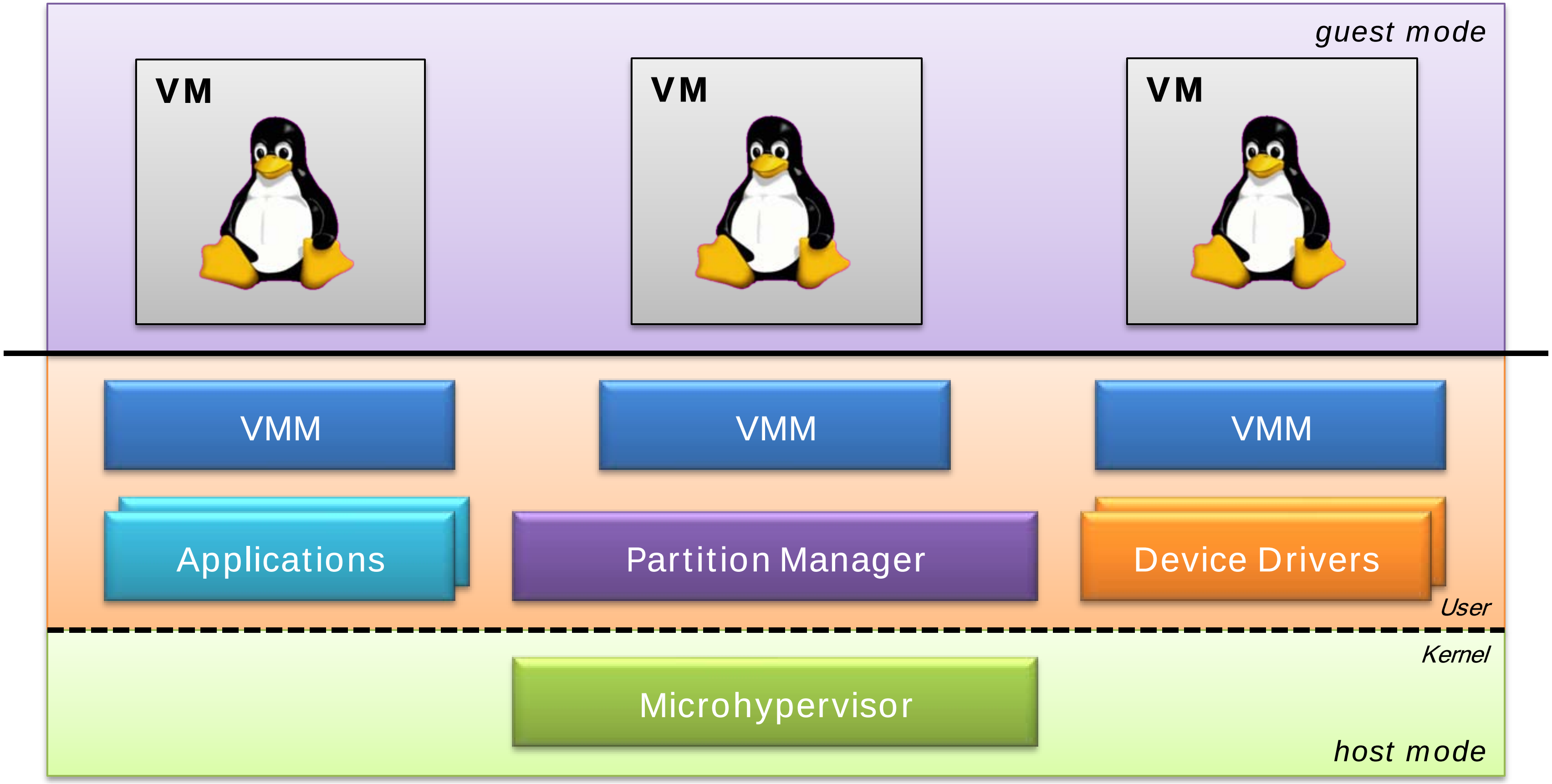
TECHNISCHE UNIVERSITÄT DRESDEN

Department of Computer Science – Institute of Systems Architecture – Operating Systems Group

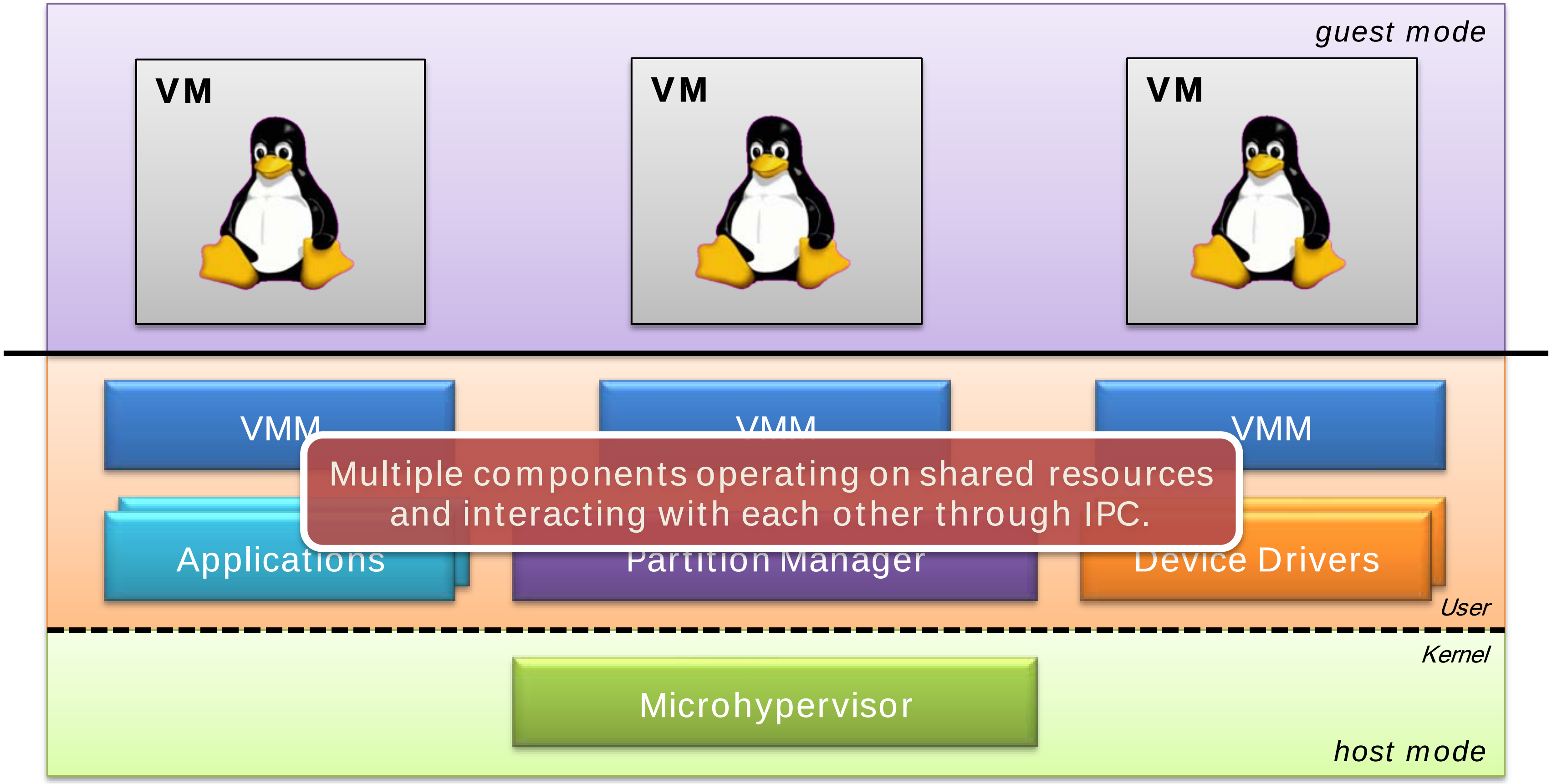
Timeslice Donation in Component-Based Systems

