EFFICIENT EXPLORATORY TESTING OF CONCURRENT SYSTEMS

Jiri Simsa, Randy Bryant, Garth Gibson, Jason Hickey* (CMU, *Google)

MOTIVATION

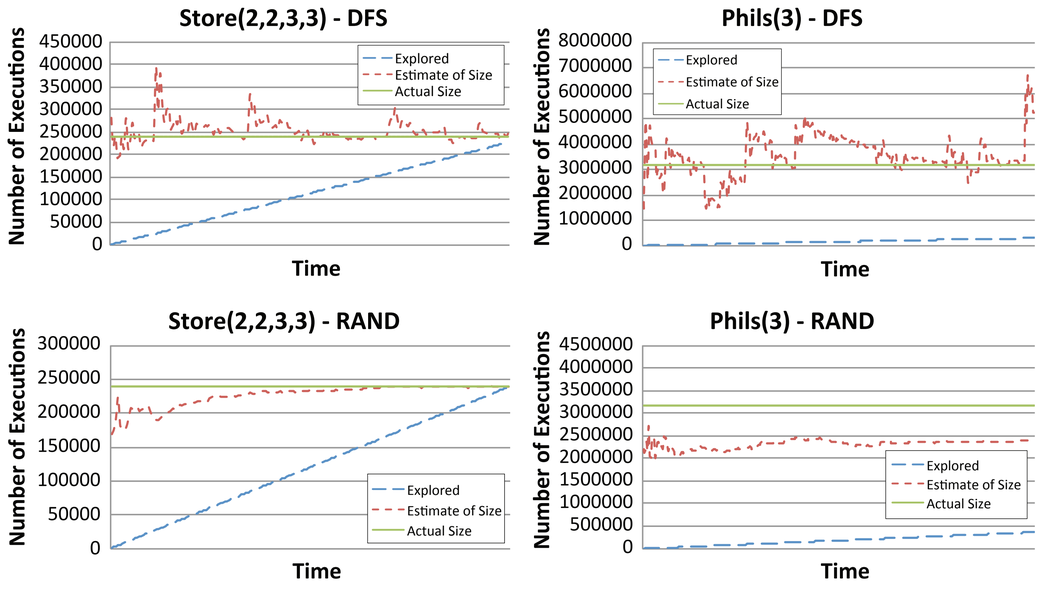
Exploratory testing of concurrent systems:

- Promising approach that tackles test non-determinism
- Platform for mitigation of combinatorial explosion
- Successfully used to find bugs even in well-tested systems:
  - MaceMC [Killian '07]: 52 bugs
  - CHESS [Musuvathi '08]: 27 bugs
  - Modist [Yang '09]: 35 bugs
  - dBug [Simsa '11]: 25 bugs

Number of bugs gives little indication of testing efficiency:

- Is exploratory testing more efficient than random testing?
- What else besides bug finding is exploratory testing good for?
- Can time to completion of an exploratory test be predicted?
- How to scale the performance of exploratory testing?

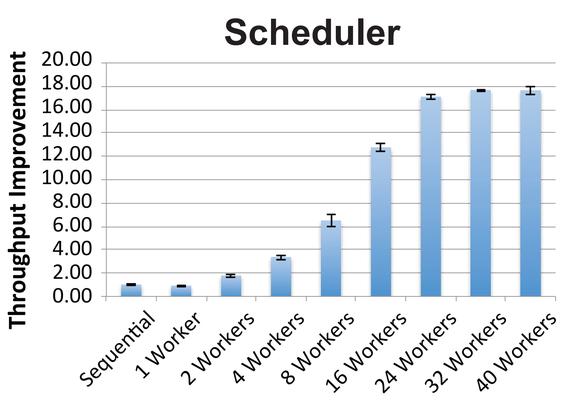
STATE SPACE SIZE ESTIMATION



- Prediction of state space size/progress with reasonable precision
- Exploration algorithm trades off precision for space complexity

