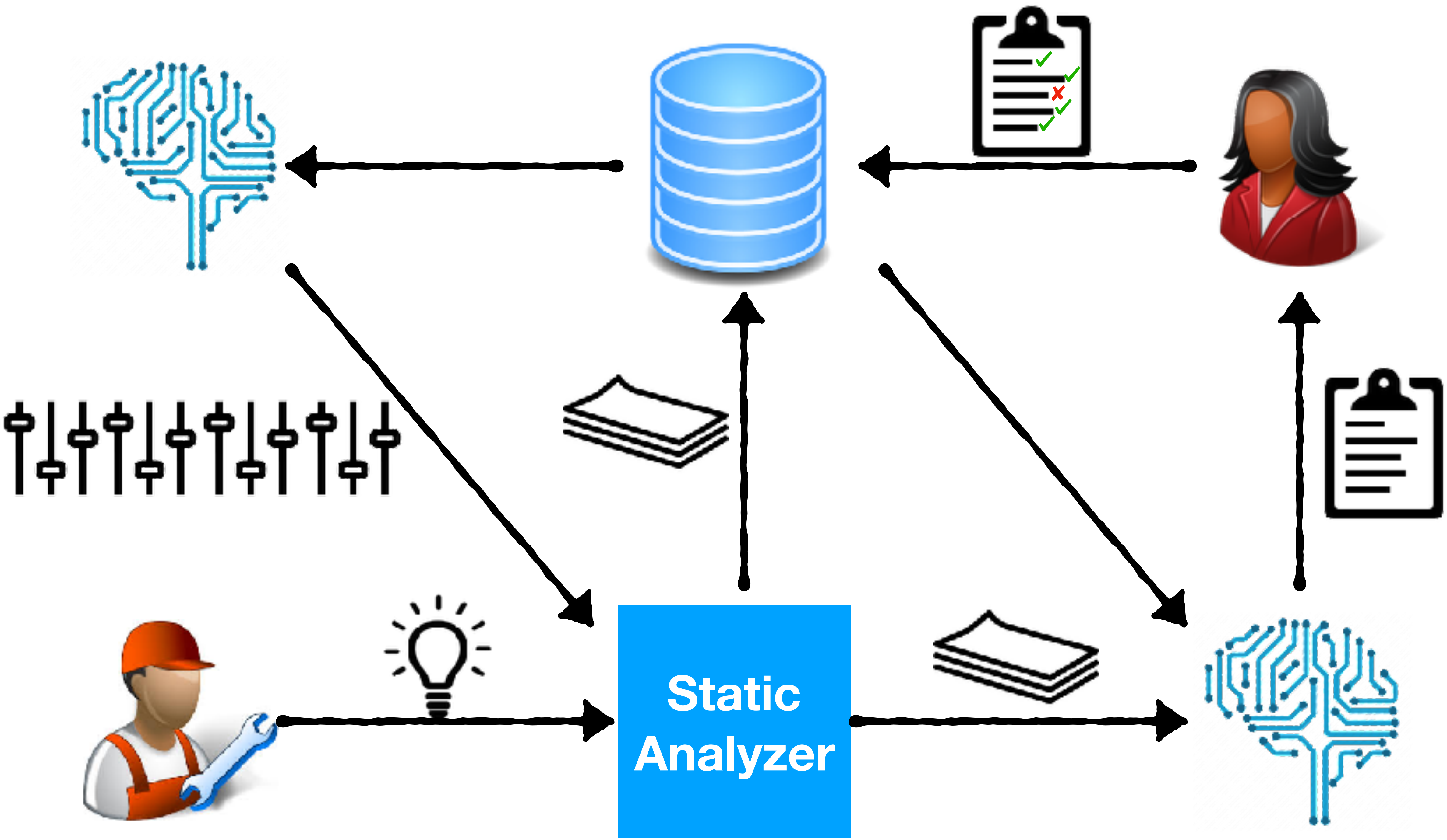
Kihong Heo
University of Pennsylvania
(cowork with Sulekha Kulkarni, Mayur Naik, Mukund Raghothaman)

