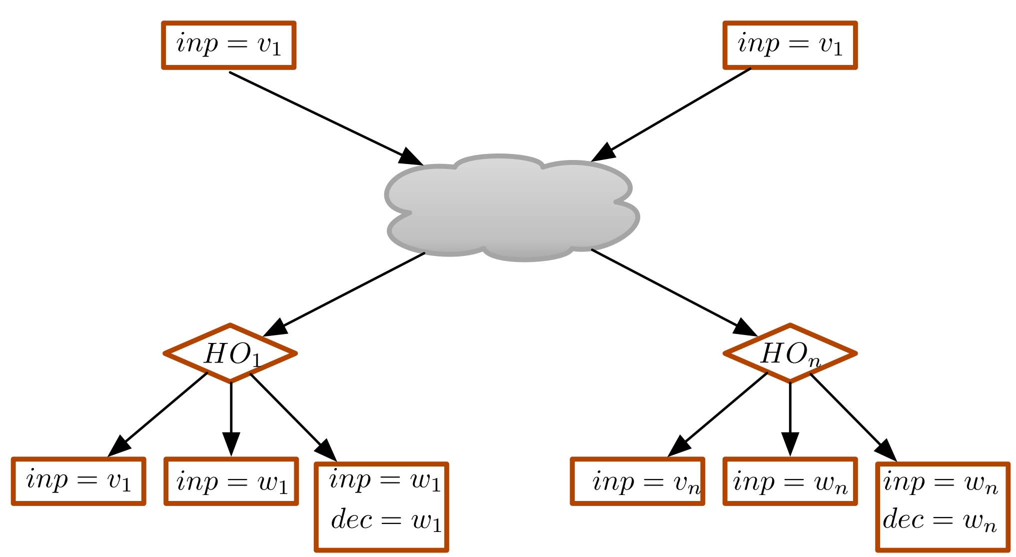
Towards verification of distributed algorithms in the Heard-of model

Igor Walukiewicz
CNRS Bordeaux
Joint work with Anca Muscholl and Balasubramanian A.R.



Fault tolerant distributed computing

cells

large computer networks

microprocessors

social interactions

Distributed computing:

understand principles and conceptual tools for design of distributed systems.

Approach:

Solving a problem in a given model, or showing impossibility, establishing lower bounds.

Fault tolerant distributed computing

cells

large computer networks

microprocessors

social interactions

Distributed computing:

understand principles and conceptual tools for design of distributed systems.

Models:

- shared memory vs. message passing
- snapshot sheared memory vs. read/write shared memory
- synchronous or asynchronous message passing

Fault tolerant distributed computing

cells

large computer networks

microprocessors

social interactions

Distributed computing:

understand principles and conceptual tools for design of distributed systems.

Results:

- impossibility of consensus in the asynchronous shared-memory model [Loui Abu-Amara ’87]
- Praxos [Lamport ’98]

Fault tolerant distributed computing

cells

large computer networks

microprocessors

social interactions

Distributed computing:

understand principles and conceptual tools for design of distributed systems.

Challenge:

(Too) big variety of models [Moses, Rajsbaum, 2002]

Fault tolerant distributed computing

cells

large computer networks

microprocessors

social interactions

Degrees of synchrony

Notion of a faulty component

Consensus problem has received the greatest amount of attention in this field

Consensus problem

At the beginning every process gets one value

The algorithm should ensure:

- Termination: every process decides on a value
- Agreement: no two processes decide on different values
- Stability: once a process decides, it cannot change his decision
- Non-triviality: Decided value can only be an initial value of one of the processes

What can verification bring to fault tolerant distributed computing

Understanding under which conditions an algorithm is correct.

Insights on limitations of a given model.

Why verification is difficult

Unboundedness in many dimensions:

- Number of processes
- Asynchrony
- Data values
- Identifiers
- Time-stamps

Heard-off model

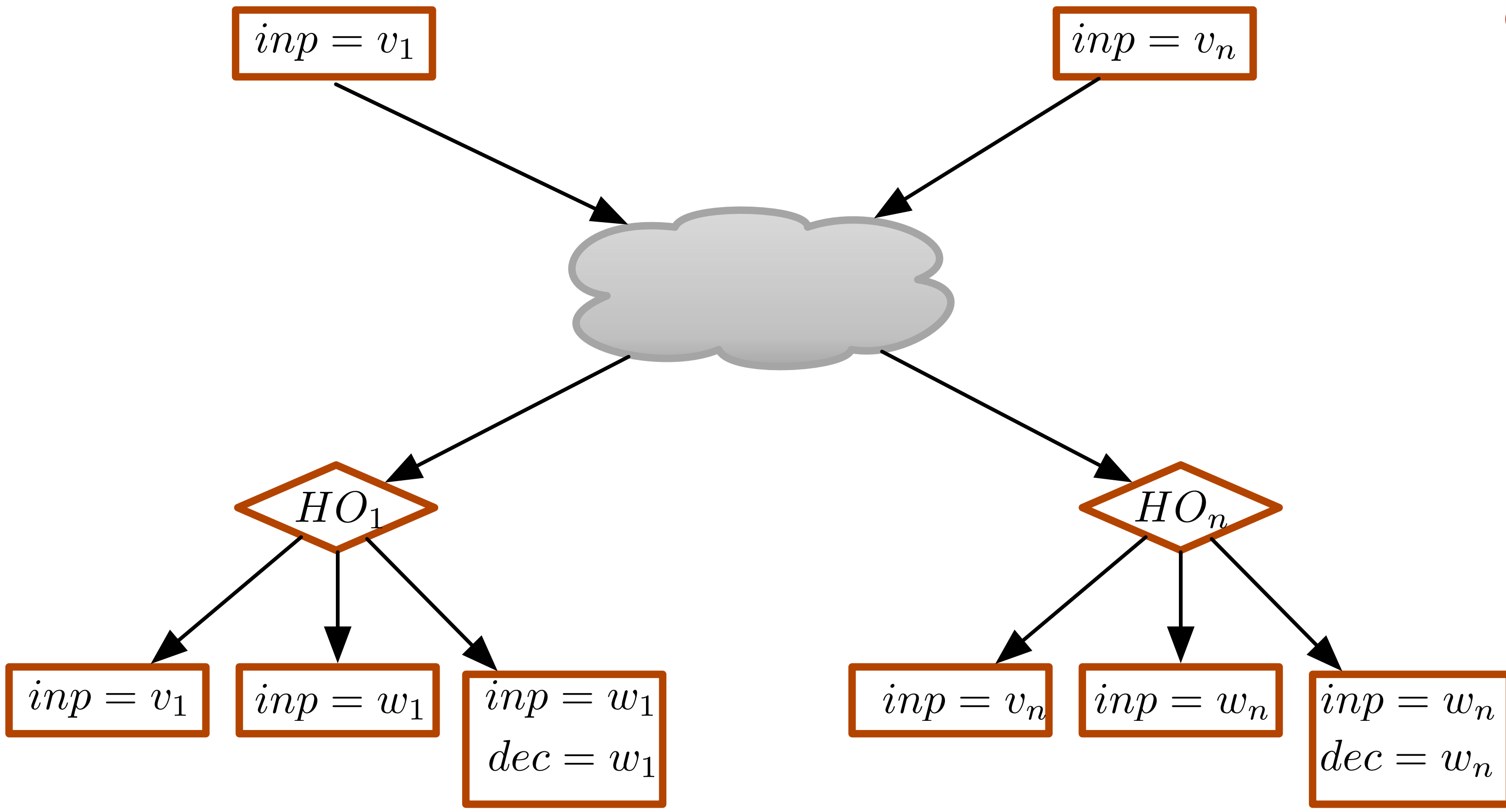
Introduced by Bernadette Charron-Bost · André Schiper in 2009

A round based model for non synchronous computing.

Unified treatment of different types of faults through transmission faults.

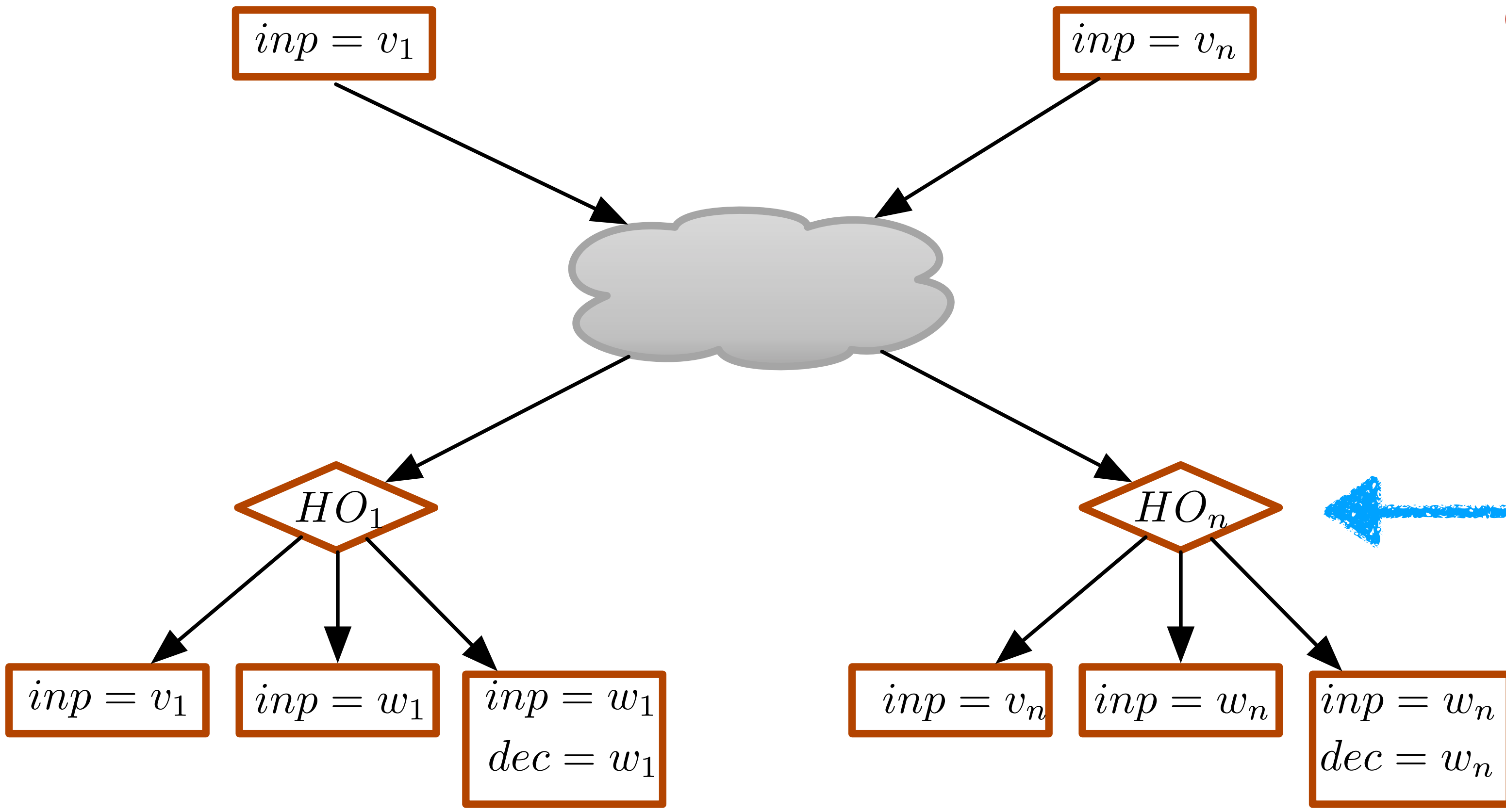
A model is relatively simple and concise:
a good candidate to develop verification methods

- [Charron-Bost, Stefan Merz,..] Efficient encoding the model in Isabelle, and TLA
- [Drăgoi, Henzinger, Zufferey,..] A semi-automatic proof method, a domain-specific language based on HO-model.
- [Ognjen Maric, Christoph Sprenger, David Basin, *Cut-off Bounds for Consensus Algorithms*], see later
- [R. Bloem, S. Jacobs, A. Khalimov, I. Konnov, S. Rubin, H. Veith, and J. Widder. *Decidability of Parameterized Verification*], a book, 2015



- No operations on variables
- No failure of components
- No process identities

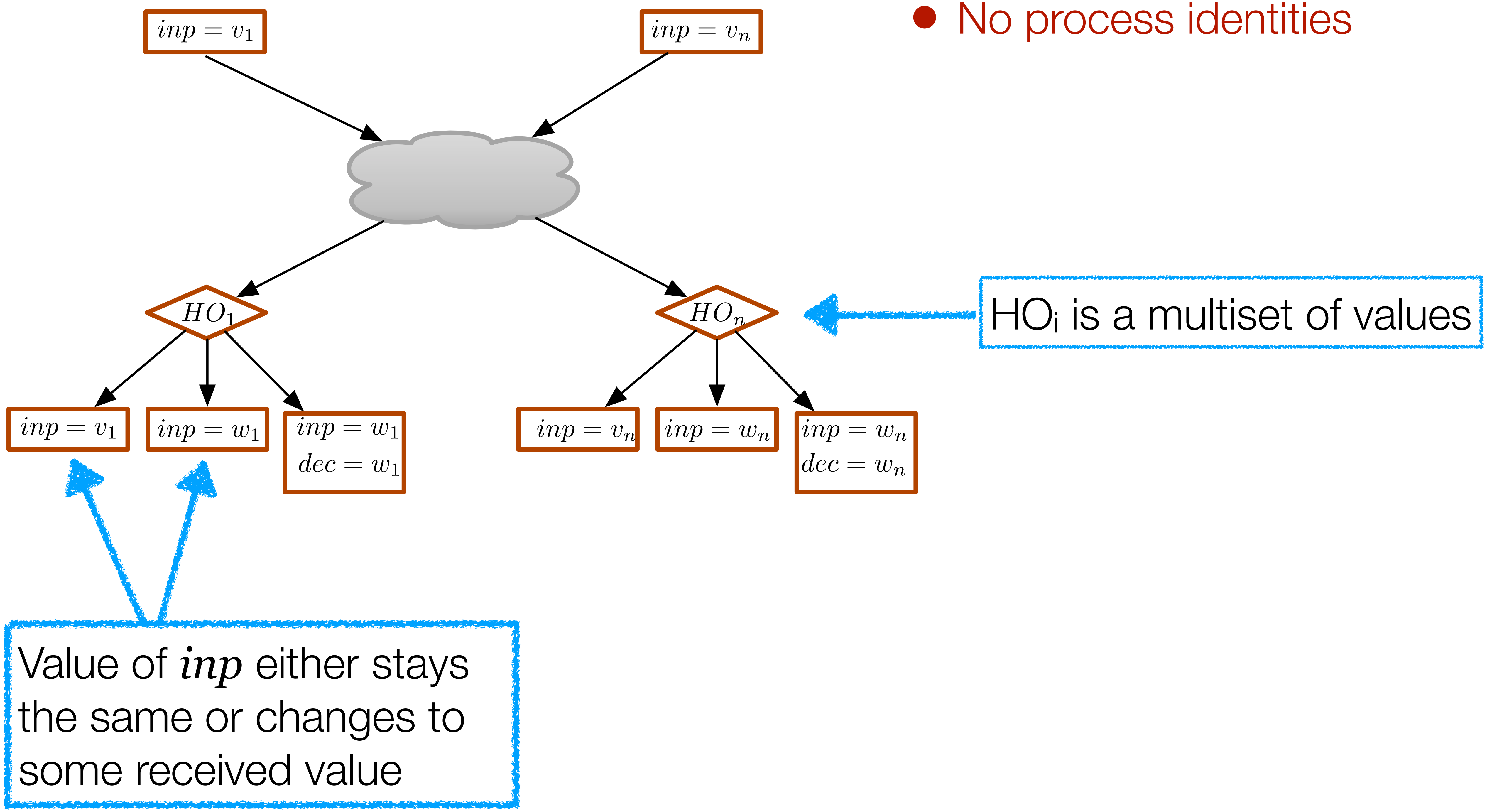
- No operations on variables
- No failure of components
- No process identities



