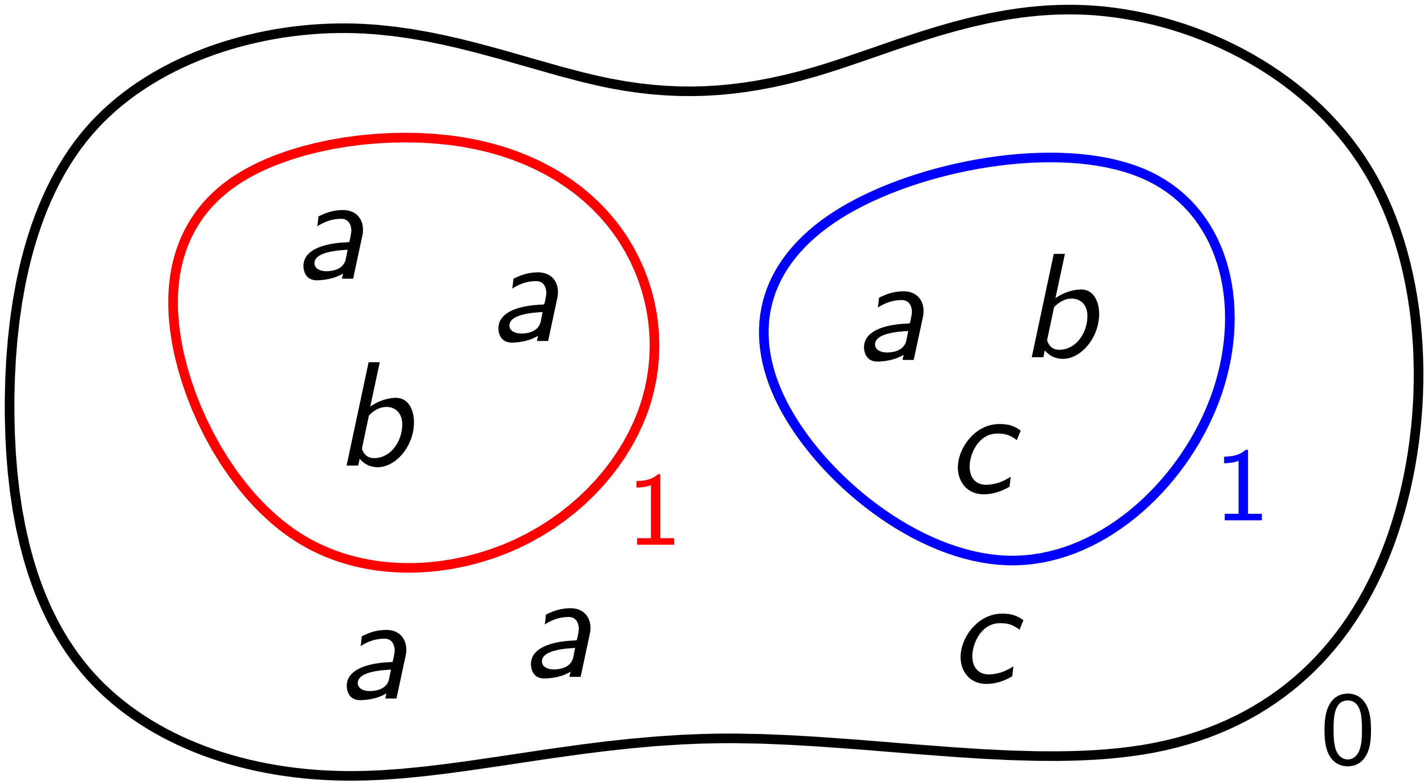
Solving a special case of the P conjecture using dependency graphs with dissolution

Alberto Leporati · Luca Manzoni · Giancarlo Mauri
Antonio E. Porreca · Claudio Zandron

Università degli Studi di Milano-Bicocca

18th Conference on Membrane Computing
27 July 2017, Bradford, UK

P systems with active membranes without charges



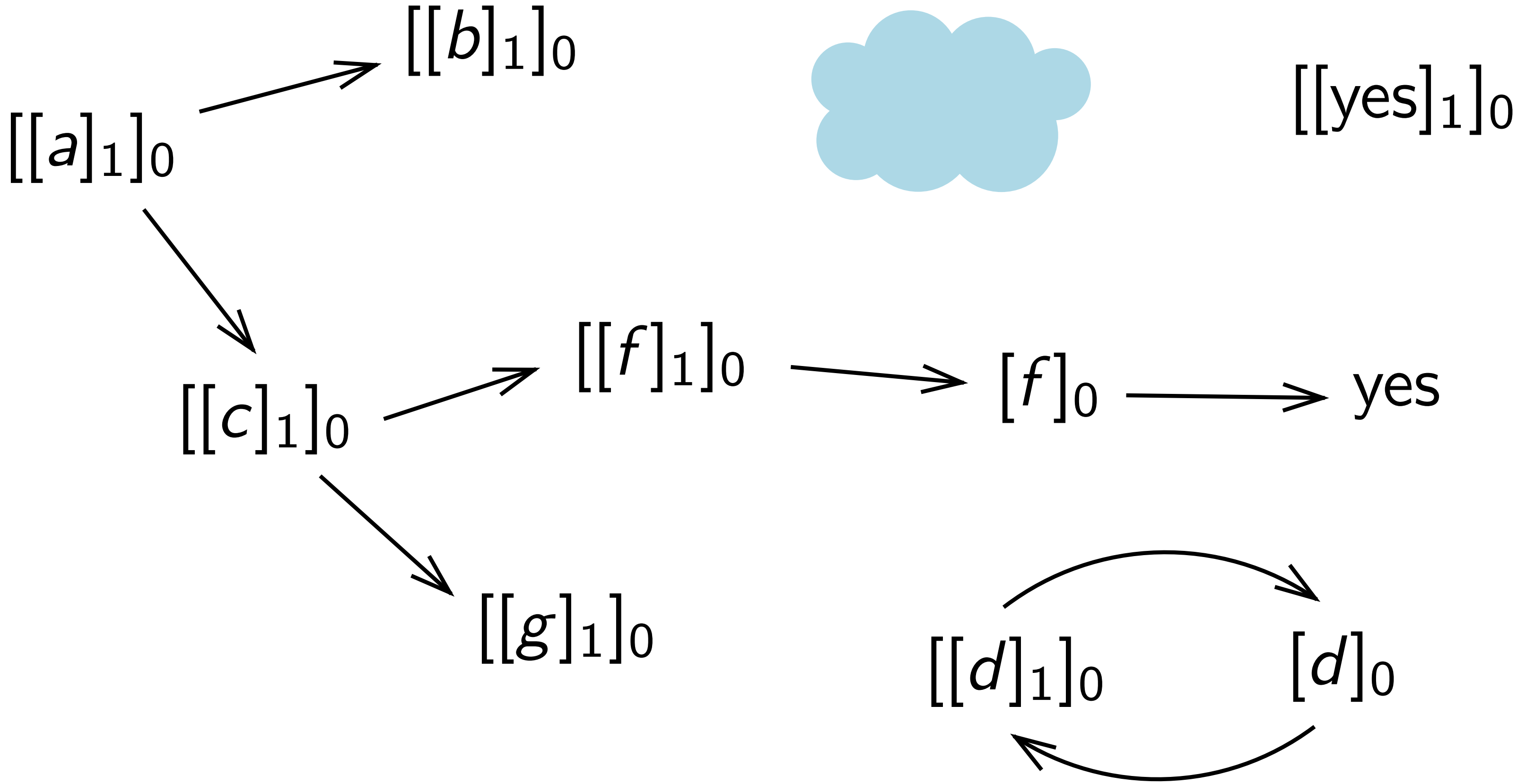
$[a \to bc]_1$ $\quad$ $[f]_1 \to [\,]_1\, f$ $\quad$ $d\,[\,]_1 \to [d]_1$

$[c]_1 \to [f]_1\,[g]_1$ $\quad$ $[f]_0 \to [\,]_0\, \text{yes}$ $\quad$ $[d]_1 \to [\,]_1\, d$

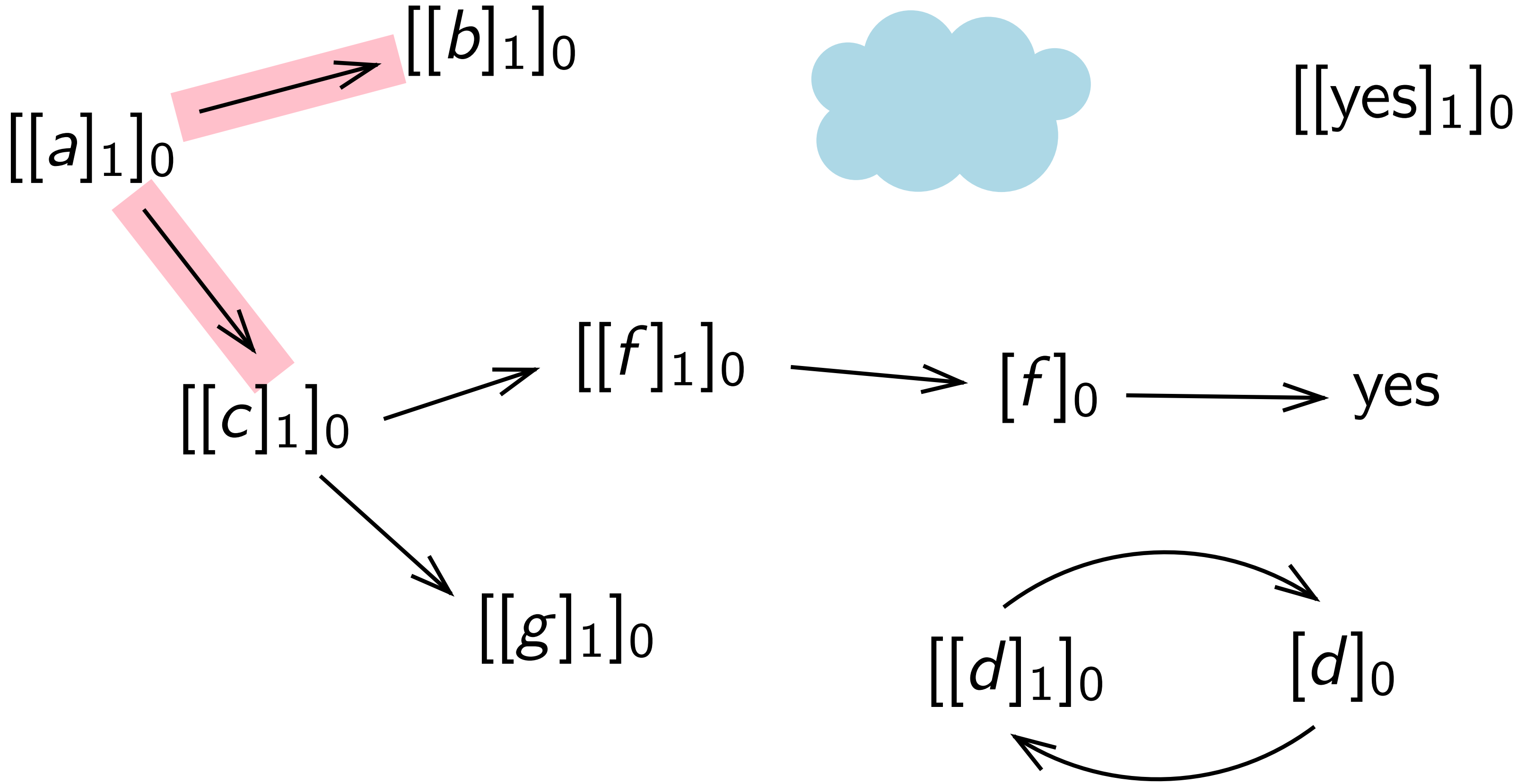
The P conjecture

> [I]t was shown that [...] for solving **NP**-complete problems [in polynomial time] two charges are enough.
>
> Can the polarizations be completely avoided? [...] The feeling is that this is not possible

Dependency graphs



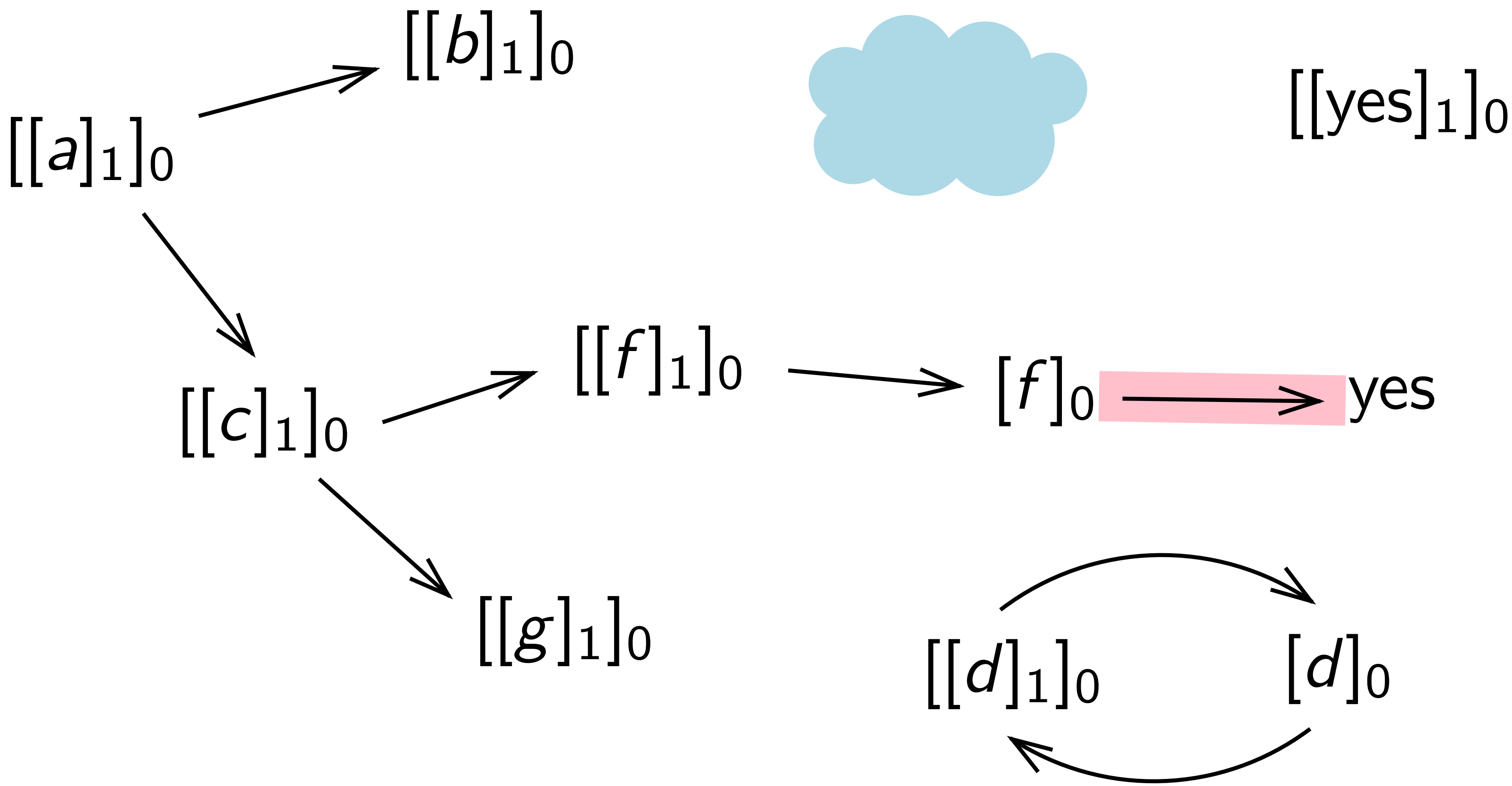
Dependency graphs



$[a \to bc]_1$ $\qquad$ $[f]_1 \to [\,]_1\, f$ $\qquad$ $d\,[\,]_1 \to [d]_1$

$[c]_1 \to [f]_1\, [g]_1$ $\qquad$ $[f]_0 \to [\,]_0\, \mathrm{yes}$ $\qquad$ $[d]_1 \to [\,]_1\, d$

