

# On Randomization versus Synchronization in Distributed Systems

Hagen Völzer

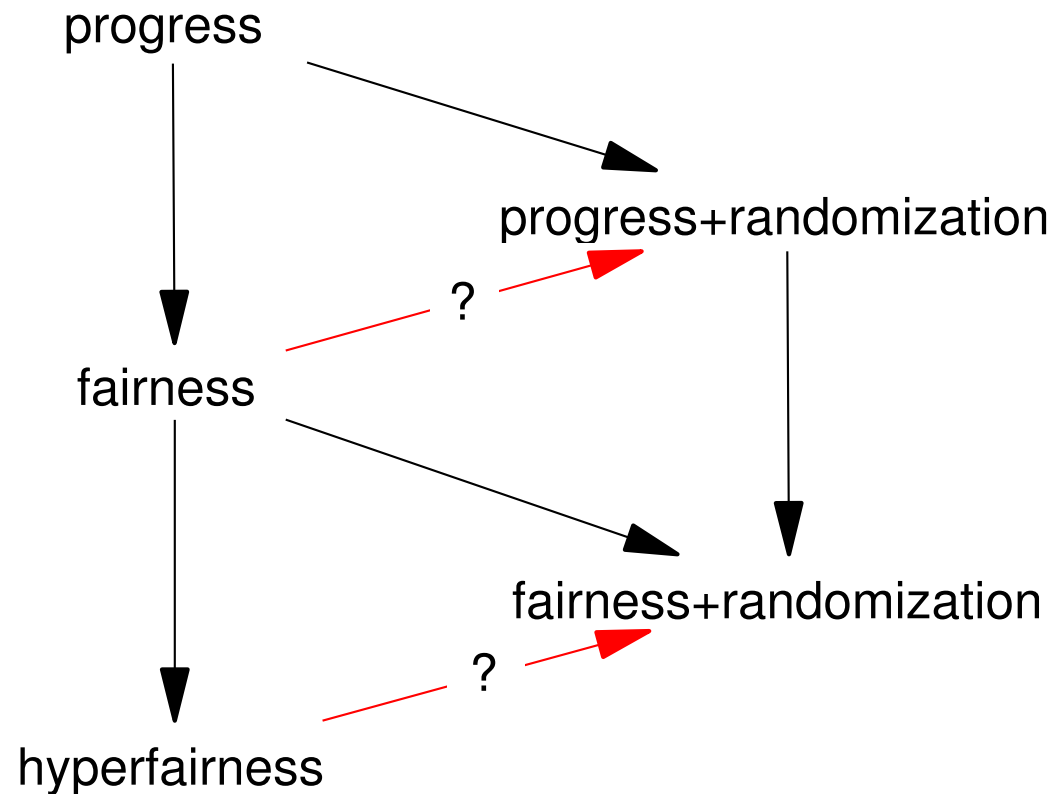
Institut für Theoretische Informatik  
Universität zu Lübeck  
Germany

July 14, 2004  
ICALP 2004, Turku

## Outline

- two new impossibility results for randomization
- complete a picture of the power of *liveness assumptions*
- consider only asynchronous systems