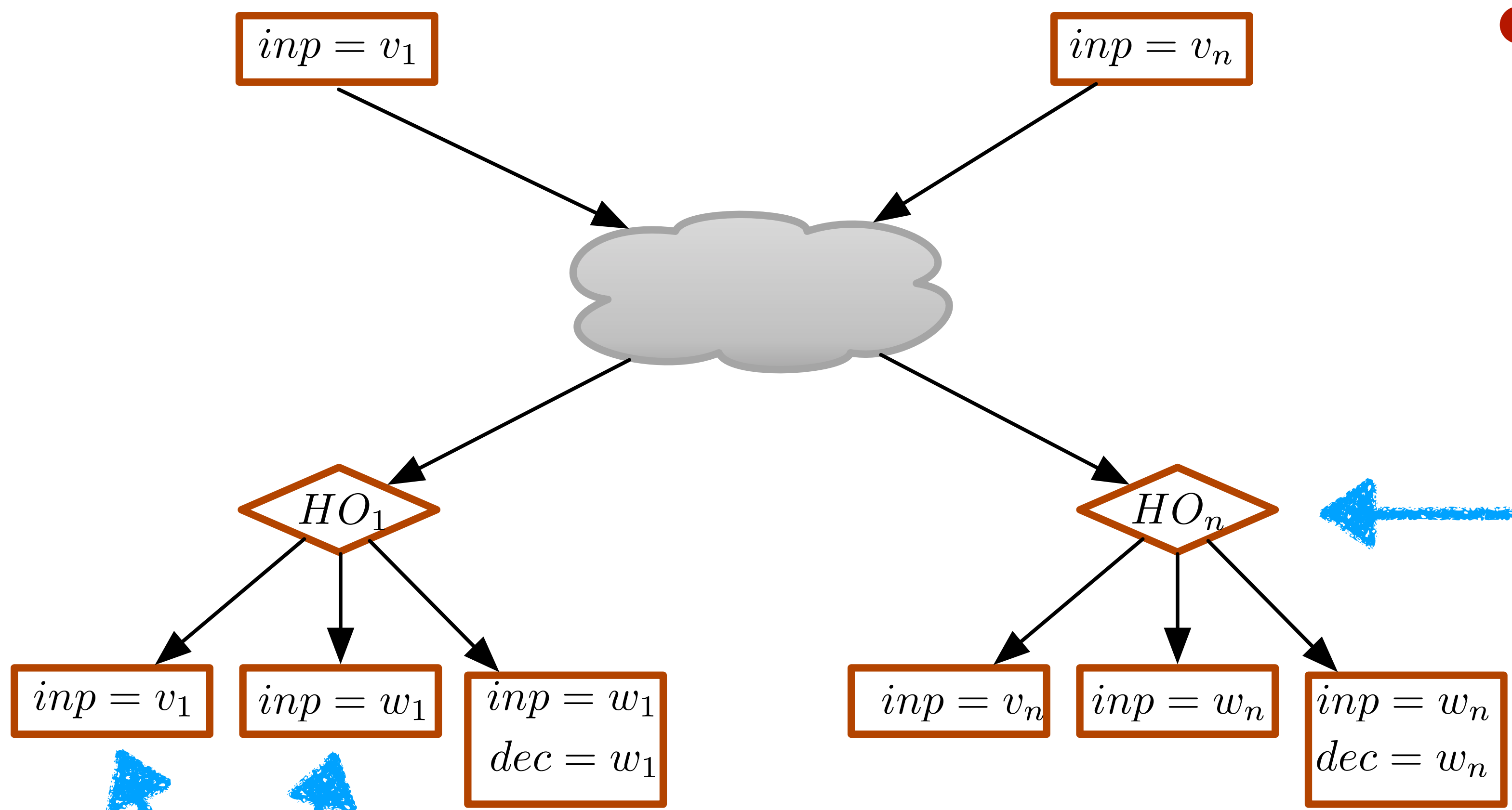
HO

HO_i is a multiset of values

- No operations on variables
- No failure of components
- No process identities



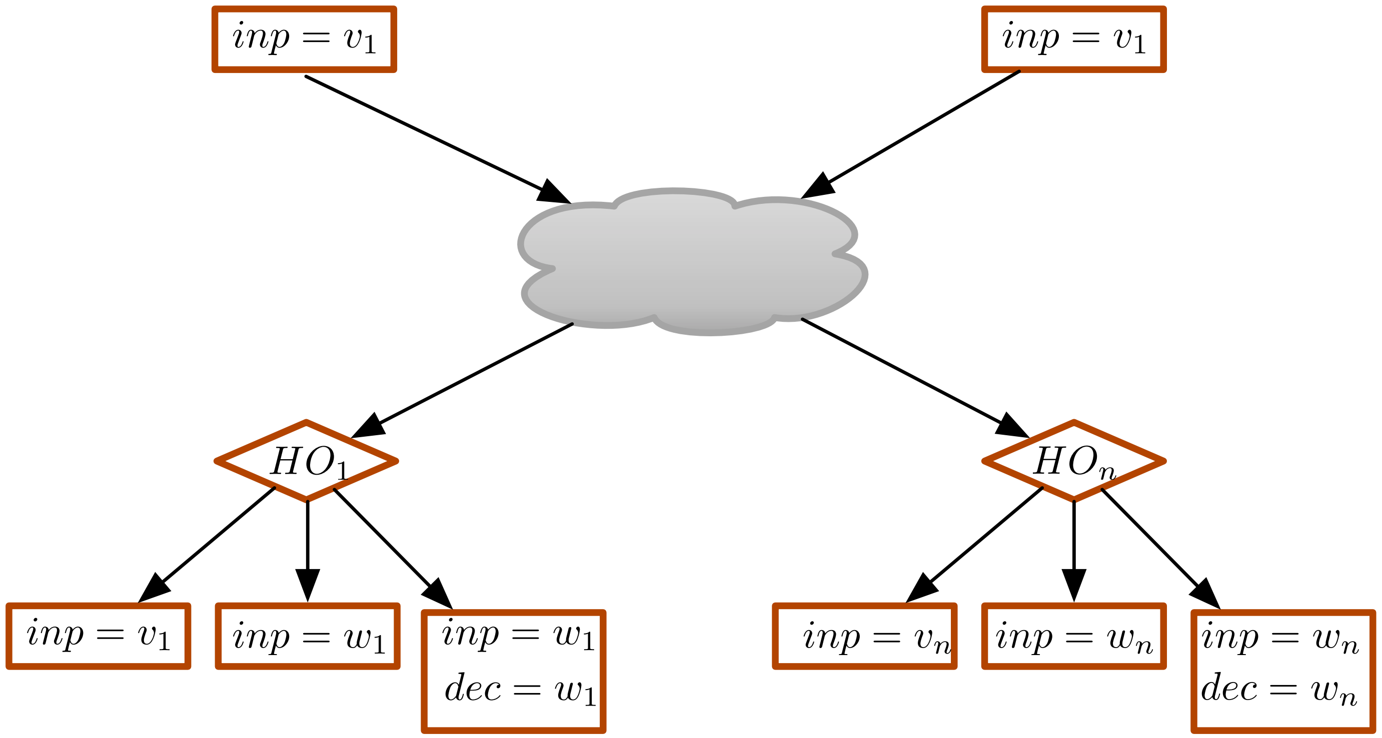
- No operations on variables
- No failure of components
- No process identities



HO$_i$ is a multiset of values

Value of ***inp*** either stays the same or changes to some received value

A process can also definitely decide on some value by setting ***dec***



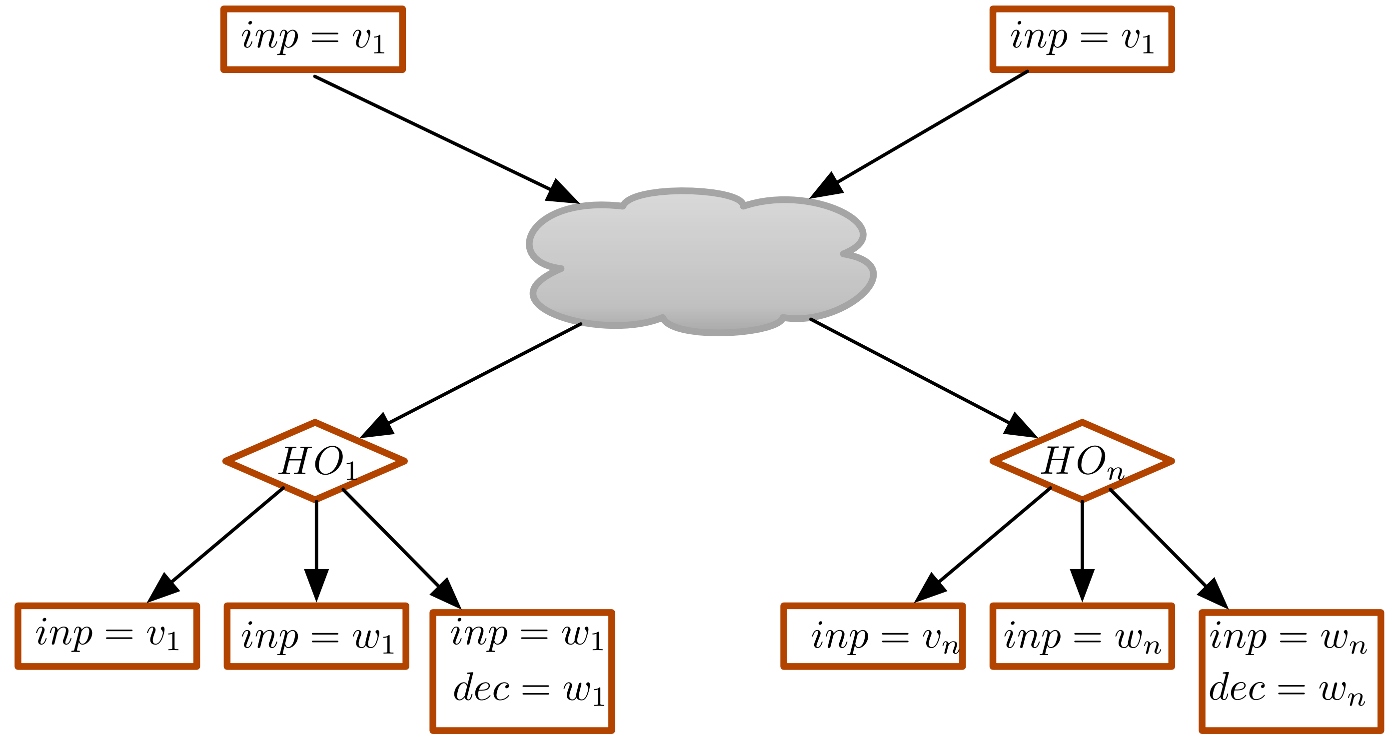
Program:

send(inp);

If |HO|>2/3 and (all=) then dec:="any received value"

If |HO|>2/3 then inp:="minimal value"

Does this program solve the consensus problem?



Program:

send(inp);

If |HO|>2/3 and (all=) then dec:="any received value"

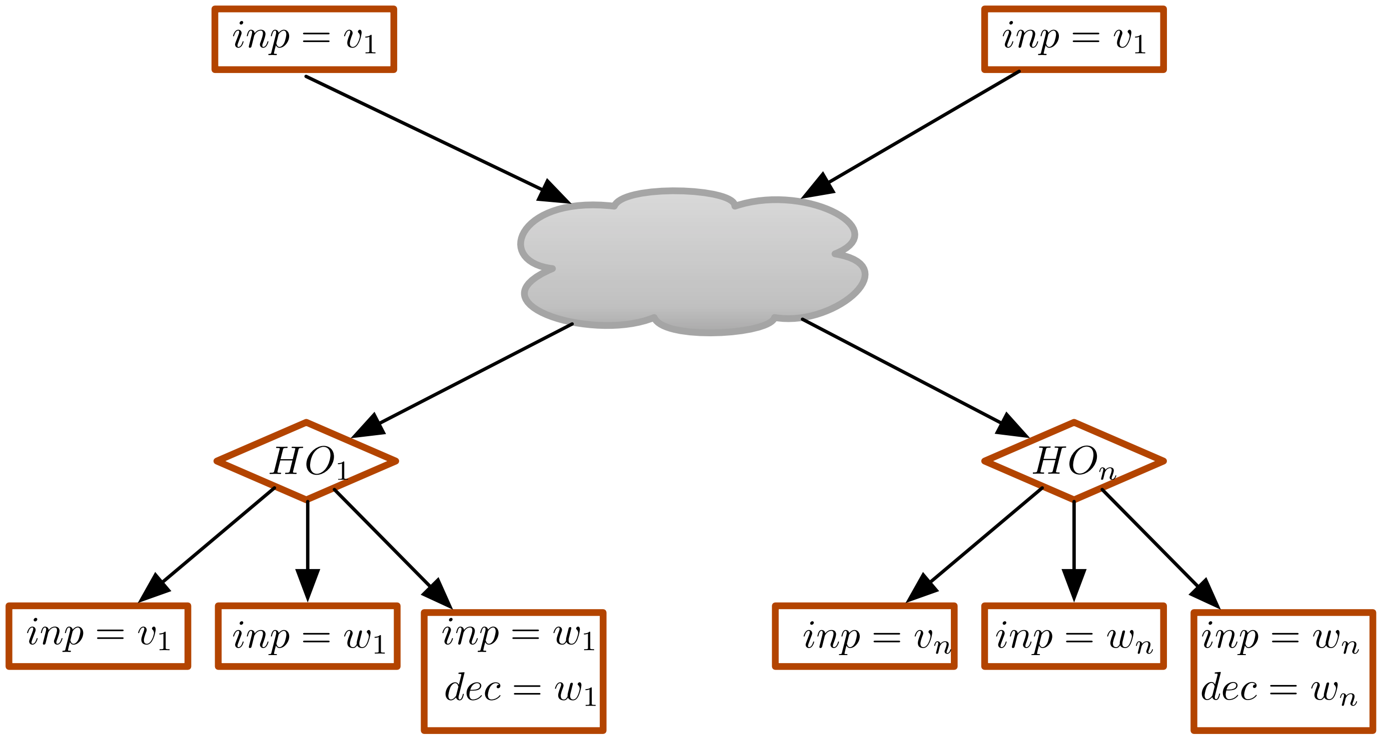
If |HO|>2/3 then inp:="minimal value"

