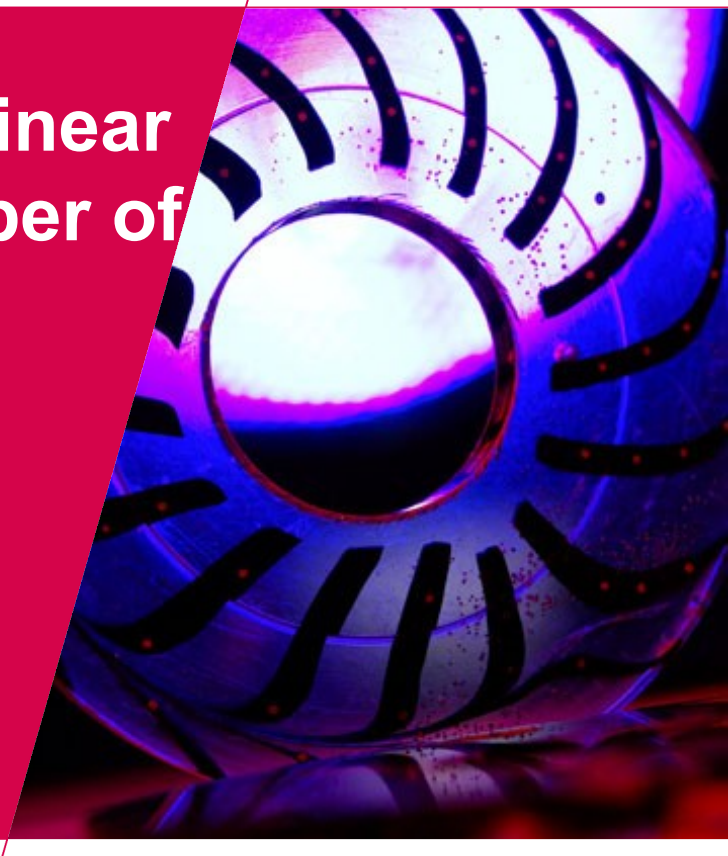
Determining Bisimulation in Linear Time with a Logarithmic Number of Processes.

Jan Friso Groote



TU/e Technische Universiteit Eindhoven University of Technology

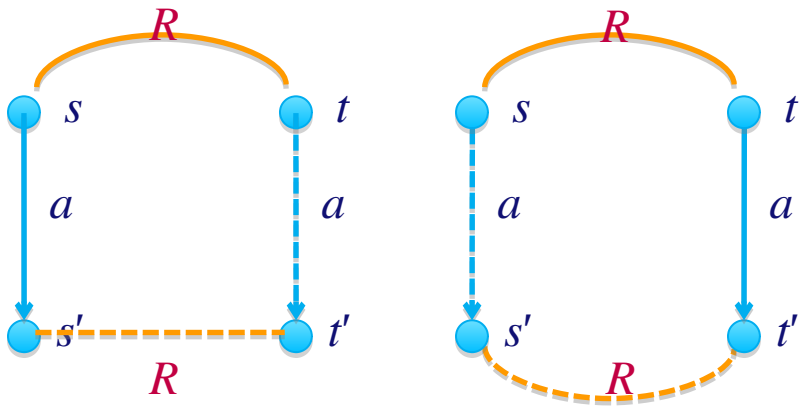
Where innovation starts

Bisimulation equivalence

Definition. Let $(S, Act, \rightarrow, s_0)$ be a labelled transition system. We say that a relation $R \subseteq S \times S$ is a bisimulation relation iff for all states $s,t \in S$ it holds that if sRt:

- if $s \xrightarrow{a} s'$ for some $s' \in S$, then there is some $t' \in S$ such that $t \xrightarrow{a} t'$ and $s' R t'$.

- if $t \xrightarrow{a} t'$ for some $t' \in S$, then there is some $s' \in S$ such that $s \xrightarrow{a} s'$ and $s' R t'$.

