Synthesis of small Population Protocols

Martin Helfrich

Technical University of Munich

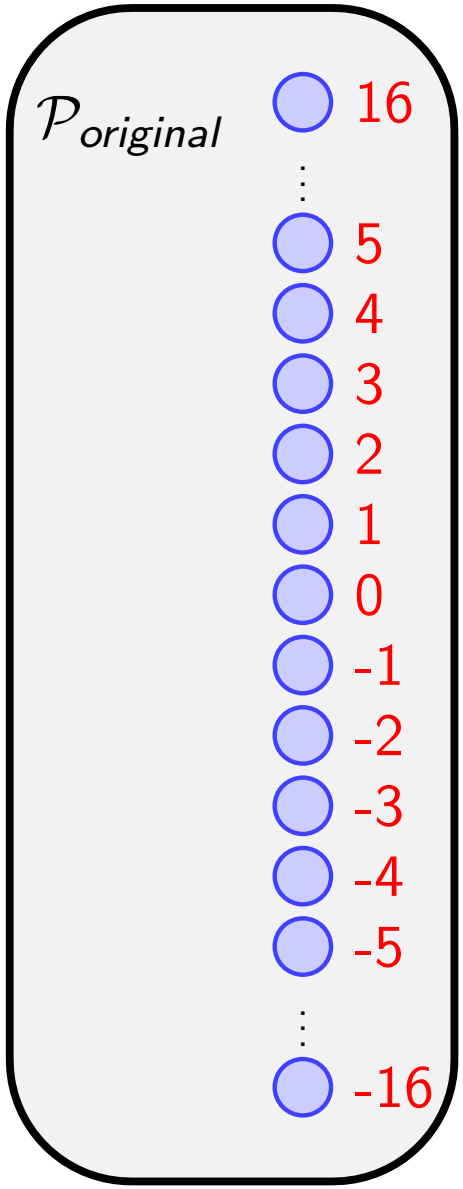
May 2019

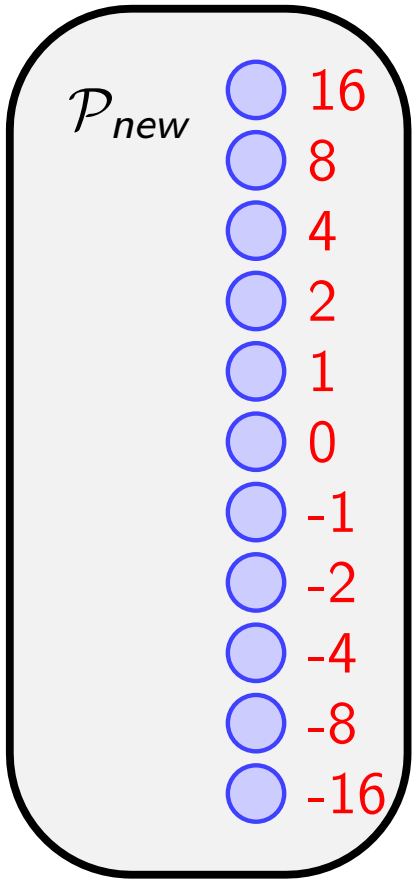
Introduction

Original

size:
EXP($|\varphi|$)

leaders: 0





Binary

size:
POLY($|\varphi|$)

leaders:
POLY($|\varphi|$)

SOPP: Single Output Population Protocols

Idea: Output is determined by single "output leader"

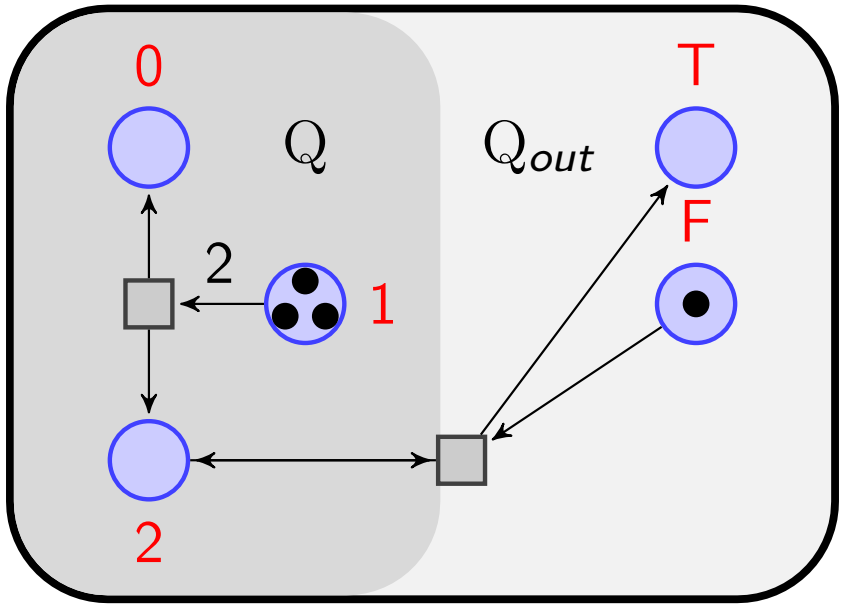


Figure: Single Output Population Protocol as Petri Net

SOPP: Single Output Population Protocols

Idea: Output is determined by single "output leader"
$\Rightarrow$ every configuration is consensus

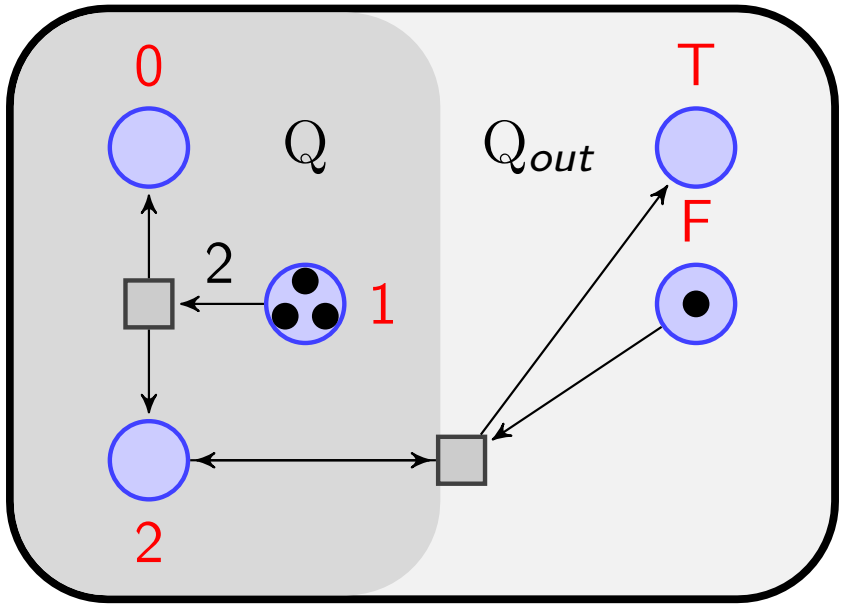


Figure: Single Output Population Protocol as Petri Net

SOPP to PP

- Add output bit to every "normal" state
  $\Rightarrow$ for each state q we get $(q, 0)$ and $(q, 1)$

SOPP to PP

- Add output bit to every "normal" state
  $\Rightarrow$ for each state q we get $(q, 0)$ and $(q, 1)$
- Transitions behave as before:

$$q_1, q_2, \ldots \to q_1', q_2', \ldots$$
$$\Downarrow$$
$$(q_1, b_1), (q_2, b_2), \ldots \to (q_1', b_1), (q_2', b_2), \ldots$$

SOPP to PP

- Add output bit to every "normal" state
  $\Rightarrow$ for each state q we get $(q, 0)$ and $(q, 1)$
- Transitions behave as before:

$$q_1, q_2, \ldots \to q_1', q_2', \ldots$$
$$\Downarrow$$
$$(q_1, b_1), (q_2, b_2), \ldots \to (q_1', b_1), (q_2', b_2), \ldots$$

- Leaders can change output bit of others:

$$T, (q, 0) \to T, (q, 1)$$
$$F, (q, 1) \to F, (q, 0)$$

Remainder

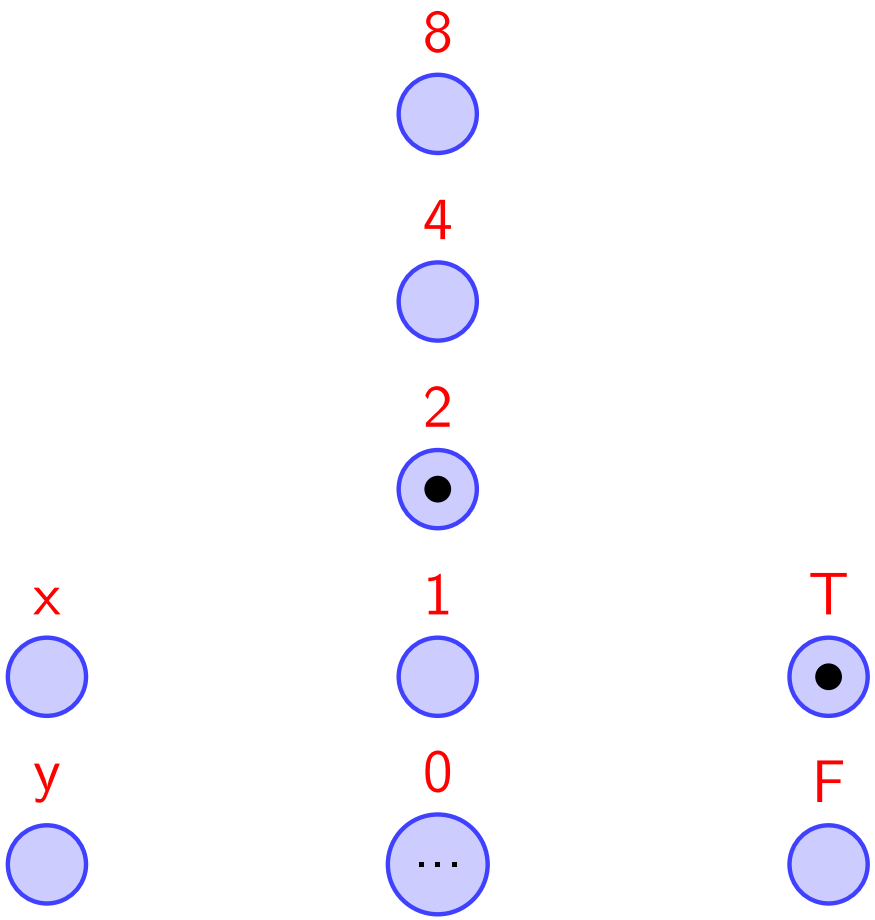
$\sum_{i=1}^{k} a_i x_i \equiv_d c$ with scaling parameter m [1]