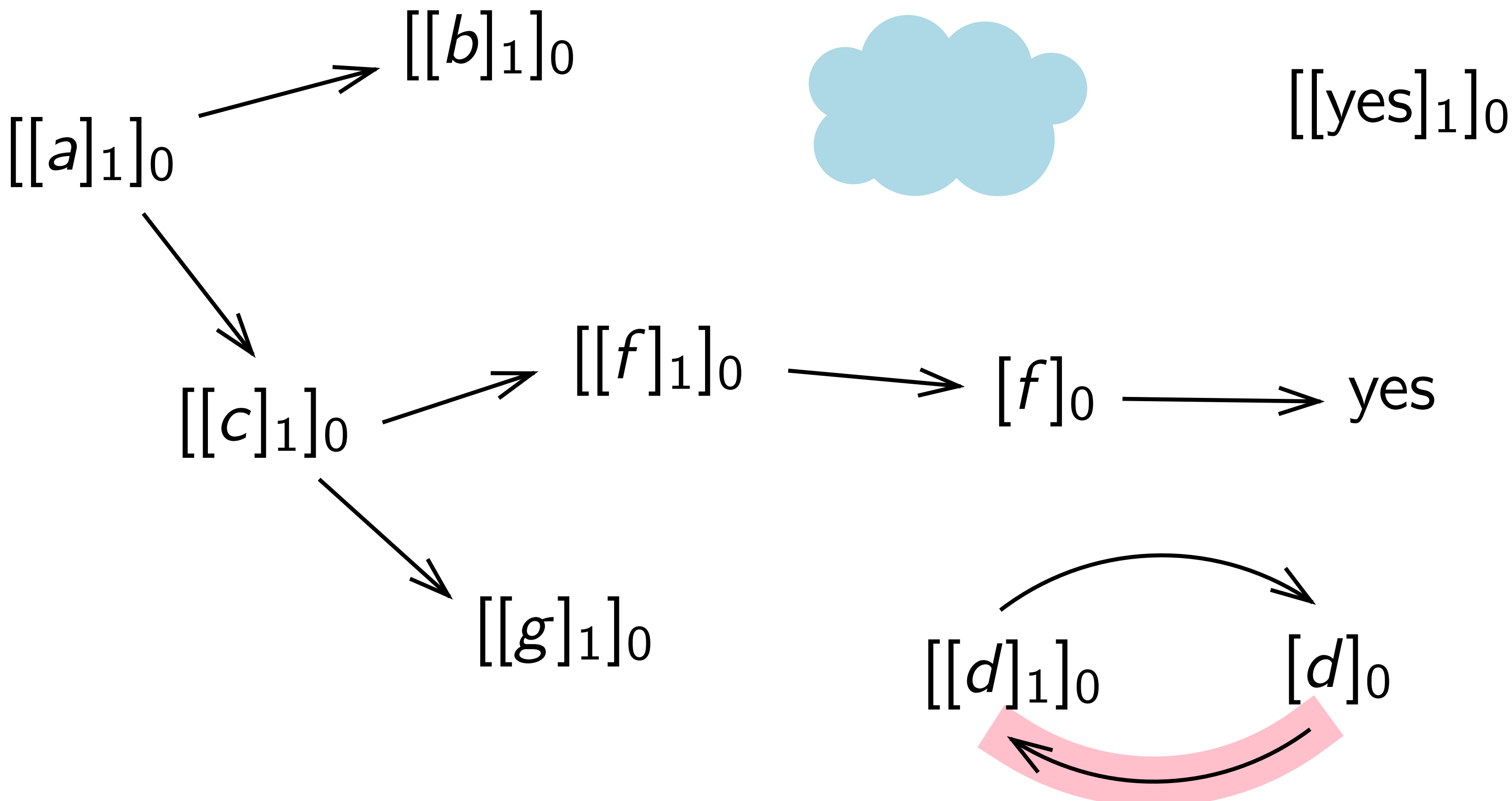
Dependency graphs



$$[a \to bc]_1 \qquad [f]_1 \to [\,]_1\, f \qquad d\, [\,]_1 \to [d]_1$$

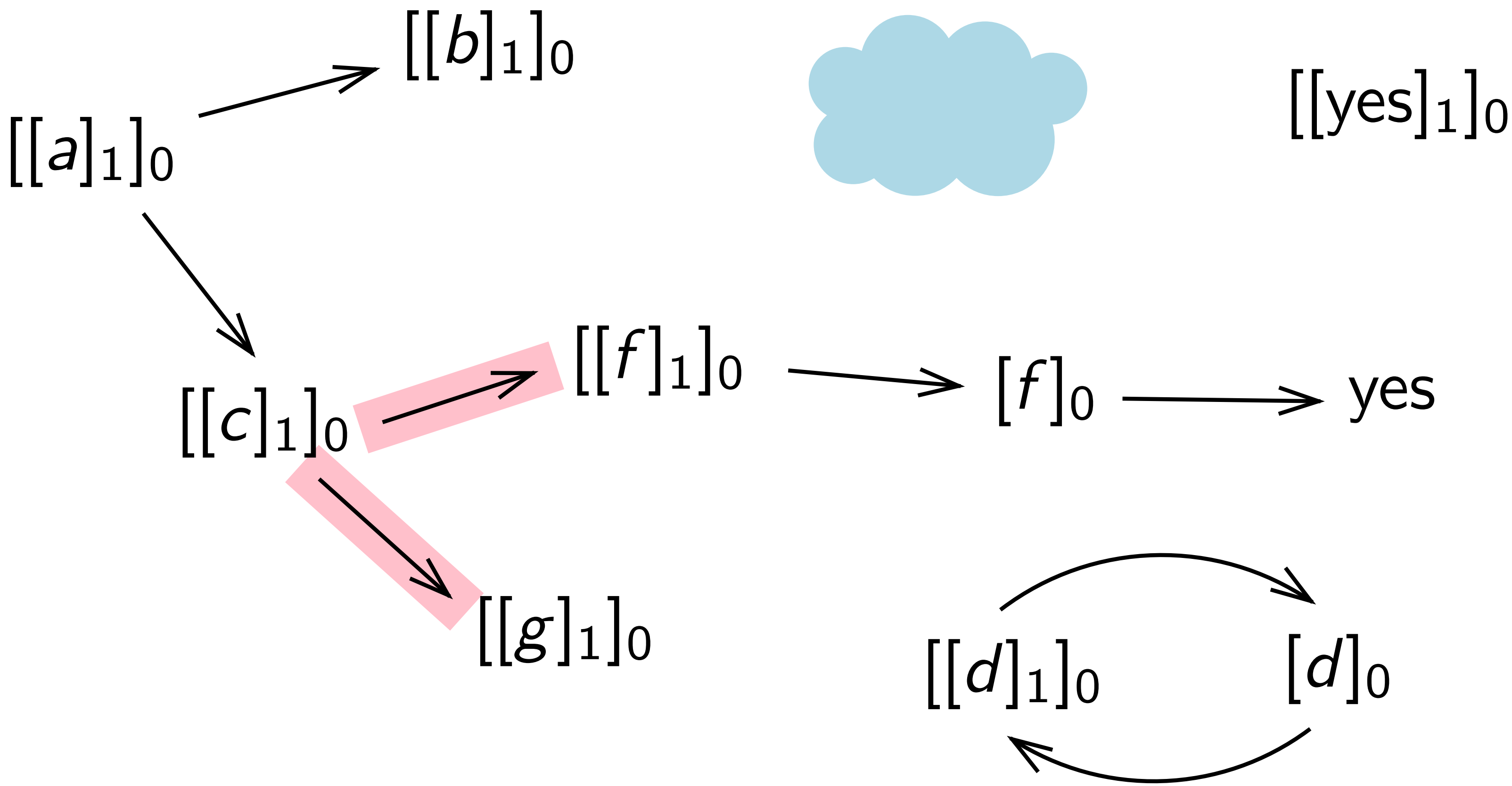
$$[c]_1 \to [f]_1\, [g]_1 \qquad [f]_0 \to [\,]_0\, \text{yes} \qquad [d]_1 \to [\,]_1\, d$$

Dependency graphs

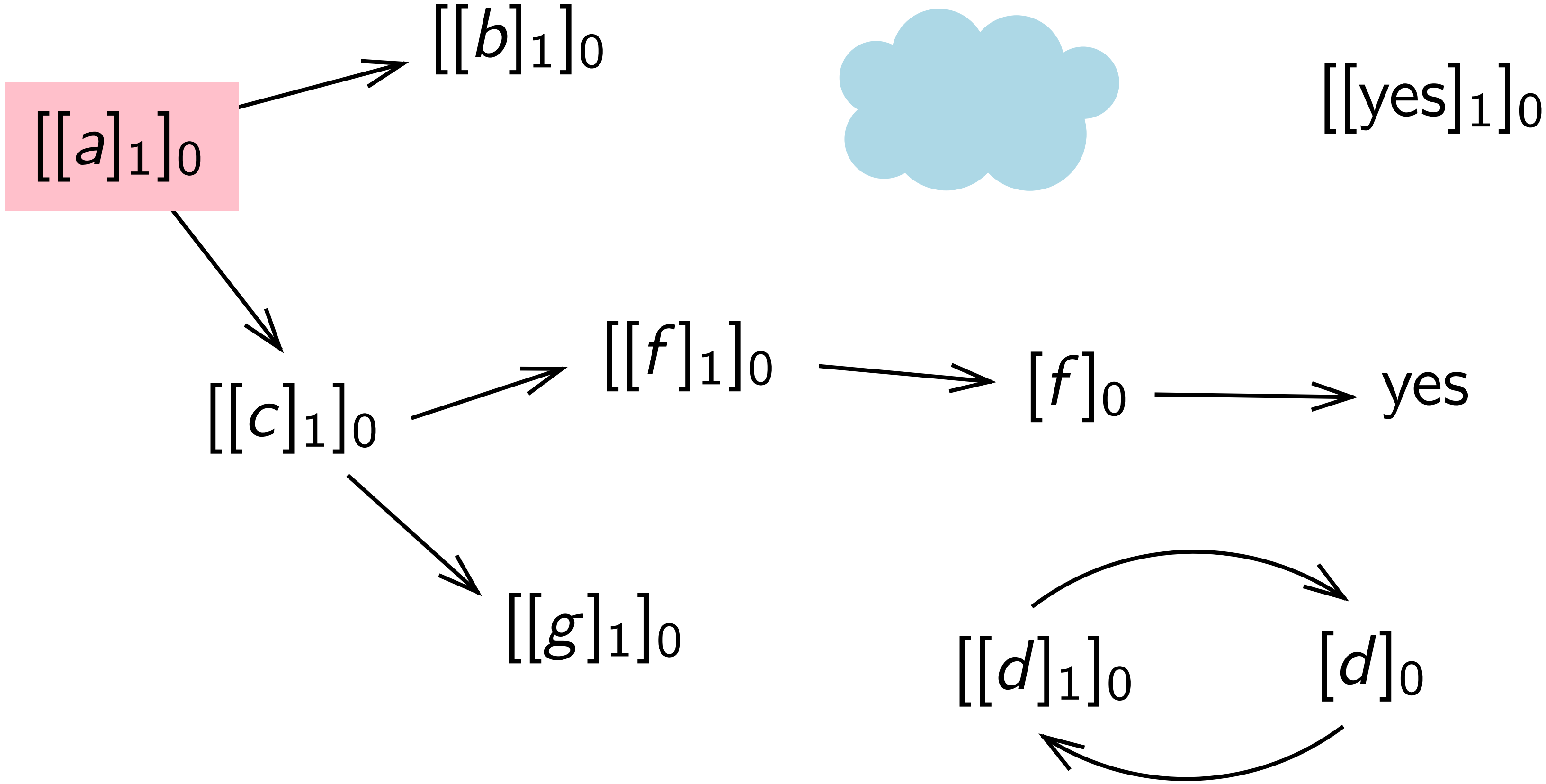


$[a \to bc]_1$	$[f]_1 \to [\,]_1\, f$	$d\,[\,]_1 \to [d]_1$
$[c]_1 \to [f]_1\,[g]_1$	$[f]_0 \to [\,]_0\,\text{yes}$	$[d]_1 \to [\,]_1\, d$

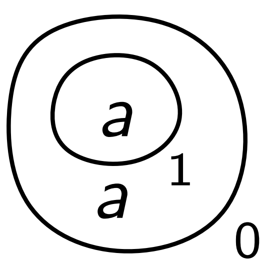
Dependency graphs



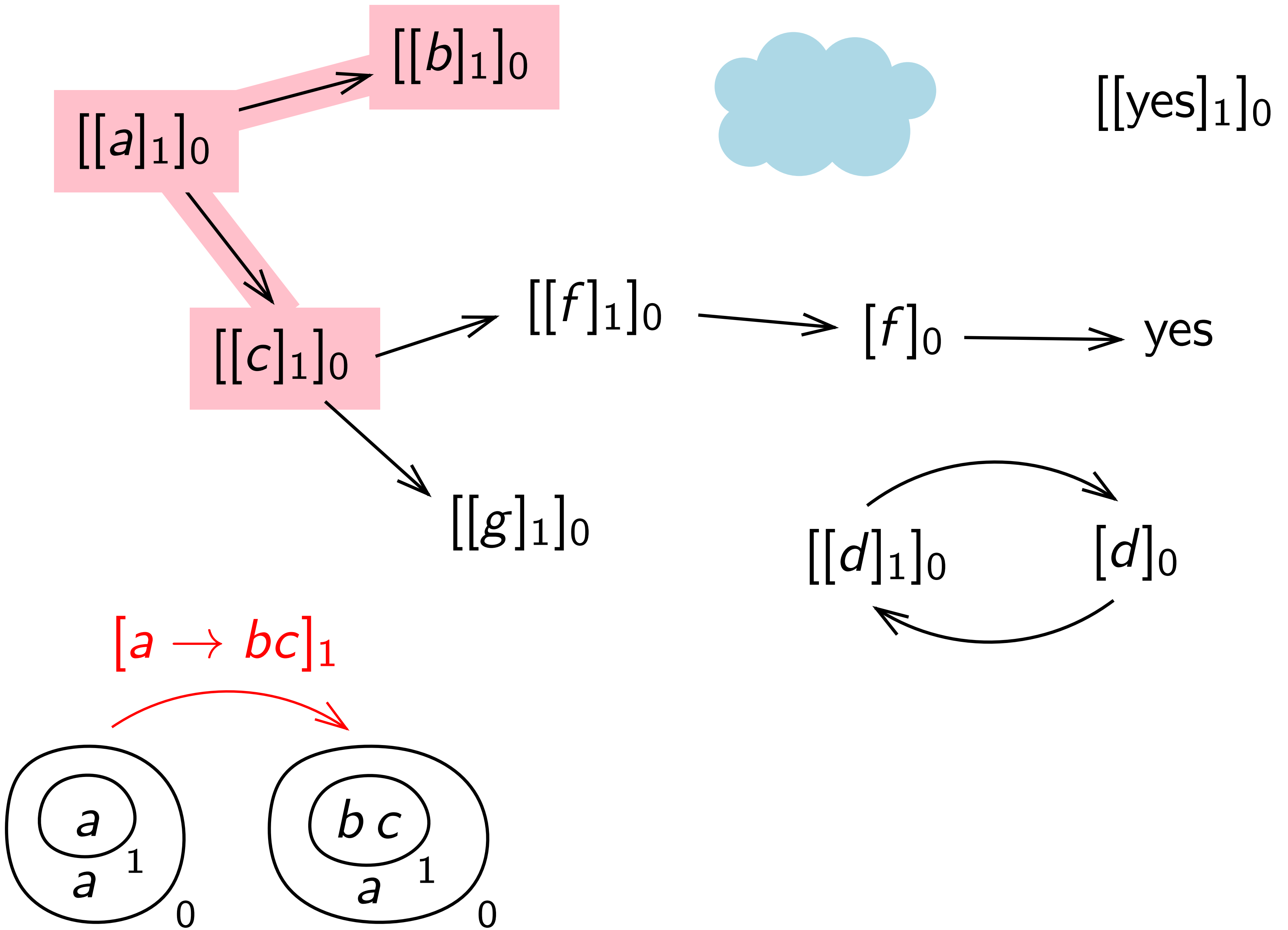
$[a \to bc]_1$	$[f]_1 \to [\,]_1\, f$	$d\,[\,]_1 \to [d]_1$
$[c]_1 \to [f]_1\, [g]_1$	$[f]_0 \to [\,]_0\, \text{yes}$	$[d]_1 \to [\,]_1\, d$



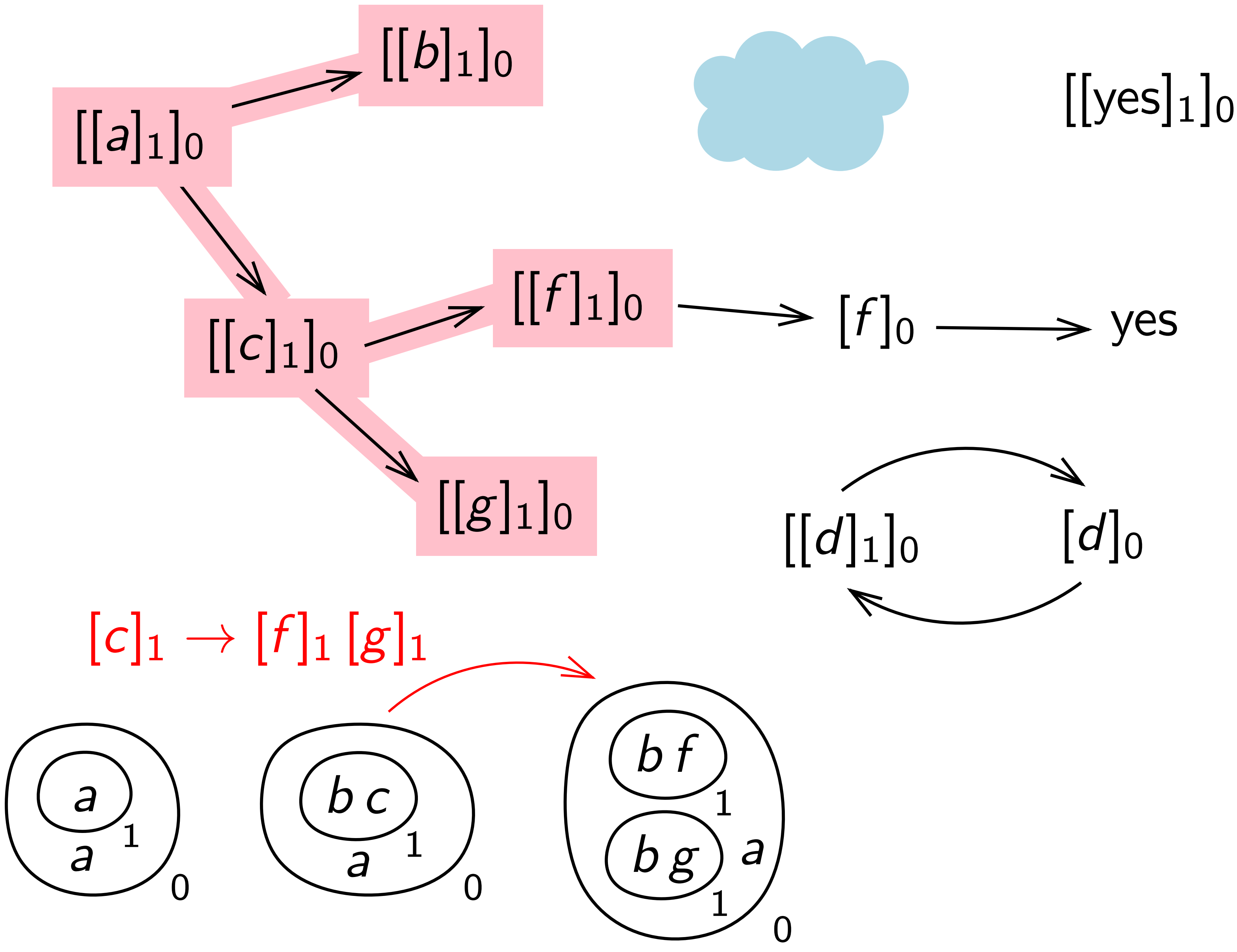
[[a]1]0
[[b]1]0
[[yes]1]0
[[c]1]0
[[f]1]0
[f]0
yes
[[g]1]0
[[d]1]0
[d]0