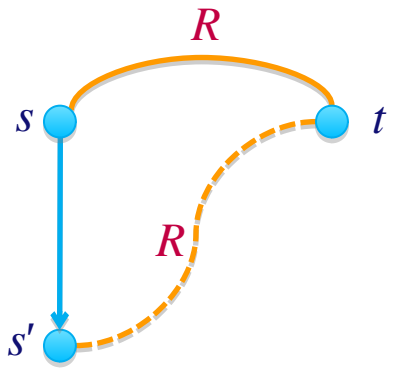


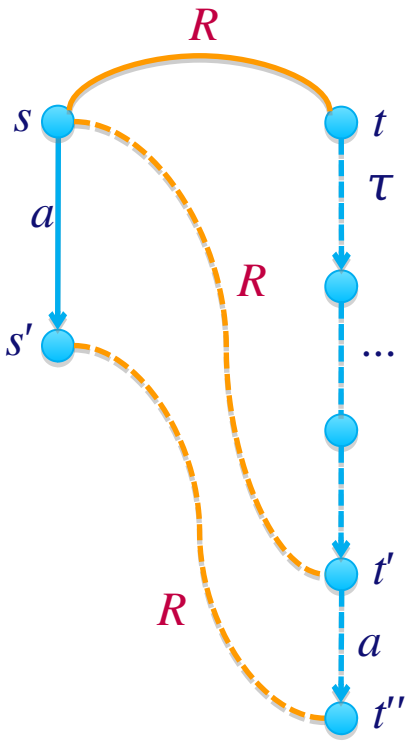
Two states $s,t \in S$ are called *bisimilar*, notation $s \leftrightarrows t$, iff there is a bisimulation relation R such that sRt. Two transition systems are bisimilar iff their initial states are bisimilar.

Branching bisimulation.

n: number of states of an LTS
m: number of transitions of an LTS

A relation R is a *branching bisimulation* relation iff





An efficient algorithm for branching bisimulation, Groote/Vaandrager, ICALP 1990.
Complexity: $O(n \cdot m)$.

Question and results.

n: number of states of an LTS
m: number of transitions of an LTS

Questions (following from the $O(m \cdot n)$ algorithm):

Is it necessary that an algorithm for branching bisimulation has complexity $O(m \cdot n)$?

No. There is an $O(m \log n)$ algorithm. [Groote/Wijs2015, Groote/Jansen Keiren/Wijs17, Jansen/Groote/Keiren/Wijs19]

Question: Is $O(m \log n)$ necessary to calculate strong/branching bisimulation? Yes.

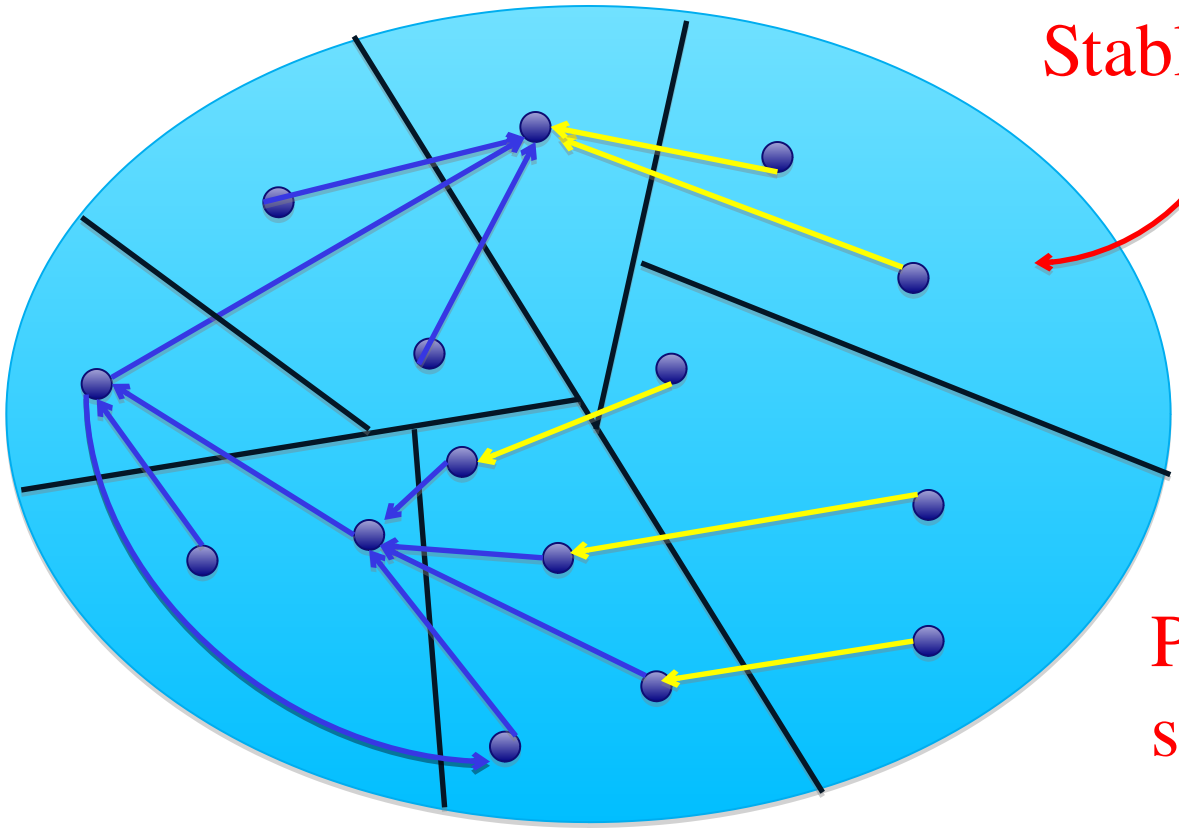
Question: Is $O(m \log n)$ necessary to calculate bisimulation in parallel?

