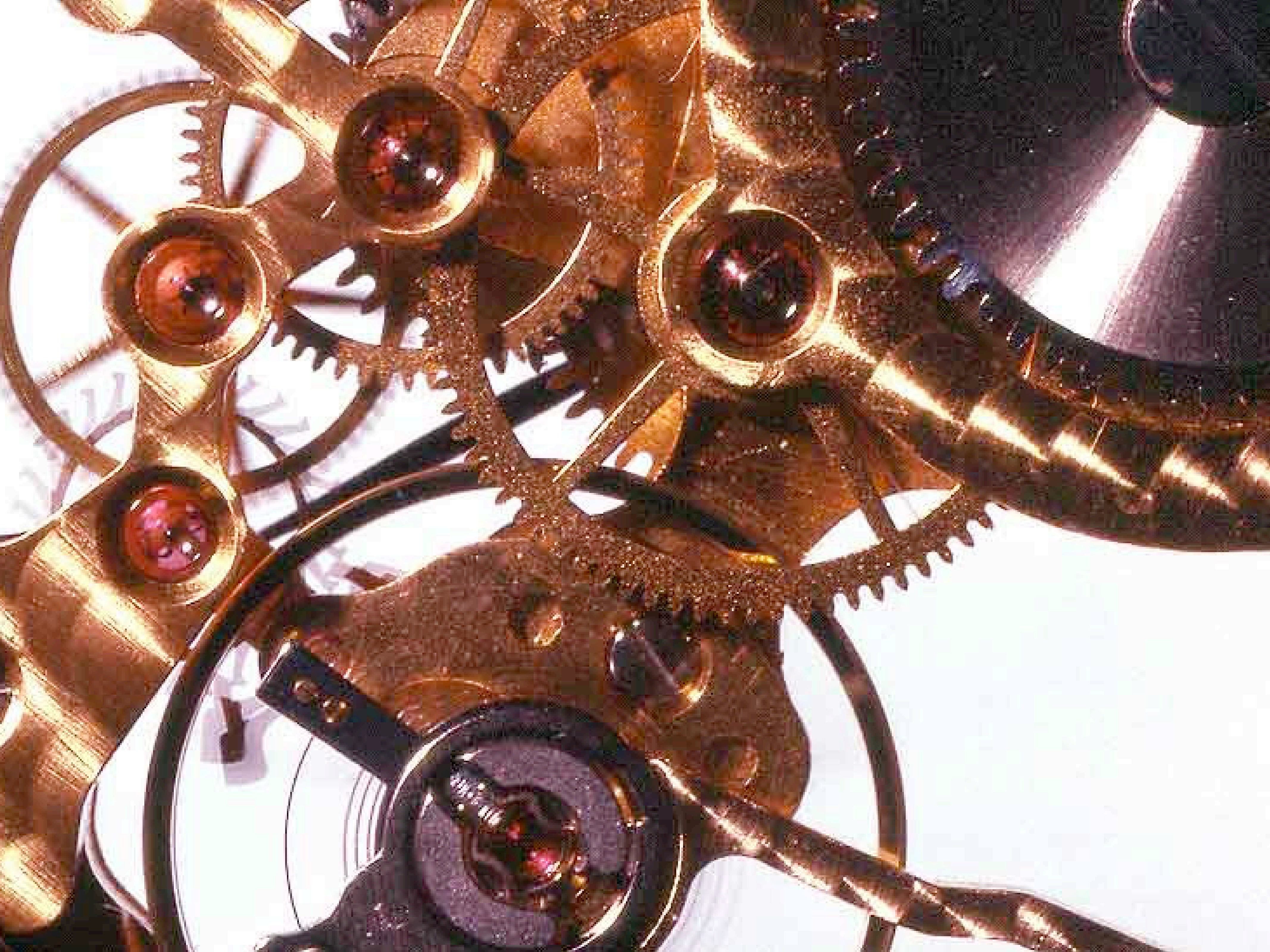


some remarks on

definability of process graphs

clemens grabmayer jan willem klop bas luttik

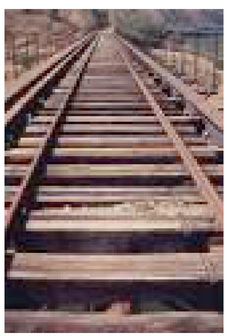
2006
concur
- bonn, germany -

Dramatis Personae

STACK

BAG

QUEUE



RAILS

KITE

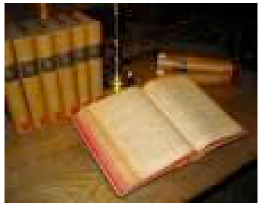
BUTTERFLY

TEMPLE

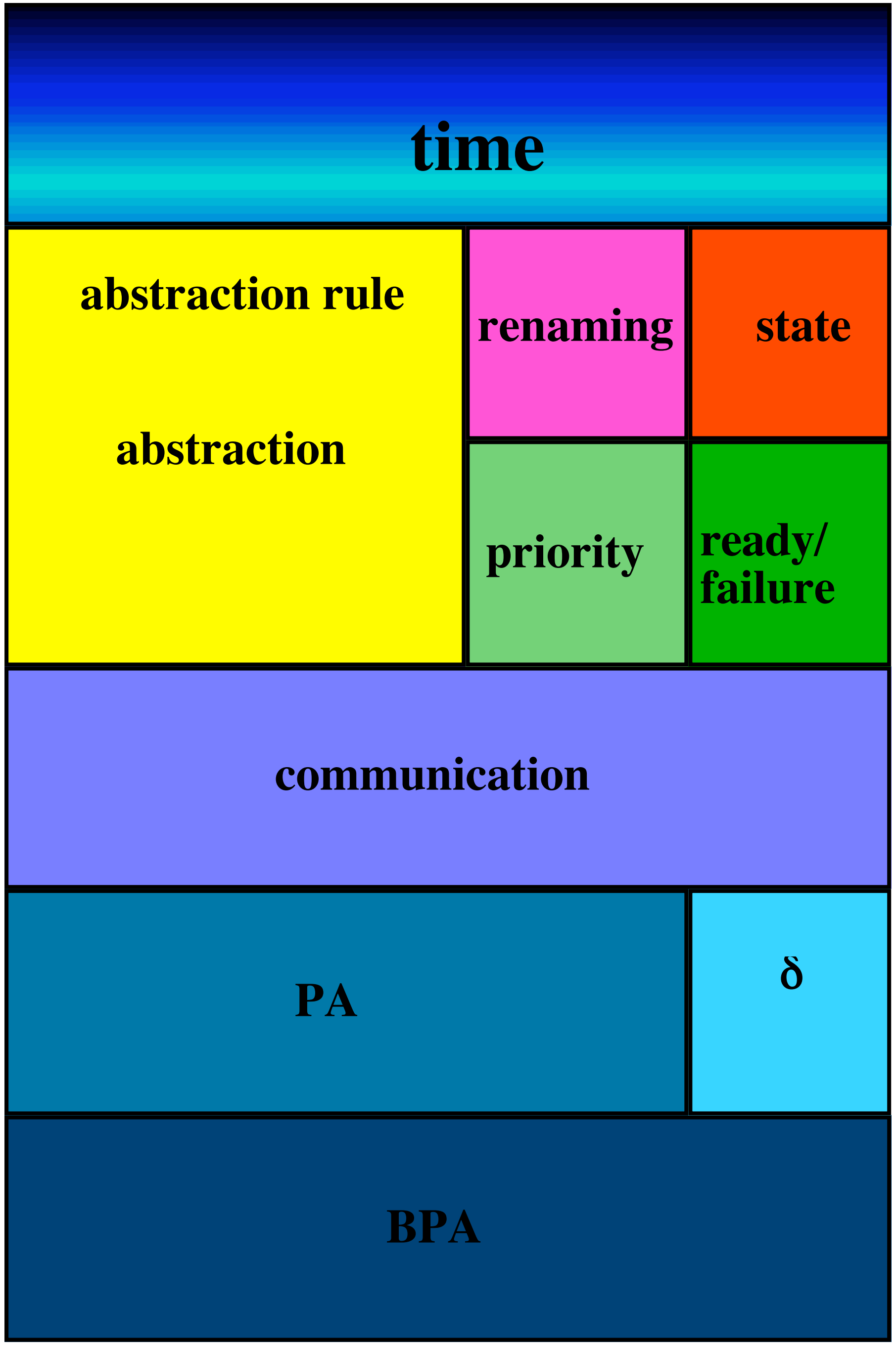
TOWER

TRIANGLE

Contents



1. *Some old definability results*
2. *Process graph dictionary*
3. *BPA graphs*
4. *Density and connectivity*
5. *Non-definability conclusions*
6. *BPP*



Modular structure of ACP

$$
\begin{aligned}
x+y &= y+x \\
x+(y+z) &= (x+y)+z \\
x+x &= x \\
(x+y)\cdot z &= x\cdot z+y\cdot z \\
(x\cdot y)\cdot z &= x\cdot(y\cdot z)
\end{aligned}
$$

Table 1: BPA (Basic Process Algebra)

$S_\lambda = push(0).S_0 + push(1).S_1$

$S_{d\sigma} = push(0).S_{0d\sigma} + push(1).S_{1d\sigma} + pop(d).S_\sigma$

(voor d = 0 of d = 1, en elke string σ)

infinite guarded BPA specification

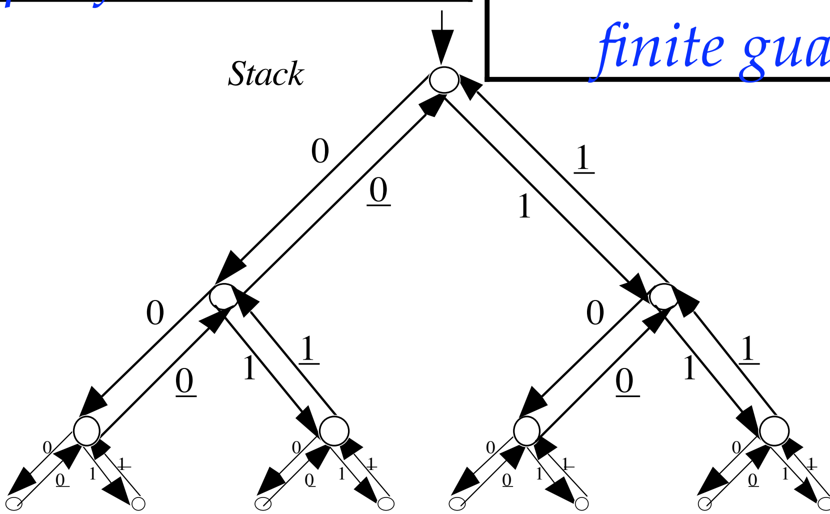
$S = T.S$

$T = push(0).T_0 + push(1).T_1$

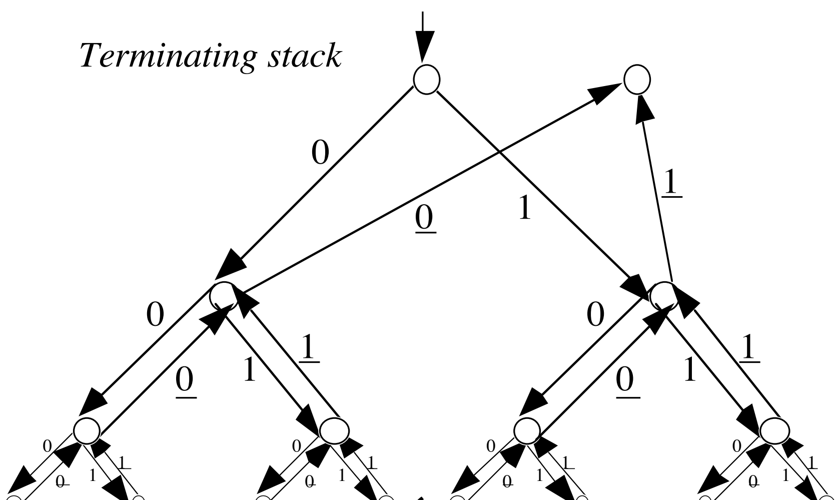
$T_0 = pop(0) + T.T_0$

$T_1 = pop(1) + T.T_1$

finite guarded BPA specification



Stack needs a terminating Stack for its definition



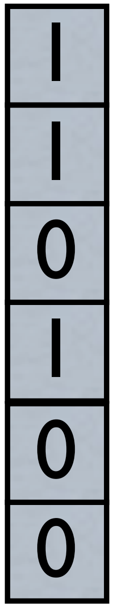


Table 1. BPA (Basic Process Algebra), left, and PA (Process Algebra), on the right

$$
\begin{aligned}
x + y &= y + x \\
x + (y + z) &= (x + y) + z \\
x + x &= x \\
(x + y) \cdot z &= x \cdot z + y \cdot z \\
(x \cdot y) \cdot z &= x \cdot (y \cdot z)
\end{aligned}
$$

$$
\begin{aligned}
x + y &= y + x \\
x + (y + z) &= (x + y) + z \\
x + x &= x \\
(x + y) \cdot z &= x \cdot z + y \cdot z \\
(x \cdot y) \cdot z &= x \cdot (y \cdot z) \\
x \parallel y &= x \mathbin{\parallel\!\!\!\lfloor} y + y \mathbin{\parallel\!\!\!\lfloor} x \\
\mathrm{a} \mathbin{\parallel\!\!\!\lfloor} x &= \mathrm{a} \cdot x \\
\mathrm{a} \cdot x \mathbin{\parallel\!\!\!\lfloor} y &= \mathrm{a} \cdot (x \parallel y) \\
(x + y) \mathbin{\parallel\!\!\!\lfloor} z &= x \mathbin{\parallel\!\!\!\lfloor} z + y \mathbin{\parallel\!\!\!\lfloor} z
\end{aligned}
$$

The left merge is an auxiliary operator necessary for a finite axiomatization of merge.

Every process which is recursively defined in PA *and has an infinite trace, has an eventually periodic trace.*

PA has unique prime decomposition:

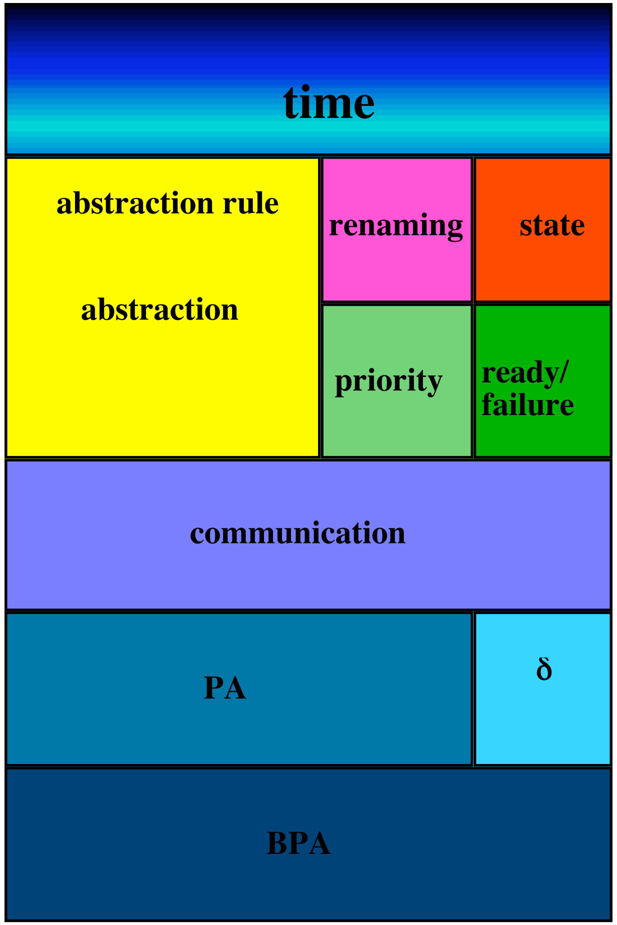
$p = p_1 \parallel \ldots \parallel p_n$

unique modulo permutation of 'parallel primes'

$$
\begin{array}{llcl}
\text{RN1} & \rho_f(v) & = & f(v) \\
\text{RN2} & \rho_f(\delta) & = & \delta \\
\text{RN3} & \rho_f(x+y) & = & \rho_f(x) + \rho_f(y) \\
\text{RN4} & \rho_f(x \cdot y) & = & \rho_f(x) \cdot \rho_f(y)
\end{array}
$$

renaming axioms

renaming can be performed by communication



(A1) $x + y = y + x$
(A2) $x + (y + z) = (x + y) + z$
(A3) $x + x = x$
(A4) $(x + y) . z = x.z + y.z$
(A5) $x.(y.z) = (x.y).z$

basic process algebra

(A6) $x + \delta = x$
(A7) $\delta.x = \delta$

deadlock

(CM1) $x \parallel y = x \llcorner y + y \llcorner x + x|y$
(CM2) $a \llcorner x = a.x$
(CM3) $ax \llcorner y = a(x \parallel y)$
(CM4) $(x + y) \llcorner z = x \llcorner z + y \llcorner z$
(CM5) $ax|b = (a|b)x$
(CM6) $a|bx = (a|b)x$
(CM7) $ax|by = (a|b)(x \parallel y)$
(CM8) $(x + y)|z = x|z + y|z$
(CM9) $x|(y + z) = x|y + x|z$