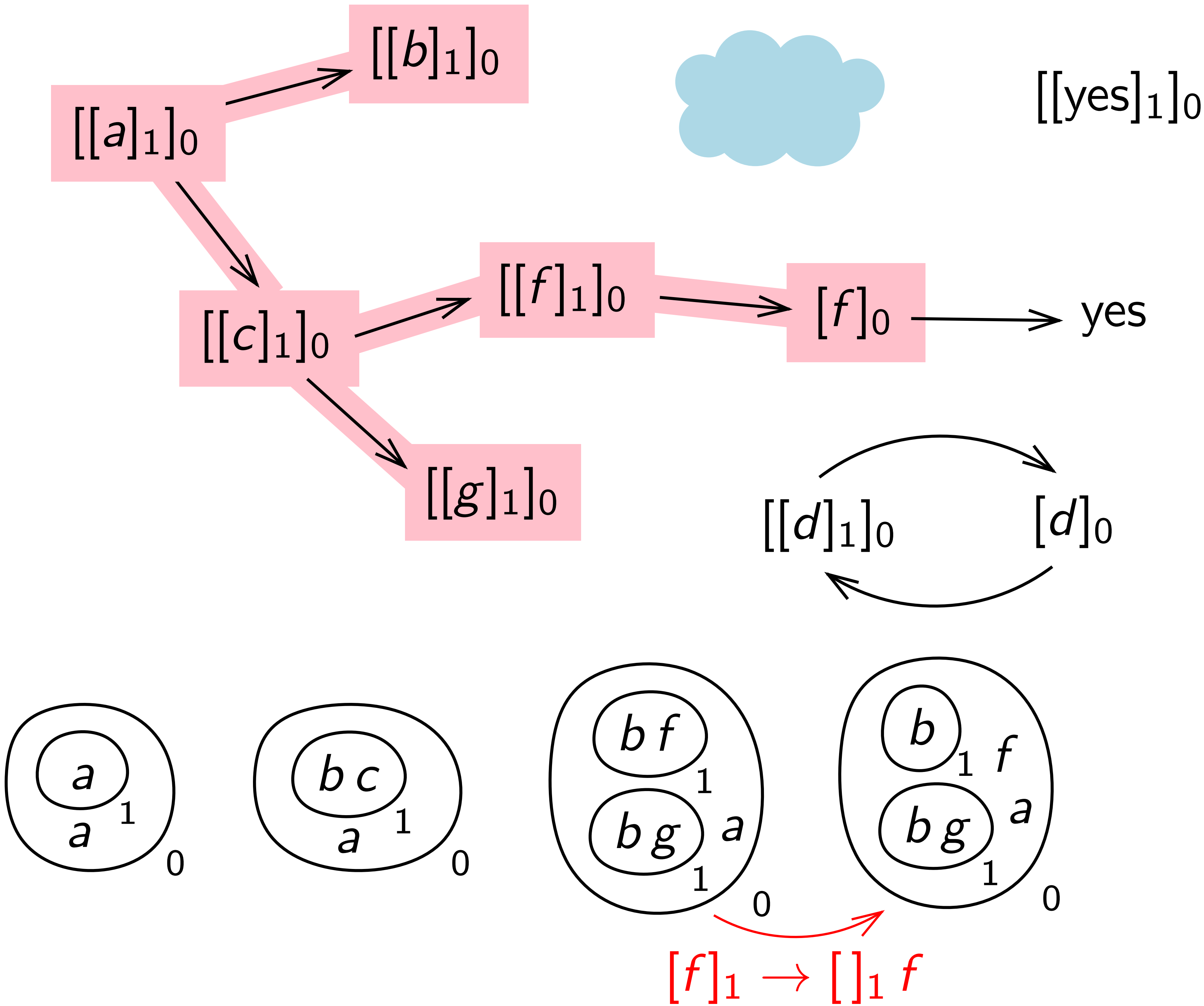
a
1
a
0



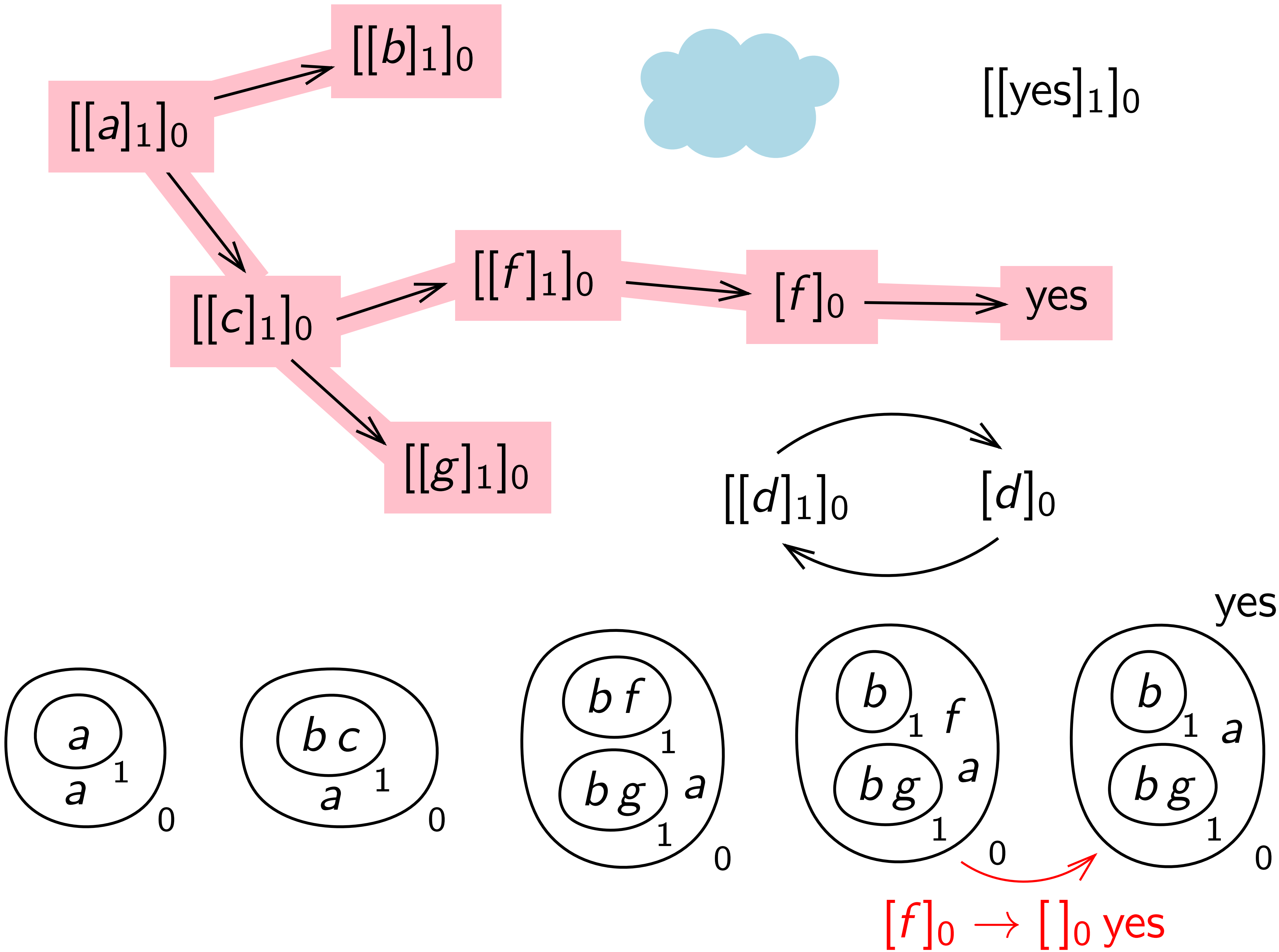
$[[a]_1]_0$
$[[b]_1]_0$
$[[yes]_1]_0$
$[[c]_1]_0$
$[[f]_1]_0$
$[f]_0$
yes
$[[g]_1]_0$
$[[d]_1]_0$
$[d]_0$
$[a \to bc]_1$
a a 1 0
b c a 1 0



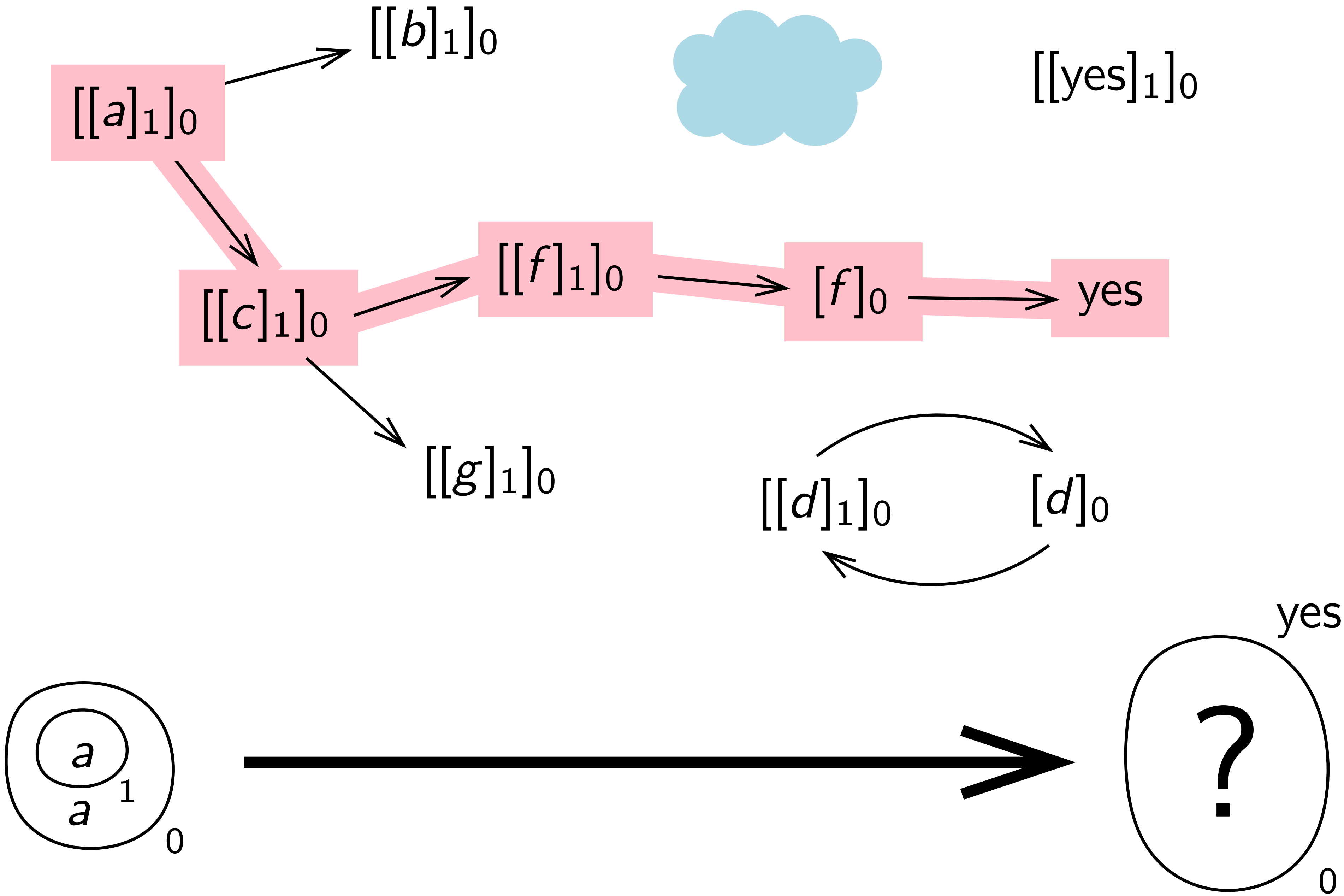
[[a]1]0
[[b]1]0
[[c]1]0
[[f]1]0
[[g]1]0
[[yes]1]0
[f]0
yes
[[d]1]0
[d]0
[c]1 → [f]1 [g]1
a
a 1
0
b c
a 1
0
b f
1
b g
a
1
0



$[[a]_1]_0$
$[[b]_1]_0$
$[[c]_1]_0$
$[[f]_1]_0$
$[f]_0$
yes
$[[g]_1]_0$
$[[yes]_1]_0$
$[[d]_1]_0$
$[d]_0$
$[f]_1 \to [\,]_1\, f$



$[[a]_1]_0$
$[[b]_1]_0$
$[[\text{yes}]_1]_0$
$[[c]_1]_0$
$[[f]_1]_0$
$[f]_0$
yes
$[[g]_1]_0$
$[[d]_1]_0$
$[d]_0$
yes
$[f]_0 \to [\,]_0\ \text{yes}$



$[[b]_1]_0$
$[[yes]_1]_0$
$[[a]_1]_0$
$[[f]_1]_0$
$[f]_0$
yes
$[[c]_1]_0$
$[[g]_1]_0$
$[[d]_1]_0$
$[d]_0$
yes
a
1
a
0
?
0

Dependency graphs can be constructed and explored in polynomial time

Dependency graphs can be constructed and explored in polynomial time

The P conjecture is true for P systems without dissolution rules

Dependency graphs can be constructed and explored in polynomial time

The P conjecture is true for P systems without dissolution rules

And false for those with both non-elementary division and dissolution (they reach **PSPACE**)

In P systems without dissolution the result actually depends on one object

In P systems without dissolution the result actually depends on one object

Consider a restricted version of P systems with dissolution:

- monodirectional
- shallow
- deterministic

The result of the computation depends on at most two objects

$$\mathcal{C}_0 \longrightarrow \mathcal{C}_1 \longrightarrow \mathcal{C}_2 \dashrightarrow \mathcal{C}_{t-1} \longrightarrow \mathcal{C}_t$$

The result of the computation depends on at most two objects

for each computation

$$\mathcal{C}_0 \longrightarrow \mathcal{C}_1 \longrightarrow \mathcal{C}_2 \dashrightarrow \mathcal{C}_{t-1} \longrightarrow \mathcal{C}_t$$

The result of the computation depends on at most two objects

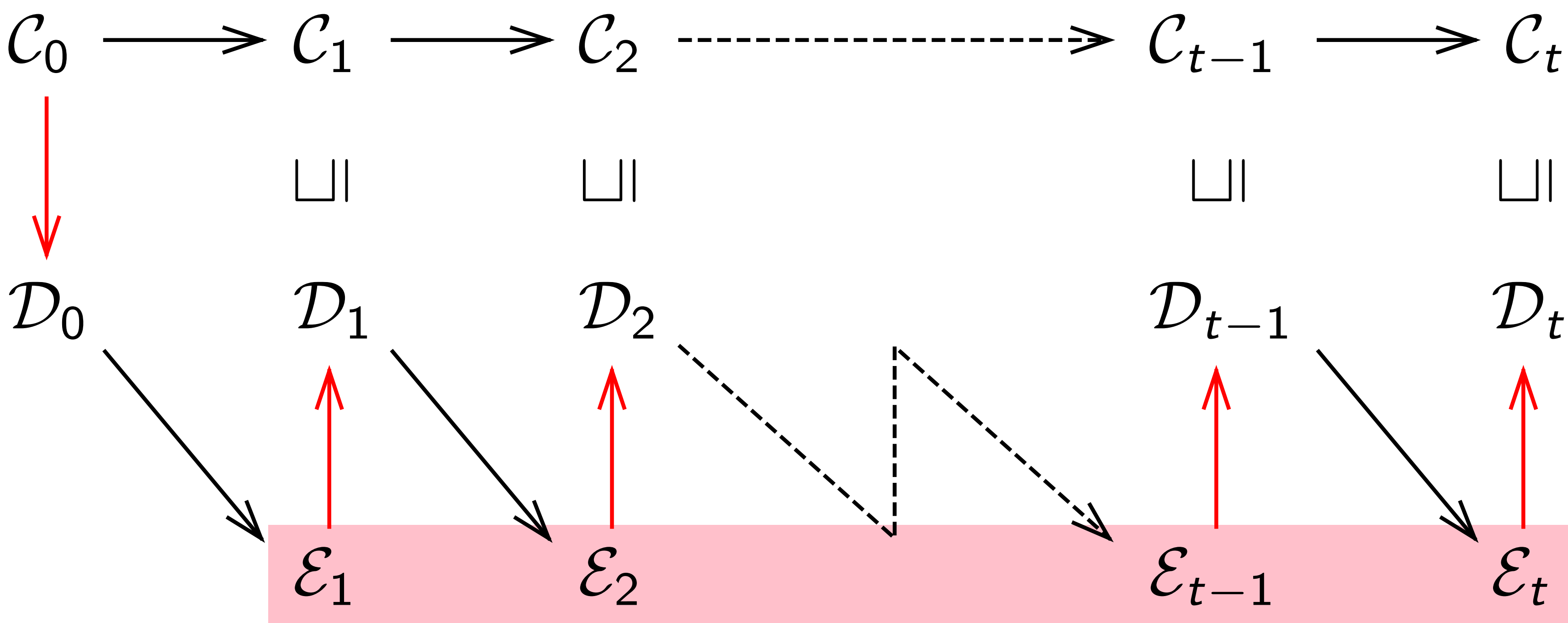
$$\begin{tikzcd}
\mathcal{C}_0 \arrow[r] \arrow[d] & \mathcal{C}_1 \arrow[r] \arrow[d, phantom, sloped, "\sqsupseteq"] & \mathcal{C}_2 \arrow[r, dashed] \arrow[d, phantom, sloped, "\sqsupseteq"] & \mathcal{C}_{t-1} \arrow[r] \arrow[d, phantom, sloped, "\sqsupseteq"] & \mathcal{C}_t \arrow[d, phantom, sloped, "\sqsupseteq"] \\
\mathcal{D}_0 & \mathcal{D}_1 & \mathcal{D}_2 & \mathcal{D}_{t-1} & \mathcal{D}_t
\end{tikzcd}$$

there exists a sequence
of small configurations

$[[a\ b]_k]_h$ $[[a]_k]_h$

$[a]_h$ yes no

The result of the computation depends on at most two objects



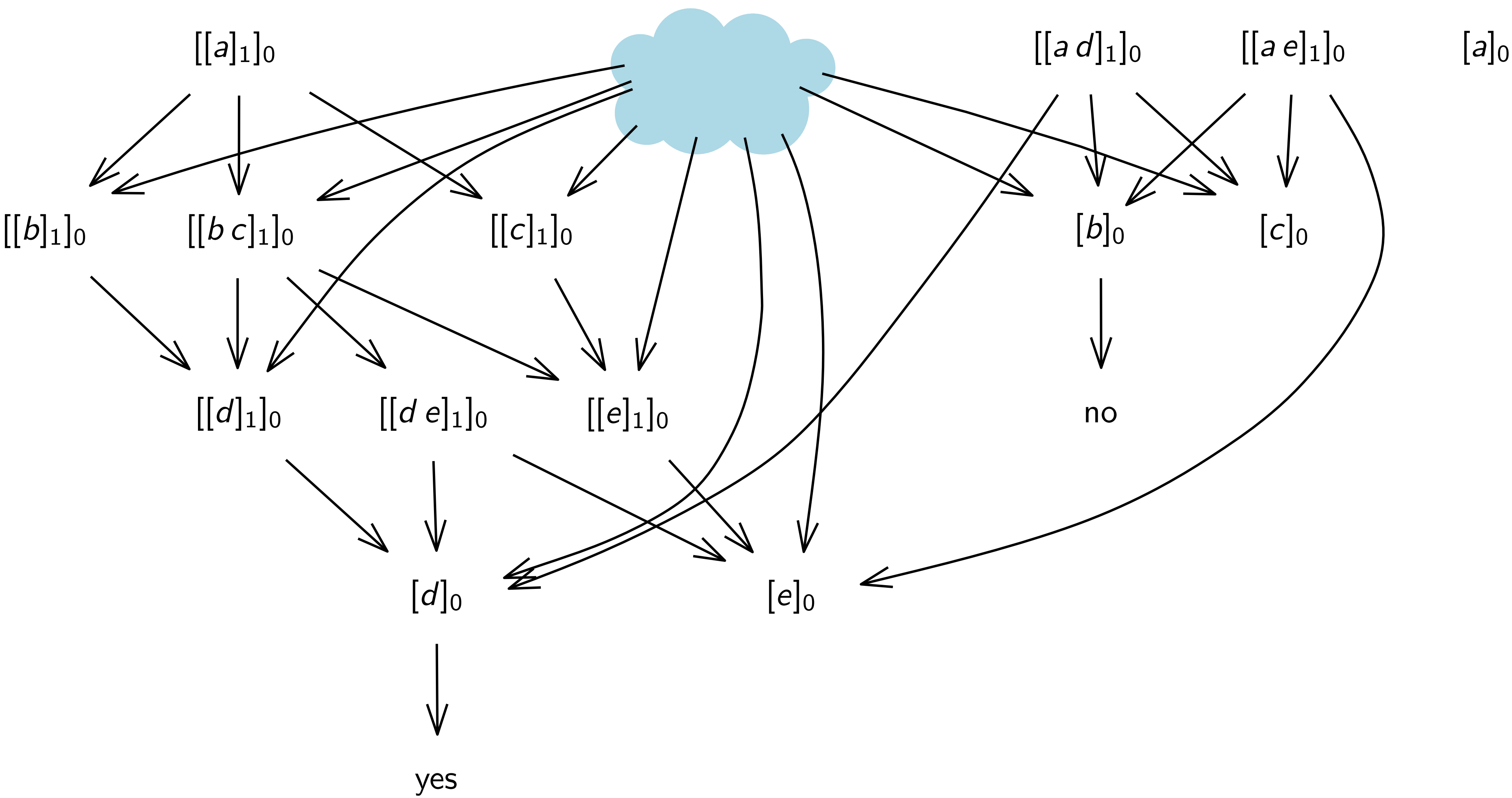
and there exists another
sequence of configurations