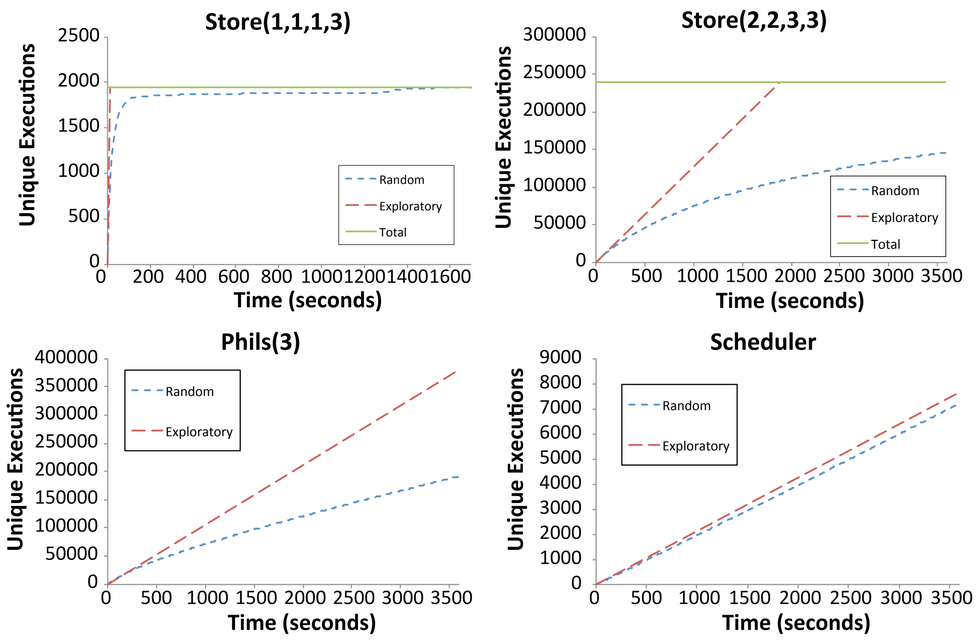
CONCURRENT STATE SPACE EXPLORATION

- A master process distributes work to worker processes
- Asynchronous RPC, callback threads record results
- State space reduction realized at workers
- State space size estimation realized at the master
- Centralized design → scalability bottleneck



Intel Science & Technology Center for Cloud Computing

COMPARING EXPLORATORY & RANDOM TESTING



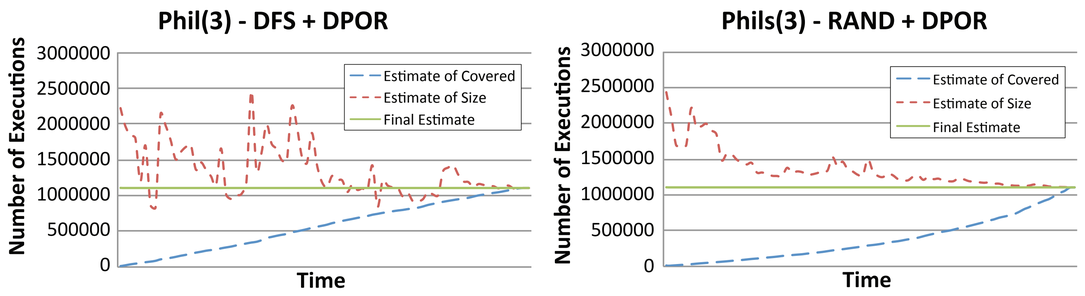
Simple test: basic exploratory testing outperforms random testing

- Question #1: How to determine if a test is simple?
- Question #2: How to scale exploratory testing to more complex tests?

DYNAMIC PARTIAL ORDER REDUCTION

- State space reduction technique by Flanagan and Godefroid

Test	# Executions	Time
Store(1,1,1,3)	289 (0.00%)	2.28s (0.01%)
Store(1,1,2,3)	765 (0.00%)	6.70s (0.89%)
Store(1,2,2,3)	1,235 (0.00%)	10.50s (0.19%)
Store(2,2,2,3)	12,216 (0.00%)	97.46s (0.62%)
Store(2,2,3,3)	25,887 (0.00%)	216.85s (1.00%)
Phils(2)	44 (0.00%)	0.22s (0.00%)
Phils(3)	67,192 (0.00%)	814.34s (8.59%)
PingPong(10)	1,024 (0.00%)	11.68s (0.05%)
PingPong(15)	32,768 (0.00%)	622.86s (4.31%)
Scheduler*	5,227 (0.32%)	3,600.00s (0.00%)



- Realized reduction ranges between 50% and 98%
- Compatible with state space size / progress estimation

FUTURE WORK & CONCLUSIONS

- Scheduler test is estimated to have 10^{17} possible executions
- Each test takes 0.5 second → 10^{9} years on a single computer

Scaling exploratory testing to new heights.

1) Avoid serialization of concurrent events
2) Improve DPOR / Adapt new reductions (e.g. DIR [Yang '11])
3) De-centralized concurrent state space exploration

State space reduction and parallel processing is the key to unlocking the potential of exploratory testing.

Carnegie Mellon University · Georgia Tech · intel · Princeton University · UC Berkeley