communication merge

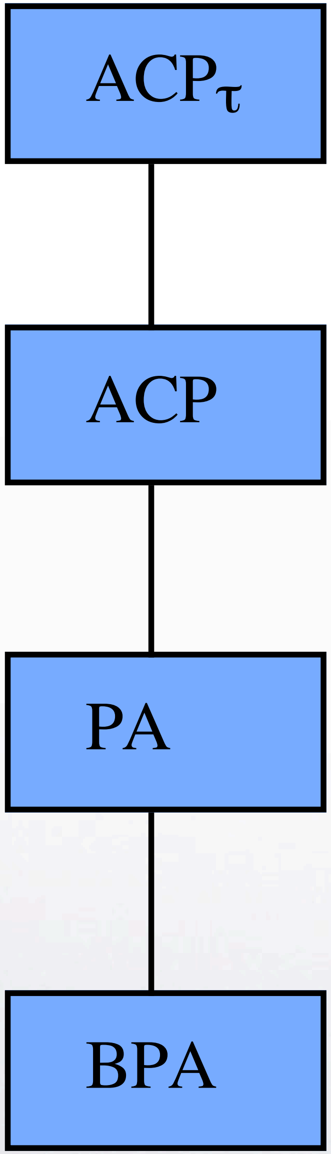
(C1) $a|b = \gamma(a, b)$ if $\gamma(a,b)$ defined,
else δ
(C2) $(a|b)|c = a|(b|c)$
(C3) $\delta|a = \delta$

communication on atoms

(D1) $\partial_H(a) = a$ if $a \notin H$
(D2) $\partial_H(a) = \delta$ if $a \in H$
(D3) $\partial_H(x + y) = \partial_H(x) + \partial_H(y)$
(D4) $\partial_H(x . y) = \partial_H(x) . \partial_H(y)$

encapsulation operator

ACP

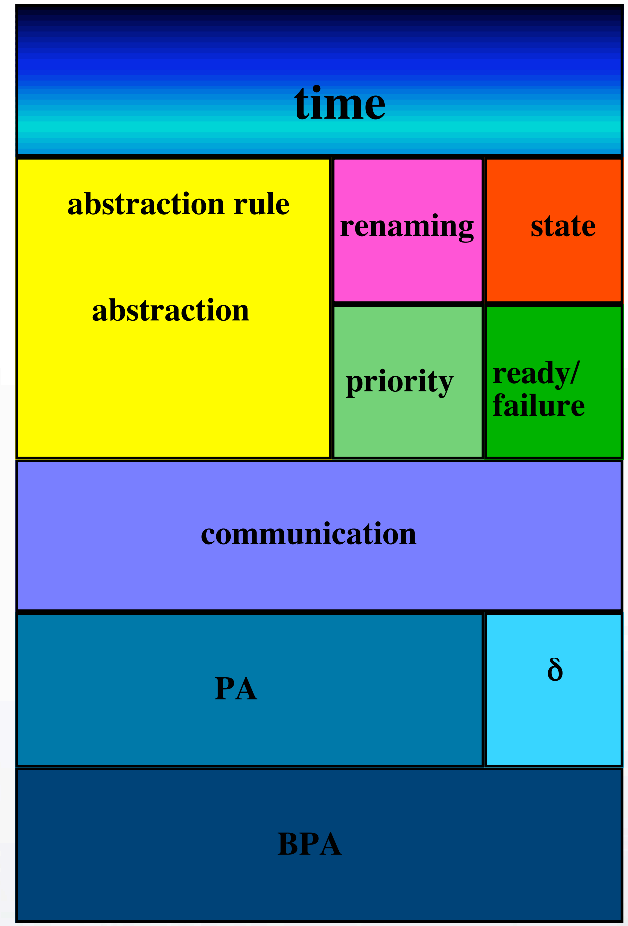


Turing machine

Queue

Bag

Stack



Modular structure of ACP

Difference in expressiveness between PA and ACP

Thue-Morse sequence:

M = 1001 0110 01101001 0110100110010110 ...
M = 1001 0110 01101001 0110100110010110 ...

M = zip M inv(M)
M = 1:0:zip(tail(M), inv(tail(M))

M can be defined in ACP with renaming, or in ACP with ternary communication. With binary communication?

M cannot be defined in PA, since its one single trace is not eventually periodic.

The process BAG *cannot be defined in BPA*.

Bergstra-Tiuryn:

Queue cannot be defined in ACP with handshaking communication

- but it can in ACP with renaming,

- or in ACP with ternary communication

QUEUE

Table 4. Queue, infinite BPA-specification

$$Q = Q_\lambda = \sum_{d \in D} r_1(d) \cdot Q_d$$
$$Q_{\sigma d} = s_2(d) \cdot Q_\sigma + \sum_{e \in D} r_1(e) \cdot Q_{e \sigma d}$$
(for $d \in D$, and $\sigma \in D^*$)

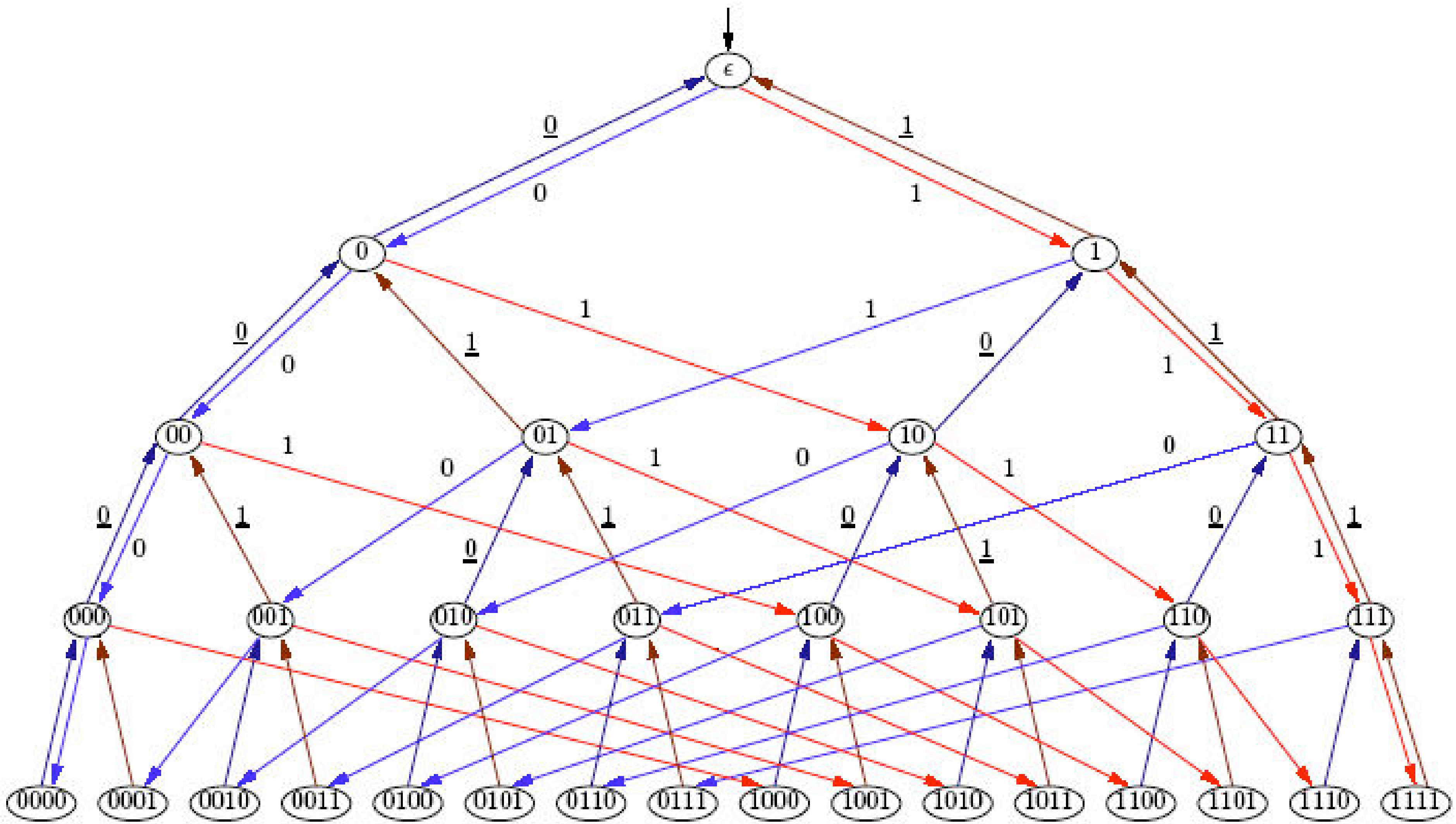


Fig. 9. The canonical process graph QUEUE of Queue

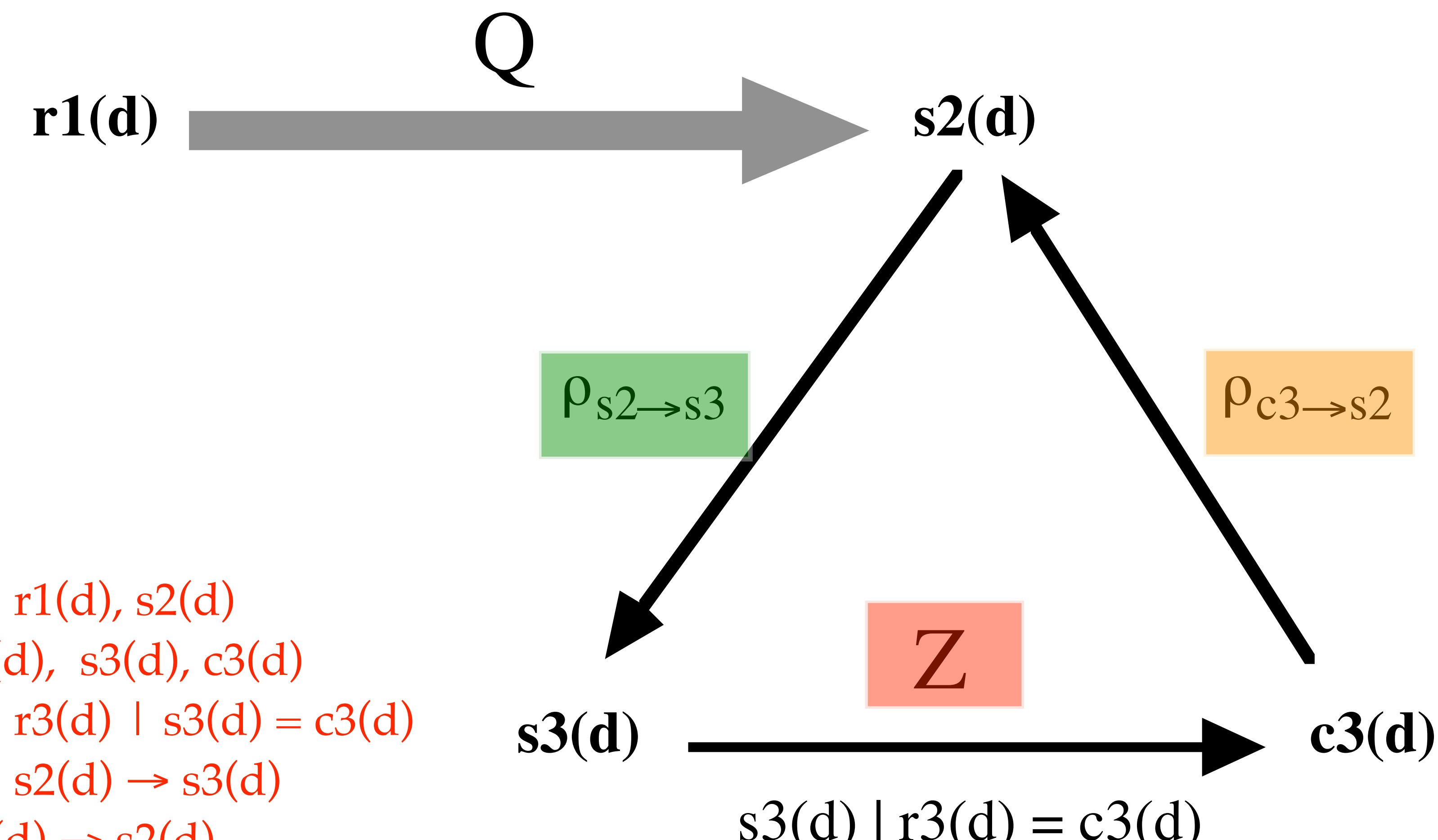
Table 5. Queue, finite ACP-specification with renaming

$$Q = \sum_{d \in D} r_1(d)(\rho_{c_3 \to s_2} \circ \partial_H)(\rho_{s_2 \to s_3}(Q) \parallel s_2(d) \cdot Z)$$
$$Z = \sum_{d \in D} r_3(d) \cdot Z$$

guard

$$Q = \Sigma_{d \in D}\, r1(d)(\rho_{c3 \rightarrow s2} \circ \partial_H)(\rho_{s2 \rightarrow s3}(Q) \| s2(d).Z)$$

$$Z = \Sigma_{d \in D}\, r3(d).Z$$



actions: r1(d), s2(d)

auxiliary actions: r3(d), s3(d), c3(d)

communication: r3(d) | s3(d) = c3(d)

$\rho_{s2 \rightarrow s3}$ renaming: s2(d) → s3(d)

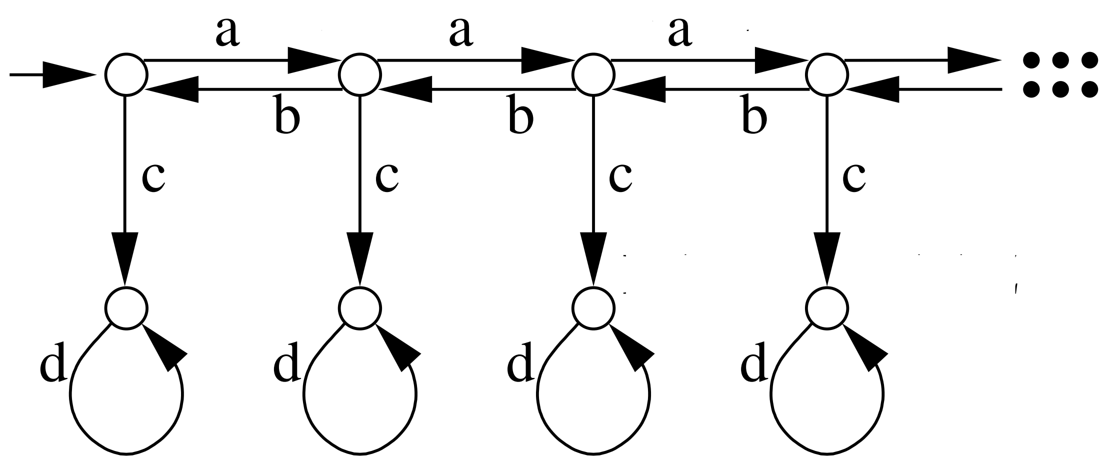
$\rho_{c3 \rightarrow s2}$ renaming: c3(d) → s2(d)

encapsulation: $H = \{s3(d), r3(d) \mid d \in D\}$

Contents

1. *Some old definability results*
2. *Process graph dictionary*
3. *BPA graphs*
4. *Density and connectivity*
5. *Non-definability conclusions*
6. *BPP*