Let:
$n \stackrel{\text{def}}{=} \lceil \log 2m \rceil + \text{size}(d)$

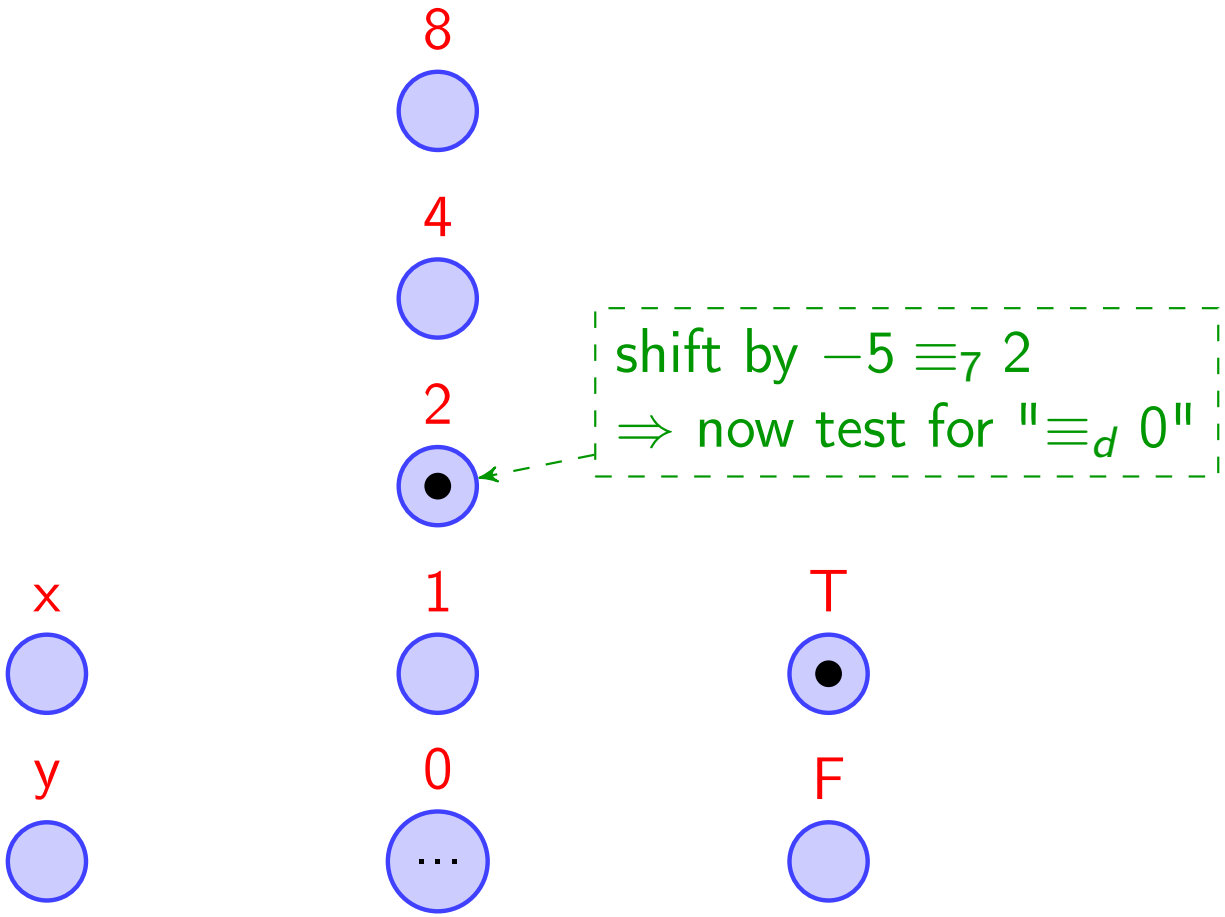
Idea: Represent values in binary!
$\Rightarrow$ every agent can remembers a power of two up to 2^n

[1] scales construction for combination of m base protocols

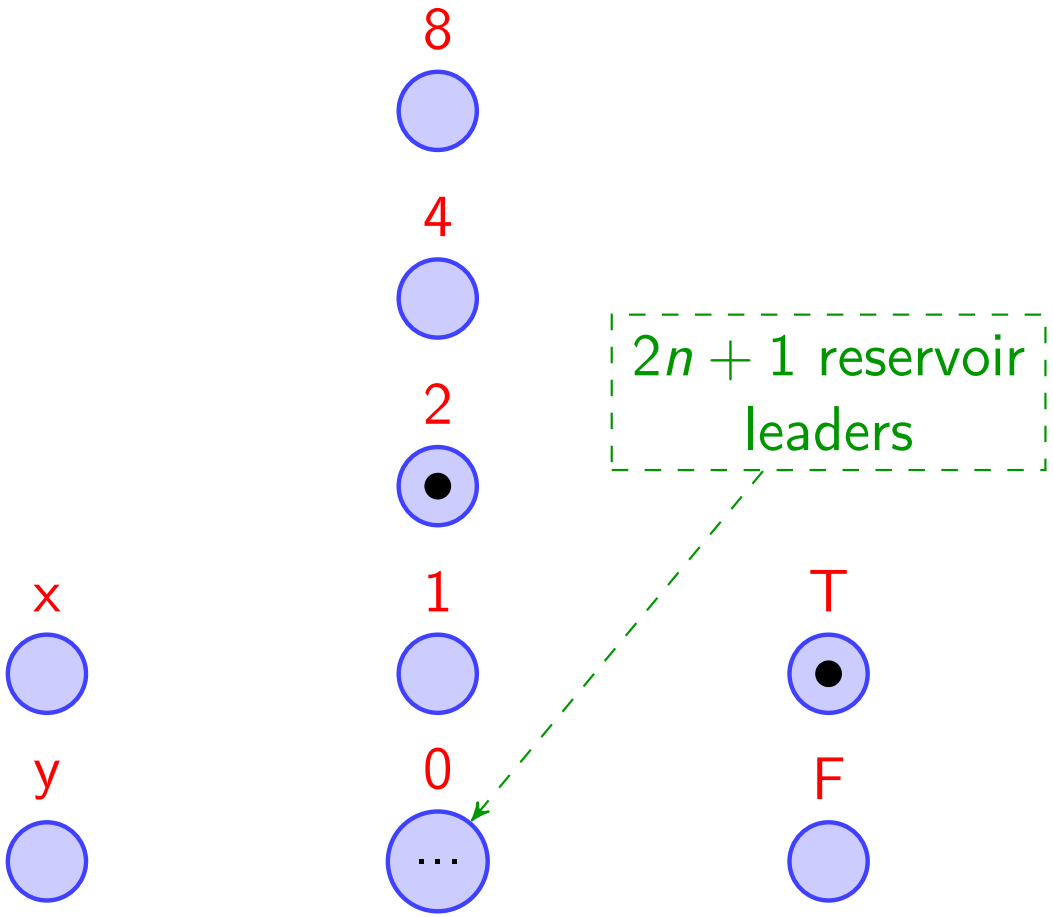
$-2x + 50y \equiv_7 5$



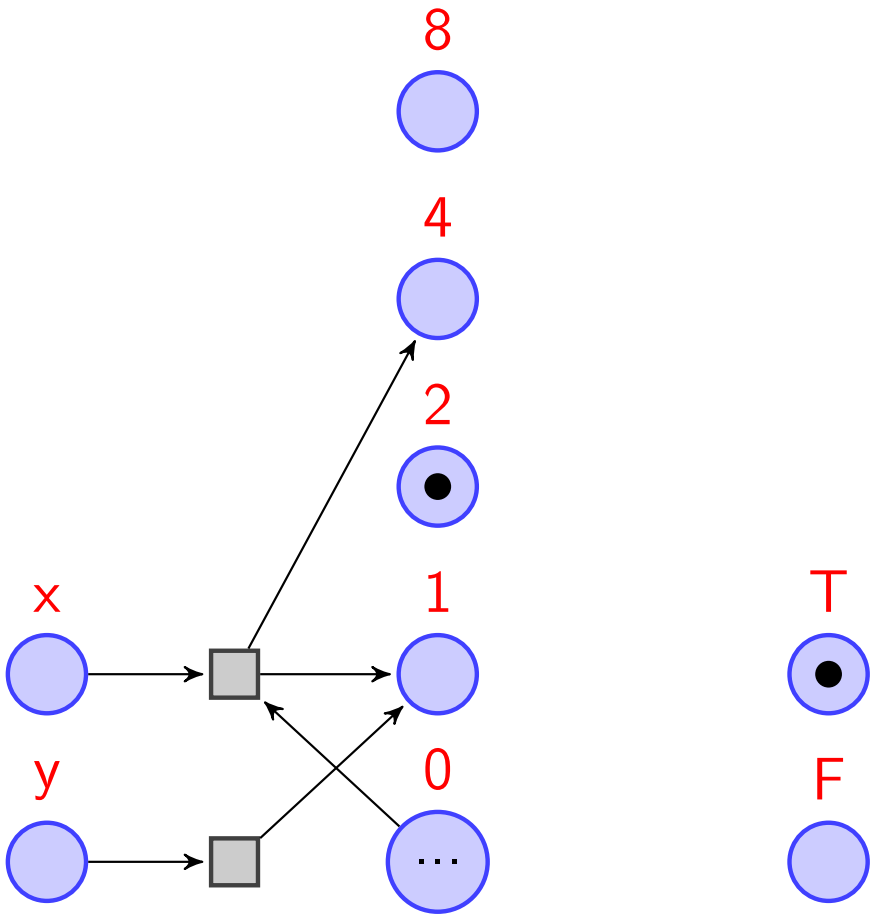
$-2x + 50y \equiv_7 5$ **shift remainder:** add leaders with value $d - c$