No. There is an $O(m+n)$ algorithm using $n+m$ processors.
[Martens/Groote/Haak/Hijma/Wijs 2021]

Question: Is $O(n)$ necessary to calculate bisimulation in parallel? Yes.

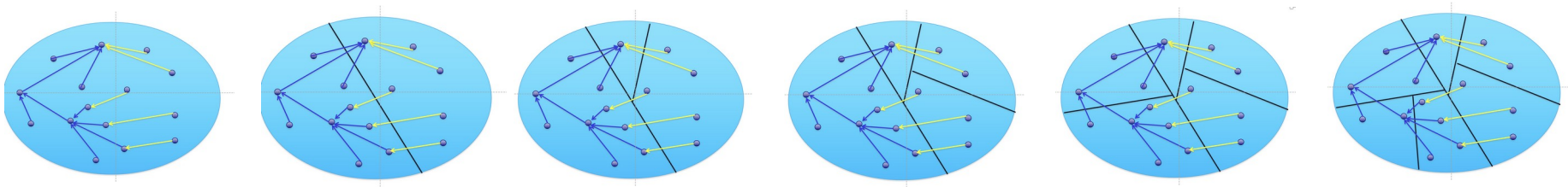
Open question:
Can bisimulation be calculated in time $O(n)$ with appr. $\log(n)$ processors

Algorithms for bisimulations are partition based.



Theorem.
If stable, states are in the same block iff they are strongly bisimilar.

Bisimilarity valid sequences of partitions

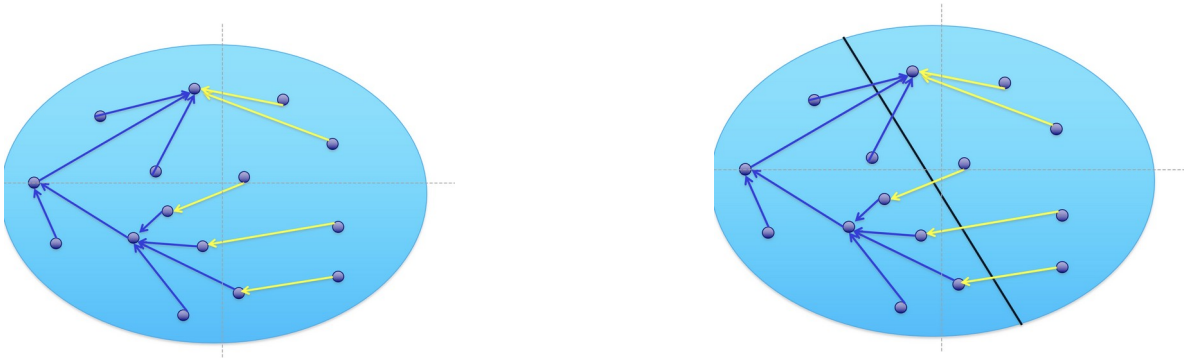


π_0 π_1 π_2 π_3 π_4 π_5

A sequence $\pi_0,\ldots,\pi_n$ is called a *bisimilarity valid sequence of partitions* iff

- π_{i+1} is a strict refinement of π_i.
- for all states s, t if s and t are in a different block in π_{i+1}, then
  (a) they are in different blocks in π_i, or
  (b) there is a state s' such that s–a→s' and for all states t' if t–a→t', then t' is in a different block than s' in π_i.

What are minimal expenses to split?