Udo Steinberg, Alexander Böttcher, Bernhard Kauer

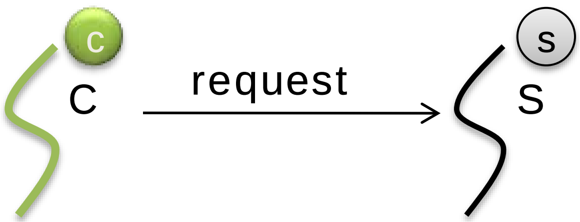
Component-Based System: NOVA



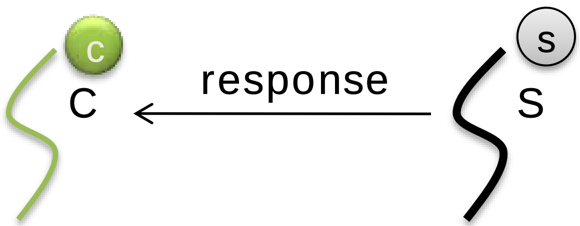
Component-Based System: NOVA



Synchronous IPC (Example: L4)

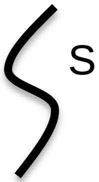


- During IPC, client donates its timeslice to the server.
  - Kernel switches from client C to server S without changing the current timeslice.



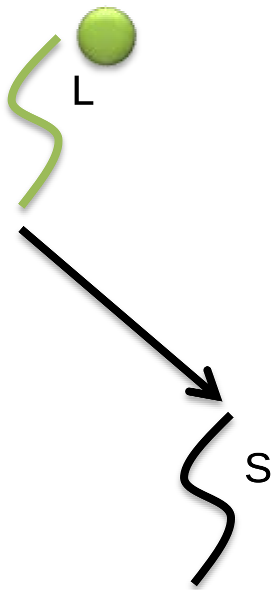
- Effect is priority inheritance, but only until S is preempted
  - If the kernel fails to recognize the dependency between C and S after preemption, S will consume its own timeslice s instead of c.

Problem #1: Classic Priority Inversion



Shared Resource

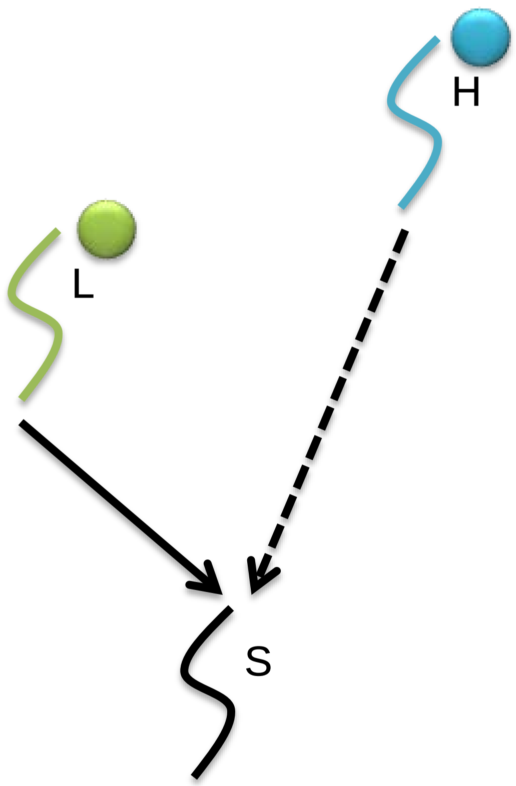
Problem #1: Classic Priority Inversion



Shared Resource

TECHNISCHE UNIVERSITÄT DRESDEN

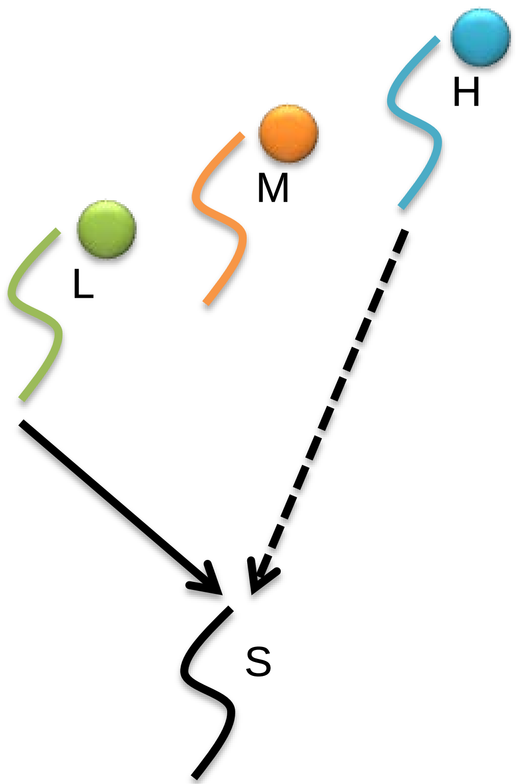
Problem #1: Classic Priority Inversion



Shared Resource

- High-priority thread H blocked by low-priority thread L holding S.
- Unbounded priority inversion if M prevents L from running and thus from releasing S.
- Our Solution: Priority Inheritance
    - Server inherits priority of its clients for the duration of their requests.
    - Kernel tracks dependencies.

Problem #1: Classic Priority Inversion

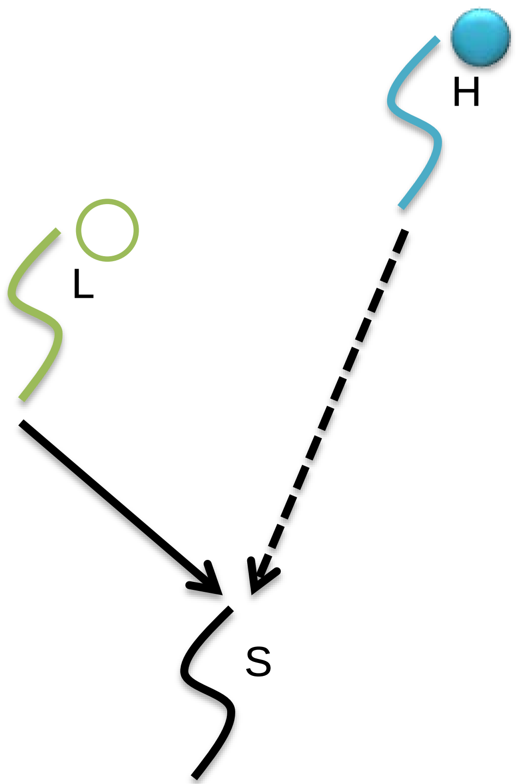


Shared Resource

- High-priority thread H blocked by low-priority thread L holding S.
- Unbounded priority inversion if M prevents L from running and thus from releasing S.
- Our Solution: Priority Inheritance
  - Server inherits priority of its clients for the duration of their requests.
  - Kernel tracks dependencies.

