Optimization of Model Checking by Large Block Encoding

Thomas Mertens

24.09.2014

Outline

1. Motivation
2. Background
   - Program and Control-Flow Automaton
   - Model Checker
3. Large-Block Encoding
   - Summarization
   - Post-processing
4. Evaluation
5. Conclusion

Outline

1. Motivation
2. Background
   - Program and Control-Flow Automaton
   - Model Checker
3. Large-Block Encoding
   - Summarization
   - Post-processing
4. Evaluation
5. Conclusion

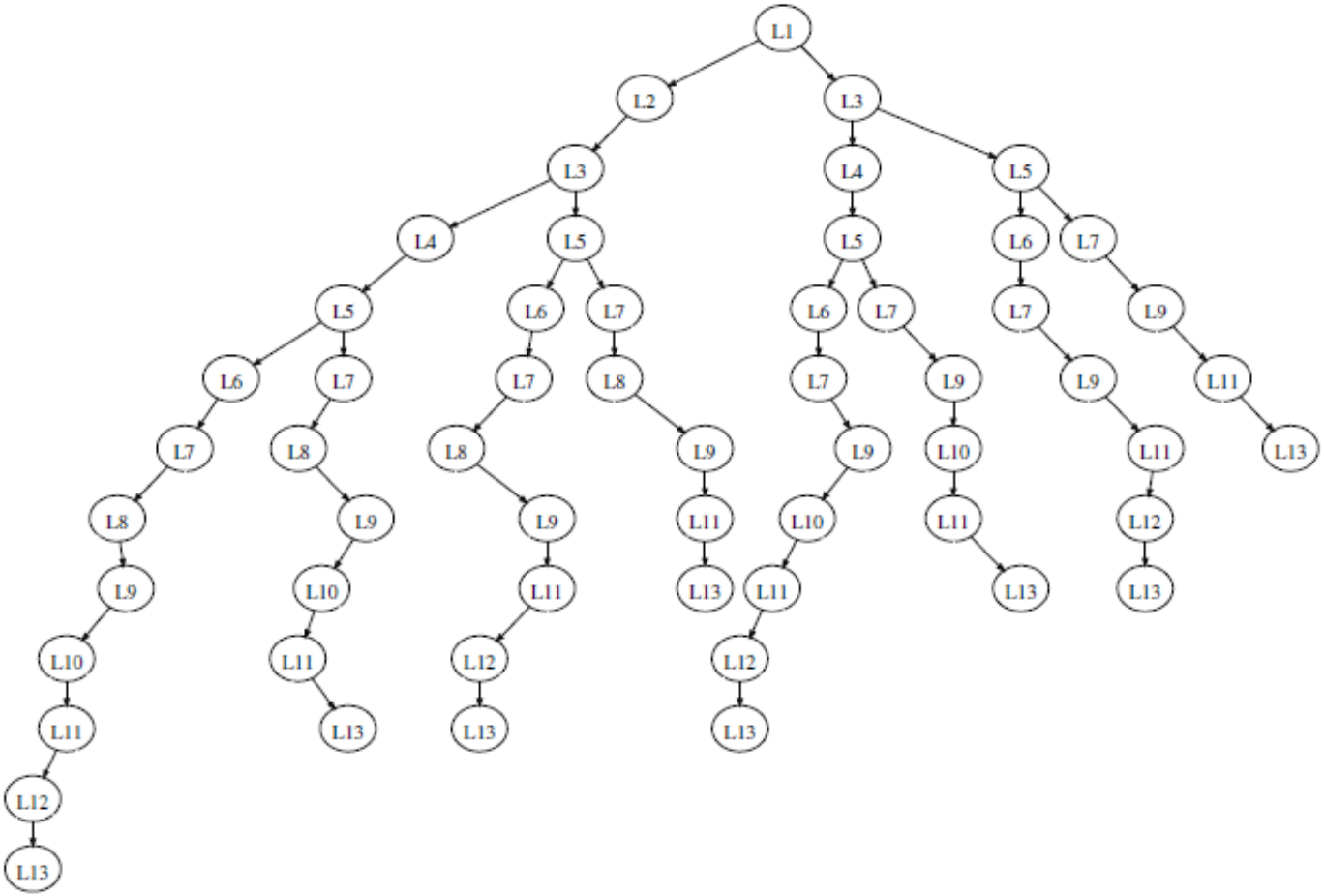
Motivation

Why do we do Model Checking?

Motivation

Why do we do Model Checking?

- Reduce runtime and memory of model checking in particular CEGAR through minimizing the abstract reachability tree (ART)



Outline

1. Motivation
2. Background
   - Program and Control-Flow Automaton
   - Model Checker
3. Large-Block Encoding
   - Summarization
   - Post-processing
4. Evaluation
5. Conclusion

Program and Control-Flow Automaton

Definition (Program)

A sequential program P with the operations Ops is defined as $P = op_0, op_1, ..., op_n$ with $op_i \in Ops$.