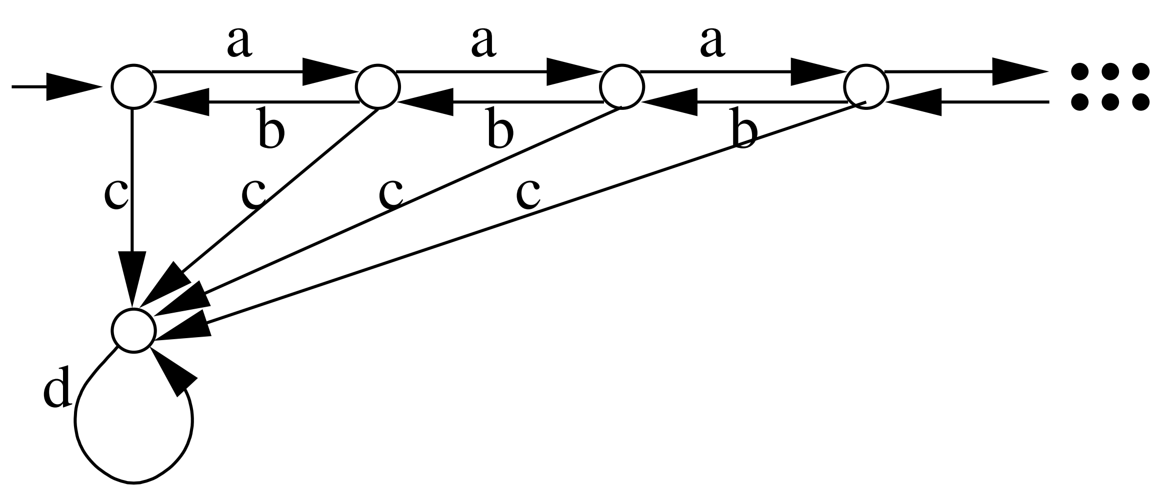
finite degree



yes

bisimilar to

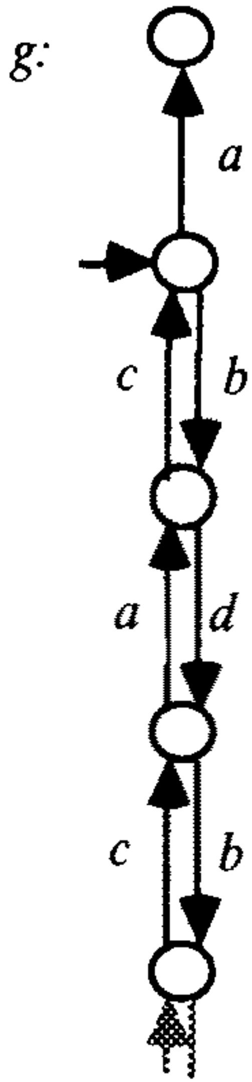
$\underline{\leftrightarrow}$



no

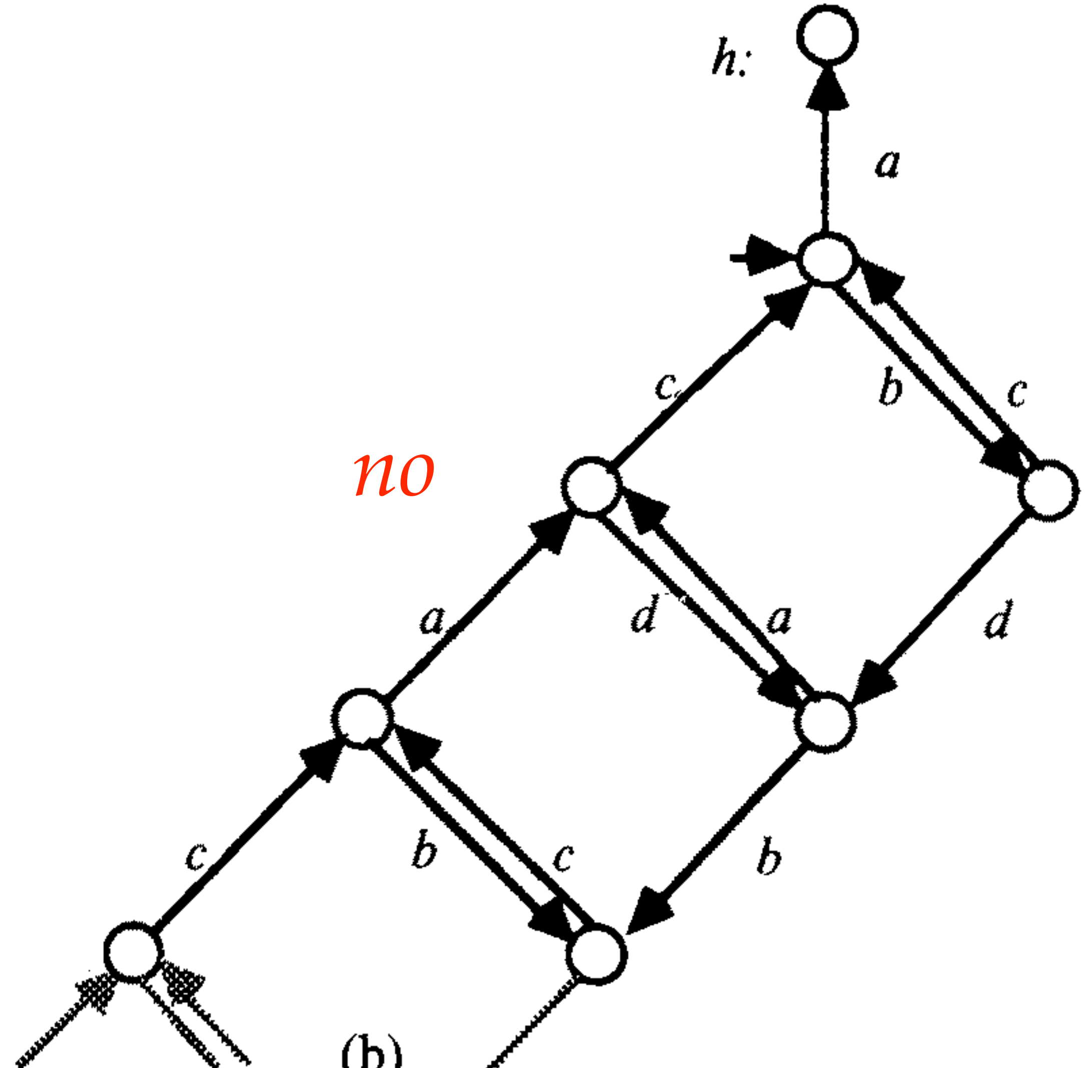
canonical

yes



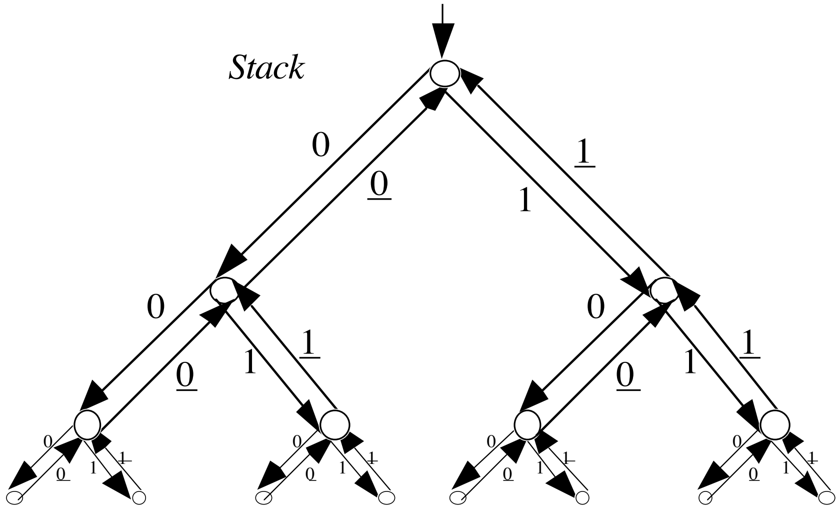
(a)

no

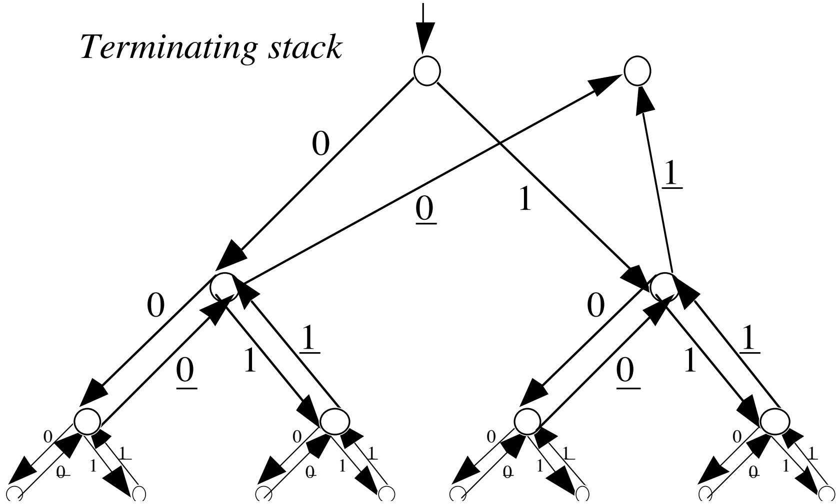


(b)

normed

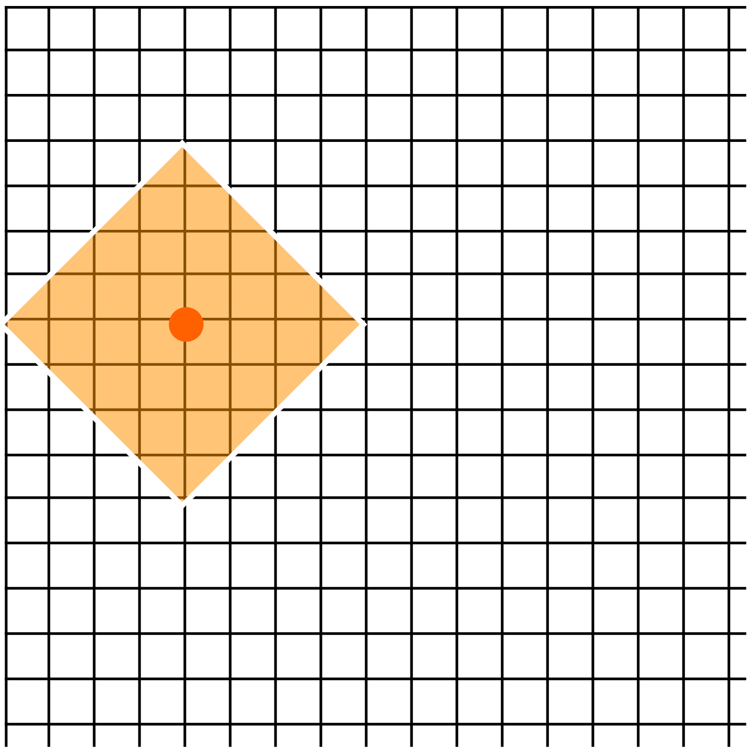


no

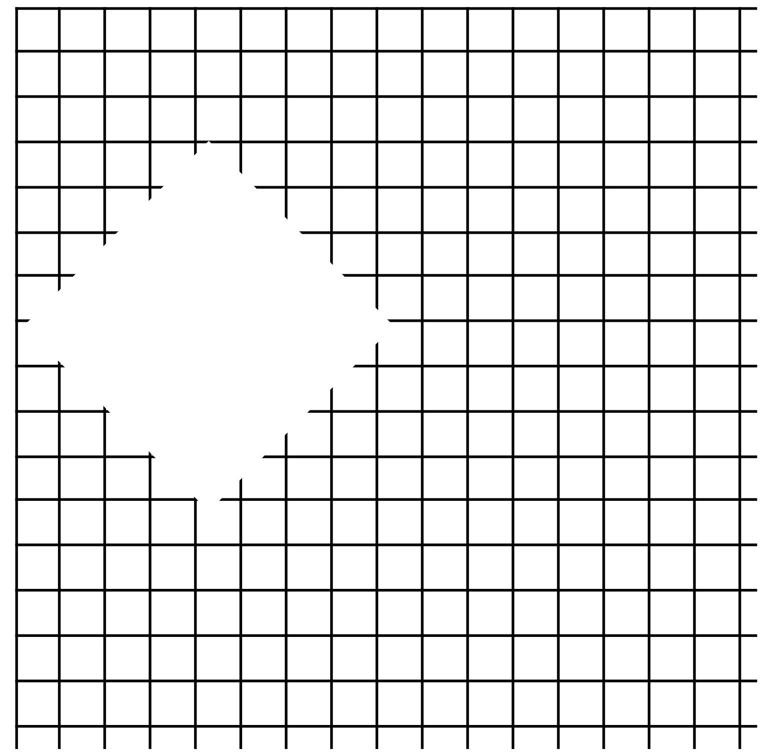


yes

In, Out

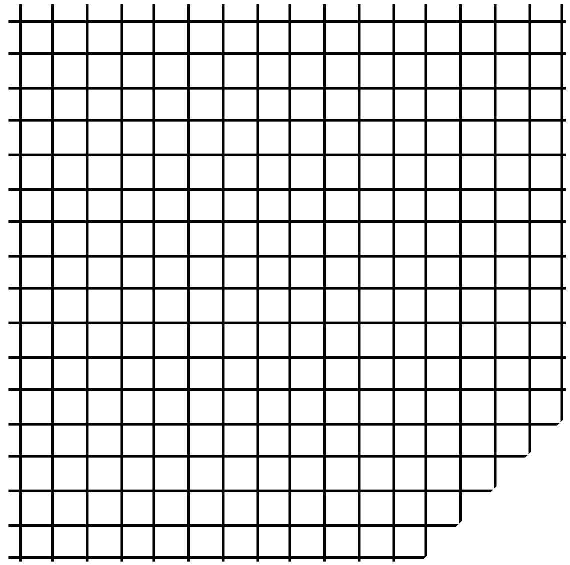


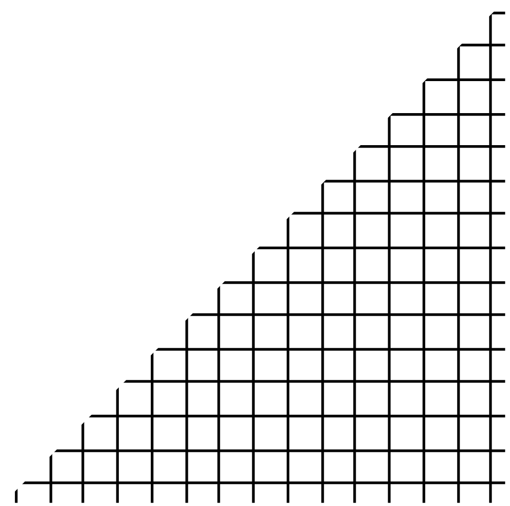
In(s, 4)



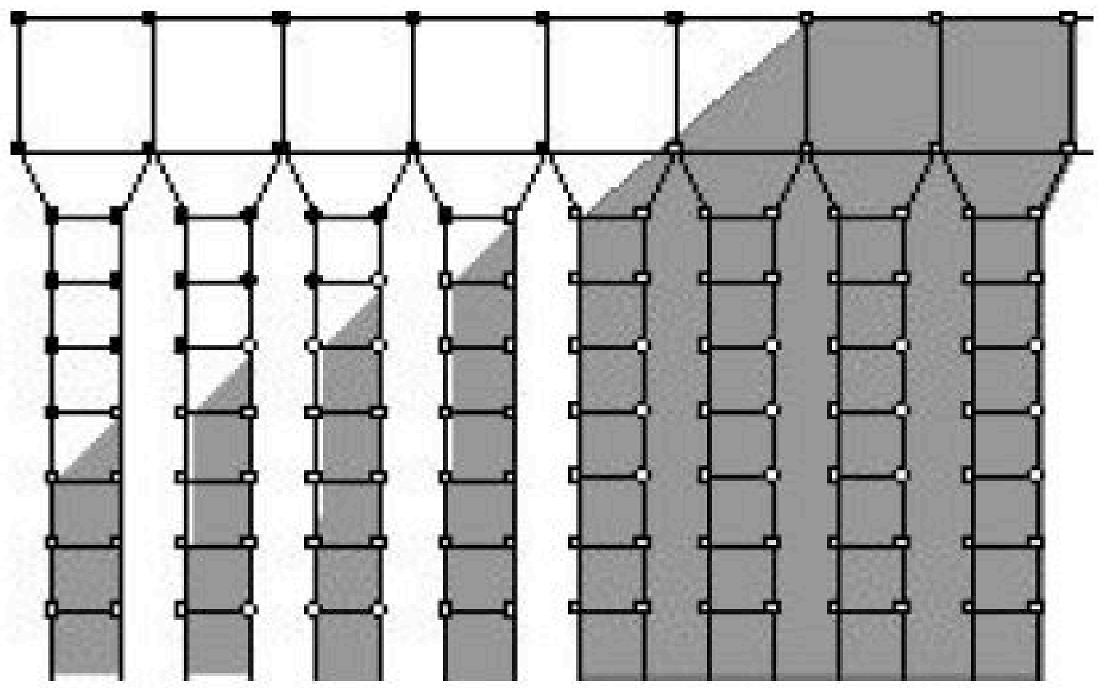
Out(s, 4)

infinite connected component (icc)





Out(r, 6) is a not connected graph with 5 icc's



Contents

why is BPA interesting?

1. *Some old defina*
2. *Process graph di*
3. *BPA graphs*
4. *Density and connectivity*
5. *Non-definability conclusions*
6. *BPP*

$$
\begin{aligned}
x+y &= y+x \\
x+(y+z) &= (x+y)+z \\
x+x &= x \\
(x+y)\cdot z &= x\cdot z+y\cdot z \\
(x\cdot y)\cdot z &= x\cdot(y\cdot z)
\end{aligned}
$$

context-free grammar in standard form (Greibach normal form)

$$S \rightarrow aB \mid bA$$

$$A \rightarrow a \mid aS \mid bAA$$

$$B \rightarrow b \mid bS \mid aBB$$

language equality undecidable

guarded nonlinear recursion system over BPA

$$S = aB + bA$$

$$A = a + aS + bAA$$

$$B = b + bS + aBB$$

process equality decidable

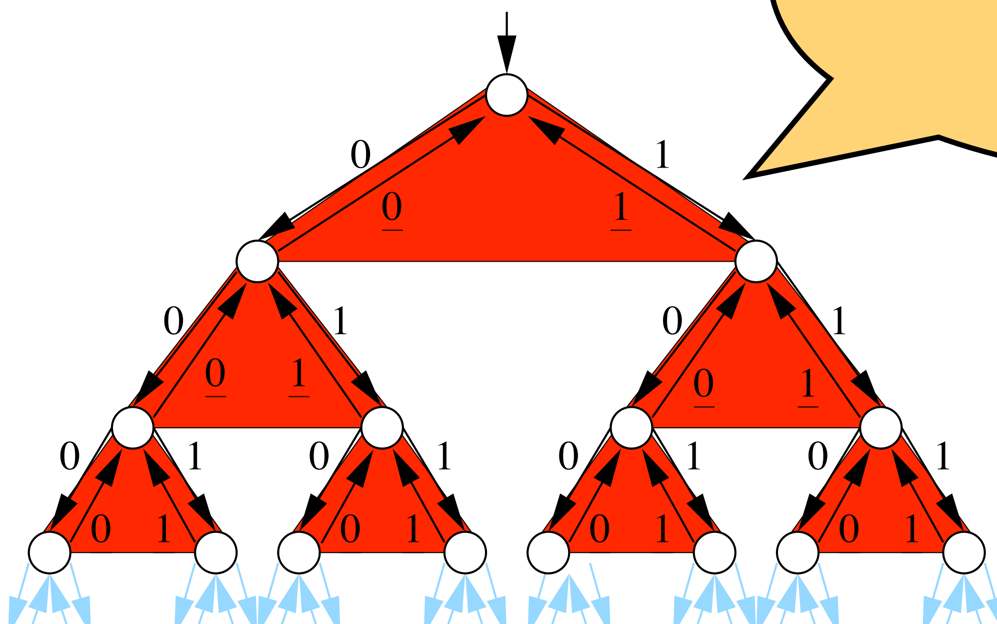
$$S_\lambda = 0 \cdot S_0 + 1 \cdot S_1$$
$$S_{d\sigma} = 0 \cdot S_{0d\sigma} + 1 \cdot S_{1d\sigma} + \underline{d} \cdot S_\sigma$$
(for $d = 0$ or $d = 1$, and any string σ)