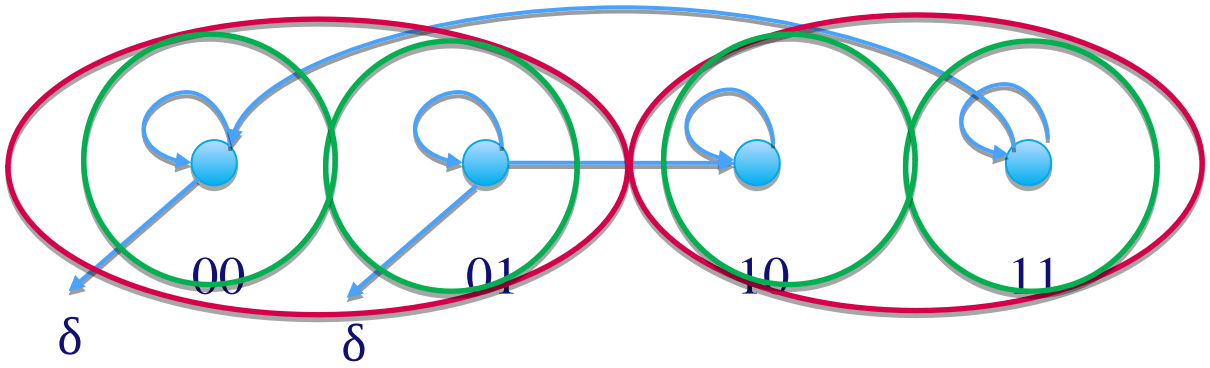
When splitting, count the minimal number of states that must move:

Intrinsic Refinement Complexity: $IRC(S)$ with S a set of states.

A nasty transition system.

States: bit sequences of length k and δ.
Transitions (unlabelled):

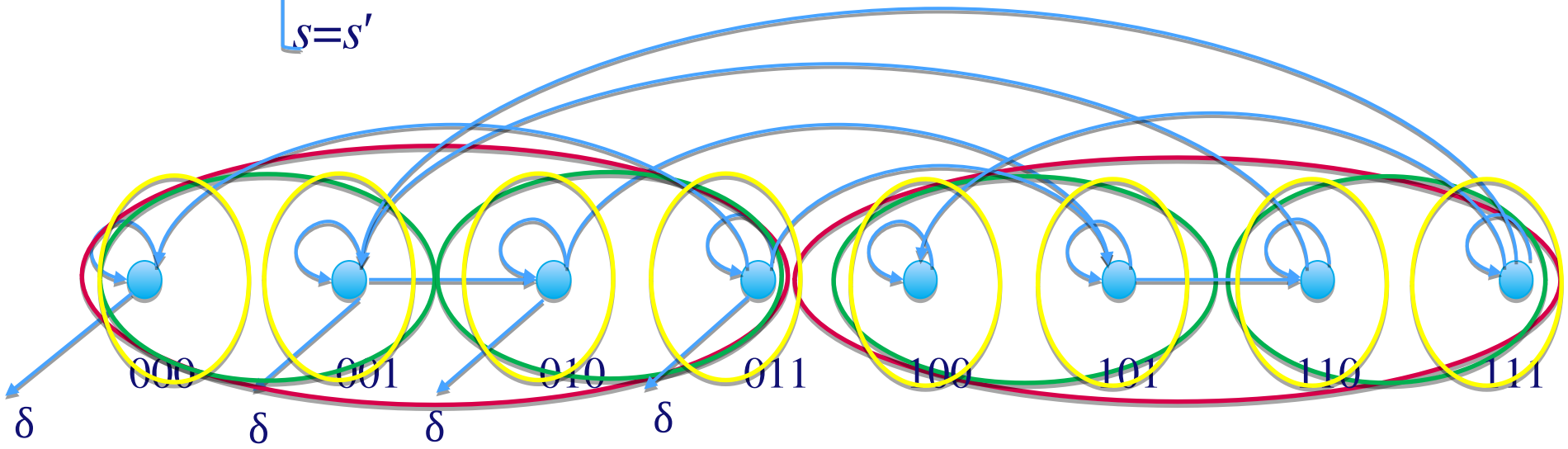
$$s \to s' \text{ iff } \begin{cases} s = \sigma_1 b 1 \sigma_2 \text{ and } s' = \sigma_1 \overline{b} 01^{|\sigma_2|} \\ s = 0\sigma \text{ and } s' = \delta \\ s = s' \end{cases}$$



A nasty transition system.

