Cloud9 Parallel Symbolic Execution for Automated **Real-World Software Testing**

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE
http://dslab.epfl.ch

Stefan Bucur
stefan.bucur@epfl.ch

Vlad Ureche
vlad.ureche@epfl.ch

Cristian Zamfir
cristian.zamfir@epfl.ch

George Candea
george.candea@epfl.ch

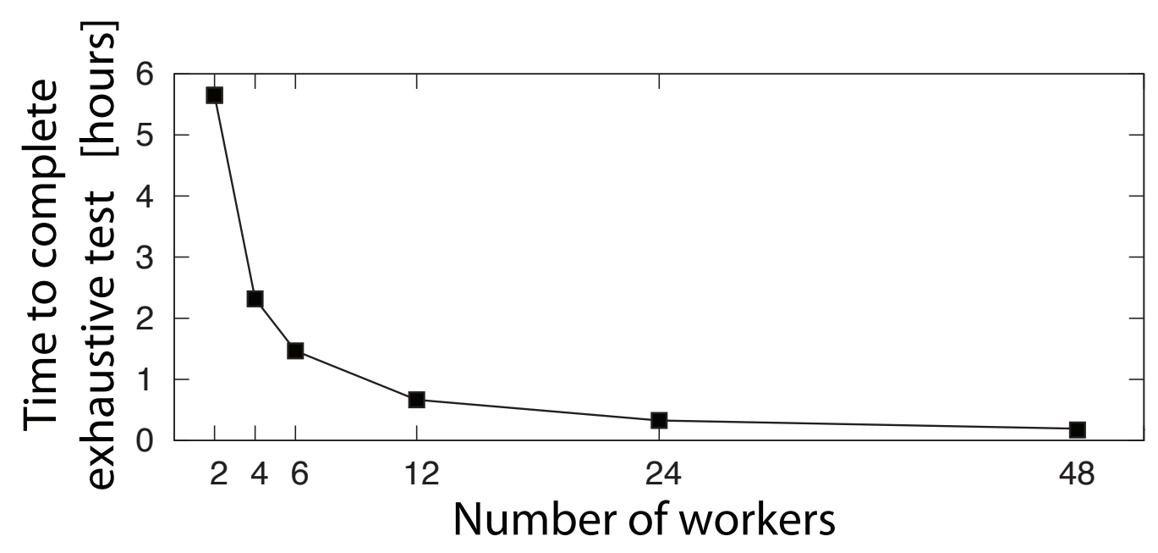
Real-World Automated Testing

We used Cloud9 to test software ranging from system utilities to large networked and distributed systems

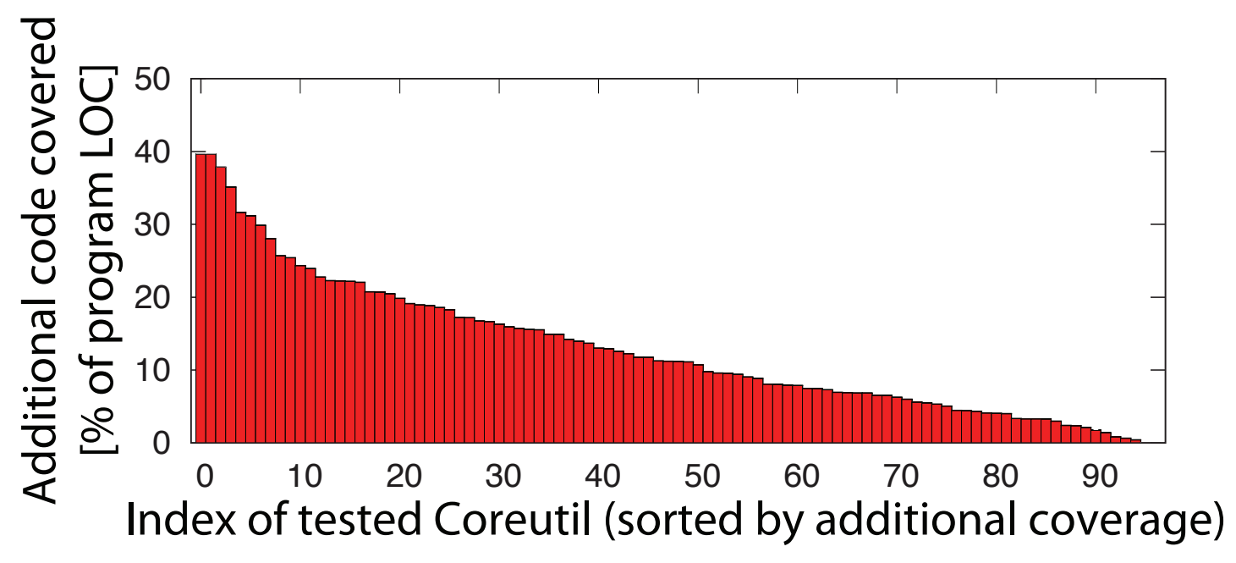
LIGHTTPD fly light. Apache Memcached GNU Coreutils python cURL