TECHNISCHE UNIVERSITÄT DRESDEN

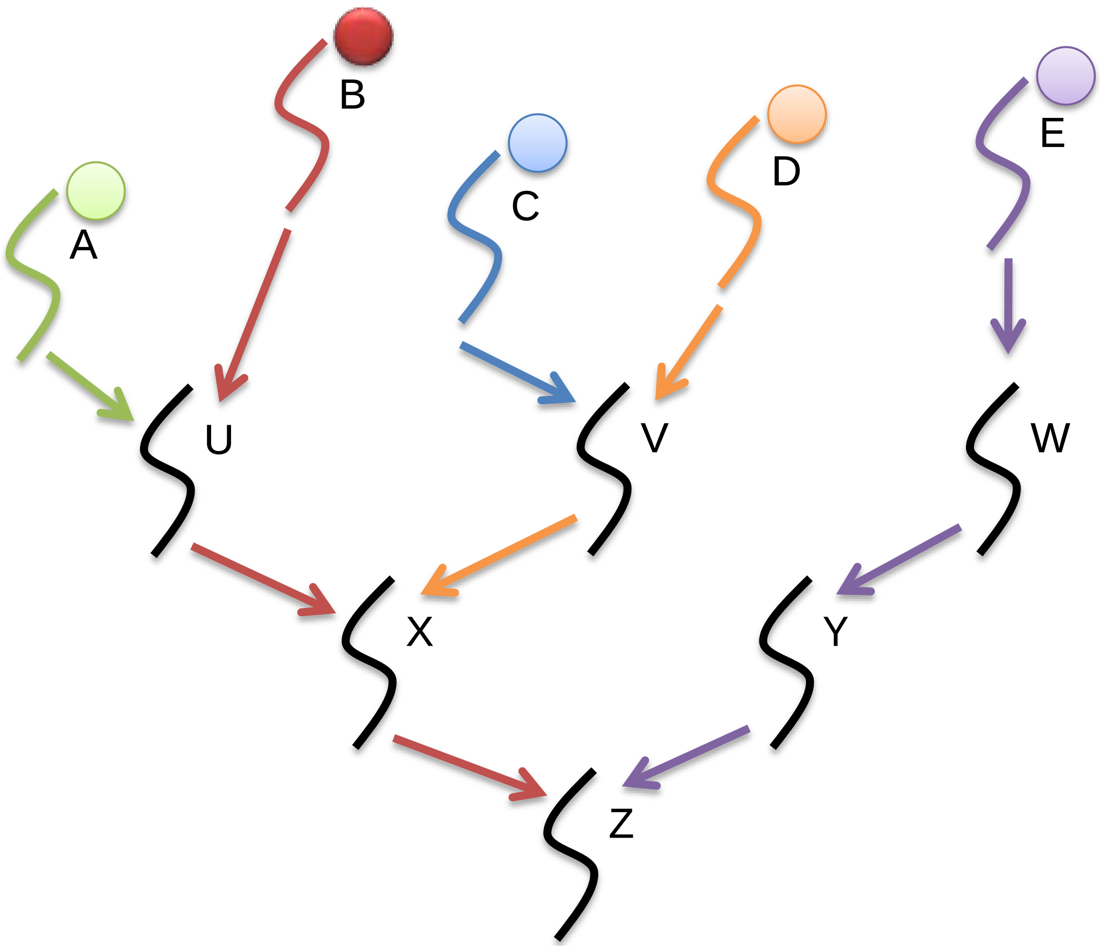
Problem #2: Time Quantum Depleted



Shared Resource

- Thread L exhausts its time quantum while holding S.
- Blocks other threads, e.g. H, from acquiring the resource until time quantum has been replenished.
- Priority boosting will not help.
- Our Solution: Bandwidth Inheritance
  - Clients donate not only their priority, but the *entire* timeslice (priority, time quantum, ...)

Old Algorithm for Dependency Tracking*



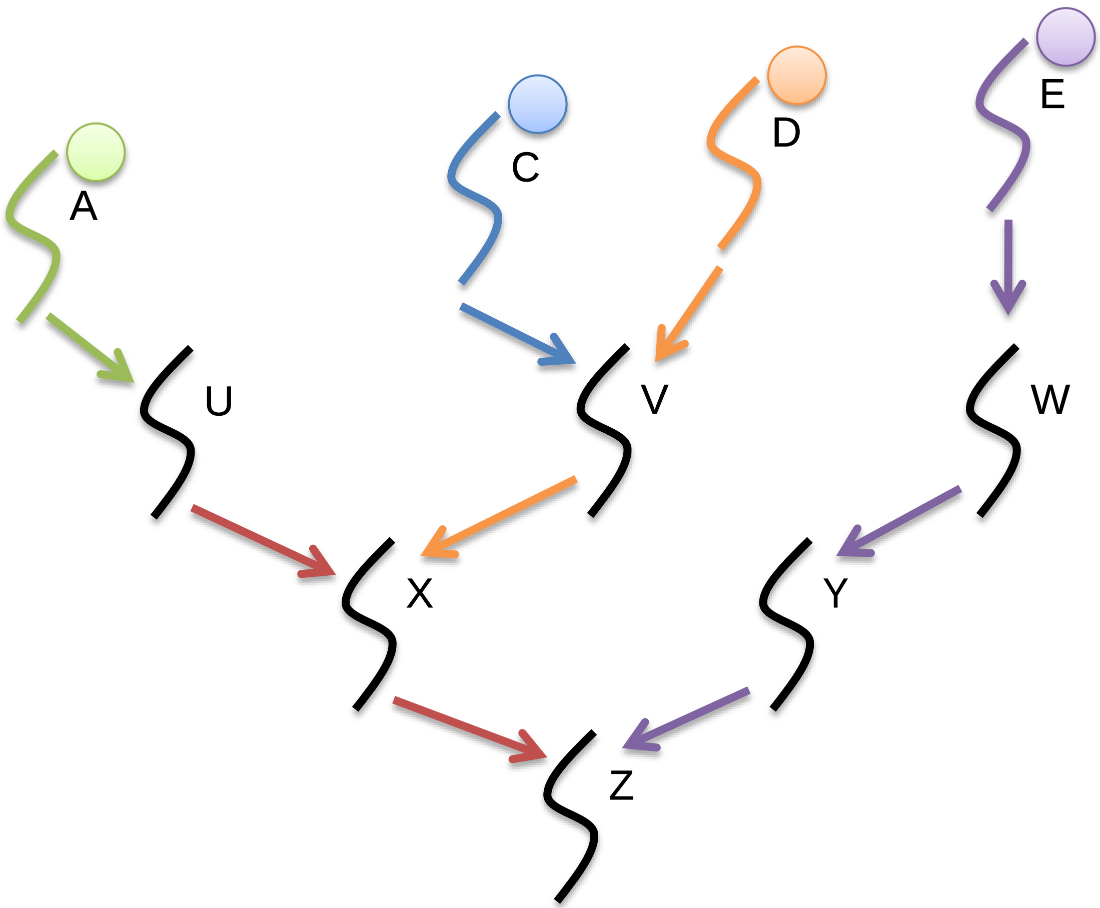
Update required when:

- New node added or new link established
- Node removed or existing link broken
- Priority of a node changes

Inconsistent state until dependency tree has been fully updated.