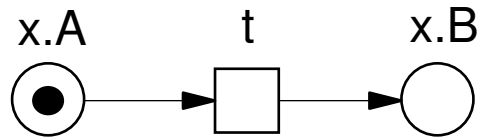
# Power of liveness assumptions



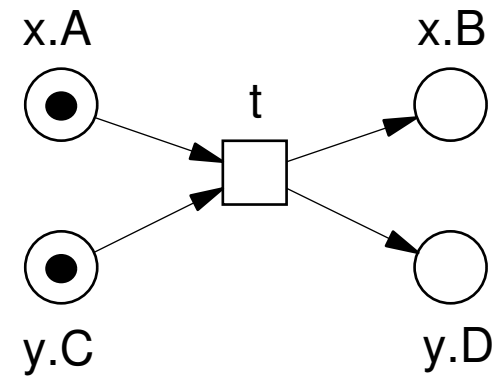
# Outline

1. Preliminaries
2. Progress
3. Fairness
4. Progress + randomization
5. Fairness + randomization
6. Hyperfairness

## Petri net modelling (1/3)

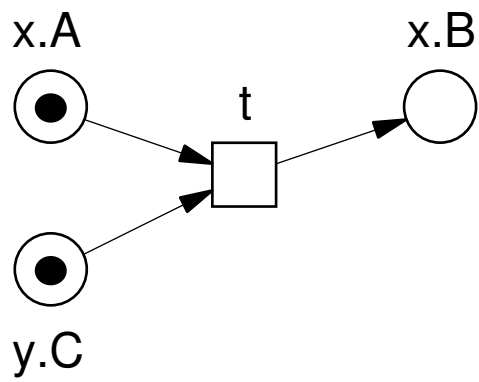


local change

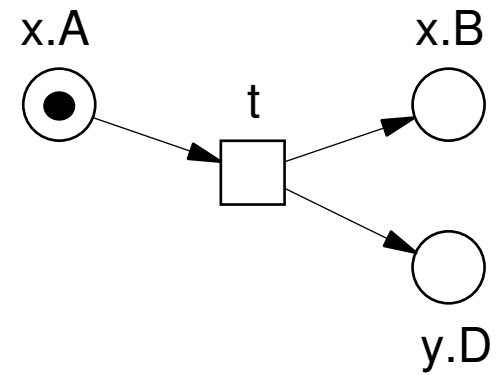


synchronization

## Petri net modelling (2/3)

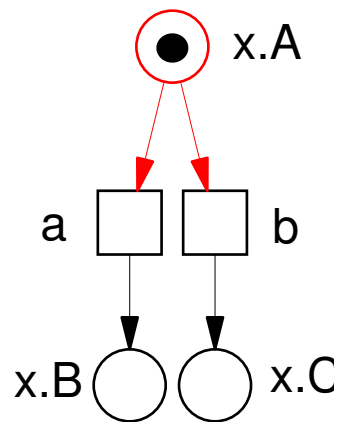


join

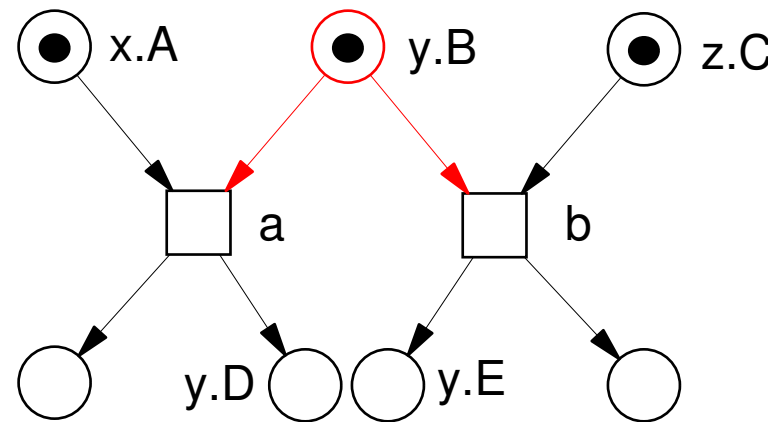


fork

## Petri net modelling (3/3)



local conflict  
(free choice)