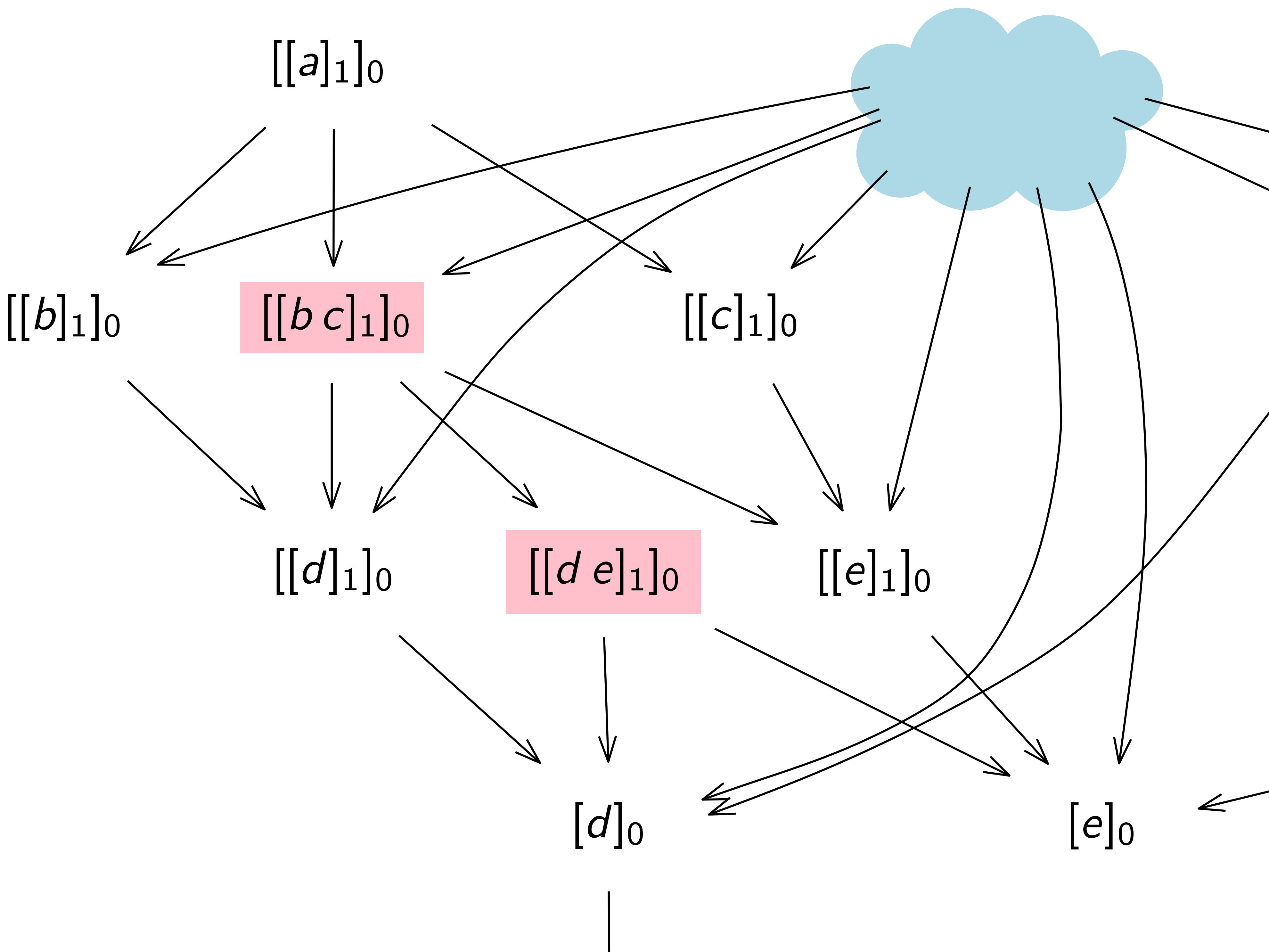
The result of the computation depends on at most two objects

$$
\begin{tikzcd}
\mathcal{C}_0 \arrow[r] \arrow[d] \arrow[drrrrr, bend right] & \mathcal{C}_1 \arrow[r] \arrow[d, phantom, "\rotatebox{90}{$\sqsubseteq$}"] & \mathcal{C}_2 \arrow[rr, dashed] \arrow[d, phantom, "\rotatebox{90}{$\sqsubseteq$}"] & & \mathcal{C}_{t-1} \arrow[r] \arrow[d, phantom, "\rotatebox{90}{$\sqsubseteq$}"] & \mathcal{C}_t \arrow[d, phantom, "\rotatebox{90}{$\sqsubseteq$}"] \\
\mathcal{D}_0 \arrow[dr] & \mathcal{D}_1 \arrow[dr] & \mathcal{D}_2 \arrow[drr, dashed] & & \mathcal{D}_{t-1} \arrow[dr] & \mathcal{D}_t \\
& \mathcal{E}_1 \arrow[u] & \mathcal{E}_2 \arrow[u] & & \mathcal{E}_{t-1} \arrow[u] & \mathcal{E}_t \arrow[u]
\end{tikzcd}
$$

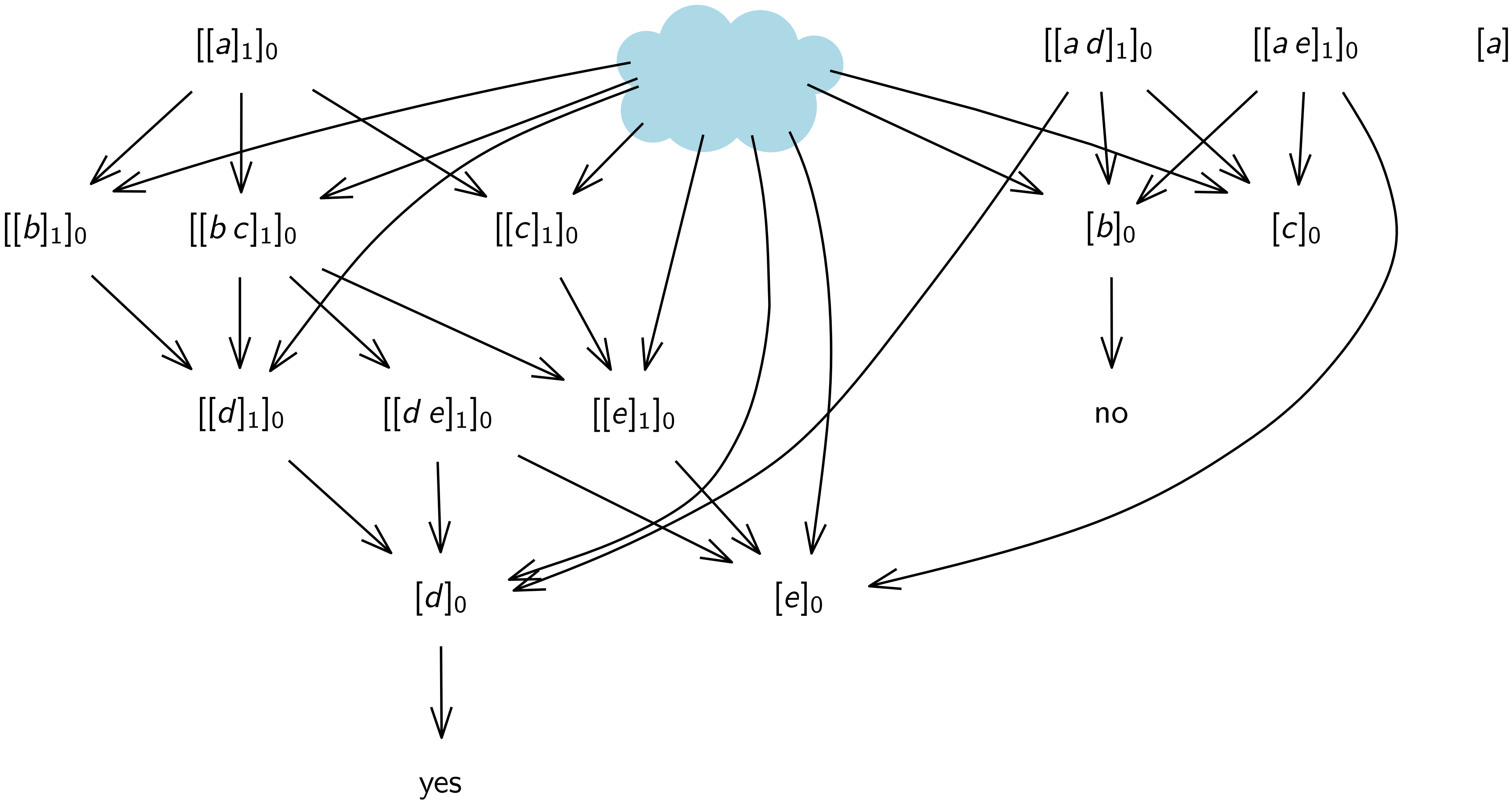
such that the diagram
"commutes" (same result)

Generalised dependency graphs





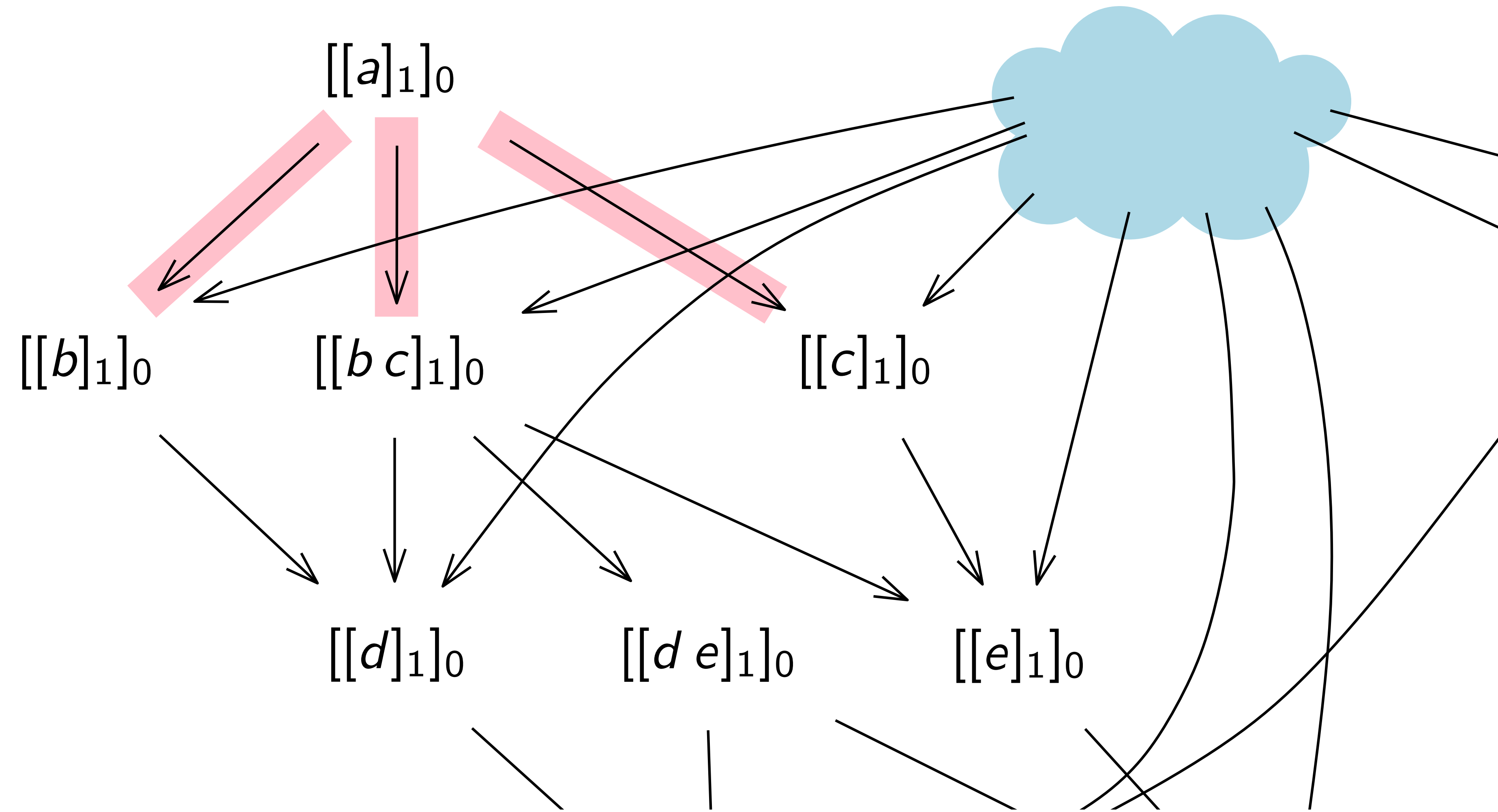
[[a]1]0
[[b]1]0
[[b c]1]0
[[c]1]0
[[d]1]0
[[d e]1]0
[[e]1]0
[d]0
[e]0



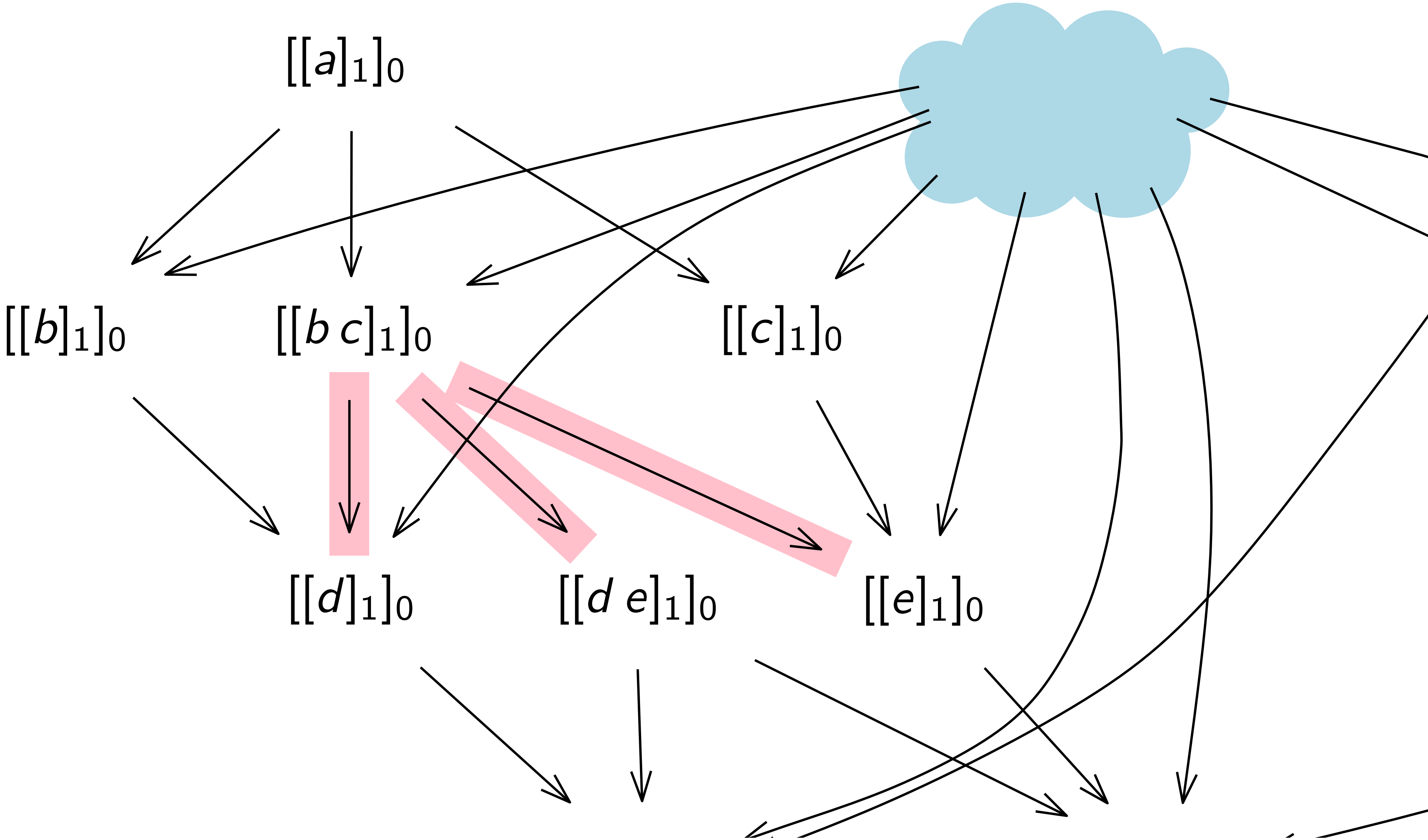
$[a \rightarrow bc]_1$ $\quad [d]_0 \rightarrow [\,]_0$ yes $\quad [e]_1 \rightarrow e \quad [f]_1 \rightarrow [g]_1\,[h]_1$

$[b \rightarrow d]_1$ $\quad [b]_0 \rightarrow [\,]_0$ no $\quad [g]_1 \rightarrow g$