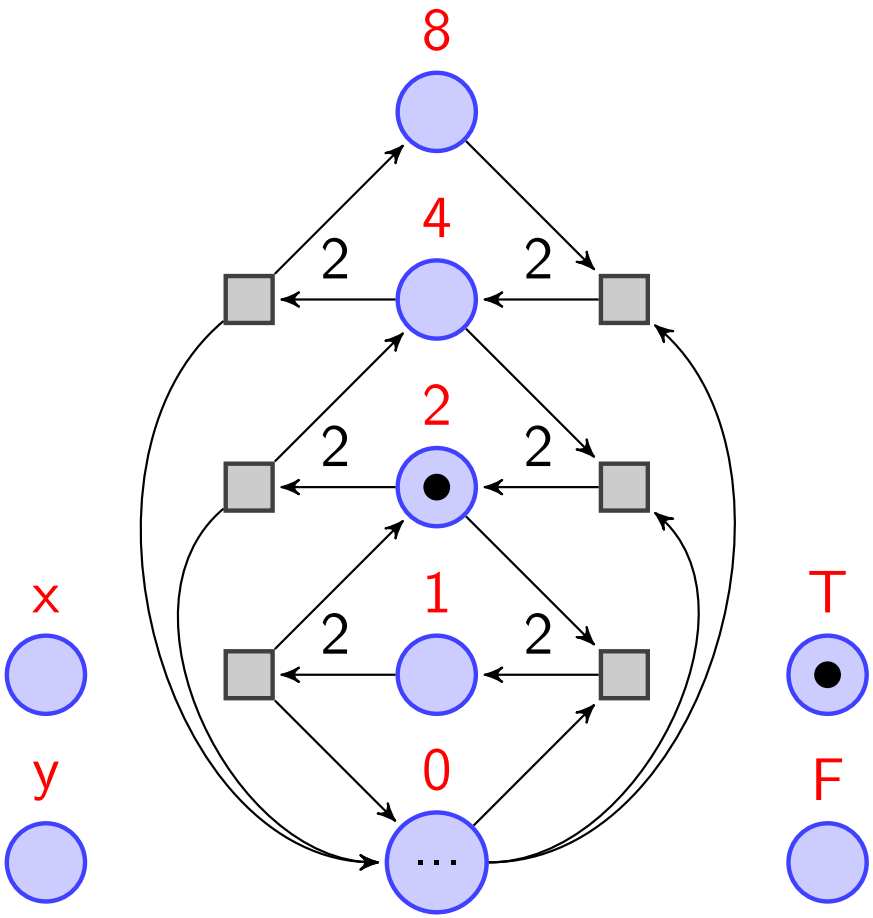
$-2x + 50y \equiv_7 5$ **reservoir:** add $2n + 1$ leaders in state 0



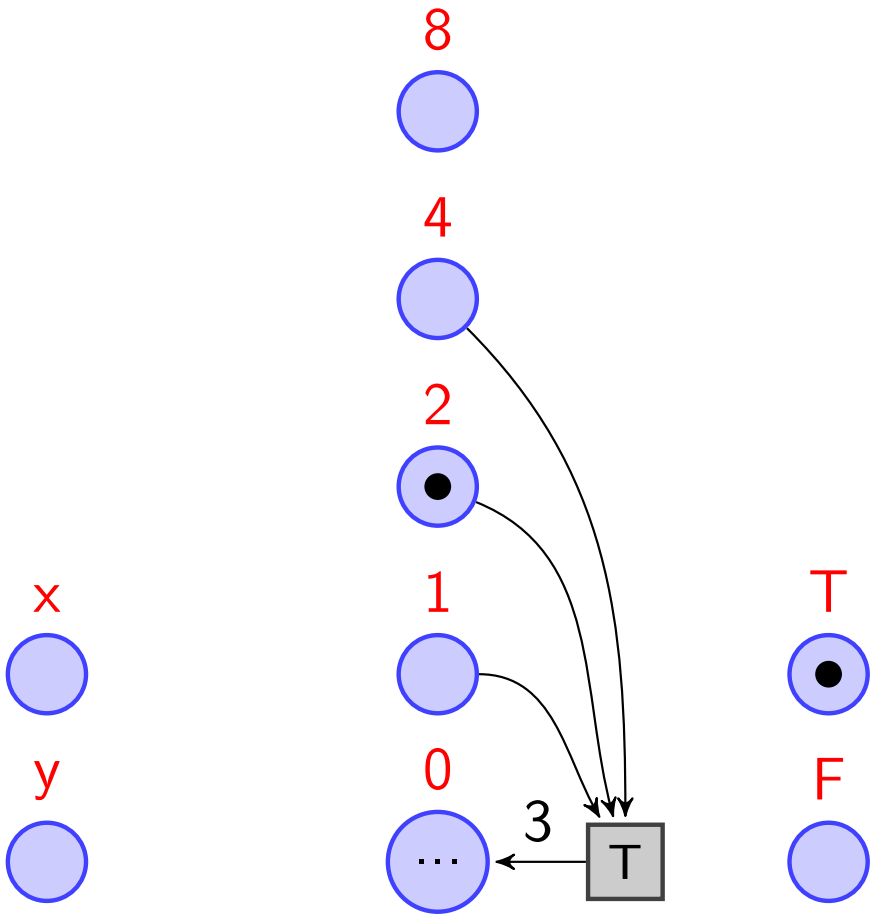
$-2x + 50y \equiv_7 5$ **input transitions:** $x_i, 0, 0, \ldots \to rep(a_i)$



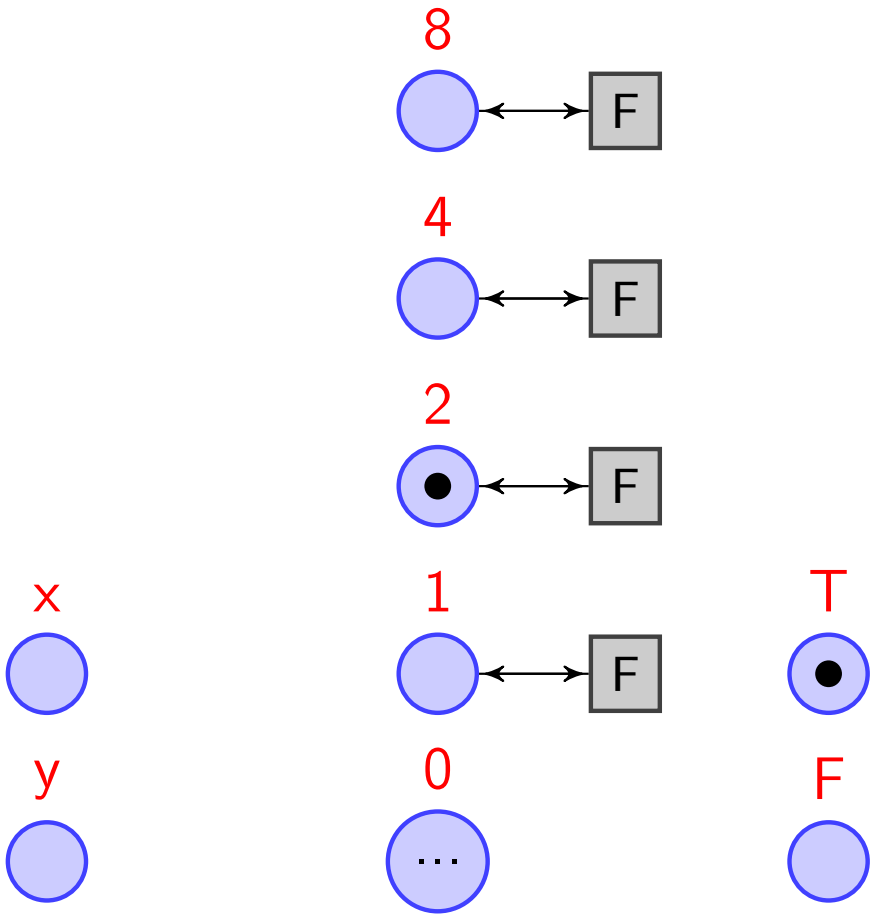
$-2x + 50y \equiv_7 5$ **up & down:** $2^i, 2^i \leftrightarrow 2^{i+1}, 0$ $-2^i, -2^i \leftrightarrow -2^{i+1}, 0$



$-2x + 50y \equiv_7 5$ **mod:** $X, rep(a_i\%d) \to T, 0, 0, \ldots$ with $X \in \{T, F\}$



$-2x + 50y \equiv_7 5$ **signal:** $2^i, T \to 2^i, F \quad -2^i, F \to -2^i, T$



Remainder: Correctness

Assumption: We can always put n agents back in state 0.

Correctness:

Remainder: Correctness

Assumption: We can always put n agents back in state 0.

Correctness:

- "total value mod d" is invariant

Remainder: Correctness

Assumption: We can always put n agents back in state 0.

Correctness:

- "total value mod d" is invariant
- all inputs will be emptied

Remainder: Correctness

Assumption: We can always put n agents back in state 0.

Correctness:

- "total value mod d" is invariant
- all inputs will be emptied
- finally: total value between 0 and $d - 1$

Remainder: Correctness

Assumption: We can always put n agents back in state 0.

Correctness:

- "total value mod d" is invariant
- all inputs will be emptied
- finally: total value between 0 and $d-1$
  - $\sum_{i=1}^{k} a_i x_i + d - c \not\equiv_d 0$: output is False because of signal

Remainder: Correctness

Assumption: We can always put n agents back in state 0.

Correctness:

- "total value mod d" is invariant
- all inputs will be emptied
- finally: total value between 0 and $d-1$
  - $\sum_{i=1}^{k} a_i x_i + d - c \not\equiv_d 0$: output is False because of signal
  - $\sum_{i=1}^{k} a_i x_i + d - c \equiv_d 0$: output is True
    because of initial output or last mod

Remainder: Assumption

"value states": $Q_{val} \stackrel{\text{def}}{=} \{0, 1, 2, 4, ..., 2^n\}$
"extreme states": $Q_{ext} \stackrel{\text{def}}{=} \{2^n\}$

To show: We can always put n agents back in state 0.

Let $C_0 \xrightarrow{\pi} C$. Let i be the number of "input transitions" in π.

Remainder: Assumption

"value states": $Q_{val} \stackrel{\text{def}}{=} \{0, 1, 2, 4, ..., 2^n\}$
"extreme states": $Q_{ext} \stackrel{\text{def}}{=} \{2^n\}$

To show: We can always put n agents back in state 0.

Let $C_0 \xrightarrow{\pi} C$. Let i be the number of "input transitions" in π.

- $C_0(Q_{val}) \geq 2n + 1$

Remainder: Assumption

"value states": $Q_{val} \stackrel{\text{def}}{=} \{0, 1, 2, 4, ..., 2^n\}$
"extreme states": $Q_{ext} \stackrel{\text{def}}{=} \{2^n\}$

To show: We can always put n agents back in state 0.

Let $C_0 \xrightarrow{\pi} C$. Let i be the number of "input transitions" in π.