Program and Control-Flow Automaton

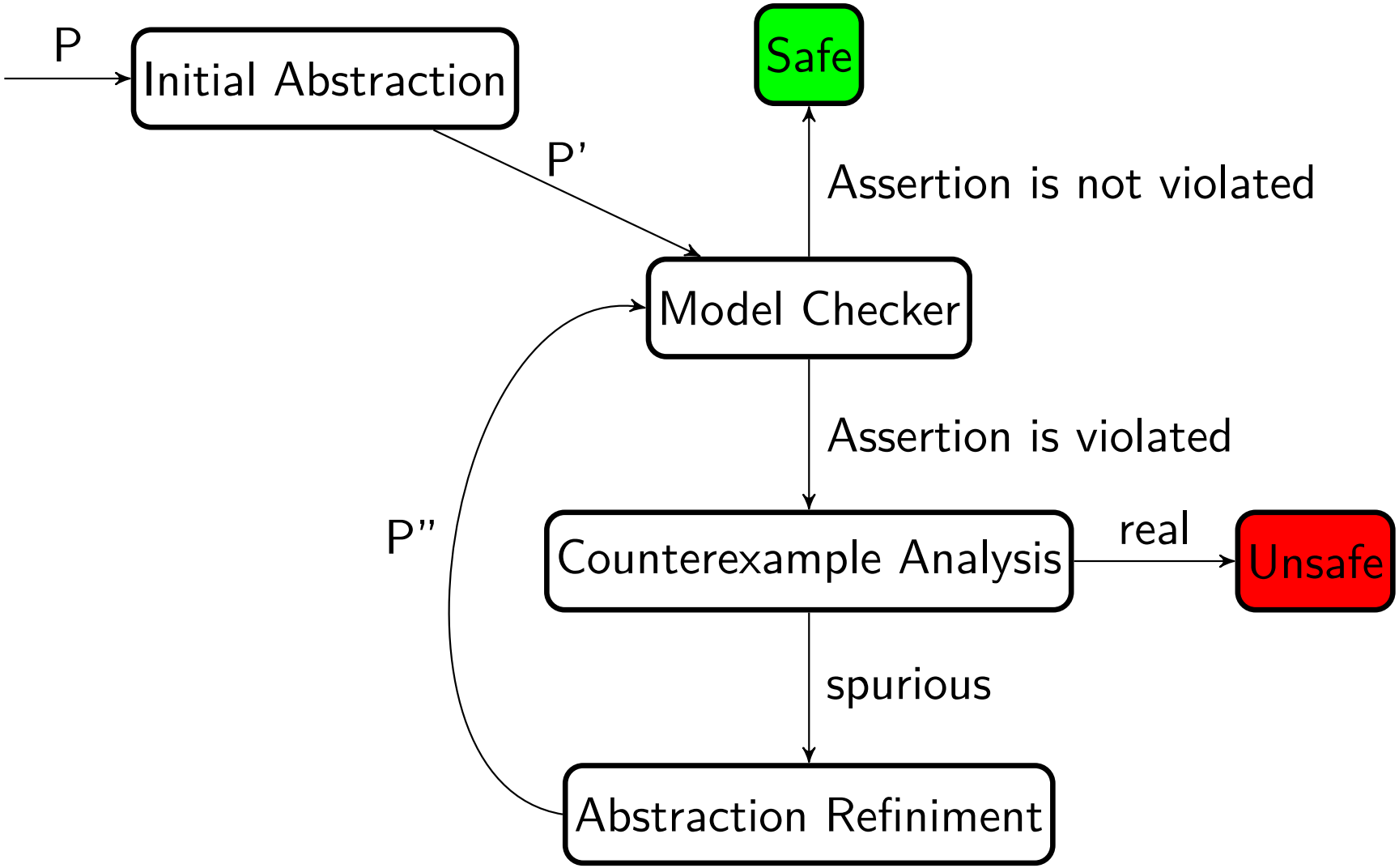
Definition (Program)

A sequential program P with the operations Ops is defined as $P = op_0, op_1, ..., op_n$ with $op_i \in Ops$.

Definition (Control-Flow Automaton)

A Control Flow Automaton is defined as $A = (L, G, l_0, L_E)$, where L is a finite set of locations, $G \subseteq L \times Ops \times L$, $l_0 \in L$ is a unique initial location and $L_E \subseteq L$ is a set of error locations.

CEGAR


P
Initial Abstraction
P'
Safe
Assertion is not violated
Model Checker
Assertion is violated
P''
Counterexample Analysis
real
Unsafe
spurious
Abstraction Refinement

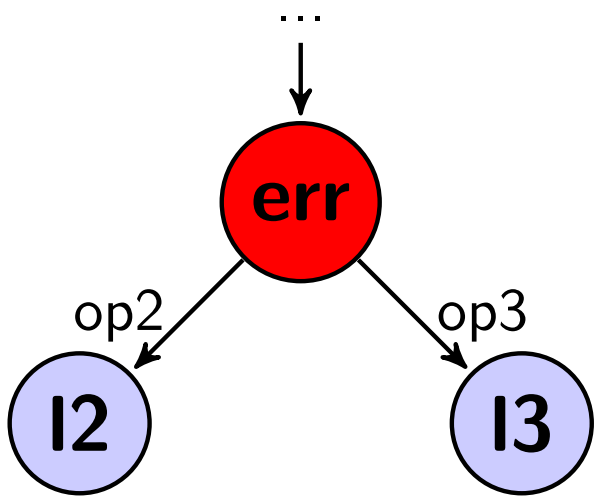
Outline

1. Motivation
2. Background
   - Program and Control-Flow Automaton
   - Model Checker
3. Large-Block Encoding
   - Summarization
   - Post-processing
4. Evaluation
5. Conclusion

Error sink rule (Rule 1)

Definition

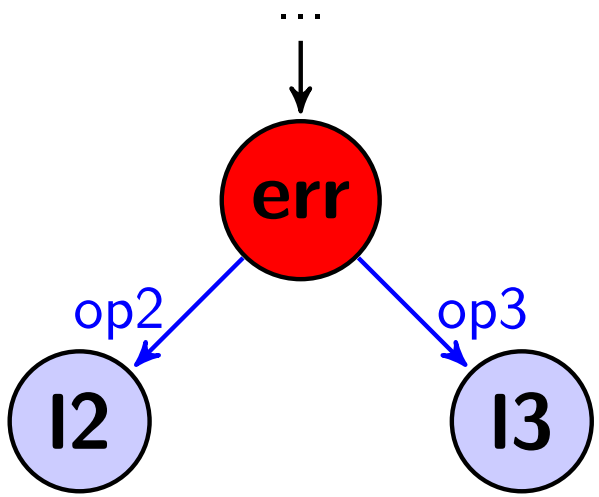
Let $l_e \in L_E$ with $outdegree(l_e) \geq 1$ and $G_{out} = \{(l_e, op_x, l_x) \in G\}$.
The error sink rule removes all edges $g \in G_{out}$.



Error sink rule (Rule 1)

Definition

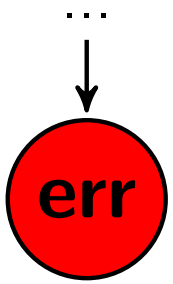
Let $l_e \in L_E$ with $outdegree(l_e) \geq 1$ and $G_{out} = \{(l_e, op_x, l_x) \in G\}$.
The error sink rule removes all edges $g \in G_{out}$.