Jan 4 2017 @ Korea University

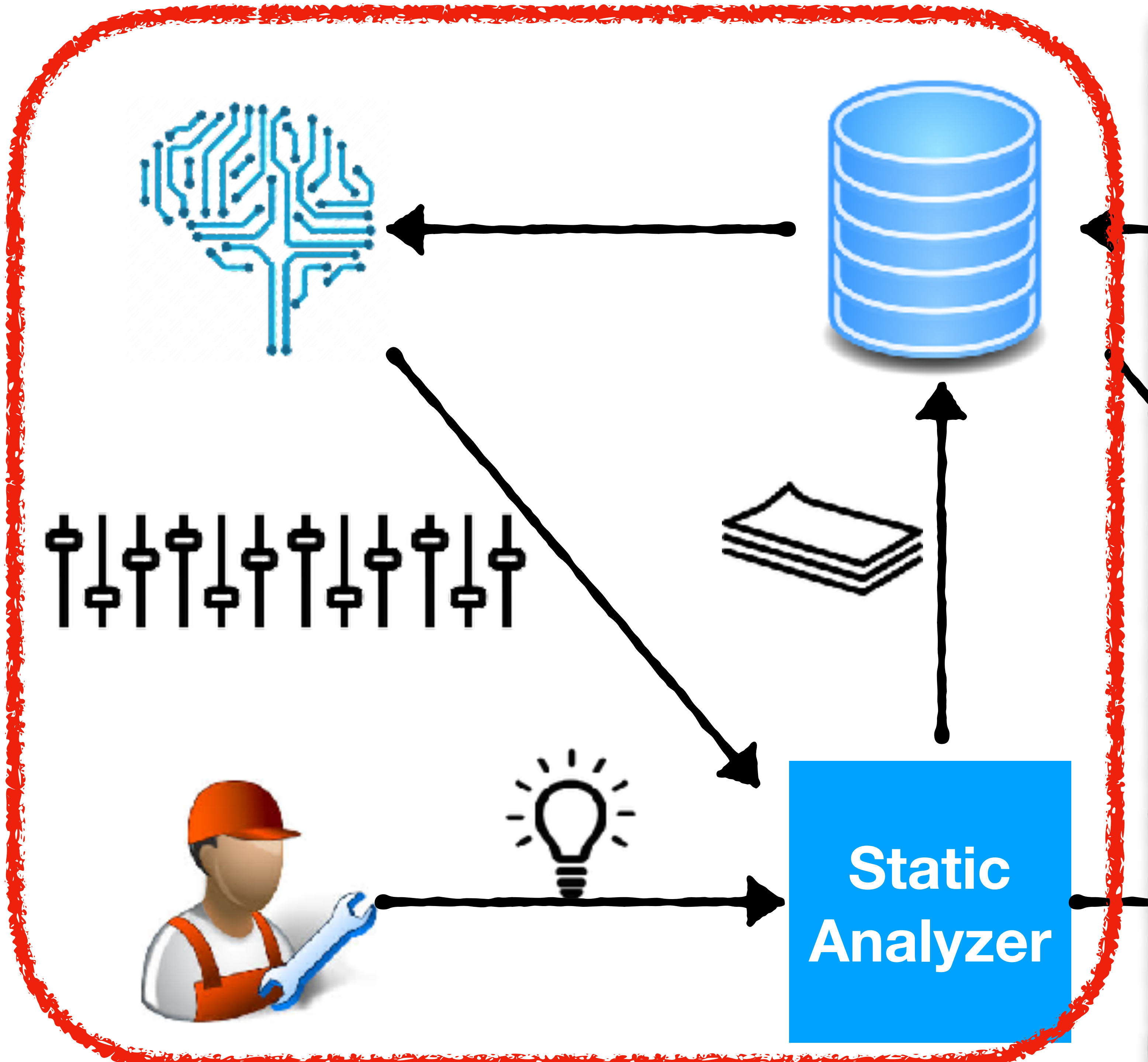
Conventional Static Analysis



Next-generation Static Analysis



Next-generation Static Analysis



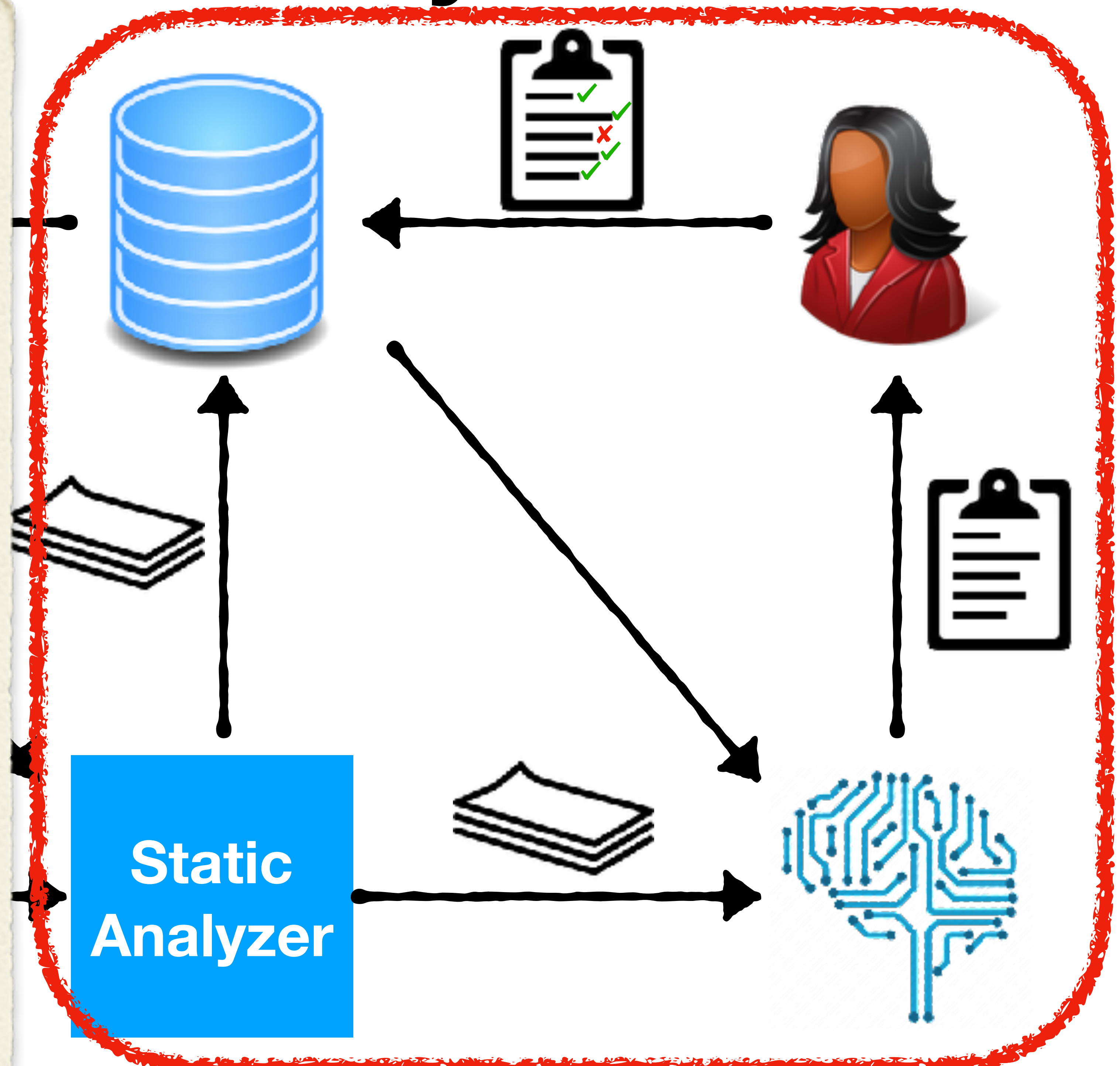
AI-based Analysis Design

- Human only provides high-level idea
- AI provides detailed design choices
- DB accumulates analysis results

Next-generation Static Analysis

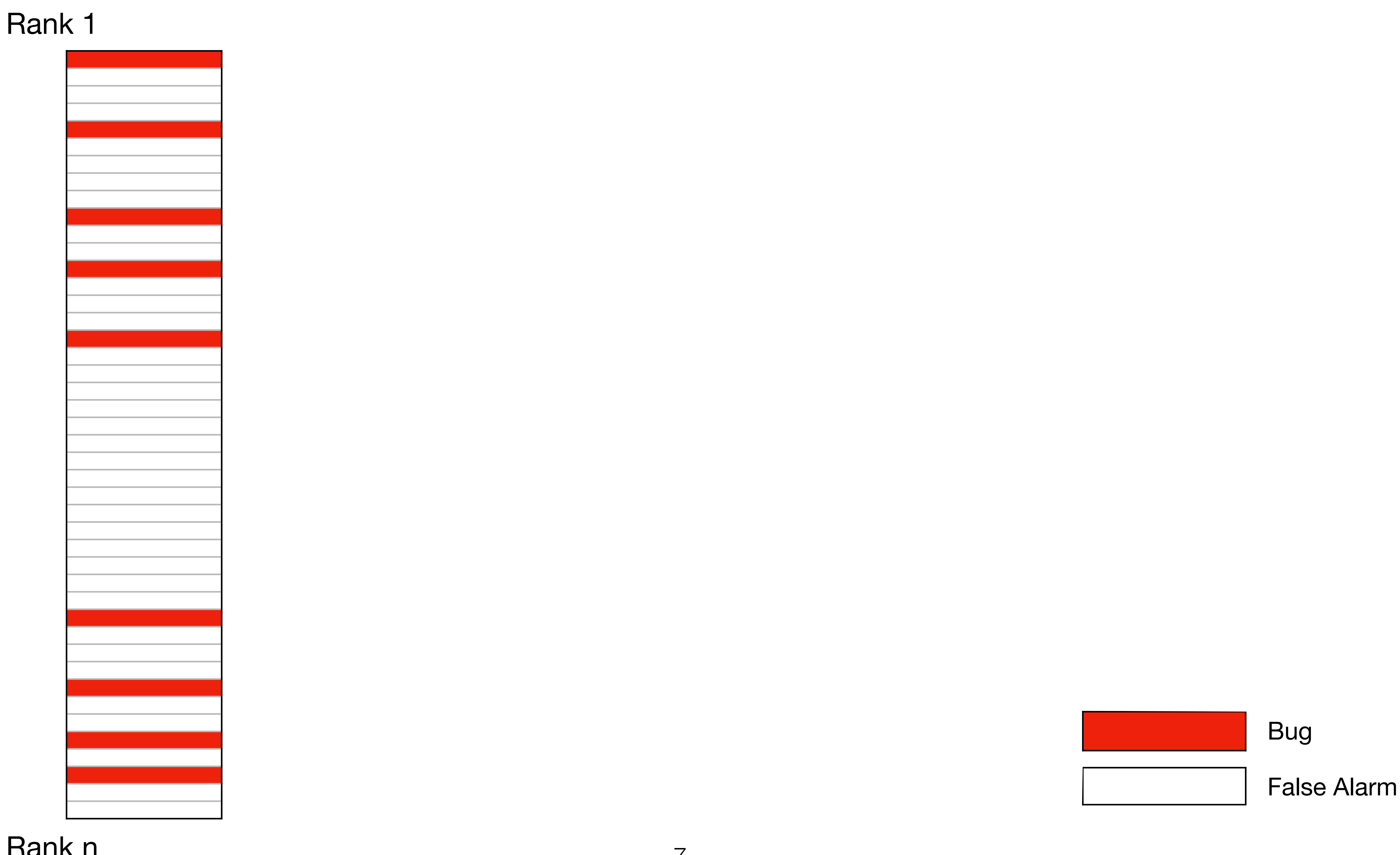
AI-based Alarm Report

- AI prioritizes/classifies analysis alarms
- Human only inspects alarms with high confidence
- DB accumulates analysis results and labeled alarms