Communication predicate:

exists round ($\theta_=$ and $\theta_{2/3}$) and later exists round $\theta_{2/3}$

$\theta_=$: says $HO_p = HO_q$ for all processes p,q

$\theta_{2/3}$: says $|HO_p|>2/3$ for all p



Program:

send(inp);

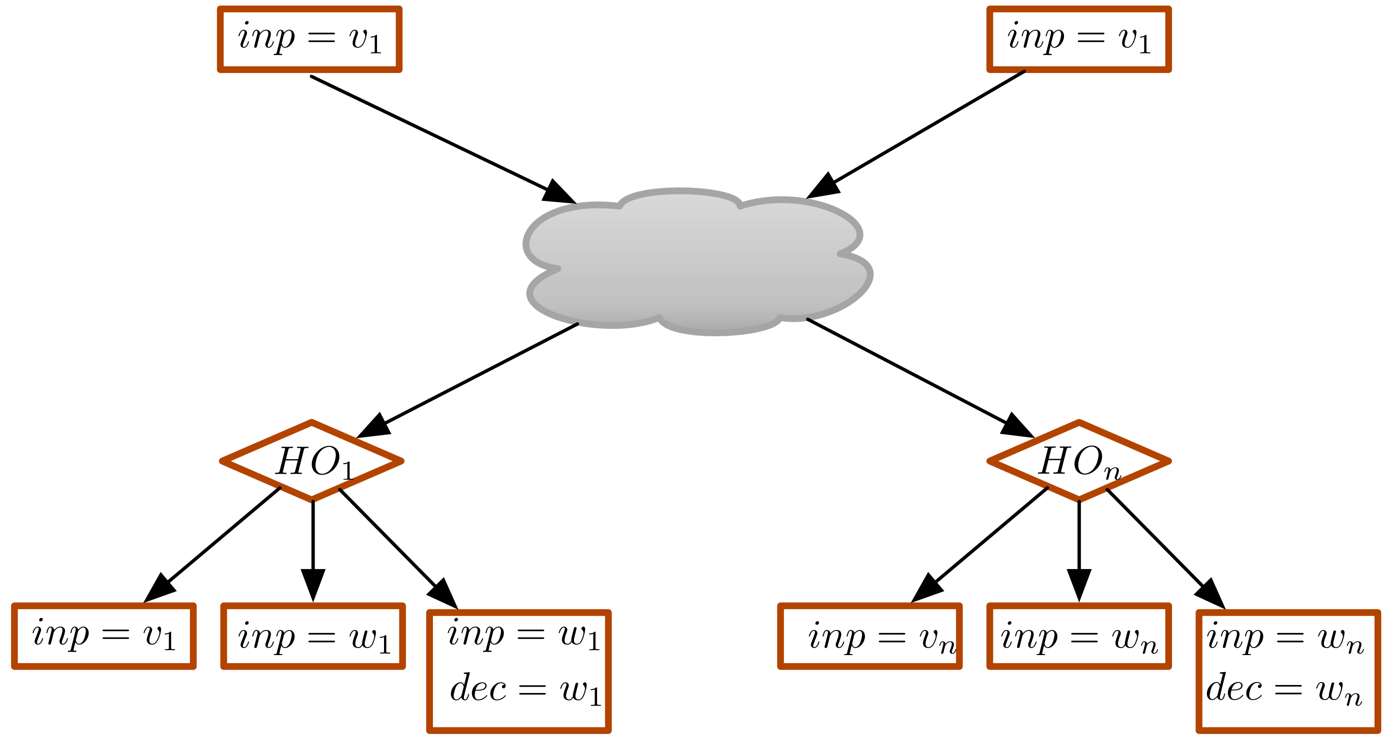
If |HO|>2/3 and (all=) then dec:="any received value"

If |HO|>2/3 then inp:="smallest most frequent value"

Communication predicate:

At some round ($\theta_=$ and $\theta_{2/3}$) and at a later round $\theta_{2/3}$

Q: What if we change to "smallest most frequent value"?



Program:

send(inp);

If |HO|>2/3 and (all=) then dec:="any received value"

If |HO|>2/3 then inp:="smallest most frequent value"

Communication predicate:

At some round ($\theta_{=}$ and $\theta_{2/3}$) and at a later round $\theta_{2/3}$

Q: What if we change the communication predicate?

Phase: a sequence of rounds

P: R_1
:
:
R_i

- Only ***inp*** and ***dec*** variables survive between phases
- ***dec*** can be set only once, and it is not sent

Every rule is a send followed by a sequence of conditional assignments

send(x);
If (some property of HO) then inp:=v

HO is a multiset of values and the property talks about frequencies of values

One of the received values

Algorithm: P_1 ; P^* ; P_2 ; $P\omega$

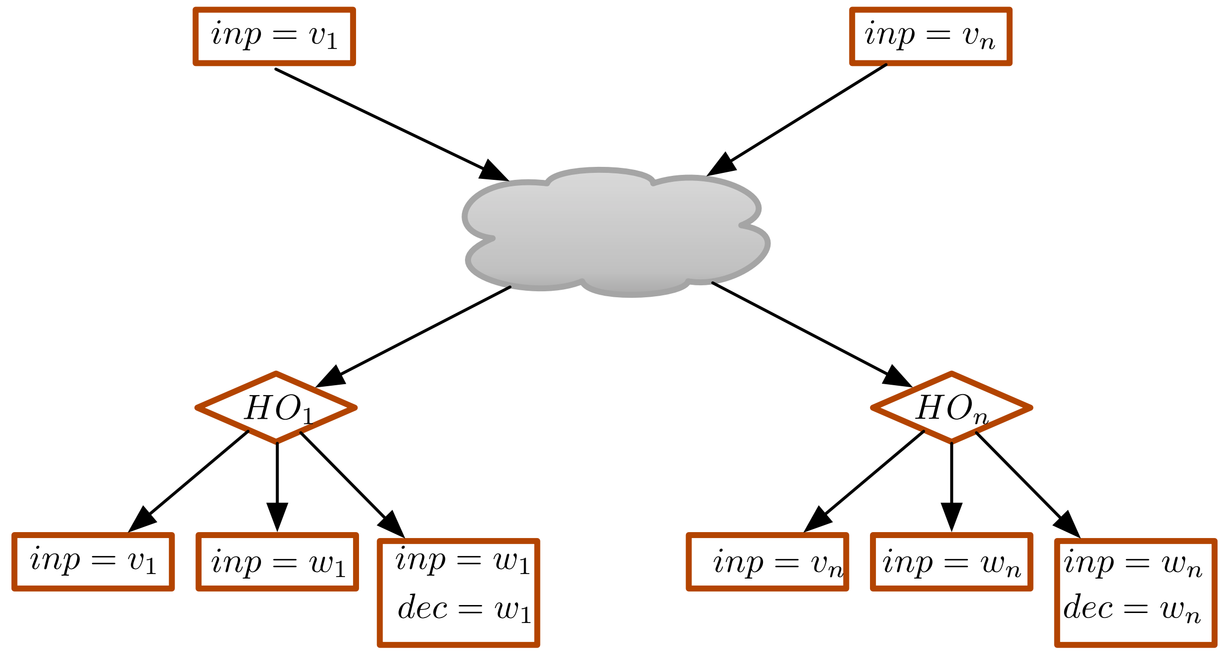
Phase:

P: R_1

:

:

R_i



Communication predicates

$$\theta_0^* \ (\theta_{2/3} \wedge \theta_=) \ \theta_0^* \ \theta_{2/3} \ \theta^\omega$$

$\theta_=$: says $HO_p = HO_q$ for all processes p,q

$\theta_{2/3}$: says $|HO_p|>2/3$ for all p

What do we want

1. Given an algorithm over a fixed set of values, decide if it solves consensus.
    - What tests are allowed?
    - What communication predicates are allowed?

2. Do we have cut-off principle: is it enough to consider some bounded number of processes?

3. Do we have 0/1 principle: is it enough to consider 2 values?

1. Given an algorithm over a fixed set of values, is it decidable to establish if the algorithm solves consensus?

2. Do we have cut-off principle: is it enough to consider some bounded number of processes?

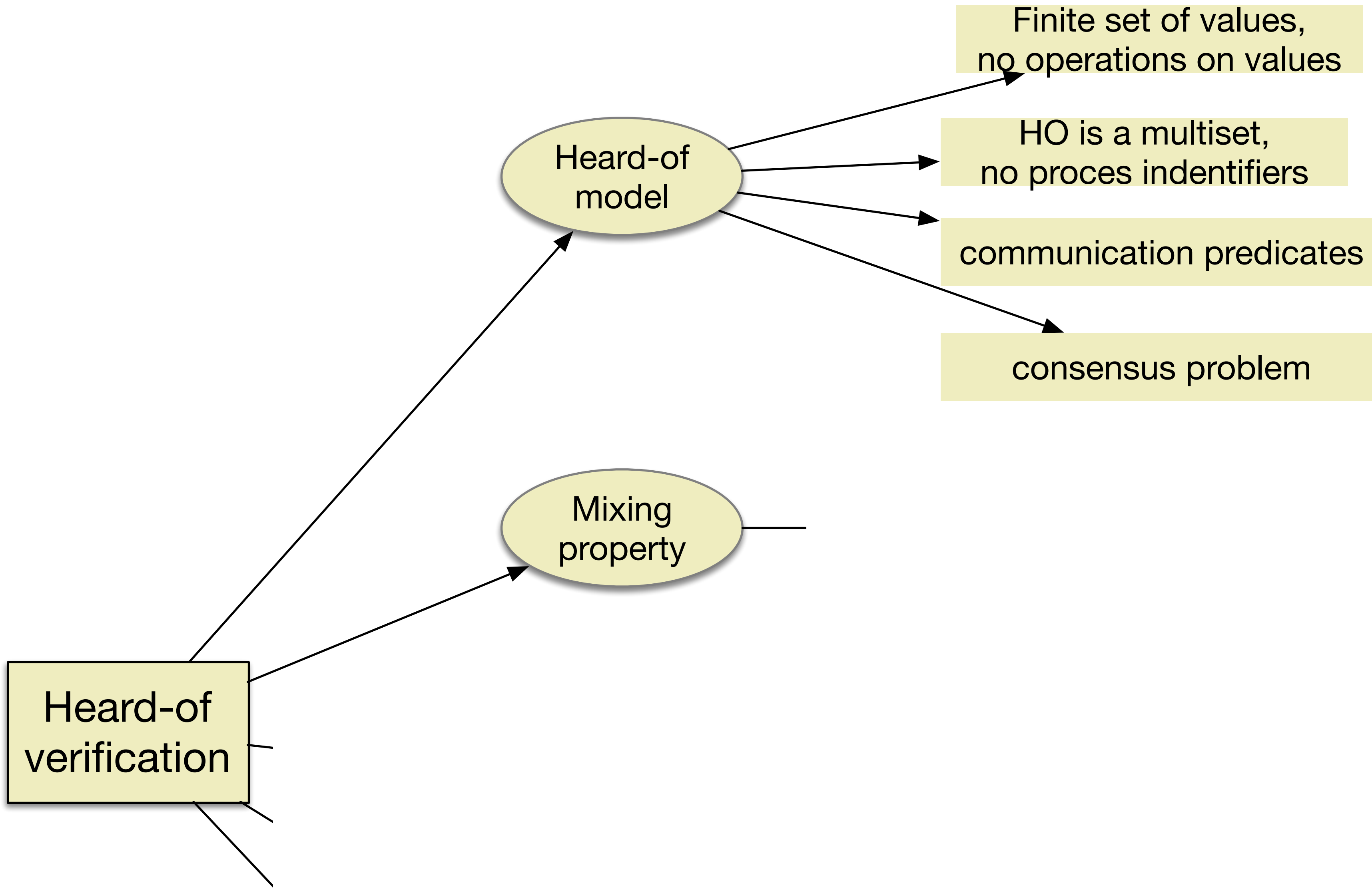
3. Do we have 0/1 principle: is it enough to consider 2 values?

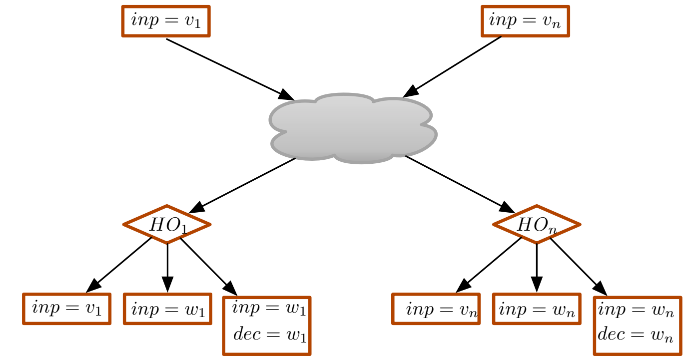
Results [Ognjen Maric, Christoph Sprenger, David Basin, CAV'17]:

- Properties 2) and 3) hold under some conditions.
- Property 3) does not always hold

Here:

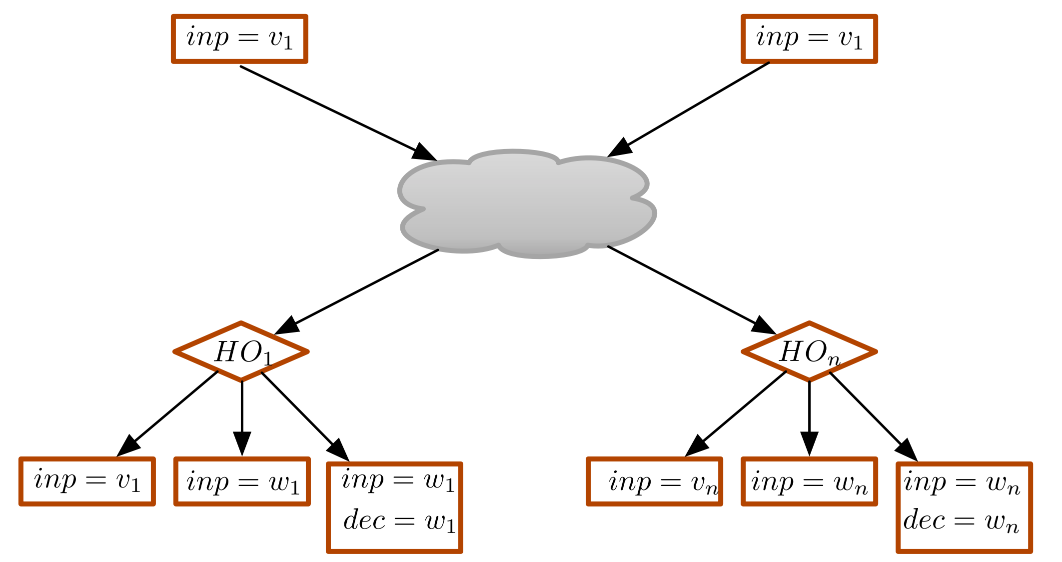
- For 2 values the problem is decidable in a quite a general case.
- For many values, and quite general tests, the problem is undecidable.
- Some cases when the problem is decidable.


