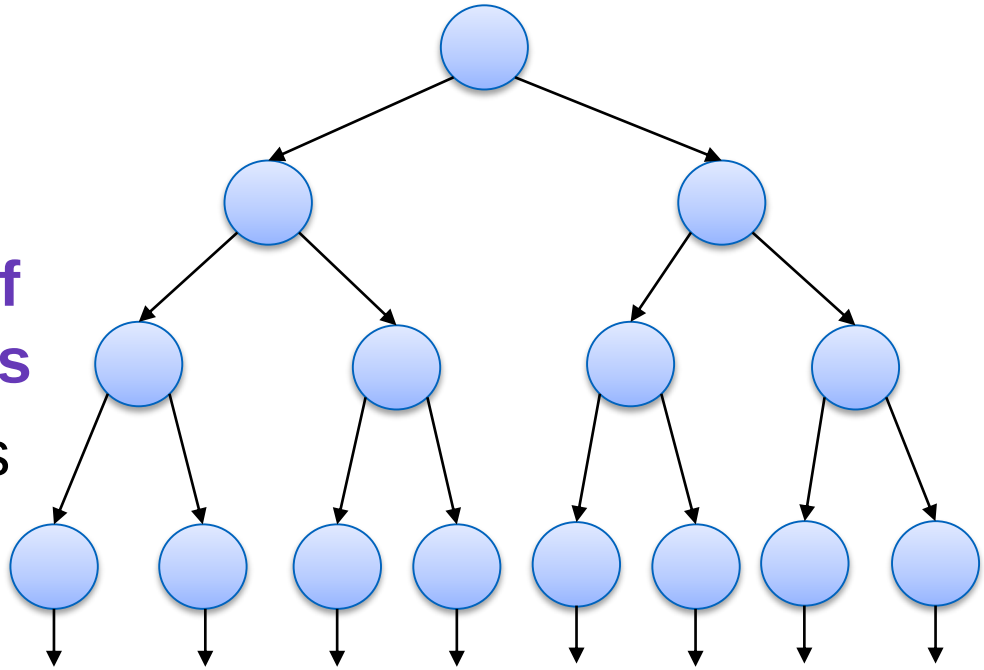
Aalto University
School of Science

LCT: A Parallel Distributed Testing Tool for Multithreaded Java Programs

Kari Kähkönen, Olli Saarikivi, Keijo Heljanko

Motivation

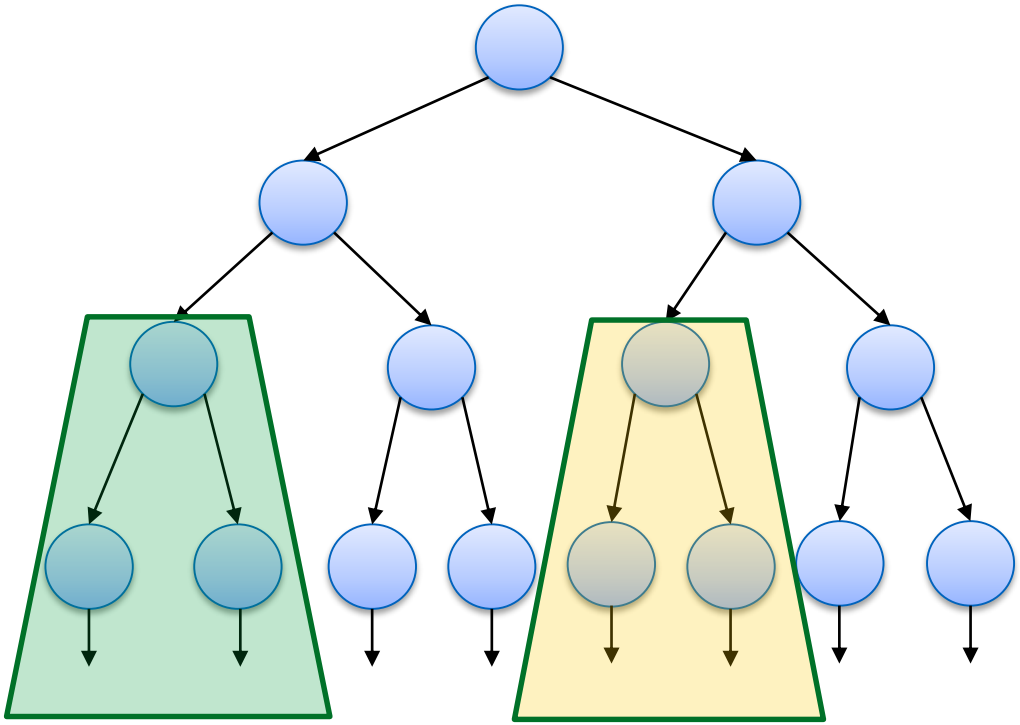
- Main problem with systematic testing: **explosion in number of paths and interleavings**
- Even state of the art has scalability problems