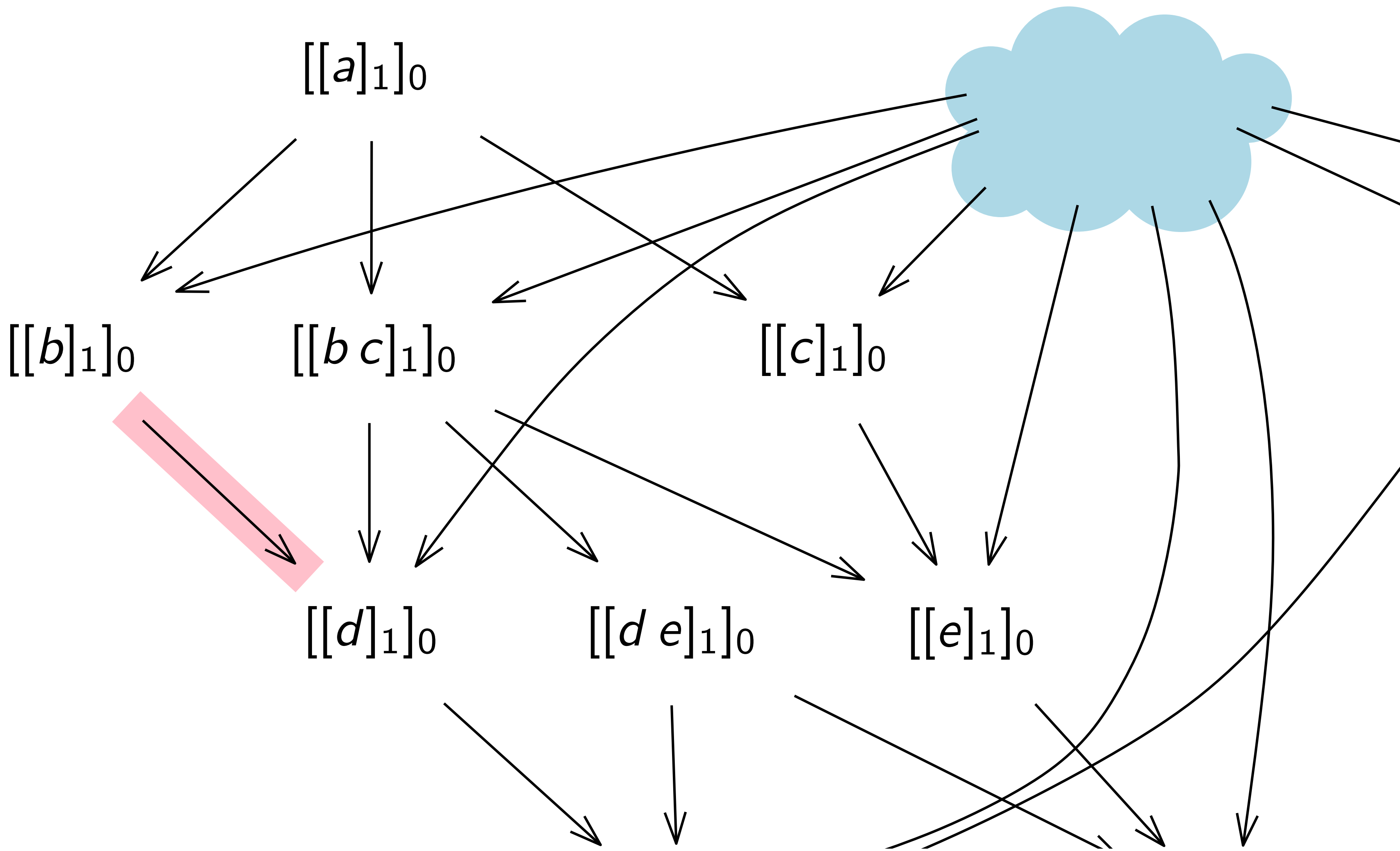
$[c \rightarrow e]_1$ $\quad [d]_1 \rightarrow d \quad [h]_1 \rightarrow h$



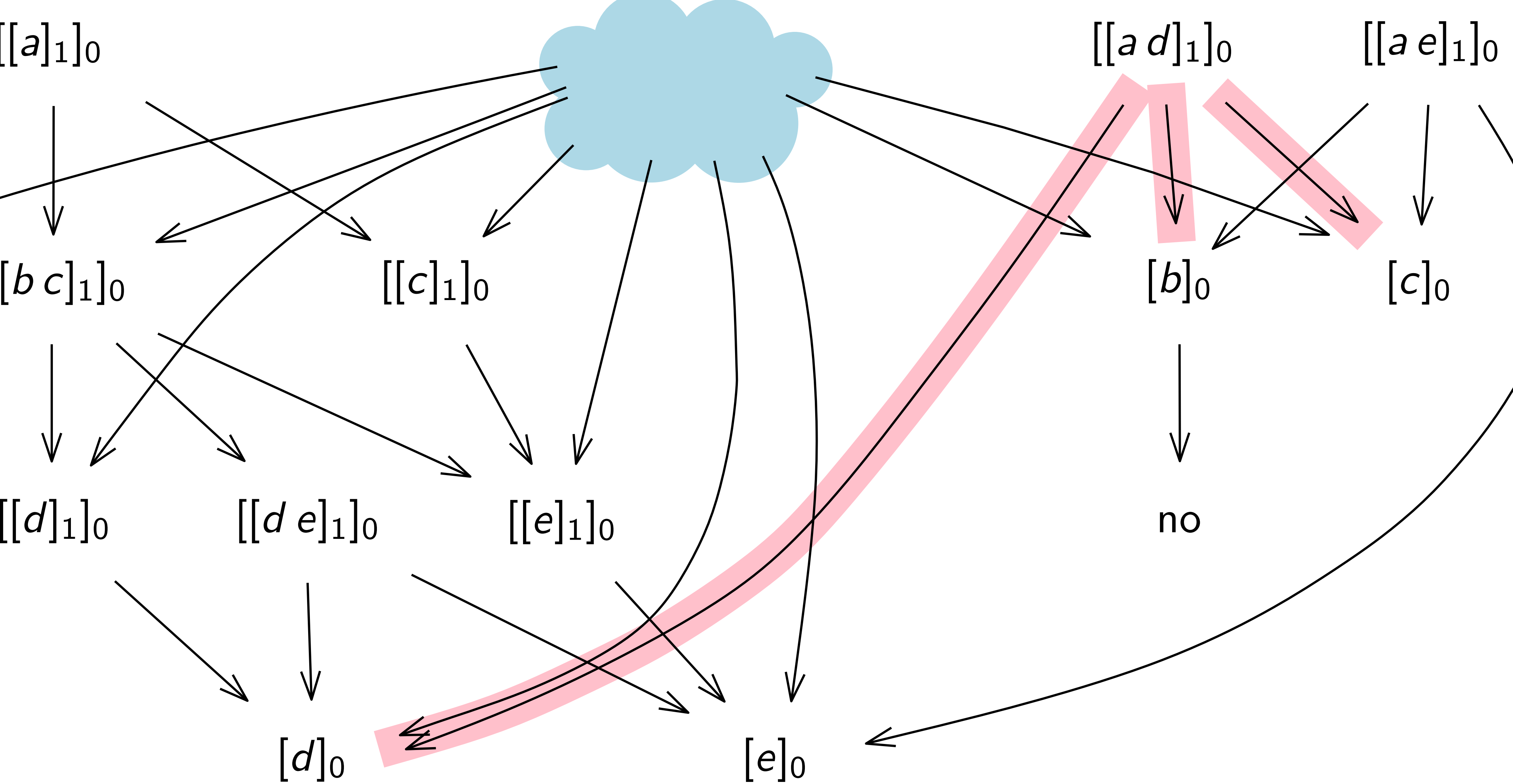
$[a \to bc]_1$	$[d]_0 \to [\ ]_0$ yes	$[e]_1 \to e$	$[f]_1 \to [g]_1\,[h]_1$
$[b \to d]_1$	$[b]_0 \to [\ ]_0$ no	$[g]_1 \to g$	
$[c \to e]_1$	$[d]_1 \to d$	$[h]_1 \to h$	



$[a \rightarrow bc]_1$	$[d]_0 \rightarrow [\,]_0$ yes	$[e]_1 \rightarrow e$	$[f]_1 \rightarrow [g]_1\,[h]_1$
$[b \rightarrow d]_1$	$[b]_0 \rightarrow [\,]_0$ no	$[g]_1 \rightarrow g$	
$[c \rightarrow e]_1$	$[d]_1 \rightarrow d$	$[h]_1 \rightarrow h$	



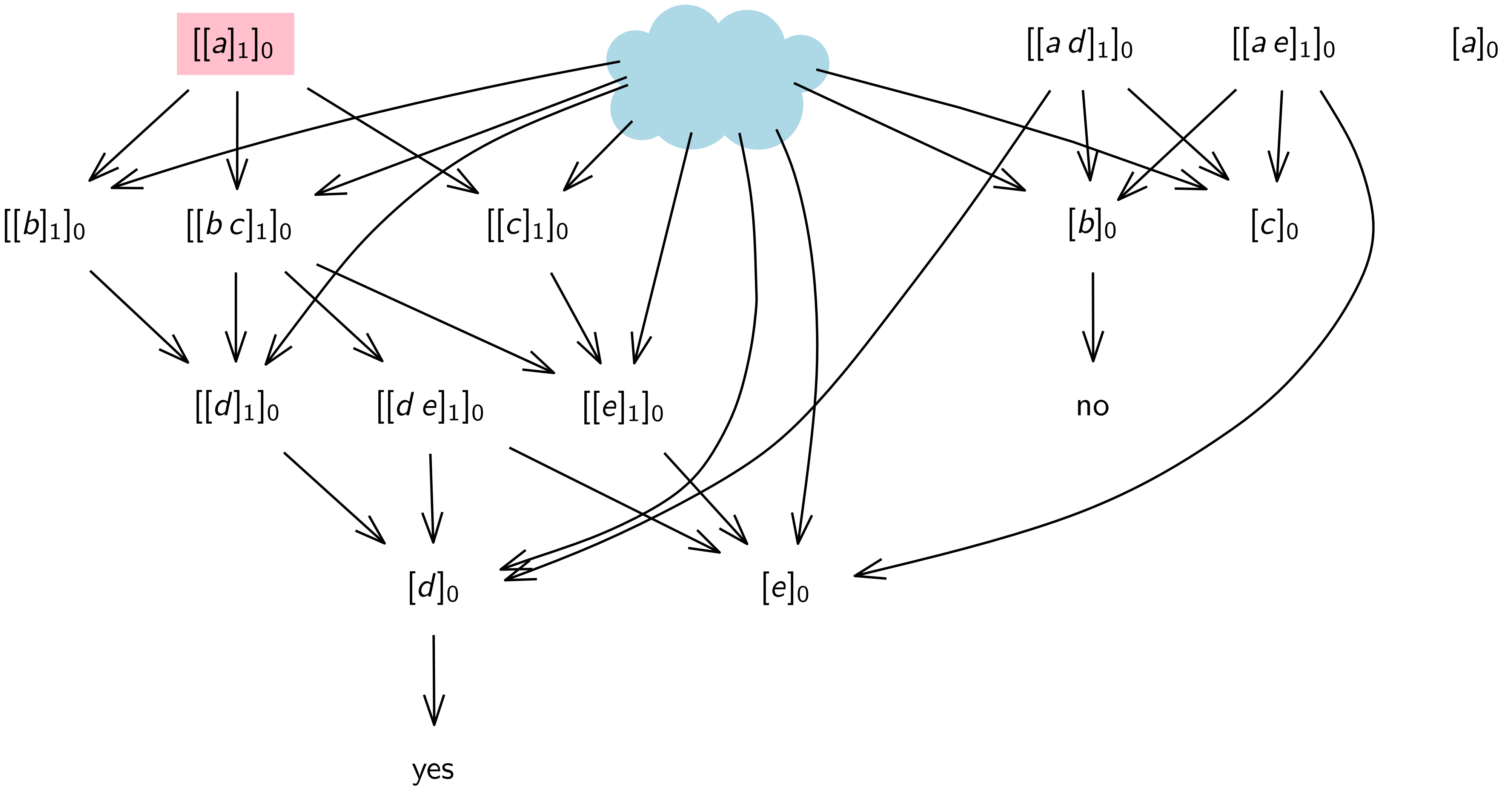
$[a \to bc]_1$	$[d]_0 \to [\ ]_0$ yes	$[e]_1 \to e$	$[f]_1 \to [g]_1\,[h]_1$
$[b \to d]_1$	$[b]_0 \to [\ ]_0$ no	$[g]_1 \to g$	
$[c \to e]_1$	$[d]_1 \to d$	$[h]_1 \to h$	

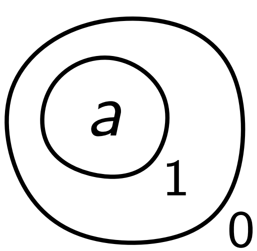


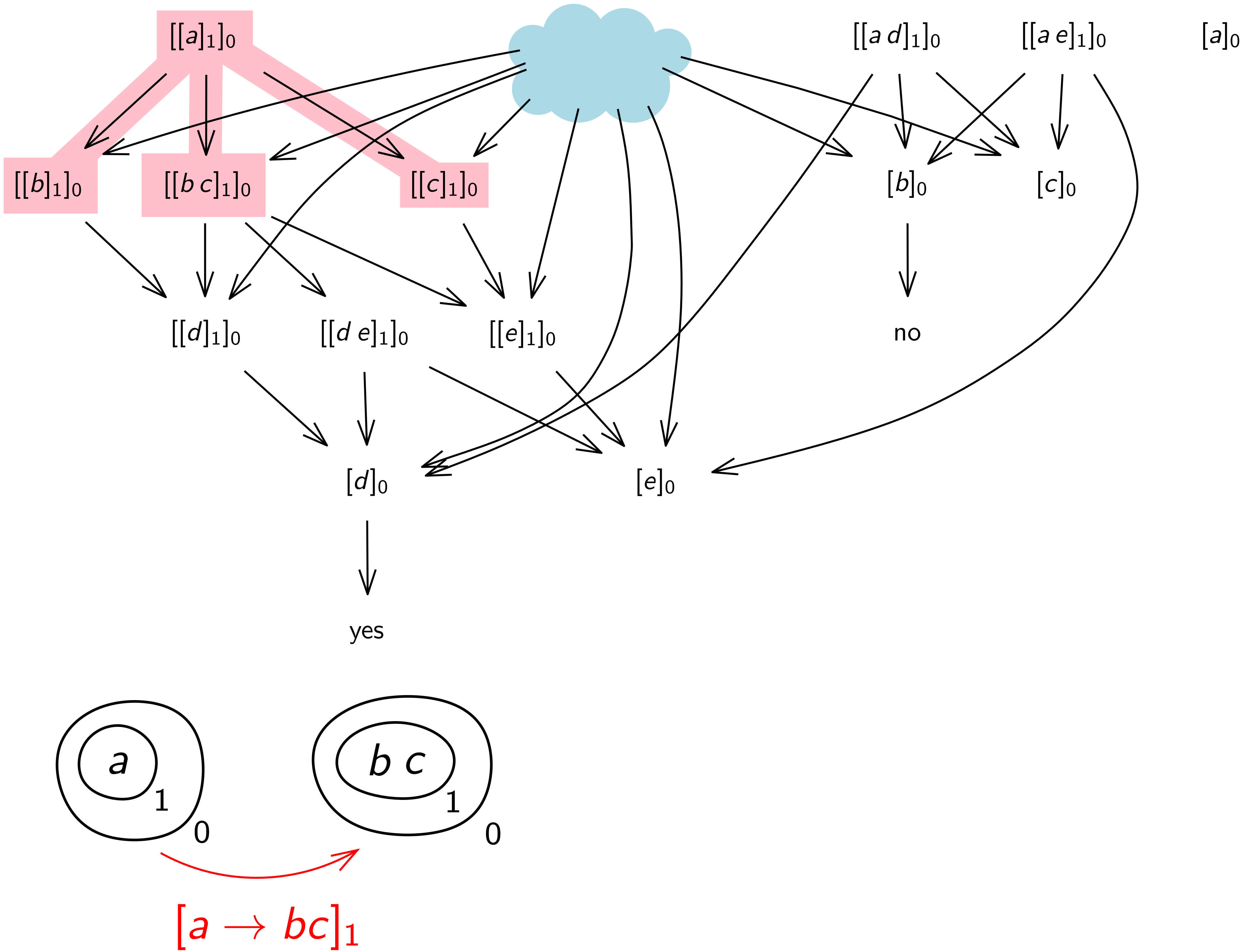
$[a \to bc]_1$ $\quad$ $[d]_0 \to [\,]_0$ yes $\quad$ $[e]_1 \to e$ $\quad$ $[f]_1 \to [g]_1\,[h]_1$

$[b \to d]_1$ $\quad$ $[b]_0 \to [\,]_0$ no $\quad$ $[g]_1 \to g$

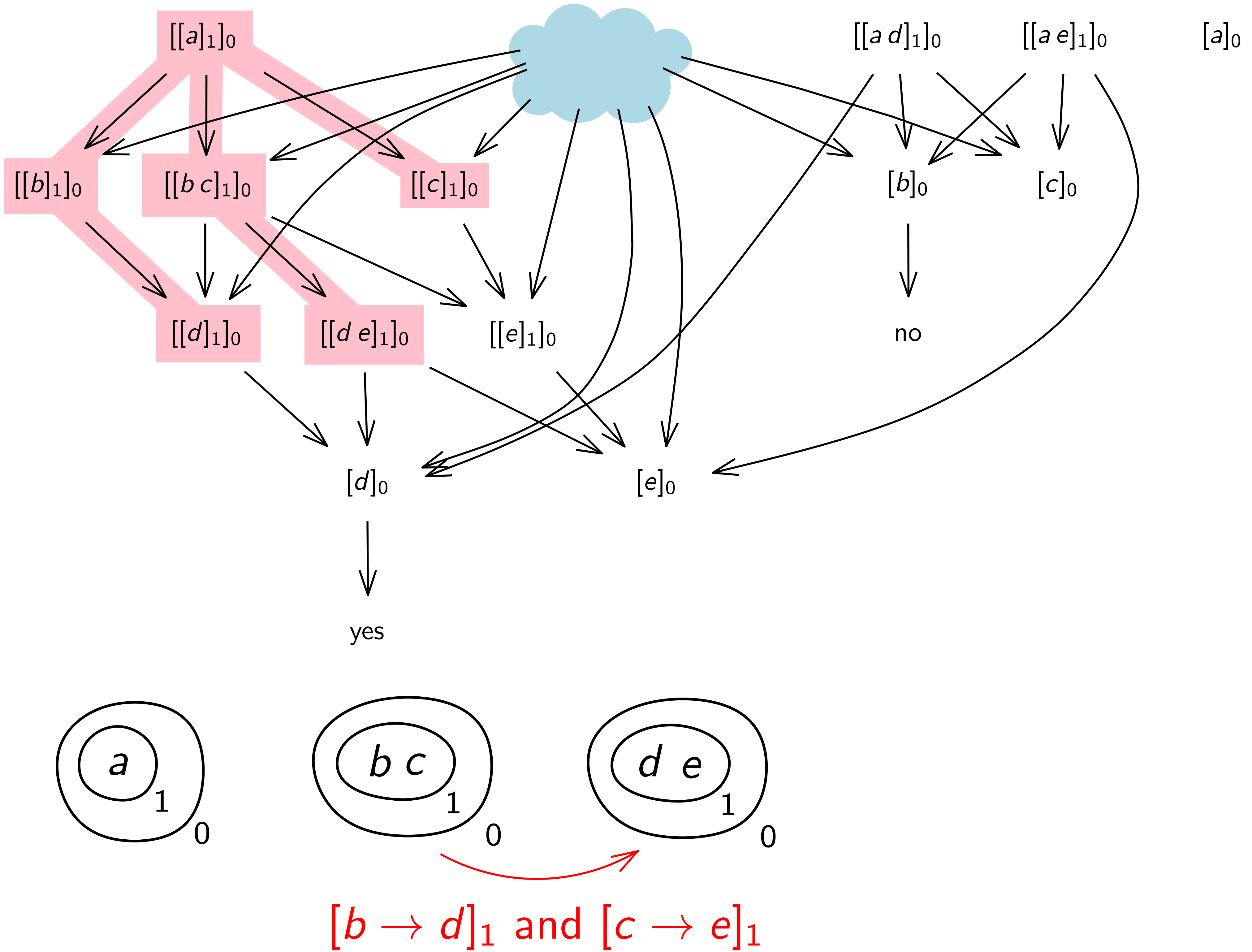
$[c \to e]_1$ $\quad$ $[d]_1 \to d$ $\quad$ $[h]_1 \to h$