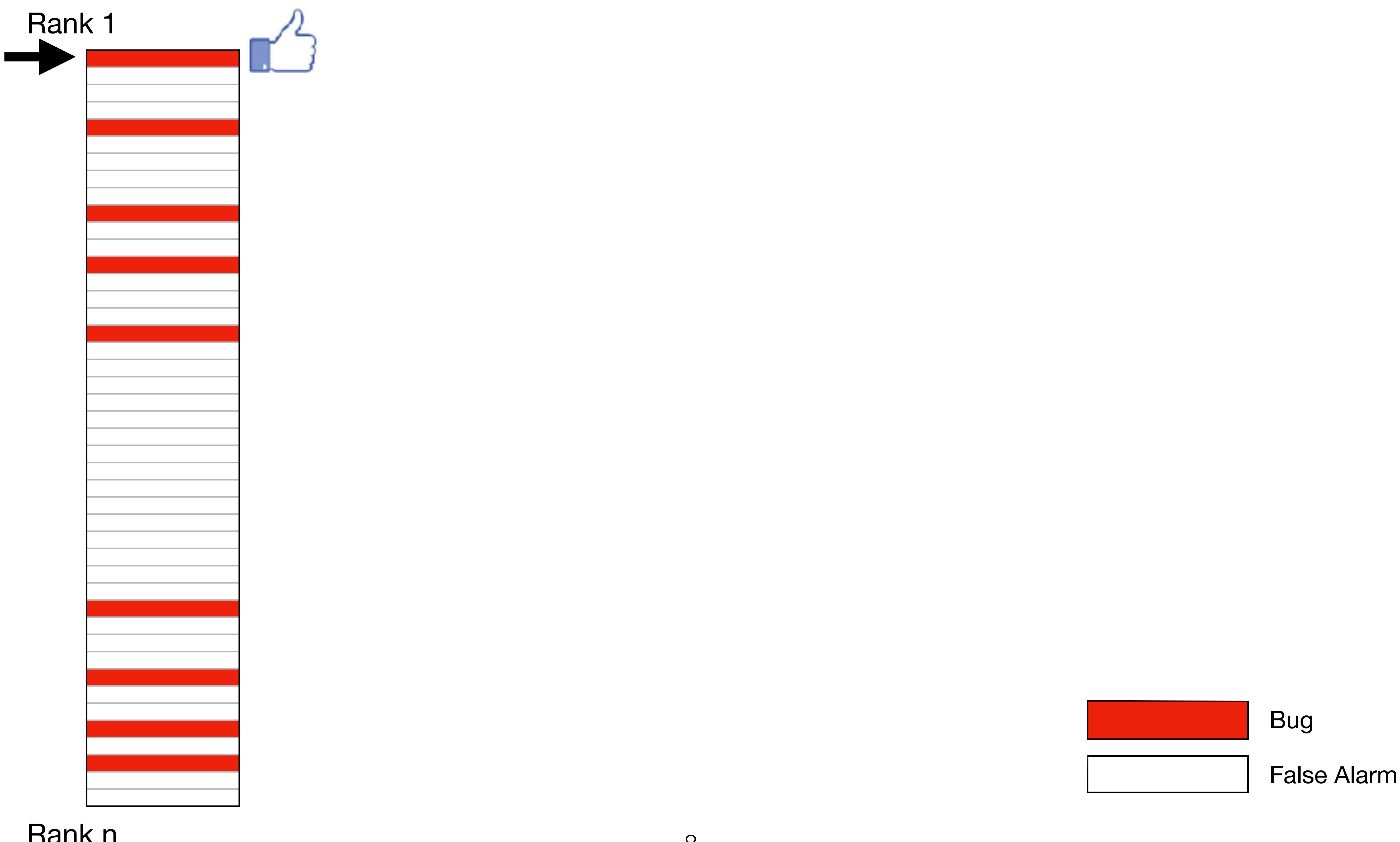


BINGO: An Interactive Alarm Ranking System

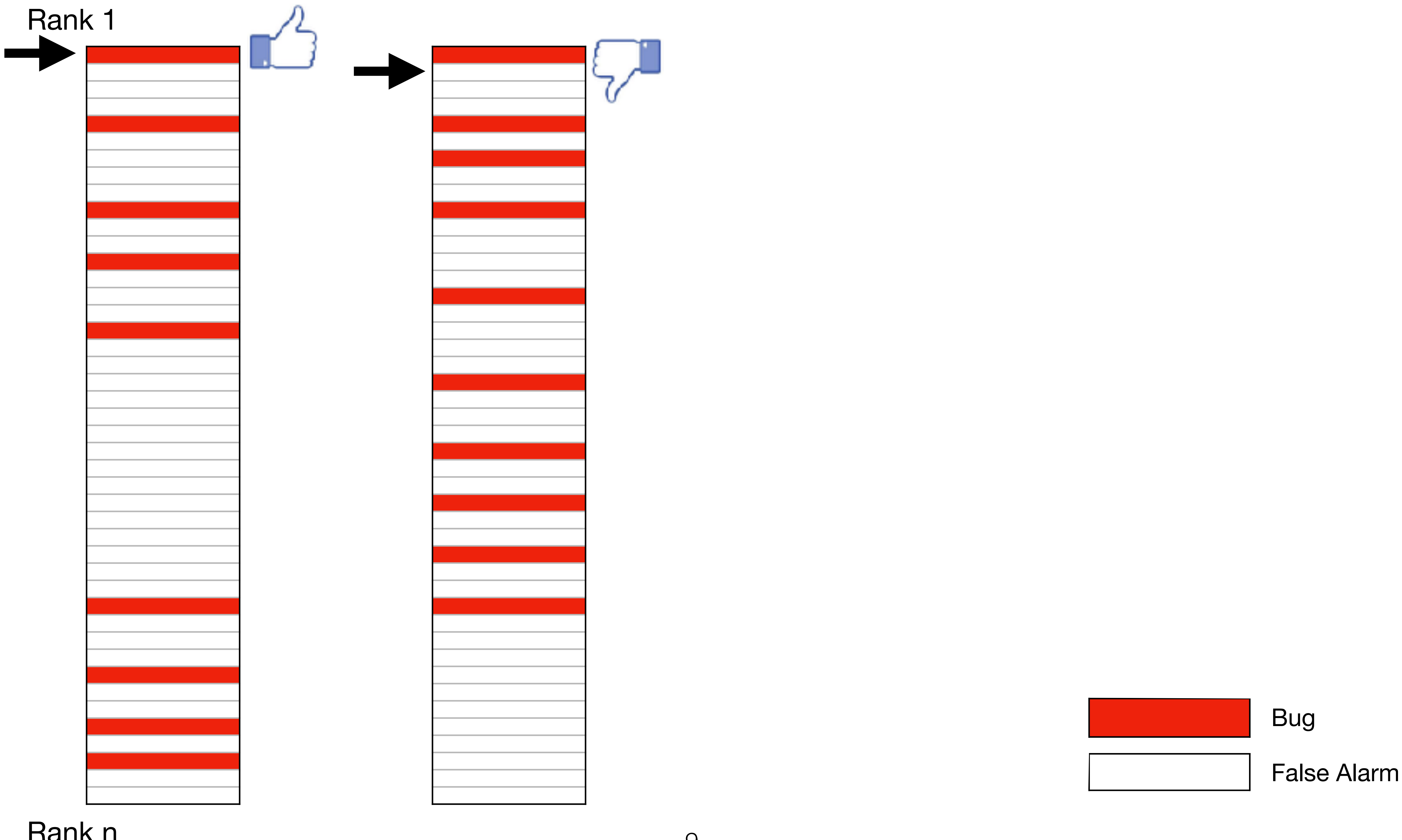
Interactive Alarm Ranker



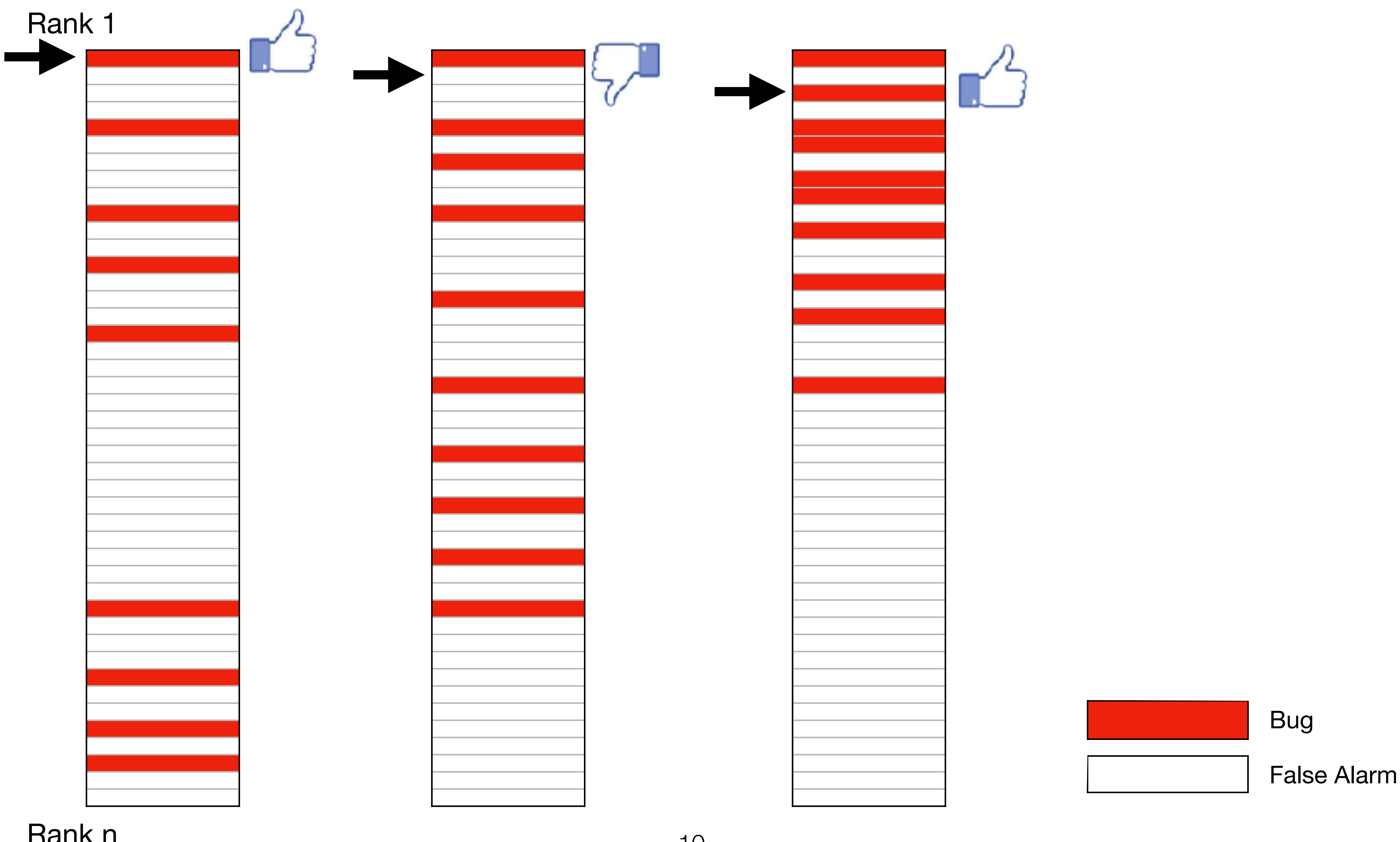
Interactive Alarm Ranker



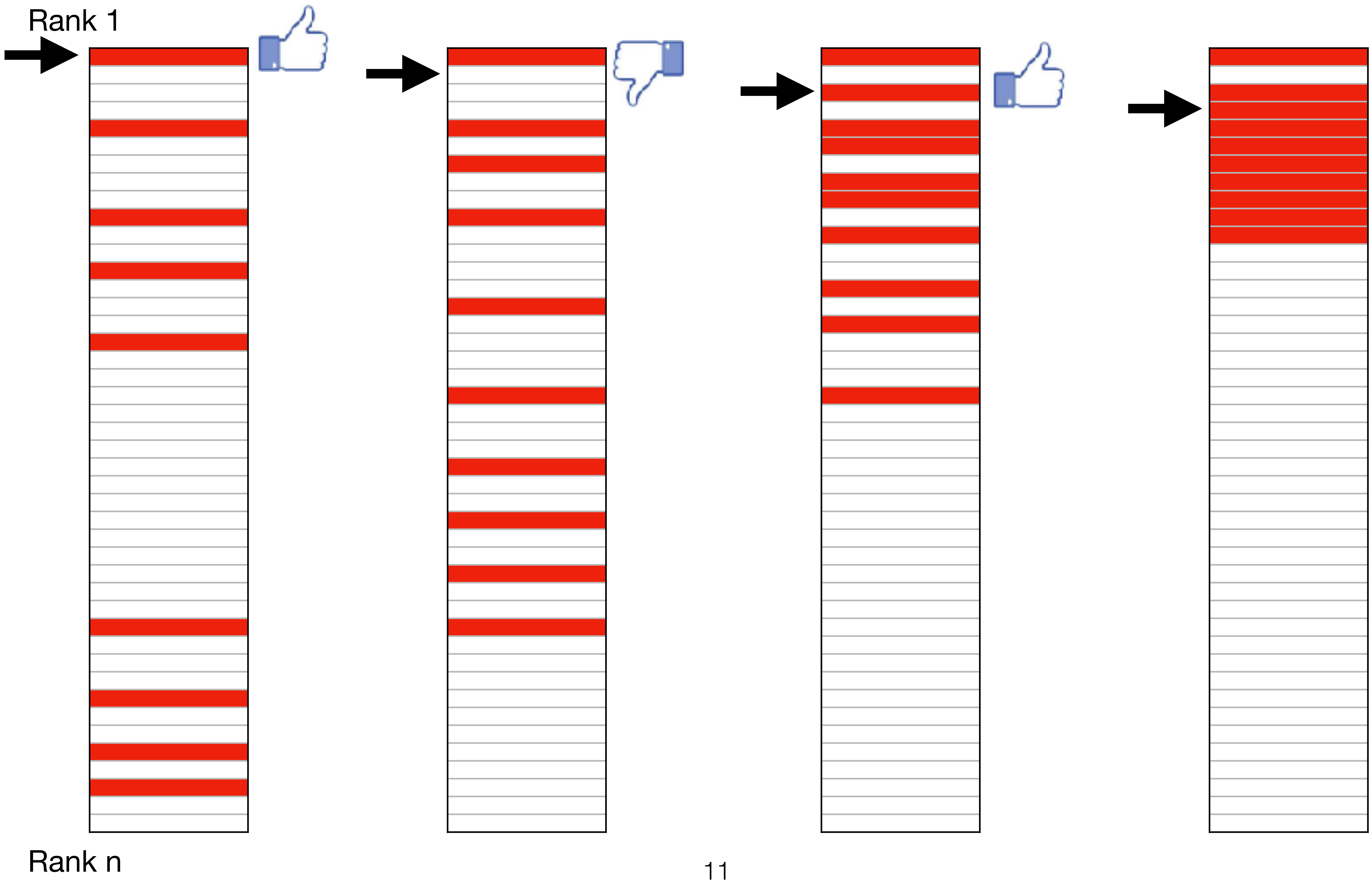