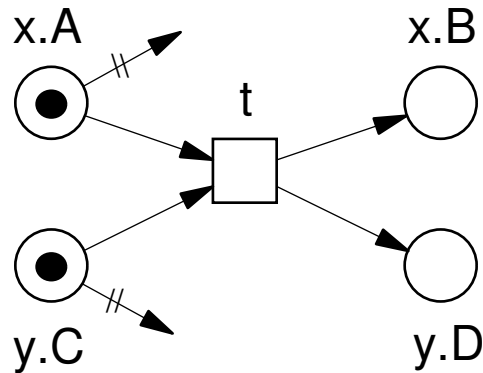


synchronization conflict

# Outline

1. Preliminaries
2. Progress
3. Fairness
4. Progress + randomization
5. Fairness + randomization
6. Hyperfairness

## Progress wrt. $t$

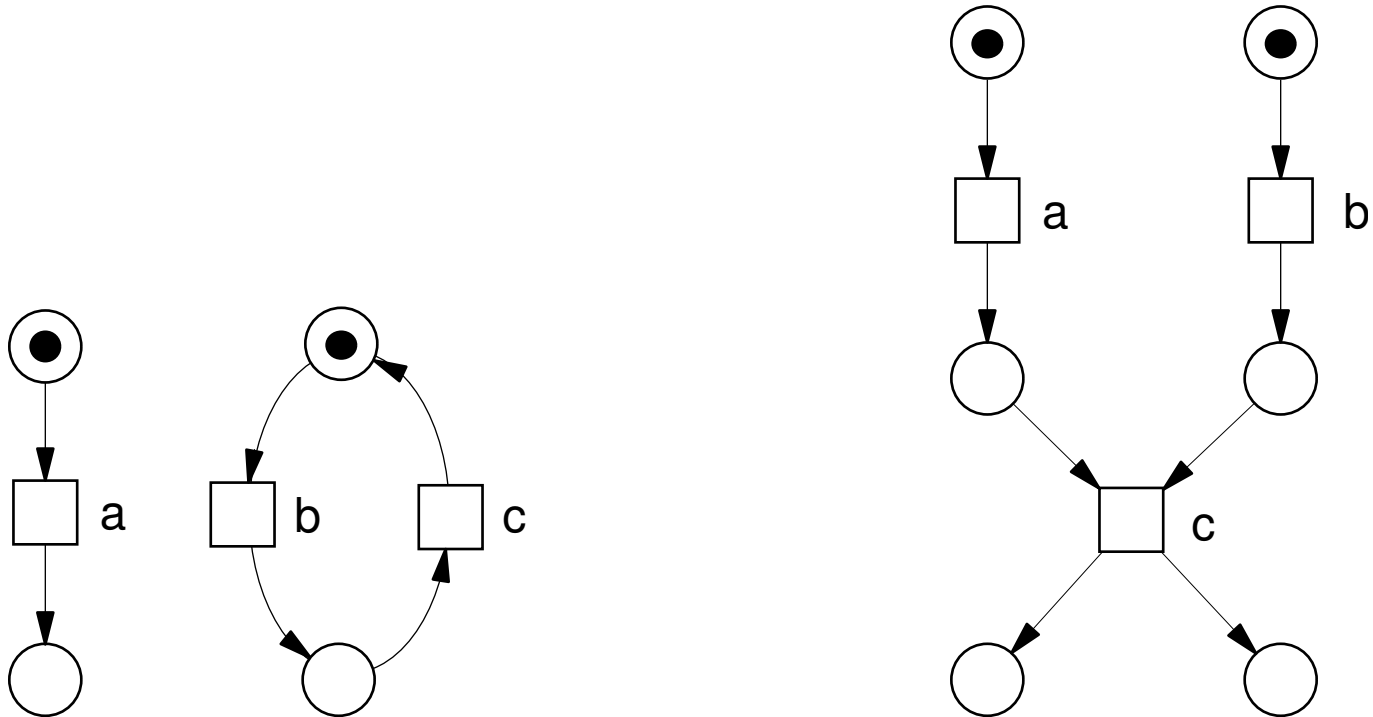


=  $t$  eventually occurs if all participants wait for  $t$

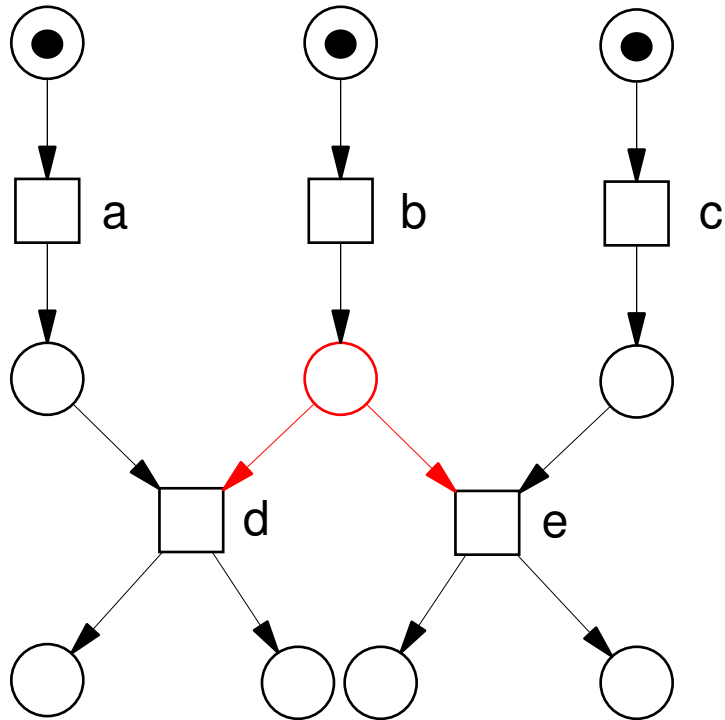
$\Leftrightarrow$  when enabled,  $t$  eventually occurs or a conflicting transition occurs

$\approx$  *weak fairness for  $t$  (justice)*

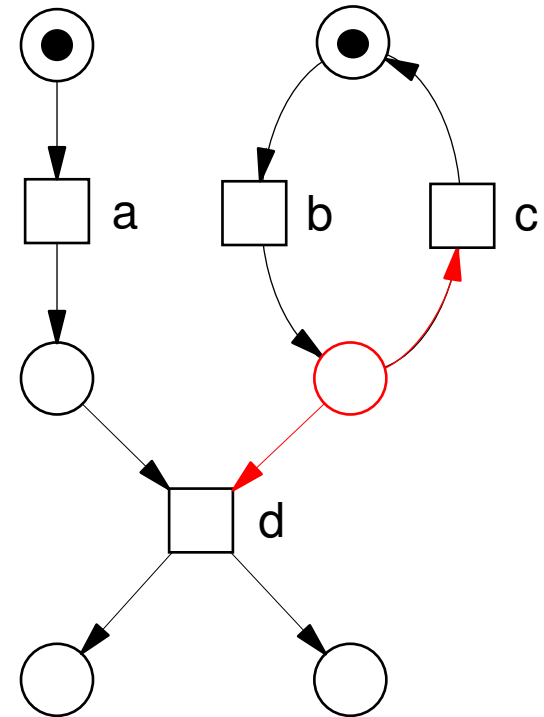
## Progress (examples)



## Progress and conflicts



conflicts are resolved eventually...



...but how is unrestricted

## Power of progress

solvable through progress:

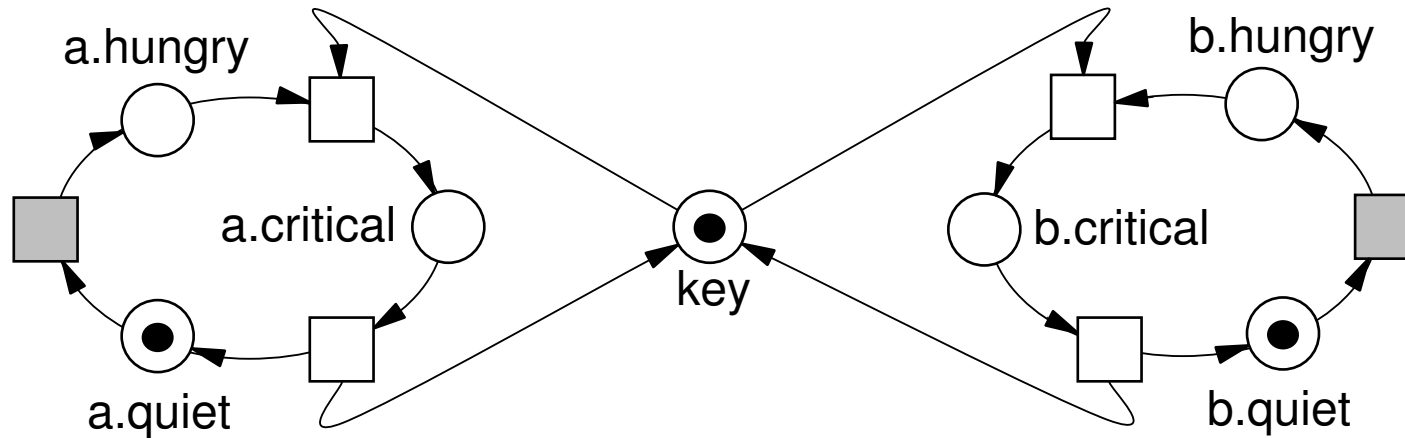
- most distributed *tasks* = compute a final distributed output depending on an initial distributed input (assuming fault-free environment)

e.g. distributed maximum finding

not solvable through progress:

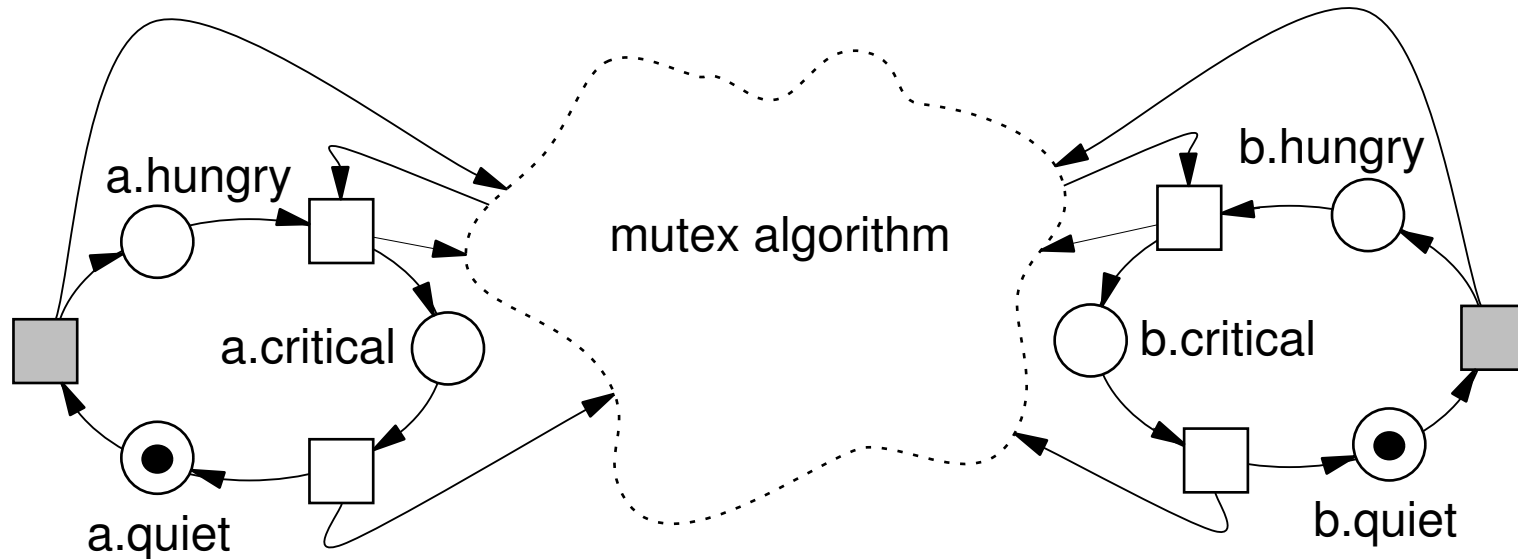
- starvation-free mutual exclusion (mutex) (Kindler and Walter 97, Vogler 97)

## Starvation free mutex



- $a$  and  $b$  never critical at the same time
- each hungry process eventually becomes critical
- progress is not sufficient

## Formalization



- algorithm may learn in which state a client is
- algorithm may only restrict transition to critical state

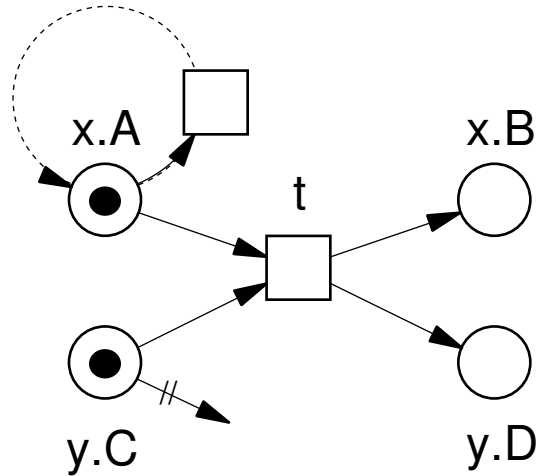
## Intuition for the impossibility

- all occurrences of critical states ( $a$  or  $b$ ) must be causally ordered
- hence there is a conflict as to which process “gets” the  $k$ -th occurrence
- conflict resolution is unrestricted, so resolve always in favour of  $a$