States: bit sequences of length k and δ.
Transitions (unlabelled):

$$s \to s' \text{ iff } \begin{cases} s = \sigma_1 b 1 \sigma_2 \text{ and } s = \sigma_1 \overline{b} 01^{|\sigma_2|} \\ s = 0\sigma \text{ and } s' = \delta \\ s = s' \end{cases}$$

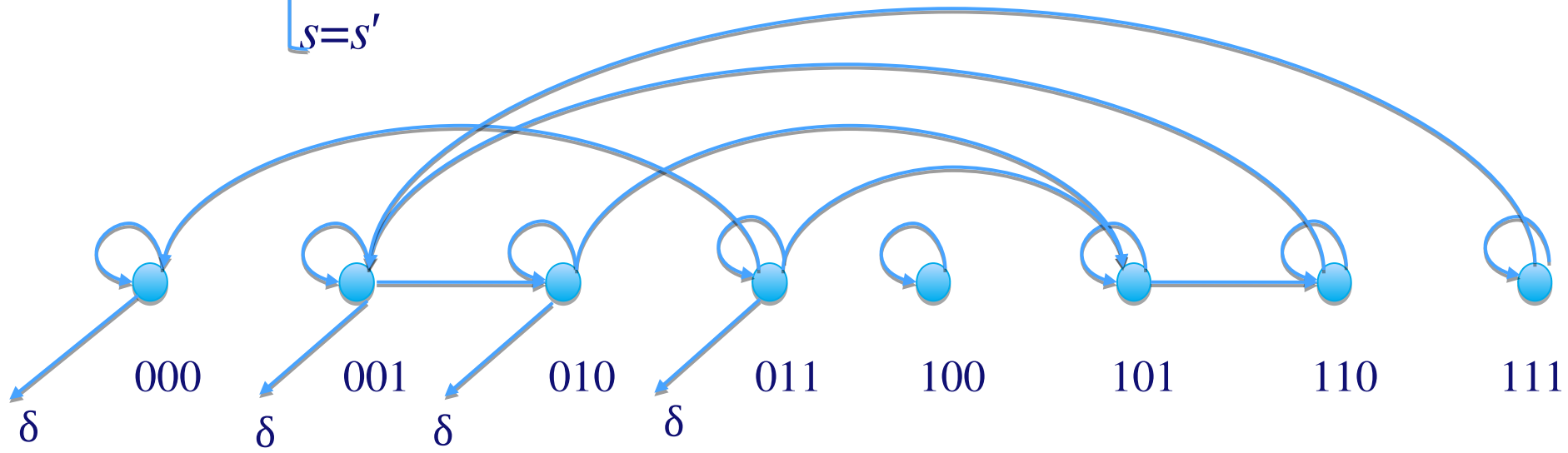


A nasty transition system.

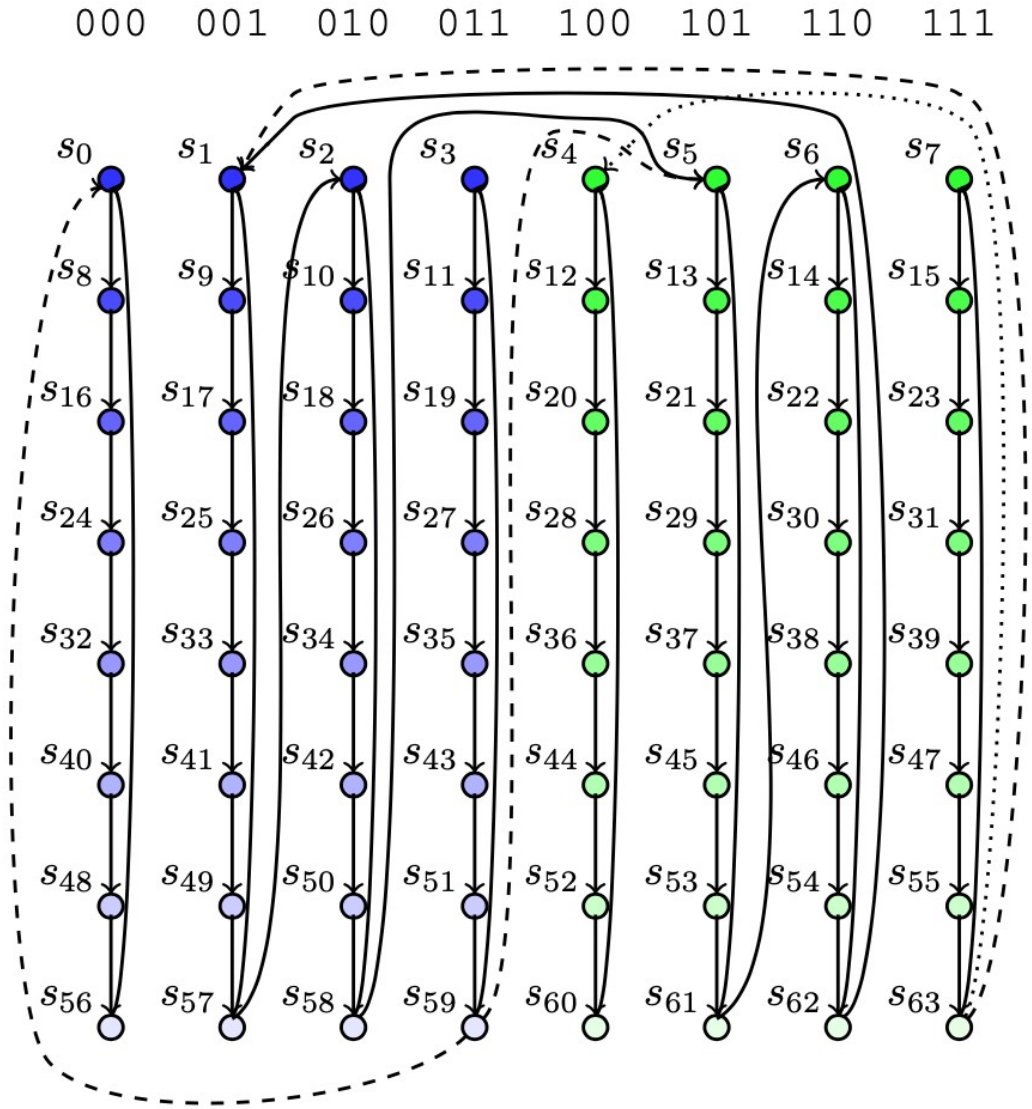
States: bit sequences of length k and δ.
Transitions (unlabelled):