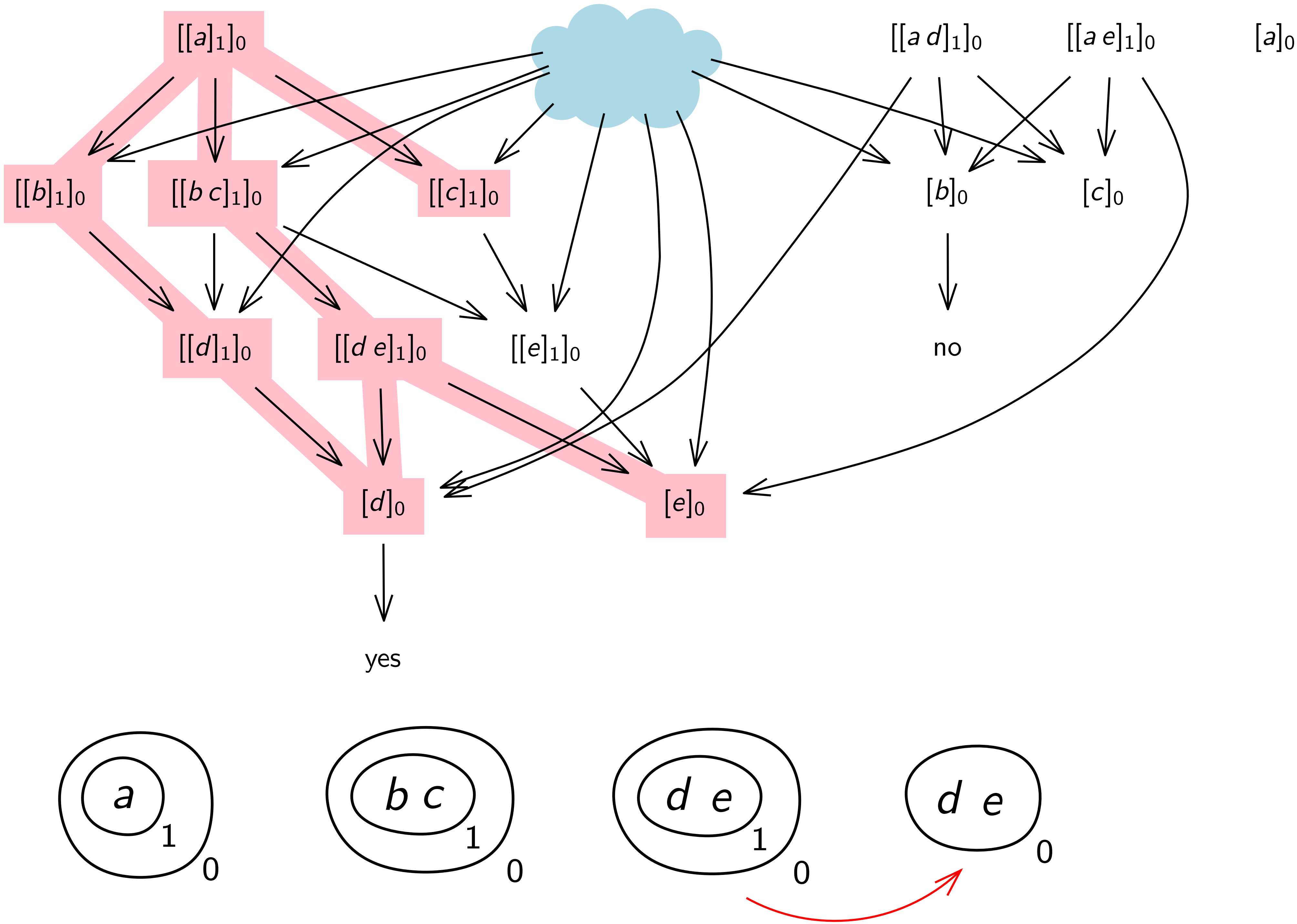

[[a]1]0
[[a d]1]0
[[a e]1]0
[a]0
[[b]1]0
[[b c]1]0
[[c]1]0
[b]0
[c]0
[[d]1]0
[[d e]1]0
[[e]1]0
no
[d]0
[e]0
yes


a
1
0

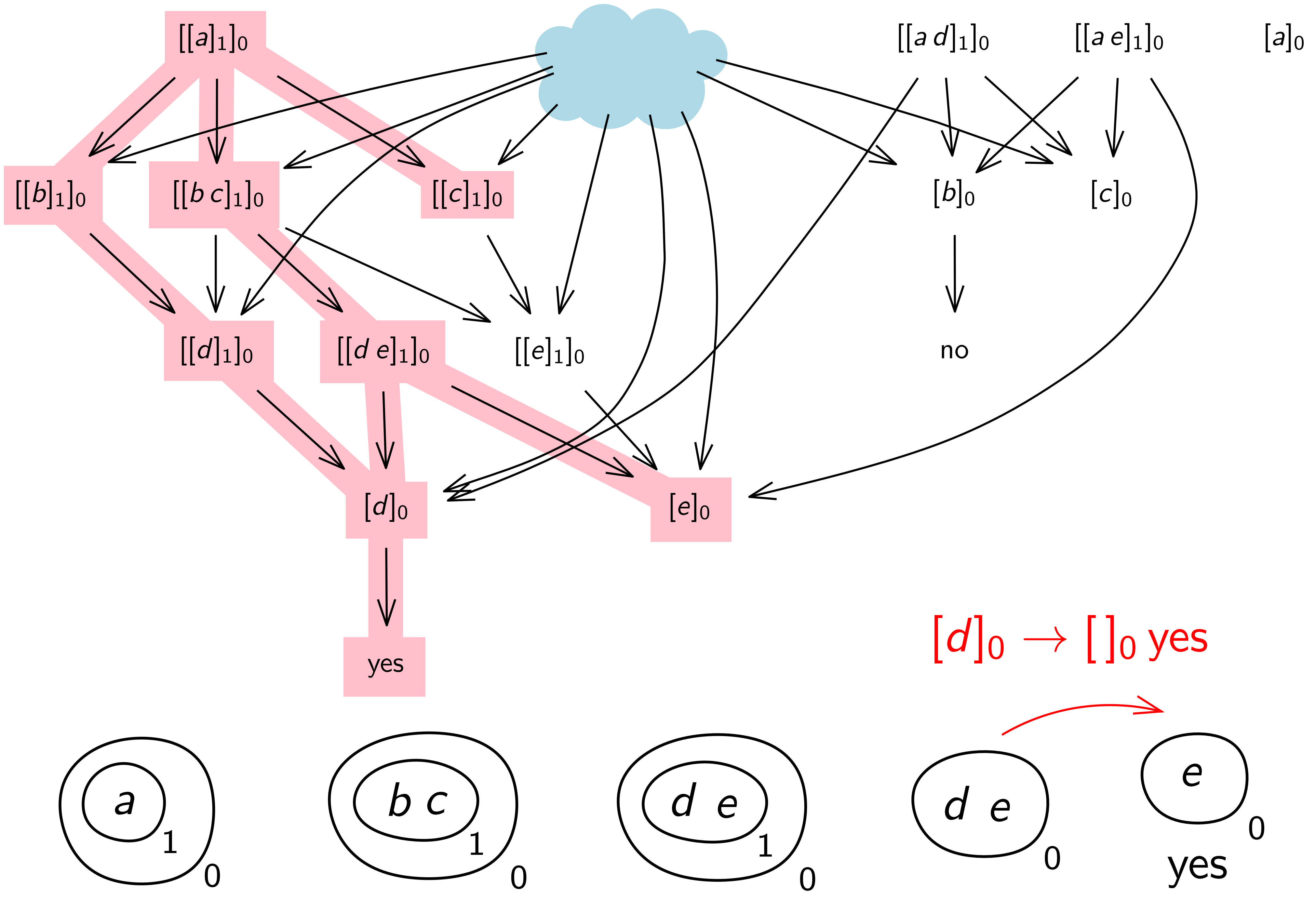


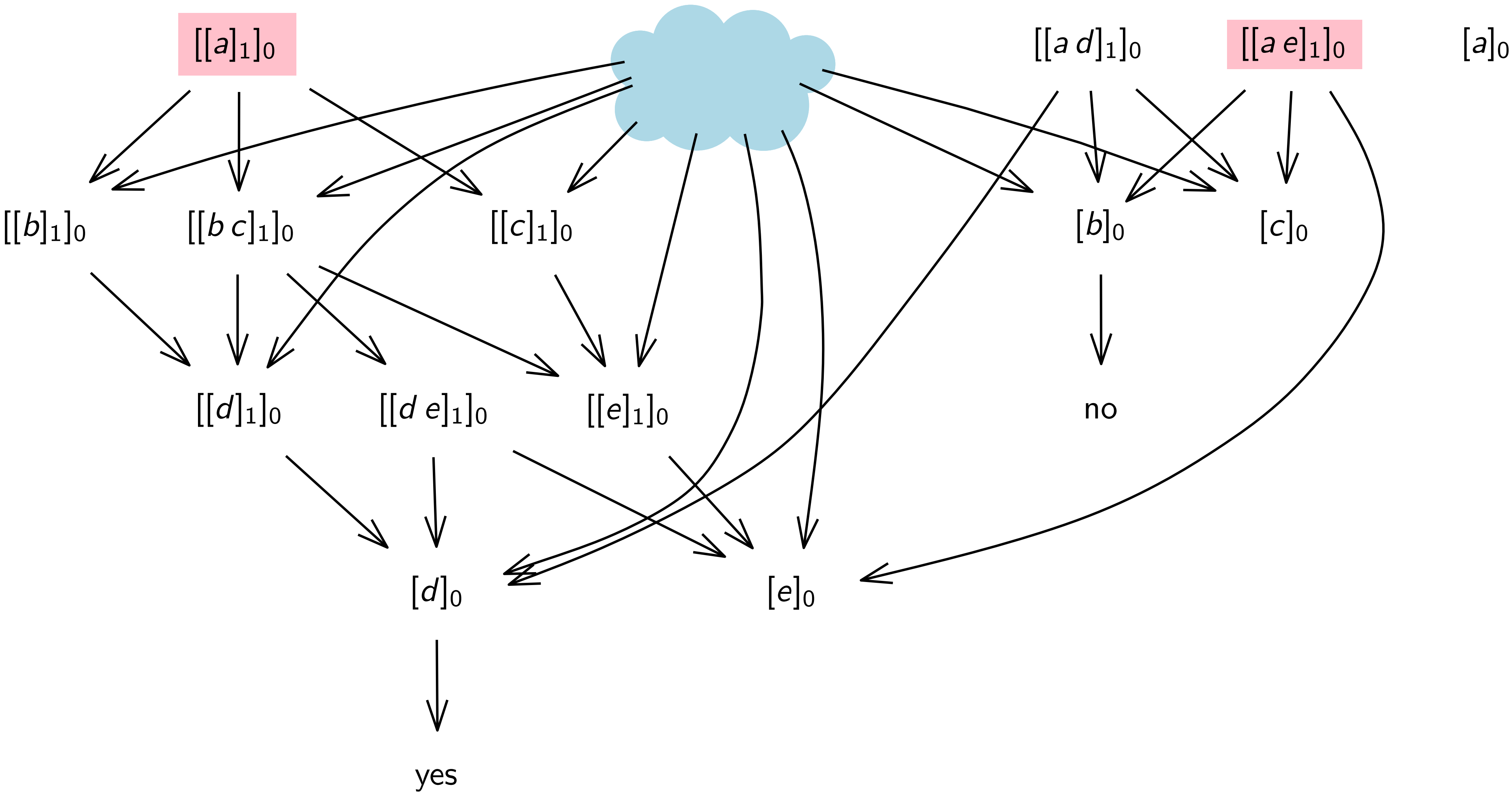
$[[a]_1]_0$
$[[a\ d]_1]_0$
$[[a\ e]_1]_0$
$[a]_0$
$[[b]_1]_0$
$[[b\ c]_1]_0$
$[[c]_1]_0$
$[b]_0$
$[c]_0$
$[[d]_1]_0$
$[[d\ e]_1]_0$
$[[e]_1]_0$
no
$[d]_0$
$[e]_0$
yes
a
1
0
$b\ c$
1
0
$[a \to bc]_1$


$[[a]_1]_0$
$[[a\ d]_1]_0$
$[[a\ e]_1]_0$
$[a]_0$
$[[b]_1]_0$
$[[b\ c]_1]_0$
$[[c]_1]_0$
$[b]_0$
$[c]_0$
$[[d]_1]_0$
$[[d\ e]_1]_0$
$[[e]_1]_0$
no
$[d]_0$
$[e]_0$
yes
a 1 0
$b\ c$ 1 0
$d\ e$ 1 0
$[b \to d]_1$ and $[c \to e]_1$

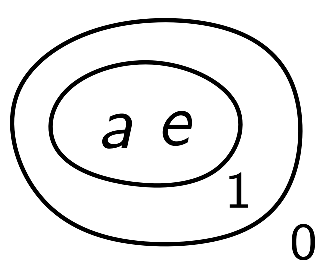


$[d]_1 \rightarrow d$ or $[e]_1 \rightarrow e$

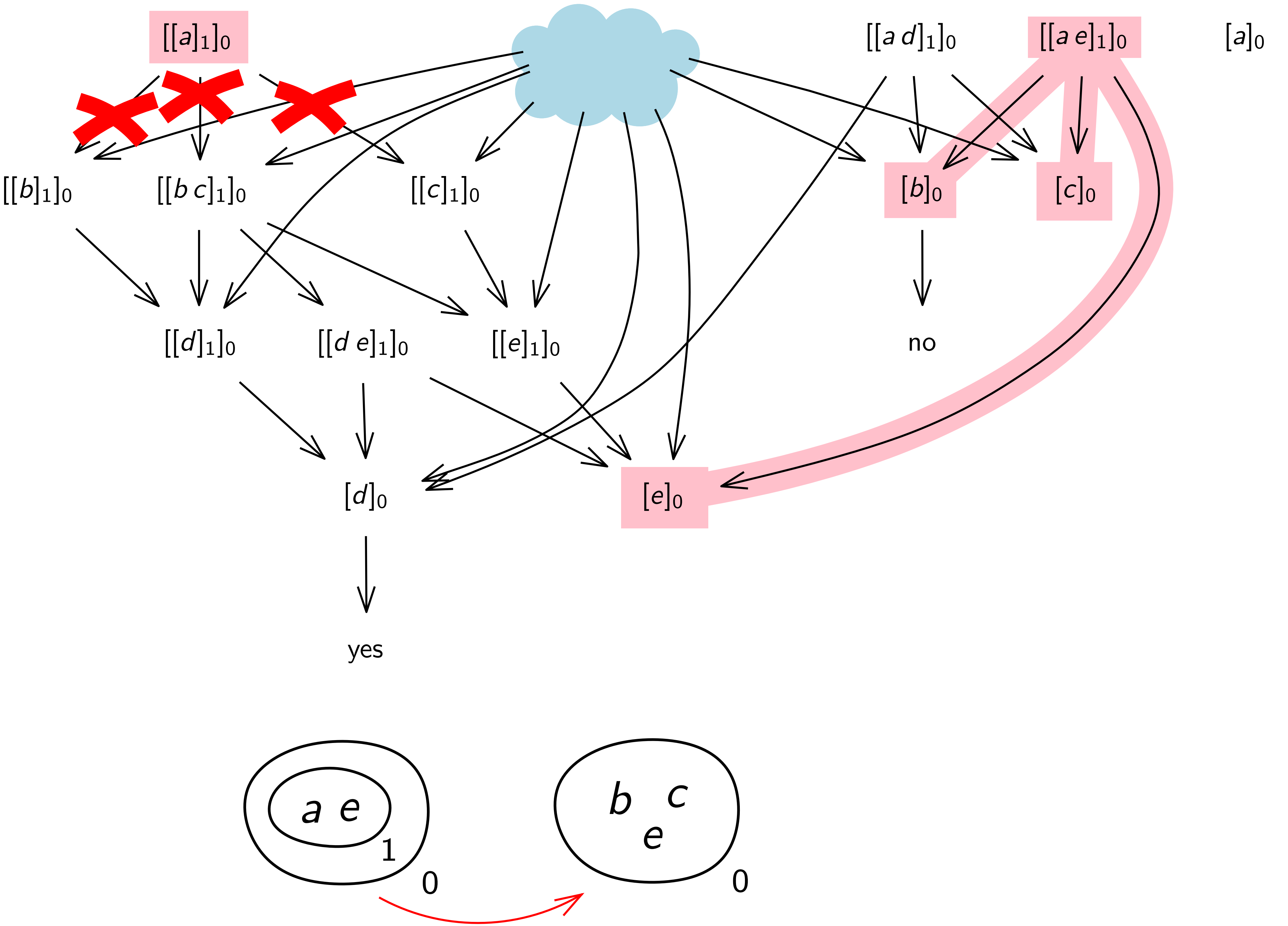

$[[a]_1]_0$
$[[a\ d]_1]_0$
$[[a\ e]_1]_0$
$[a]_0$
$[[b]_1]_0$
$[[b\ c]_1]_0$
$[[c]_1]_0$
$[b]_0$
$[c]_0$
$[[d]_1]_0$
$[[d\ e]_1]_0$
$[[e]_1]_0$
no
$[d]_0$
$[e]_0$
yes
$[d]_0 \rightarrow [\ ]_0$ yes
a 1 0
$b\ c$ 1 0
$d\ e$ 1 0
$d\ e$ 0
e 0
yes