Error sink rule (Rule 1)

Definition

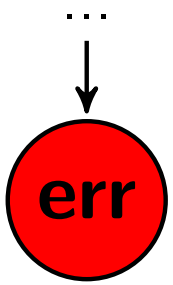
Let $l_e \in L_E$ with $outdegree(l_e) \geq 1$ and $G_{out} = \{(l_e, op_x, l_x) \in G\}$.
The error sink rule removes all edges $g \in G_{out}$.



Error sink rule (Rule 1)

Definition

Let $l_e \in L_E$ with $outdegree(l_e) \geq 1$ and $G_{out} = \{(l_e, op_x, l_x) \in G\}$.
The error sink rule removes all edges $g \in G_{out}$.



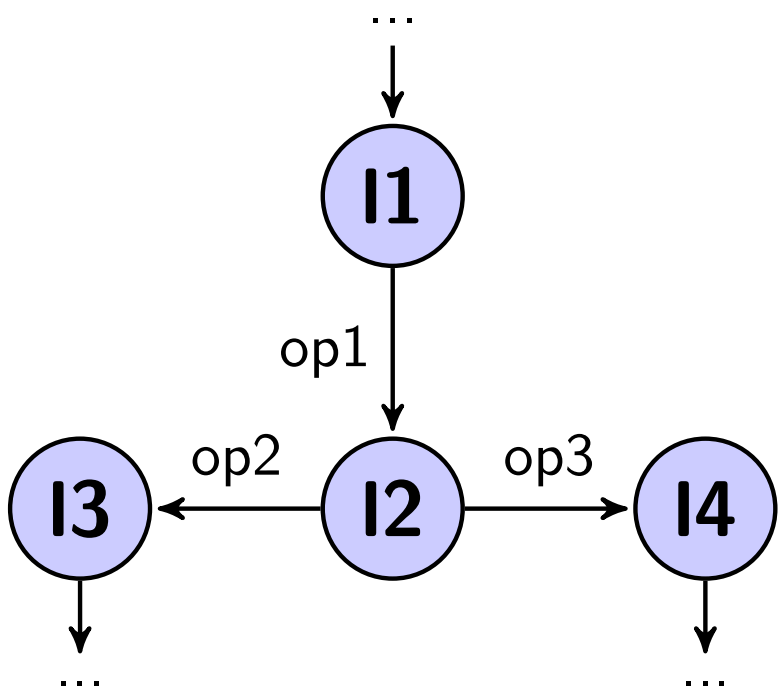
Complexity

$\mathcal{O}(k)$, where k is the outdegree of l_e

Sequence rule (Rule 2)

Definition

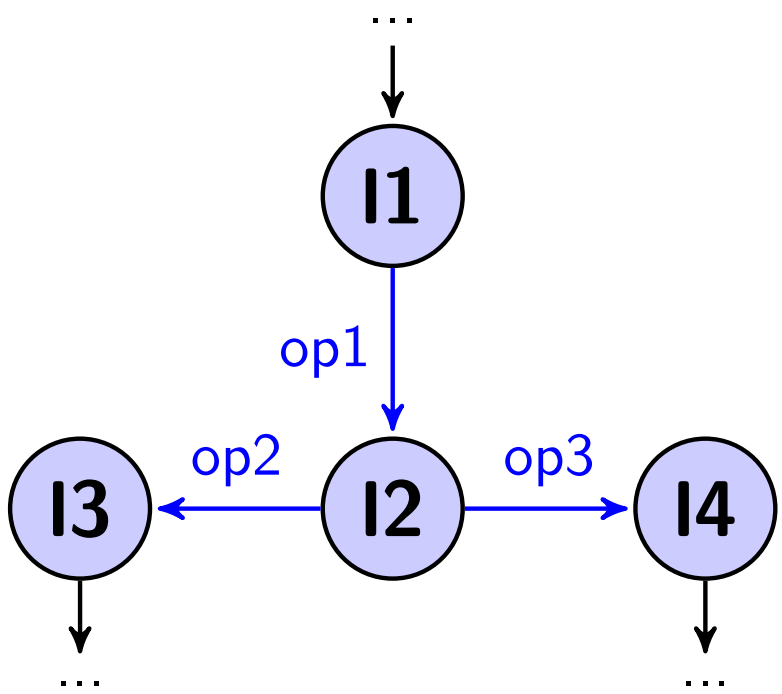
Let $l_x \in L \setminus L_E$ with $indegree(l_x) = 1$, $outdegree(l_x) \geq 1$,$l_x \notin succ(l_x)$ and the edge $(l_y, op_y, l_x) \in G$,
$G_{out} = \{(l_x, op_x, l_z) \in G\}$.
The sequence rule adds new edges with $(l_y, op_y; op_x, l_z)\ \forall g \in G_{out}$ to the CFA and removes all edges $g \in \{(l_y, op_y, l_x) \in G\} \cup G_{out}$.



Sequence rule (Rule 2)

Definition

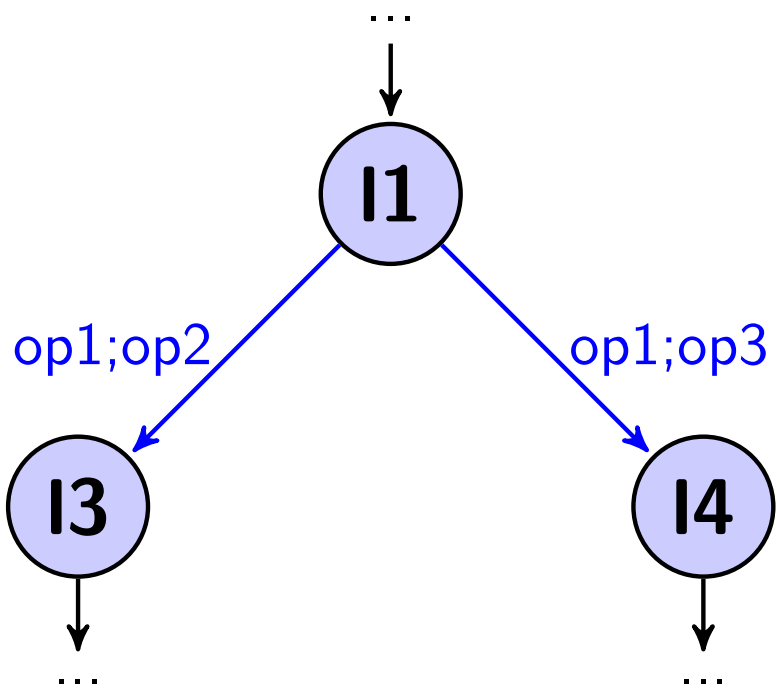
Let $l_x \in L \setminus L_E$ with $indegree(l_x) = 1$, $outdegree(l_x) \geq 1$,$l_x \notin succ(l_x)$ and the edge $(l_y, op_y, l_x) \in G$,
$G_{out} = \{(l_x, op_x, l_z) \in G\}$.
The sequence rule adds new edges with $(l_y, op_y; op_x, l_z)\ \forall g \in G_{out}$ to the CFA and removes all edges $g \in \{(l_y, op_y, l_x) \in G\} \cup G_{out}$.