# Outline

1. Preliminaries
2. Progress
3. Fairness
4. Progress + randomization
5. Fairness + randomization
6. Hyperfairness

## Fairness wrt. $t$

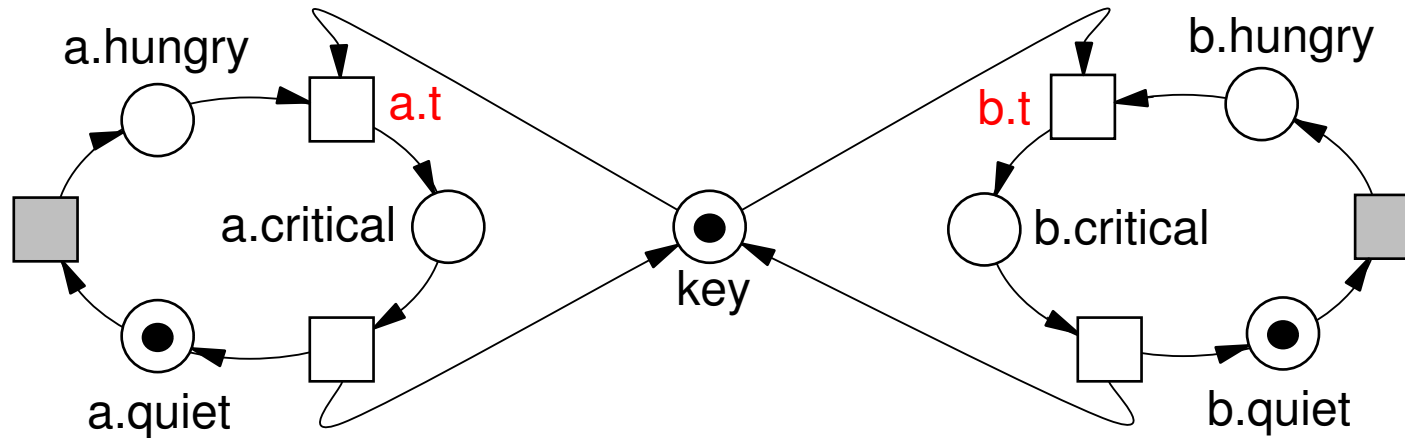


$\equiv t$  eventually occurs if one participant waits for  $t$  while the other is always eventually ready for  $t$

$\approx$  *strong fairness for  $t$  (compassion)*



## Mutex revisited



- each hungry process eventually becomes critical if
- fairness is assumed for  $a.t$  and  $b.t$

## Power of fairness

solvable through fairness:

- what we can solve through progress
- mutex

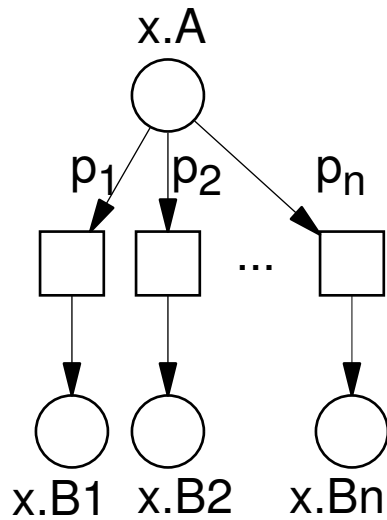
not solvable through fairness:

- crash-tolerant message-passing consensus  
(Fischer, Lynch, Paterson 83)

