THE UNIVERSITY OF TEXAS AT AUSTIN

Error and fault abstractions

Mattan Erez

UT Austin

Who should care about faults and errors?

Ideally, only system cares about **masked faults**?

- Assuming application bugs are not called faults
- Assuming system reporting for analysis
- Assuming automatic rebalancing when needed

What is the cost of masking faults?

- Many faults masked naturally
- Many faults are not

Error correction requires error detection

- Both aren't free
- But, many **errors masked** by algorithm/application

Which errors can be masked / detected cheaply?

- Application dependent
- How do we find out?

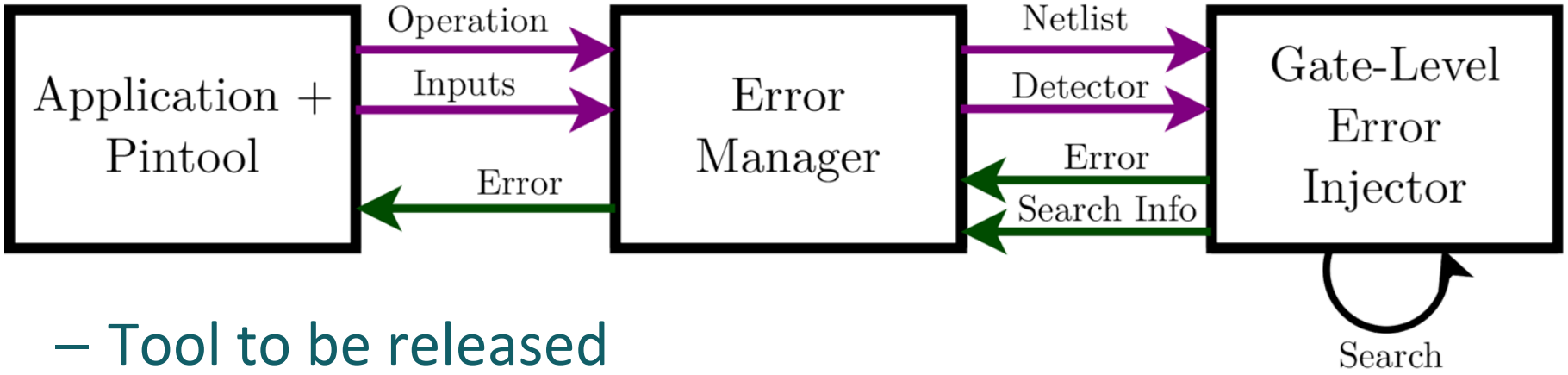
Application or system error models?

System models are hard

- Bottom up
- Large space
- Input dependent
- Approximations not really known

Quick(ish) way to search the error space

- Multi-mode simulation
- Skip over detectable errors



- Tool to be released
  - Uses only public tools

Requirement 1:

What is the result of an injection experiment?

- Automatic correctness check
- Check has to be believable
- Check should be fast

Requirement 2:

Abstraction for detectable / masked abstractions for error patterns

- Magnitude of noise terms?
- Requirements on error rates?

Application-level models needed

Another example?

Racy code with occasional stale data

- F. Niu et al., "Hogwild", NIPS'11
  Racy Parallel Stochastic Gradient Decent

Proposition 4.1 *Suppose in Algorithm 1 that the lag between when a gradient is computed and when it is used in step j — namely, $j - k(j)$ — is always less than or equal to τ, and γ is defined to be*

$$\gamma = \frac{\vartheta \epsilon c}{2LM^2\left(1 + 6\rho\tau + 4\tau^2\Omega\Delta^{1/2}\right)}. \tag{11}$$

for some $\epsilon > 0$ and $\vartheta \in (0, 1)$. Define $D_0 := \|x_0 - x_\star\|^2$ and let k be an integer satisfying

$$k \geq \frac{2LM^2\left(1 + 6\tau\rho + 6\tau^2\Omega\Delta^{1/2}\right)\log(LD_0/\epsilon)}{c^2\vartheta\epsilon}. \tag{12}$$

Then after k updates of x, we have $\mathbb{E}[f(x_k) - f_\star] \leq \epsilon$.

What do we do with masked errors?

- What you can’t measure …

Applications and system must coordinate error handling

- (Standard) abstractions needed
  - That make sense to both system and algorithm
- End-to-end checkable benchmarks needed
  - With multiple scales, times, and inputs
  - Ideally with methodology for interpreting results