Sequence rule (Rule 2)

Definition

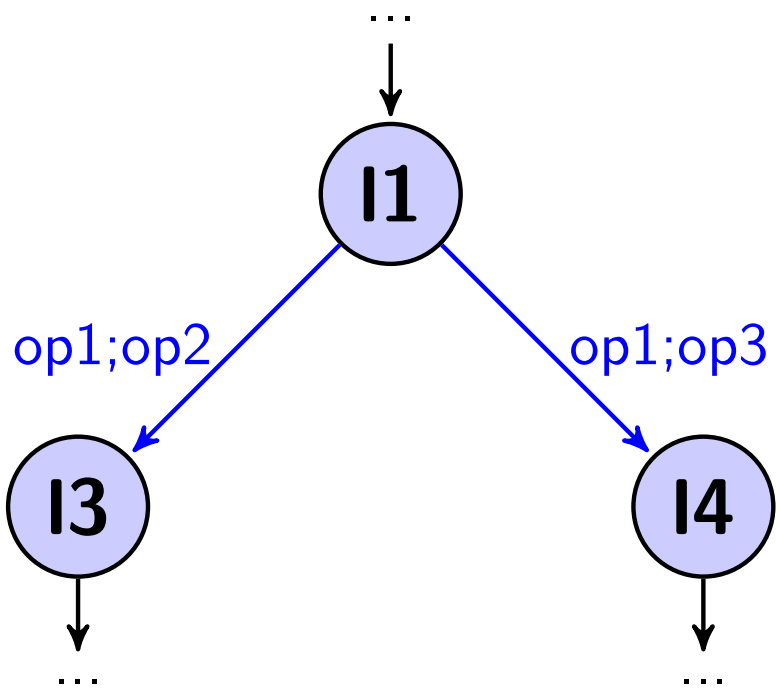
Let $l_x \in L \setminus L_E$ with $indegree(l_x) = 1$, $outdegree(l_x) \geq 1$,$l_x \notin succ(l_x)$ and the edge $(l_y, op_y, l_x) \in G$,
$G_{out} = \{(l_x, op_x, l_z) \in G\}$.
The sequence rule adds new edges with $(l_y, op_y; op_x, l_z)\ \forall g \in G_{out}$ to the CFA and removes all edges $g \in \{(l_y, op_y, l_x) \in G\} \cup G_{out}$.



Sequence rule (Rule 2)

Definition

Let $l_x \in L \setminus L_E$ with $indegree(l_x) = 1$, $outdegree(l_x) \geq 1$,$l_x \notin succ(l_x)$ and the edge $(l_y, op_y, l_x) \in G$,
$G_{out} = \{(l_x, op_x, l_z) \in G\}$.
The sequence rule adds new edges with $(l_y, op_y; op_x, l_z)\ \forall g \in G_{out}$ to the CFA and removes all edges $g \in \{(l_y, op_y, l_x) \in G\} \cup G_{out}$.



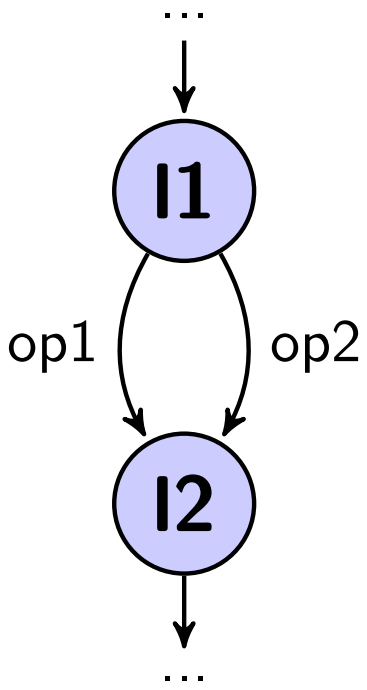
Complexity

$\mathcal{O}(k)$, where k is the outdegree of l_y

Choice rule (Rule 3)

Definition

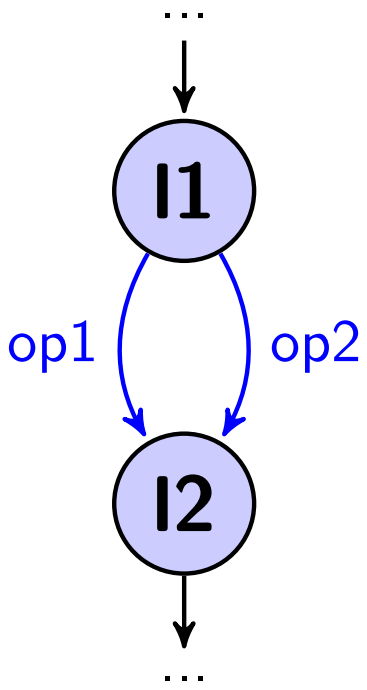
Let $l_x \in L \setminus L_E$ with $l_y \in succ(l_x)$, $G_{con} = connecting\_edges(l_x, l_y)$ and $\mid G_{con} \mid = n > 1$.
The choice rule adds a new edge (l_x, op, l_y) with $op = op_1 \mid\mid op_2 \mid\mid ... \mid\mid op_n$ to the CFA and removes all edges $g \in G_{con}$.