Interactive Alarm Ranker



Interactive Alarm Ranker

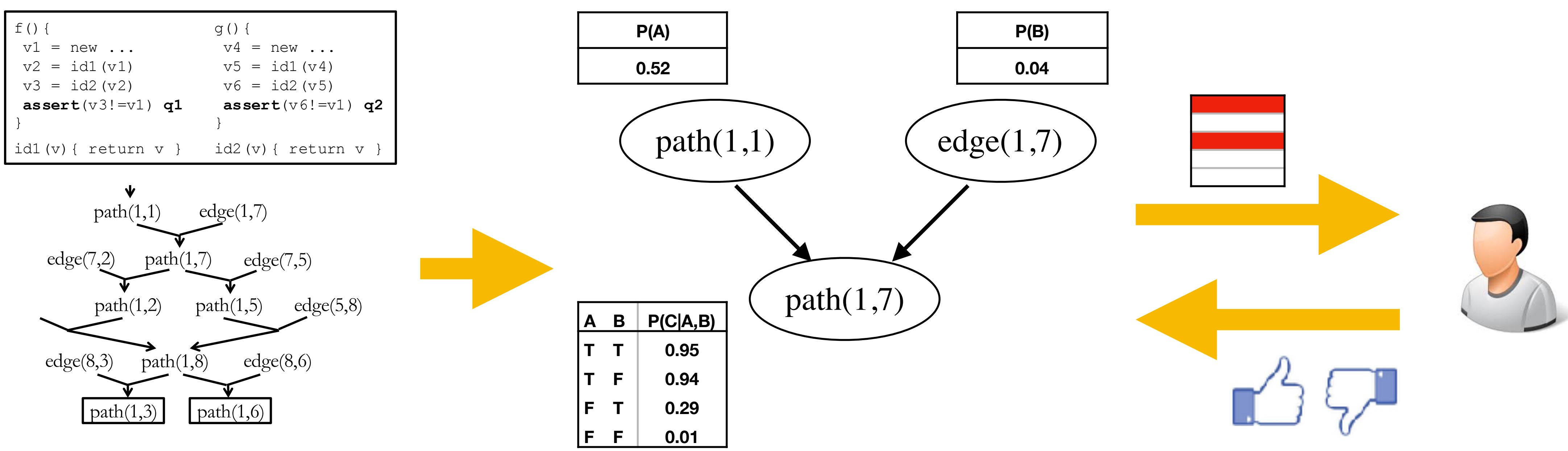


Interactive Alarm Ranker



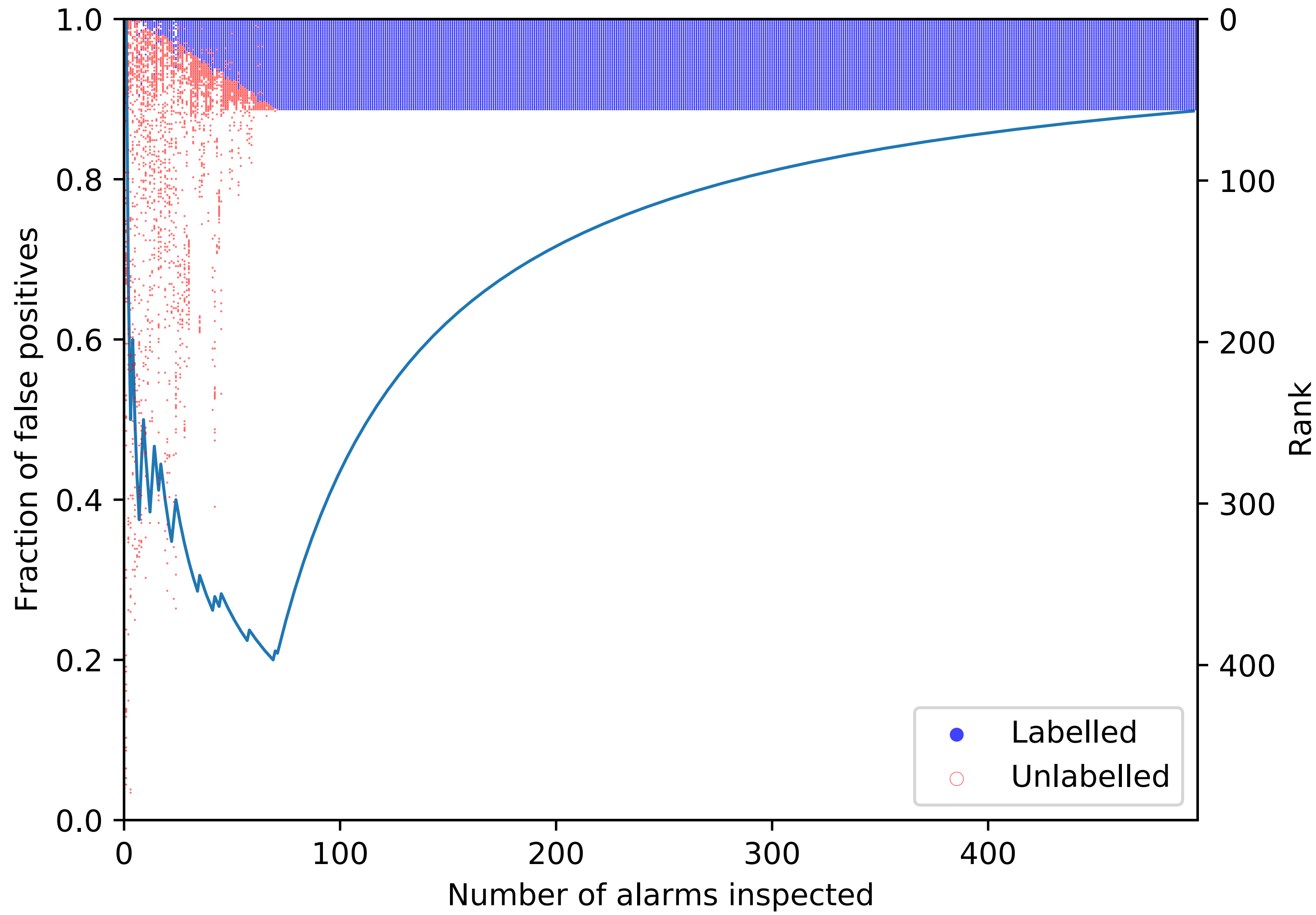
Interactive Alarm Ranker

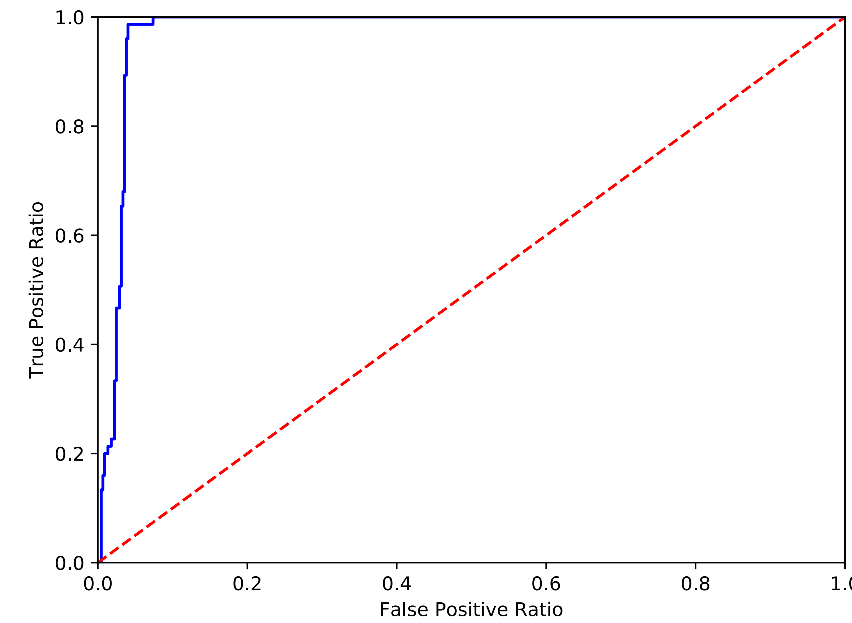
- Key idea: Human in the loop + Bayesian inference