Scalable Cluster-Based Testing

- **Parallel symbolic execution** on large clusters of commodity hardware
- Suitable for running on public and private **cloud infrastructures**, such as Amazon EC2 or Eucalyptus



Time to exhaustively test **memcached** with symbolic packets



Cloud9 code coverage improvements on the **96 Coreutils** (1-worker vs. 12-worker)

Testing Platform API

- An API that developers can use to produce **symbolic test cases** and control the behavior of the OS environment:
  - **Inject symbolic data**
  - **Symbolic fault injection**
  - **Thread scheduling**
- Cloud9 can test programs with complex environment interactions:

✓**multithreading** ✓**IPC** ✓**networking**
✓**multiple processes** ✓**synchronization** ✓**filesystem**

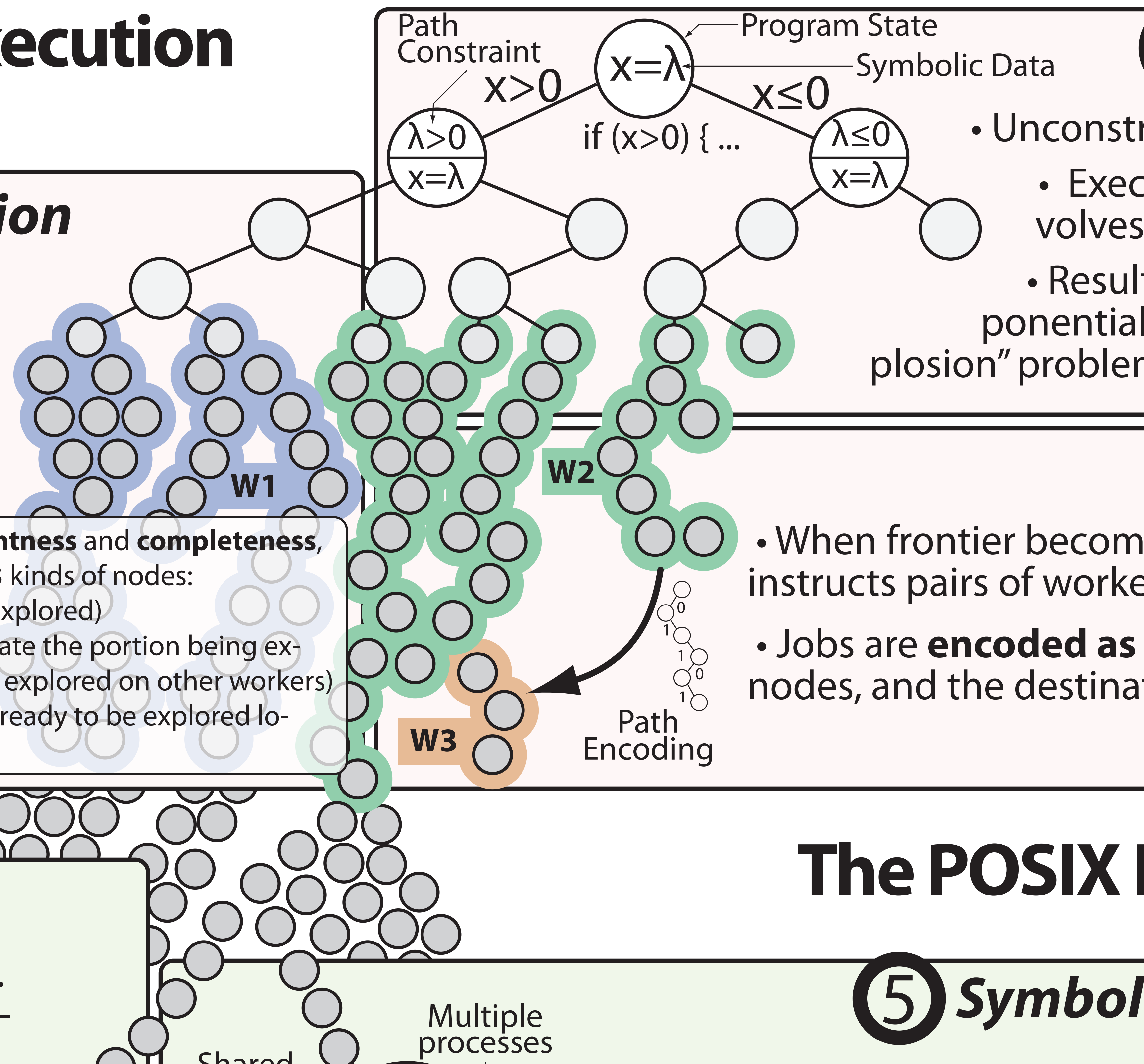
Example: Preparing a symbolic HTTP packet to test a header extension

```
char httpData[10];
make_symbolic(httpData);
strcat(req, “X-NewExtension: ”);
strcat(req, hData);

ioctl(ssock, SIO_PKT_FRAGMENT, RD);
ioctl(ssock, SIO_FAULT_INJ, RD | WR);
```

