Towards 'Verifying' a Water Treatment System

Jingyi Wang
(with Jun Sun, Yifan Jia, Shengcao Qin and Zhiwu Xu)
jingyi_wang@sutd.edu.sg

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

SHENZHEN UNIVERSITY 1983

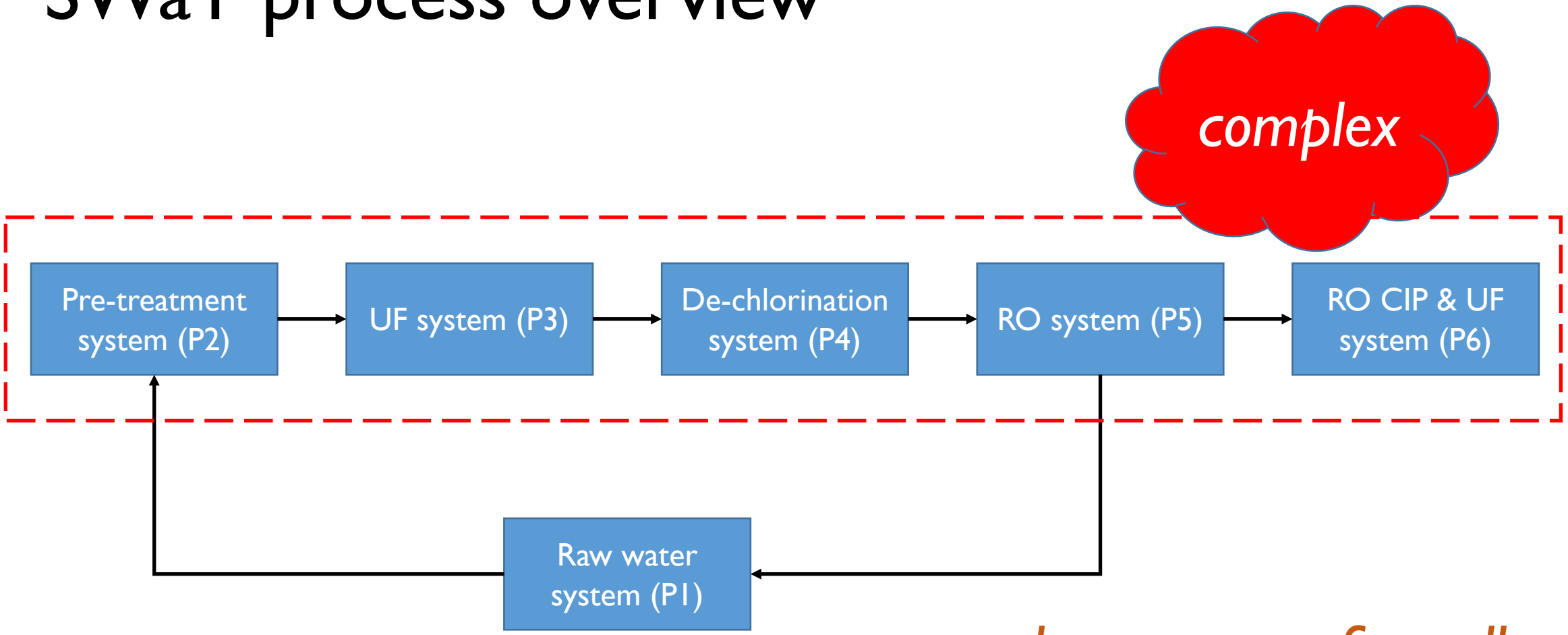
TÜV SÜD

Teesside University

The Secure Water Treatment System (SWaT)

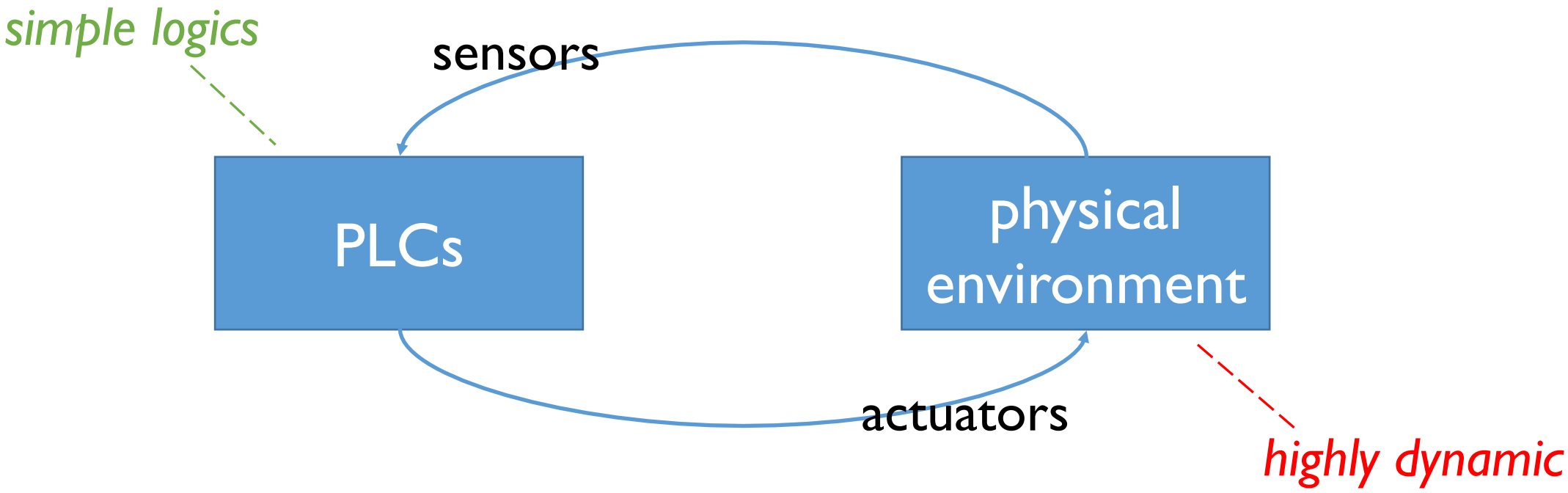
- *fully functioning*
- *supervised experiments*
- *public dataset*
- *a wide range of research areas*

SWaT process overview



how can we formally 'verify' the system?