Execution tree

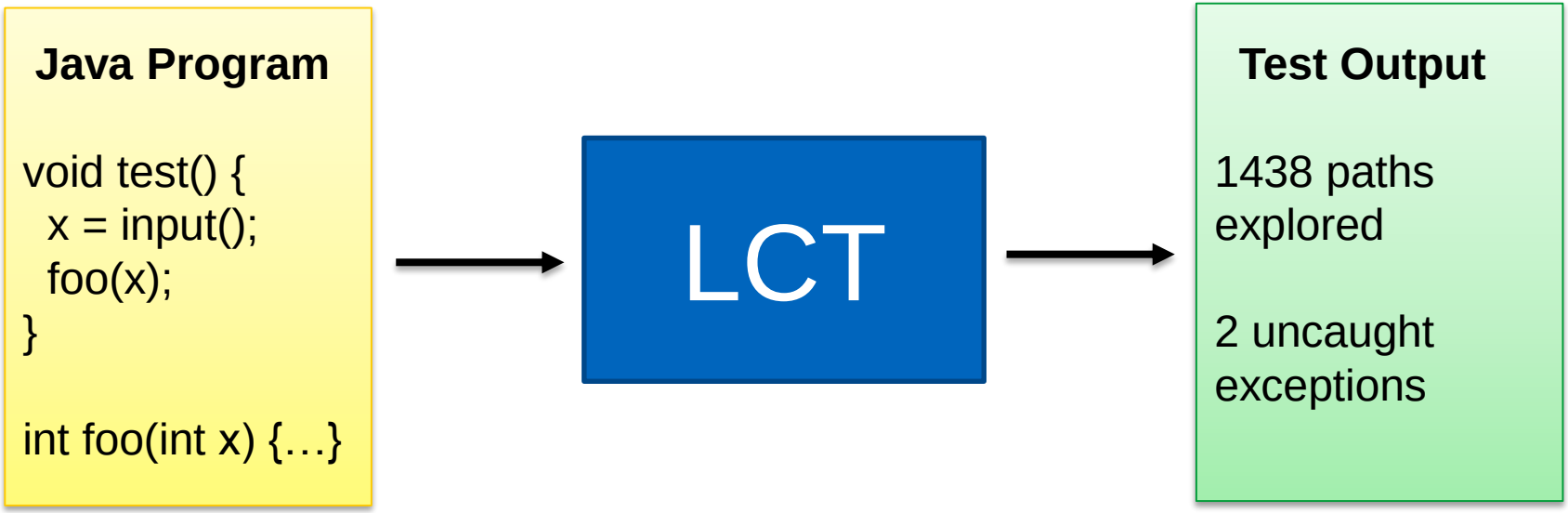
Alleviating the Path Explosion

- Testing different execution paths is mainly independent
- Great potential for parallelization