Choice rule (Rule 3)

Definition

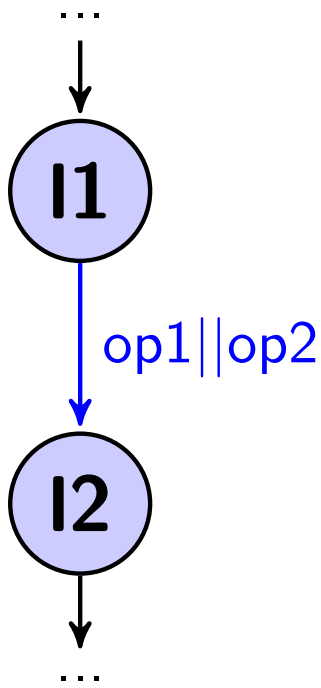
Let $l_x \in L \setminus L_E$ with $l_y \in succ(l_x)$, $G_{con} = connecting\_edges(l_x, l_y)$ and $\mid G_{con} \mid = n > 1$.
The choice rule adds a new edge (l_x, op, l_y) with $op = op_1 \mid\mid op_2 \mid\mid ... \mid\mid op_n$ to the CFA and removes all edges $g \in G_{con}$.



Choice rule (Rule 3)

Definition

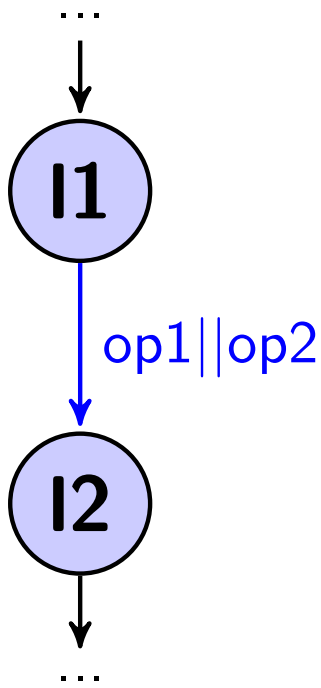
Let $l_x \in L \setminus L_E$ with $l_y \in succ(l_x)$, $G_{con} = connecting\_edges(l_x, l_y)$ and $\mid G_{con} \mid = n > 1$.
The choice rule adds a new edge (l_x, op, l_y) with $op = op_1 \mid\mid op_2 \mid\mid ... \mid\mid op_n$ to the CFA and removes all edges $g \in G_{con}$.



Choice rule (Rule 3)

Definition

Let $l_x \in L \setminus L_E$ with $l_y \in succ(l_x)$, $G_{con} = connecting\_edges(l_x, l_y)$ and $\mid G_{con} \mid = n > 1$.
The choice rule adds a new edge (l_x, op, l_y) with $op = op_1 \mid\mid op_2 \mid\mid ... \mid\mid op_n$ to the CFA and removes all edges $g \in G_{con}$.



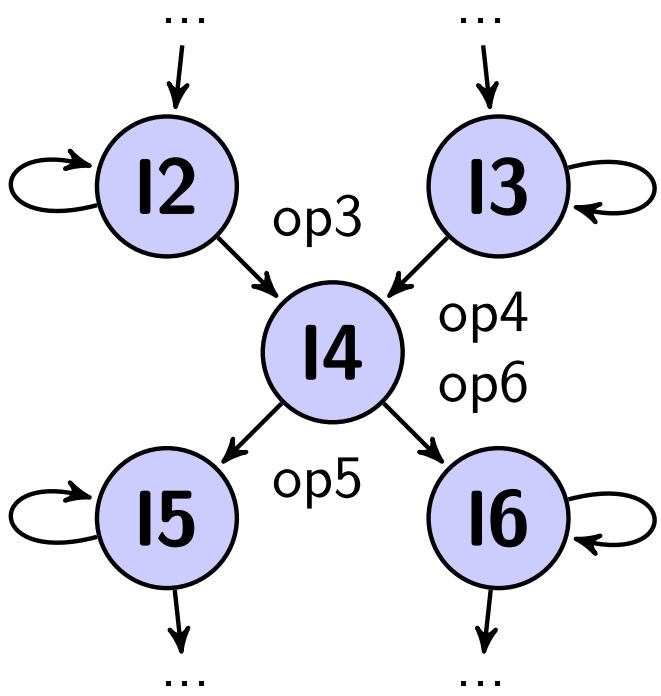
Complexity

$\mathcal{O}(k)$, where k is the outdegree of l_x

Advanced sequence rule (Rule 4)

Definition

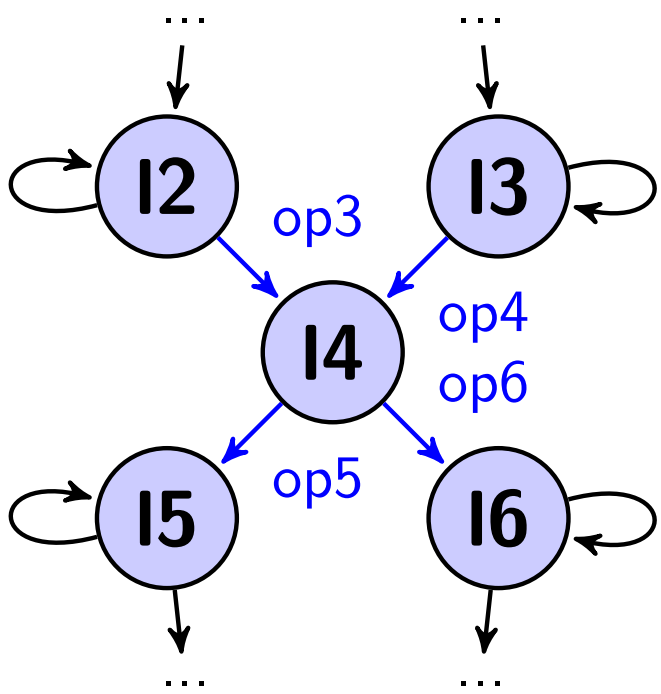
Let $l_x \in L \setminus L_E$ with $indegree(l_x) > 1$, $outdegree(l_x) > 1$, $l_x \notin succ(l_x)$ and $G_{in} = \{(l_y, op_y, l_x) \in G\}$, $G_{out} = \{(l_x, op_x, l_z) \in G\}$.
The sequence rule adds new edges with $(l_y, op_y; op_x, l_z)$ for all possible connections of edges in G_{in} and G_{out} to the CFA and removes all edges $g \in G_{in} \cup G_{out}$.