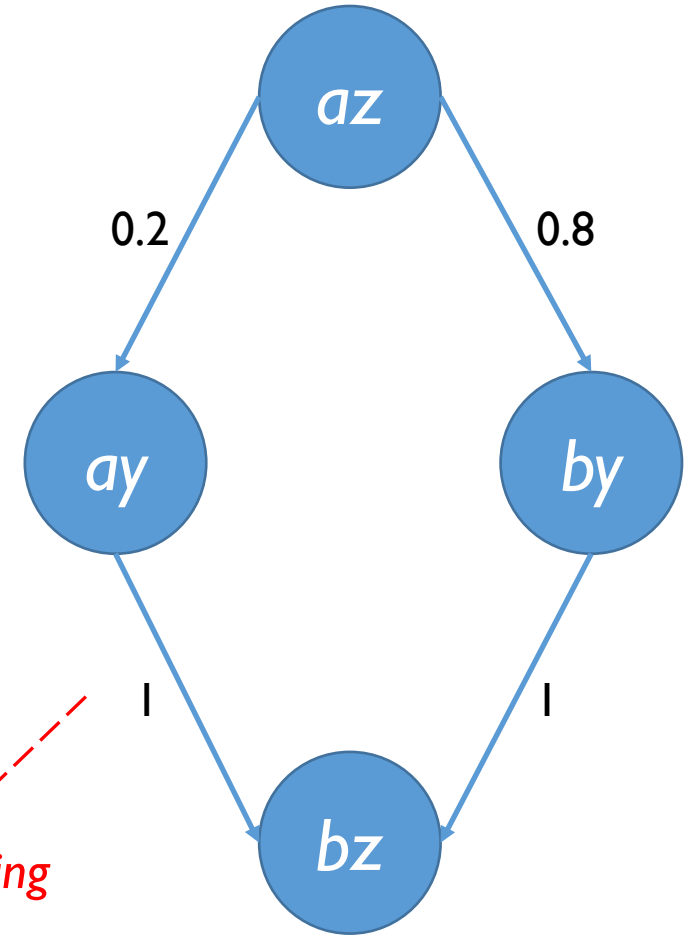
Modelling CPS



Can we learn a probabilistic model instead?

time	sensors	actuators
t0	a	z
t1	a	y
t2	b	y
t3	b	z
…	…	…

learning



probabilistic model checking
runtime monitoring…

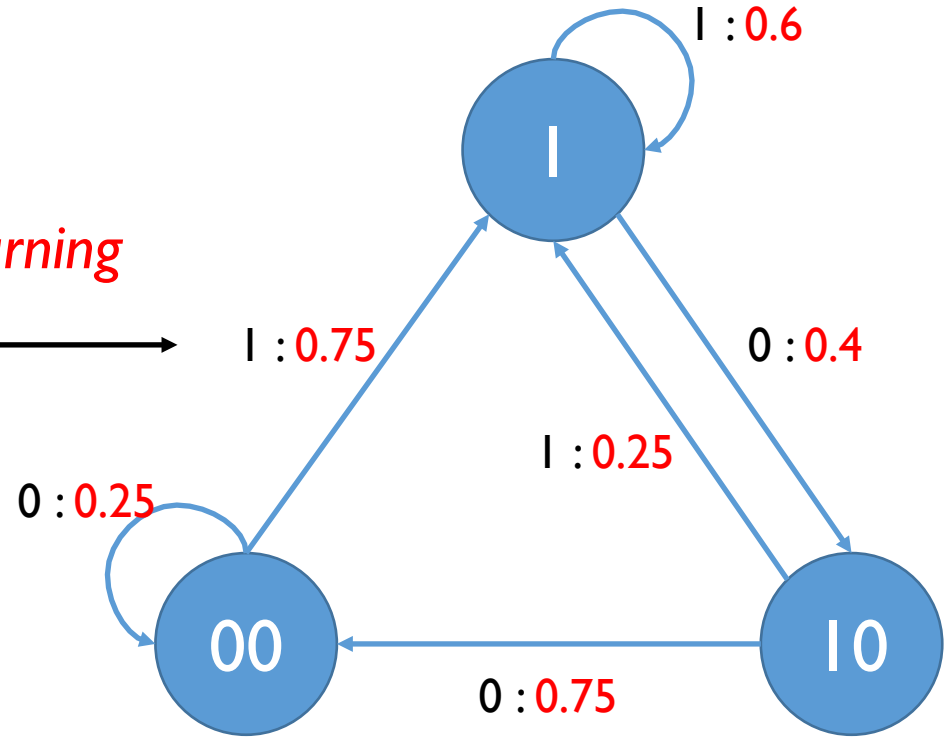
Probabilistic learning sketch

alphabet: {0, 1}

input: 0010001000111…

probabilistic learning

output model:



SWaT dataset

- *26 sensors (float) + 25 actuators*
- *7 days system log under normal operation + 4 days system log under attacks*
- *28800 + 208800*

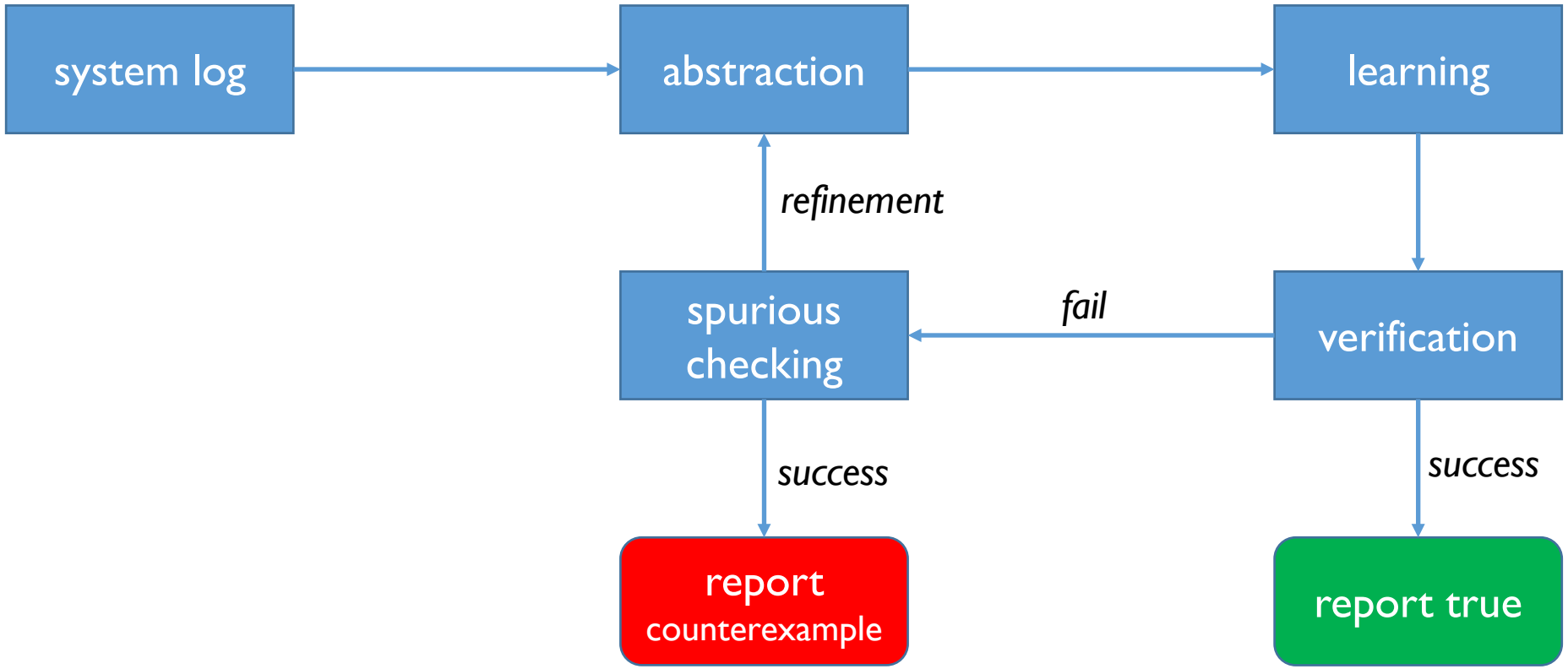
Predicate abstraction

$FIT101$	$LIT101$	$MV101$	$P101$	$P102$	$AIT201$	$AIT202$	$AIT203$	$FIT201$
2.470294	261.5804	2	2	1	244.3284	8.19008	306.101	2.471278
2.457163	261.1879	2	2	1	244.3284	8.19008	306.101	2.468587
2.439548	260.9131	2	2	1	244.3284	8.19008	306.101	2.467305
2.428338	260.285	2	2	1	244.3284	8.19008	306.101	2.466536
2.424815	259.8925	2	2	1	244.4245	8.19008	306.101	2.466536
2.425456	260.0495	2	2	1	244.5847	8.19008	306.101	2.465127
2.472857	260.2065	2	2	1	244.5847	8.19008	306.101	2.464742