Under The Hood

Parallel Symbolic Execution

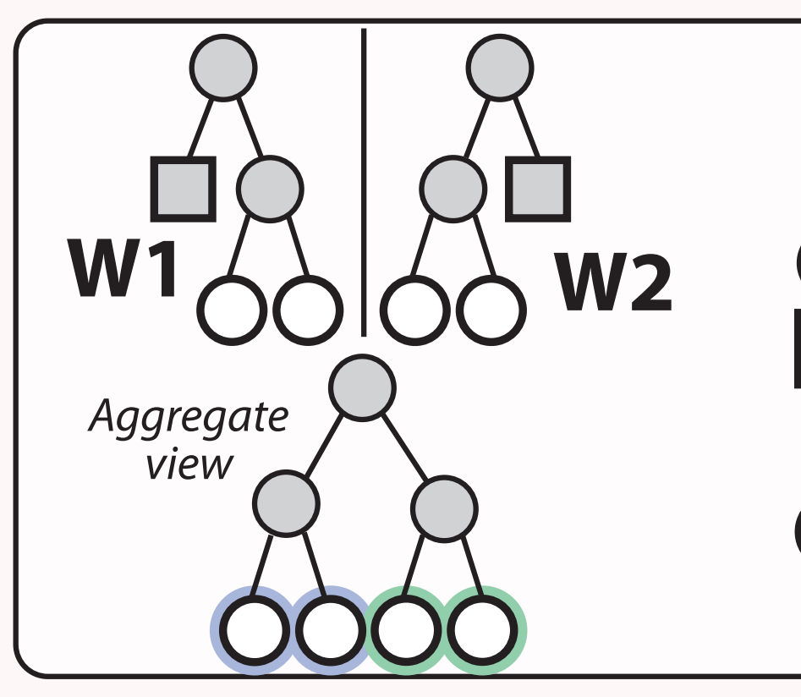


① *Symbolic Execution*

- Unconstrained **symbolic** data
  - Execution **forks** when a branch involves symbolic values
  - Resulting **execution tree** increases exponentially with program size — “path explosion” problem