Heard-of verification
Heard-of model
Mixing property
Finite set of values,
no operations on values
HO is a multiset,
no proces indentifiers
communication predicates
consensus problem


$inp = v_1$
$inp = v_n$
HO_1
HO_n
$inp = v_1$
$inp = w_1$
$inp = w_1$
$dec = w_1$
$inp = v_n$
$inp = w_n$
$inp = w_n$
$dec = w_n$

Some observations

- What can be written depends only on frequencies of values
- Processes cannot test their state
  - Only *inp* and *dec* variables survive between phases
  - *dec* can be set only once, and it is not sent



Mixing property

Let $\texttt{write}(C, P)$ be the set of sets of values that can be written after phase P started in C. Ex $\{\{a, b\}, \{a, \bot\}\}$

Take $S \in \texttt{write}(C, P)$.

If $\bot \notin S$ then $C \to (v'_1, \ldots, v'_n)$ for $v'_i \in S$.

If $\bot \in S$ then $C \to (v'_1, \ldots, v'_n)$ where either $v'_i = v_i$ or $v'_i \in S$.

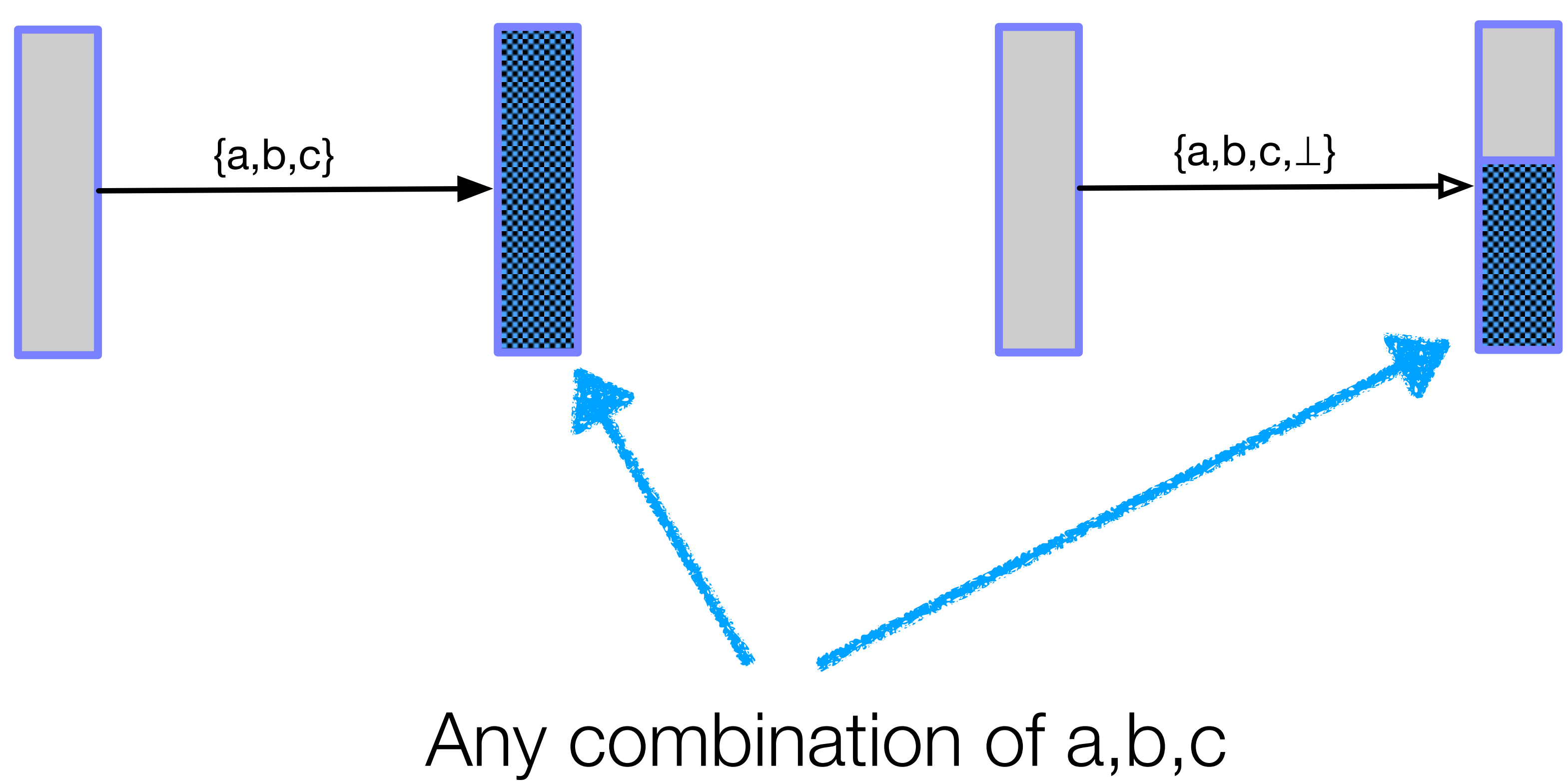
S determines possible next configurations

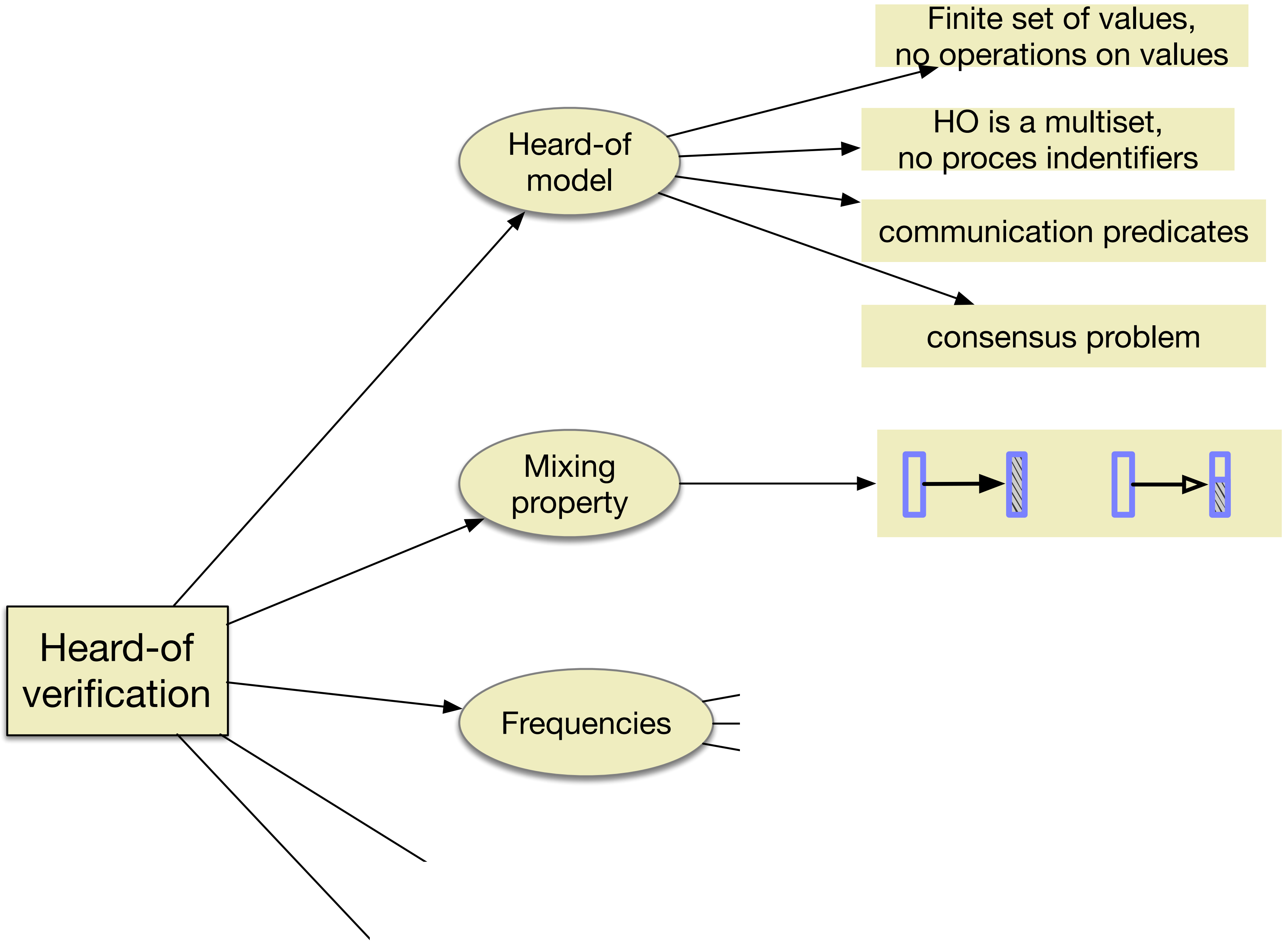
Mixing property

Let $\texttt{write}(C, P)$ be the set of sets of values that can be written after phase P started in C. Ex $\{\{a, b\}, \{a, \perp\}\}$

Take $S \in \texttt{write}(C, P)$.
If $\perp \notin S$ then $C \to (v'_1, \ldots, v'_n)$ for $v'_i \in S$.
If $\perp \in S$ then $C \to (v'_1, \ldots, v'_n)$ where either $v'_i = v_i$ or $v'_i \in S$.

S determines possible next configurations





Heard-of verification
Heard-of model
Finite set of values, no operations on values
HO is a multiset, no proces indentifiers
communication predicates
consensus problem
Mixing property
Frequencies

Frequencies

A configuration $(v_1, \ldots, v_n)$ determines a frequency $f : D \to [0,1]$

An algorithm determines a transition system:

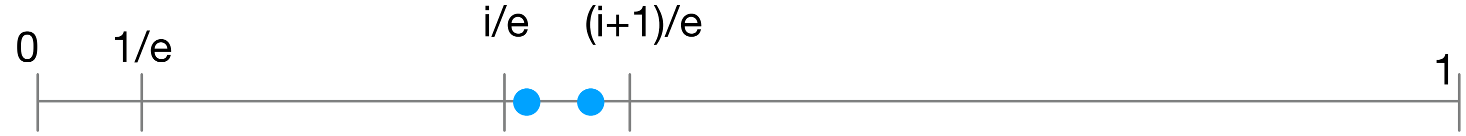
$$(i, f) \xrightarrow{S} ((i+1) \mod k, f')$$

Q: can we have a finite bisimulation quotient of this TS?

Is there a finite bisimulation quotient?

A configuration $(v_1, \ldots, v_n)$ determines a frequency $f : D \to [0, 1]$

Fix $e \in \mathbb{N}$. Let $r \sim_e r'$ when
$r \in [i/e, (i+1)/e]$ iff $r' \in [i/e, (i+1)/e]$, and $r = i/d$ iff $r' = i/d$.



For two frequencies we put $f \sim_e f'$ if $f(d) \sim_e f(d')$ for all $d \in D$.

We put $f \approx_e f'$ if for all $S \subseteq D$, $\sum_{d \in S} f(d) \sim_e \sum_{d \in S} f'(d)$

Fact: For D of size 3, the relations $\sim_e$ and $\approx_e$ are the same and are bisimulations.

For D of bigger sizes, both relations are not bisimulations

Tame algorithms

A configuration C defines a frequency $f_C : D \to [0,1]$.