Advanced sequence rule (Rule 4)

Definition

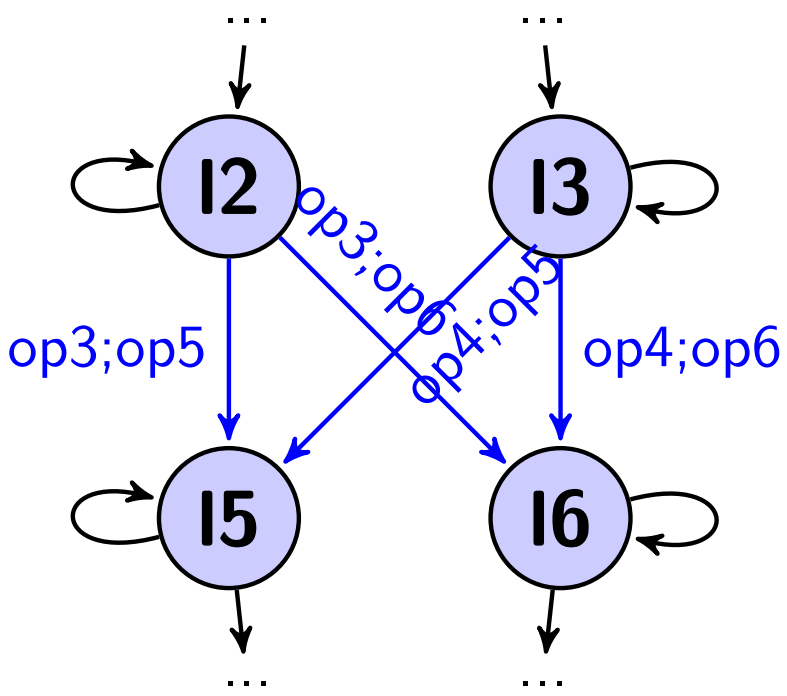
Let $l_x \in L \setminus L_E$ with $indegree(l_x) > 1$, $outdegree(l_x) > 1$, $l_x \notin succ(l_x)$ and $G_{in} = \{(l_y, op_y, l_x) \in G\}$, $G_{out} = \{(l_x, op_x, l_z) \in G\}$.
The sequence rule adds new edges with $(l_y, op_y; op_x, l_z)$ for all possible connections of edges in G_{in} and G_{out} to the CFA and removes all edges $g \in G_{in} \cup G_{out}$.



Advanced sequence rule (Rule 4)

Definition

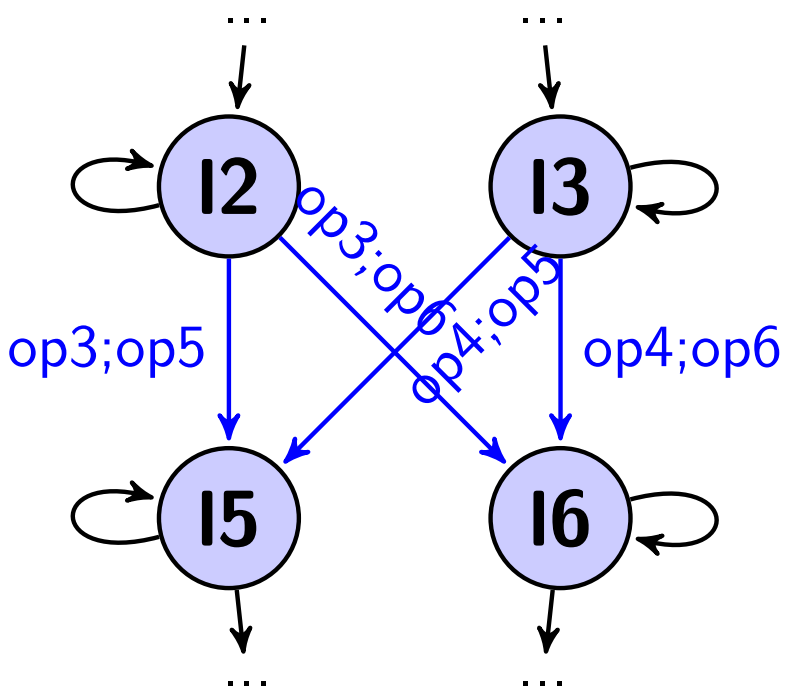
Let $l_x \in L \setminus L_E$ with $indegree(l_x) > 1$, $outdegree(l_x) > 1$, $l_x \notin succ(l_x)$ and $G_{in} = \{(l_y, op_y, l_x) \in G\}$, $G_{out} = \{(l_x, op_x, l_z) \in G\}$.
The sequence rule adds new edges with $(l_y, op_y; op_x, l_z)$ for all possible connections of edges in G_{in} and G_{out} to the CFA and removes all edges $g \in G_{in} \cup G_{out}$.



Advanced sequence rule (Rule 4)

Definition

Let $l_x \in L \setminus L_E$ with $indegree(l_x) > 1$, $outdegree(l_x) > 1$, $l_x \notin succ(l_x)$ and $G_{in} = \{(l_y, op_y, l_x) \in G\}$, $G_{out} = \{(l_x, op_x, l_z) \in G\}$.
The sequence rule adds new edges with $(l_y, op_y; op_x, l_z)$ for all possible connections of edges in G_{in} and G_{out} to the CFA and removes all edges $g \in G_{in} \cup G_{out}$.



