

Debug Determinism

The Sweet Spot for Replay-Based Debugging

Cristian Zamfir, Gautam Altekar, George Candea, Ion Stoica

EPFL, UC Berkeley



ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE



Debugging is Hard

- hard to reproduce non-deterministic bugs
- may take days-months to diagnose¹

¹ *Concurrency at Microsoft – An Exploratory Survey*, P. Godefroid and N. Nagappan, CAV Workshop on Exploiting Concurrency Efficiently and Correctly, 2008

Replay Debugging

User

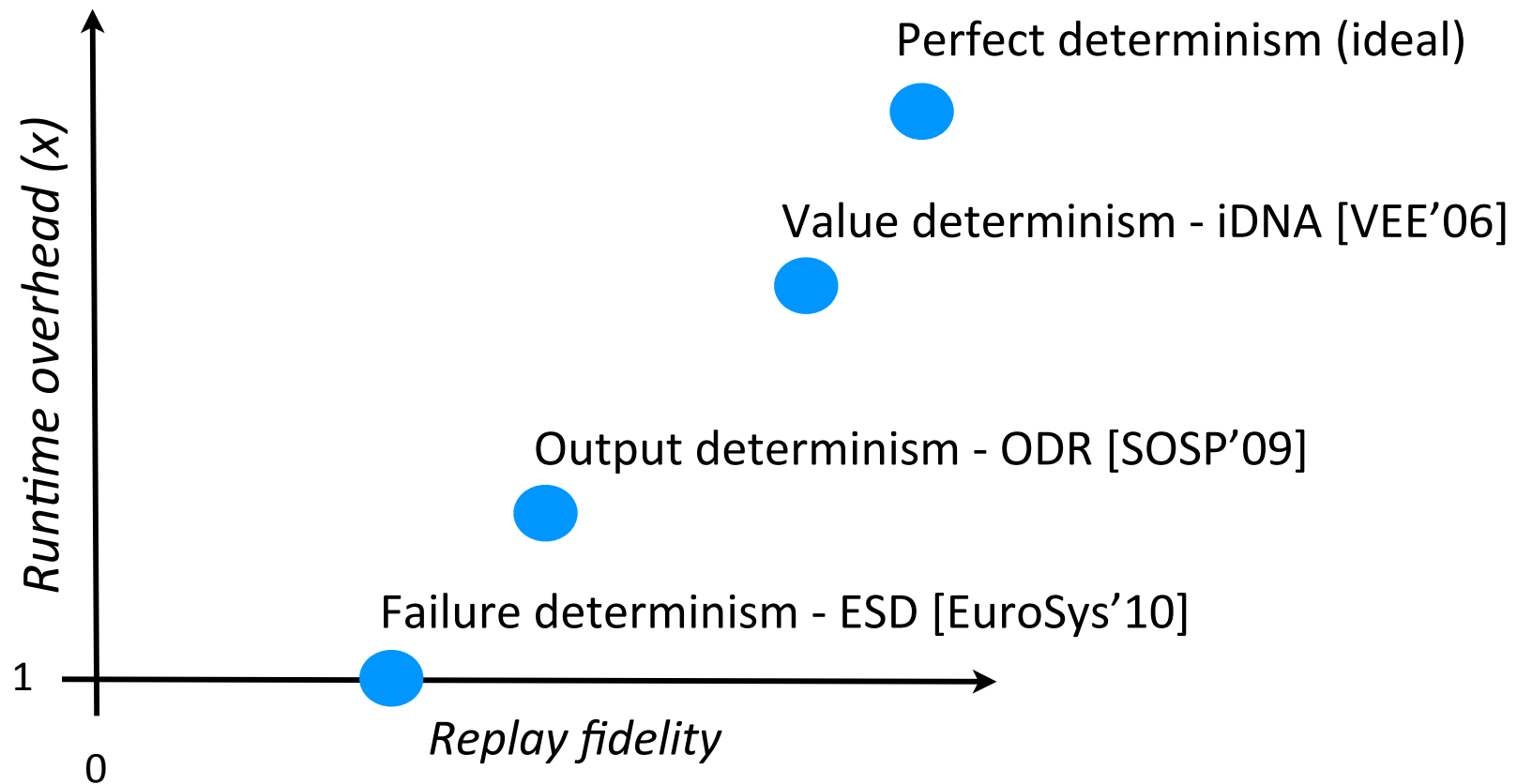
Developer



Challenge: low overhead recording

Relaxed Determinism

Determinism = replay fidelity



Perils of Over-Relaxation

Thread 1

```
int a = readSocket();  
int b = readSocket();  
  
printf("%d", a+b);
```

Thread 2

```
b++;
```

$2 + 2 = 5$
data race

```
> record ./add
```

```
5
```

```
> replay --last
```

```
./add
```

```
5
```