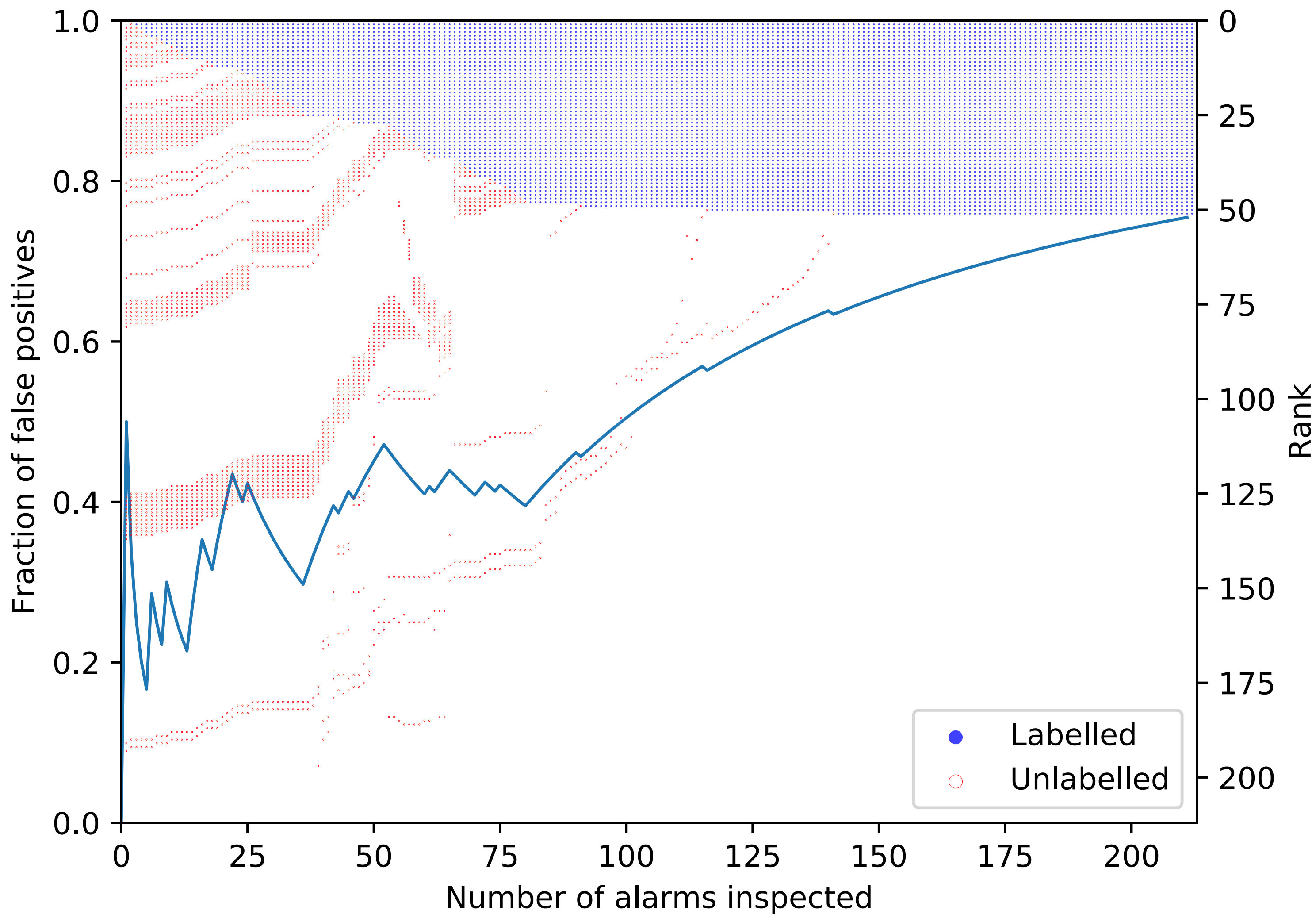
Static Analysis Result **Bayesian Network** **User**

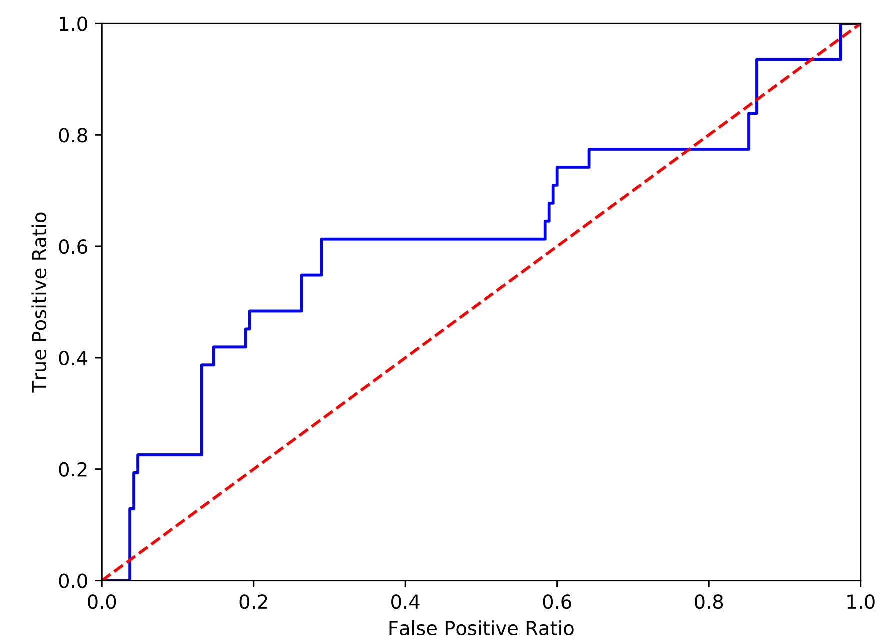
Case Study: Datarace





Case Study: Information Flow





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).

***Apache FTP Server**

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**

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).
```

False alarm

False alarm

***Apache FTP Server**

Derivation Graph

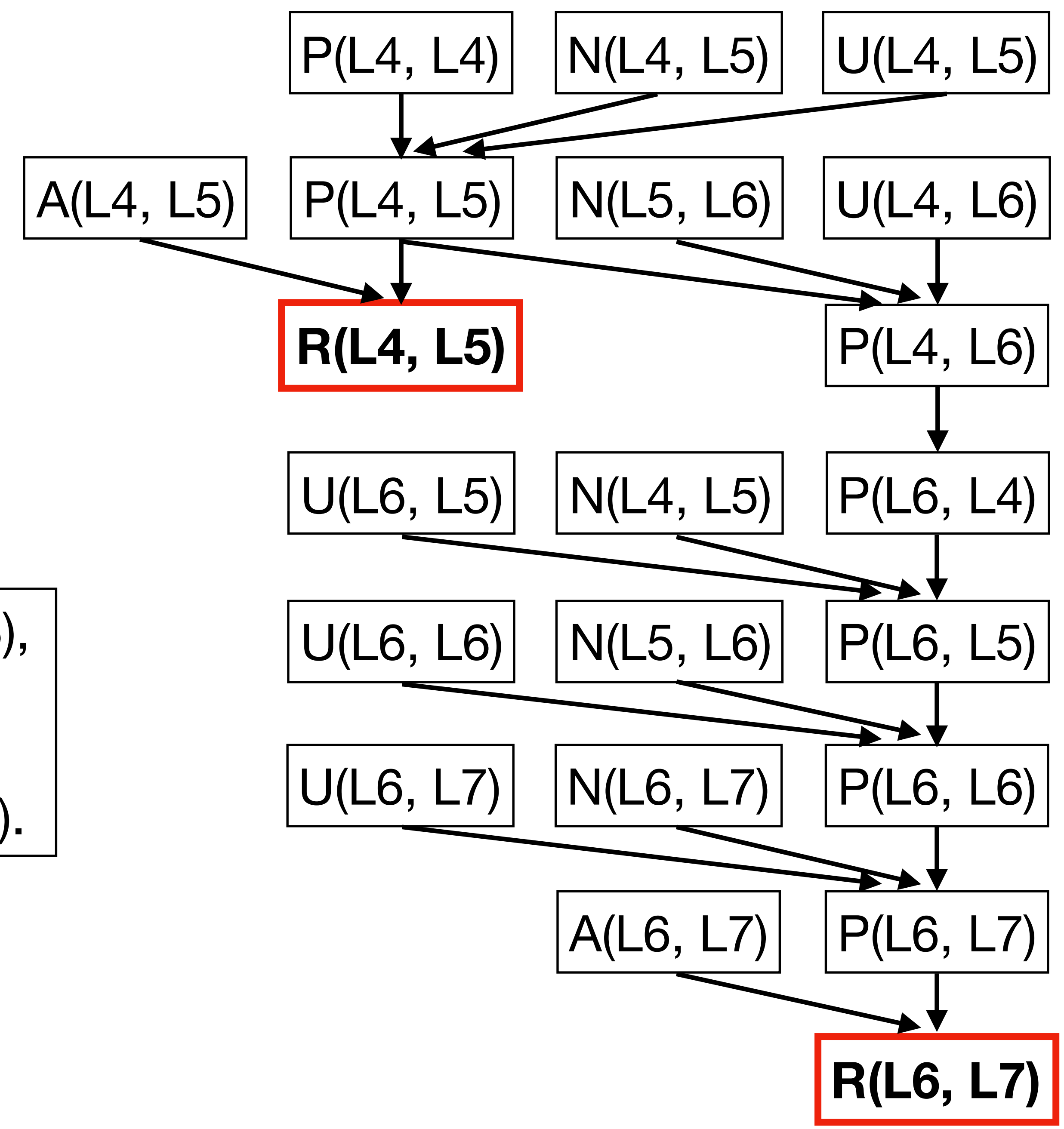
Program

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

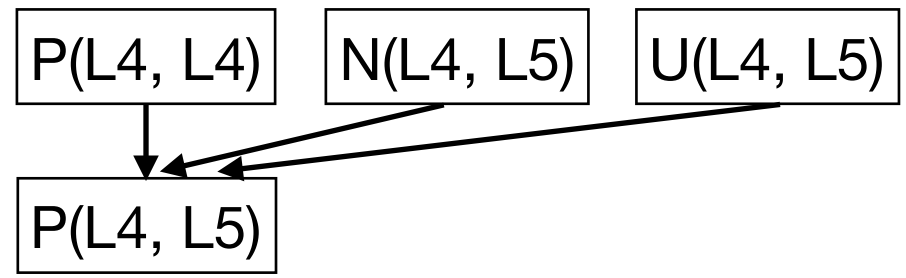
Datalog Rule

