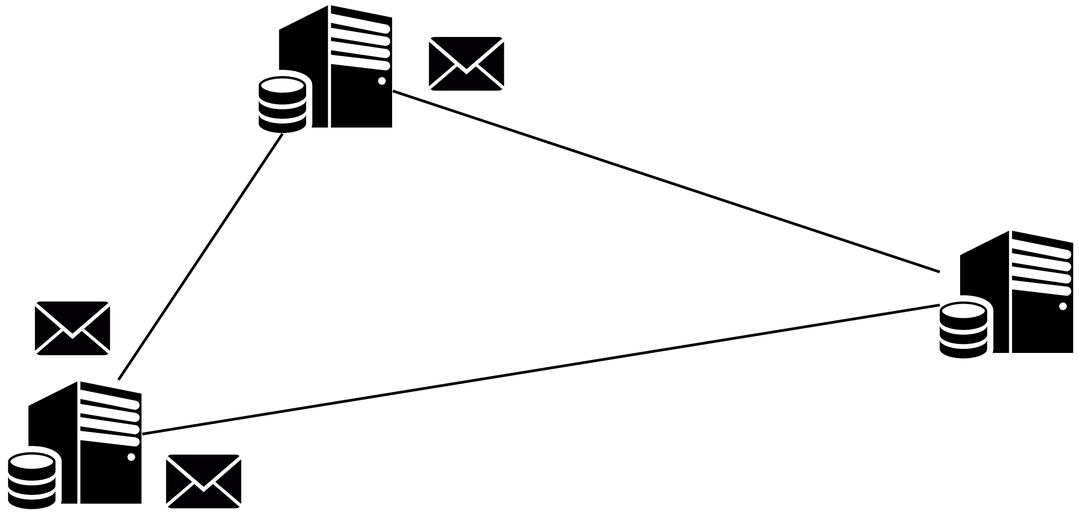
Randomized Testing of Distributed Systems with Probabilistic Guarantees

Burcu Kulahcioglu Ozkan[1], Rupak Majumdar[1], **Filip Niksic**[2],
Mitra Tabaei Befrouei[3], Georg Weissenbacher[3]

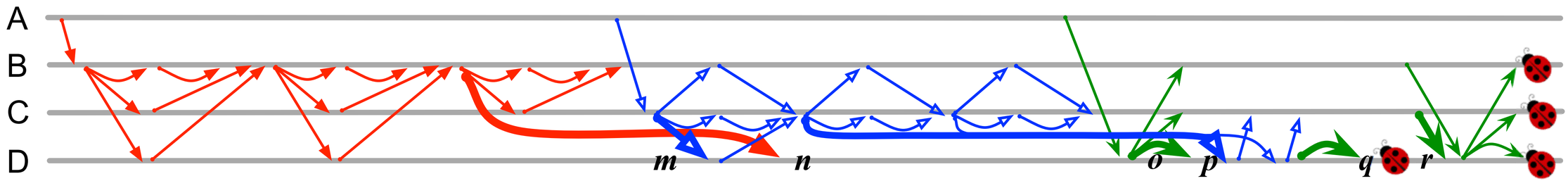
[1] Max Planck Institute for Software Systems (MPI-SWS)
[2] University of Pennsylvania
[3] Vienna University of Technology

Distributed systems are prone to bugs



- Messages are exchanged concurrently
- Nodes and the network can fail, and the system must tolerate failures

Many bugs have "small depth"



Paxos bug in Cassandra [Leesatapornwongsa et al. ASPLOS '16]

- Bug of depth **d** — depends on the relative ordering of **d** specific events
- Approach: Fix a small **d**, and search for schedules that hit bugs of depth **d**

Hitting families of schedules

For a partially ordered set of **n** events, and a fixed integer **d**:

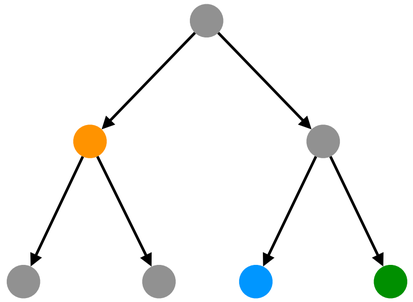
- A **schedule** is a total order (linearization) of the events
- A schedule **α hits** a **d**-tuple **$(e_1, \ldots, e_d)$** if it orders the events as $e_1 <_\alpha \ldots <_\alpha e_d$
- A set of schedules $\mathcal{F}$ is a **d-hitting family** if it hits every **d**-tuple of events

Research question:

How to construct "small" hitting families of schedules?

Hitting families can indeed be small

In our **CAV '16** paper we study partial orders structured as trees:



For trees of height **h**:

- **d = 3**: explicit construction of hitting families of size **4h** (optimal)
- **d > 3**: explicit construction of hitting families of size **$O(\exp(d)\, d!\, h^{d-1})$**

In this work (OOPSLA '18)

Partial orders:

- have **arbitrary structure**
- are **revealed online (during program execution)**

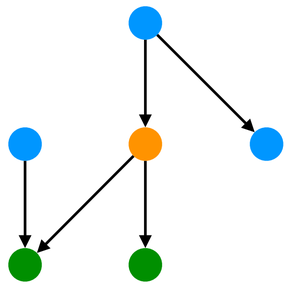
Our result for partial orders of size at most **n** and width at most **w**:

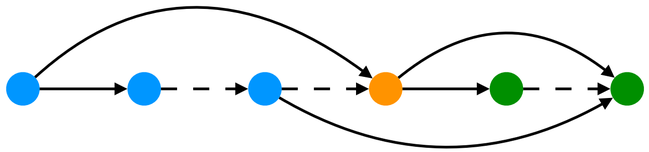
- Explicit construction of **d**-hitting families of size $\mathbf{w^2 n^{d-2}}$
- Randomized scheduler **PCTCP** that hits a bug of depth **d** with probability at least $\mathbf{1/(w^2 n^{d-2})}$
- Experimentation on P# programs, Cassandra, and Zookeeper

Arbitrary partial orders

For a partial order with **n** events:

- There is a **d**-hitting family of size $\mathbf{O(n^d)}$:
  For each **d**-tuple $\mathbf{(e_1, \ldots, e_d)}$, construct a schedule that hits $\mathbf{(e_1, \ldots, e_d)}$
- There is a **d**-hitting family of size $\mathbf{O(n^{d-1})}$:
  For each **(d-1)**-tuple $\mathbf{(e_2, \ldots, e_d)}$, construct a schedule that **strongly** hits $\mathbf{(e_2, \ldots, e_d)}$, i.e. every $\mathbf{e_i}$ is delayed as much as possible in the schedule

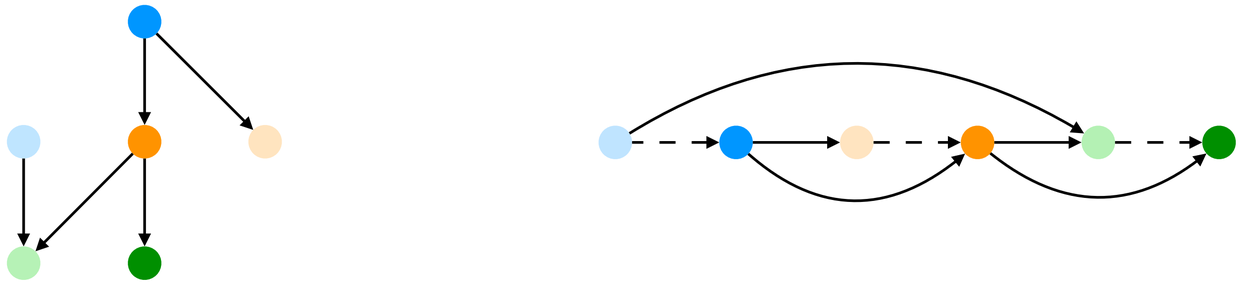




Partition into chains

Theorem (Dilworth). Every partial order of width **w** can be partitioned into **w** chains.