- $C_0(Q_{val}) \geq 2n + 1$
- $C(Q_{val}) = C_0(Q_{val}) + i$
- $C(Q_{ext}) \leq i$

Remainder: Assumption

"value states": $Q_{val} \stackrel{\text{def}}{=} \{0, 1, 2, 4, ..., 2^n\}$
"extreme states": $Q_{ext} \stackrel{\text{def}}{=} \{2^n\}$

To show: We can always put n agents back in state 0.

Let $C_0 \xrightarrow{\pi} C$. Let i be the number of "input transitions" in π.

- $C_0(Q_{val}) \geq 2n + 1$
- $C(Q_{val}) = C_0(Q_{val}) + i$
- $C(Q_{ext}) \leq i$

$\Rightarrow$ use "up transitions" as long as possible

Remainder: Assumption

"value states": $Q_{val} \stackrel{\text{def}}{=} \{0, 1, 2, 4, ..., 2^n\}$
"extreme states": $Q_{ext} \stackrel{\text{def}}{=} \{2^n\}$

To show: We can always put n agents back in state 0.

Let $C_0 \xrightarrow{\pi} C$. Let i be the number of "input transitions" in π.

- $C_0(Q_{val}) \geq 2n + 1$
- $C(Q_{val}) = C_0(Q_{val}) + i$
- $C(Q_{ext}) \leq i$

$\Rightarrow$ use "up transitions" as long as possible
$\Rightarrow$ only one agent in each of the states: $1, 2, 4, ..., 2^{n-1}$

Remainder: Assumption

"value states": $Q_{val} \stackrel{\text{def}}{=} \{0, 1, 2, 4, ..., 2^n\}$
"extreme states": $Q_{ext} \stackrel{\text{def}}{=} \{2^n\}$

To show: We can always put n agents back in state 0.

Let $C_0 \xrightarrow{\pi} C$. Let i be the number of "input transitions" in π.

- $C_0(Q_{val}) \geq 2n + 1$
- $C(Q_{val}) = C_0(Q_{val}) + i$
- $C(Q_{ext}) \leq i$

$\Rightarrow$ use "up transitions" as long as possible
$\Rightarrow$ only one agent in each of the states: $1, 2, 4, ..., 2^{n-1}$
$\Rightarrow$ n agents in state 0

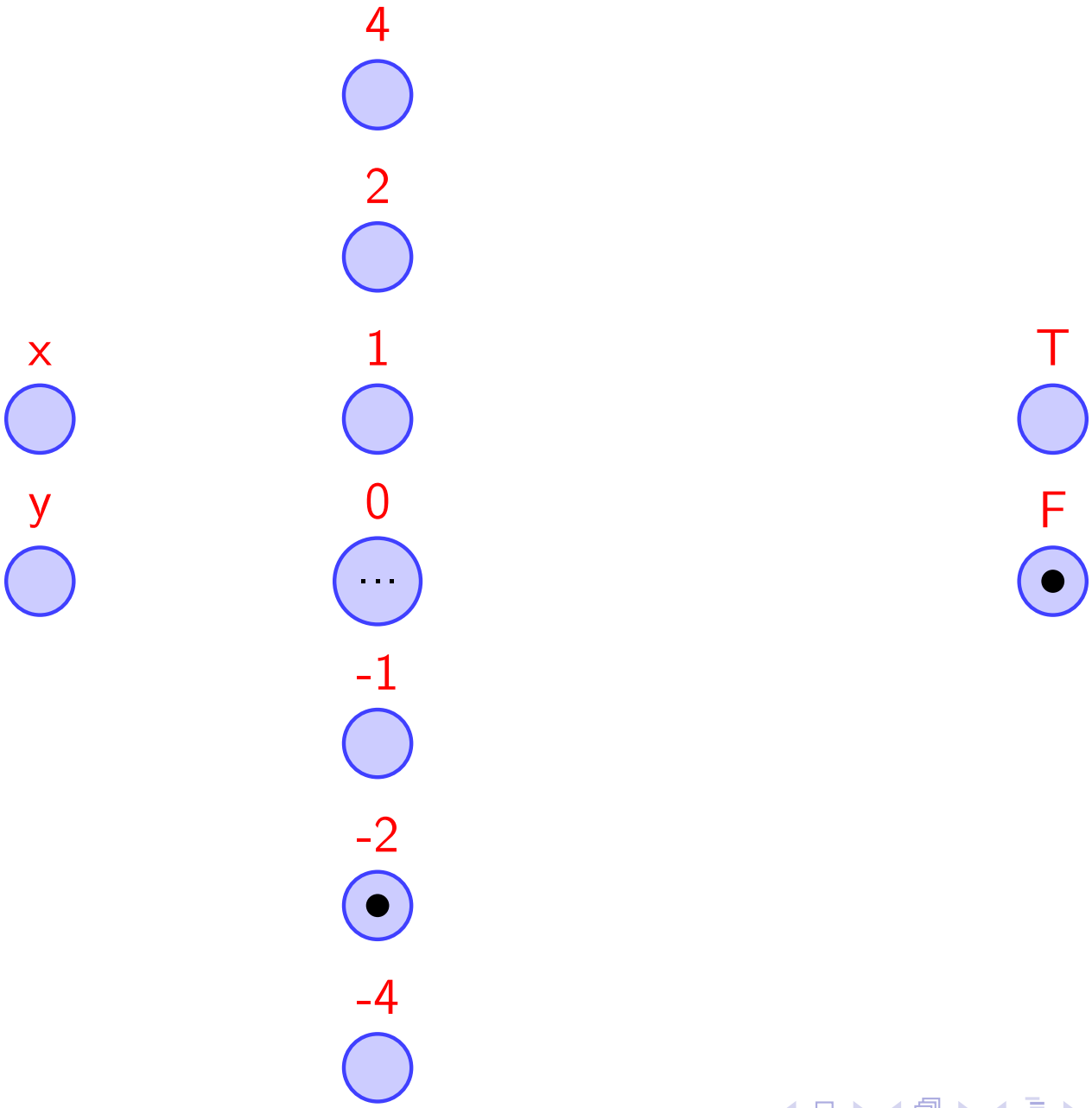
Threshold

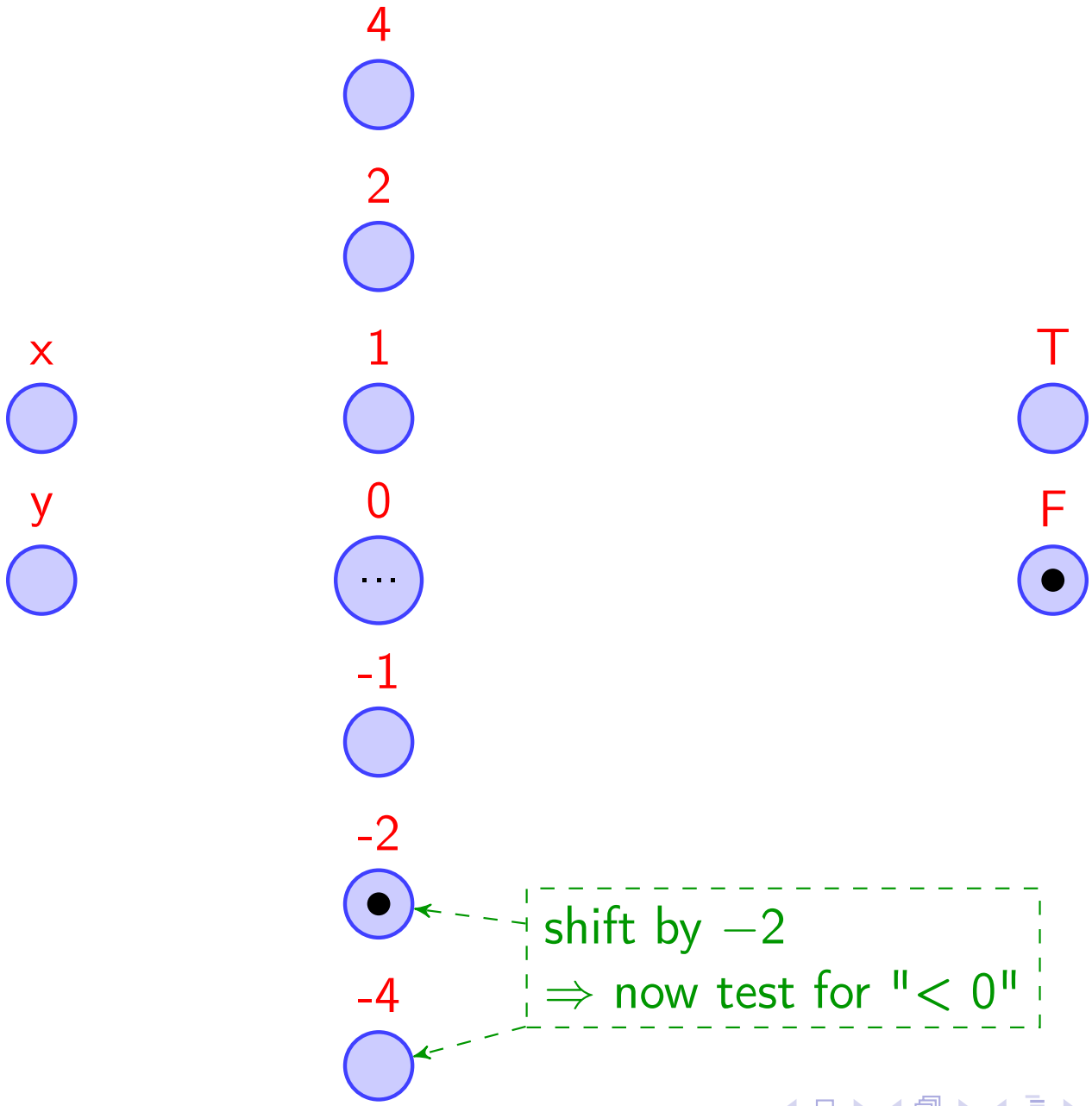
$\sum_{i=1}^{k} a_i x_i < c$ with scaling parameter m [2]