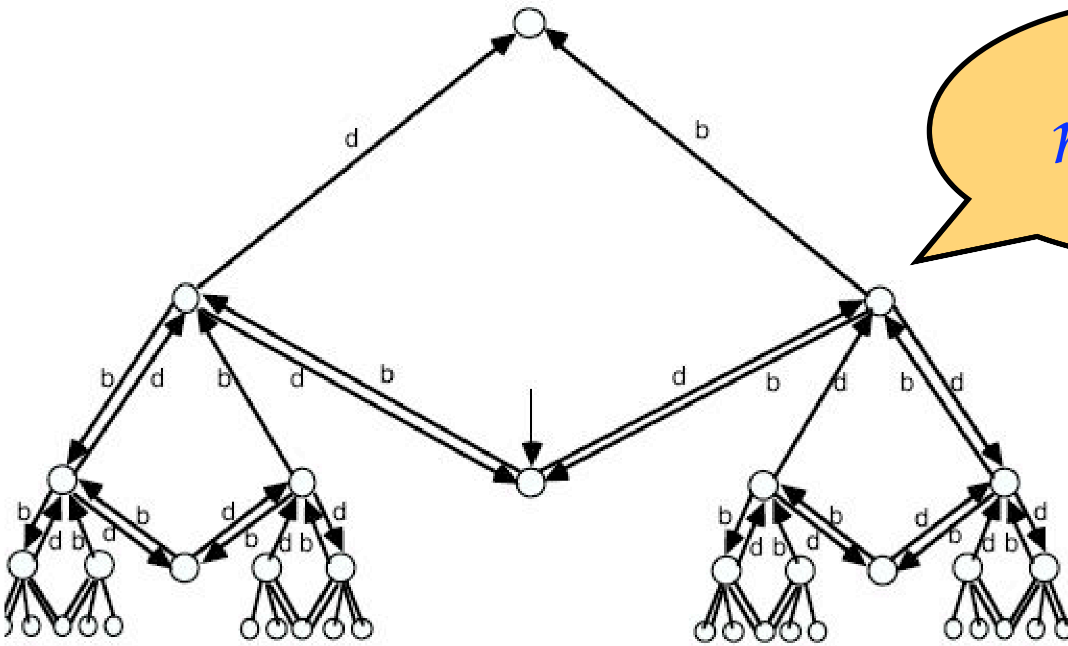
$$S = T \cdot S$$
$$T = 0 \cdot T_0 + 1 \cdot T_1$$
$$T_0 = \underline{0} + T \cdot T_0$$
$$T_1 = \underline{1} + T \cdot T_1$$

Table 2: Stack, an infinite linear and a finite no

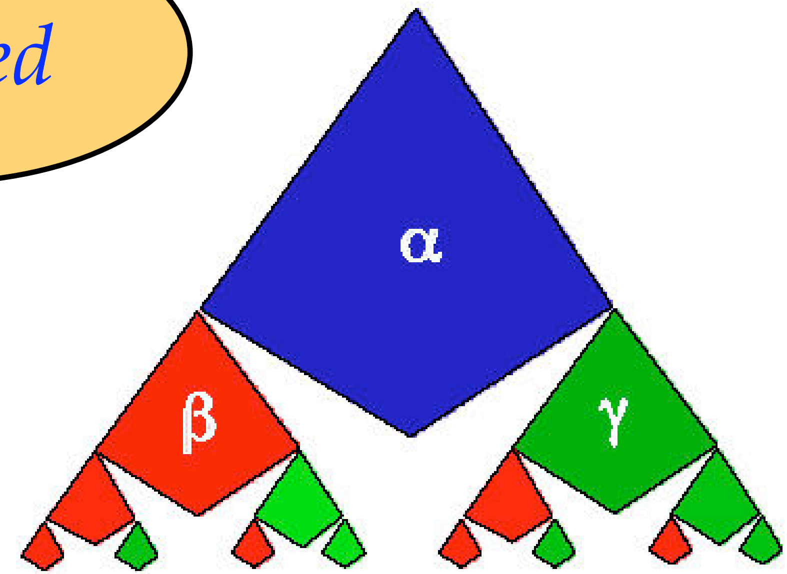
periodicity!

STACK





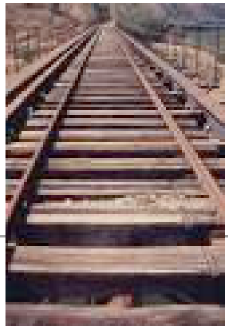
normed



$$X = bY + dZ$$
$$Y = b + bX + dYY$$
$$Z = d + dX + bZZ.$$

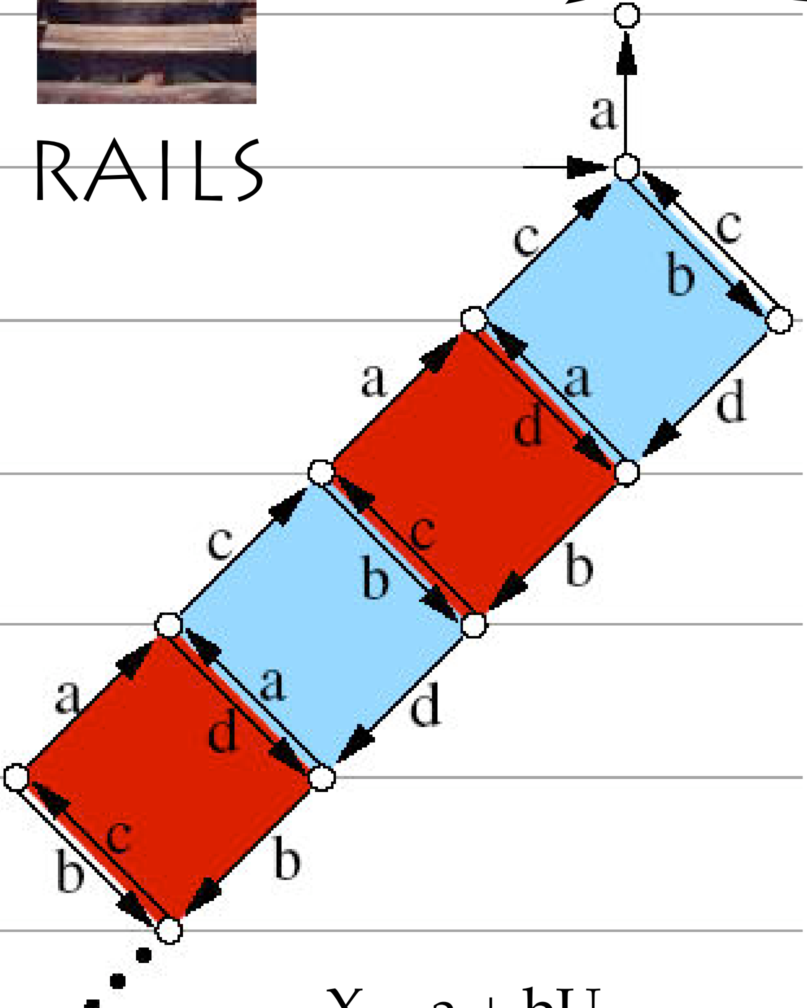
KITE

context free language of words with just as many b's as d's



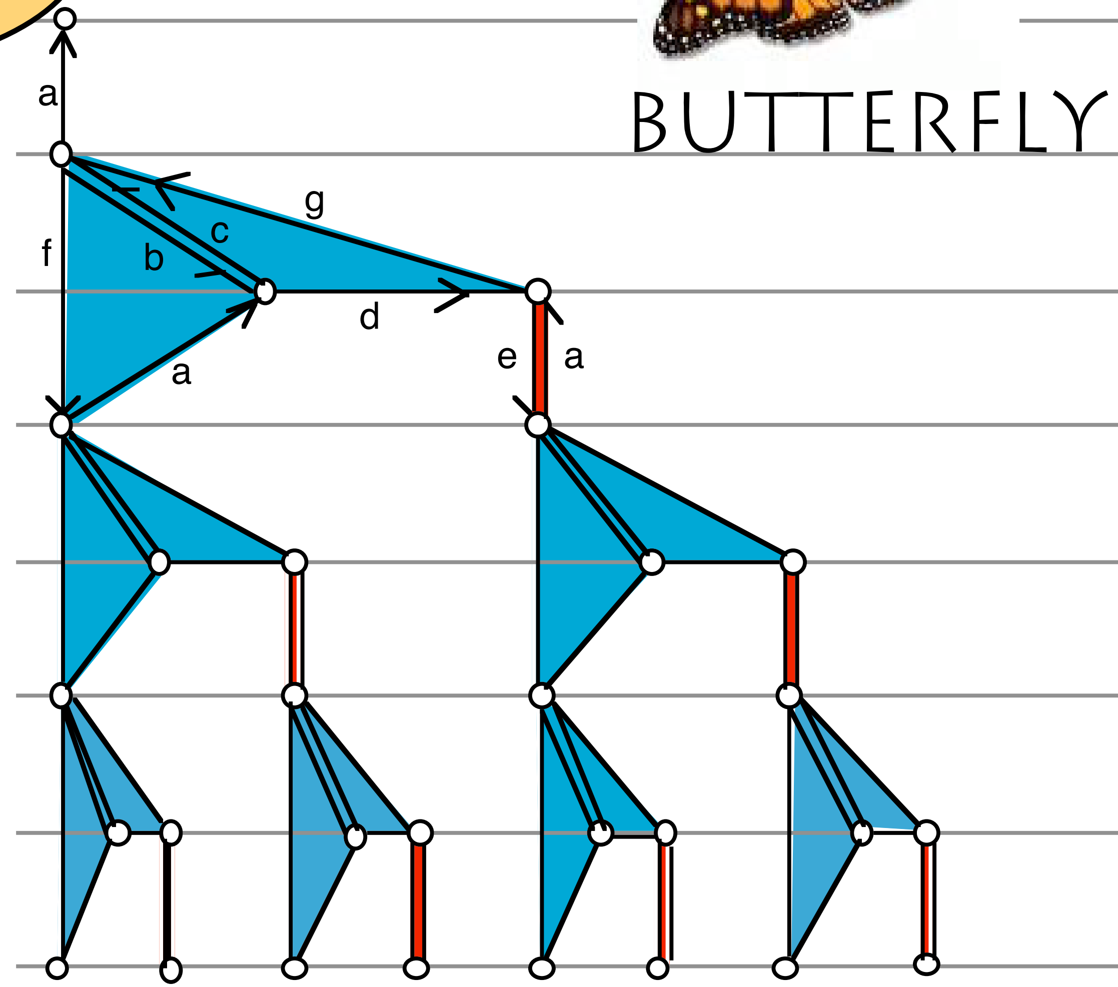
RAILS

normed



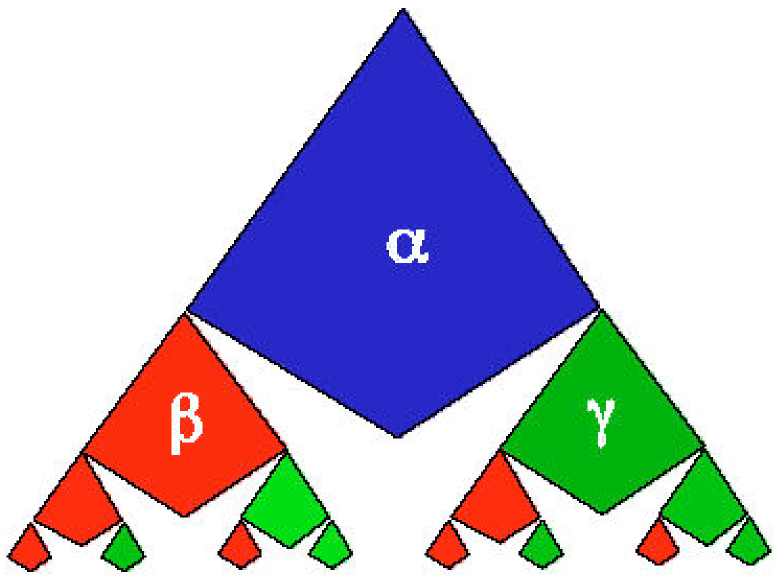
$$X = a + bU$$
$$U = cX + dZX$$
$$Y = c + dZ$$
$$Z = aY + bUY$$

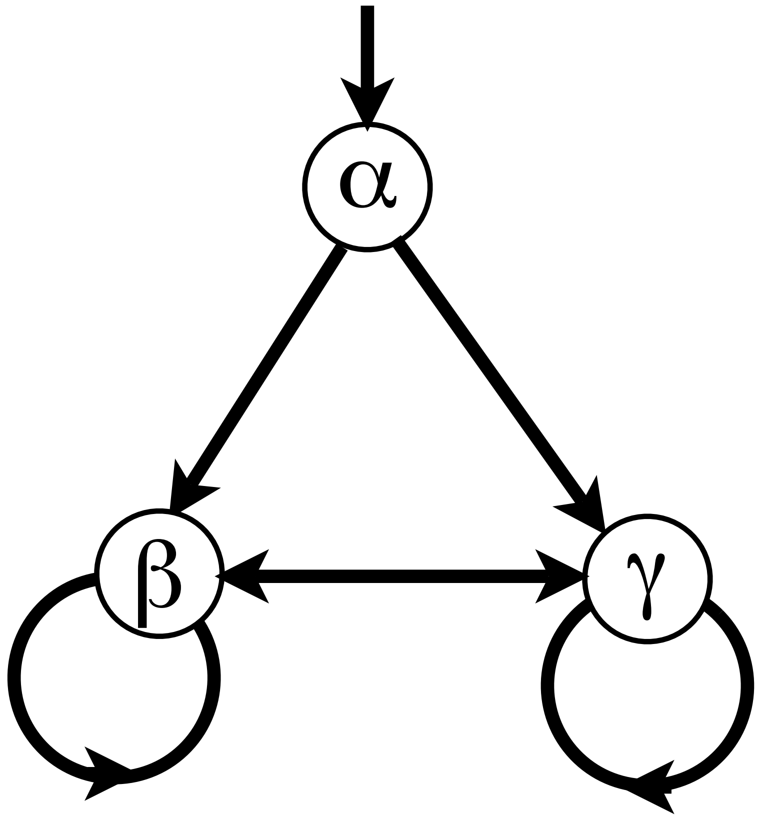
BUTTERFLY



$$X = a + bY + fXY$$
$$Y = cX + dZ$$
$$Z = gX + eXZ.$$

fragment structure of KITES





TEMPLE

$$\langle \mathrm{A} \mid \mathrm{A} = \mathrm{a} + \mathrm{b\,AB},\ \mathrm{B} = \mathrm{a} + \mathrm{b\,BC},\ \mathrm{C} = \mathrm{a} \rangle$$