2^k+1 states.
2^k+k2^{k-1} transitions.
$k2^{k-1}$ movements of states to new blocks.

$$s \to s' \text{ iff } \begin{cases} s=\sigma_1 b 1 \sigma_2 \text{ and } s=\sigma_1 \overline{b} 01^{|\sigma_2|} \\ s=0\sigma \text{ and } s'=\delta \\ s=s' \end{cases}$$



Solution: make pilars of height 2^k



$2^{2k}+1$ states.

$k2^{k-1} + 2^{2k}$ transitions.

$k2^{2k-1}$ movements of states to new blocks.

Define $n=2^{2k}+1$ to be the number of states

$k=1/2 \log(n-1)$

Then:

$k2^{2k-1}=1/4\ (n-1) \log\ (n-1)$ is the number of movements of states to a new block.

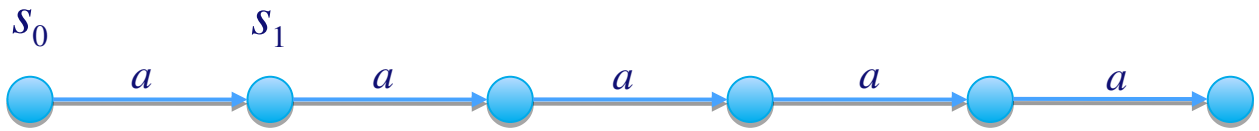
Extension to deterministic transition systems.

Algorithm using bisimilarity valid sequences:

Labelled transition systems:
- nondeterministic $\Omega(n \log n)$.
- deterministic with at least 2 action labels: $\Omega(n \log n)$.
- deterministic with one action label: $O(n+m)$.
- without loops $O(n+m)$.

Kripke structures
- nondeterministic $\Omega(n \log n)$.
- deterministic $\Omega(n \log n)$.
  but via 3-Roberts' algorithm $O(n+m)$.