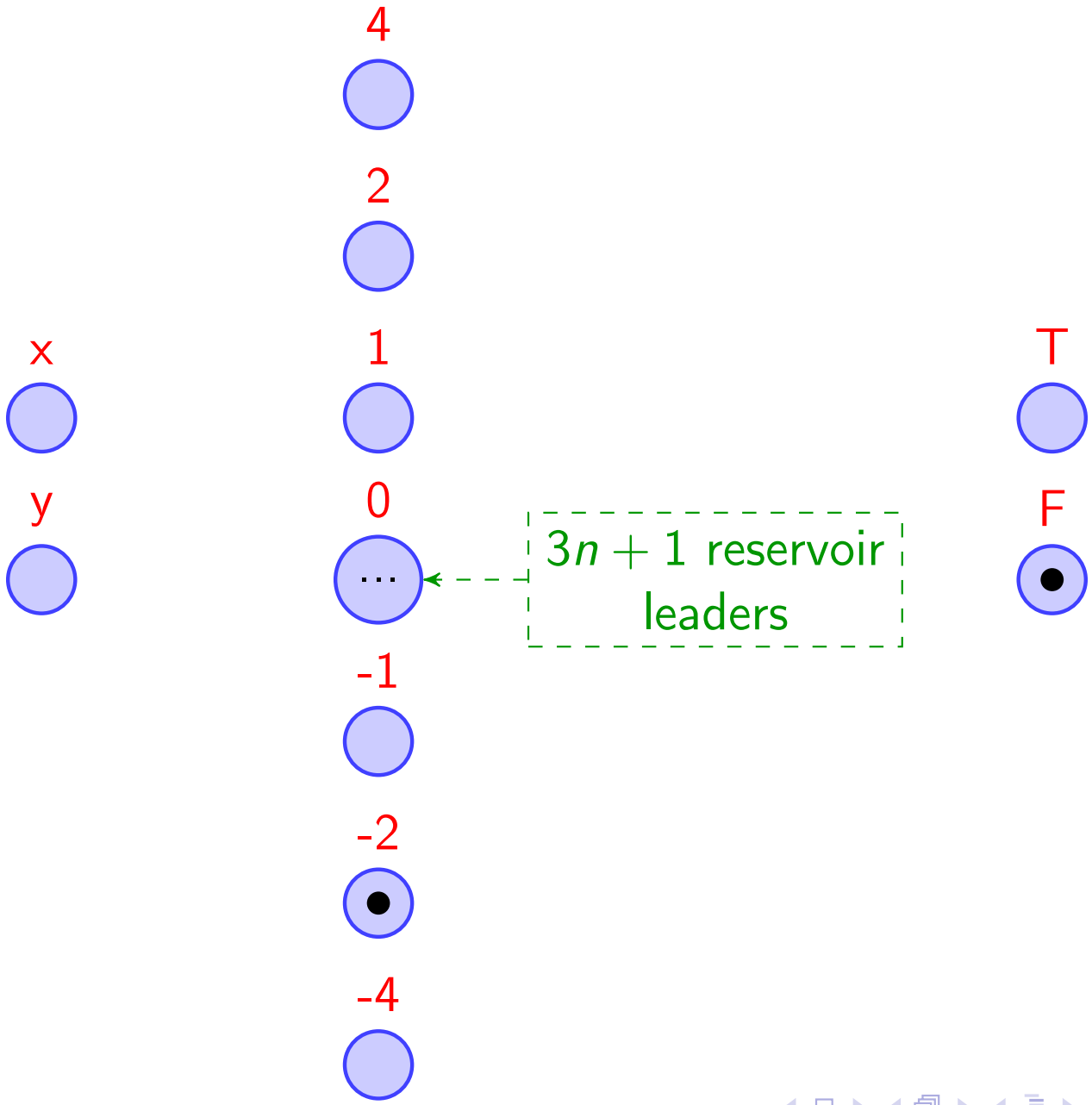
Let:
$b_{max} \stackrel{\text{def}}{=} \max(1, \max_i |a_i|, |c|)$
$n \stackrel{\text{def}}{=} \lceil \log 2m \rceil + \text{size}(b_{max})$