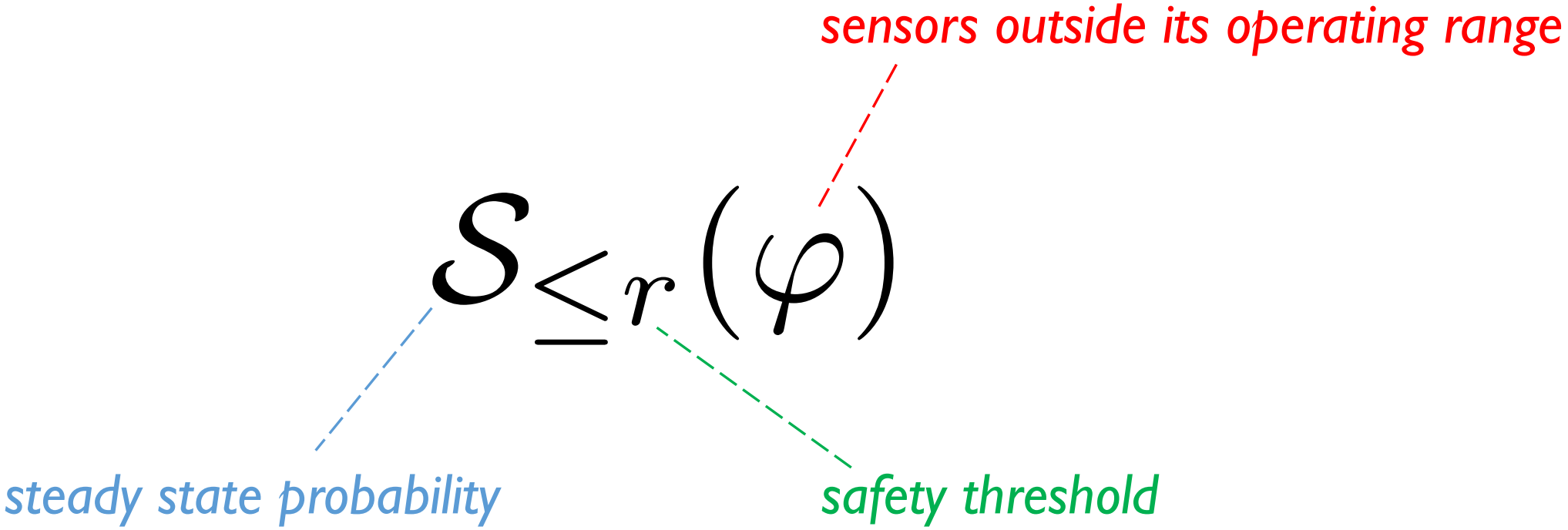
LIT101>1100

$LIT101 > 1100$
0
0
0
0
0
0
0

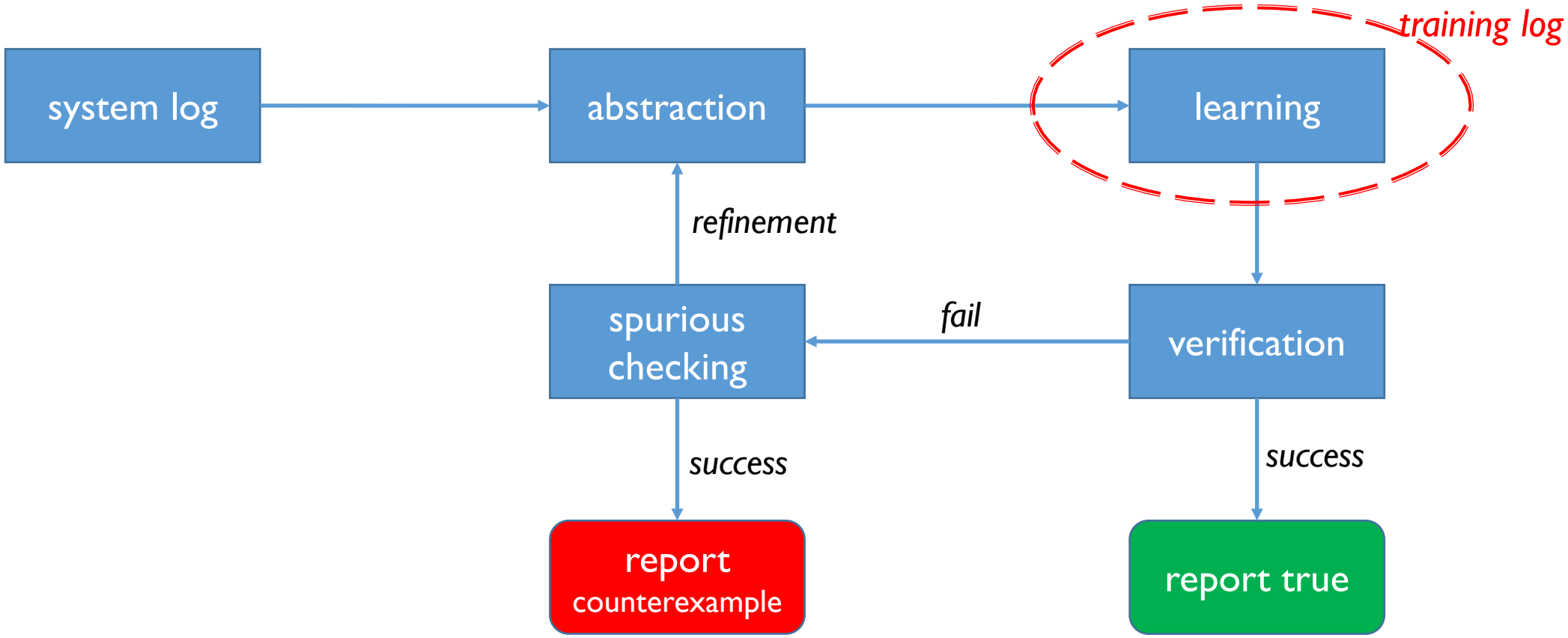
Overall framework



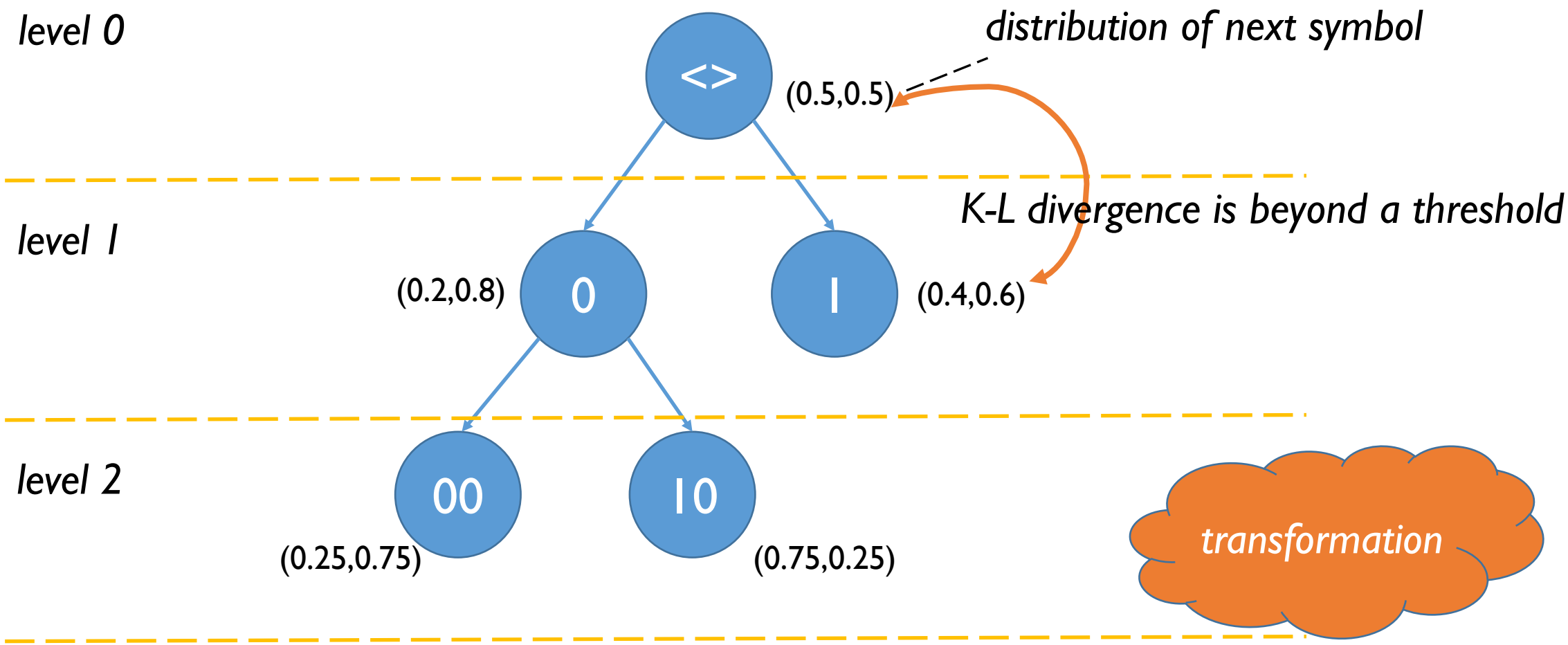
Safety properties



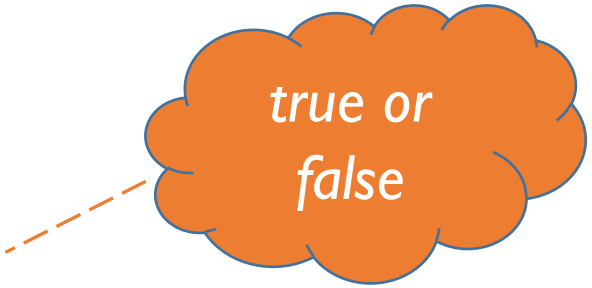
Learning


system log
abstraction
learning
training log
refinement
spurious checking
fail
verification
success
success
report counterexample
report true

Learning


level 0
level 1
level 2
<>
(0.5,0.5)
distribution of next symbol
K-L divergence is beyond a threshold
(0.2,0.8)
0
1
(0.4,0.6)
00
10
(0.25,0.75)
(0.75,0.25)
transformation