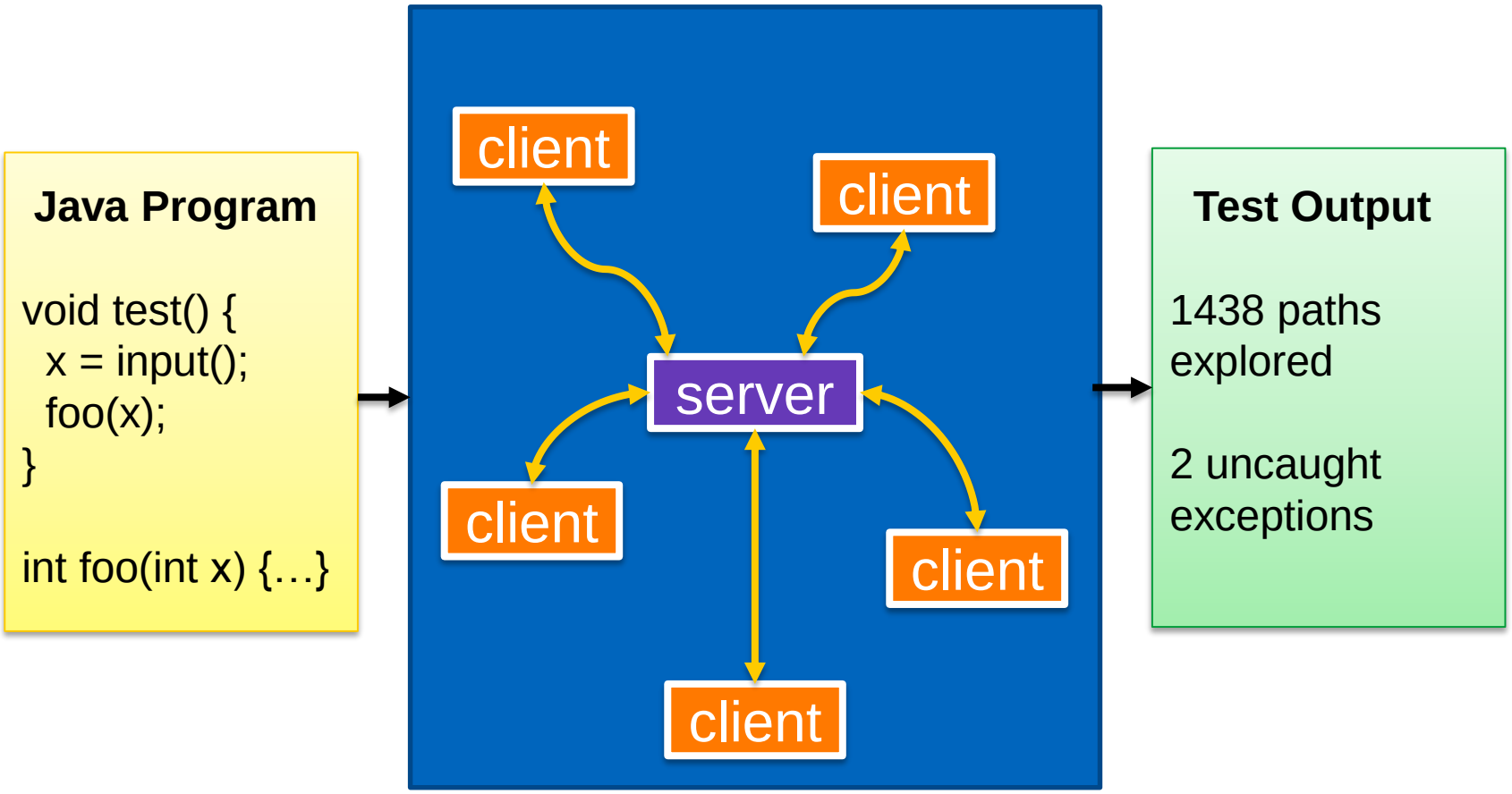


Execution tree

LCT – Lime Concolic Tester



LCT – Lime Concolic Tester



How Does LCT Work?