[2] scales construction for combination of m base protocols

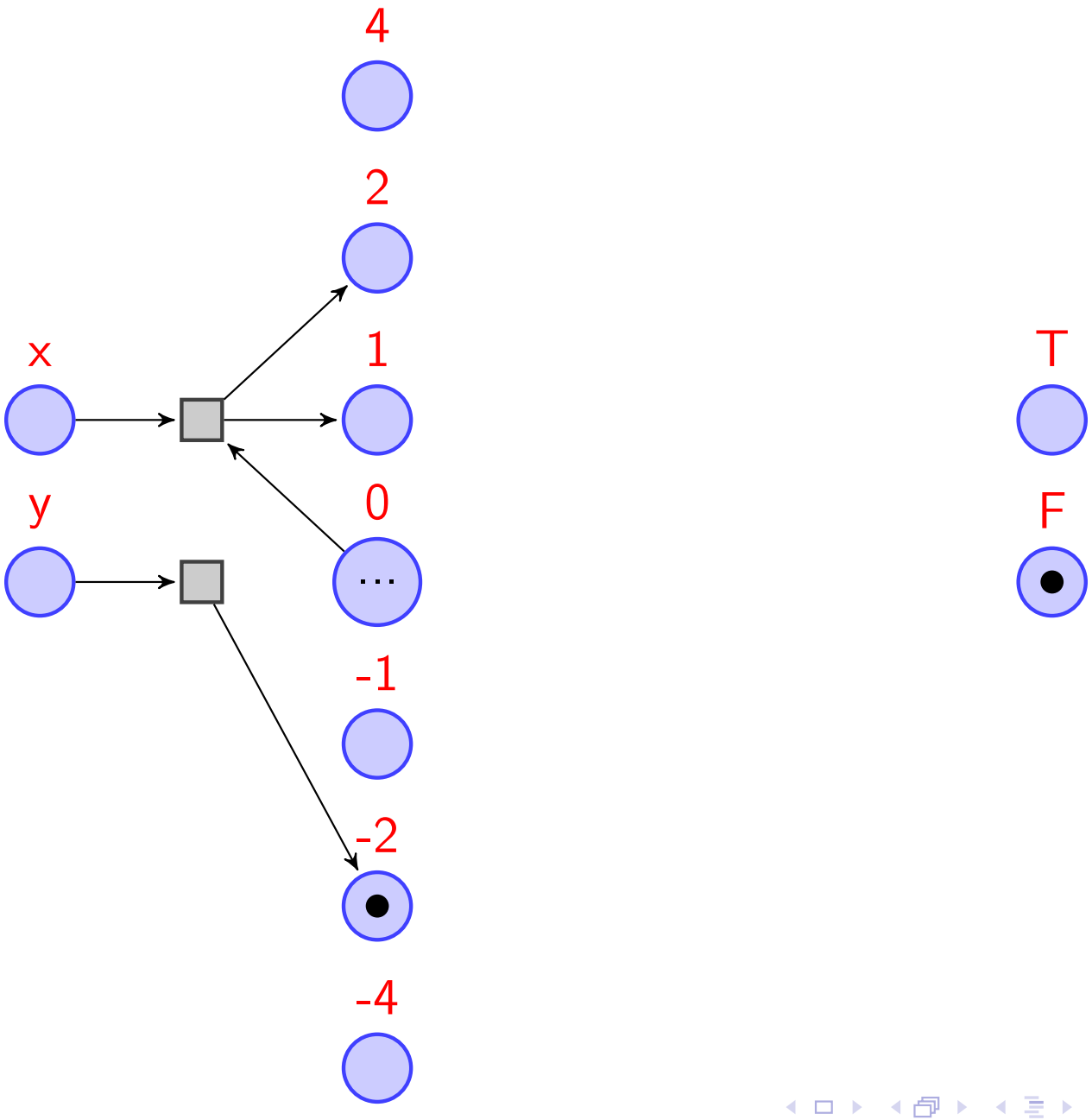
$3x - 2y < 2$

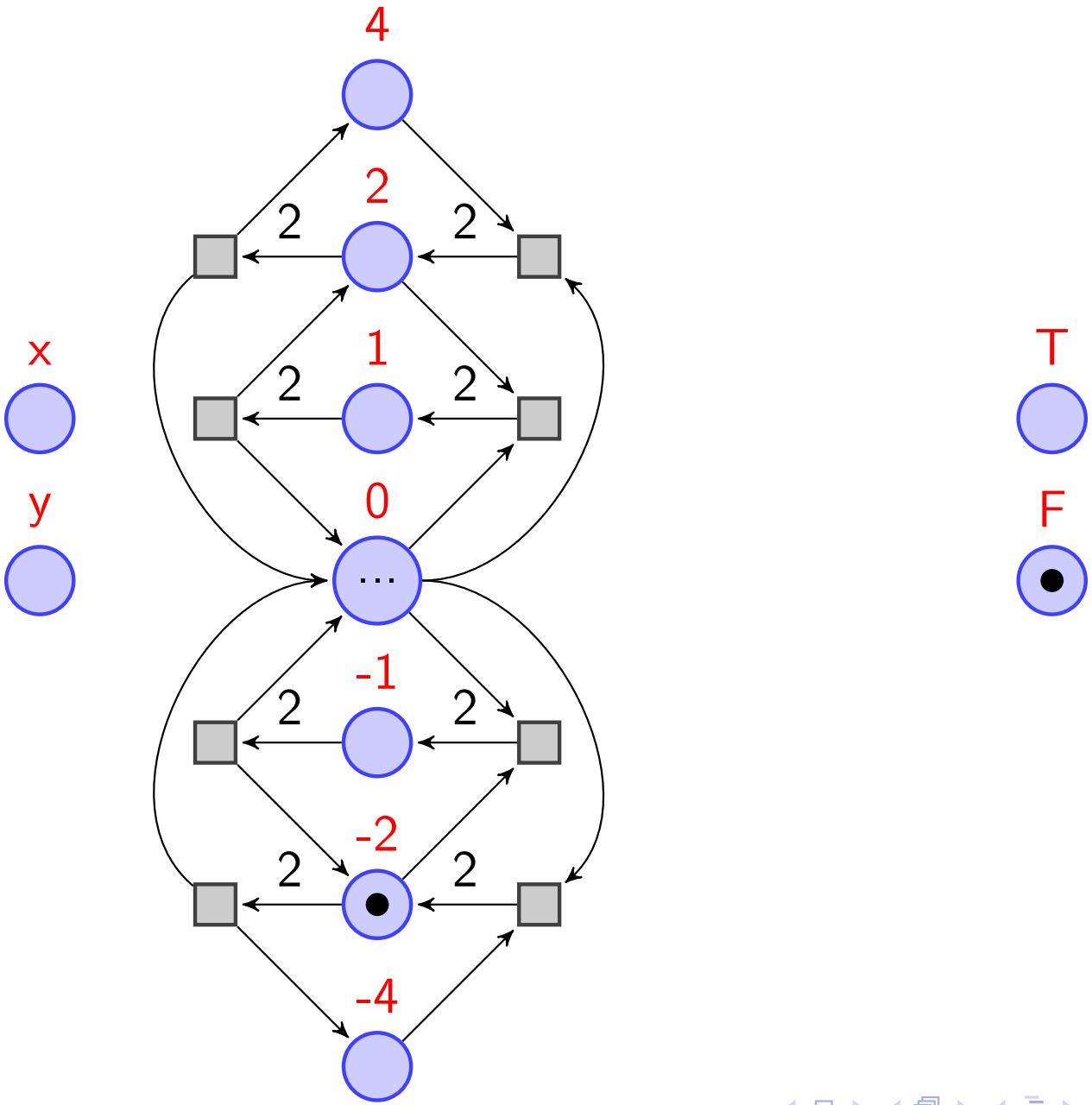



4
2
x
1
T
y
0
F
...
-1
-2
shift by −2
⇒ now test for "< 0"
-4

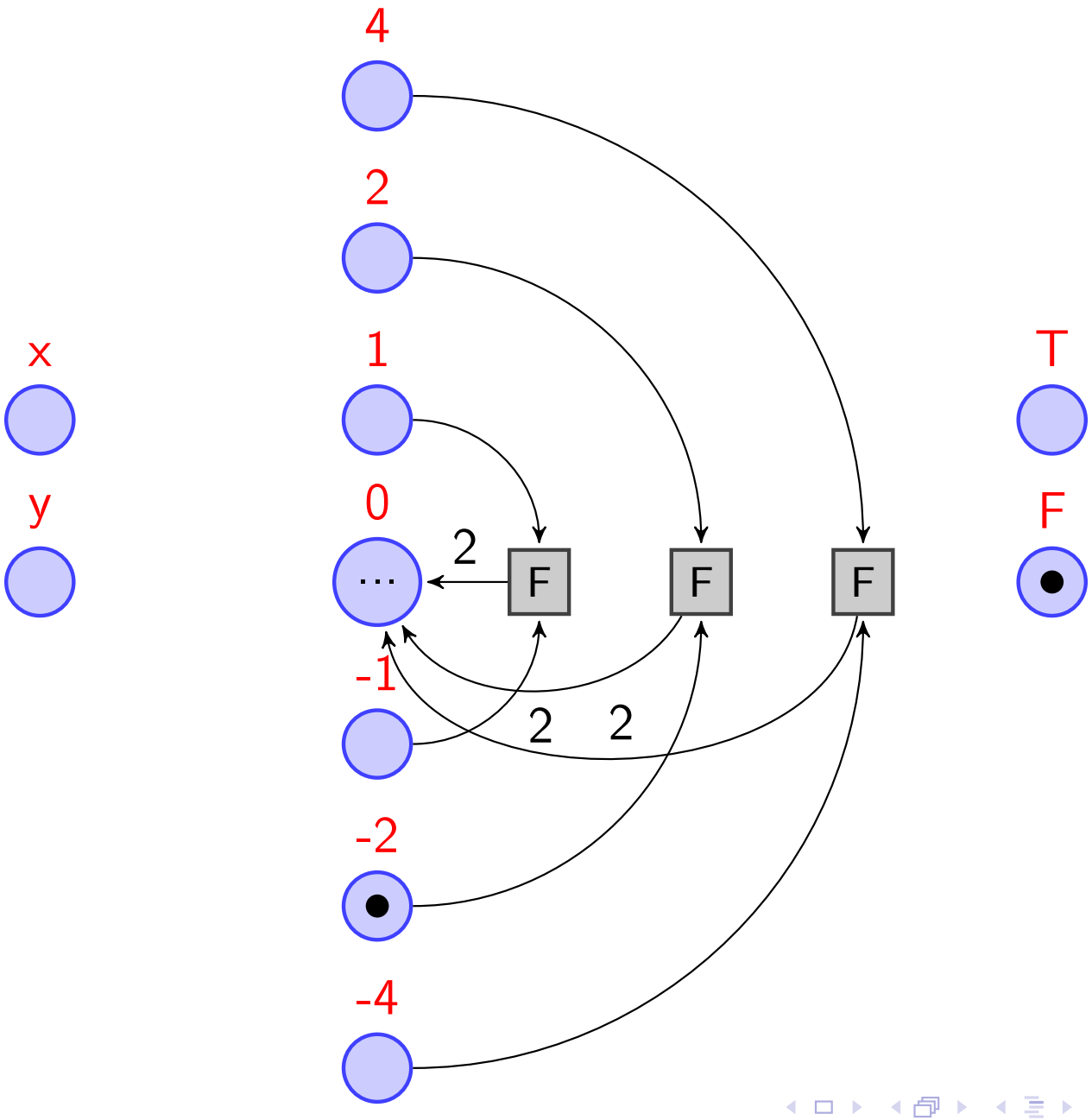

4
2
x
1
T
y
0
F
...
3n + 1 reservoir leaders
-1
-2
-4

$3x - 2y < 2$ **input transitions:** $x_i, 0, 0, \ldots \to rep(a_i)$

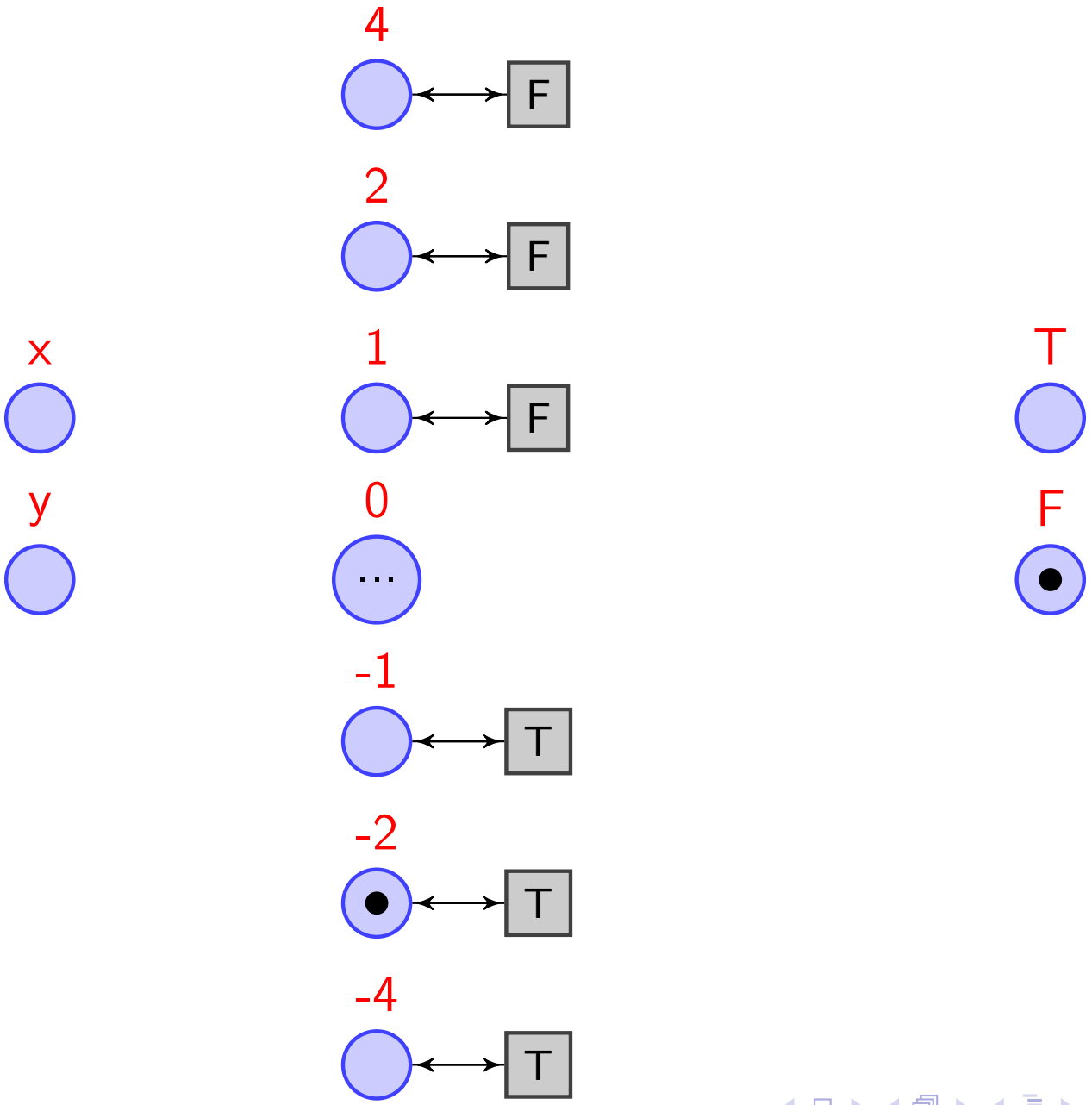



4
2
2
2
x
1
2
2
T
y
0
...
F
-1
2
2
-2
2
2
-4

$3x - 2y < 2$ **cancel:** $2^i, -2^i, X \to 0, 0, F$ with $X \in \{T, F\}$



$3x - 2y < 2$ **signal:** $2^i, T \to 2^i, F \quad -2^i, F \to -2^i, T$



Threshold: Correctness

Assumption: We can always put n agents back in state 0.

Correctness:

Threshold: Correctness

Assumption: We can always put n agents back in state 0.

Correctness:

- total value of all agents is invariant

Threshold: Correctness

Assumption: We can always put n agents back in state 0.

Correctness:

- total value of all agents is invariant
- all inputs will be emptied

Threshold: Correctness

Assumption: We can always put n agents back in state 0.

Correctness:

- total value of all agents is invariant
- all inputs will be emptied
- finally only non-negative or non-positive agents

Threshold: Correctness

Assumption: We can always put n agents back in state 0.