Tame algorithm: For every phase P and every $S \subseteq D \cup \{\bot\}$ we have an existentially quantified set of linear constraints $L(P,S)$ s.t. for every configuration C:

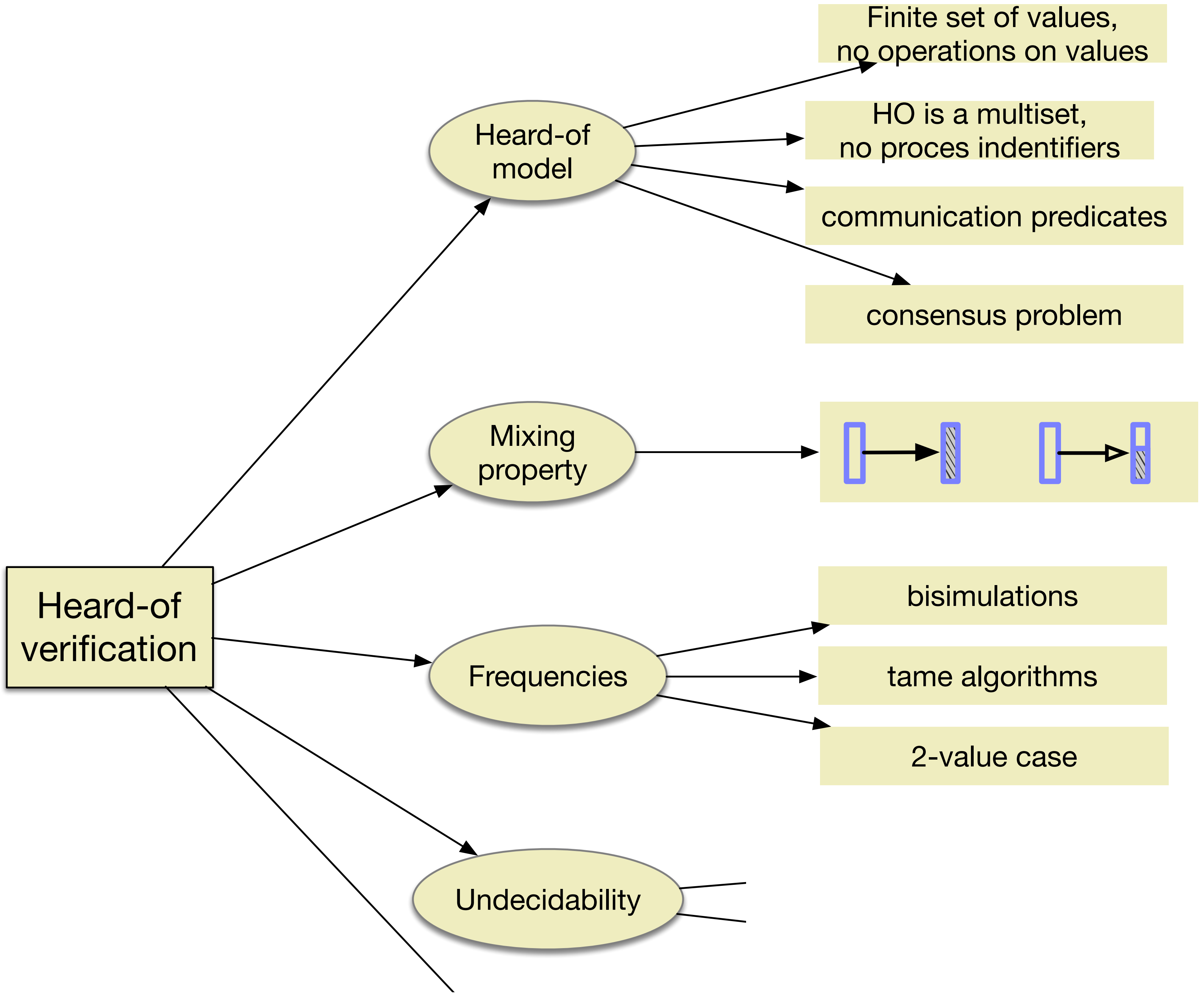
$$S \in \mathtt{write}(C,P) \qquad \text{iff} \qquad f_C \vDash L(P,S)$$

Example

If $(HO > 2/3)$ then inp:=smor $\qquad S = \{b, \bot\}$

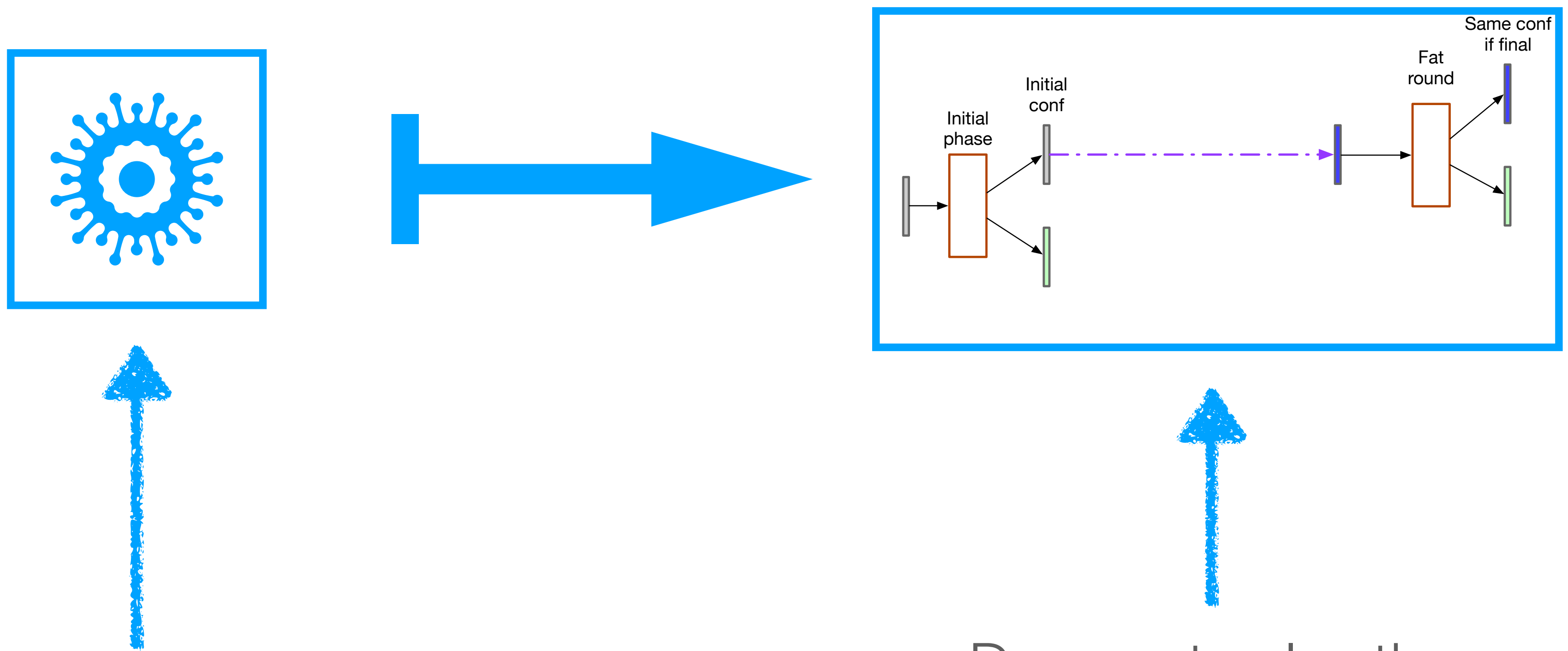
$$\begin{aligned} \exists x'_a, x'_c, x'_d. \quad & x'_a \leq x_a \wedge x'_c \leq x_c \wedge x'_d \leq x_d \\ & x_b > x'_a \wedge x_b \geq x'_c \wedge x_b \geq x'_d \\ & x'_a + x_b + x'_c + x'_d > 2/3 \end{aligned}$$

Thm: Every tame HO algorithm over 2 values has a cut-off.


Heard-of verification
Heard-of model
Finite set of values, no operations on values
HO is a multiset, no proces indentifiers
communication predicates
consensus problem
Mixing property
Frequencies
bisimulations
tame algorithms
2-value case
Undecidability

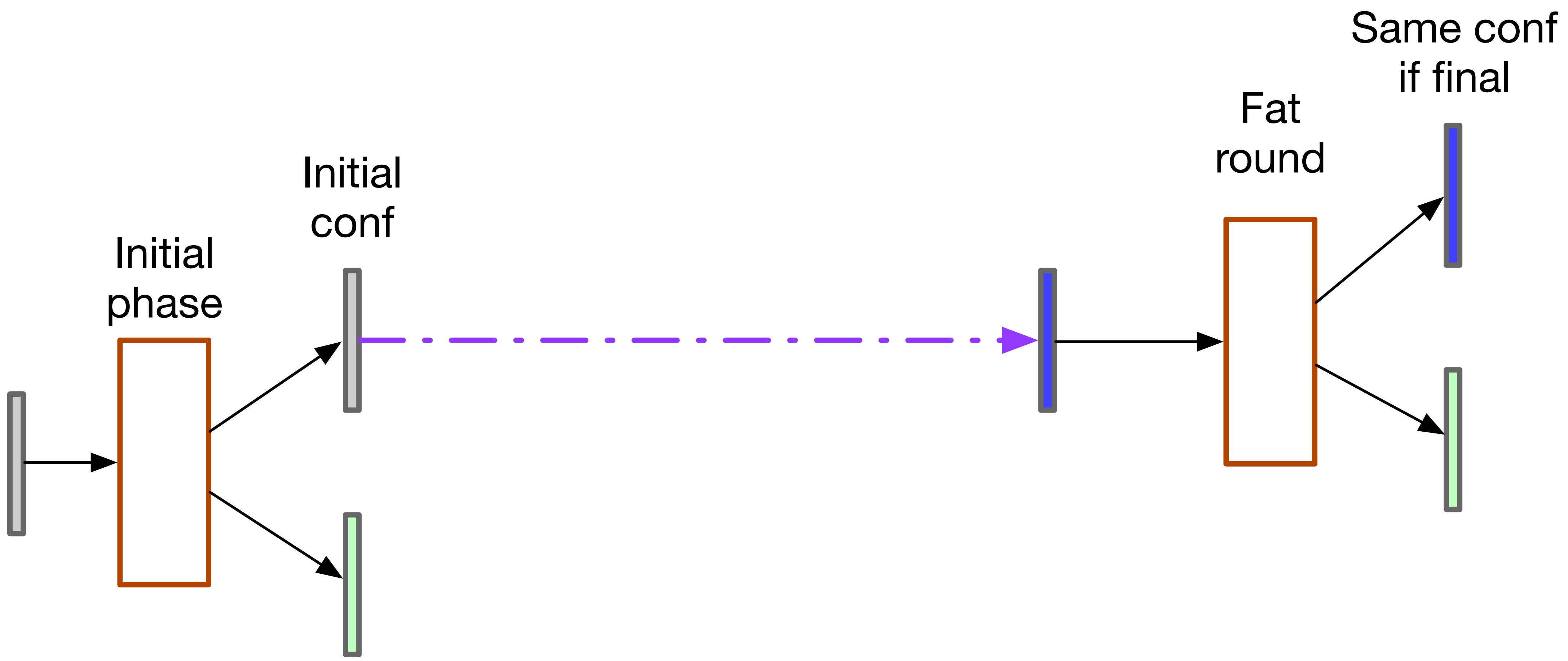
2-counter machine

HO-algorithm



Reaches the final state iff Does not solve the consensus problem

Thm: It is not decidable if a given HO algorithm solves consensus.



No consensus iff there exists a computation from initial to final.

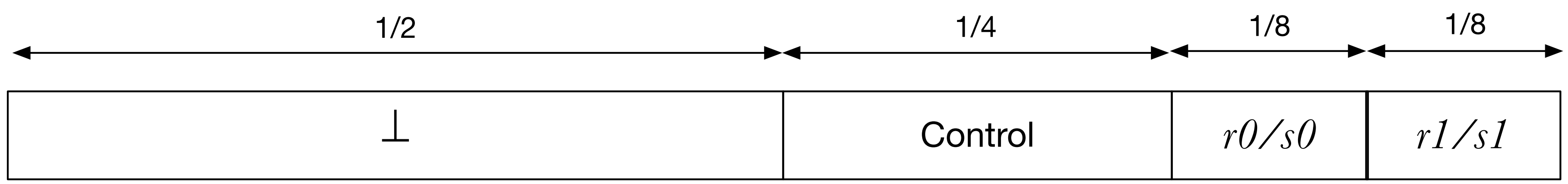
Fix a 1-counter machine

Values

$$(q,b),\quad (q,b,>0,\mathtt{dec}),\quad (q,b,\mathtt{dec}),$$

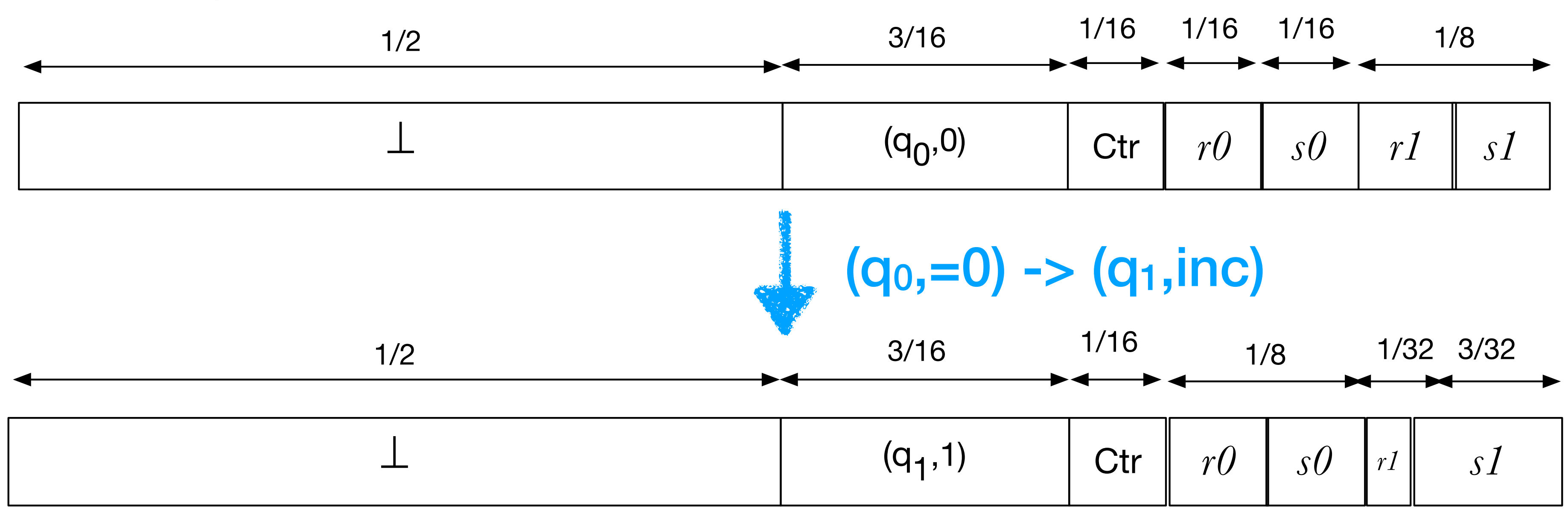
$$r^b, s^b \quad \text{for } b=0,1 \qquad \text{and} \qquad \bot$$