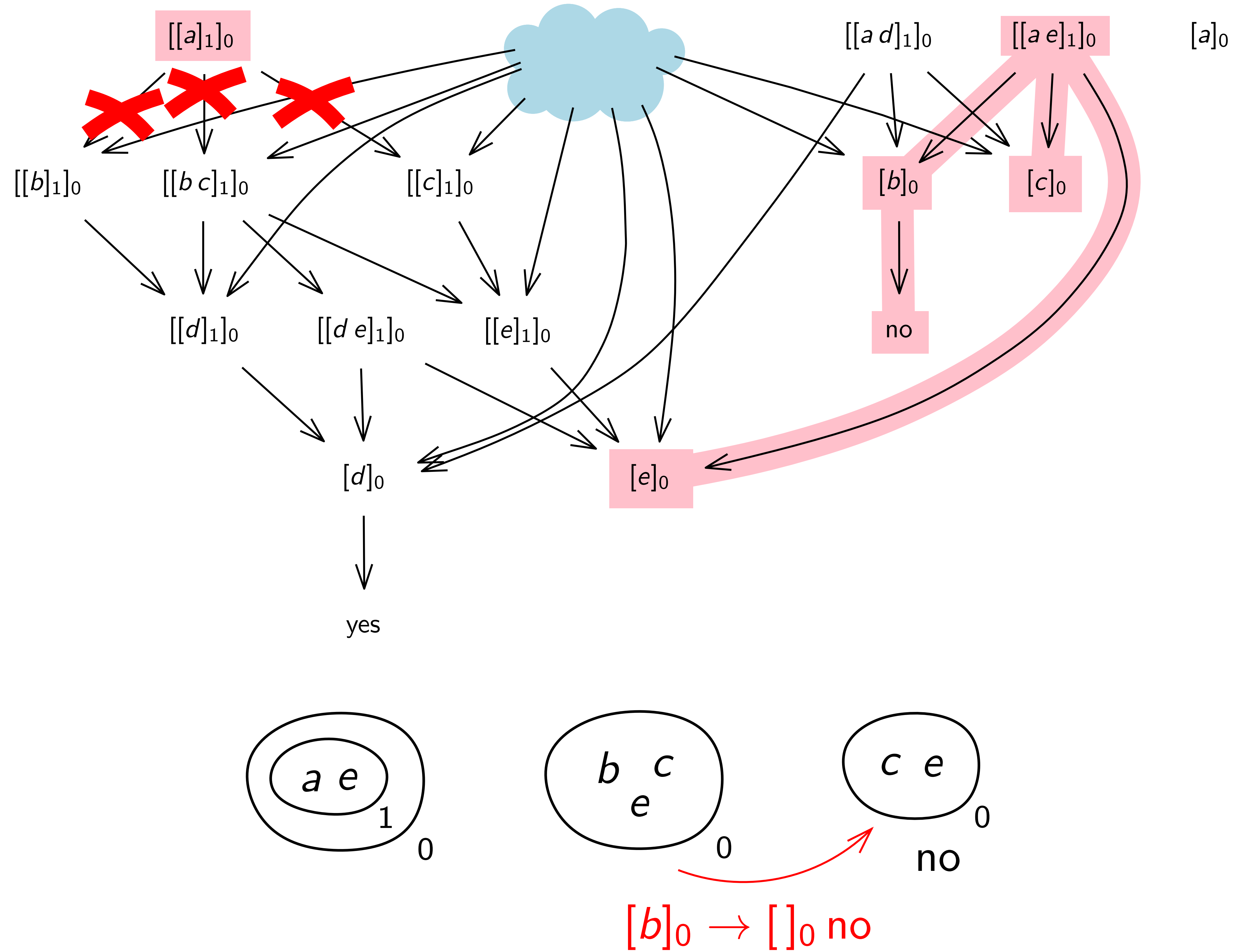
[[a]1]0
[[a d]1]0
[[a e]1]0
[a]0
[[b]1]0
[[b c]1]0
[[c]1]0
[b]0
[c]0
[[d]1]0
[[d e]1]0
[[e]1]0
no
[d]0
[e]0
yes



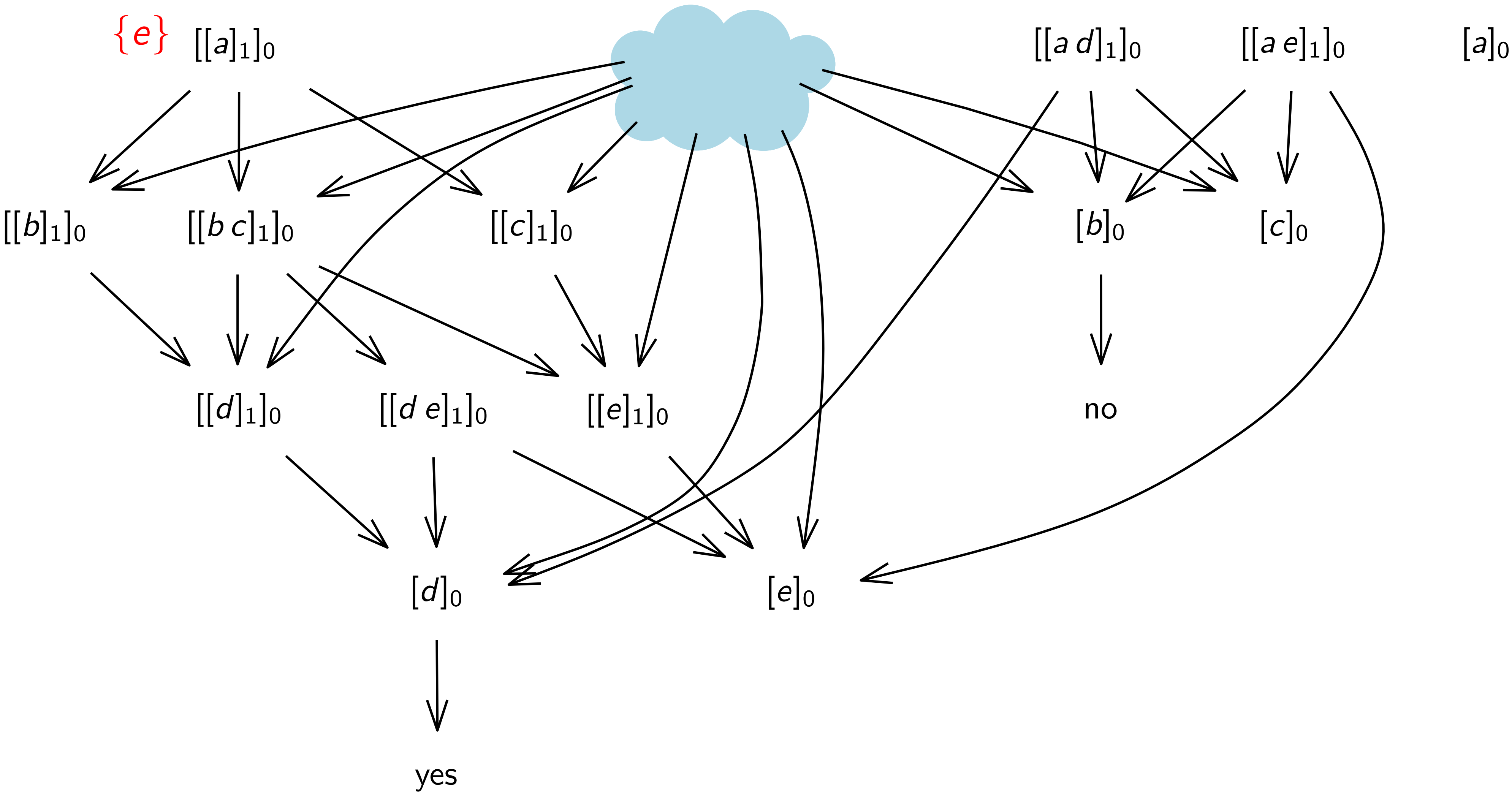
a e
1
0



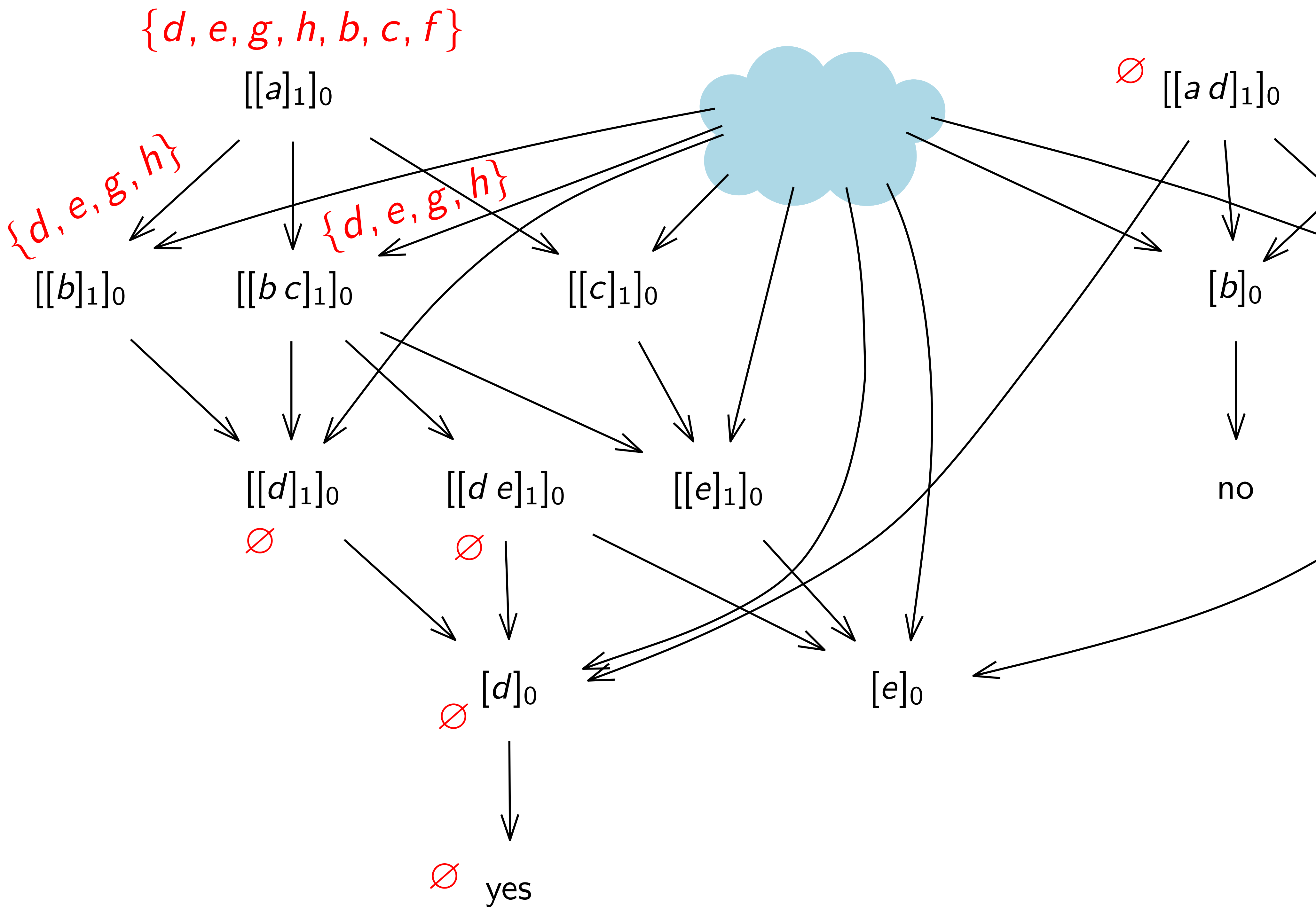
$[a \to bc]_1$ and $[e]_1 \to e$



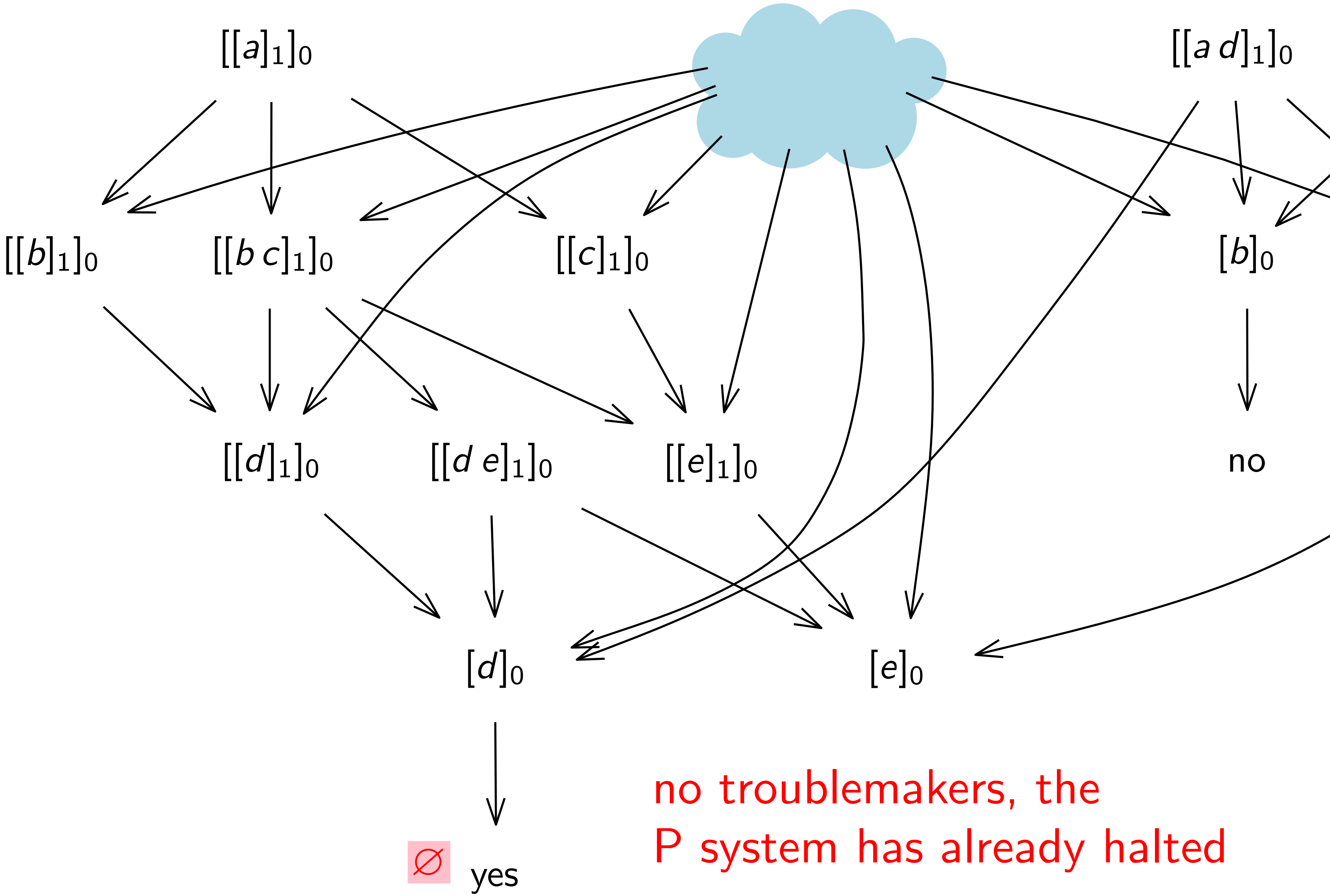
[[a]1]0
[[a d]1]0
[[a e]1]0
[a]0
[[b]1]0
[[b c]1]0
[[c]1]0
[b]0
[c]0
[[d]1]0
[[d e]1]0
[[e]1]0
no
[d]0
[e]0
yes
a e
1
0
b c
e
0
c e
0
no
[b]0 → [ ]0 no