* Presented at ECRTS'05

Old Algorithm for Dependency Tracking*



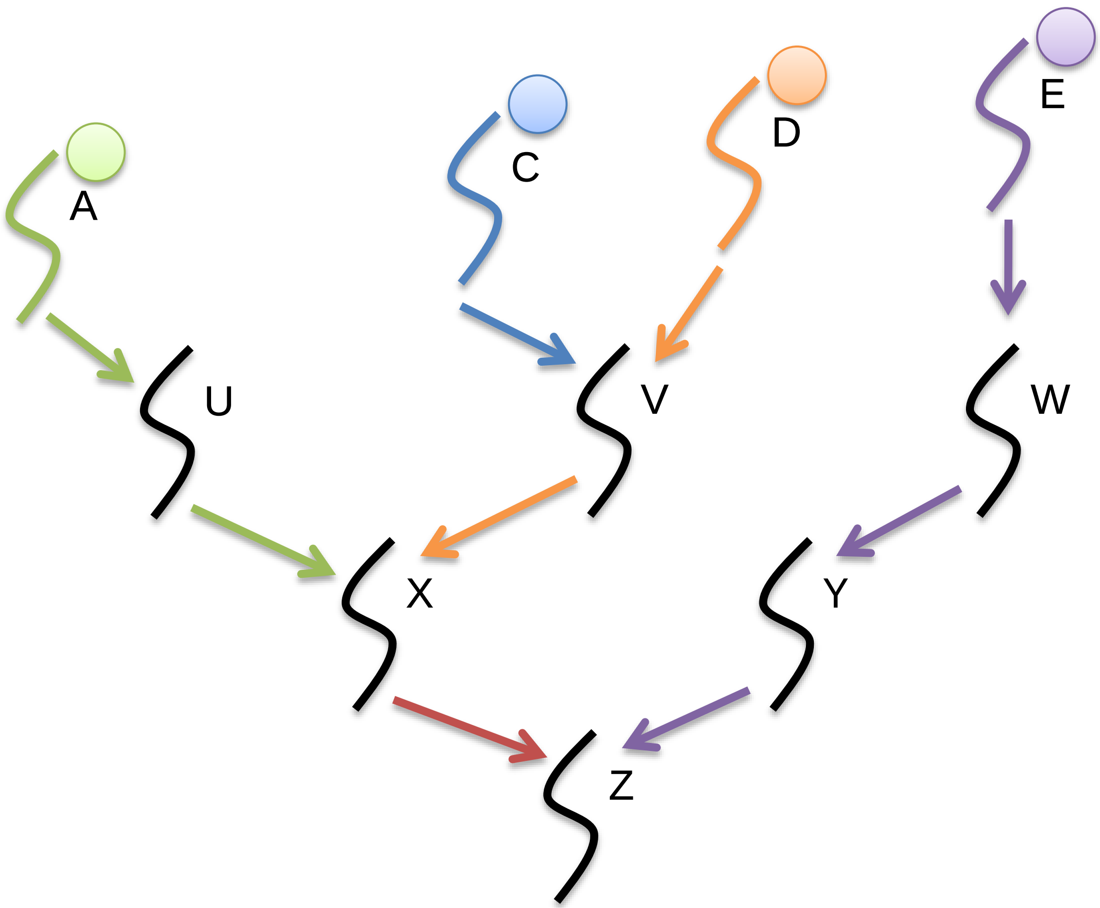
Update required when:

- New node added or new link established
- Node removed or existing link broken
- Priority of a node changes

Inconsistent state until dependency tree has been fully updated.

* Presented at ECRTS'05

Old Algorithm for Dependency Tracking*



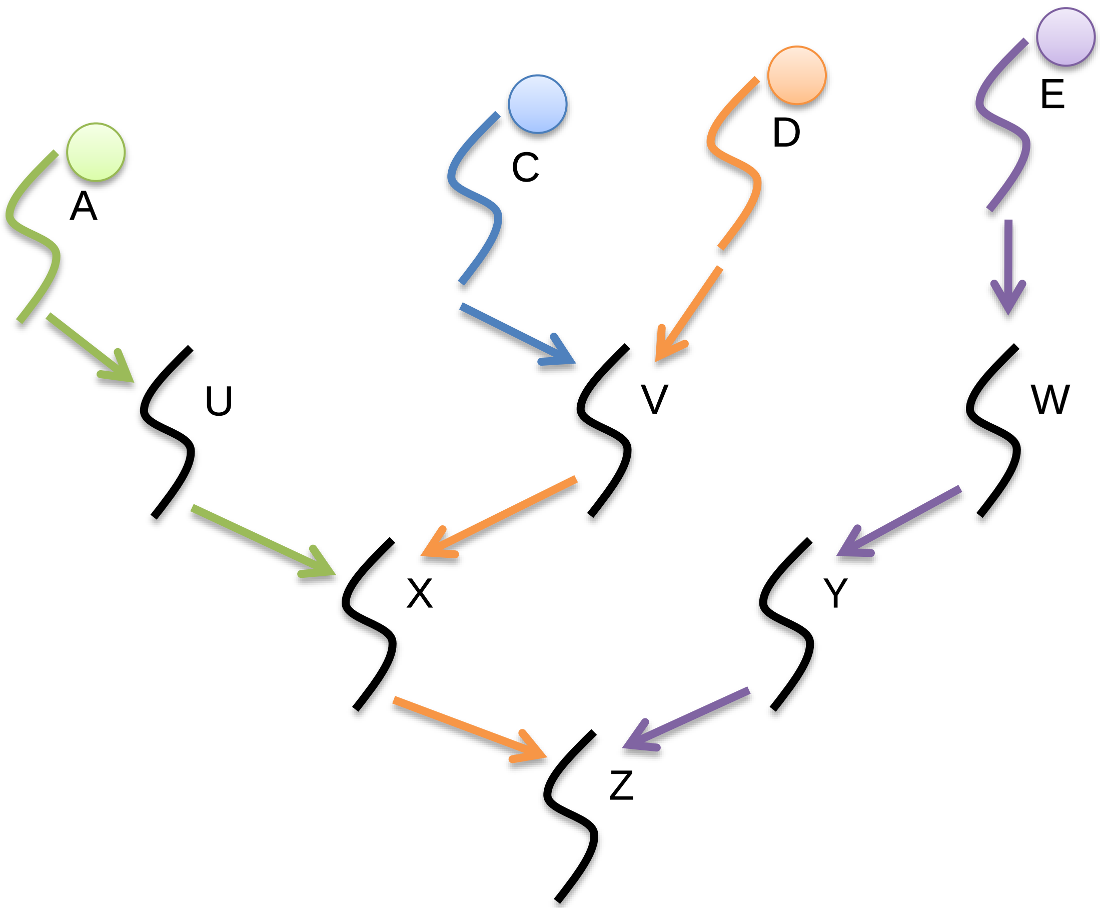
Update required when:

- New node added or new link established
- Node removed or existing link broken
- Priority of a node changes

Inconsistent state until dependency tree has been fully updated.