② *Parallel Tree Exploration*

- Parallel symbolic execution engine runs on **commodity clusters**
- **Worker nodes** run independent symbolic engines and are coordinated by a **load balancer**



To ensure exploration **disjointness** and **completeness**, each worker’s local tree has 3 kinds of nodes:
● **internal nodes** (already explored)
□ **fence nodes** that demarcate the portion being explored (correspond to nodes explored on other workers)
○ **candidate nodes** (nodes ready to be explored locally)

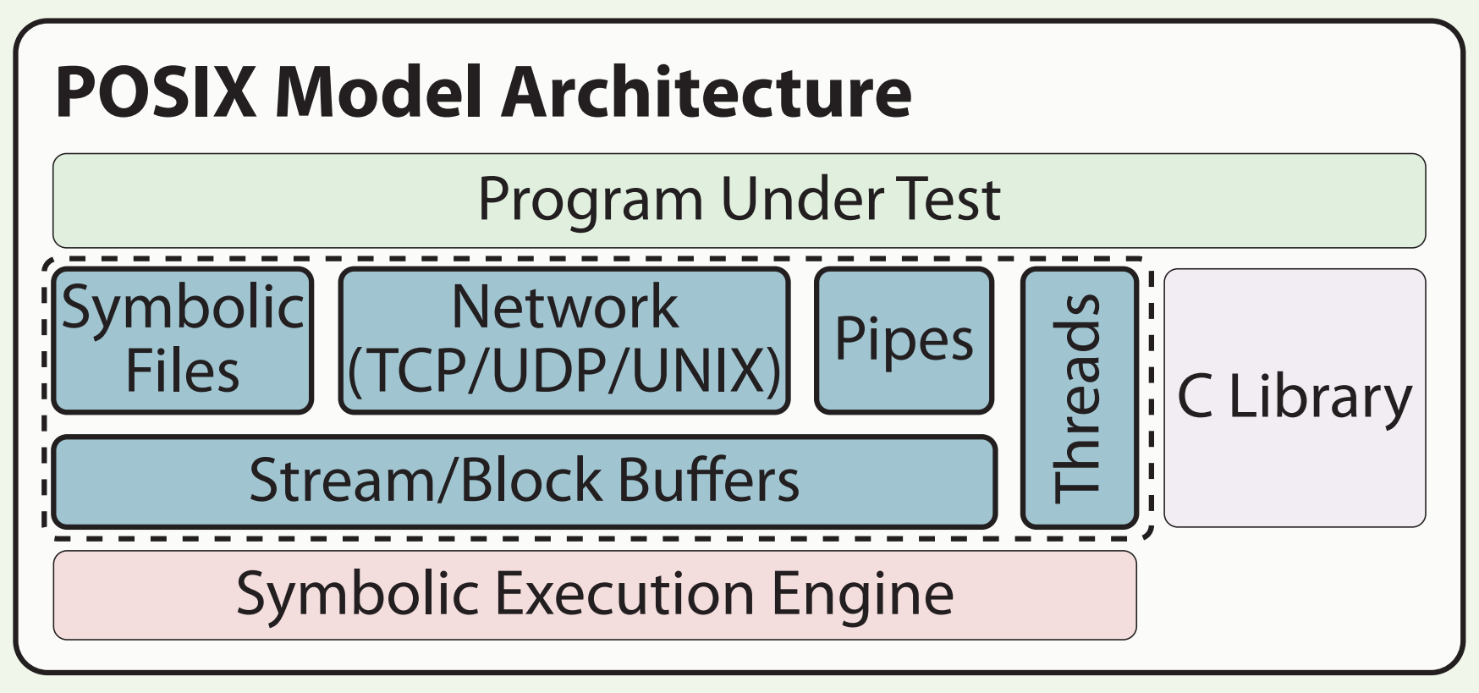
③ *Work Transfer*

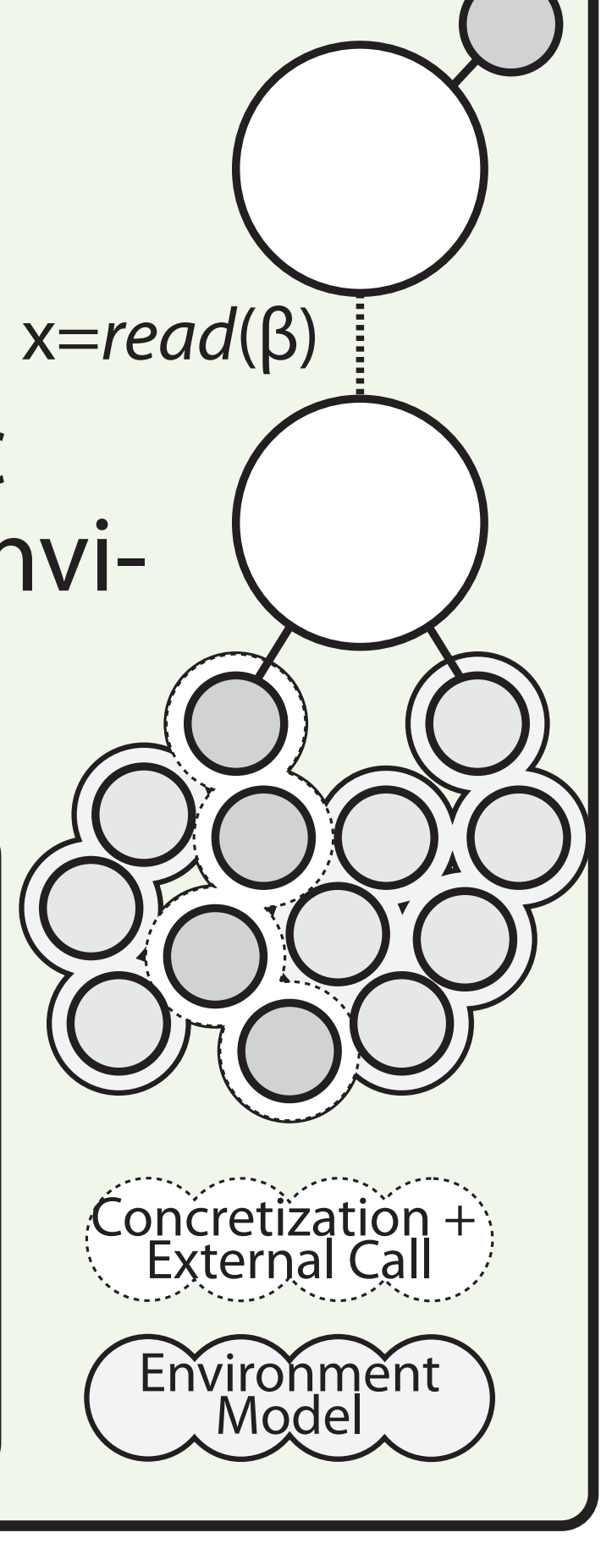
- When frontier becomes unbalanced, the load balancer instructs pairs of workers to exchange jobs
- Jobs are **encoded as paths** from the tree root to the nodes, and the destination node **“replays”** that path