```
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

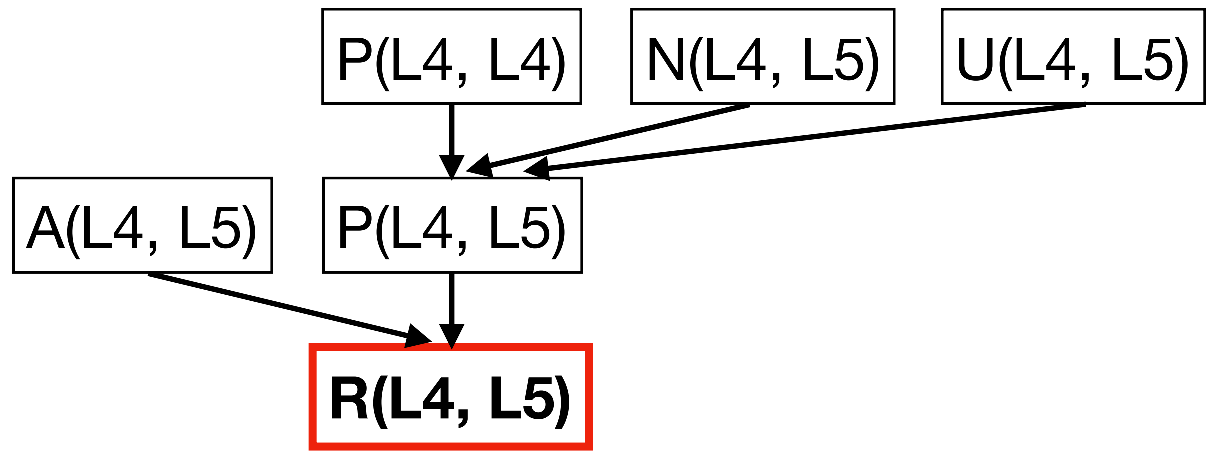
```
Parallel(p1, p3) :- Parallel(p1, p2), Next(p2, p3),
                    Unguarded(p1, p3).
Parallel(p1, p2) :- Parallel(p2, p1).
  Race(p1, p2) :- Parallel(p1, p2), Alias(p1, p2).
```

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

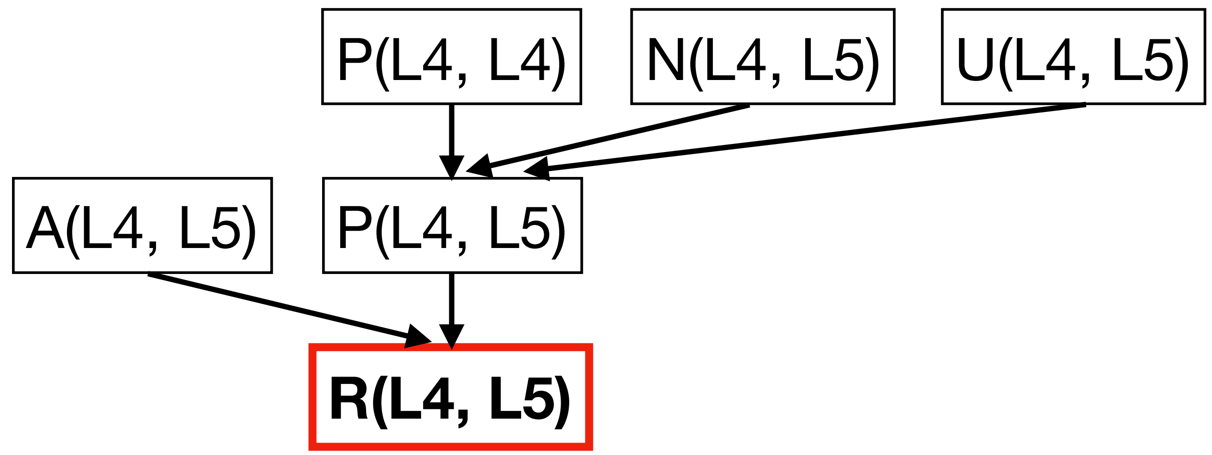
H = P(L4,L4) ∧ N(L4,L5) ∧ U(L4,L5)

Marginal Inference



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

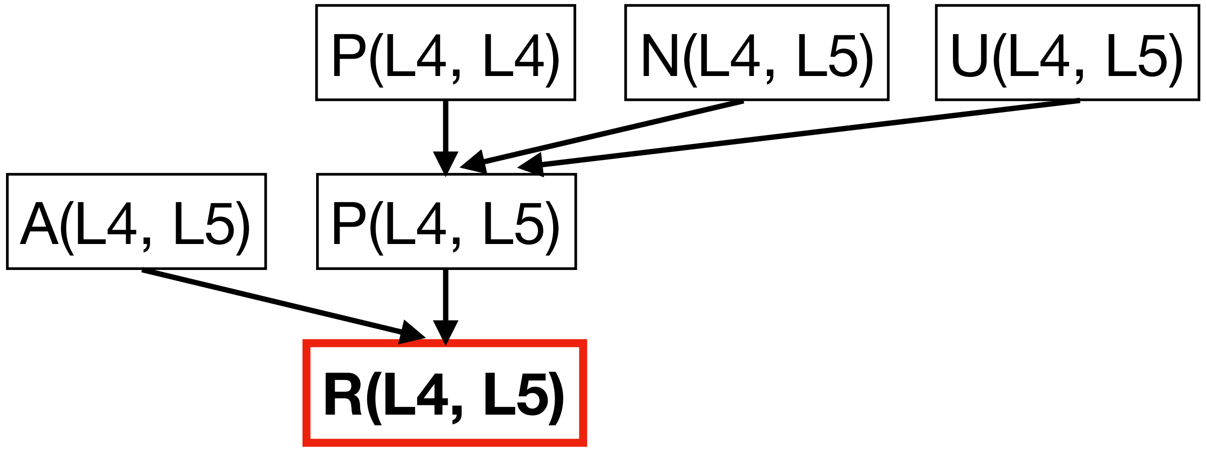
Marginal Inference



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

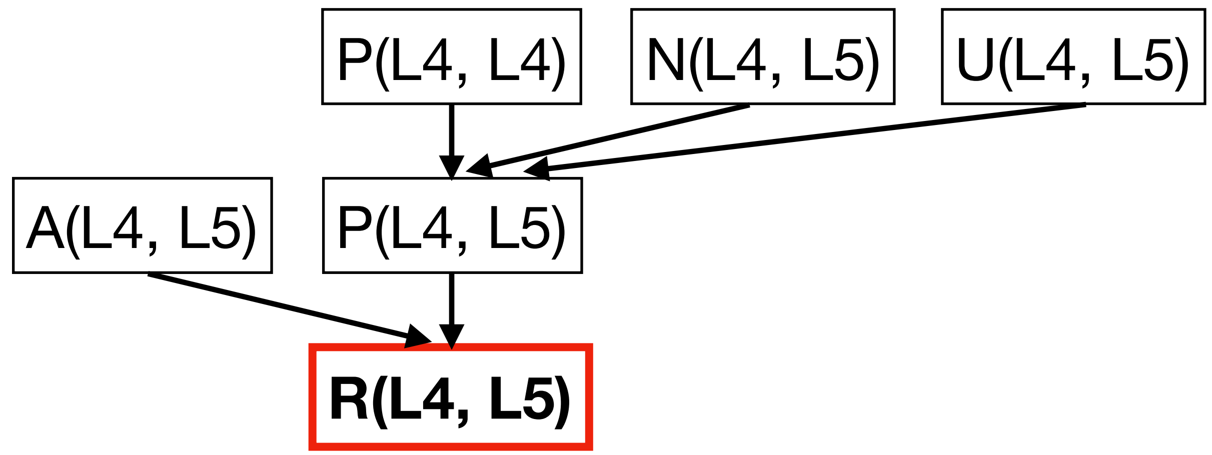
If any of the antecedents fail, then the race cannot happen.