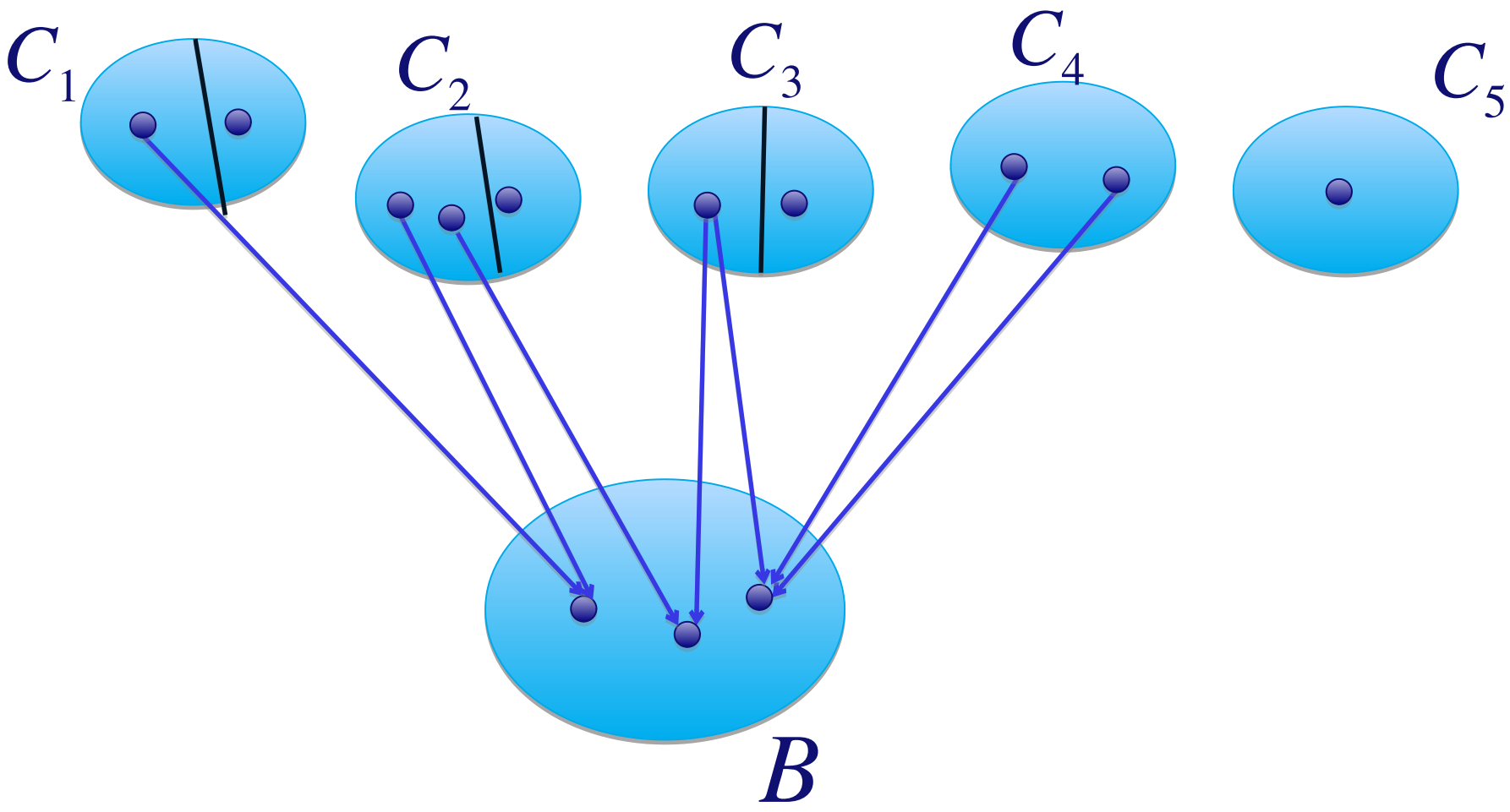
Lowerbound for parallel algorithms.



a^k takes at least k consecutive rounds to determine that state s_0 differs from state s_1, using a bisimilarity valid sequence of partitions.