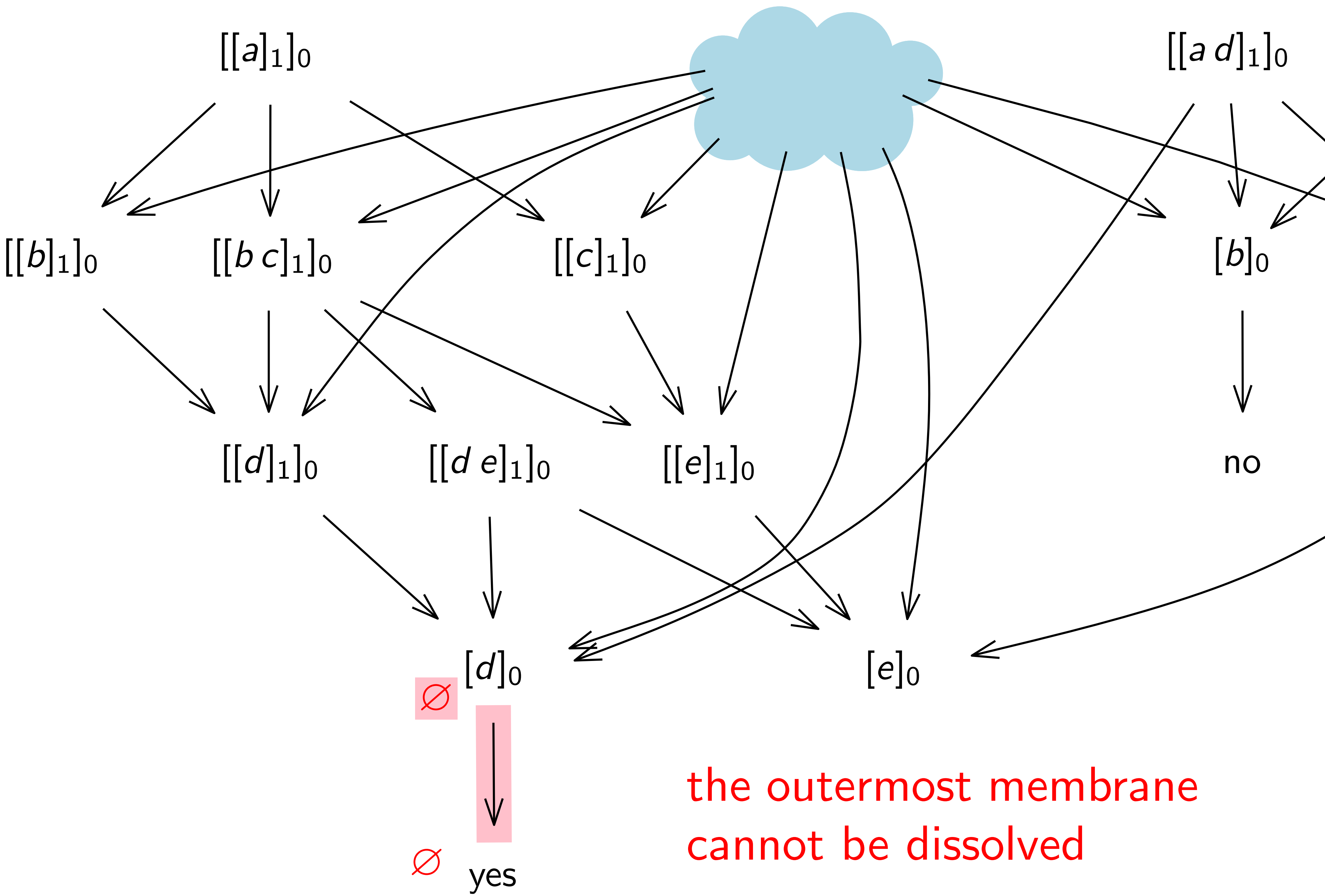
e is a troublemaker for $[[a]_1]_0$



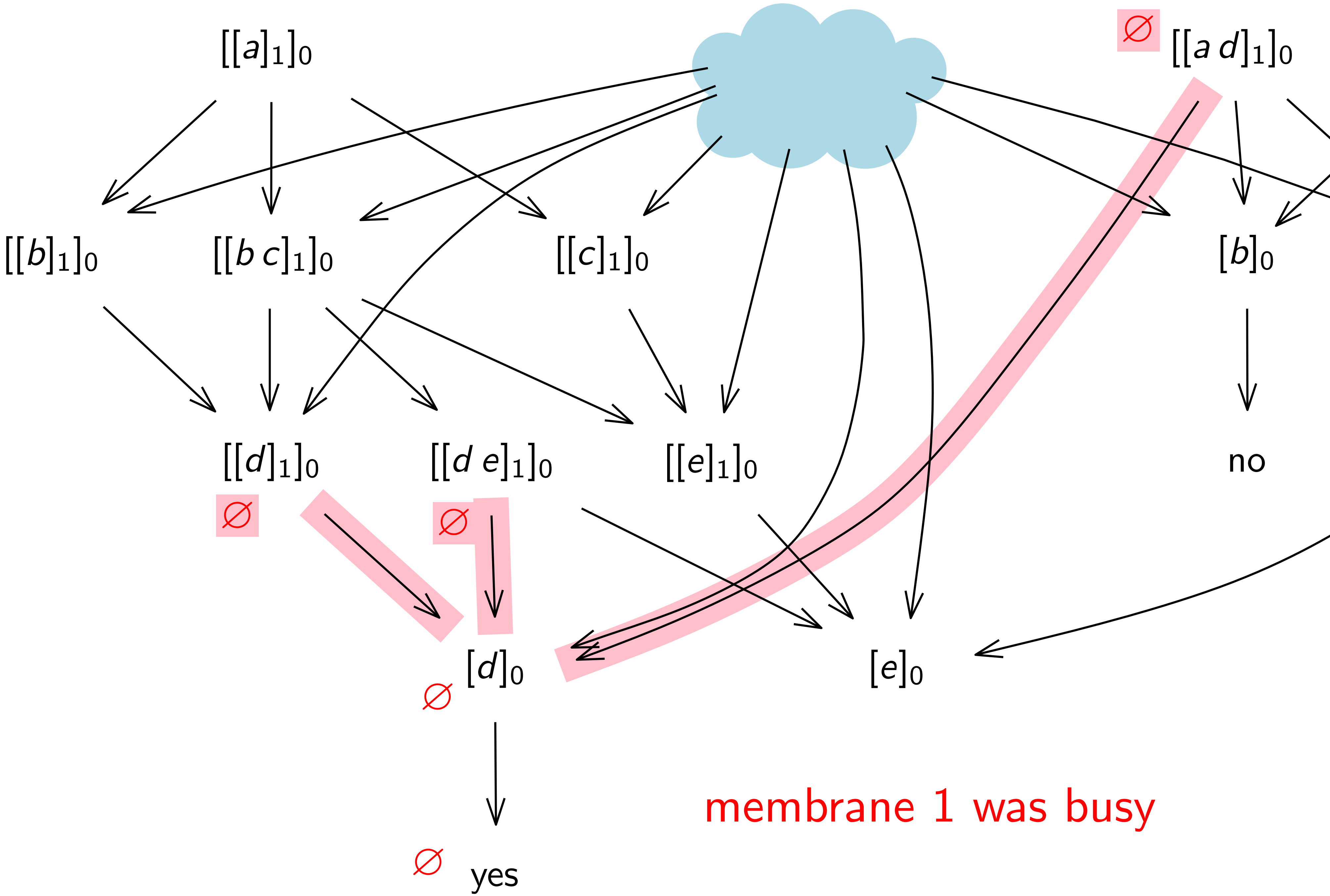
{d, e, g, h, b, c, f}
[[a]1]0
∅
[[a d]1]0
{d, e, g, h}
{d, e, g, h}
[[b]1]0
[[b c]1]0
[[c]1]0
[b]0
[[d]1]0
[[d e]1]0
[[e]1]0
no
∅
∅
[d]0
[e]0
∅
∅
yes

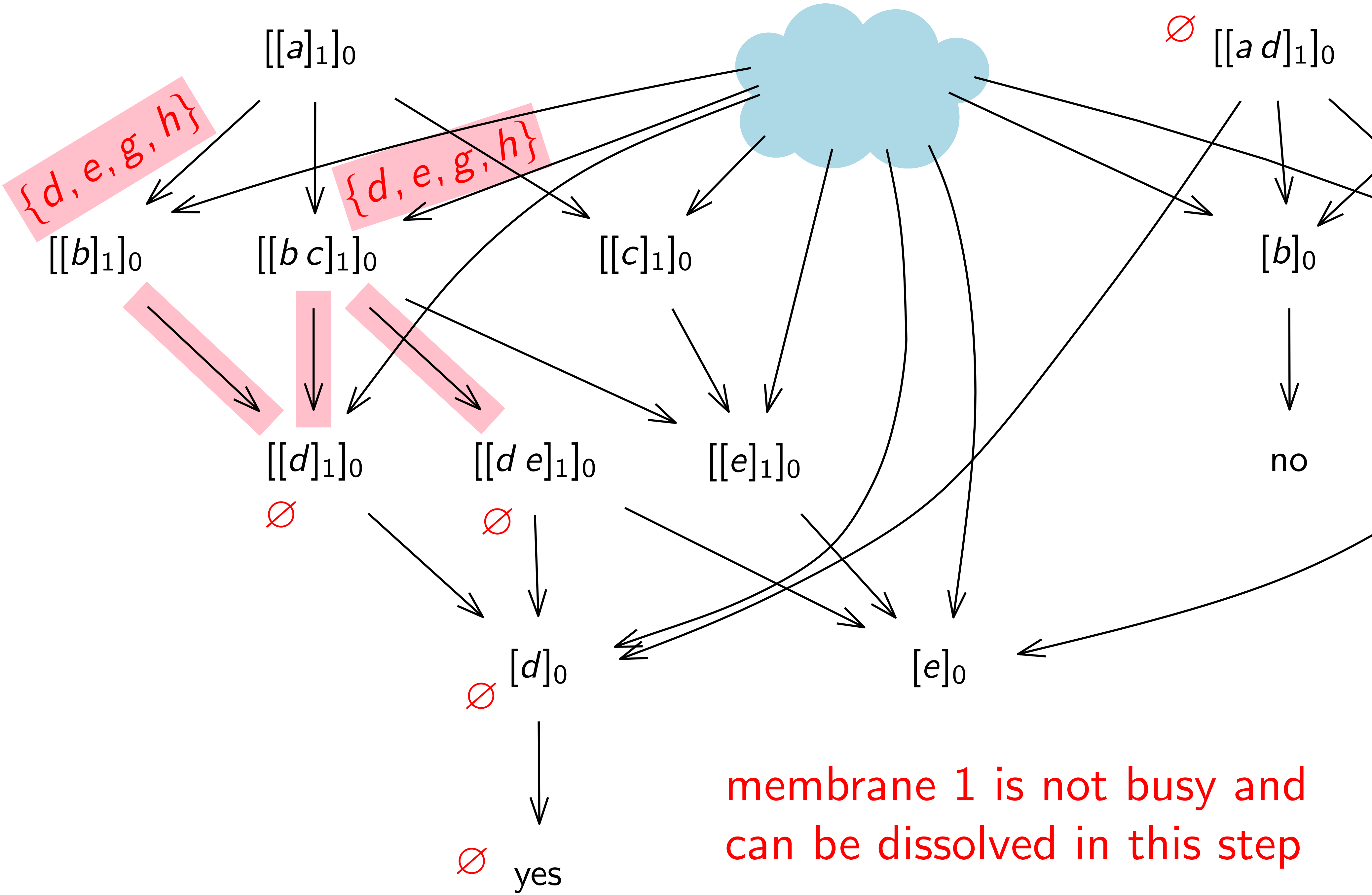


no troublemakers, the
P system has already halted

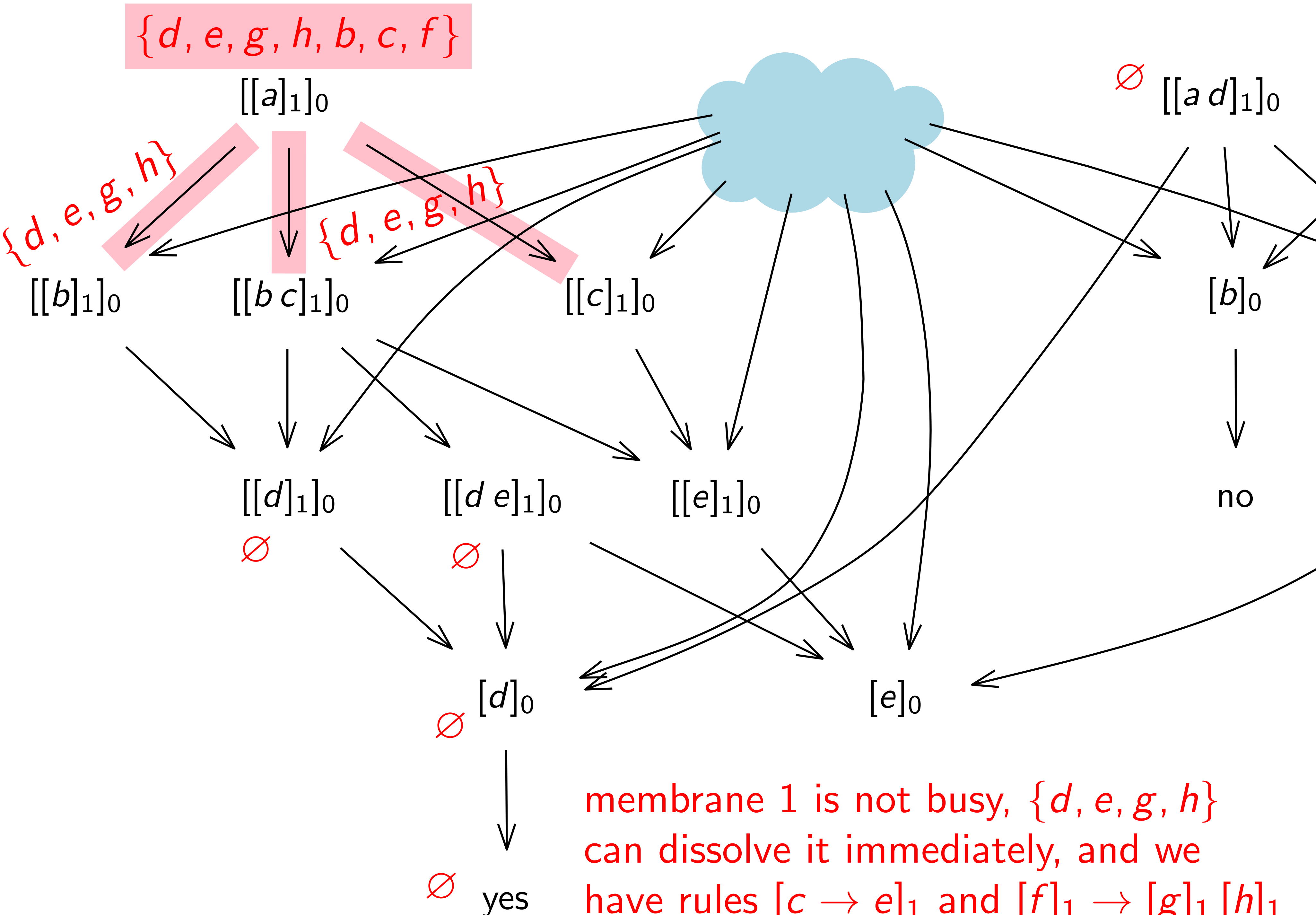


the outermost membrane
cannot be dissolved

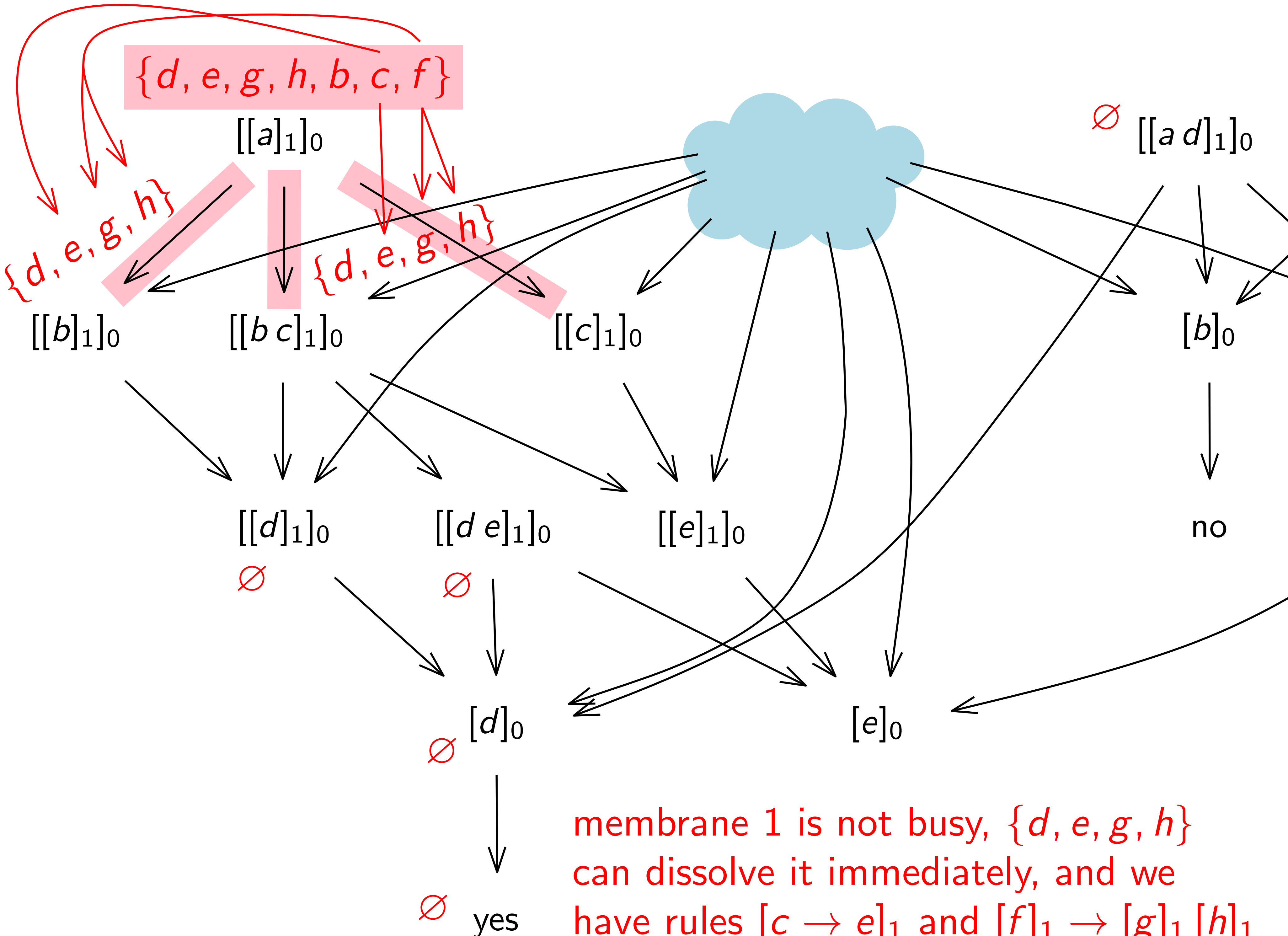

$[[a]_1]_0$
$[[a\ d]_1]_0$
$[[b]_1]_0$
$[[b\ c]_1]_0$
$[[c]_1]_0$
$[b]_0$
$[[d]_1]_0$
$[[d\ e]_1]_0$
$[[e]_1]_0$
no
$[d]_0$
$[e]_0$
yes
membrane 1 was busy



membrane 1 is not busy and can be dissolved in this step



membrane 1 is not busy, $\{d, e, g, h\}$ can dissolve it immediately, and we have rules $[c \to e]_1$ and $[f]_1 \to [g]_1\, [h]_1$



membrane 1 is not busy, $\{d, e, g, h\}$ can dissolve it immediately, and we have rules $[c \to e]_1$ and $[f]_1 \to [g]_1\ [h]_1$

A configuration $\mathcal{C}$ of the P system is untroubled if it contains a vertex connected to yes but none of its troublemakers

A configuration $\mathcal{C}$ of the P system is untroubled if it contains a vertex connected to yes but none of its troublemakers

If $\mathcal{C}$ is untroubled and $\mathcal{C} \rightarrow \mathcal{D}$, then $\mathcal{D}$ is untroubled

A configuration $\mathcal{C}$ of the P system is untroubled if it contains a vertex connected to yes but none of its troublemakers

If $\mathcal{C}$ is untroubled and $\mathcal{C} \to \mathcal{D}$, then $\mathcal{D}$ is untroubled

If $\mathcal{D}$ is untroubled and $\mathcal{C} \to \mathcal{D}$, then $\mathcal{C}$ is untroubled

A configuration $\mathcal{C}$ of the P system is untroubled if it contains a vertex connected to yes but none of its troublemakers

If $\mathcal{C}$ is untroubled and $\mathcal{C} \to \mathcal{D}$, then $\mathcal{D}$ is untroubled

If $\mathcal{D}$ is untroubled and $\mathcal{C} \to \mathcal{D}$, then $\mathcal{C}$ is untroubled

Theorem. A P system accepts iff its initial configuration is untroubled

The graph has $O(alphabet^2 \times labels)$ vertices and is constructed by iterating over the rules

The graph has $O(\mathit{alphabet}^2 \times \mathit{labels})$ vertices and is constructed by iterating over the rules

The troublemakers are computed by depth-first search of the (transposed) dependency graph

The graph has $O(\mathit{alphabet}^2 \times \mathit{labels})$ vertices and is constructed by iterating over the rules

The troublemakers are computed by depth-first search of the (transposed) dependency graph

Untroubledness of the initial configuration of the P system is checked by looking at the vertices it contains

The graph has $O(\mathit{alphabet}^2 \times \mathit{labels})$ vertices and is constructed by iterating over the rules

The troublemakers are computed by depth-first search of the (transposed) dependency graph

Untroubledness of the initial configuration of the P system is checked by looking at the vertices it contains

Theorem. We can check in polynomial time if the P system accepts

The monodirectional, shallow, deterministic P conjecture is true

The monodirectional, shallow, deterministic P conjecture is true

$$\mathbf{DPMC}_{\mathcal{D}}^{[\star]} = \mathbf{P} = \mathbf{DMC}_{\mathcal{D}}^{[\star]}$$

Open problems

Prove the result for confluent
(not just deterministic) P systems

Open problems

Prove the result for confluent
(not just deterministic) P systems

Prove the result for P systems with
deeper membrane structures

Open problems

Prove the result for confluent
(not just deterministic) P systems

Prove the result for P systems with
deeper membrane structures

Prove the result for bidirectional P systems
(no idea if we can do this)

Open problems

Prove the result for confluent
(not just deterministic) P systems

Prove the result for P systems with
deeper membrane structures

Prove the result for bidirectional P systems
(no idea if we can do this)

Use generalised dependency graphs for other
variants of P systems to prove **P** upper bound
or find "borderlines" for efficiency

Thanks for your attention!

Grazie per l'attenzione!

Any questions?