Complexity

$\mathcal{O}(k \cdot m)$, where k is the outdegree of $l_x \in L$ and m is the indegree of $l_x \in L$

Complexity

- Rule 1: $\mathcal{O}(k)$, where k is the outdegree of l_e
- Rule 2: $\mathcal{O}(k)$, where k is the outdegree of $l_y \in L$
- Rule 3: $\mathcal{O}(k)$, where k is the outdegree of l_y
- Rule 4: $\mathcal{O}(k \cdot m)$, where k is the outdegree of $l_x \in L$ and m is the indegree of $l_x \in L$

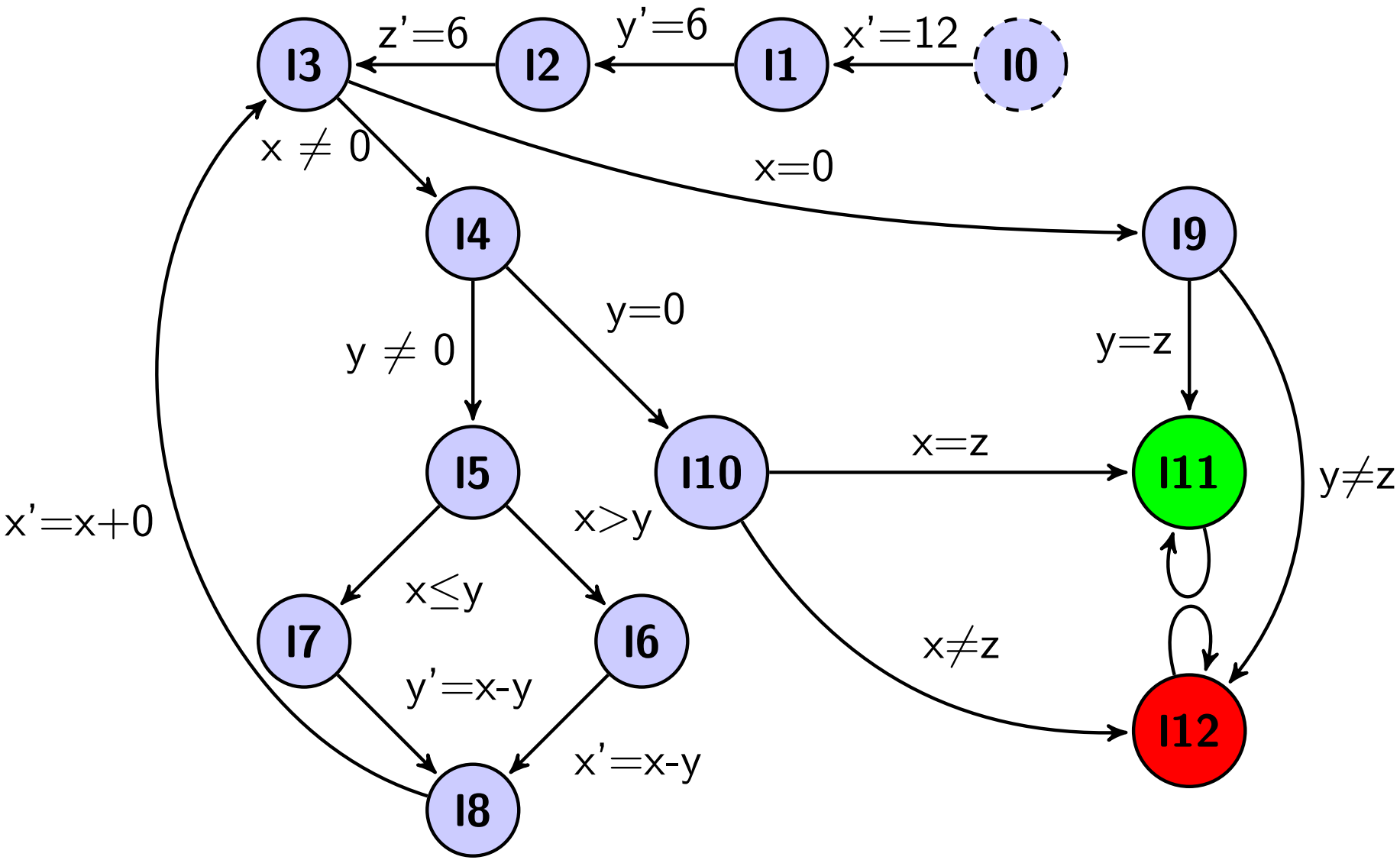
Complexity

- Rule 1: $\mathcal{O}(k)$, where k is the outdegree of l_e
- Rule 2: $\mathcal{O}(k)$, where k is the outdegree of $l_y \in L$
- Rule 3: $\mathcal{O}(k)$, where k is the outdegree of l_y
- Rule 4: $\mathcal{O}(k \cdot m)$, where k is the outdegree of $l_x \in L$ and m is the indegree of $l_x \in L$