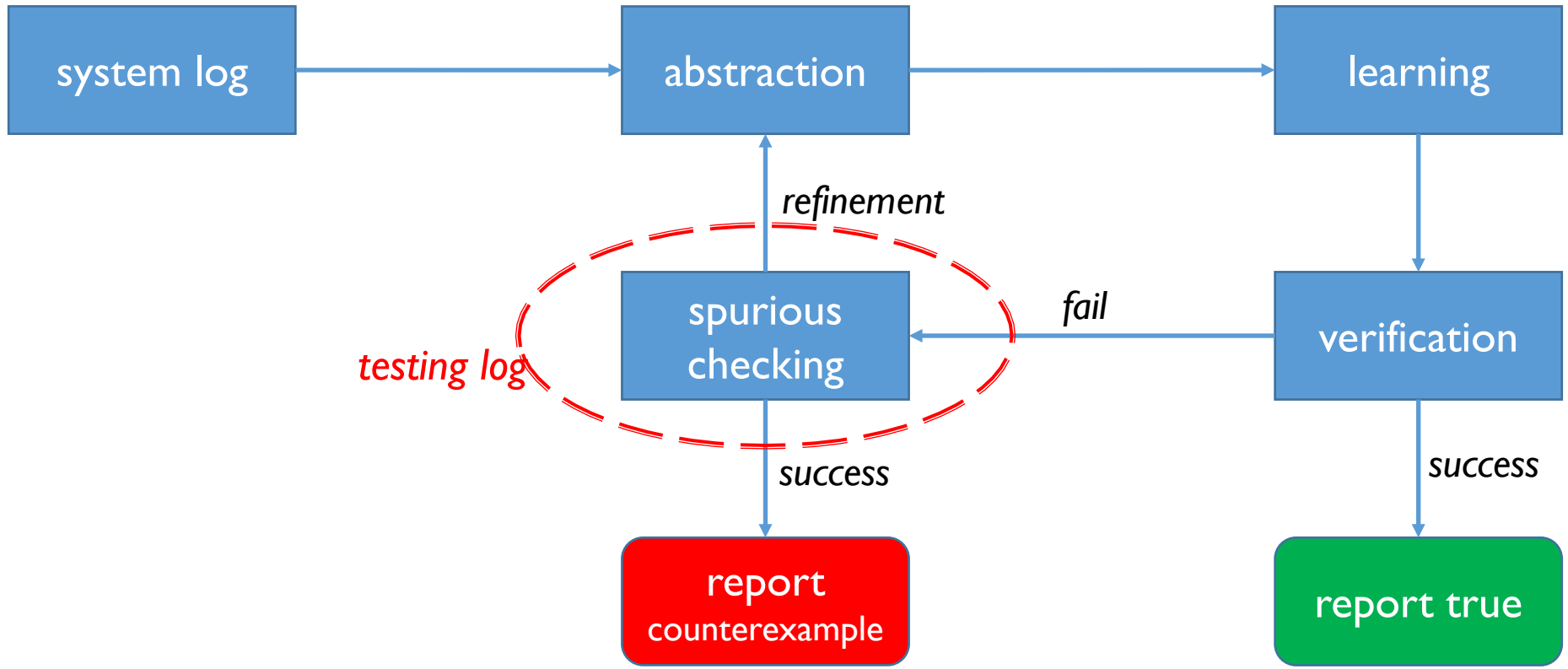
Verify a safety property



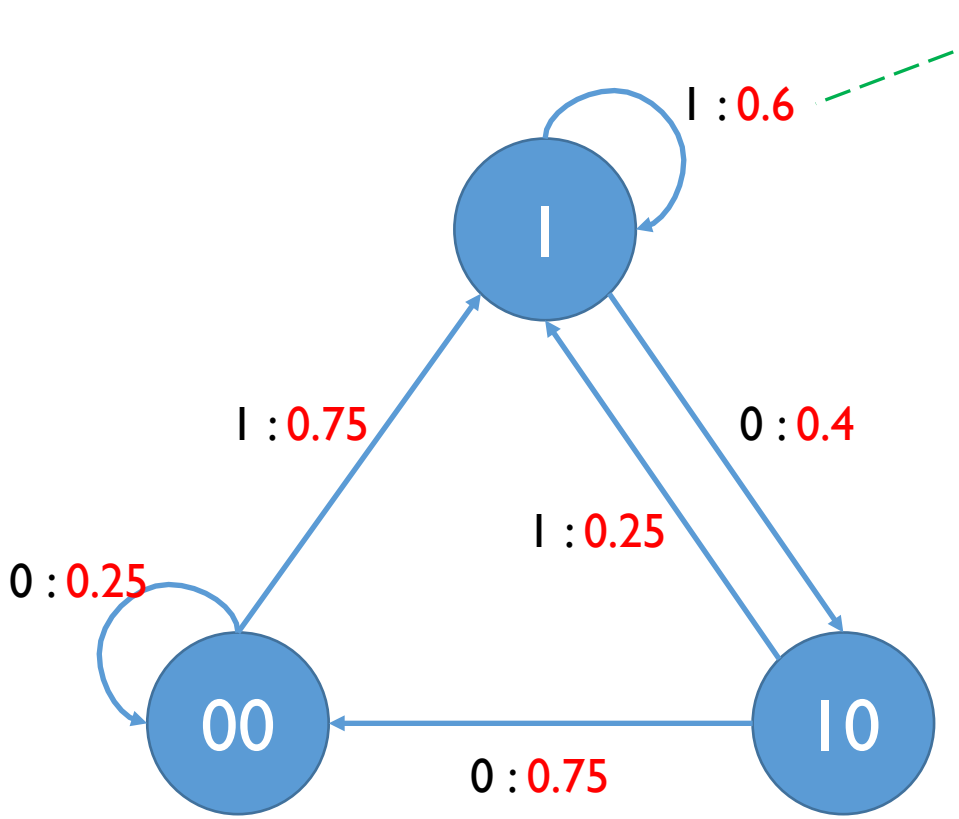
$$\mathcal{S}(\varphi) \leq r \quad ?$$

1. *compute steady state probability distribution of the learned model*
2. *sum up the probability of unsafe states*