## Outline

1. Preliminaries
2. Progress
3. Fairness
4. Progress + randomization
5. Fairness + randomization
6. Hyperfairness

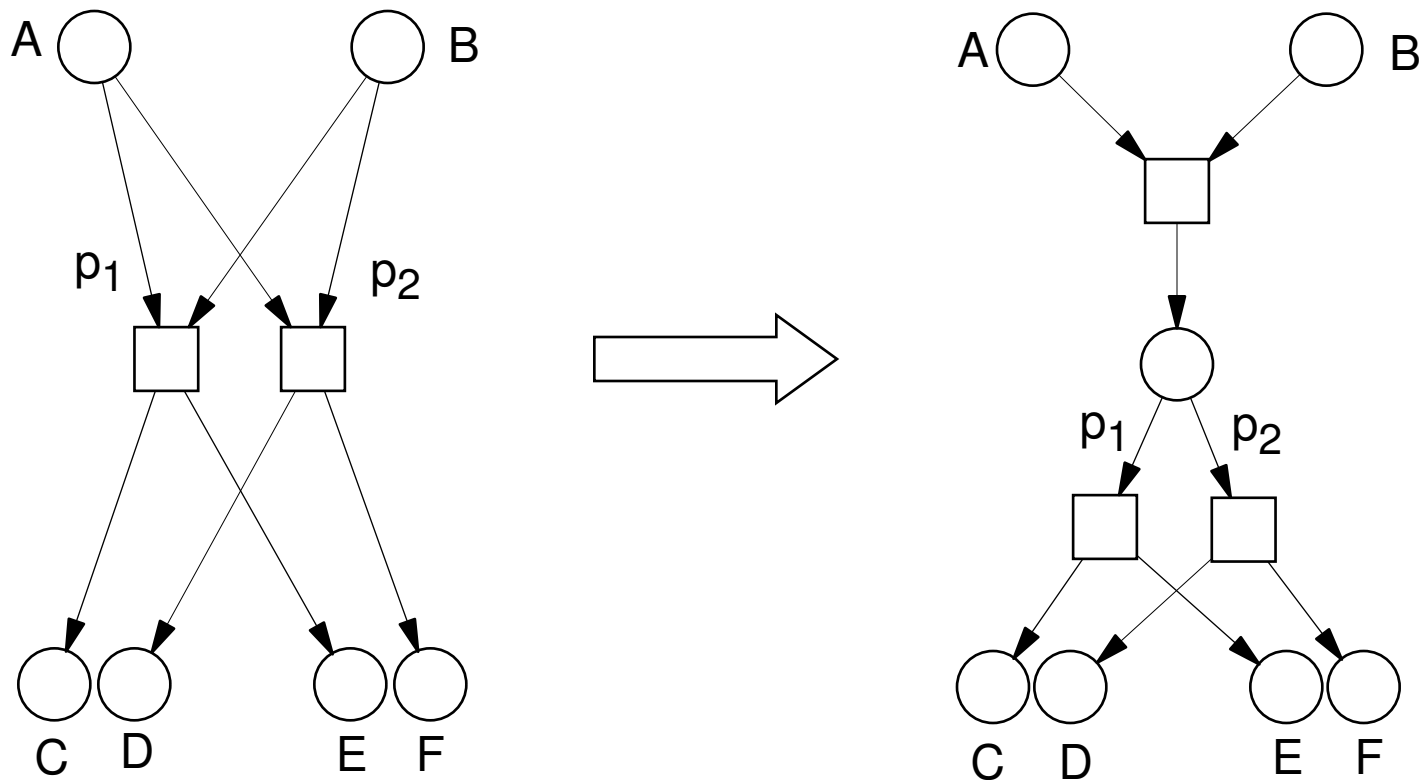
## Randomization



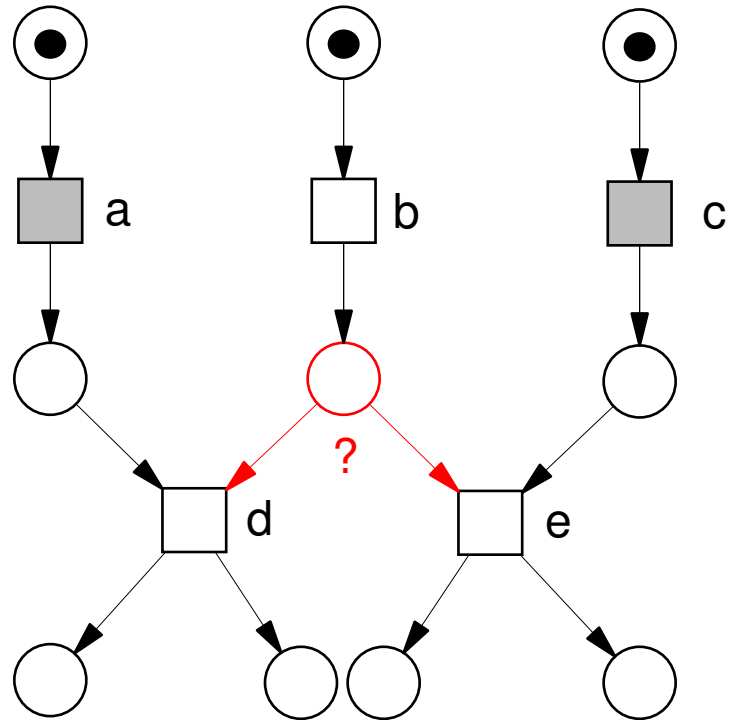
- finite local conflict  
(i.e. without synchronization)
- $0 < p_i < 1$
- $\sum_{i=1}^n p_i = 1$

an object flipping an  $n$ -sided coin

# Synchronization + coin flip



## Synchronization and randomization



## Power of randomization

solvable through progress+randomization with probability 1:

- crash-tolerant consensus (we don't need fairness!)  
(Ben-Or 83)

not solvable through progress+randomization:

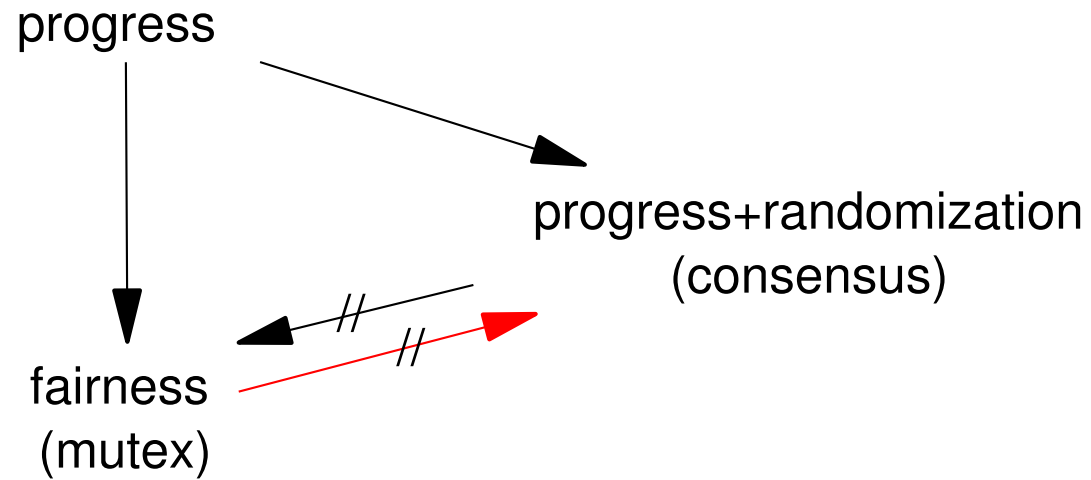
- ?
- Does randomization help for mutex?
- Is fairness implementable through randomization?