$2 + 3 = 5$
correct

Some systems may not reproduce

- *the root cause*
- *the failure*

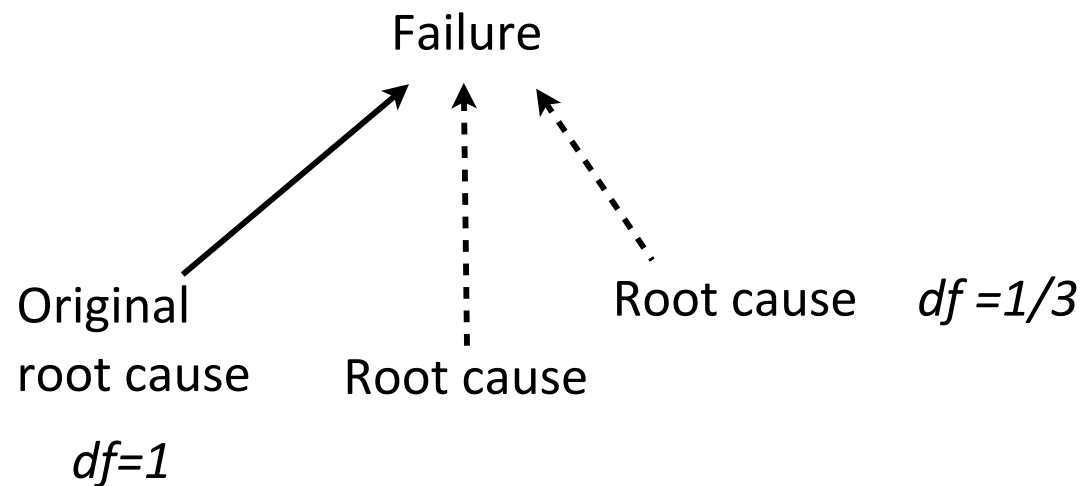
=> cannot debug

Debug Determinism

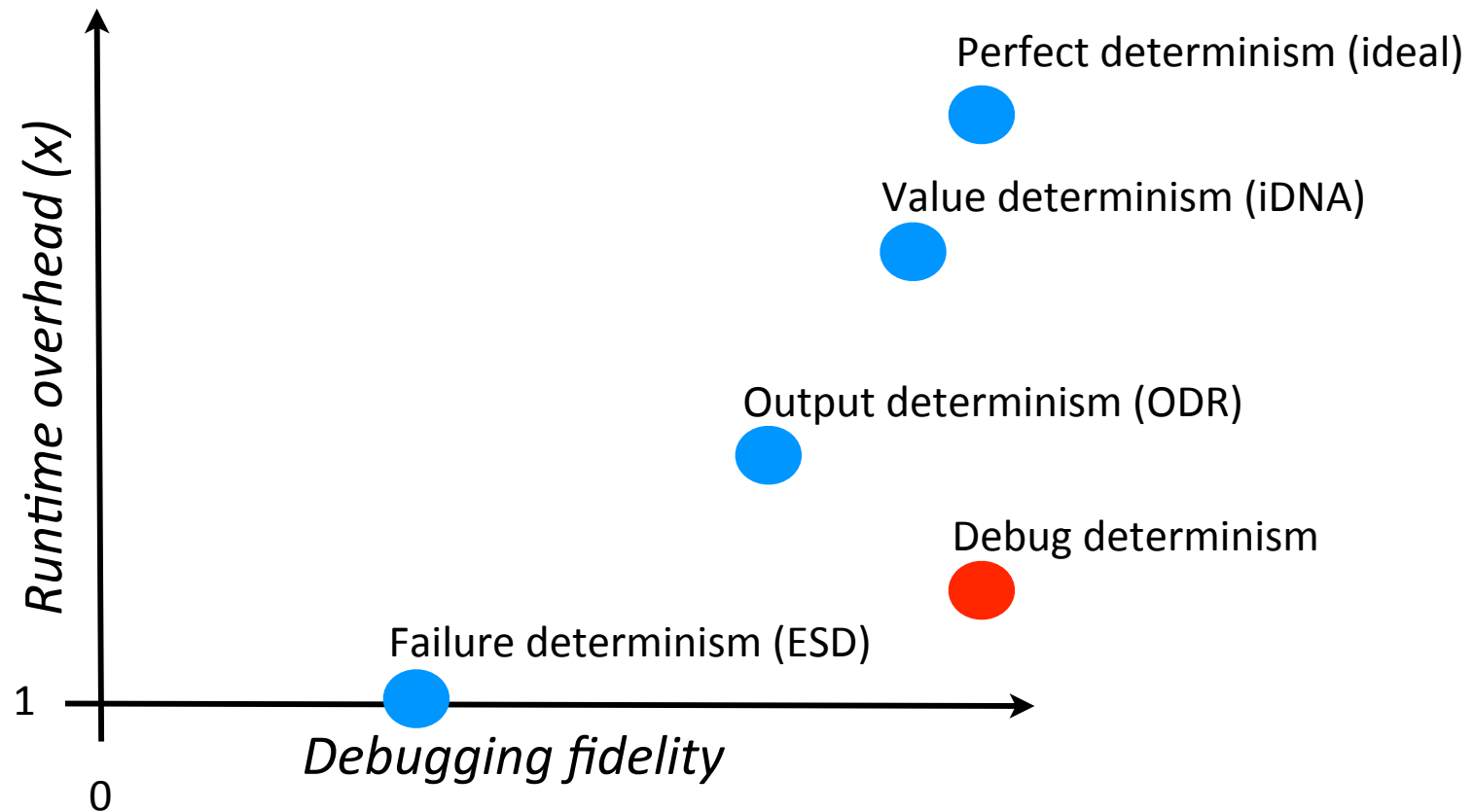
To be useful for debugging, should at a minimum reproduce the original failure and root cause.

Debugging Fidelity

Debugging fidelity = the ability to reproduce the original root cause and the failure