Spurious checking



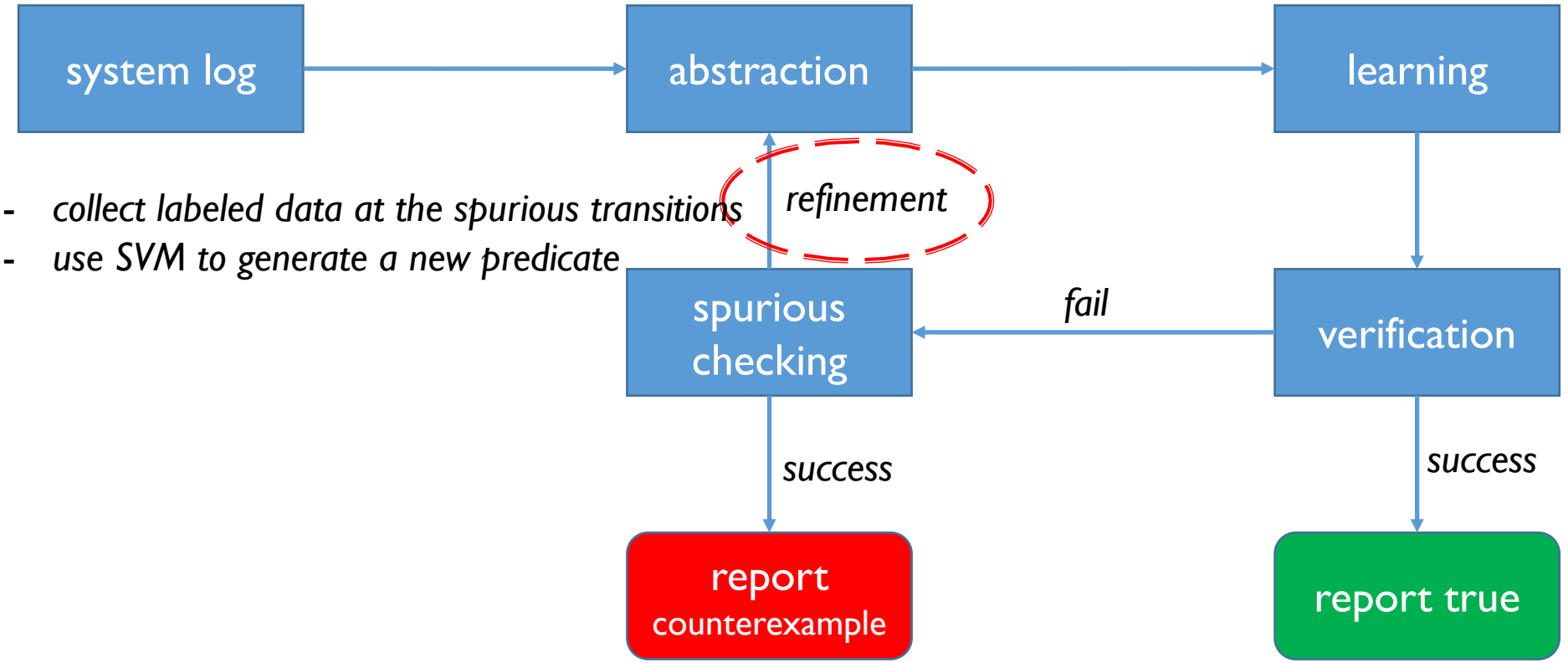
Find spurious transitions



- *calculate transition probabilities in the testing log*
- *compare the differences with the learned probability*
- *select those inflated transitions*

Refinement

- *collect labeled data at the spurious transitions*
- *use SVM to generate a new predicate*