## First result

**Theorem:** Starvation-free mutex cannot be solved with probability 1 through progress + randomization.

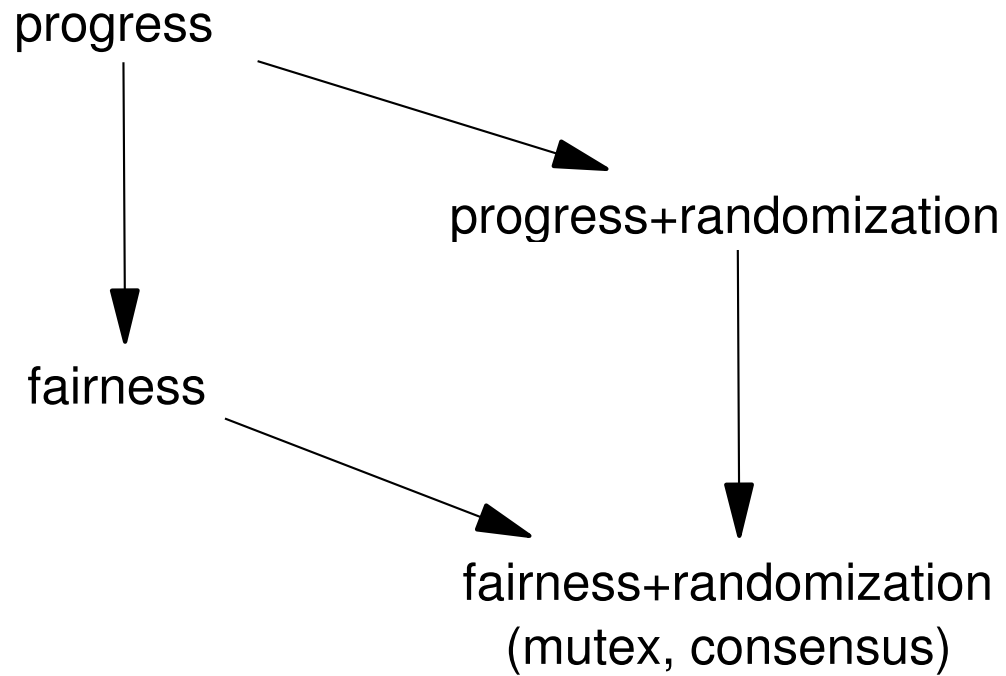
- inherent unfairness cannot be “implemented away” through randomization
- proof for deterministic case can be adapted to the randomized case
- proof uses a partial-order semantics for randomization (Völzer 2001)

## Overview



Fairness cannot be implemented through progress + randomization.

## Overview



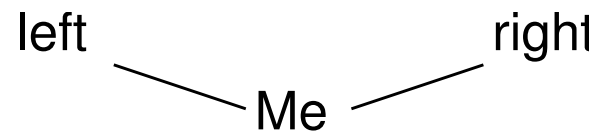
Where does the power of fairness + randomization end?

## Outline

1. Preliminaries
2. Progress
3. Fairness
4. Progress + randomization
5. Fairness + randomization
6. Hyperfairness

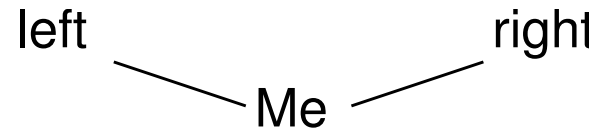
## Dining philosophers

consider irreflexive and symmetric relation on processes, e.g.



- solve mutex for each pair of neighbours simultaneously:
  - neighbours are never critical at the same time
  - each hungry process eventually becomes critical
- fairness is necessary and sufficient

## Crash-tolerant dining philosophers



- processes may crash (permanently)
  - revised requirements:
    - neighbours are never critical at the same time
    - each hungry process becomes critical unless itself or one of its neighbours crashes
- ⇒ if *left* is hungry then it eventually becomes critical even if *right* crashes

## Second result

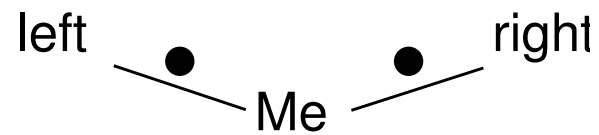
**Theorem:** Crash-tolerant dining philosophers cannot be solved with probability 1 through fairness + randomization.

### Background:

- this problem inherently contains the possibility of *conspiracy*
- fairness and randomization is not enough to “implement away” conspiracy

## Conspiracy

Dining philosophers with keys (forks):

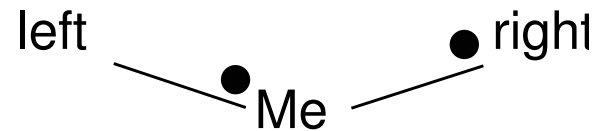


Conspiracy against me:

*left* and *right* become alternately critical in such a way that my keys are never available at the same time

## Conspiracy and crash-tolerance

Try: Take first available key and wait for second.



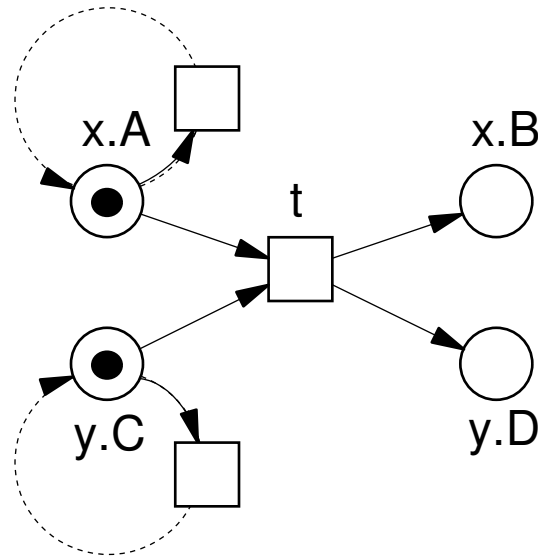
Problem crash-tolerance:

- I hold left key and wait for right key.
- *right* crashes (with key in hand)
- left will not be able to become critical