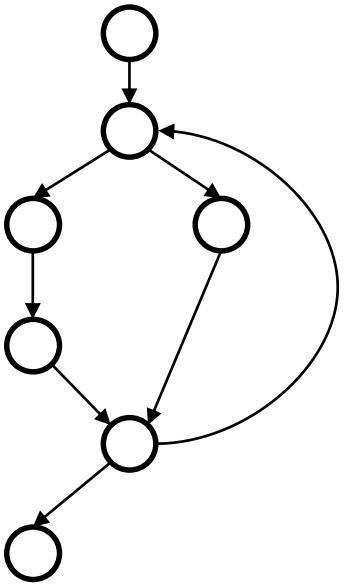
- Input value generation: **concolic testing**
- Eliminating irrelevant interleavings**: dynamic partial order reduction**
- Work distribution: **client-server architecture**

Concolic Testing

- Concolic testing aims to explore different execution paths of the program under test

```
x = input
x = x + 5

if (x > 10) {
  ...
}
...
```



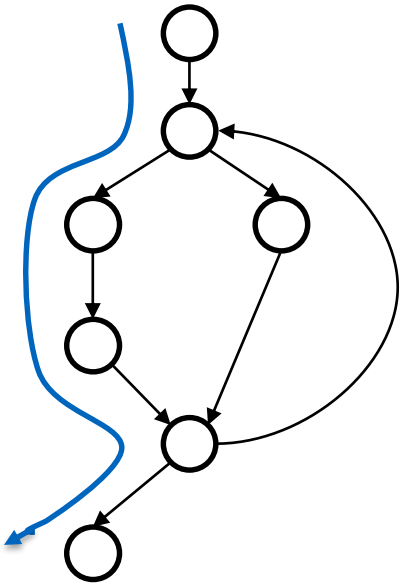
Control flow graph

Concolic Testing

- Concolic typically starts with a random execution

```
x = input
x = x + 5

if (x > 10) {
  ...
}
...
```



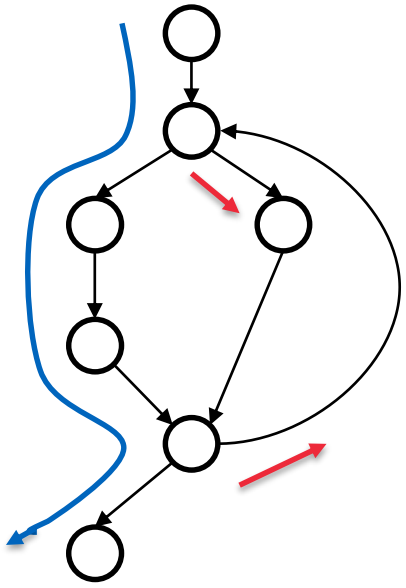
Control flow graph

Concolic Testing

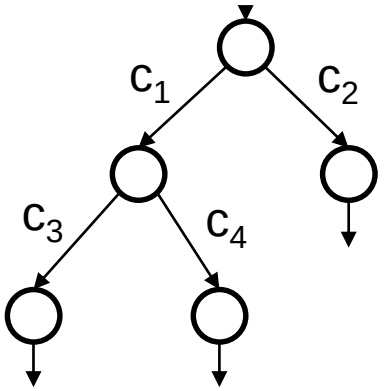
- Symbolic execution generates constraints that can be solved to obtain new test inputs for unexplored paths

```
x = input
x = x + 5

if (x > 10) {
  ...
}
...
```