* Presented at ECRTS‘05

Old Algorithm for Dependency Tracking*



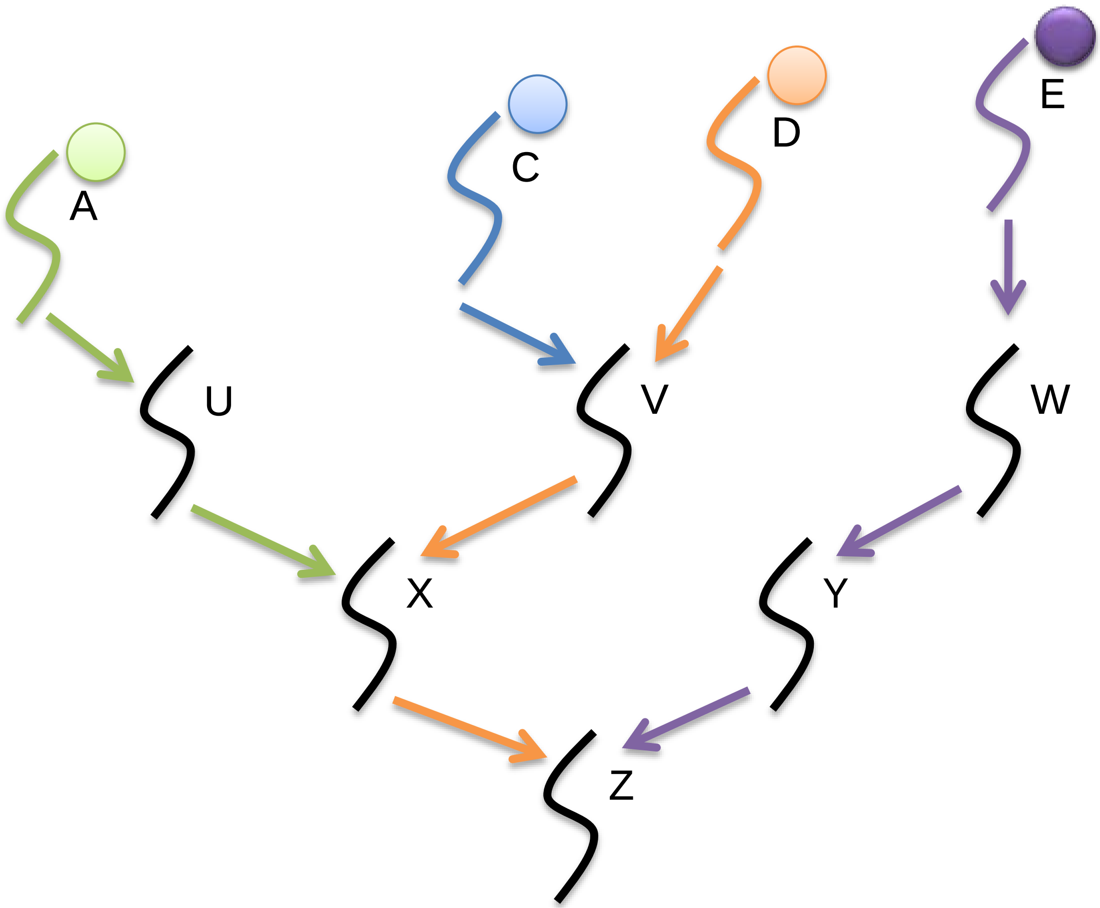
Update required when:

- New node added or new link established
- Node removed or existing link broken
- Priority of a node changes

Inconsistent state until dependency tree has been fully updated.

* Presented at ECRTS'05

Old Algorithm for Dependency Tracking*



Update required when:

- New node added or new link established
- Node removed or existing link broken
- Priority of a node changes

Inconsistent state until dependency tree has been fully updated.

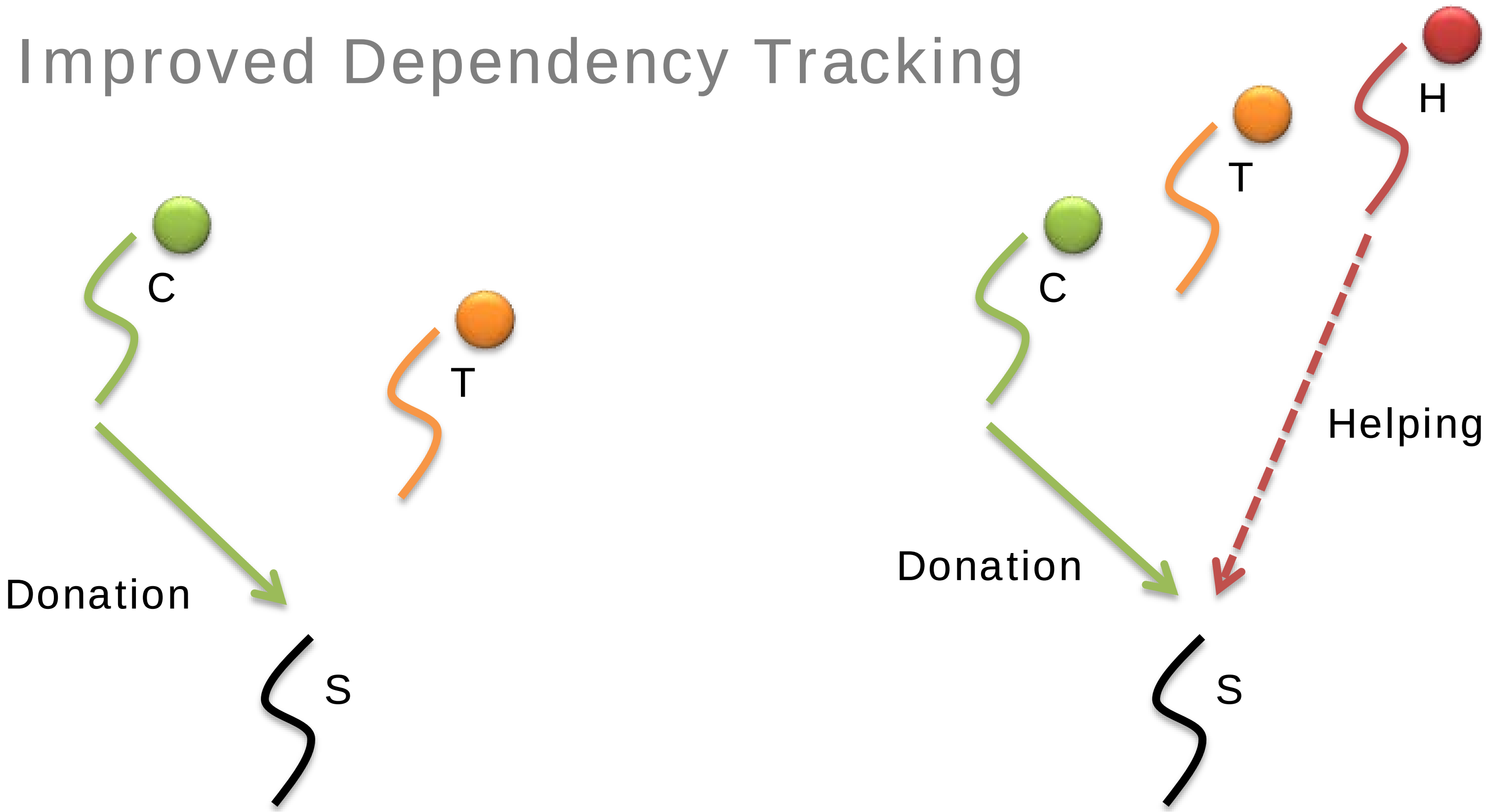
* Presented at ECRTS'05

New Requirements

- Dependency updates must be preemptible operations
  - Prevents DoS attacks via updates to long call chains
- Accounting
  - Dependency tracking must be accounted to the client that initiated the request
- Move expensive tracking operations from the IPC path into slower paths (scheduler)