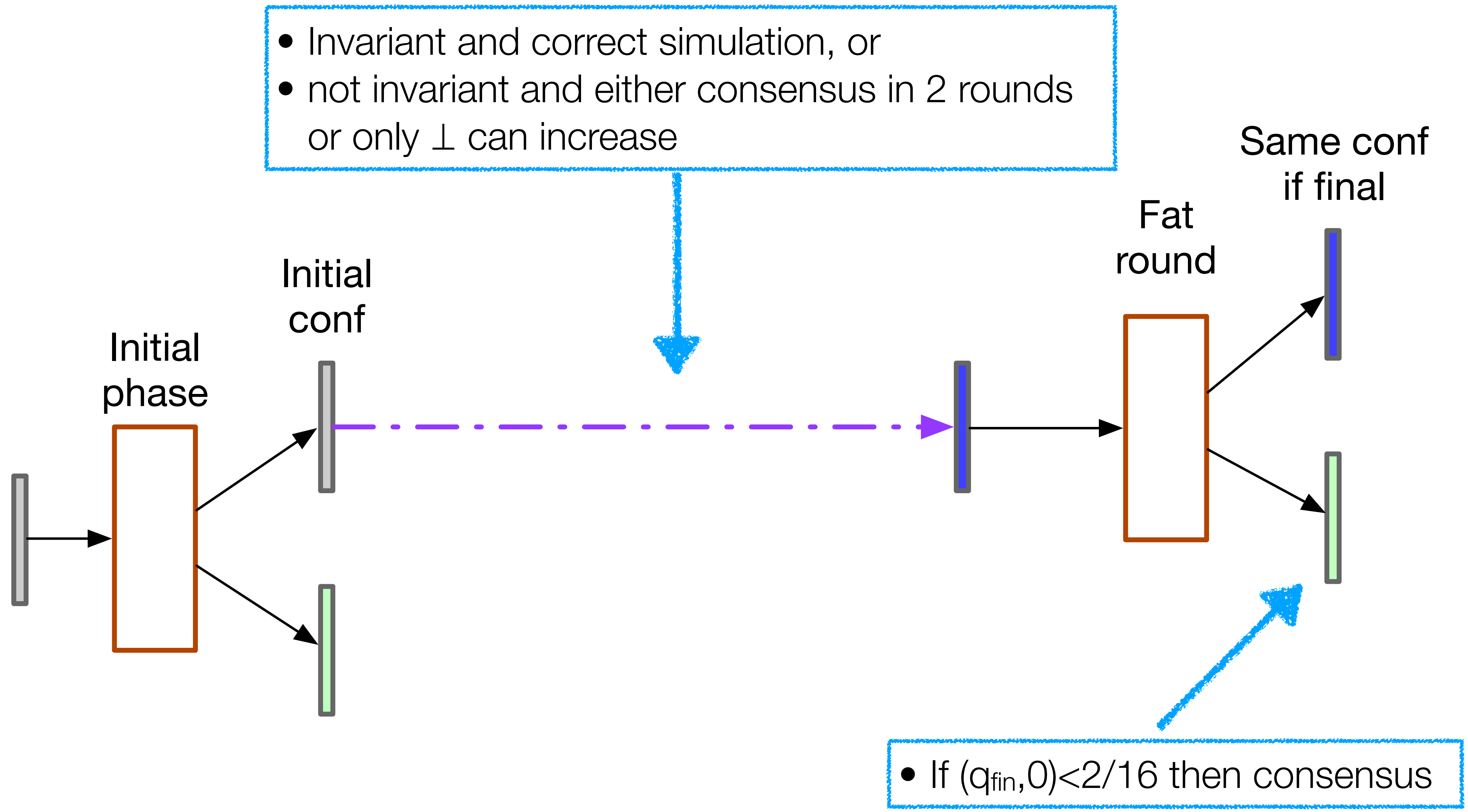
Invariant



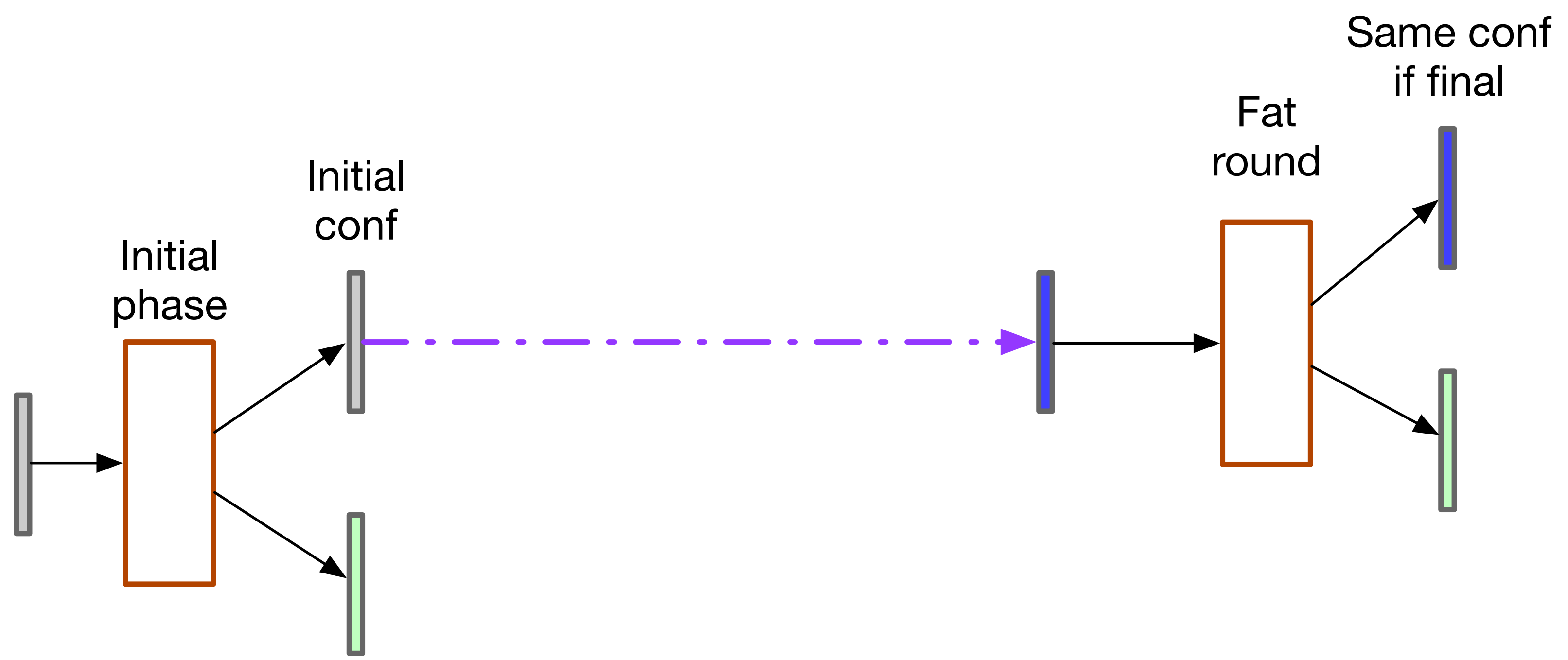
$$\text{counter} = k \text{ if } |r^b| = 1/2^{4+k}$$

Computation step



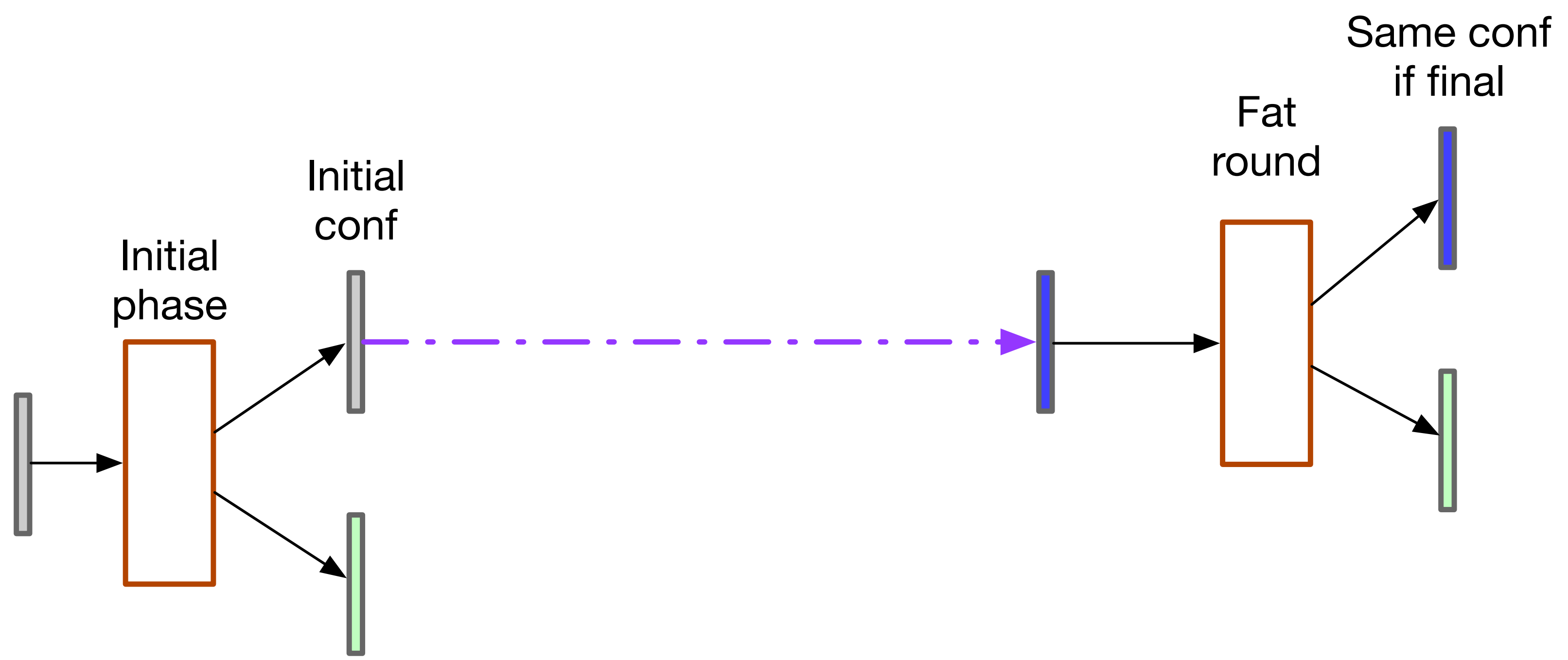


Communication predicate $(\theta_{1/2} \wedge \theta_{=})(\theta_{1/2} \wedge \theta_{=})\theta^{*}_{1/2}(\theta_{15/16})\theta^{\omega}_{1/2}$



Communication predicate $(\theta_{1/2} \wedge \theta_{=})(\theta_{1/2} \wedge \theta_{=})\theta^{*}_{1/2}(\theta_{15/16})\theta^{\omega}_{1/2}$

Thm: It is not decidable if a given HO algorithm solves consensus.



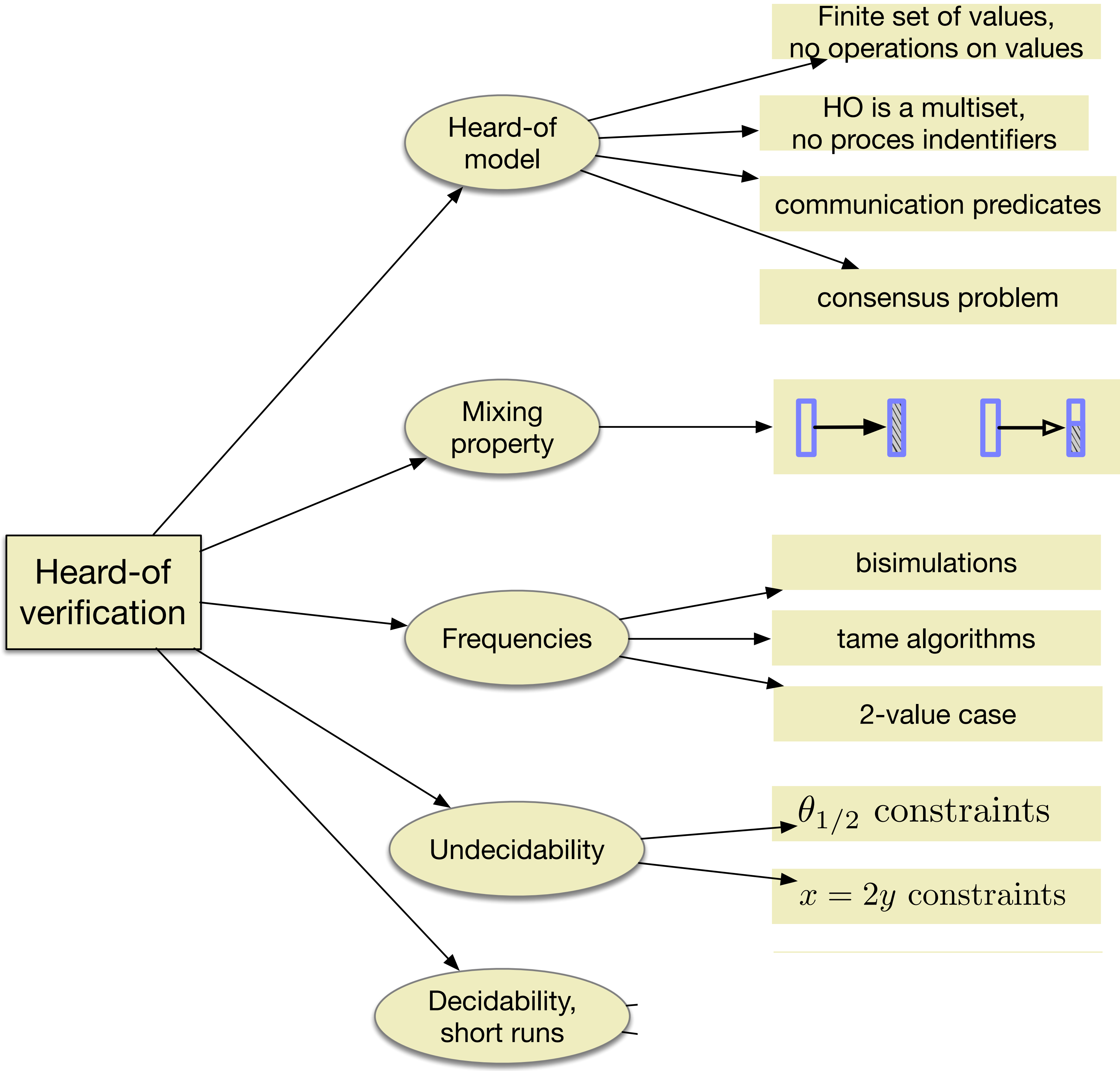
Communication predicate $(\theta_{1/2} \wedge \theta_{=})(\theta_{1/2} \wedge \theta_{=})\theta^{*}_{1/2}(\theta_{15/16})\theta^{\omega}_{1/2}$

Thm: It is not decidable if a given HO algorithm solves consensus.

Two questionable points:

We need $\theta_{1/2}$ saying that $|HO| \geq 1/2$ (non-strict inequality)

We need tests $x_a = 2x_b$



Heard-of verification
Heard-of model
Finite set of values, no operations on values
HO is a multiset, no proces indentifiers
communication predicates
consensus problem
Mixing property
Frequencies
bisimulations
tame algorithms
2-value case
Undecidability
$\theta_{1/2}$ constraints
$x = 2y$ constraints
Decidability, short runs

Decidability via short runs

An algorithm has *short run property* if there is a bound b s.t.:

for every run $C \longrightarrow^* C'$ there is a run $C \xrightarrow{\leq b} C'$

(both runs satisfy the communication predicate)

Suppose that the algorithm consists of one phase: it is P^*

Sporadic communication predicate: $\exists_{r_1 \leq \dots \leq r_k} \bigwedge \theta_i(r_i) \wedge \forall_{r \neq r_1, \dots, r_k} \theta(r)$

Full transition: $C_1 \xrightarrow{\bullet} C_2$ if $\texttt{val}(C_2) \subseteq \texttt{write}(C_1)$.