# Outline

1. Preliminaries
2. Progress
3. Fairness
4. Progress + randomization
5. Fairness + randomization
6. Hyperfairness

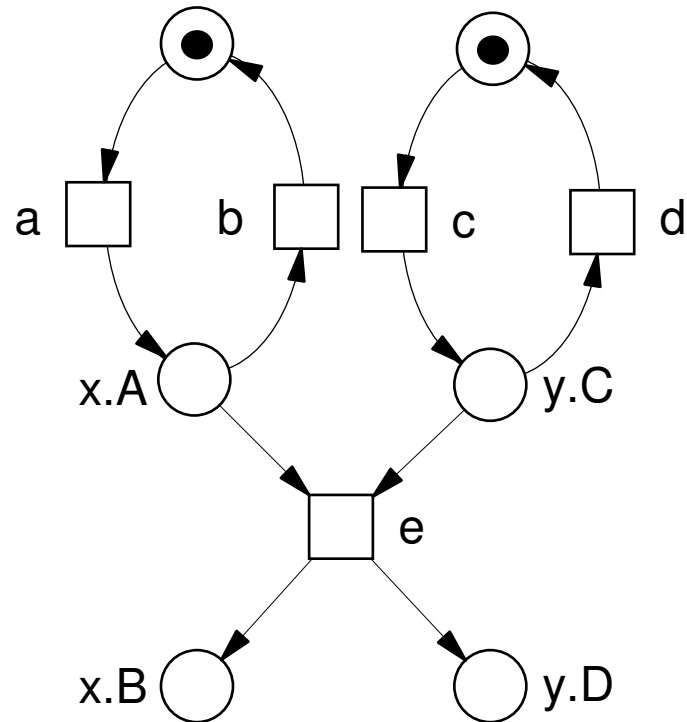
## Hyperfairness wrt. $t$



$\equiv t$  eventually occurs if all participants are always eventually ready for  $t$  *independently of each other*

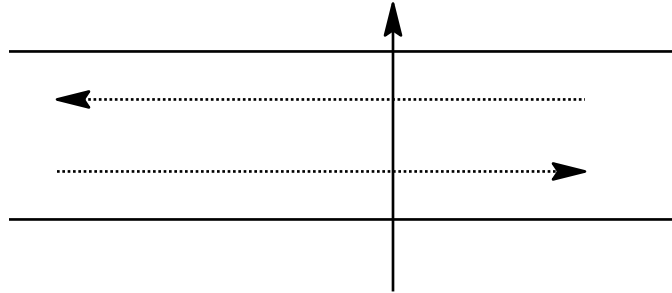
- formalization needs partial-order semantics

## Hyperfairness (example)



hyperfairness postulates the absence of conspiracy

## An intuition



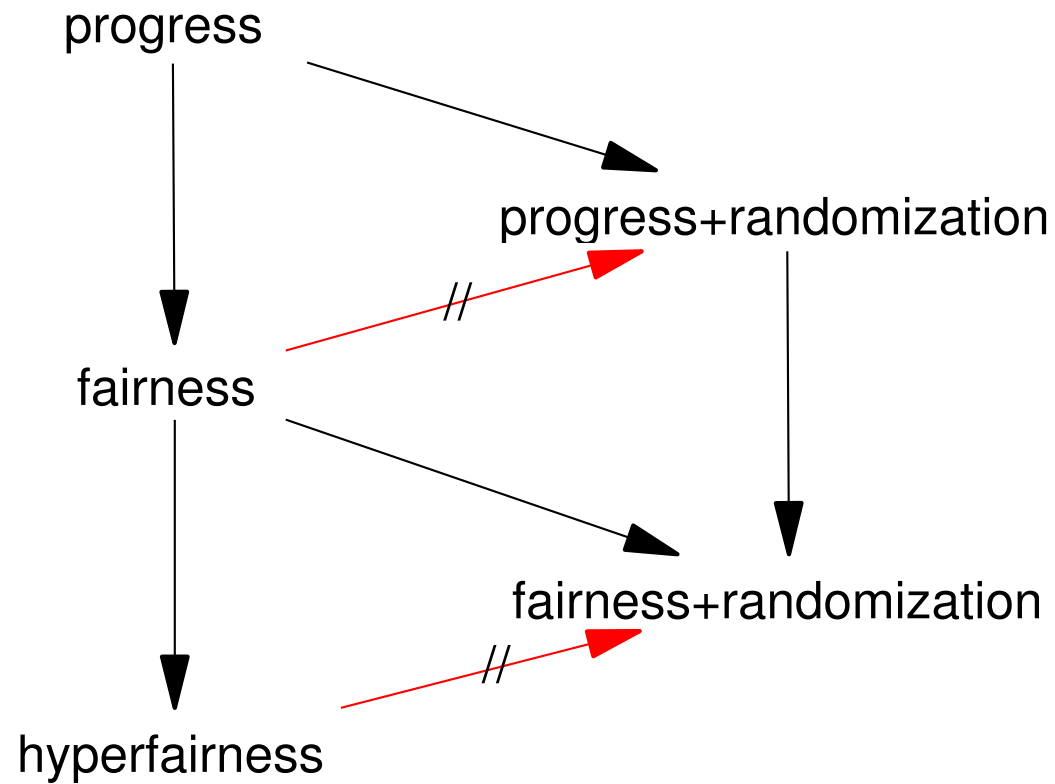
crossing a two-lane road

## Power of hyperfairness

solvable through hyperfairness:

- crash-tolerant dining philosophers
- crash-tolerant consensus

## Summary



## Conclusion

- hierarchy **progress** → **fairness** → **hyperfairness** separates computational power well
- adding randomization leads outside that hierarchy
- partial-order semantics used for proofs
- that also strengthens results