Correctness:

- total value of all agents is invariant
- all inputs will be emptied
- finally only non-negative or non-positive agents
  - $\sum_{i=1}^{k} a_i x_i - c \neq 0$: correct output because of signal

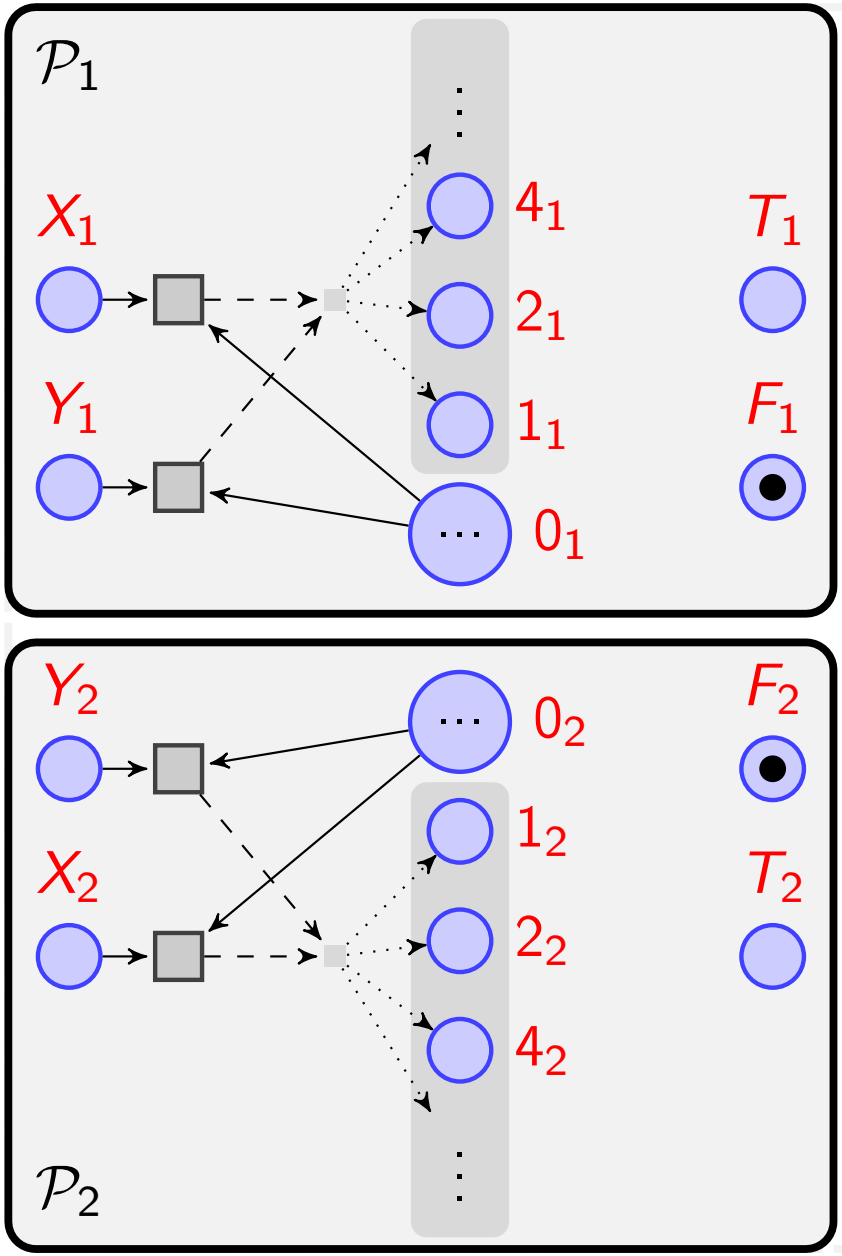
Threshold: Correctness

Assumption: We can always put n agents back in state 0.

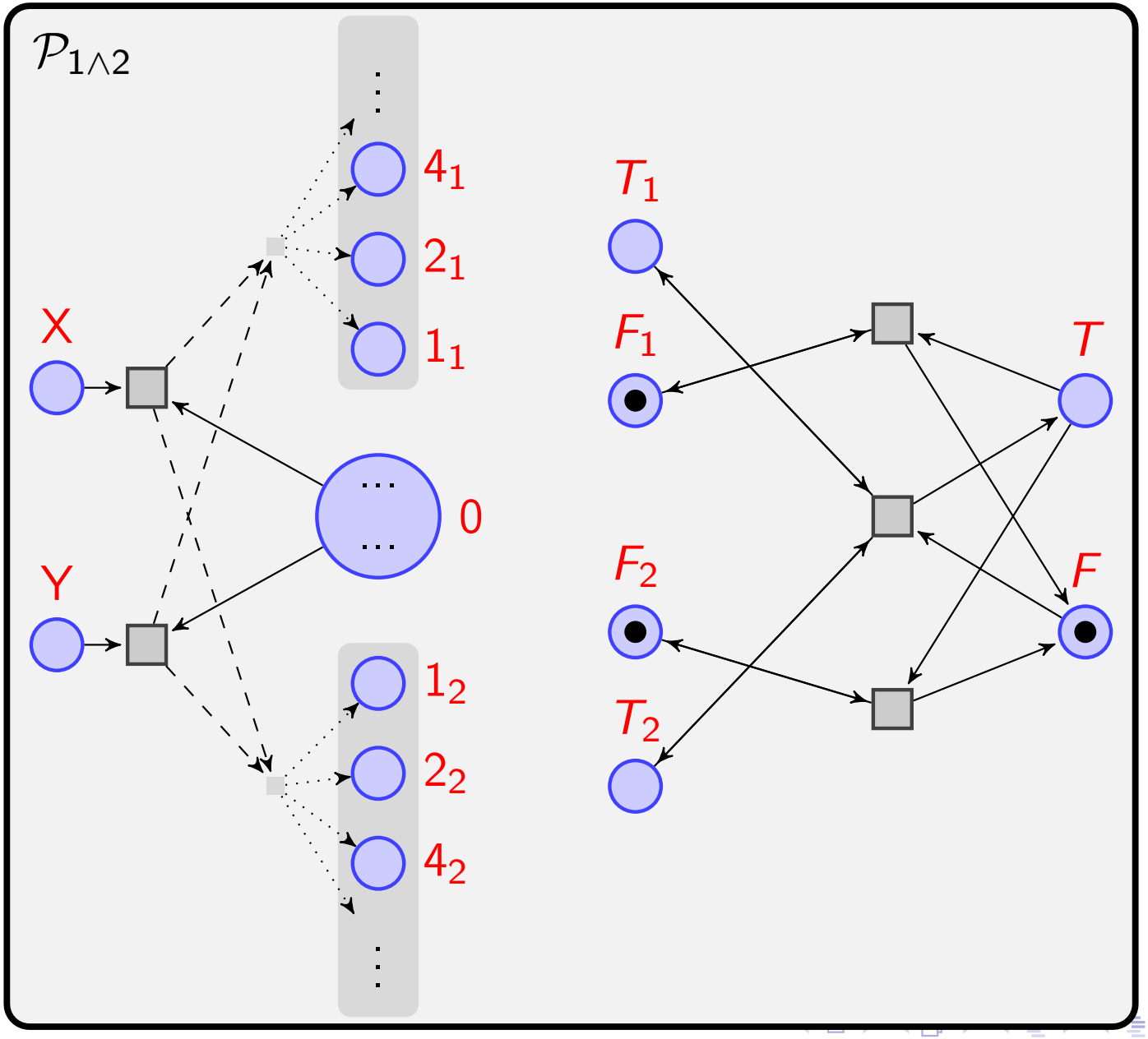
Correctness:

- total value of all agents is invariant
- all inputs will be emptied
- finally only non-negative or non-positive agents
  - $\sum_{i=1}^{k} a_i x_i - c \neq 0$: correct output because of signal
  - $\sum_{i=1}^{k} a_i x_i - c = 0$: output is False
    because of initial output or last cancel