Control flow graph

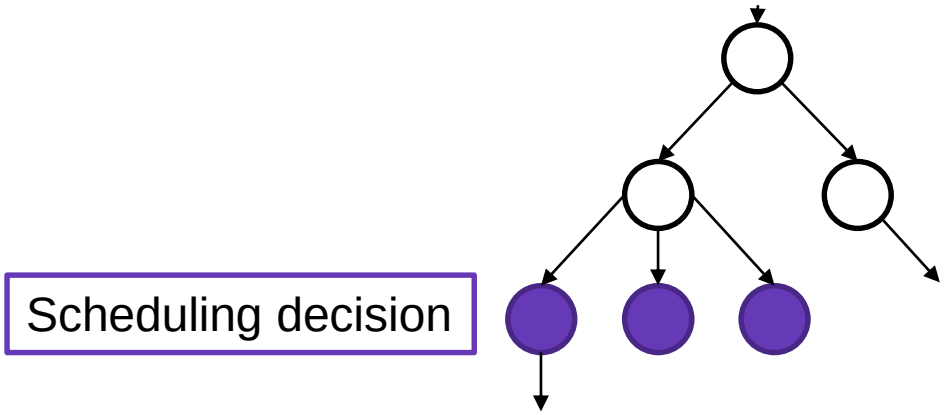


$c_1 = input_1 + 5 > 10$

$c_2 = input_1 + 5 \leq 10$

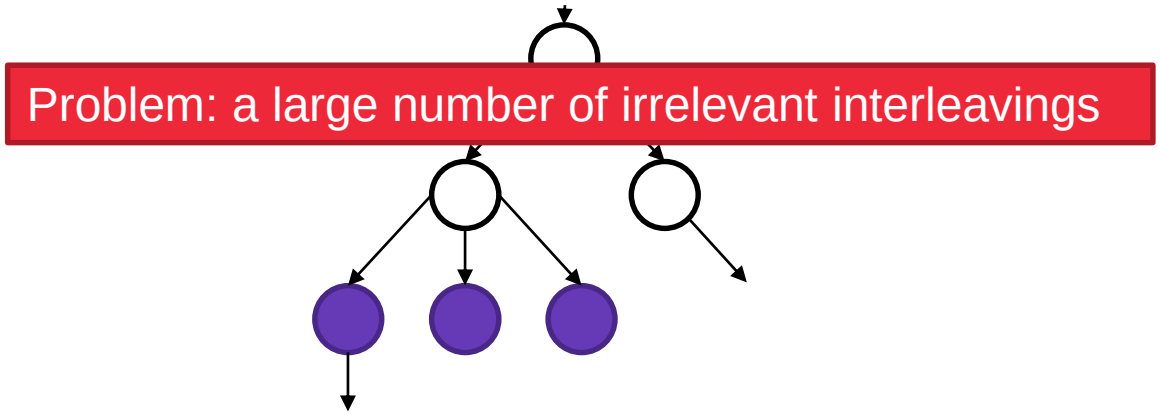
What about Multithreaded Programs?

- Execute threads one by one until a global operation (e.g., access shared variable) is reached
- Branch the execution tree for each enabled operation



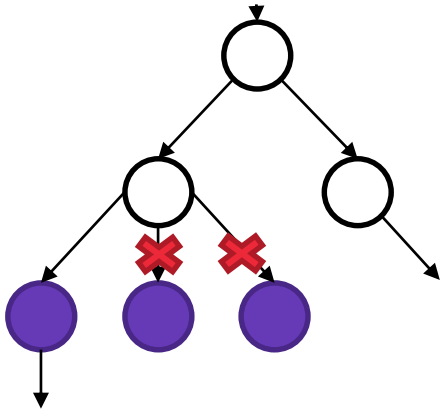
What about Multithreaded Programs?

- Execute threads one by one until a global operation (e.g., access shared variable) is reached
- Branch the execution tree for each enabled operation