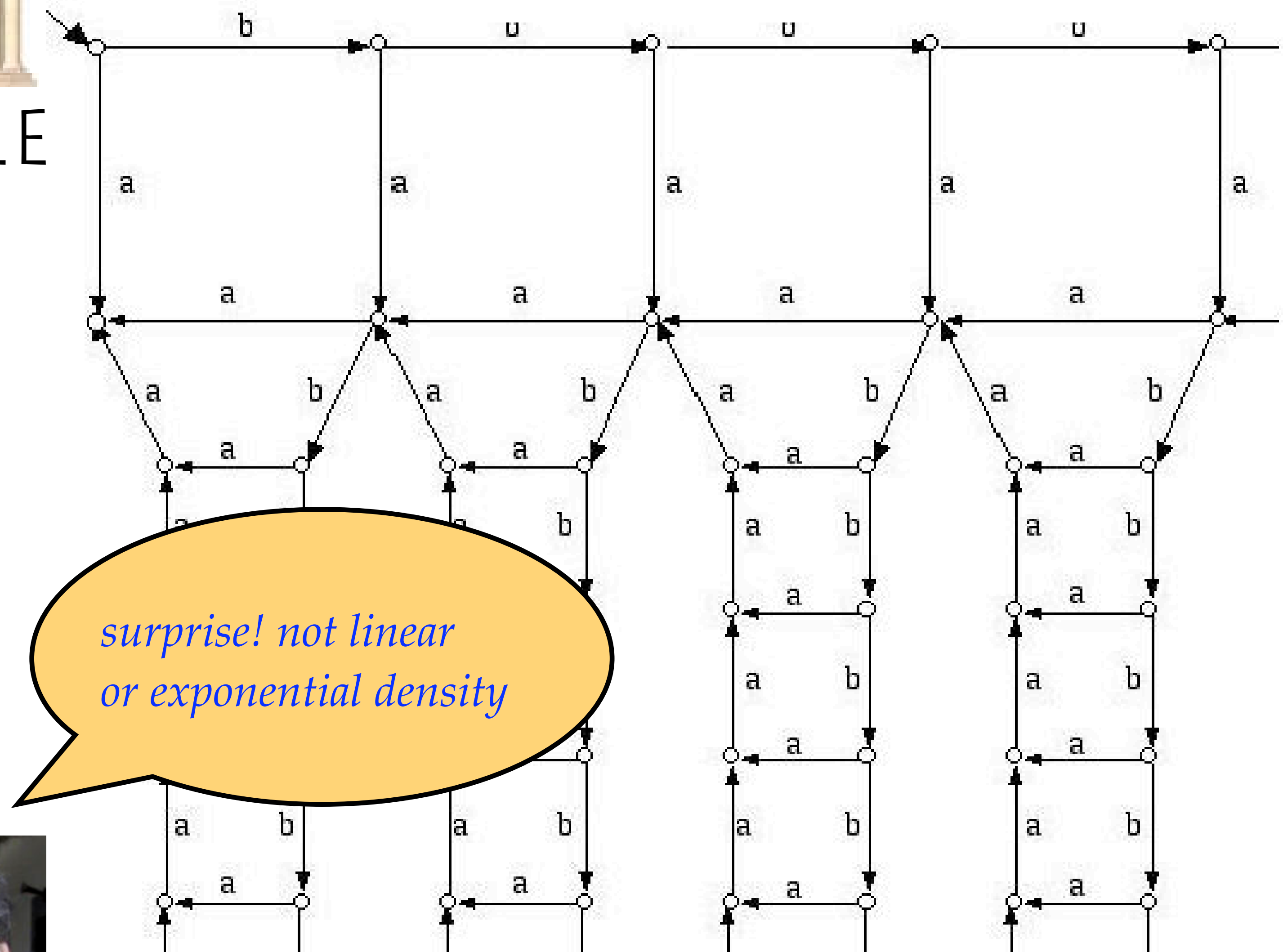


Fig. 2. The labeled transition graph TEMPLE

fragment structure of TEMPLE

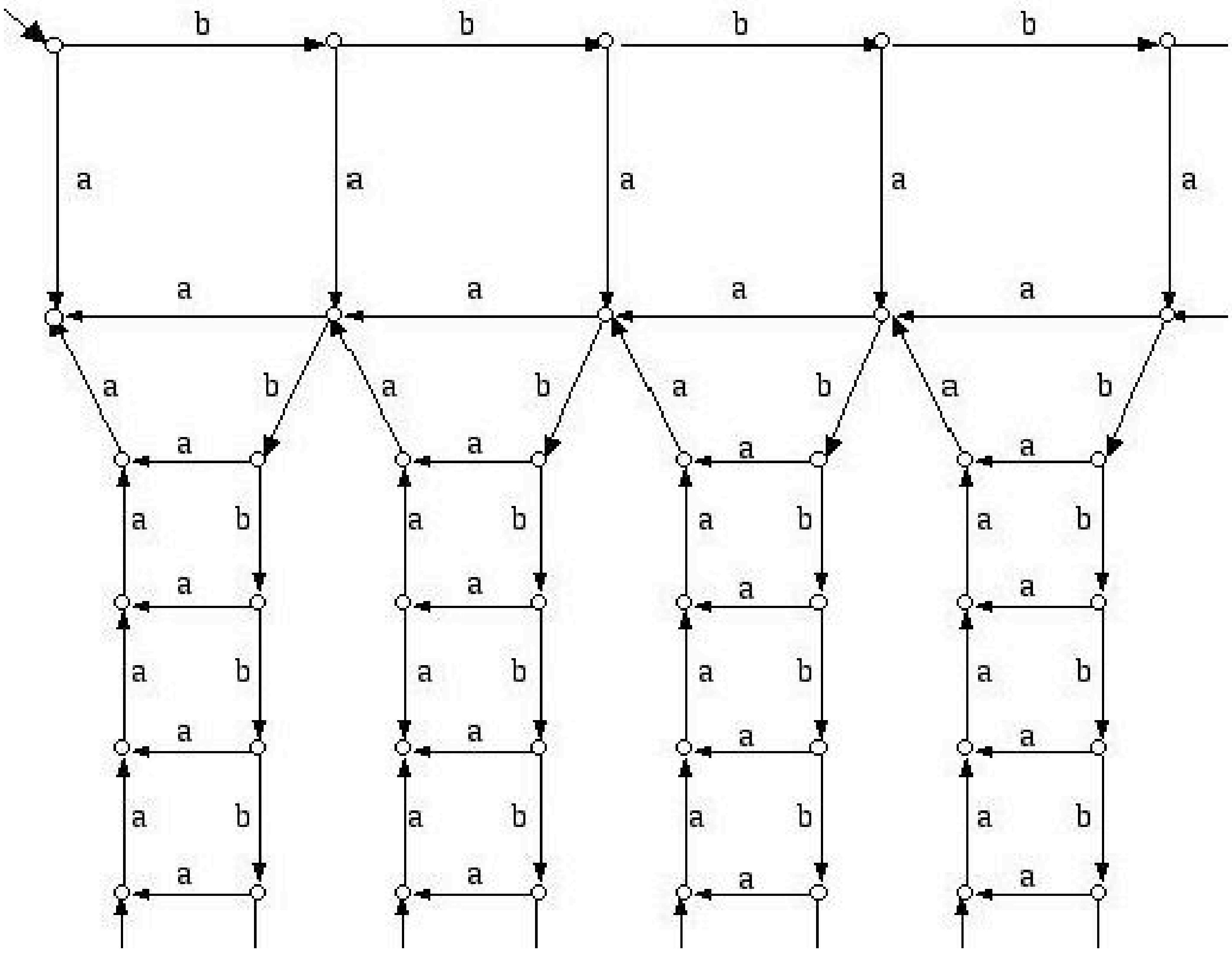
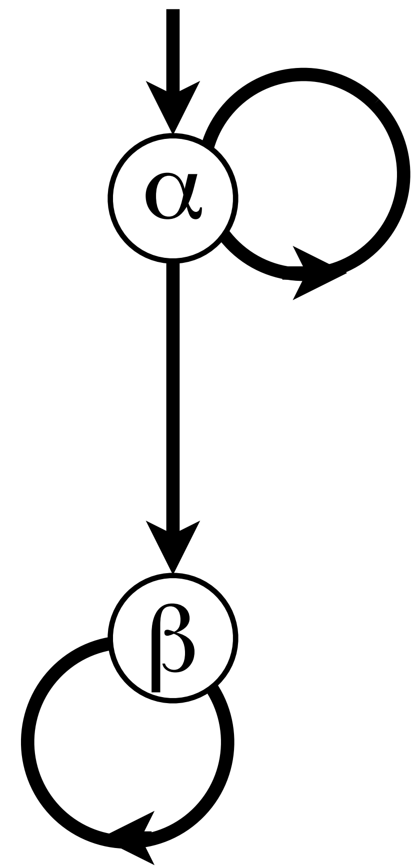


Fig. 2. The labeled transition graph TEMPLE



equivalent notations

system of BPA equations

$$\langle \mathrm{A} \mid \mathrm{A} = \mathrm{a} + \mathrm{b}\,\mathrm{AB},\ \mathrm{B} = \mathrm{a} + \mathrm{b}\,\mathrm{BC},\ \mathrm{C} = \mathrm{a} \rangle$$

process rewrite system

$$R = \{A \xrightarrow{a} \lambda,\ A \xrightarrow{b} AB,\ B \xrightarrow{a} \lambda,\ B \xrightarrow{b} BC,\ C \xrightarrow{a} \lambda\}\ .$$

Theorem (BBK 1986)

(i) BPA graphs have a periodic decomposition

(ii) normed BPA graphs have decidable equality

Theorem (Hüttel, Stirling)

All BPA graphs have decidable equality

Theorem (Caucal 1990)

The class of normed BPA graphs is closed under minimization.

NB (BCS)

normed is necessary here

NB (Caucal)

the reverse of (i) fails: a graph with periodic decomposition need not be a BPA graph.

$$E = \{X = a + bU, U = cX + dZX, Y = c + dZ, Z = aY + bYU\}.$$

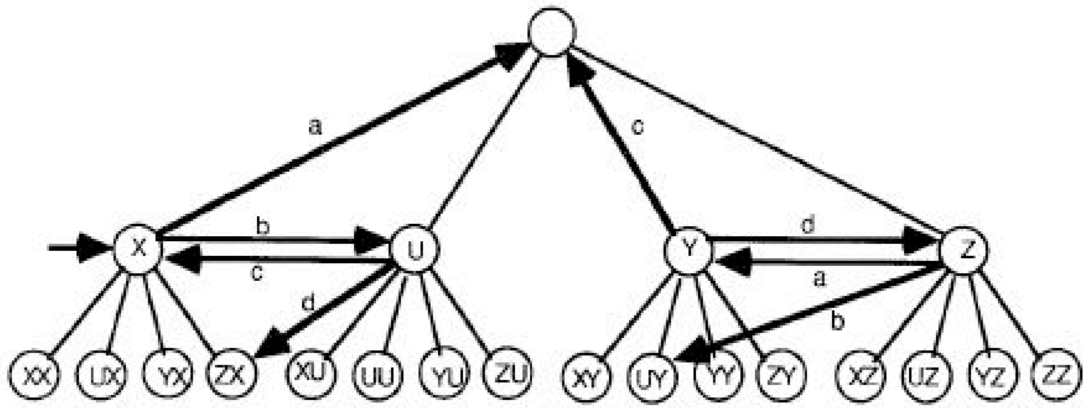


FIGURE 8

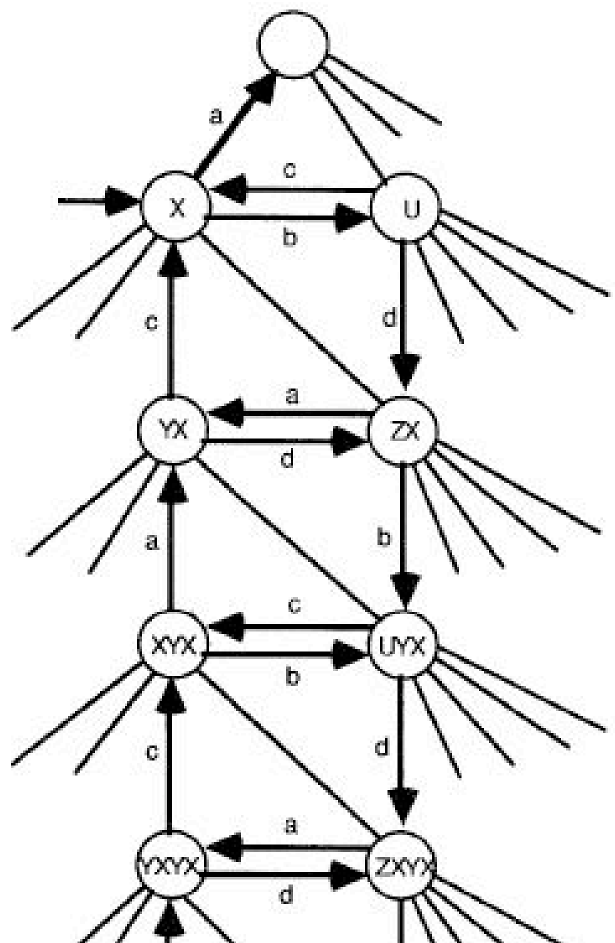


FIGURE 9

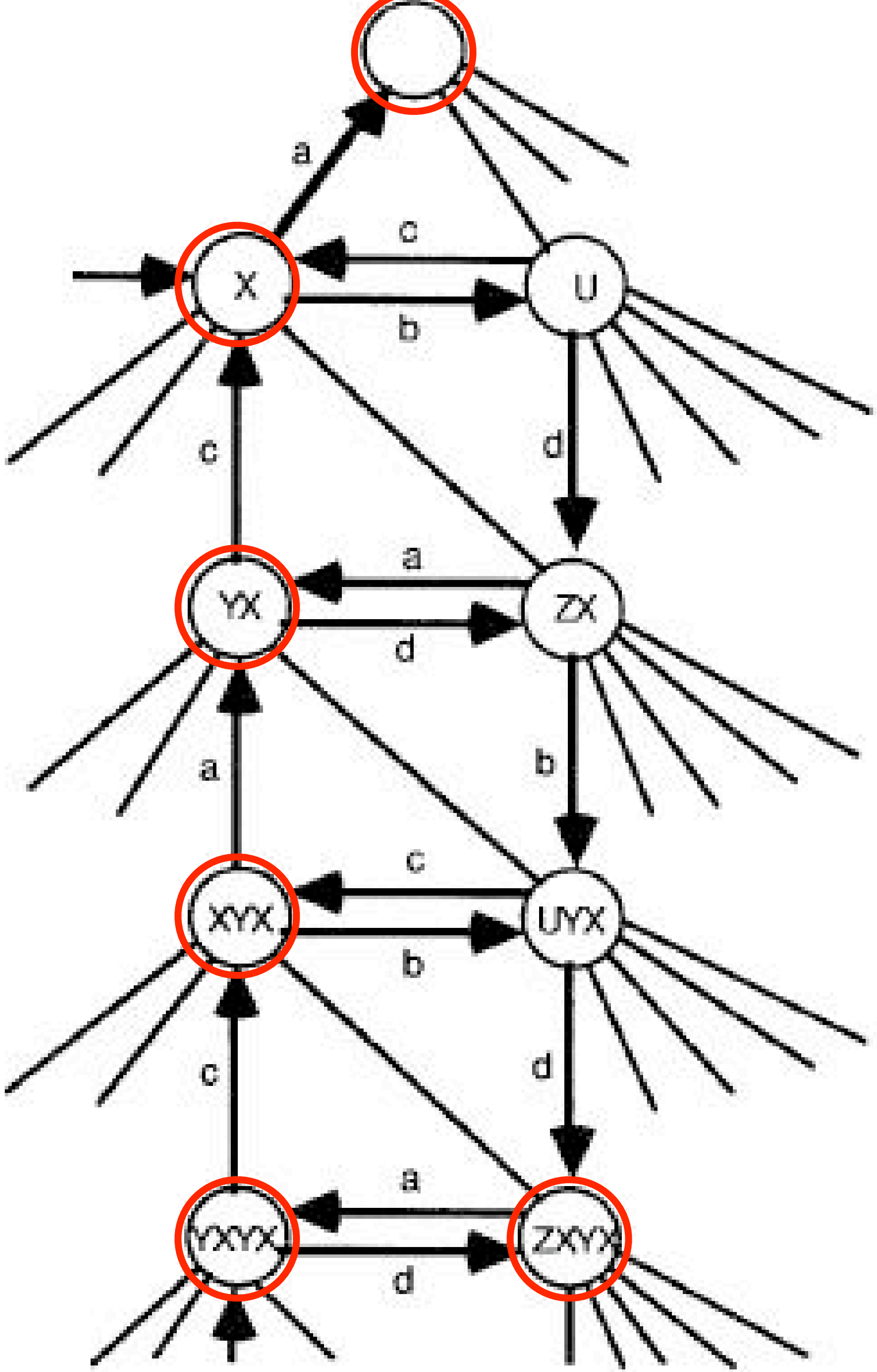
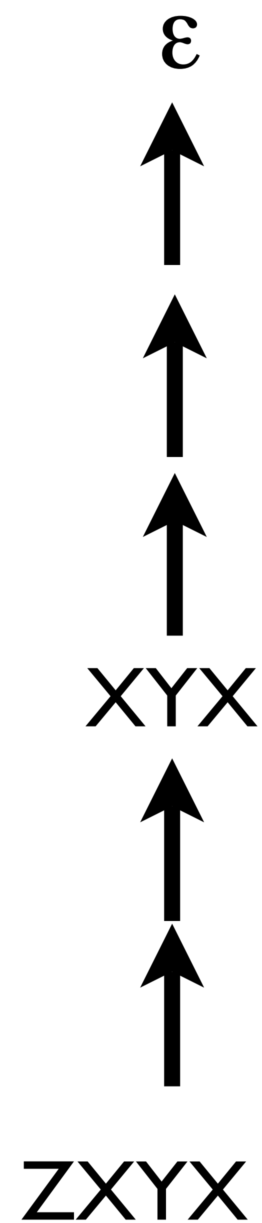
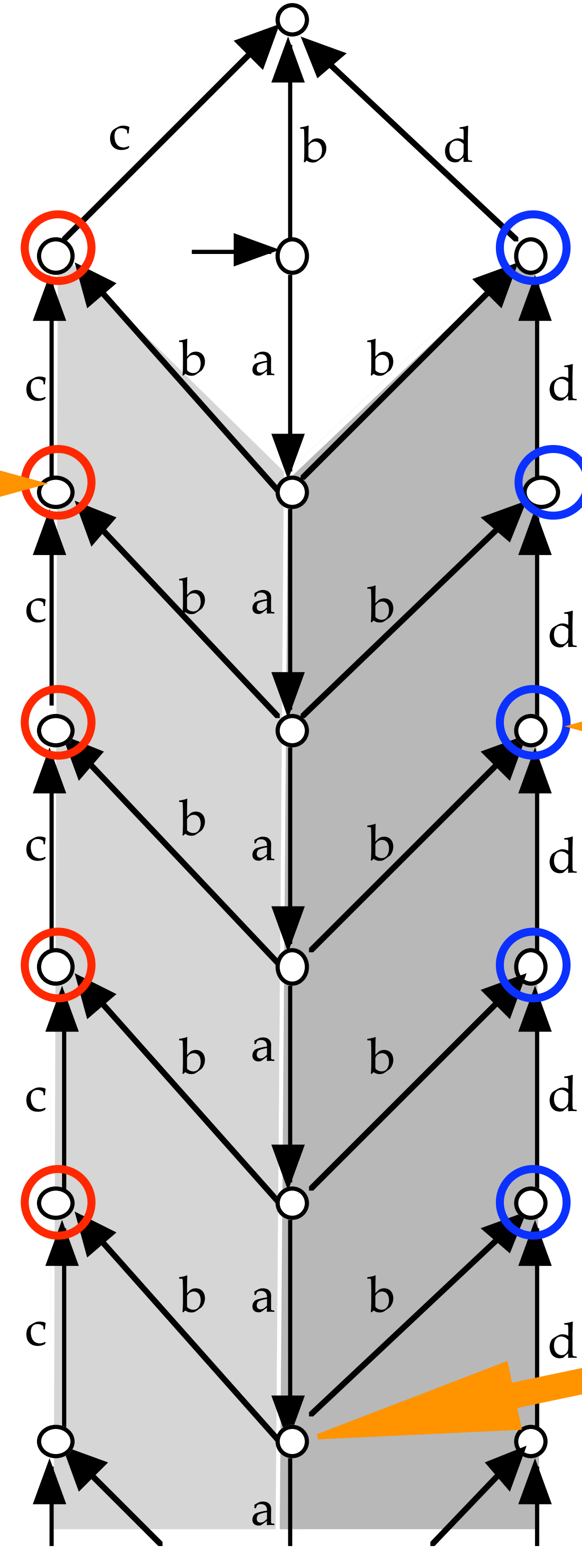