Shortening rule 1: $C_1 \xrightarrow{\bullet} C_2 \longrightarrow^* C_3 \xrightarrow{\bullet} C_4$ to $C_1 \xrightarrow{\bullet} C_4$

Obs: If $C_1 \longrightarrow C_2$ then $\texttt{val}(C_1) \supseteq \texttt{val}(C_2)$.

Shortening rule 2: $C_1 \longrightarrow C_2 \longrightarrow C_3$ to $C_1 \longrightarrow C_3$

Stability property: if $C_1 \longrightarrow C_2$ then $\texttt{write}(C_1) \supseteq \texttt{write}(C_2)$.

An algorithm has *short run property* if there is a bound b s.t.:

for every run $C \longrightarrow^* C'$ there is a run $C \xrightarrow{\leq b} C'$

(both runs satisfy the communication predicate)

Sporadic communication predicate: $\exists_{r_1 \leq \cdots \leq r_k} \bigwedge \theta_i(r_i) \wedge \forall_{r \neq r_1, \ldots, r_k} \theta(r)$

Shortening rule 1: $C_1 \xrightarrow{\bullet} C_2 \longrightarrow^* C_3 \xrightarrow{\bullet} C_4$ to $C_1 \xrightarrow{\bullet} C_4$

Shortening rule 2: $C_1 \longrightarrow C_2 \longrightarrow C_3$ to $C_1 \longrightarrow C_3$

Stability property: if $C_1 \longrightarrow C_2$ then $\mathtt{write}(C_1) \supseteq \mathtt{write}(C_2)$.

These rules allow to shorten any run to a run of length < 4k

For tame algorithms existence of a short run can be encoded as an existentially quantified linear program.

Thm: For tame algorithms with sporadic communication predicates and stability property it is decidable if an algorithm solves consensus.

Decidability for a syntactic fragment

If (HO=S and |HO|> thr_s) then inp,dec:=min(HO),smor(HO)

Special case:
Only two thresholds, one for singletons and one for other sets.

One can show that the only possible forms of instructions are:

For singletons:

If (HO={a} and |HO|>thr_s) then inp:=smor(HO); dec:=smor(HO)

For other sets:

If (HO=S and |HO|>thr_s) then inp:=smor(HO);

Obs 1: $thr_u \geq 1/2$

Obs 2: $thr_m \geq 2(1 - thr_u)$

Decidability for a syntactic fragment

For singletons:

If (HO={a} and |HO|>thr_s) then inp:=smor(HO); dec:=smor(HO)

For other sets:

If (HO=S and |HO|>thr_s) then inp:=smor(HO);

Obs 1: $thr_u \geq 1/2$

Obs 2: $thr_m \geq 2(1 - thr_u)$

Sporadic communication predicate: $\exists_{r_1 \leq \dots \leq r_k} \bigwedge \theta_i(r_i) \wedge \forall_{r \neq r_1, \dots, r_k} \theta(r)$

There must be i<j with:

$$\theta_i \equiv HO_= \wedge |HO| > c_1 \cdot |HO| \qquad \theta_j \equiv |HO| > c_2 \cdot |HO|$$

Decidability for a bigger syntactic fragment

Tame algorithm: For every phase P and every $S \subseteq D \cup \{\bot\}$ we have an existentially quantified set of linear constraints $L(P,S)$ s.t. for every configuration C:

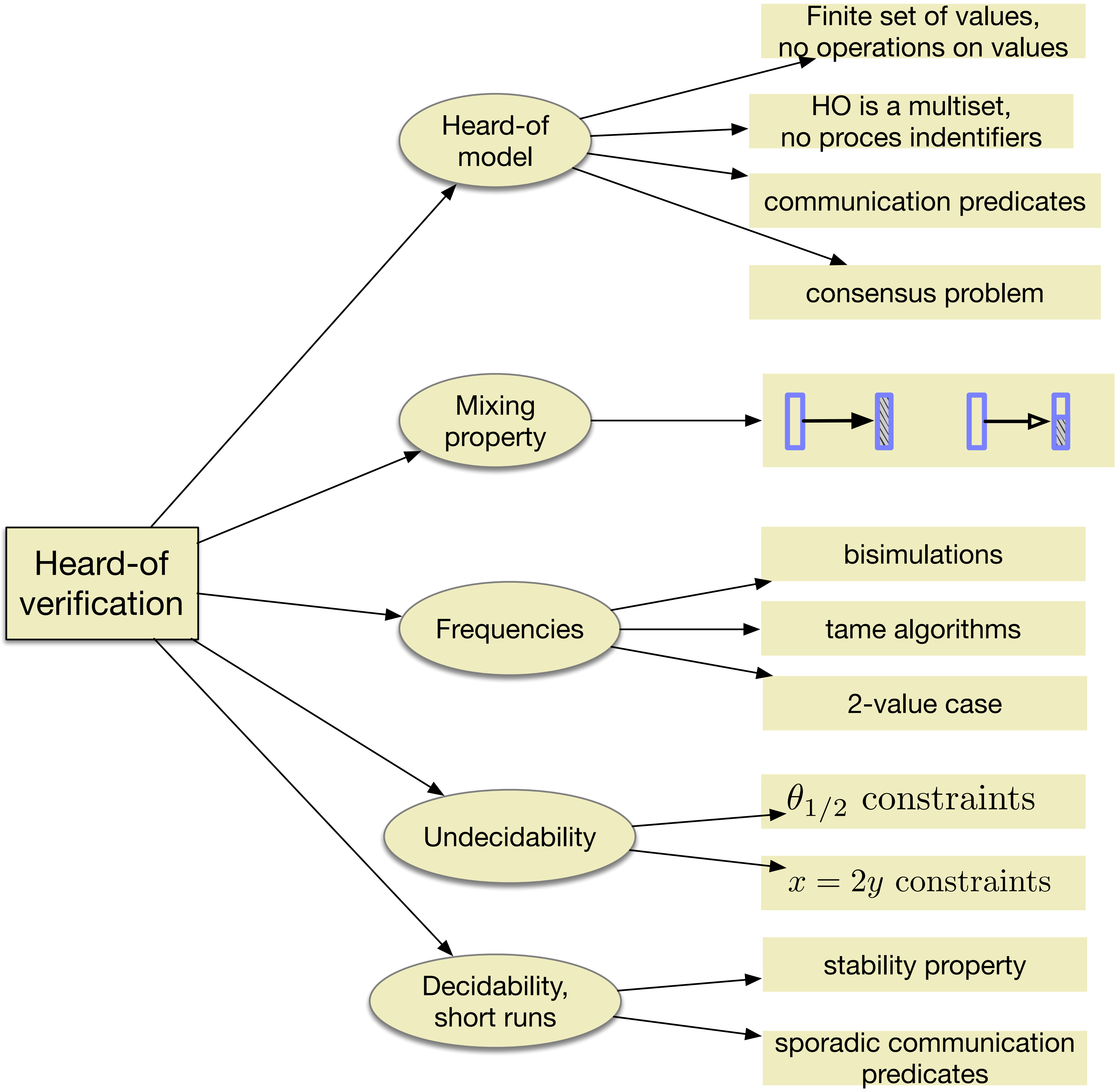
$$S \in \mathtt{write}(C,P) \qquad \text{iff} \qquad f_C \vDash L(P,S)$$

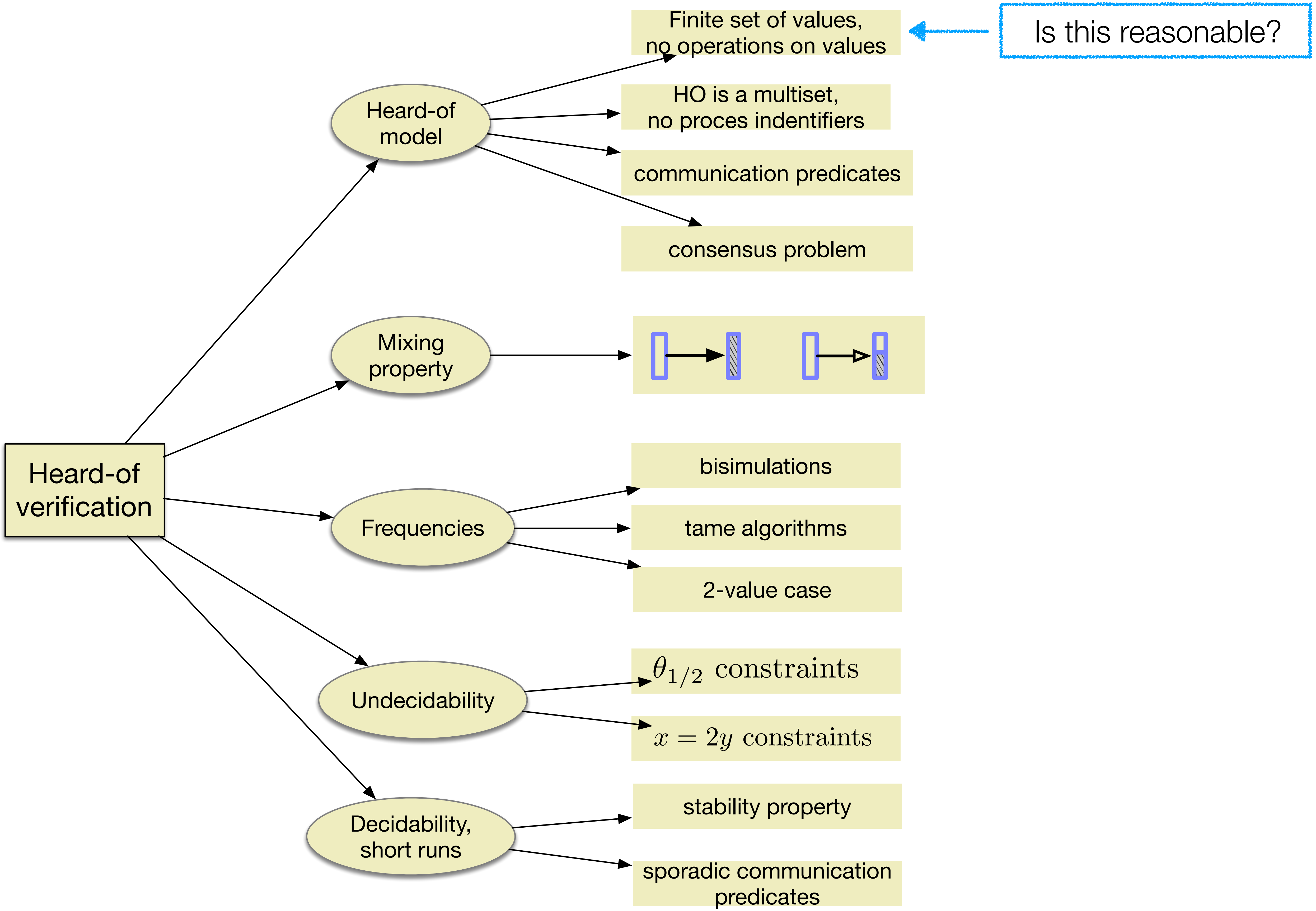
Relative linear constraints

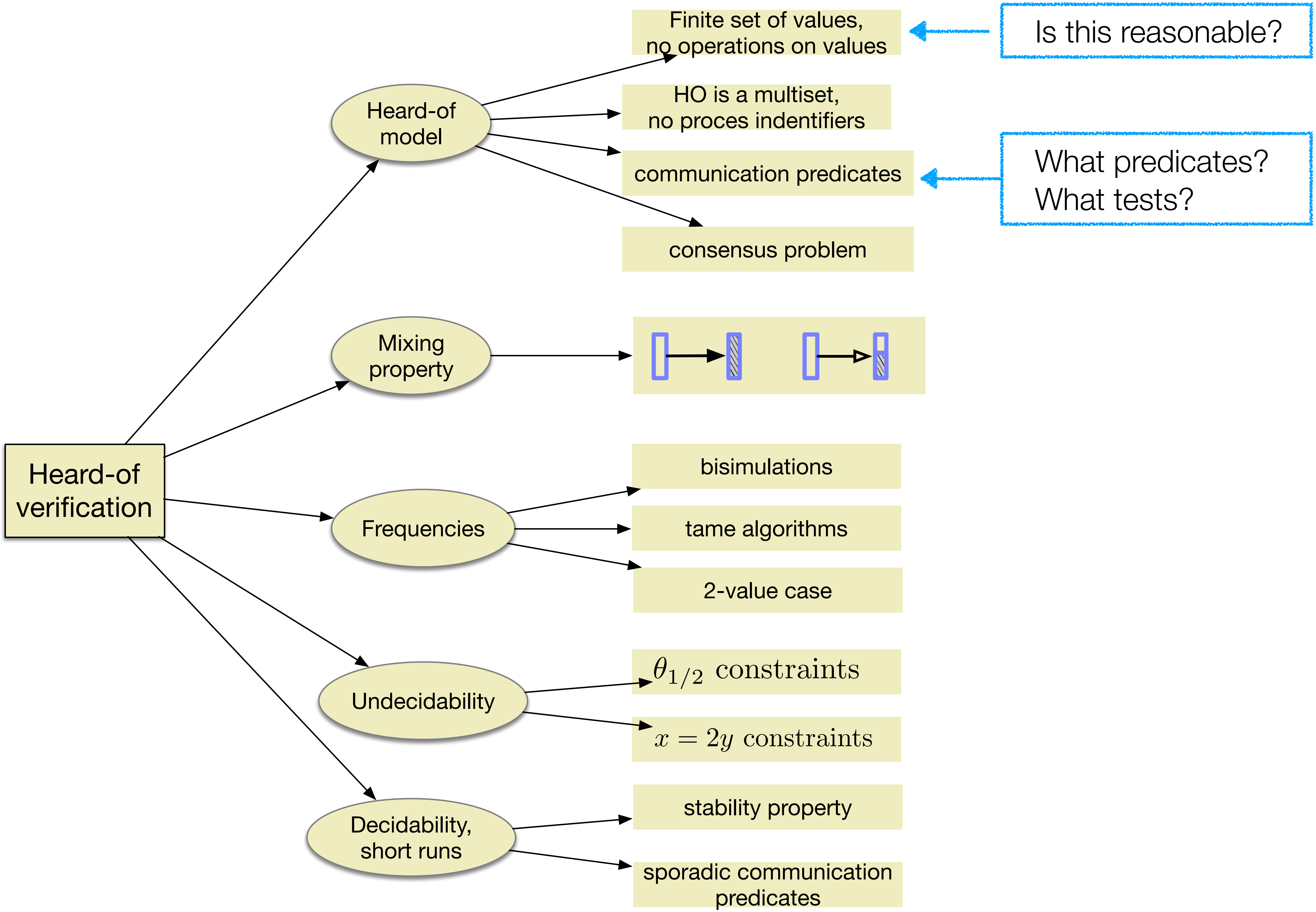
$$\text{if } (HO = S \wedge |HO| \geq thr_s|\Pi|) \text{ then}$$

$$\begin{array}{l} \text{if } L_1(HO) \text{ then } inp = a_1 \\ \vdots \\ \text{if } L_k(HO) \text{ then } inp = a_k \end{array}$$