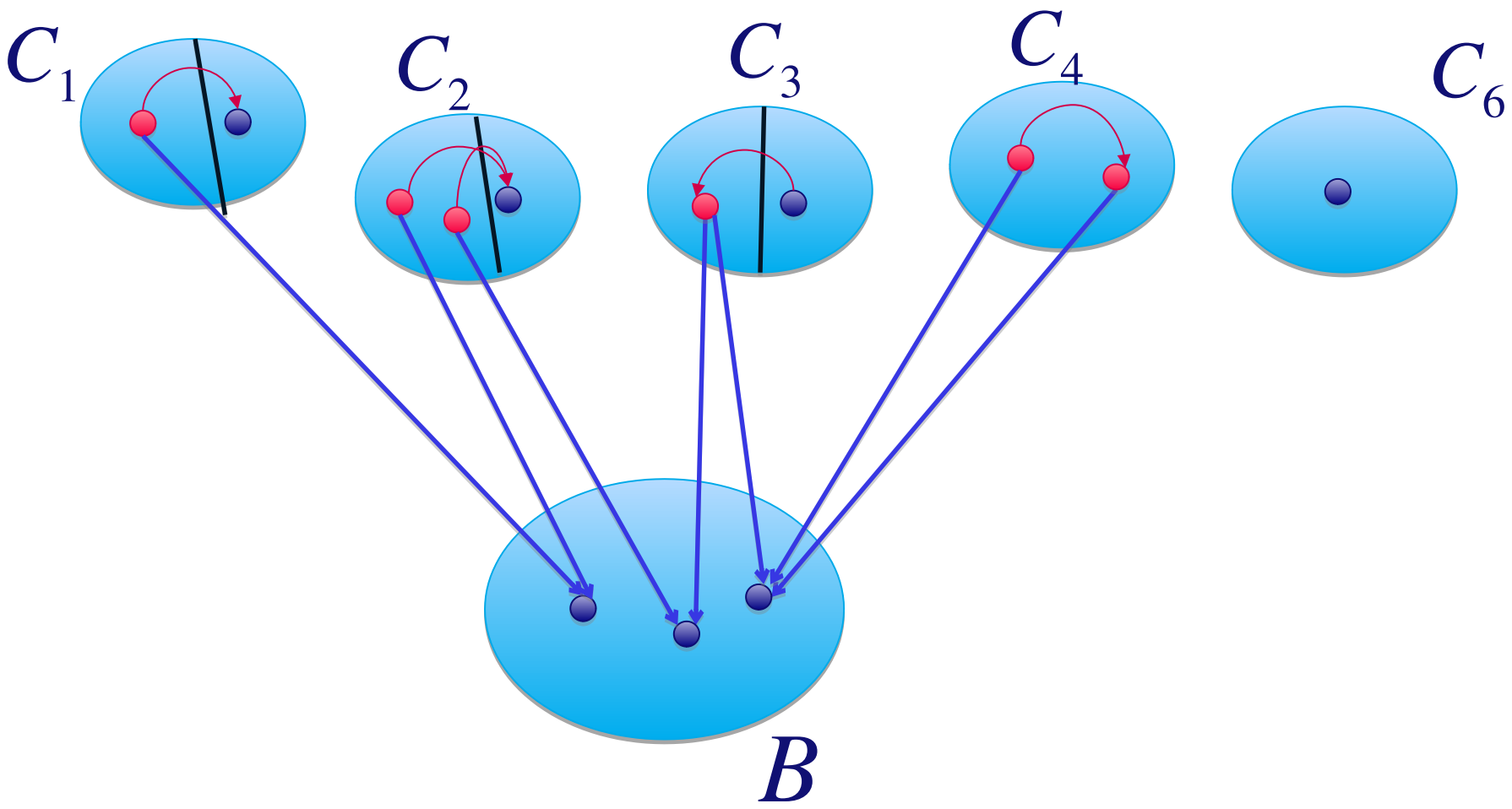
Requires $\Omega(n)$ time.

Parallel bisimulation



Split all blocks by B by one processor per transition.

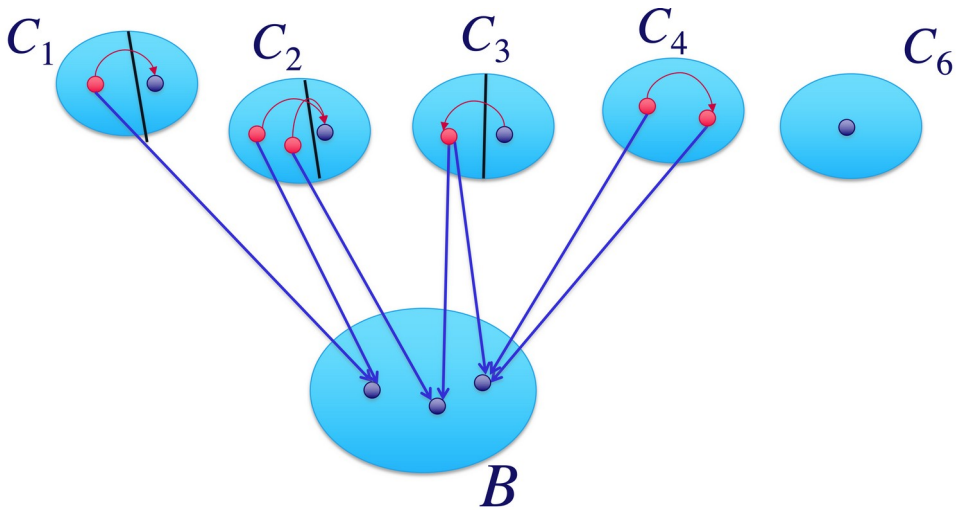
Parallel bisimulation



Splitting: Check whether the marked state has the same mark as the representative of the block. If yes, that is the representative of the new block. If no propose the marked state as the representative. One state wins non-deterministically, and is chosen.

Open question

n: number of states of an LTS
m: number of transitions of an LTS



Bisimulation requires $\Omega(m \log n)$ work. And in parallel $\Omega(n)$ time.

Open question:
Can we find a parallel optimal algorithm requiring $O(n)$ time on appr. $m/n \log n$ processors.

One would need appr. 30 processors for a typical large LTS.