FIGURE 9



interpolation

TOWER

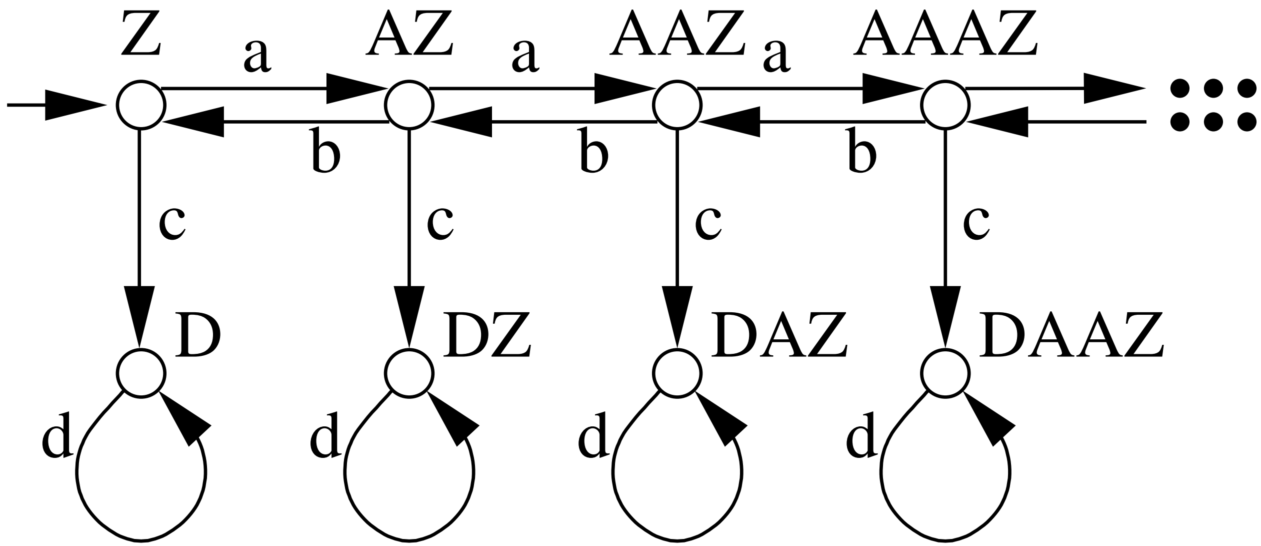


XYX

XYX

ZXYX

periodically decomposable, still not a BPA graph

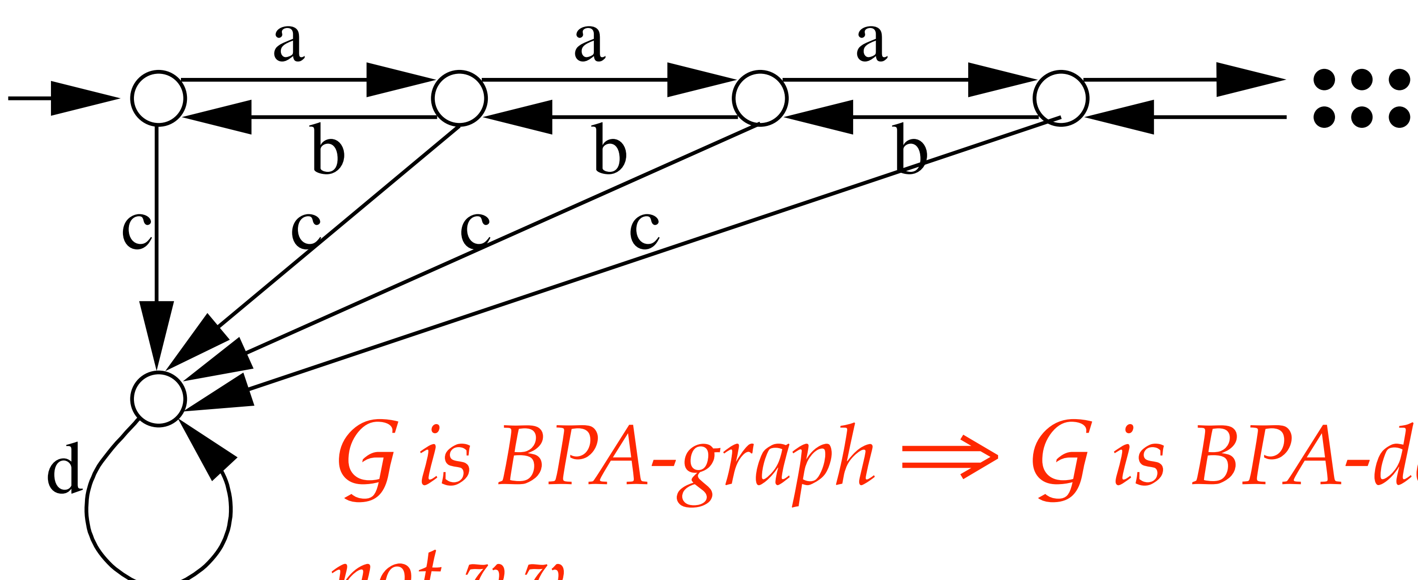


$Z = aAZ + cD$

$A = aAA + cD + b$

$D = dD$

not normed !



$\mathcal{G}$ is BPA-graph $\Rightarrow$ $\mathcal{G}$ is BPA-definable, not v.v.

Theorem 1 (Caucal, 1990) The class of normed BPA-graphs is closed under minimization.

$$B = 0(\underline{0} \parallel B) + 1(\underline{1} \parallel B)$$

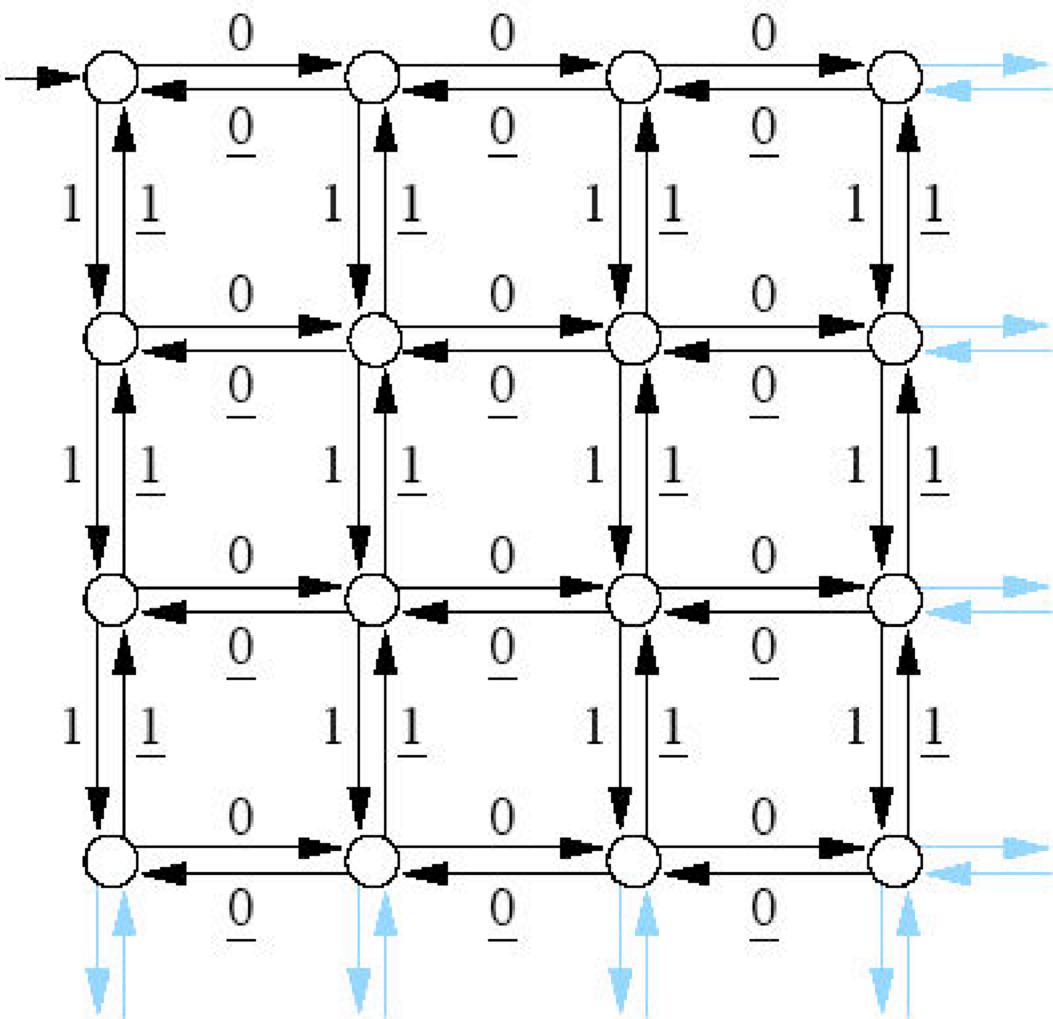


Figure 5: The process Bag.

BAG

class-room question:
is this graph a BPA-graph?

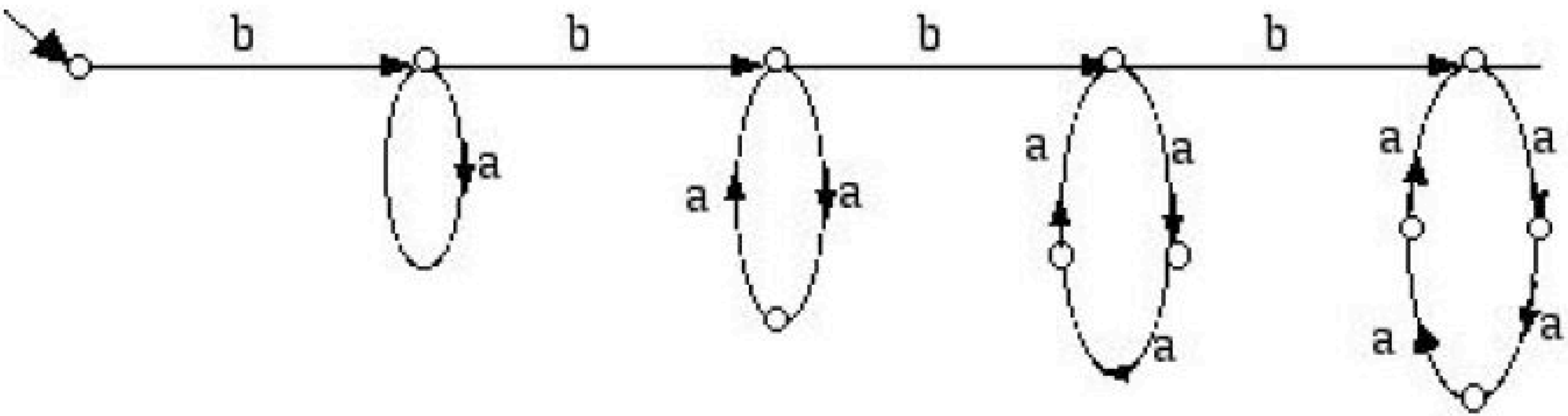
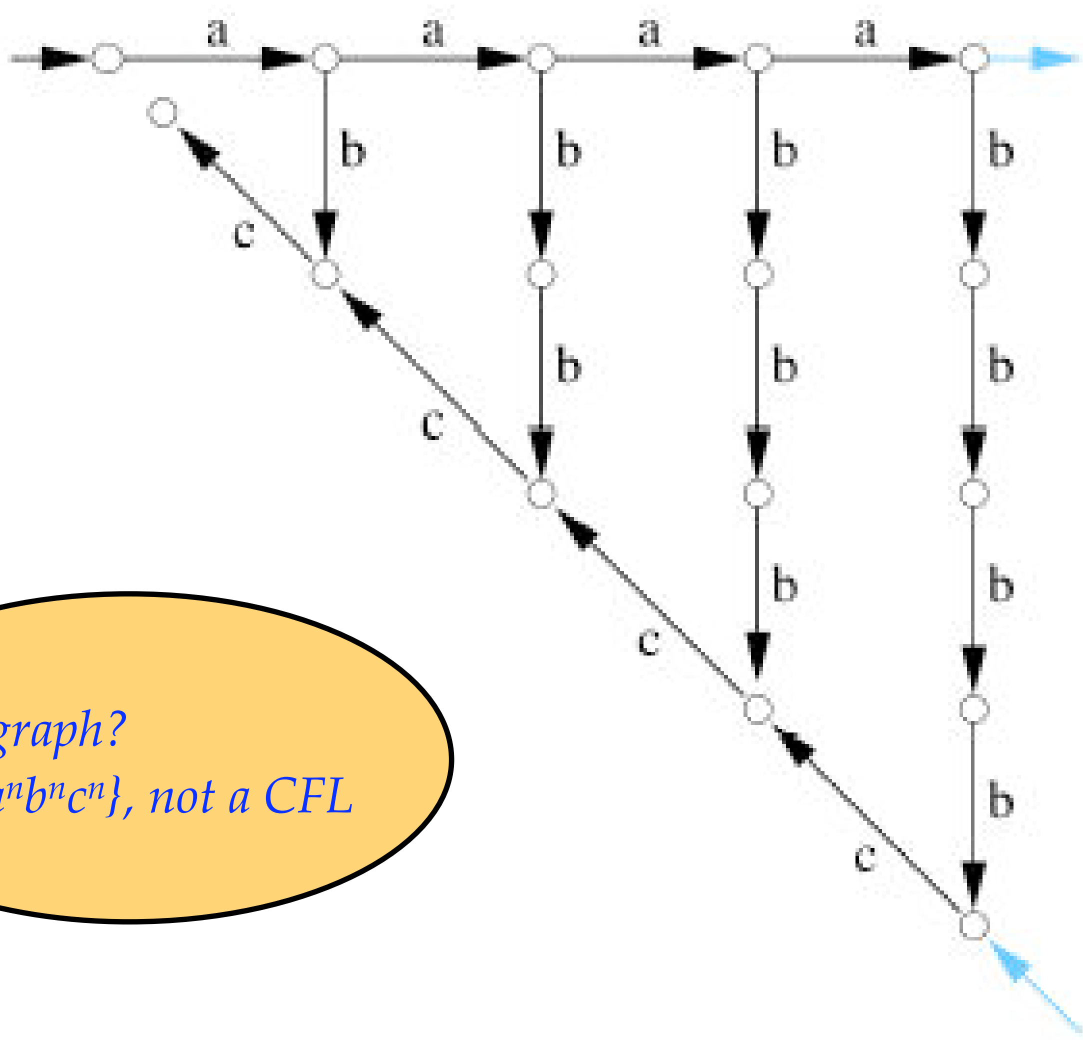


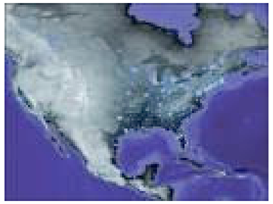
Fig. 8. The labeled transition graph RINGS

canonical, not normed

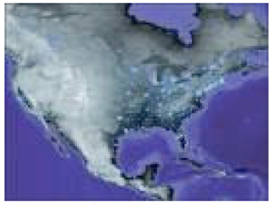


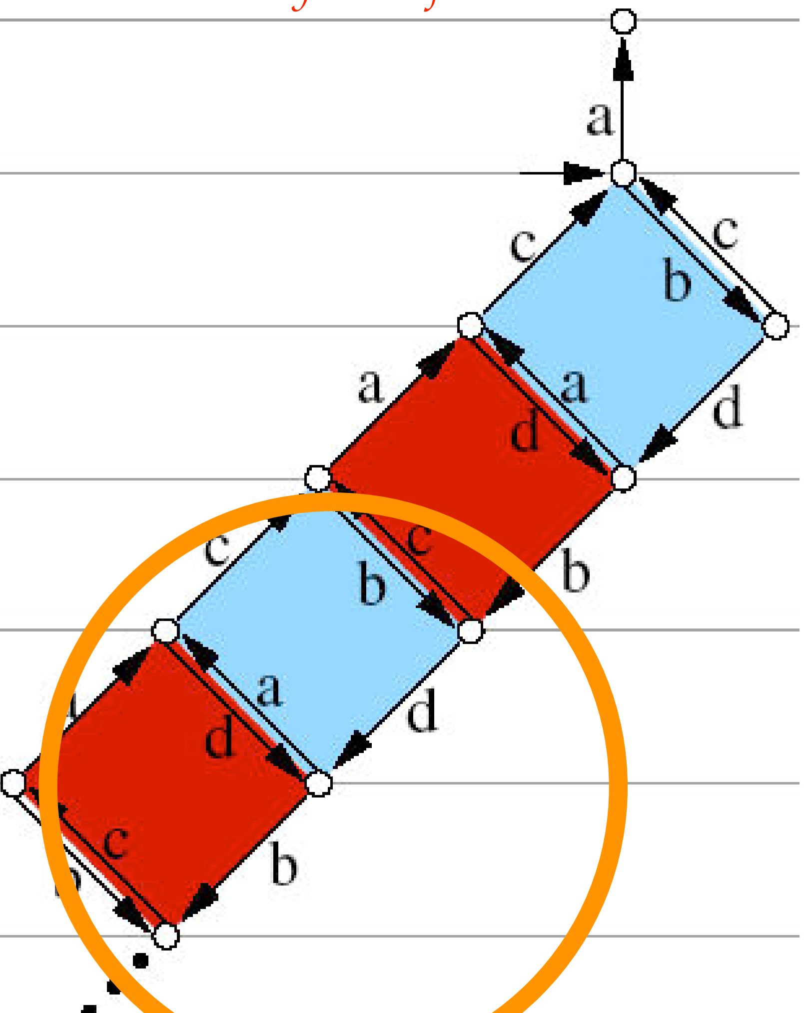
BPA graph?
L = {$a^n b^n c^n$}, not a CFL