The POSIX Environment Model

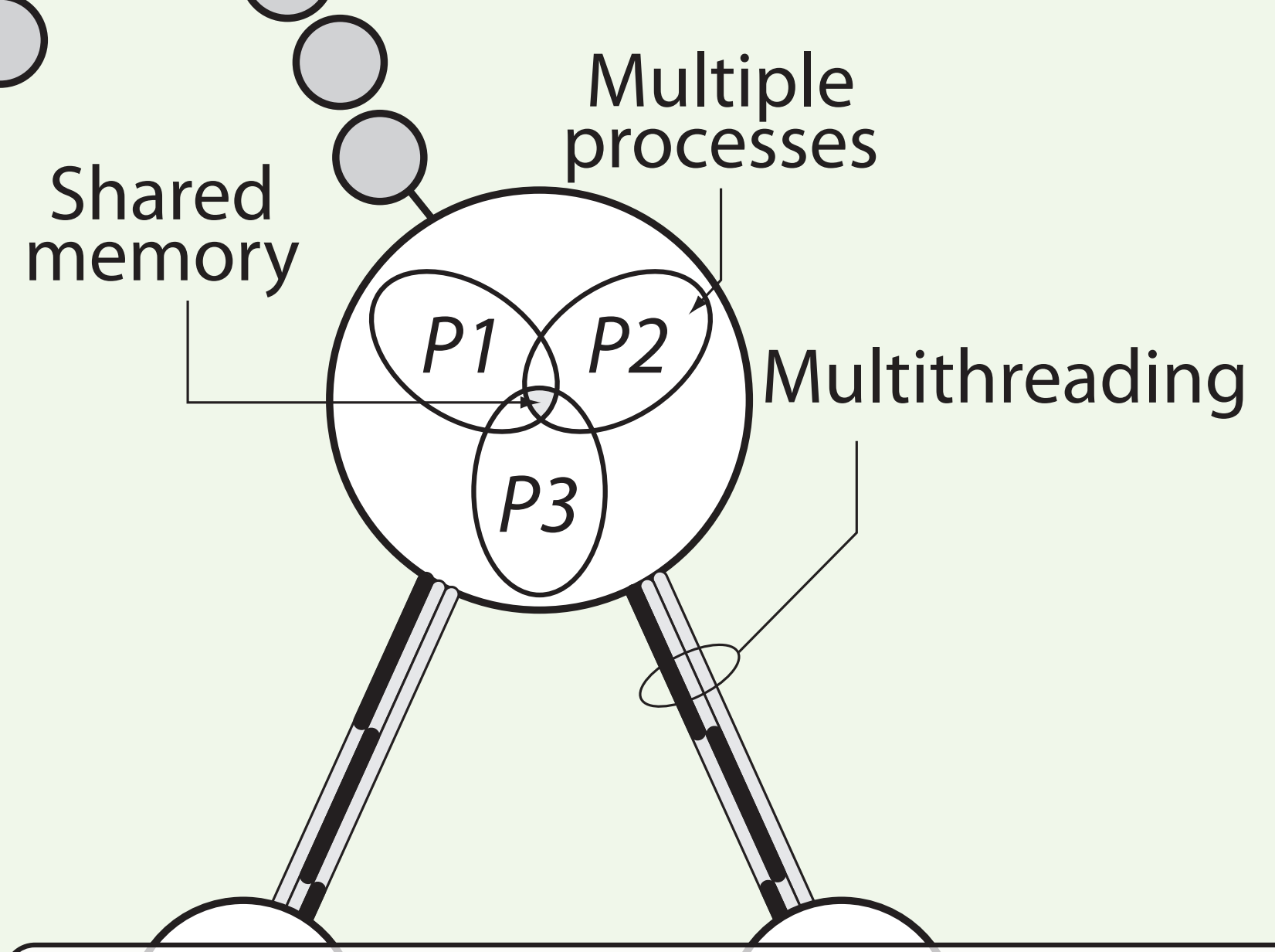
④ *Full POSIX Model*

- Outside the **symbolic domain** (e.g. the program under test), the environment is complex
- One may “concretize” calls and lose completeness
- A model extends the symbolic domain, while simplifying the environment behavior





⑤ *Symbolic Engine Modifications*



Symbolic System Calls
Replaced the standard OS system calls with a simplified set of **calls into the symbolic execution engine**:
- create/destroy threads
- fork/terminate processes
- share memroy across processes
- sleep/wake-up on waiting queues

Multithreading and Scheduling

- Cooperative scheduler simplifies model implementation
- Deterministic (round-robin) or symbolic scheduling

Address Spaces

- Use copy-on-write (CoW) to reuse memory between processess, as well as across states
- Put address spaces in CoW domains, to permit memory sharing

http://cloud9.epfl.ch