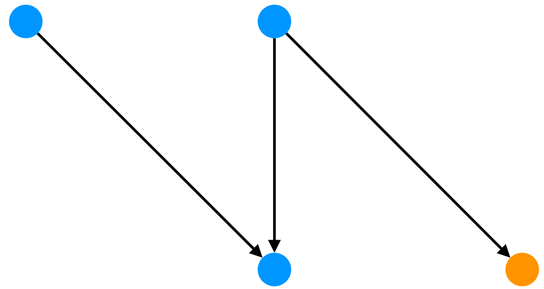
Idea: Strongly hit every event in a chain. [Felsner '97, Kloch '07]



This gives **d**-hitting families of size $\mathbf{wn^{d-2}}$.
A randomized version is the PCT scheduler of Burckhardt et al. [ASPLOS '10]

Online partition into chains

If the partial order of width **w** is revealed online (during execution), we can be forced to use $\mathbf{w^2}$ chains in the partition. [Felsner '97]



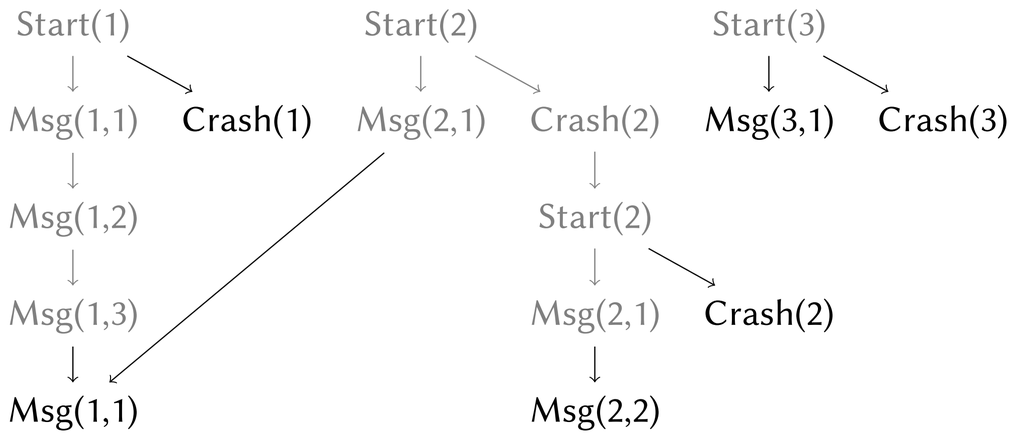
A chain partitioning algorithm by Agarwal and Garg [2007] uses at most $\mathbf{w^2}$ chains.

This gives **d**-hitting families of size $\mathbf{w^2\, n^{d-2}}$.
A randomized version is our PCTCP scheduler.

Experiments with Zookeeper

We implemented PCTCP in a tool called HitMC:

- controls messages during Zookeeper's leader election; can crash and restart nodes



We search for **inconsistencies** (not necessarily bugs) in leader election

We compare PCTCP with random walk

Experiments with Zookeeper

Results for 1,000 runs with different scheduling strategies, with 1 node crash and 1 reboot per run

Strategy	Avg Events	Max Events	Avg Chains	Max Chains	Inconsistent Runs
Random Walk	20.9	36	-	-	12
PCTCP (d=2, n=50)	20.6	31	6.4	10	39

Summary

For **arbitrary** partial orders that are revealed **online**:

- Construction of **d**-hitting families of size $\mathbf{w^2\ n^{d-2}}$
- Randomized scheduler PCTCP that finds depth-**d** bugs with probability $\mathbf{1/(w^2\ n^{d-2})}$

Check out our OOPSLA '18 paper for:

- Technical details and proofs
- Additional experiments with P# programs, Cassandra, and Zookeeper