Marginal Inference



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

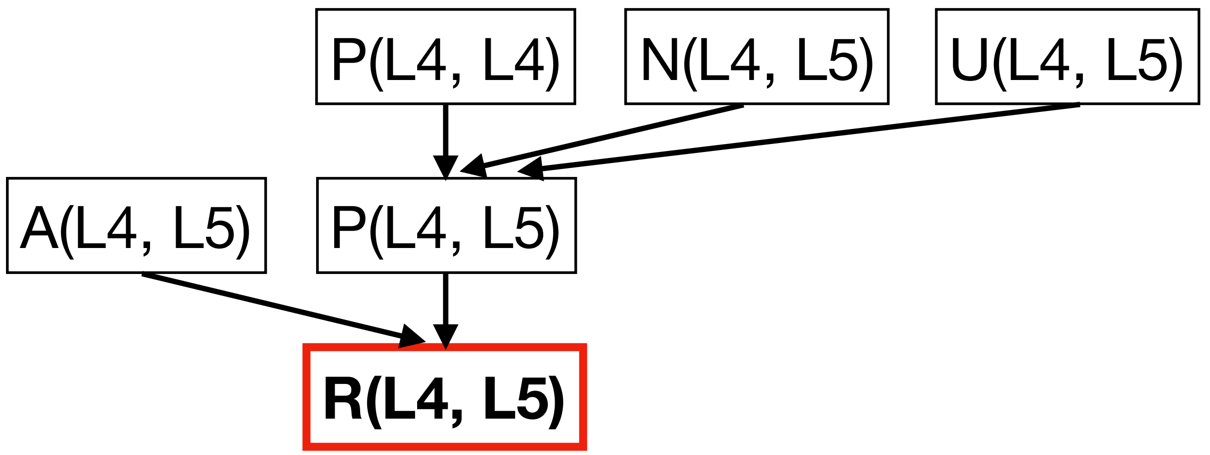
Marginal Inference



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

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

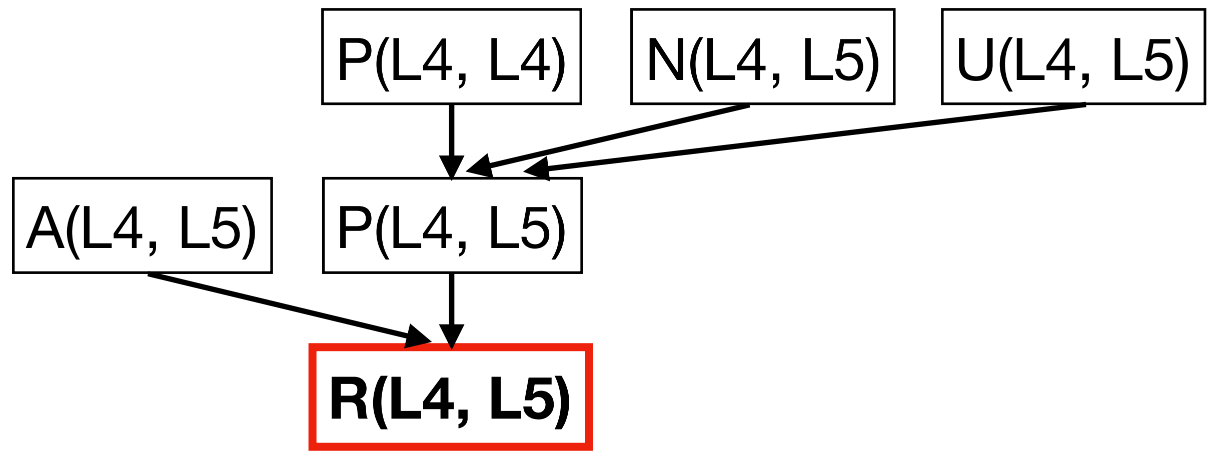
Marginal Inference



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 * Pr(P(L4,L5), Pr(P(L4,L4)), Pr(N(L4,L5), Pr(U(L4,L5))

Assume that the probability of firing each rule and input tuple is 0.95 and 1.0.

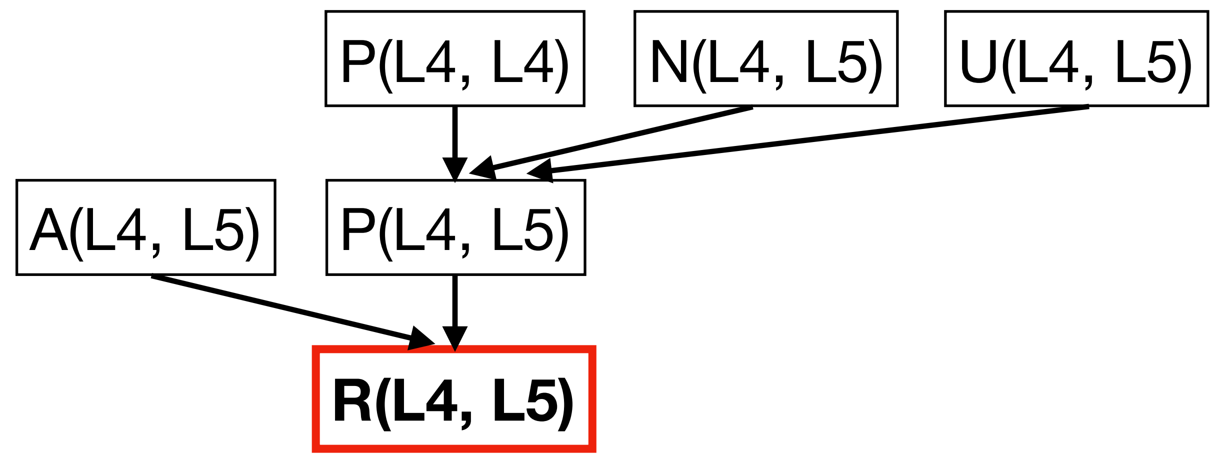
Marginal Inference



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

```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
  }
}
```

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

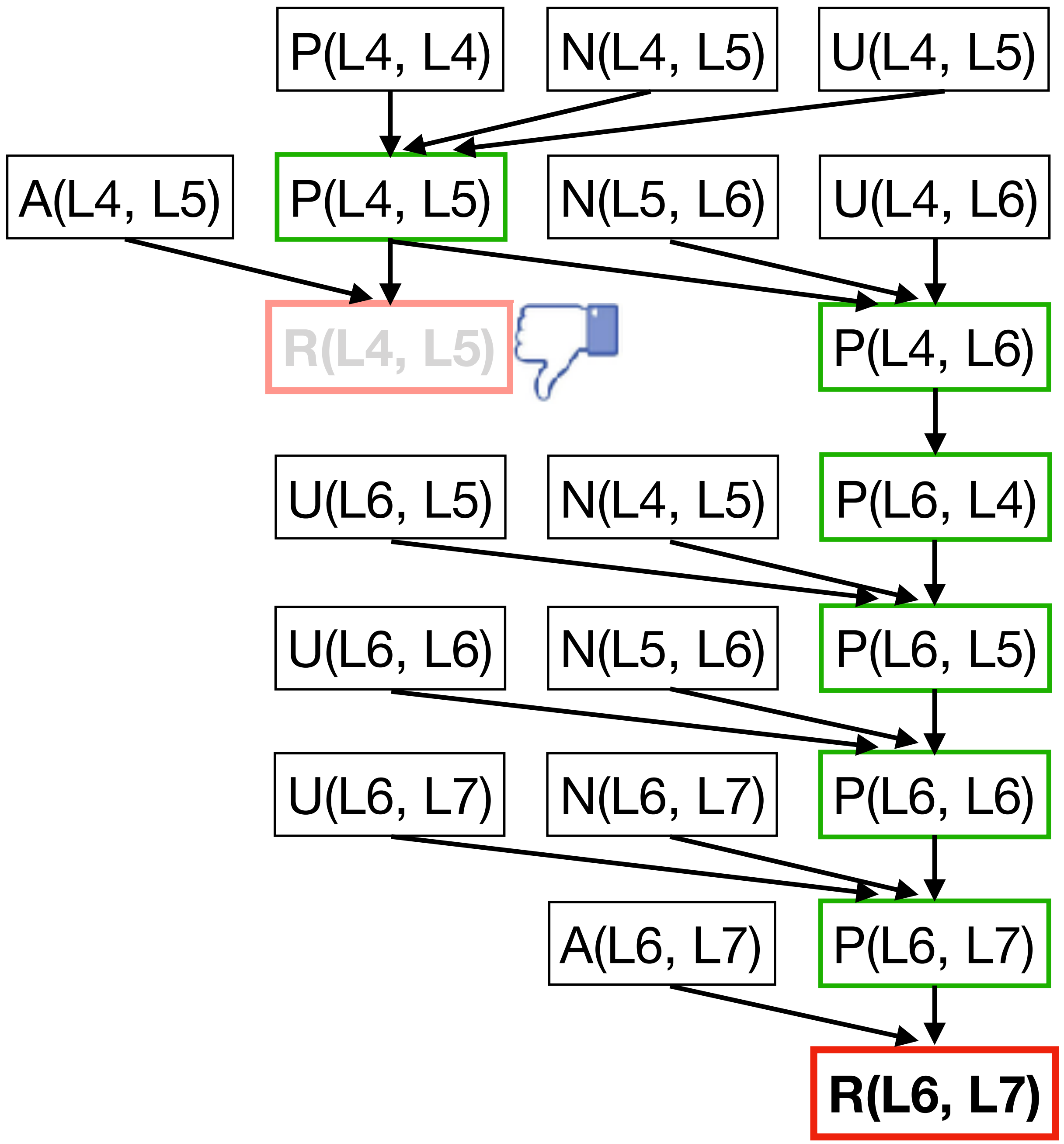
Alarm Ranking

```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
 }
}
```

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?