Factoring in Debugging Fidelity



Achieving Debug Determinism

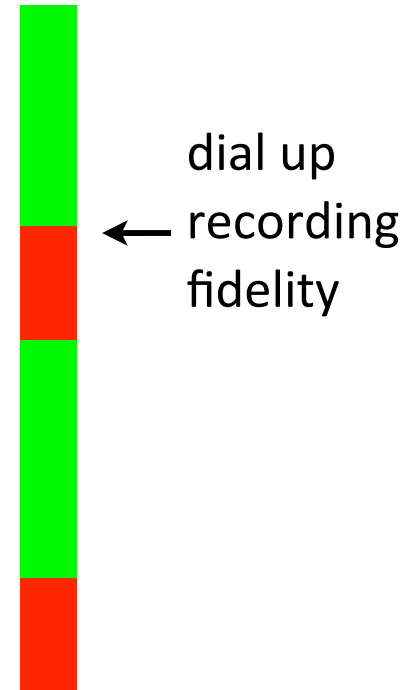
- Record with high fidelity

- *the root cause*
- *the failure*

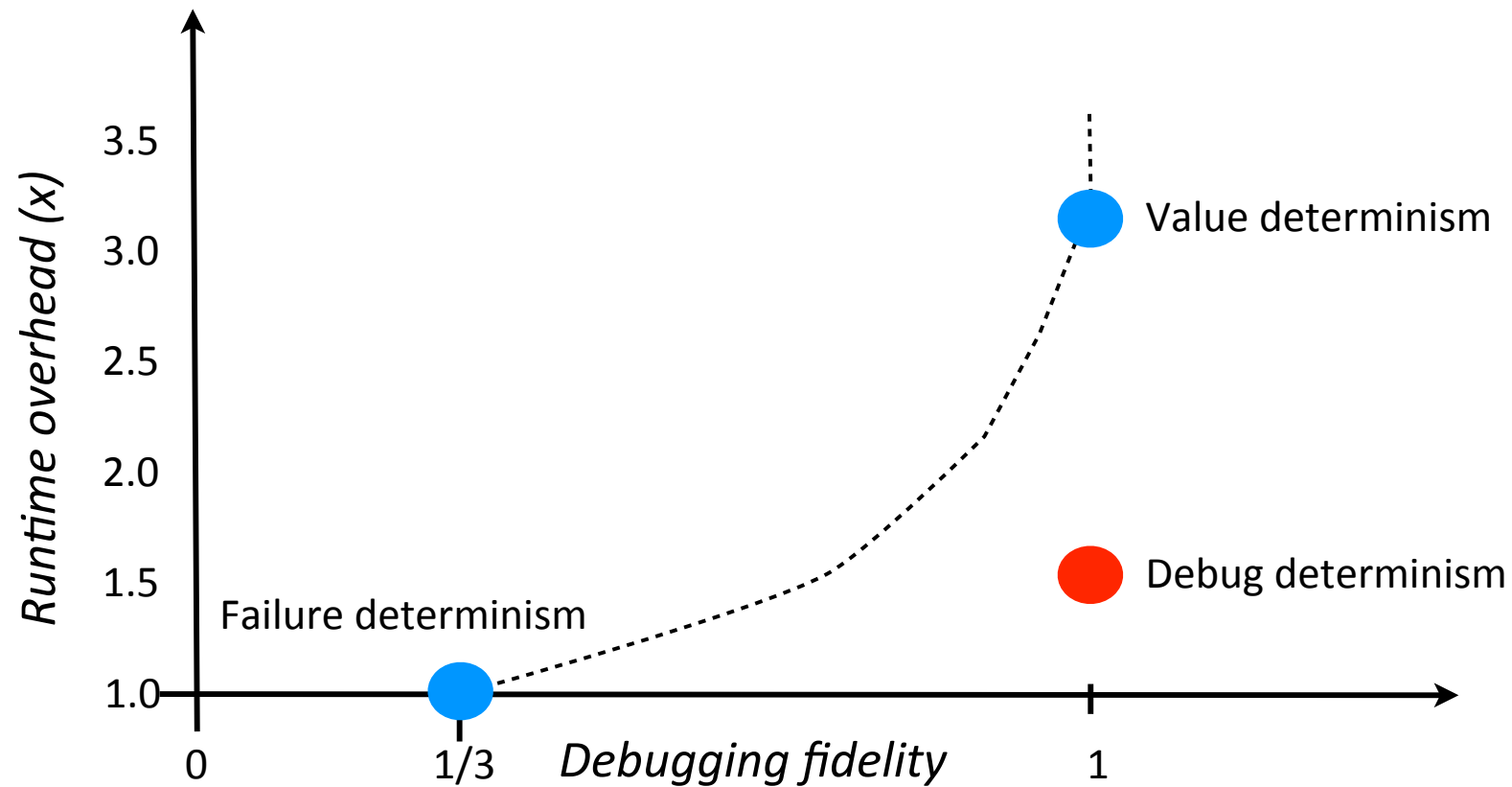
- Challenge

- *the root cause and the failure are not known a priori*

- Key idea: guess the location of possible root causes



Example: data corruption bug in Hypertable



Topics for Open Mic

- Debug determinism: the right model?
- How to achieve debug determinism?