Contents

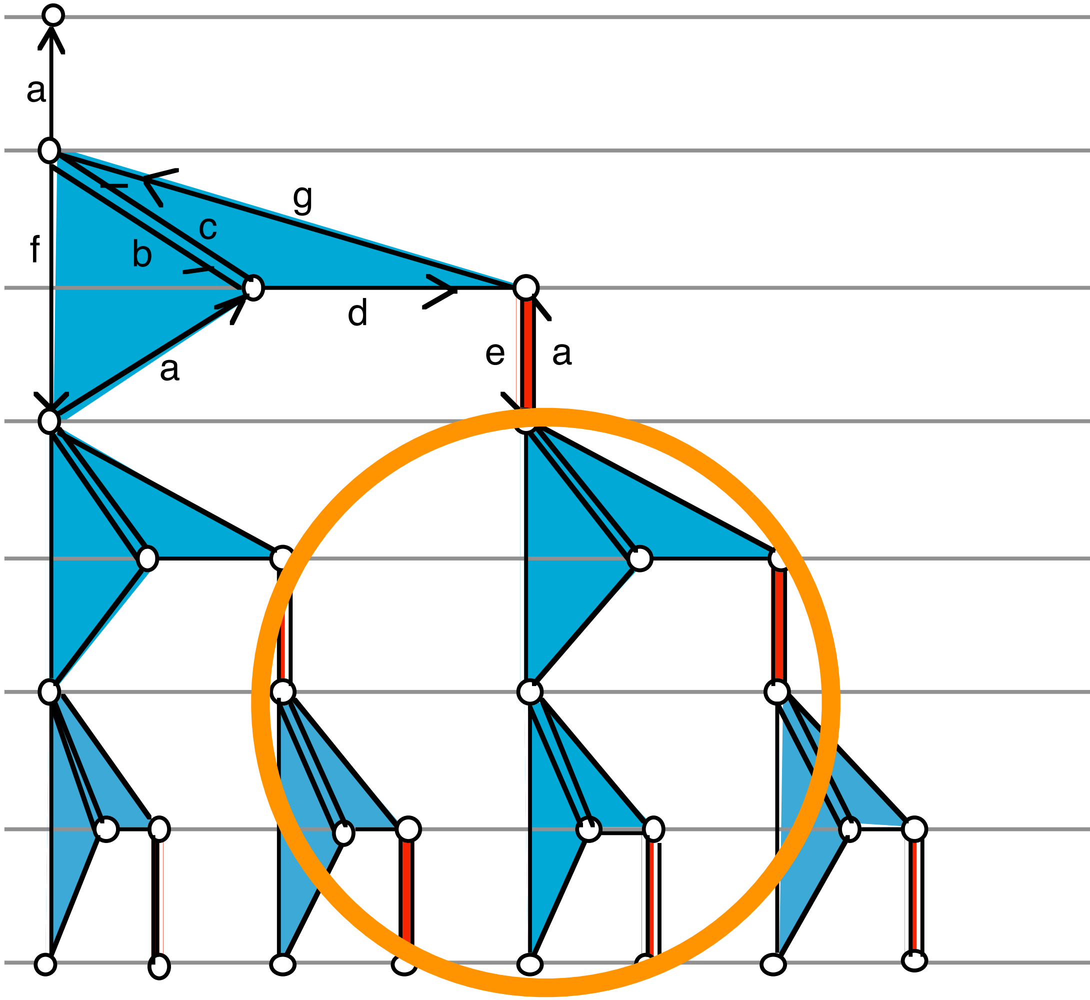


density λn. #In(r, n, $\mathcal{G}$)





linear density



exponential

connectivity: $\lim_{n\to\infty}$ *# icc's of Out(r, n, G)*

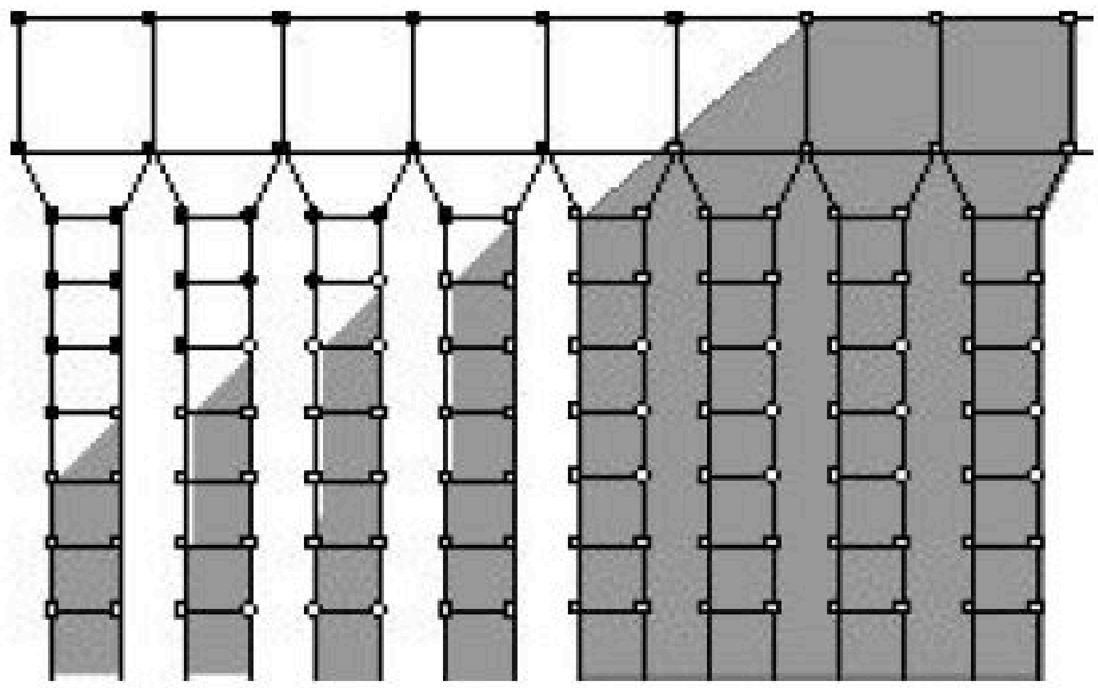


Fig. 7. Determining the connectivity of the graph TEMPLE

connectivity: number of ways to infinity

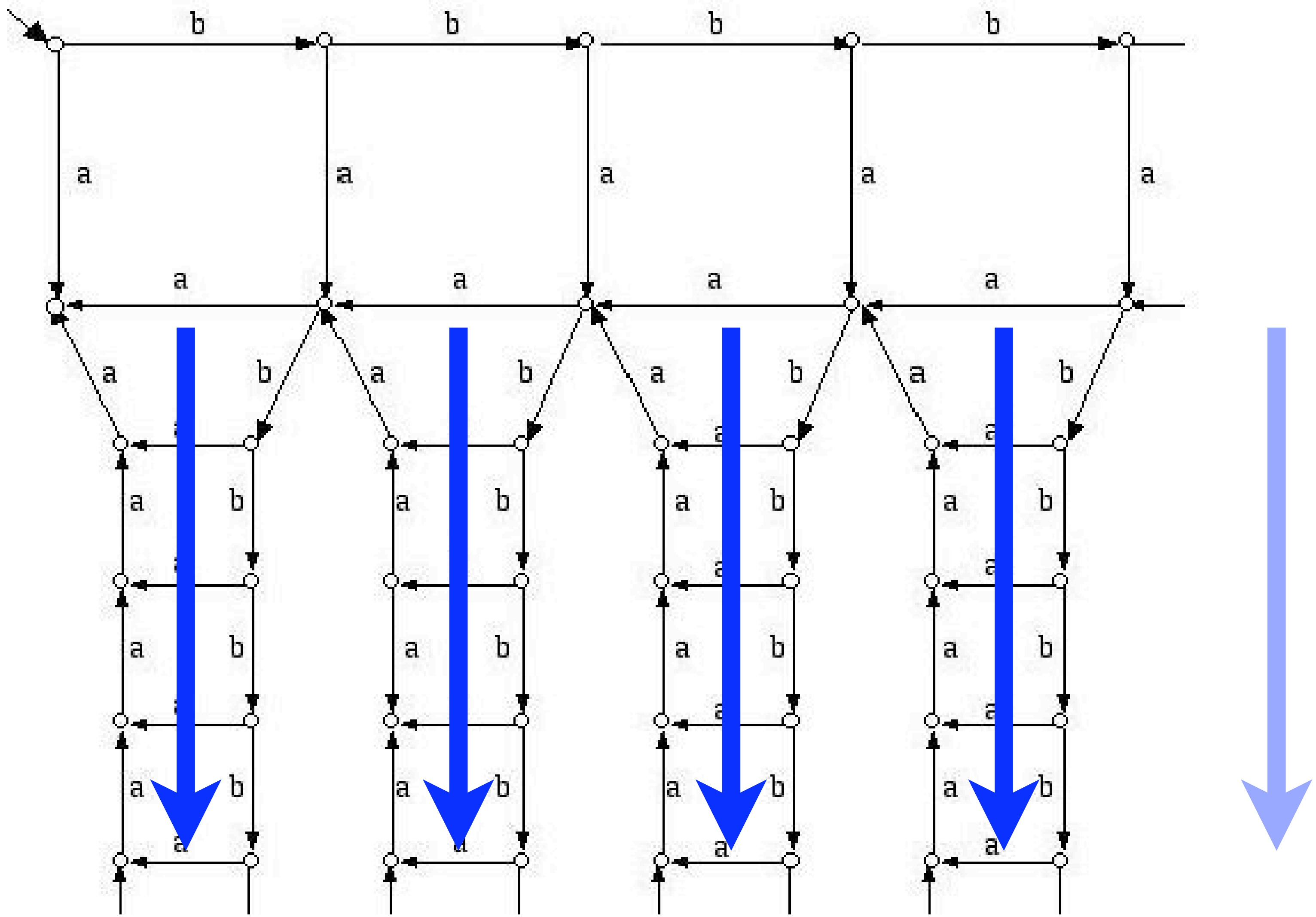
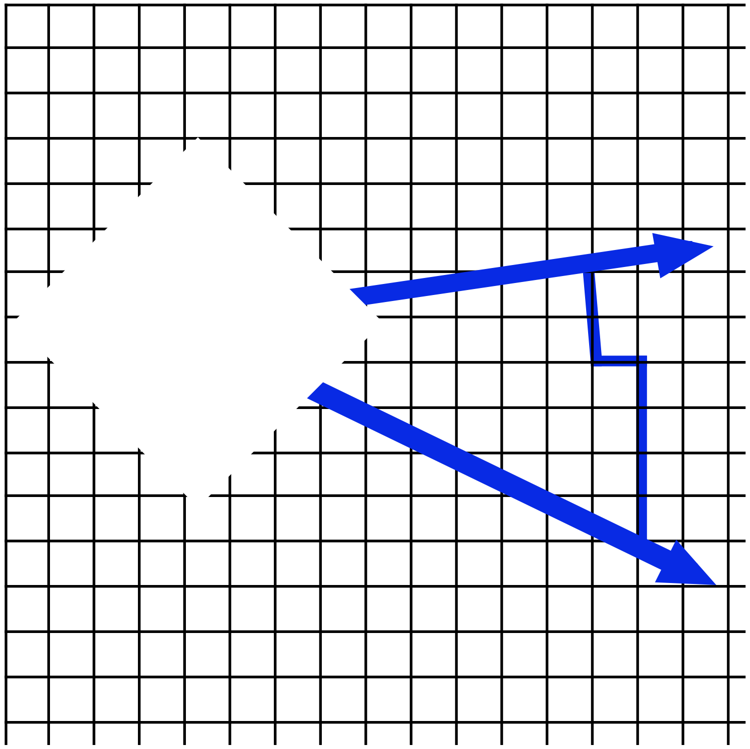
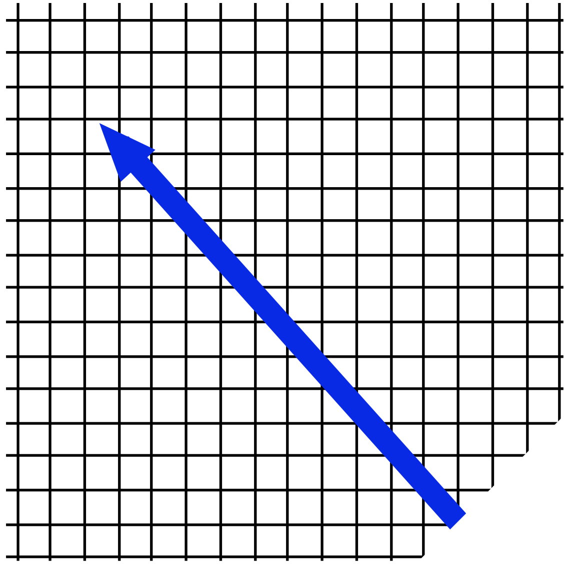
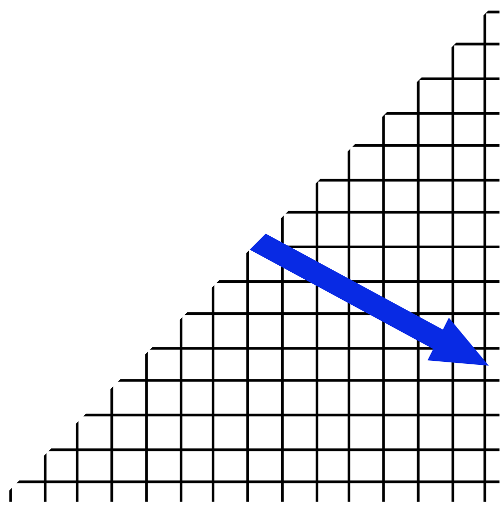


Fig. 2. The labeled transition graph TEMPLE



two equivalent ways to infinity





two non-equivalent ways to infinity: $c = 2$

density = exponential

connectivity = 1

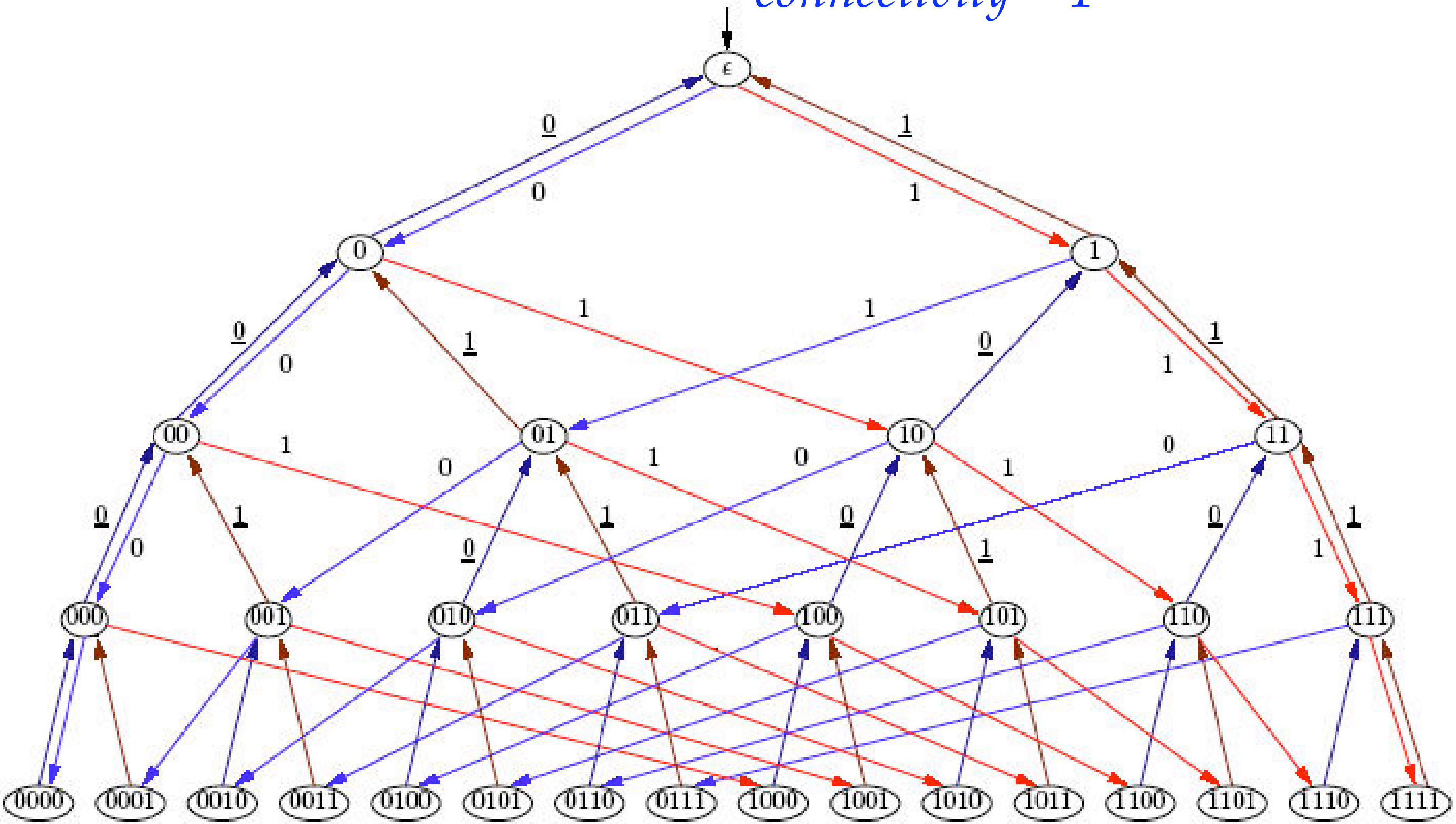


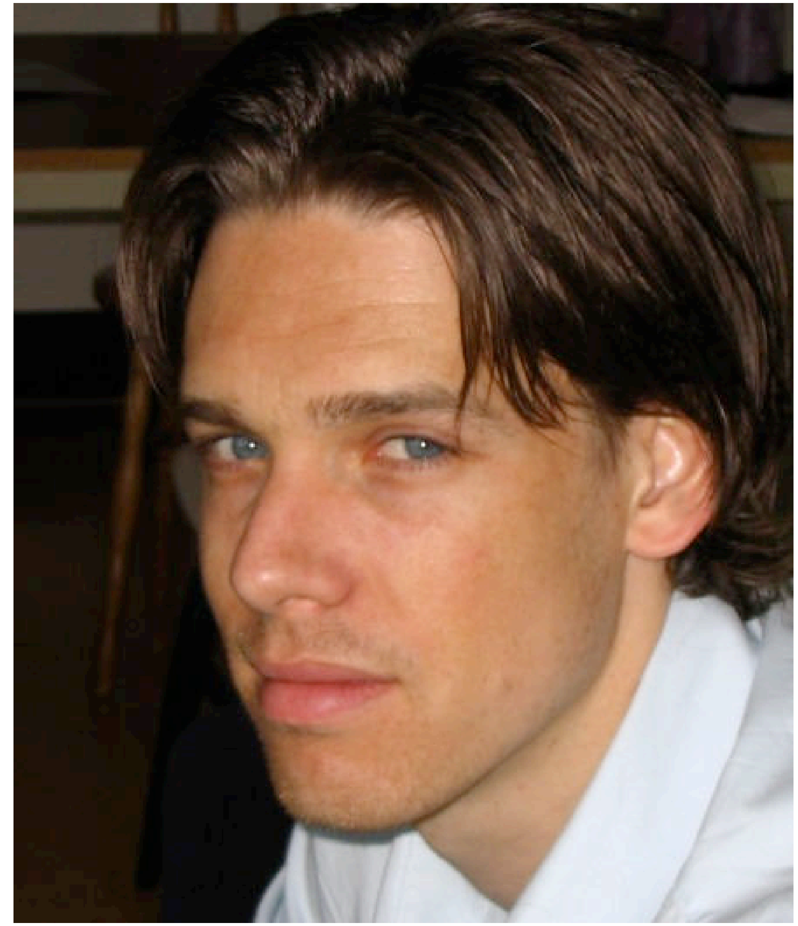
Fig. 9. The canonical process graph QUEUE of Queue

connectivity versus density for BPA graphs

c \ d	const	linear	polynom	exponen
0				
n > 0		RAILS ~~TOWER~~	BAG RINGS TRIANGLE	QUEUE
∞			TEMPLE	STACK KITES BUTTERFLY