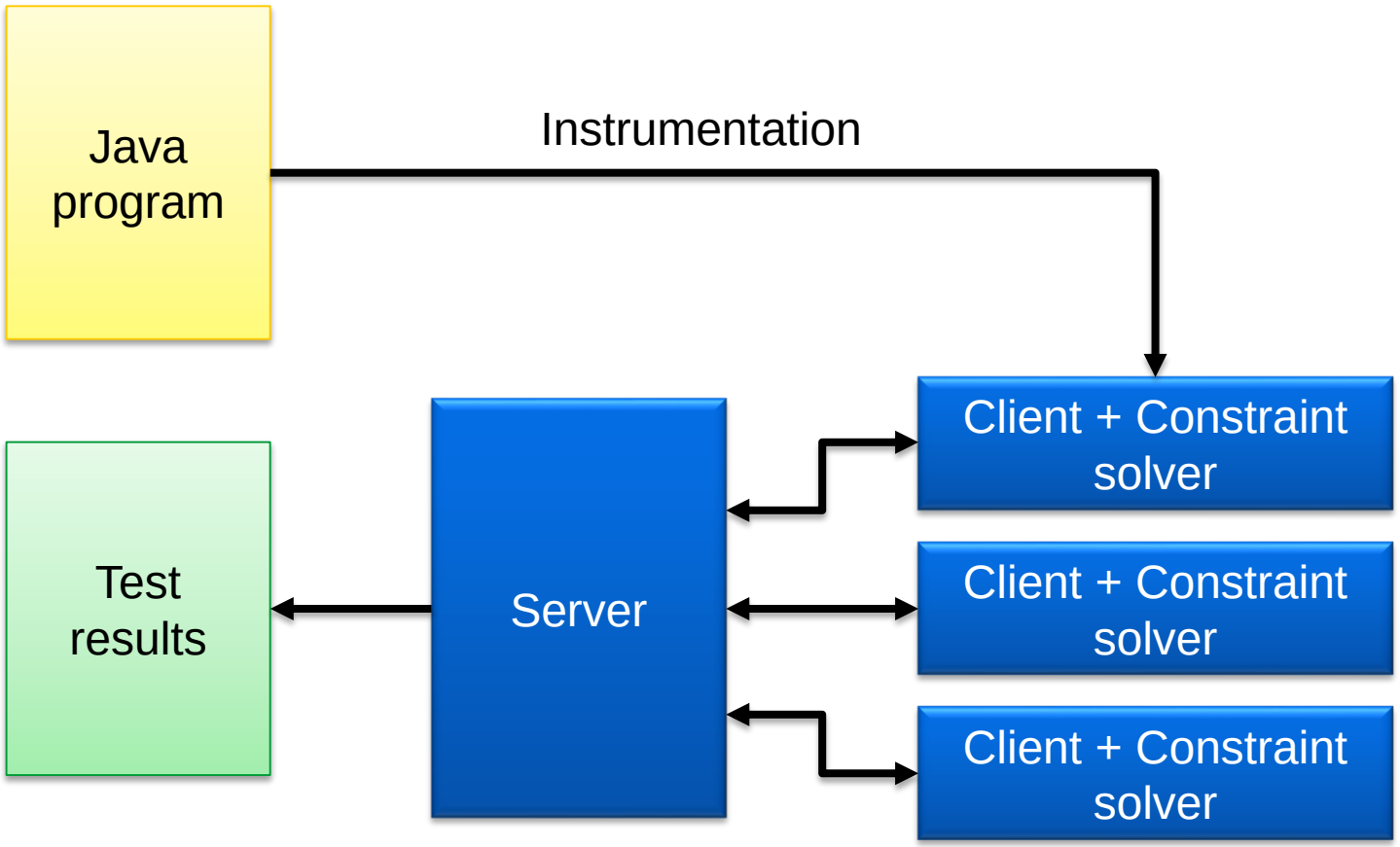


Dynamic Partial-Order Reduction (DPOR)

- Ignore provably irrelevant parts of the symbolic execution tree
- LCT uses the variant of DPOR that offers most reduction and works in a client server setting [ACSD 2012]



Distributing the Testing Process



Experiments

	1 client		2, 5, 10 and 20 clients			
program	**paths**	**time**	**Speedups**			
Indexer	671	285s	1.89	4.68	8.94	16.97
File System	138	47s	1.92	4.55	8.88	14.91
Parallel Pi	1252	250s	1.95	4.73	9.14	18.06
Synthetic 1	1020	176s	1.99	4.91	9.74	18.13
Synthetic 2	4496	783s	2.00	4.86	9.61	18.17

Can DPOR keep a large number of clients busy?
(Yes, it can)

Conclusions

- LCT can automatically test multithreaded Java programs
- Testing can be efficiently distributed to multiple workers
- Scales well at least up to 20 clients
- LCT is open source

http://www.tcs.hut.fi/Software/lime