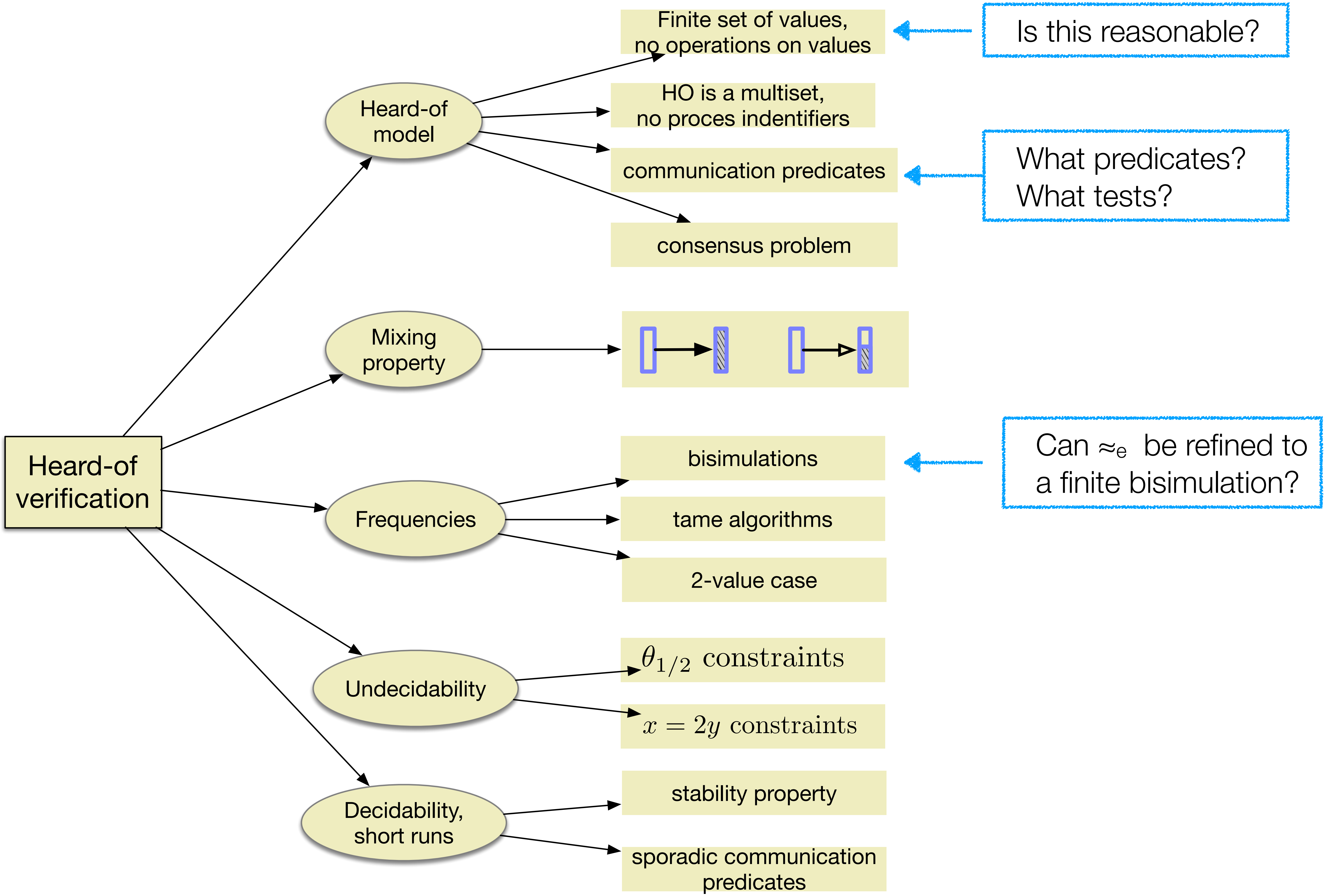
Thm: Consensus is decidable for this fragment.

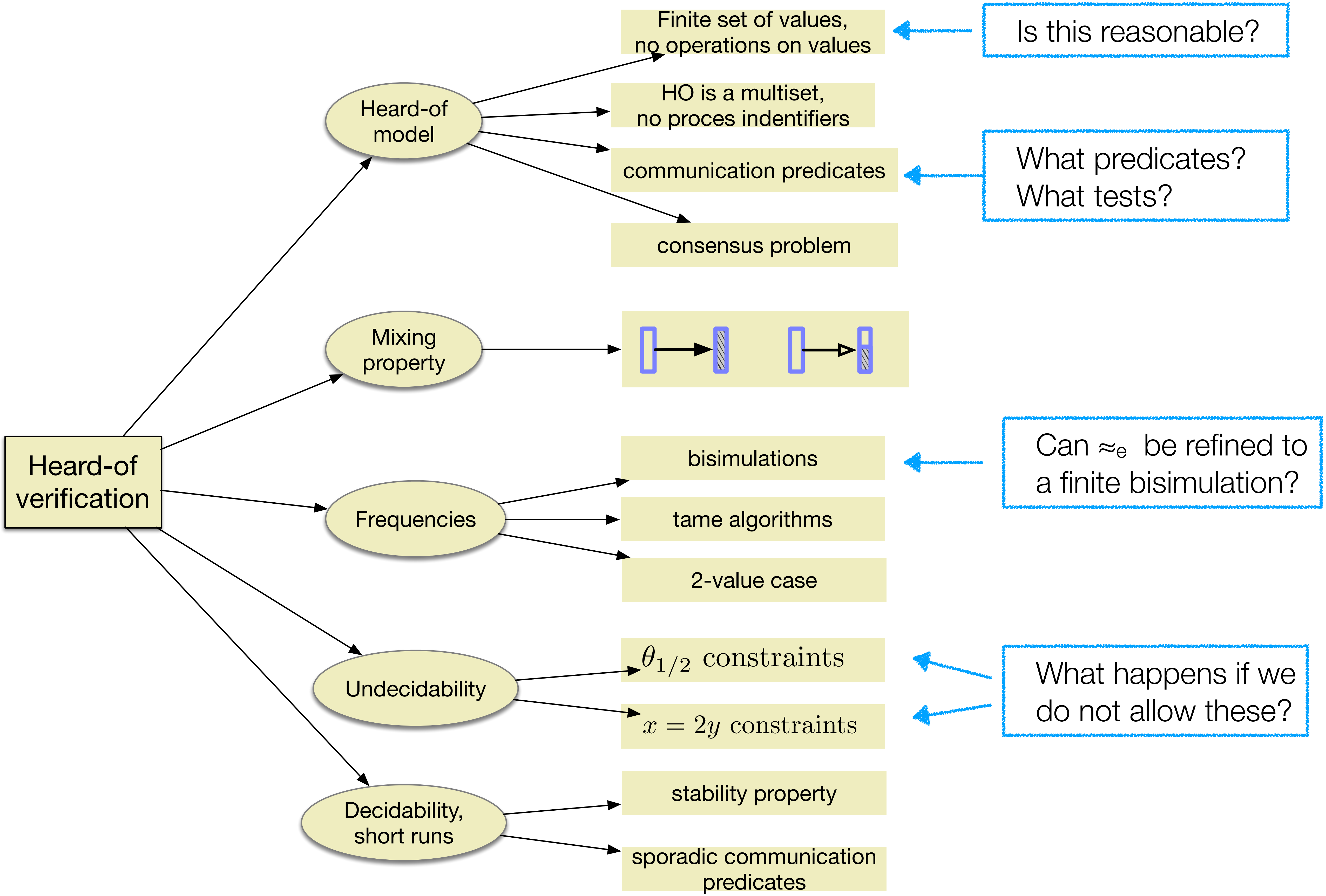

Heard-of verification
Heard-of model
Finite set of values, no operations on values
HO is a multiset, no proces indentifiers
communication predicates
consensus problem
Mixing property
Frequencies
bisimulations
tame algorithms
2-value case
Undecidability
$\theta_{1/2}$ constraints
$x = 2y$ constraints
Decidability, short runs
stability property
sporadic communication predicates


Is this reasonable?
Finite set of values,
no operations on values
HO is a multiset,
no proces indentifiers
communication predicates
consensus problem
Heard-of
model
Mixing
property
Heard-of
verification
bisimulations
tame algorithms
2-value case
Frequencies
$\theta_{1/2}$ constraints
$x = 2y$ constraints
Undecidability
stability property
sporadic communication
predicates
Decidability,
short runs

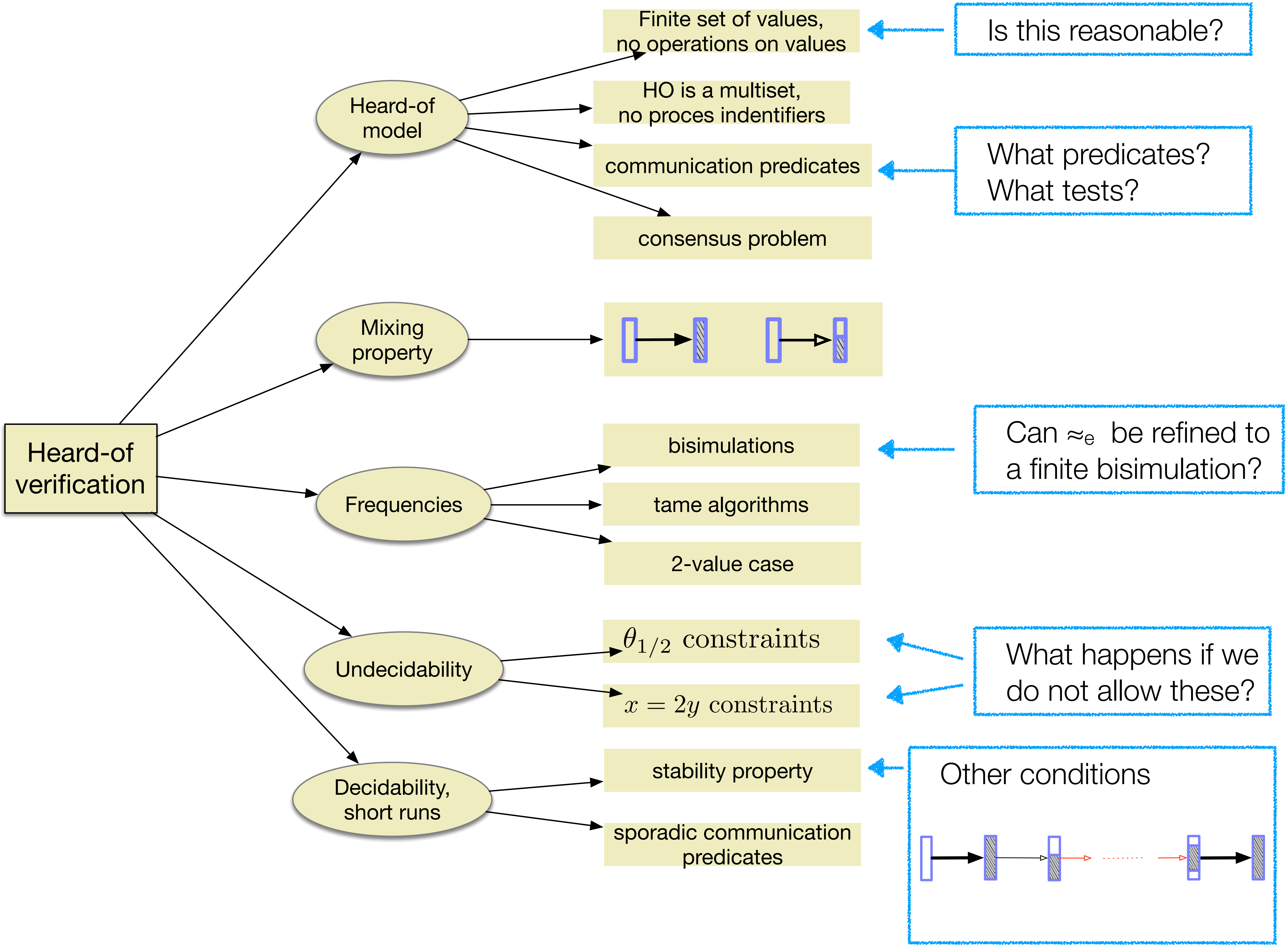


Heard-of verification
Heard-of model
Finite set of values, no operations on values
HO is a multiset, no proces indentifiers
communication predicates
consensus problem
Is this reasonable?
What predicates? What tests?
Mixing property
Frequencies
bisimulations
tame algorithms
2-value case
Undecidability
$\theta_{1/2}$ constraints
$x = 2y$ constraints
Decidability, short runs
stability property
sporadic communication predicates


Finite set of values,
no operations on values
Is this reasonable?
HO is a multiset,
no proces indentifiers
Heard-of
model
communication predicates
What predicates?
What tests?
consensus problem
Mixing
property
Heard-of
verification
bisimulations
Can $\approx_e$ be refined to
a finite bisimulation?
Frequencies
tame algorithms
2-value case
$\theta_{1/2}$ constraints
Undecidability
$x = 2y$ constraints
stability property
Decidability,
short runs
sporadic communication
predicates



Heard-of verification
Heard-of model
Finite set of values, no operations on values
HO is a multiset, no proces indentifiers
communication predicates
consensus problem
Is this reasonable?
What predicates? What tests?
Mixing property
Frequencies
bisimulations
tame algorithms
2-value case
Can ≈e be refined to a finite bisimulation?
Undecidability
$\theta_{1/2}$ constraints
$x = 2y$ constraints
What happens if we do not allow these?
Decidability, short runs
stability property
sporadic communication predicates


Heard-of verification
Heard-of model
Finite set of values, no operations on values
HO is a multiset, no proces indentifiers
communication predicates
consensus problem
Is this reasonable?
What predicates? What tests?
Mixing property
Frequencies
bisimulations
tame algorithms
2-value case
Can $\approx_e$ be refined to a finite bisimulation?
Undecidability
$\theta_{1/2}$ constraints
$x = 2y$ constraints
What happens if we do not allow these?
Decidability, short runs
stability property
sporadic communication predicates
Other conditions