TECHNISCHE UNIVERSITÄT DRESDEN

Improved Dependency Tracking

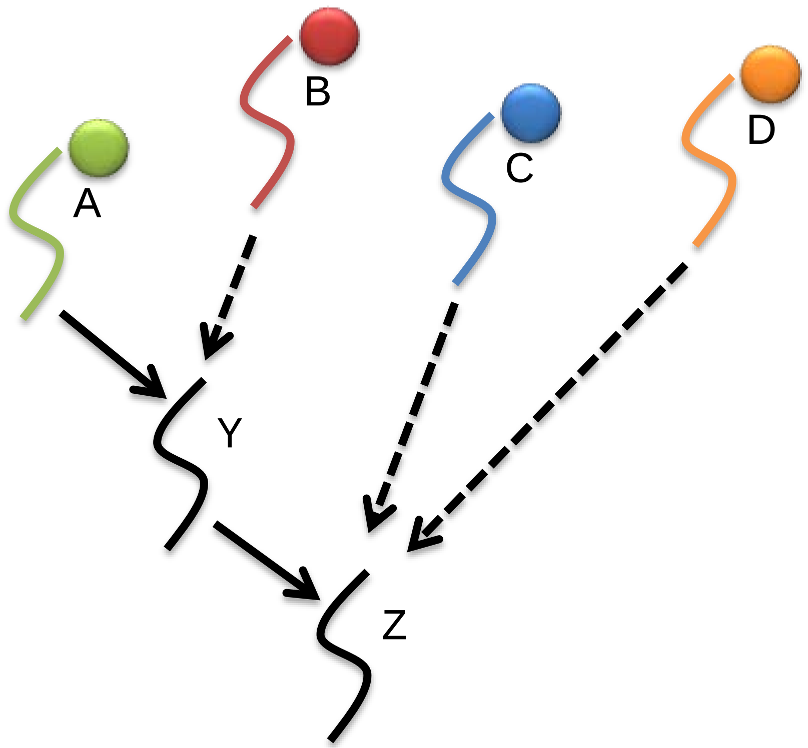


Donation: Kernel follows explicit link from C to S

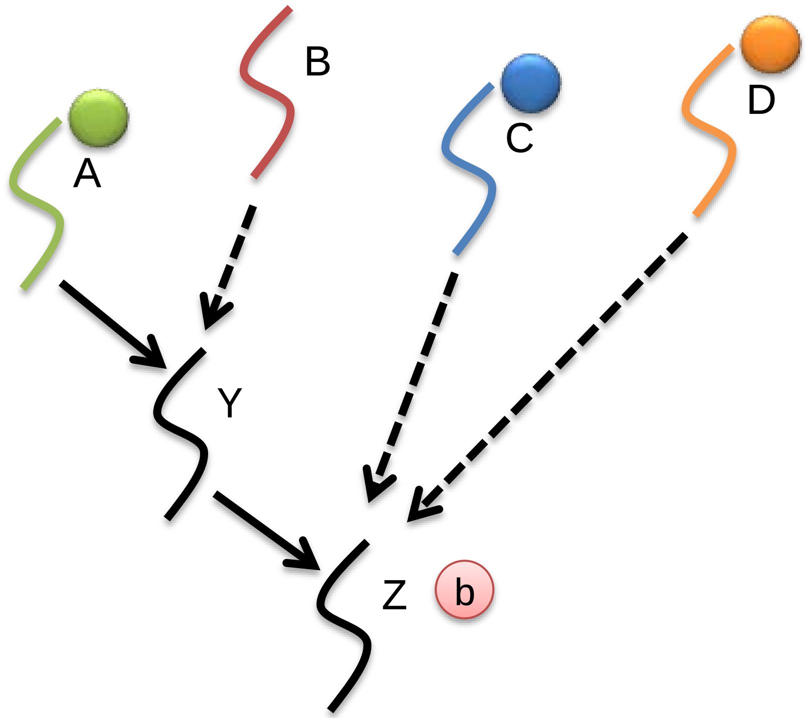
Helping: H retries its operation; switches to S if rendezvous fails

Both mechanisms are transitive.

Blocked Resource Holder

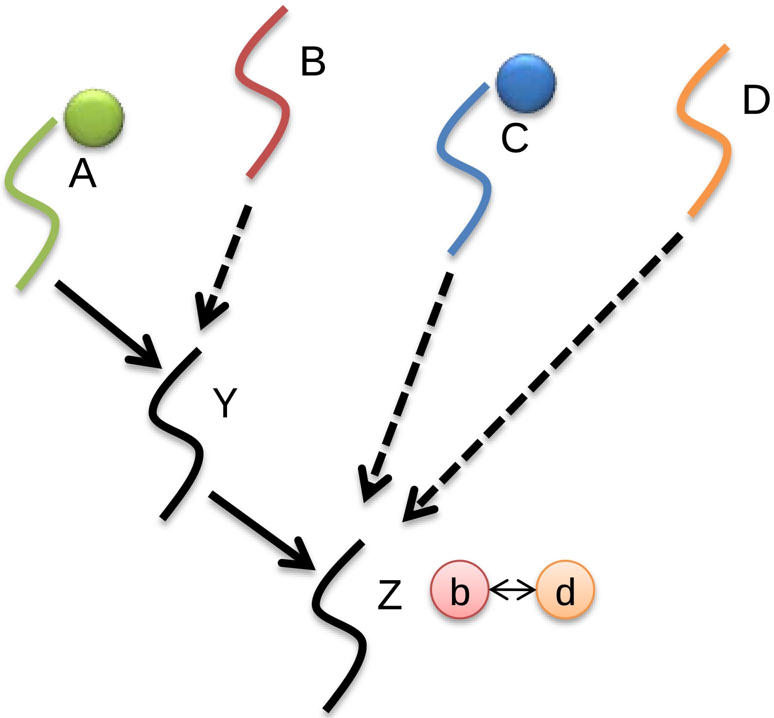


Blocked Resource Holder



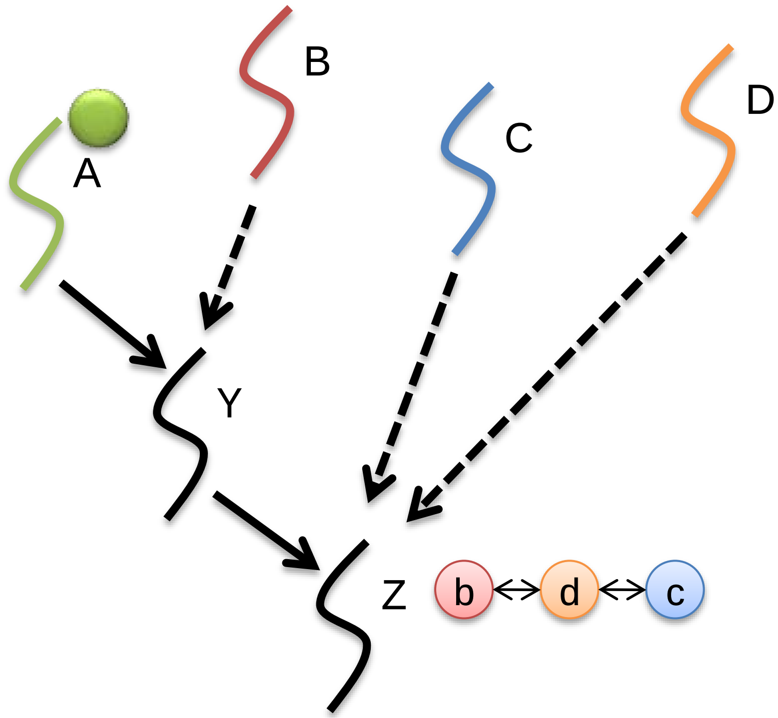
- Initially Z is running
  - consumes highest-priority timeslice B
- When Z blocks
  - B added to list of timeslices blocked on Z
- When scheduler selects D, C, A
  - These timeslices become blocked on Z as well