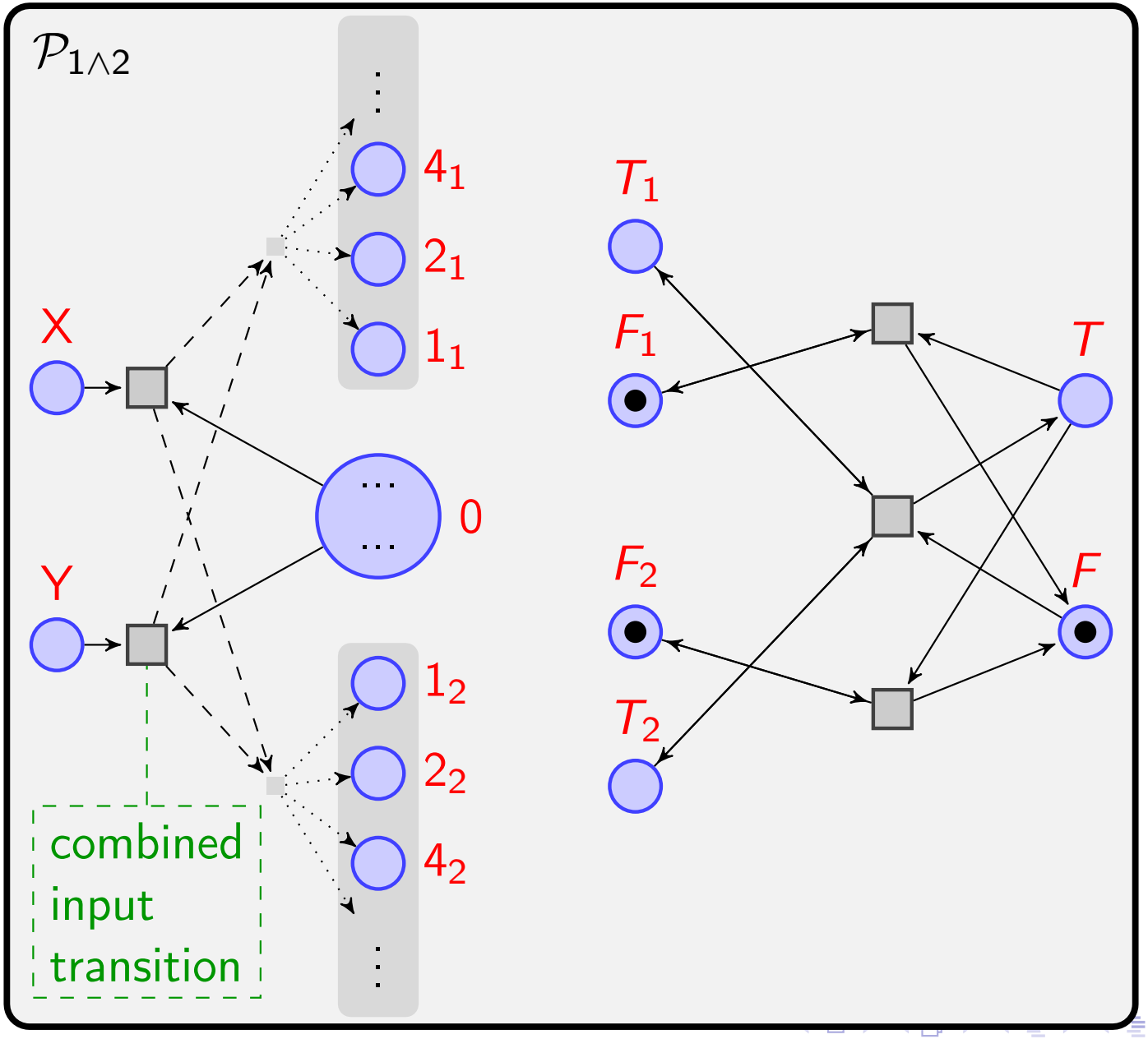
Boolean Combination



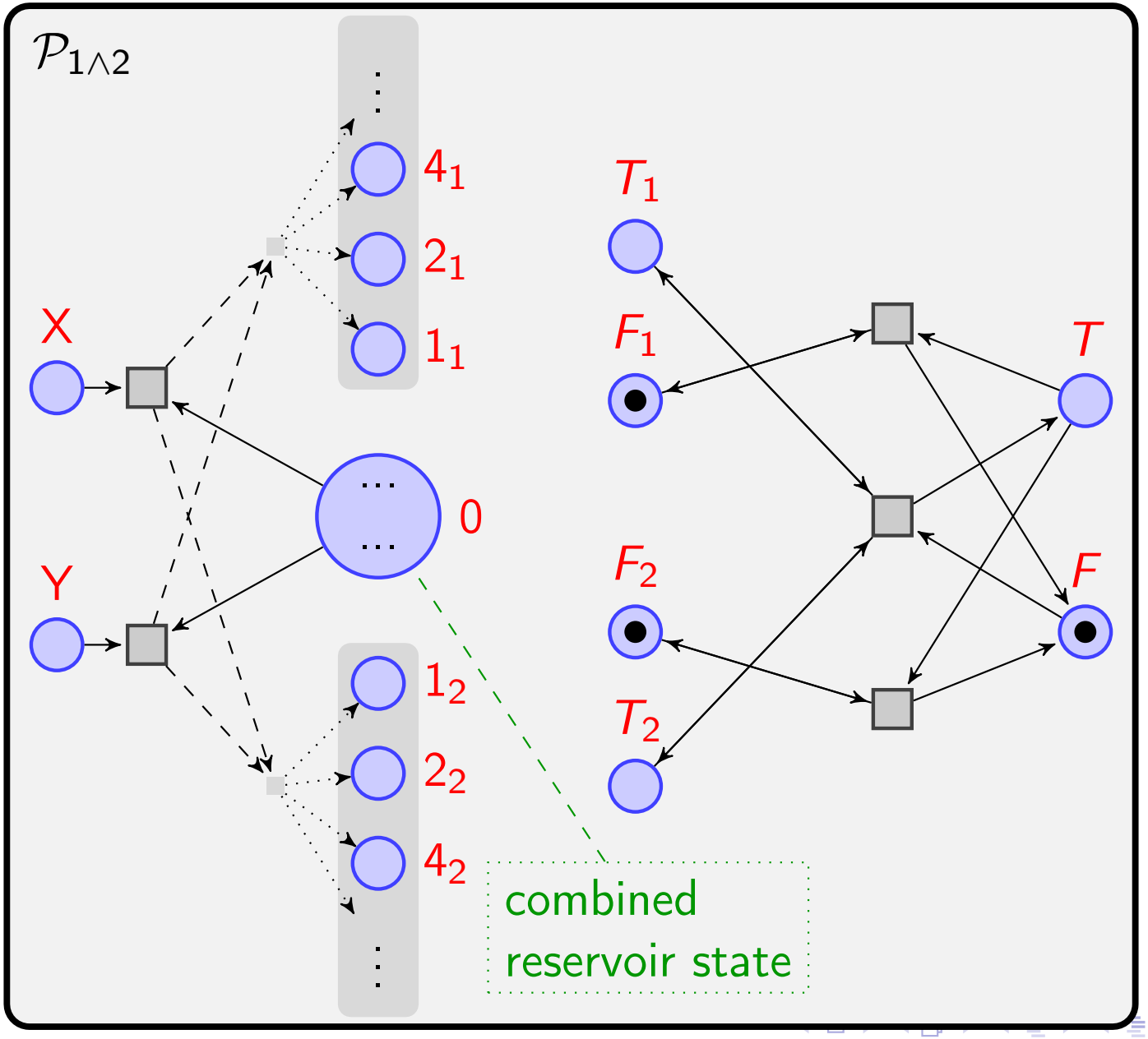
Boolean Combination



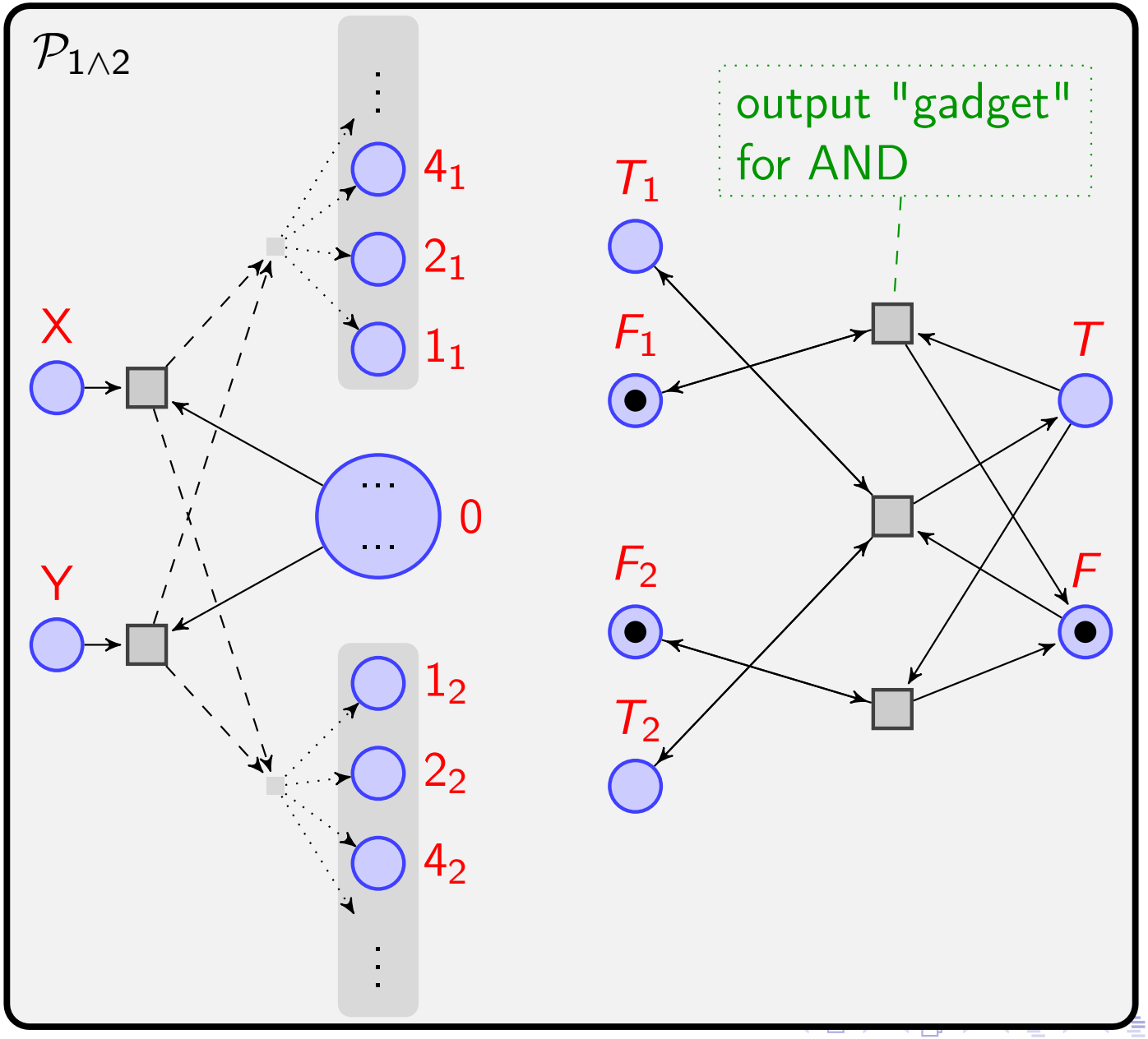
Boolean Combination



Boolean Combination



Boolean Combination



Boolean Combination: Correctness

Assumption: We can always put $n^* \stackrel{\text{def}}{=} n_1 + n_2$ agents back in state 0.

Correctness:

Boolean Combination: Correctness

Assumption: We can always put $n^* \stackrel{\text{def}}{=} n_1 + n_2$ agents back in state 0.

Correctness:

- all inputs will be emptied

Boolean Combination: Correctness

Assumption: We can always put $n^* \stackrel{\text{def}}{=} n_1 + n_2$ agents back in state 0.

Correctness:

- all inputs will be emptied
- finally: output of $\mathcal{P}_1$ and $\mathcal{P}_2$ is correct
  $\Rightarrow$ overall output is correct

Boolean Combination: Assumption

Assumption: We can always put $n^* \stackrel{\text{def}}{=} n_1 + n_2$ agents back in state 0.

Boolean Combination: Assumption

Assumption: We can always put $n^* \stackrel{\text{def}}{=} n_1 + n_2$ agents back in state 0.

Boolean Combination of m base protocols

Boolean Combination: Assumption

Assumption: We can always put $n^* \stackrel{\text{def}}{=} n_1 + n_2$ agents back in state 0.

Boolean Combination of m base protocols
$\Rightarrow$ at most $2m$ "extreme states"

Boolean Combination: Assumption

Assumption: We can always put $n^* \stackrel{\text{def}}{=} n_1 + n_2$ agents back in state 0.

Boolean Combination of m base protocols
$\Rightarrow$ at most $2m$ "extreme states"
$\Rightarrow$ make sure: number of agents in a "extreme state" can only increase every $2m$ input transitions

Boolean Combination: Assumption

Assumption: We can always put $n^* \stackrel{\text{def}}{=} n_1 + n_2$ agents back in state 0.

Boolean Combination of m base protocols
$\Rightarrow$ at most $2m$ "extreme states"
$\Rightarrow$ make sure: number of agents in a "extreme state" can only increase every $2m$ input transitions
$\Rightarrow$ scale base constructions

$n_i \stackrel{\text{def}}{=} \lceil \log 2m \rceil + \text{size}(b_{i,max})$

Result

Theorem

Let φ be a quantifier-free Presburger formula. Let $|\varphi|$ be the length of φ when encoded in binary. There exists a 2-way population protocol of size $O(|\varphi|^2 \cdot \log |\varphi|)$ with $O(|\varphi| \cdot \log |\varphi|)$ leaders that computes φ.