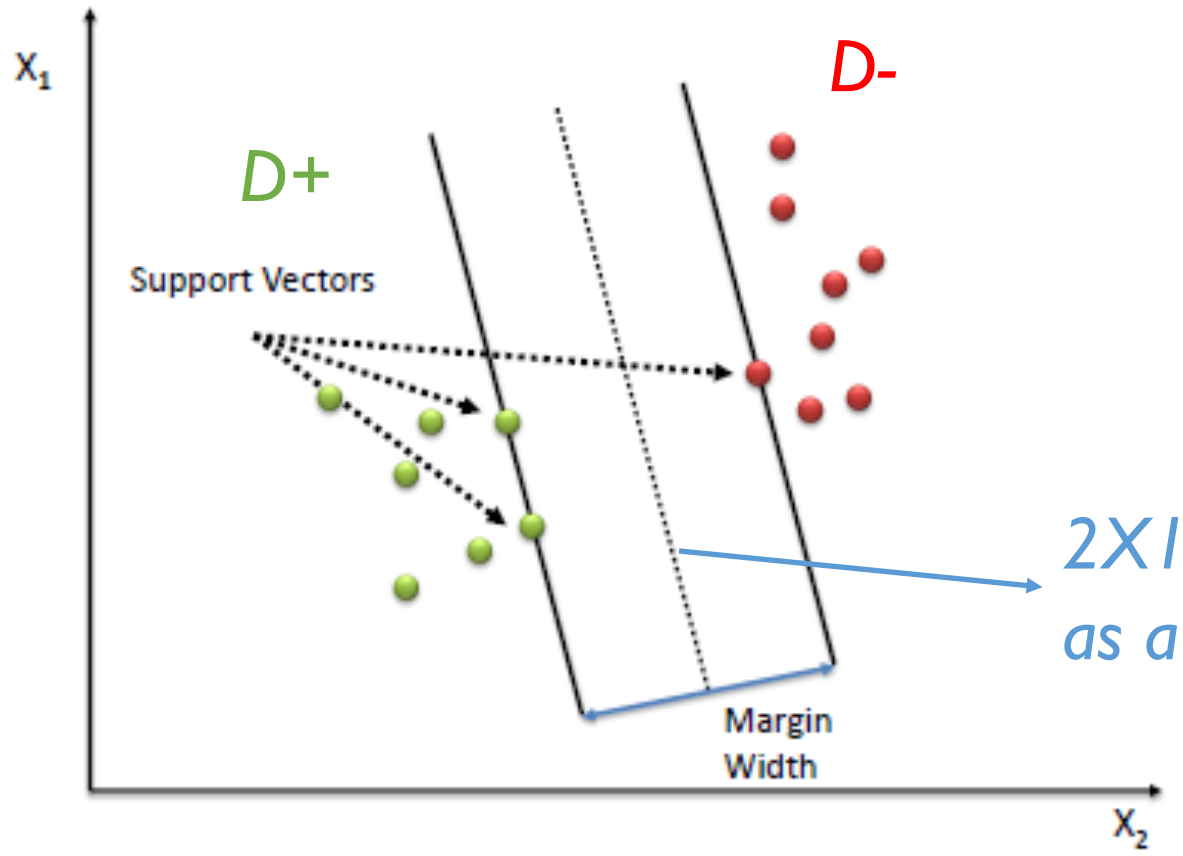
Collect labeled data

spurious transition: $1 \xrightarrow{1} 1$

system log: 0 0 I 0 0 0 I I 0 0 …

D- *D+*

Learn a new predicate by SVM



2X1+X2=1

as a new predicate

Summary of result

47 properties	*19 never violated*
	24 verified
	4 violations

details at https://github.com/wang-jingyi/Ziqian

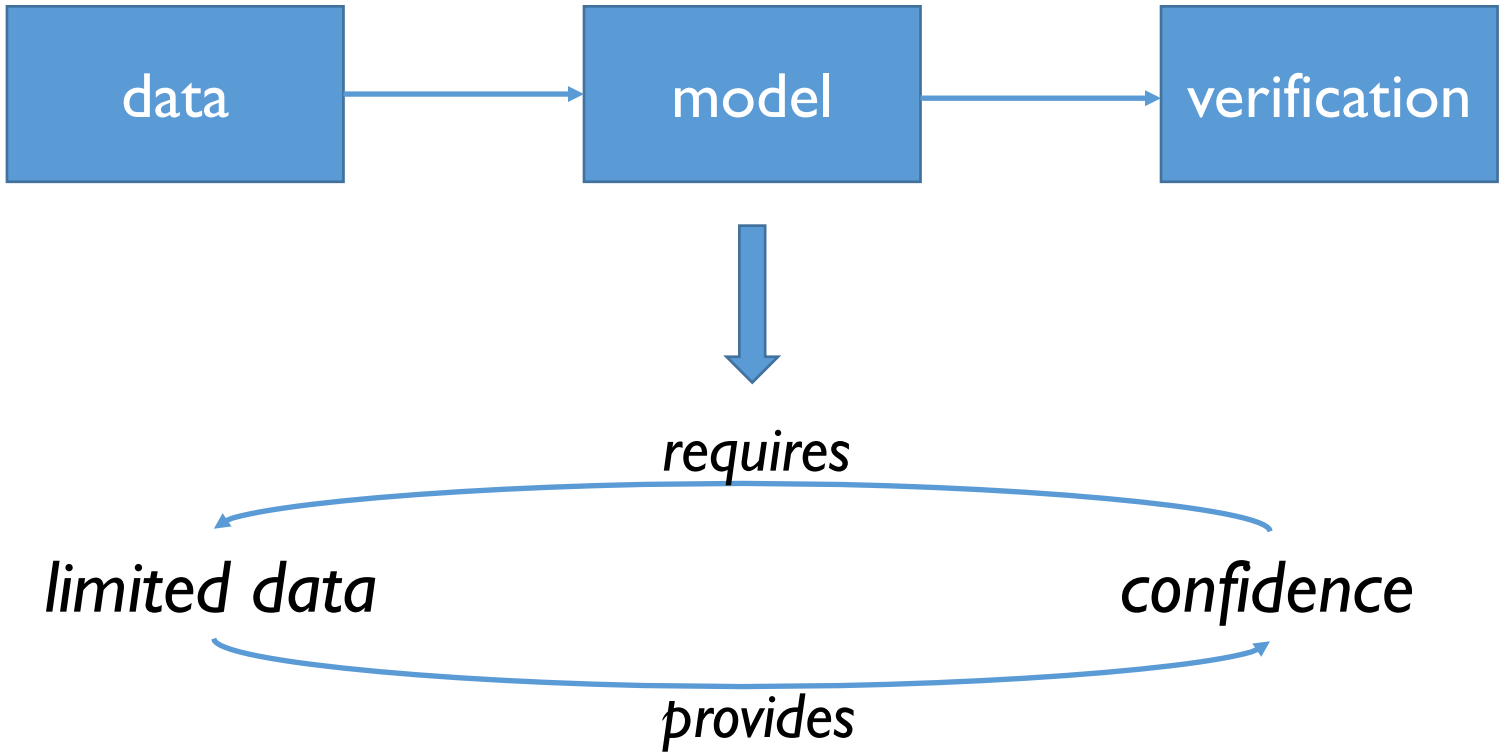
Observations

- *the learned models are precise*
- *the learned models are small*
  - *2 to 208 states*
- *group behaviors*
  - *FIT401<1.5, FIT502<1.1, FIT503<0.7, FIT504<0.25*
  - *FIT501<1, PIT501<20, PIT503<10*
- *safety violations*
  - *AIT401>100, PIT501>30, PIT502>0.2, PIT503>20*
  - *high in the training log, l in learned model and the testing log*

Discussions

- *safety margin: 20%*
- *hyper parameter in the learning algorithm*
- *sub-sampling*
- *limited data*

Ongoing and future work



some preliminary results at https://link.springer.com/chapter/10.1007/978-3-319-66335-7_23

Our facilities are available!

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

iTrust
Centre for Research in Cyber Security

Take-home points

- *experience on automatic verification of a real-world CPS from data*
- *applied an abstraction-based learning algorithm*
- *the learned models are potentially used for subsequent analysis*