Blocked Resource Holder



- Initially Z is running
  - consumes highest-priority timeslice B
- When Z blocks
  - B added to list of timeslices blocked on Z
- When scheduler selects D, C, A
  - These timeslices become blocked on Z as well

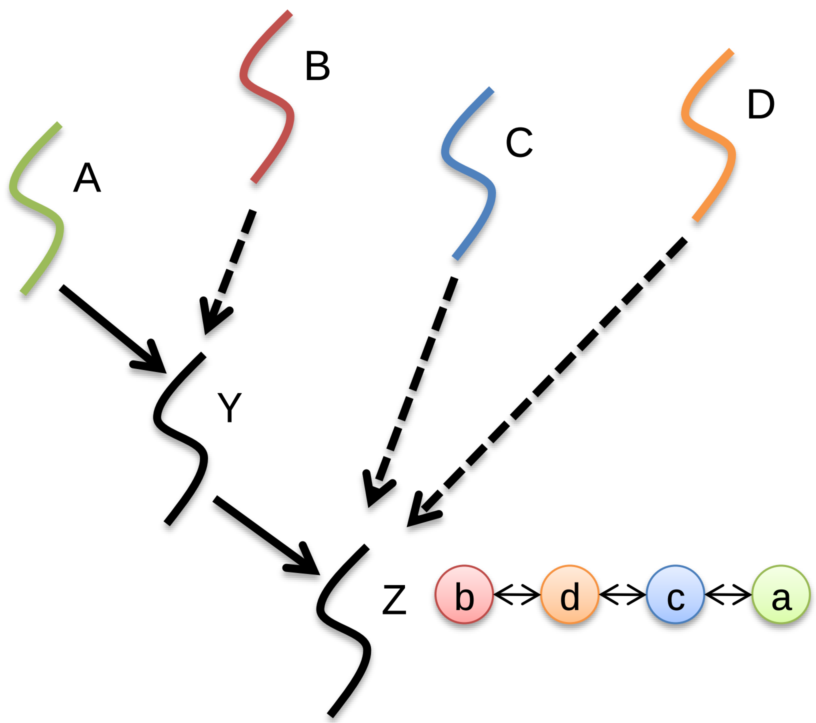
Blocked Resource Holder



- Initially Z is running
  - consumes highest-priority timeslice B
- When Z blocks
  - B added to list of timeslices blocked on Z
- When scheduler selects D, C, A
  - These timeslices become blocked on Z as well

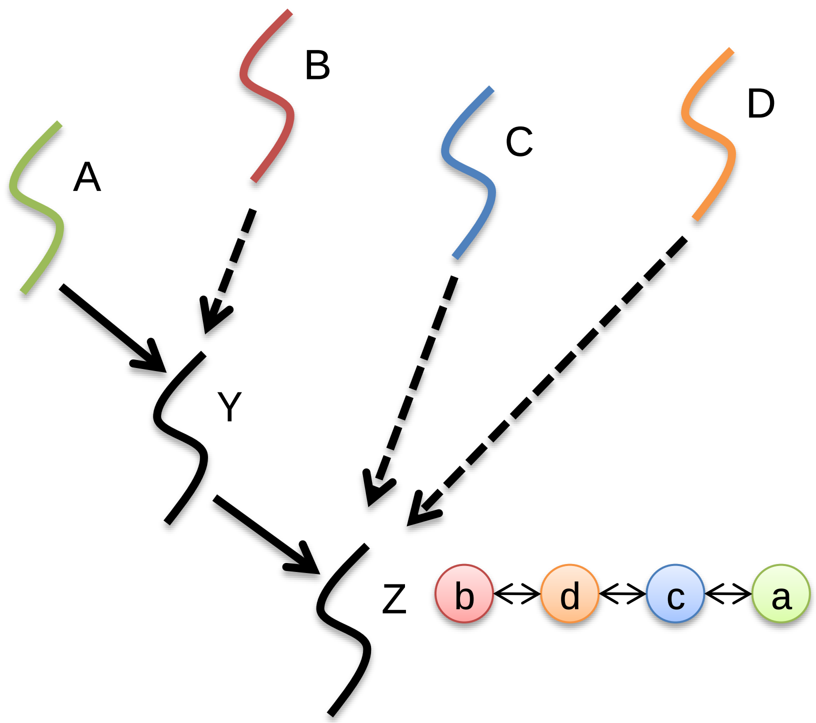
TECHNISCHE UNIVERSITÄT DRESDEN

Blocked Resource Holder

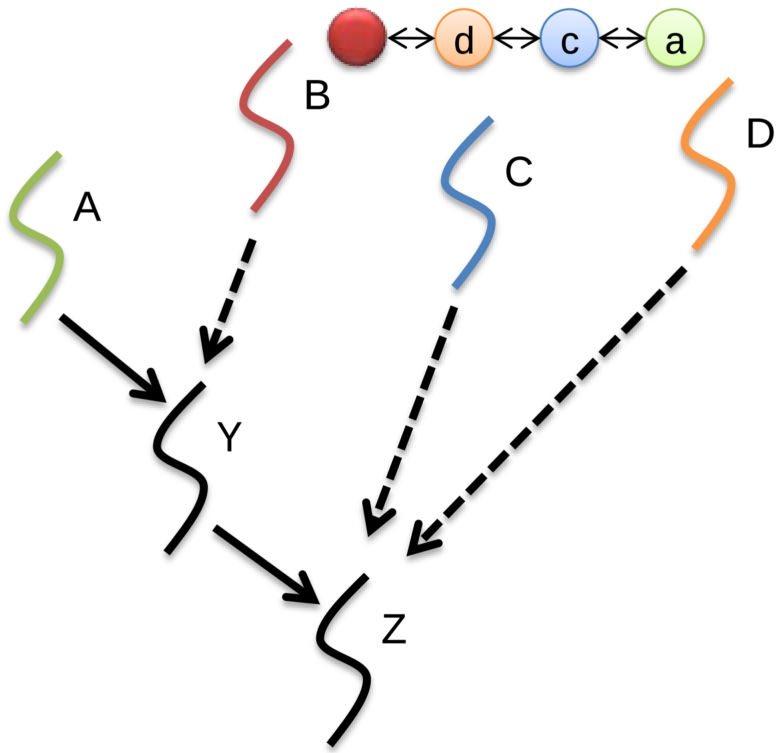


- Initially Z is running
  - consumes highest-priority timeslice B
- When Z blocks
  - B added to list of timeslices blocked on Z
- When scheduler selects D, C, A
  - These timeslices become blocked on Z as well