Alarm Ranking



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

- Datarace

Pgm	#Bugs	#Alarms	#Iters	AUC
hedc	12	152	67	0.81
ftp	75	522	103	0.98
weblech	6	30	11	0.84
jspider	9	257	20	0.97
avrora	29	978	410	0.75
luindex	2	940	14	0.99
sunflow	171	958	838	0.79
xalan	75	1870	273	0.91

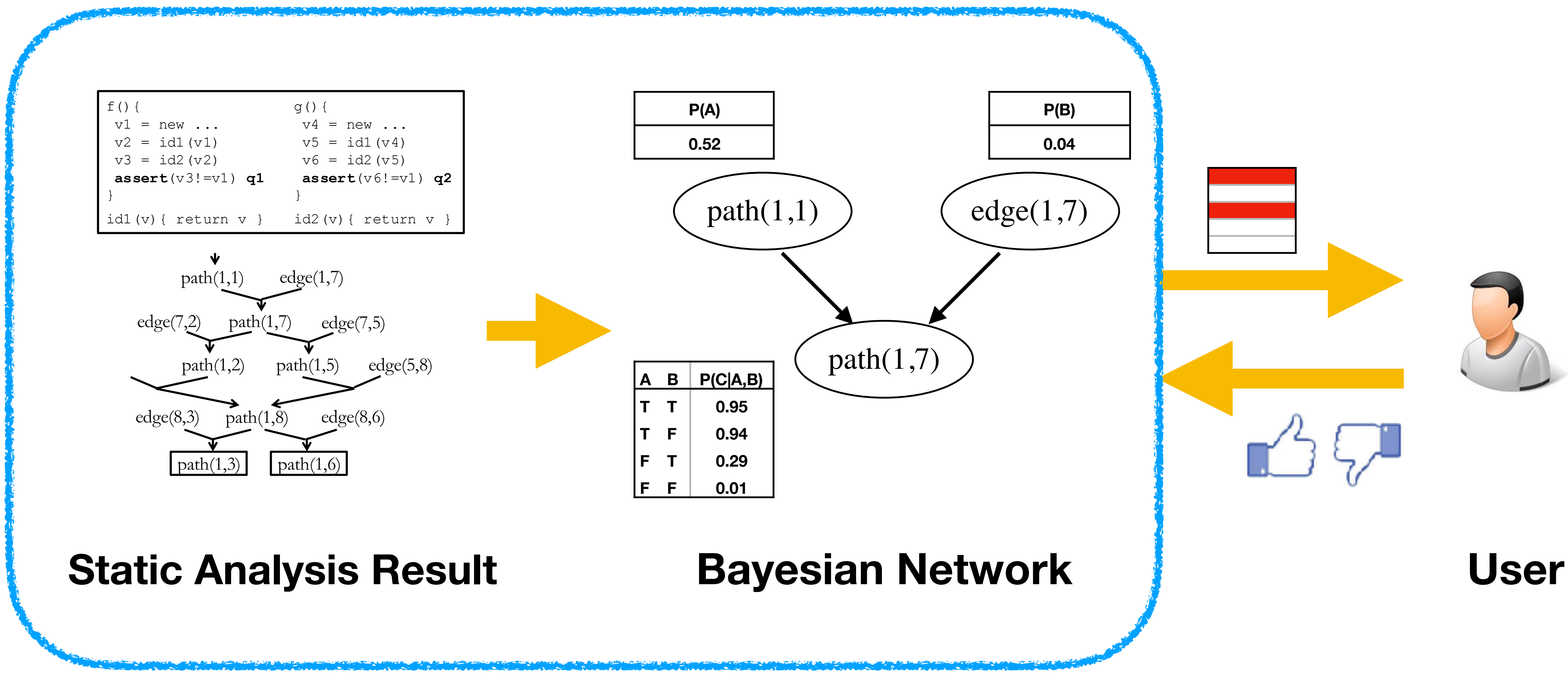
Experimental Results

- Information flow

Pgm	#Bugs	#Alarms	#Iters	AUC
app-324	15	110	51	0.83
noisy-sound	52	212	135	0.89
app-ca7	157	393	206	0.96
app-kQm	160	817	255	0.93
tilt-mazes	150	352	221	0.95
ardors-trail	7	156	14	0.98
ginger-master	87	437	267	0.84
app-018	46	420	288	0.85

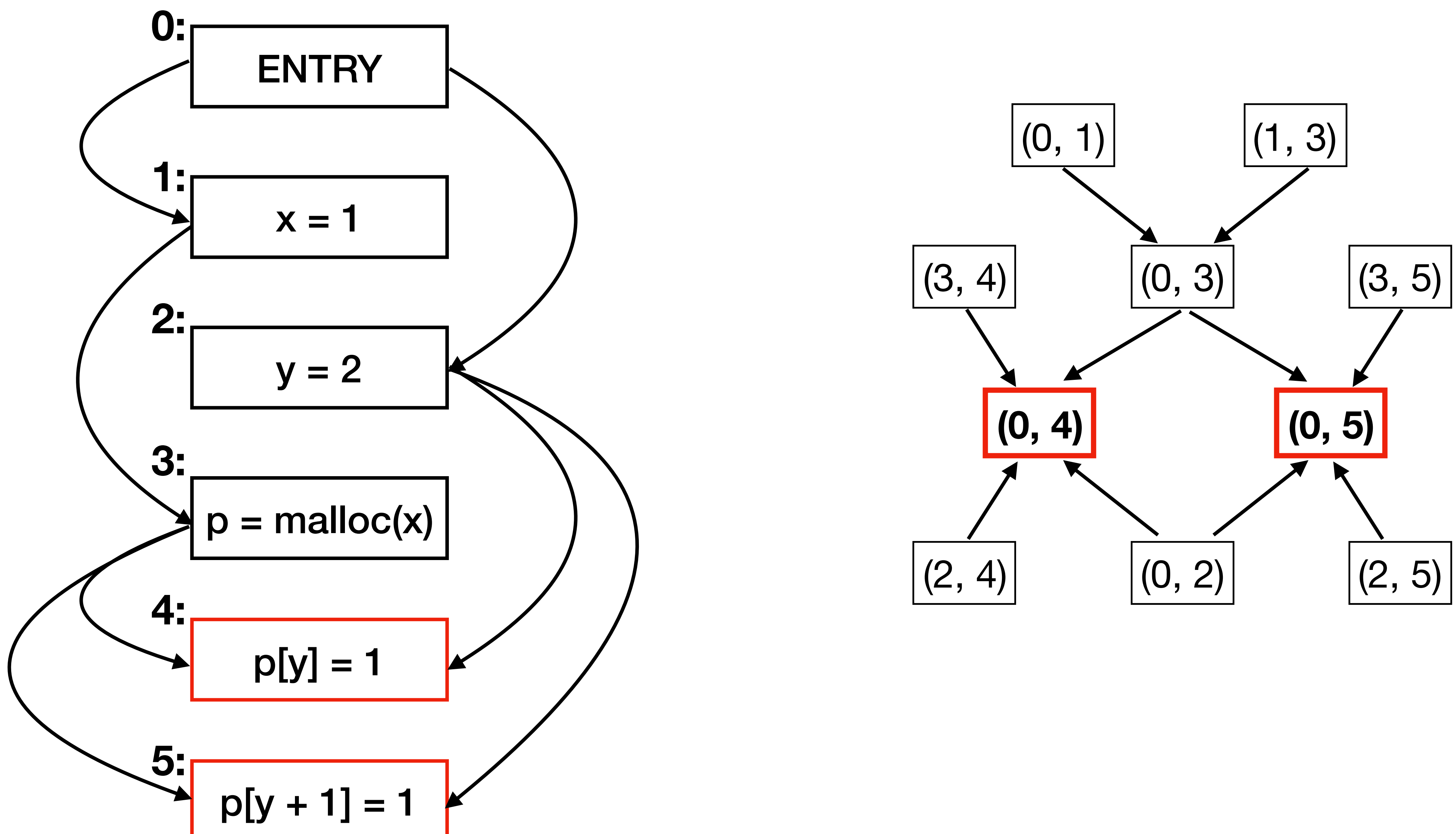
Future Work

- How transform non-datalog analysis results to Bayesian network?



Future Work

- How transform non-datalog analysis results to Bayesian network?



Future Work

- Learning the prior probability distribution
- Optimizing the marginal inference solver
- Transferring the learned knowledge to other programs
- Designing more fine-grained interaction models

Conclusion

- First interactive alarm ranking system
- Logical + probabilistic reasoning using Bayesian network
- Hope to generalize for other static analyses

Conclusion

- First interactive alarm ranking system
- Logical + probabilistic reasoning using Bayesian network
- Plan to generalize for other static analyses

Thank You