Clemens

Bas

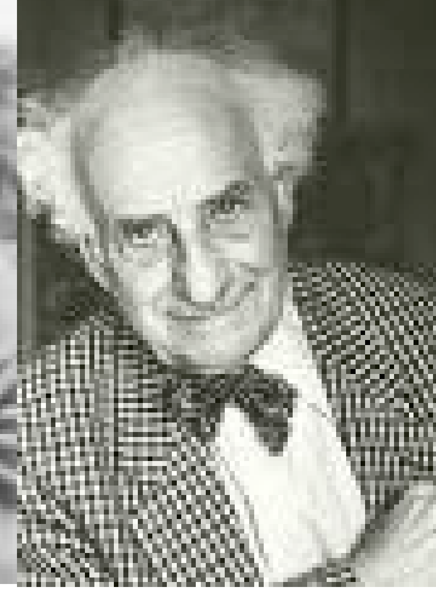


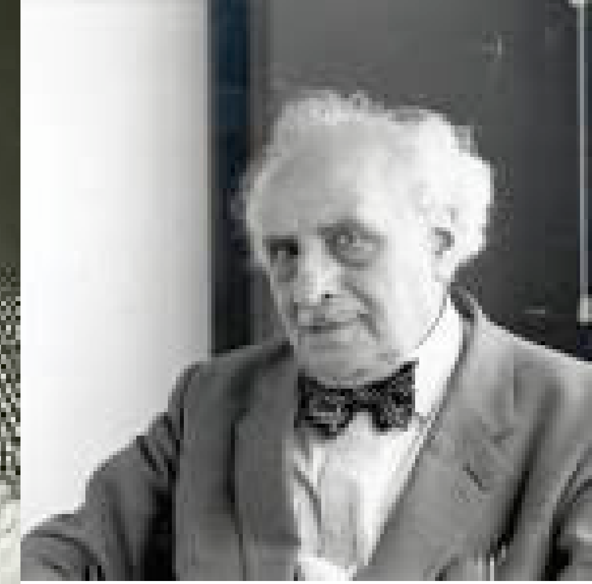
Corollary 2.3. *The possible connectivity-density value pairs* $\langle \mathbf{c}, \mathbf{d} \rangle$ *for context-free graphs (in the sense of Muller and Schupp) are described by Table 1.*

Corollary 2.4. *The possible connectivity-density value pairs* $\langle \mathbf{c}, \mathbf{d} \rangle$ *or regular graphs (rooted pattern graphs) of finite degree are also described by Table 1.*

Hans Freudenthal

1905- 1990





Lincos;: Design of a language for cosmic intercourse (Studies in logic and the foundations of mathematics)

Mathematics as an Educational Task

Über die Enden topologischer Räume und Gruppen.

Von

Hans Freudenthal in Laren (Nordholland).

Obzwar die Eigenschaften im Kleinen einer Lieschen kontinuierlichen Gruppe in hohem Maße ihre Eigenschaften im Großen bestimmen, leistet die Liesche Theorie doch wenig für die Erkenntnis dieser Eigenschaften.

Contents

THEOREM

Let $\mathcal{G}$ be a normed canonical graph of finite degree that is not a BPA-graph.
Then $\mathcal{G}$ is not BPA-definable.

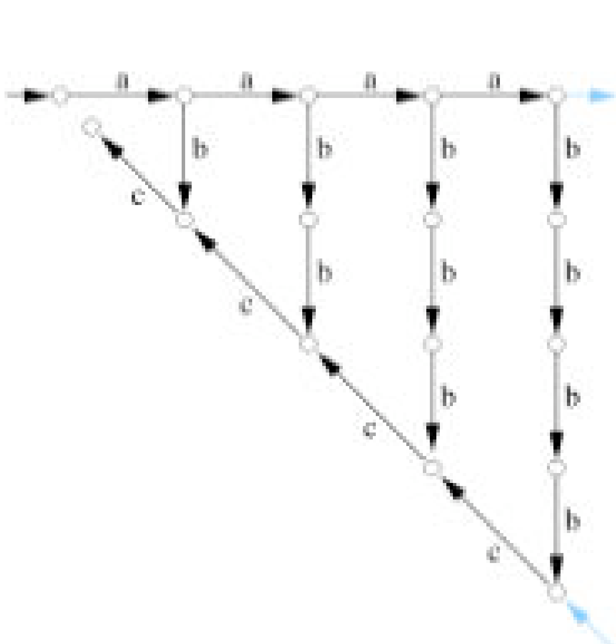
PROOF. *Let $\mathcal{G}$ be a normed canonical graph of finite degree that is not a BPA-graph. Suppose $\mathcal{G}$ is BPA-definable. That is, for some BPA-graph $\mathcal{G}'$: $\mathcal{G} \underline{\leftrightarrow} \mathcal{G}'$. Then $\mathcal{G}'$ is also normed and of finite degree. And $can(\mathcal{G}') = \mathcal{G}$. So $can(\mathcal{G}')$ is not a BPA-graph. However, by Caucal's theorem can $(\mathcal{G}')$ is again a BPA-graph. Contradiction. Hence $\mathcal{G}$ is not BPA-definable.*

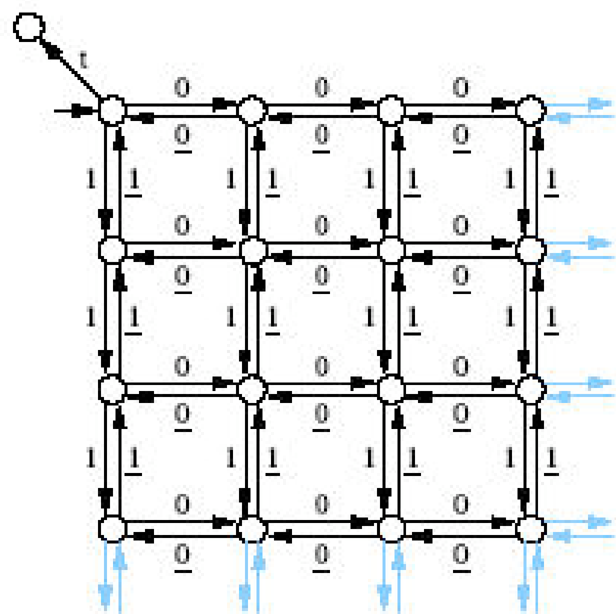
THEOREM

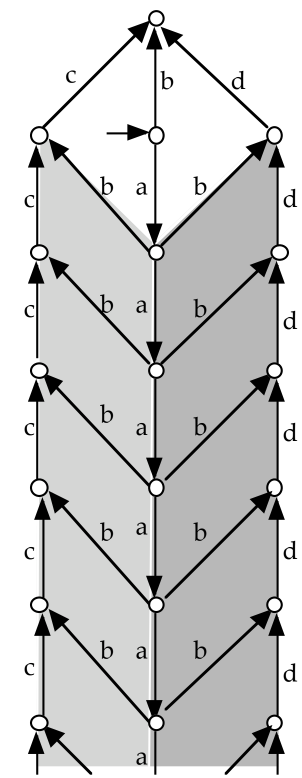
Let G be a normed canonical graph of finite degree that is not a BPA-graph.
Then G is not BPA-definable.

APPLICATION.

TRIANGLE, TBAG, TOWER
are not even BPA-definable







How to show for unnormed graphs that they are not BPA-definable?

Burkart, Caucal, Steffen: if g is a BPA graph, can(g) is a pattern graph

Caucal: pattern graphs of finite degree are context free graphs a la Muller and Schupp

pattern graph =
context free graph =
graph with periodical decomposition

hmmm...
easy to see,
hard to prove

THEOREM (Corollary of BCS)

Let $\mathcal{G}$ be a canonical graph of finite degree without periodic decomposition.

Then $\mathcal{G}$ is not BPA-definable.

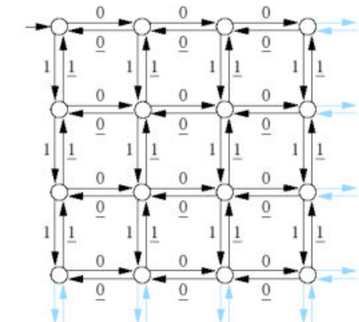


Figure 5: The process Bag.

PROOF. *Let $\mathcal{G}$ be a canonical graph of finite degree without period. decomp. Suppose $\mathcal{G}$ is BPA-definable. That is, for some BPA-graph $\mathcal{G}'$: $\mathcal{G} \underline{\leftrightarrow} \mathcal{G}'$. Then can($\mathcal{G}'$) = $\mathcal{G}$ is by BCS a pattern graph = graph with period. decomp. Contradiction. Hence $\mathcal{G}$ is not BPA-definable.*

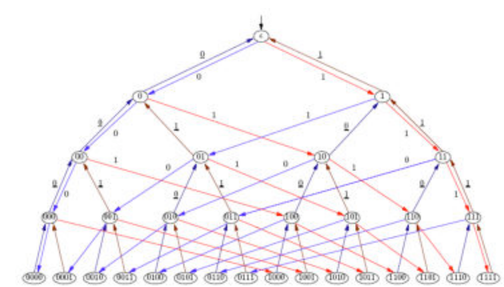
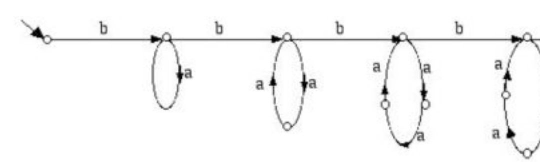


Fig. 9. The canonical process graph QUEUE of Queue

APPLICATION.

BAG, RINGS, QUEUE *are not even BPA-definable*



hmmm...
easy to see,
hard to prove

Why does BAG not have a p.d. like TEMPLE?

Unwinding the fragment structure graph we obtain infinite branches of connected fragments. In a BPA graph (so with p.d.), these branches are eventually separated (far apart). In the chess-board tiling of BAG with identical fragments, this separation does not take place.

Actually, we rather use the equivalent criterion context free of Muller and Schupp. It is fairly easy to show that BAG is not c.f.

FRONTIER THEOREM (a la Muller-Schupp)

Let G be a canonical graph such that:

The number of non-bisimilar frontier points in Out(r,n, G) tends to infinity, with increasing n.

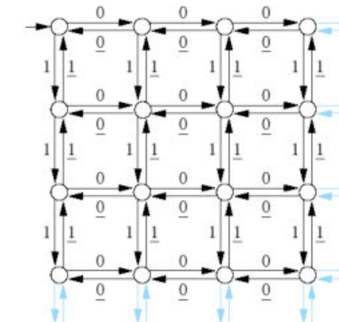


Figure 5: The process Bag.

(frontier points in Out(r,n, G) are points with distance just n of the root r)

Then G is not BPA-definable.

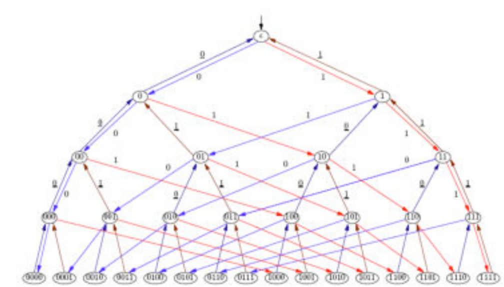
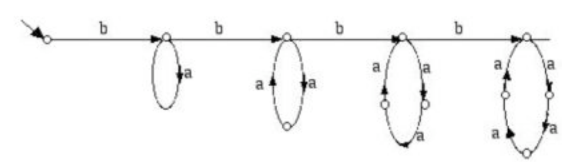
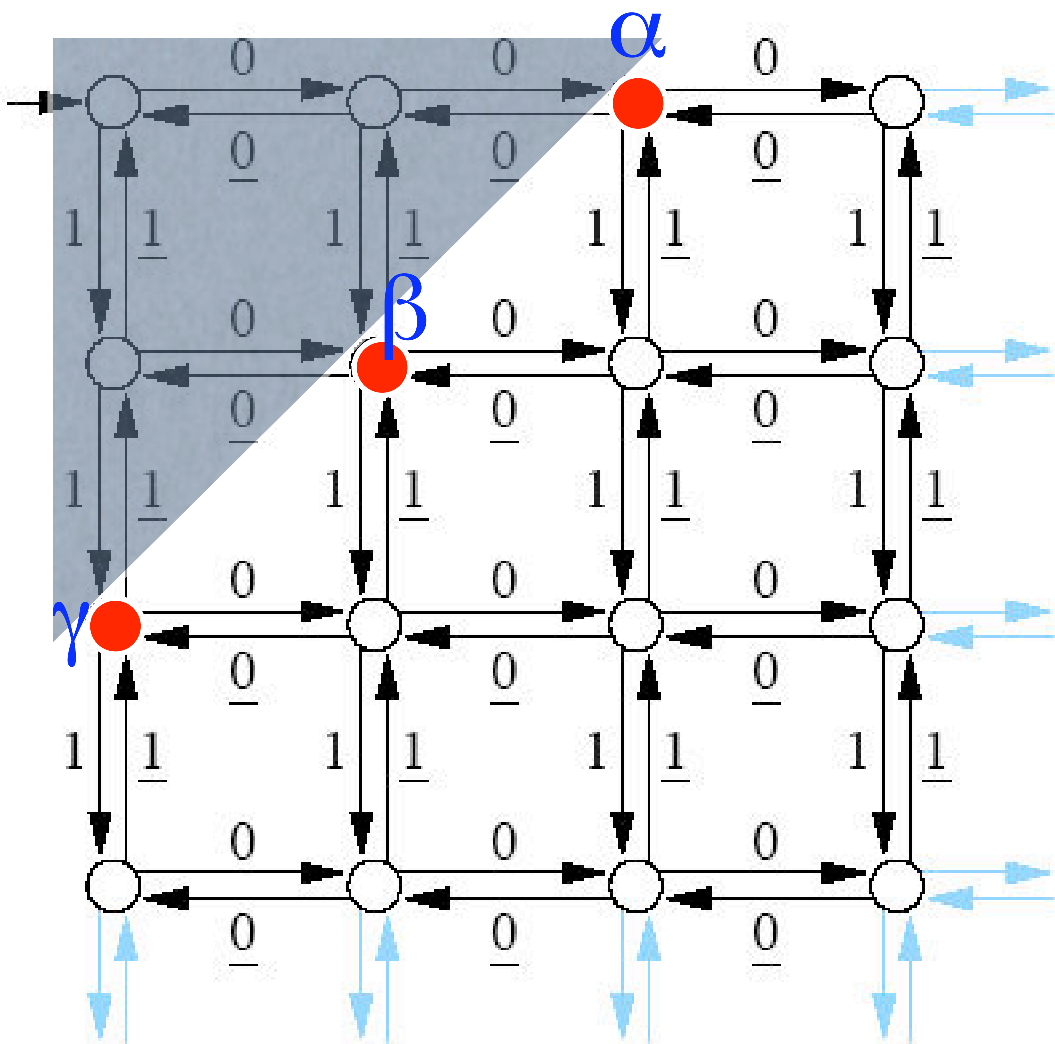


Fig. 9. The canonical process graph QUEUE of Queue

COROLLARY.

BAG, RINGS, QUEUE *are not even BPA-definable*





frontier points α, β, γ not bisimilar in remainder graph

Contents

1. *Some old definability results*
2. *Process graph dictionary*
3. *BPA graphs*
4. *Densit… tivity*
5. *Non-a… lusions*
6. *BPP*

see our paper in the proceedings!

Acknowledgement. We are very grateful to Didier Caucal for the graph TEMPLE with its specification, and for pointing out its consequences. We thank Henk Barendregt for discussions about this paper and posing the definability questions for the graphs RINGS and TRIANGLE.