Staggered Wakeup



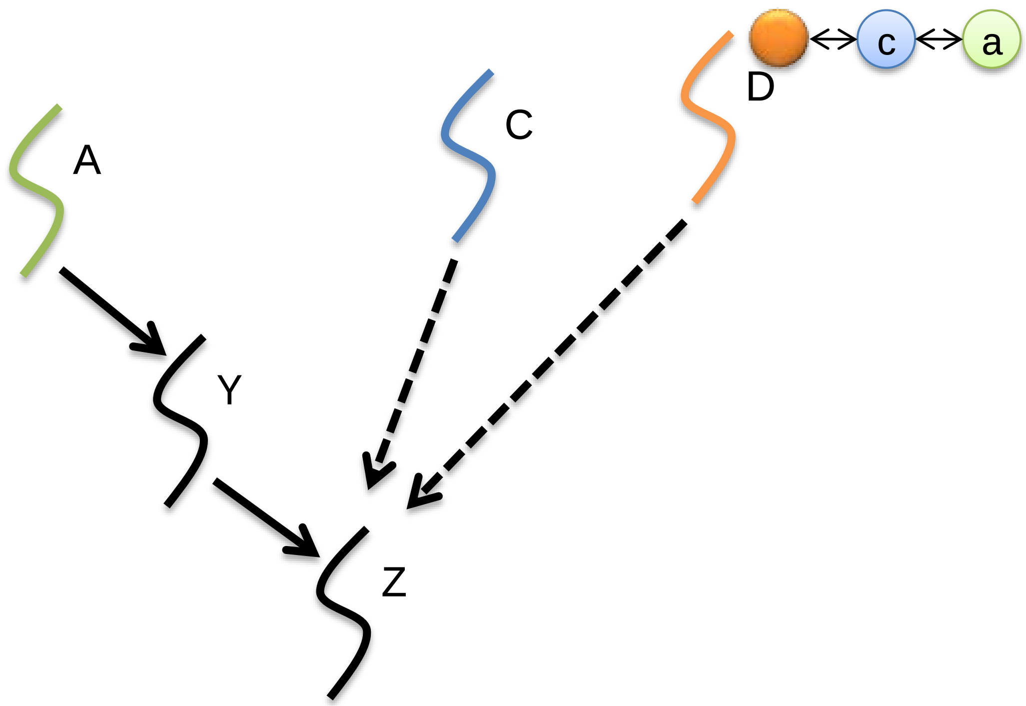
Staggered Wakeup



- Z unblocks
  - Highest-priority timeslice B added back to runqueue.
  - Other timeslices are not released yet and remain linked to B.

- When B is removed from the runqueue, next timeslice D is released.

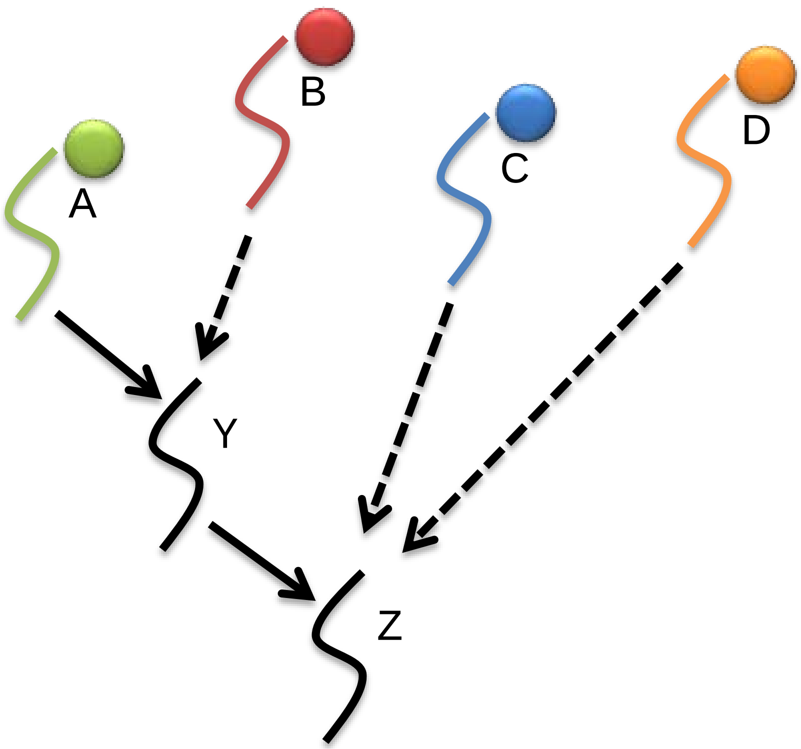
Staggered Wakeup



- Z unblocks
  - Highest-priority timeslice B added back to runqueue.
  - Other timeslices are not released yet and remain linked to B.

- When B is removed from the runqueue, next timeslice D is released.

Direct Switching

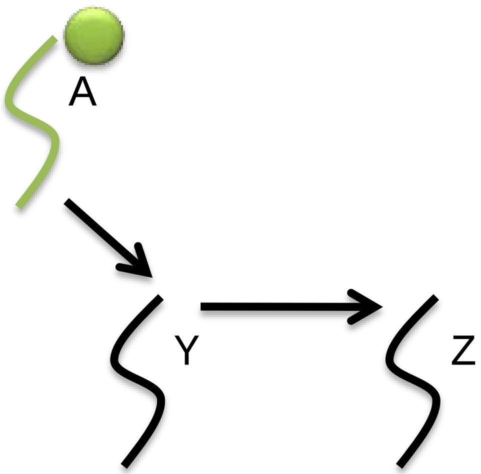


Donation Counter: A=2, B=1, C=0, D=0

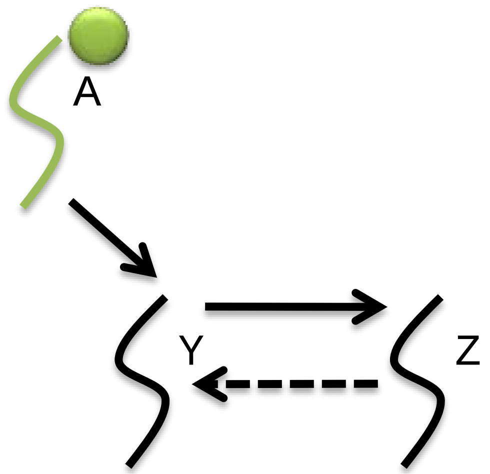
- Can the kernel switch directly from Z back to Y without scheduler invocation?
- Kernel must know if incoming link of current timeslice and reply link are the same.

- Count number of consecutive donation links during traversal
  - Per-CPU *donation counter* indicates how often kernel can switch back directly.
  - Counter update is the only overhead added to IPC

Livelock Detection



Livelock Detection



- A calls Y; Y calls Z
- Deadlock occurs if Z calls Y back
  - On most systems the deadlock will go unnoticed, freezing A, Y, and Z.
- Helping mechanism will turn the deadlock into a livelock
  - Per-CPU *helping counter* tracks number of helping links for the current timeslice.
  - Livelock detected if helping counter exceeds # of threads.

TECHNISCHE UNIVERSITÄT DRESDEN

Conclusion

- Timeslice Donation mechanism in NOVA facilitates:
  - Priority Inheritance
  - Bandwidth Inheritance
- Minimal Overhead:
  - IPC: Donation counter update (1 Add/Sub Operation)
  - Donation Link: Pointer Dereference (1 Cache Miss)
  - Helping Link: Thread Switch
- Dependency tracking is a preemptible operation that is accounted to the thread that initiated the IPC request.