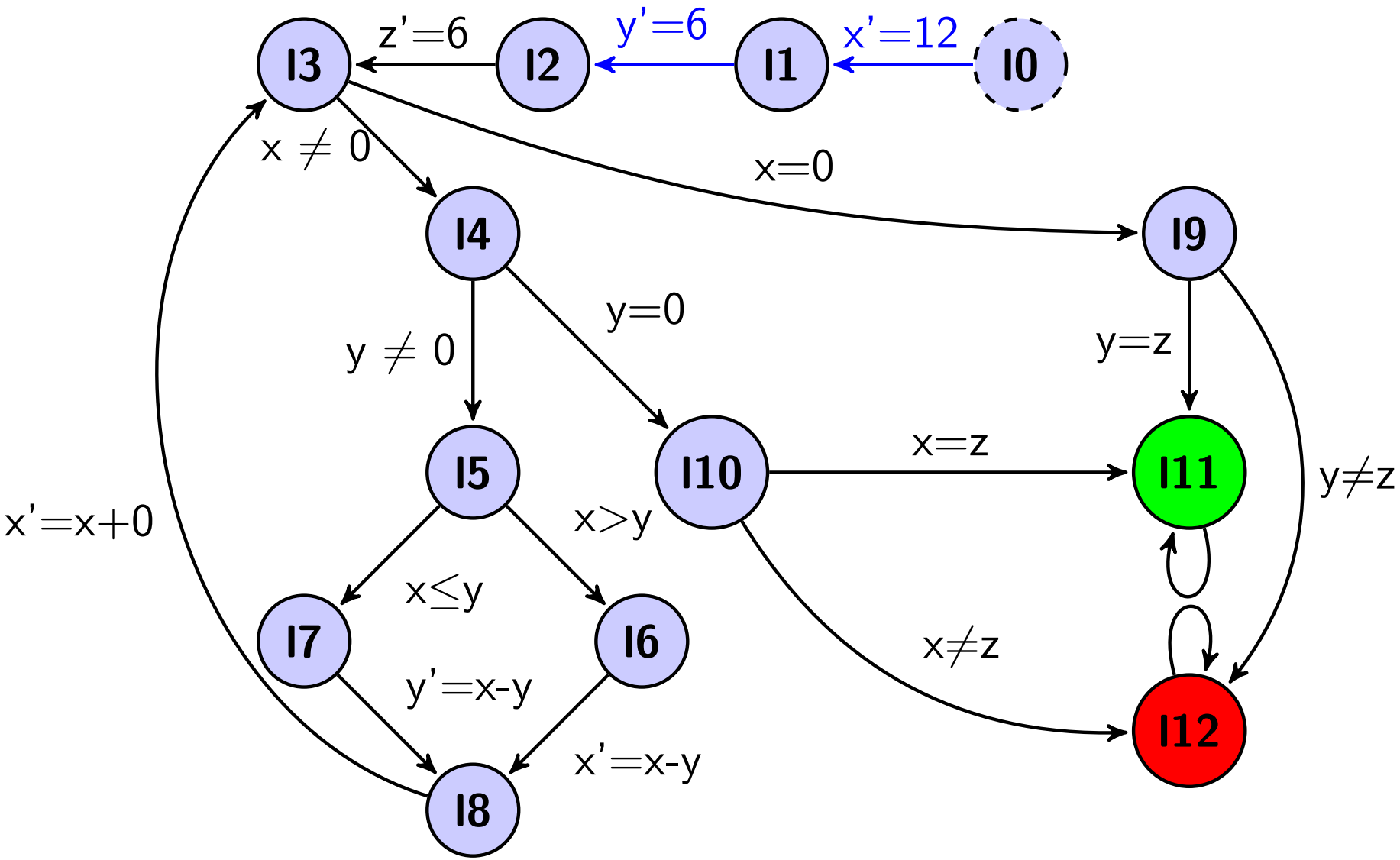
Complexity in total

$\mathcal{O}(\mid G \mid \cdot \mid L \mid \cdot k \cdot m)$

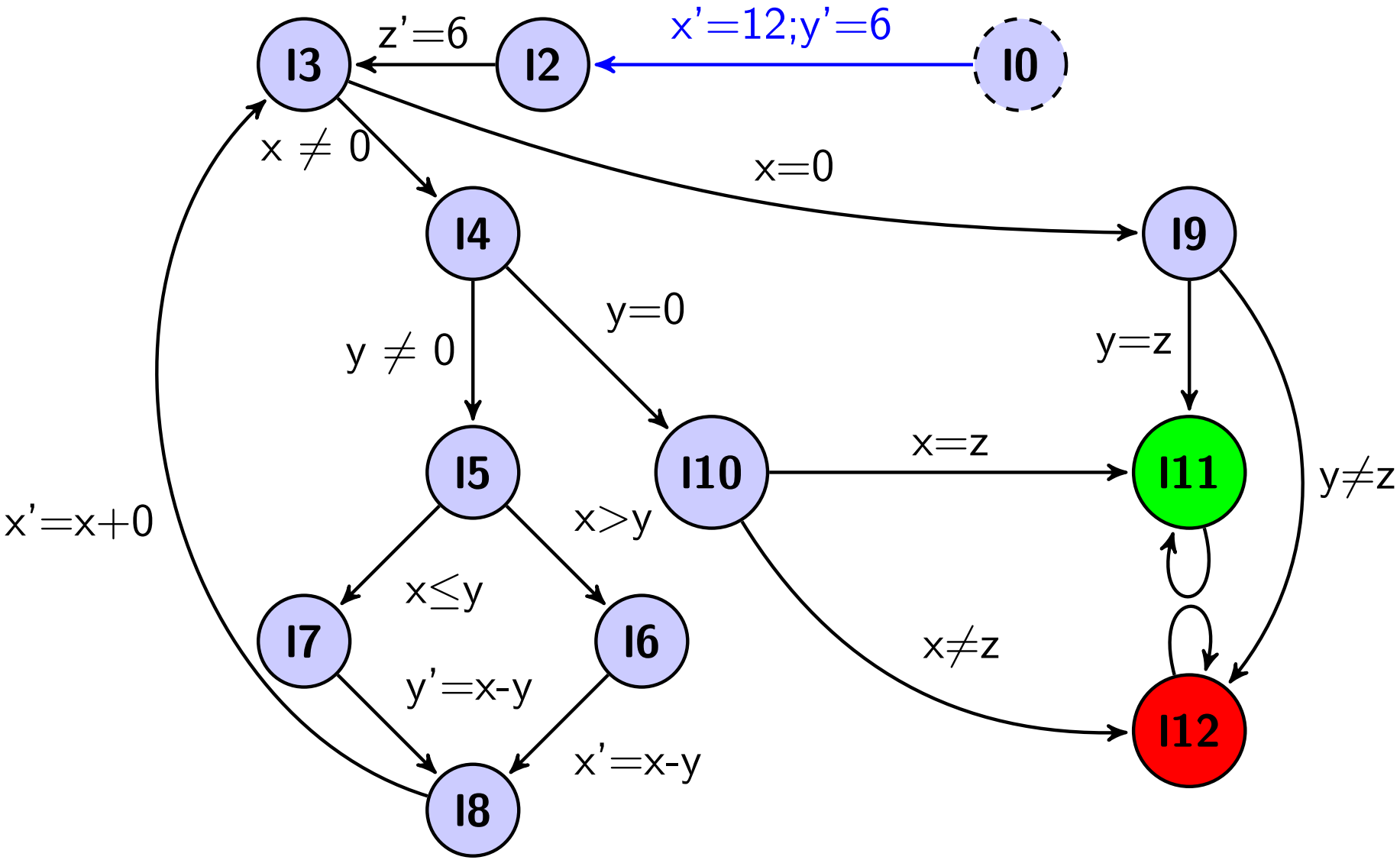
Example


l0
l1
l2
l3
l4
l5
l6
l7
l8
l9
l10
l11
l12
x'=12
y'=6
z'=6
x ≠ 0
x=0
y ≠ 0
y=0
y=z
x=z
y≠z
x>y
x≤y
x≠z
y'=x-y
x'=x-y
x'=x+0

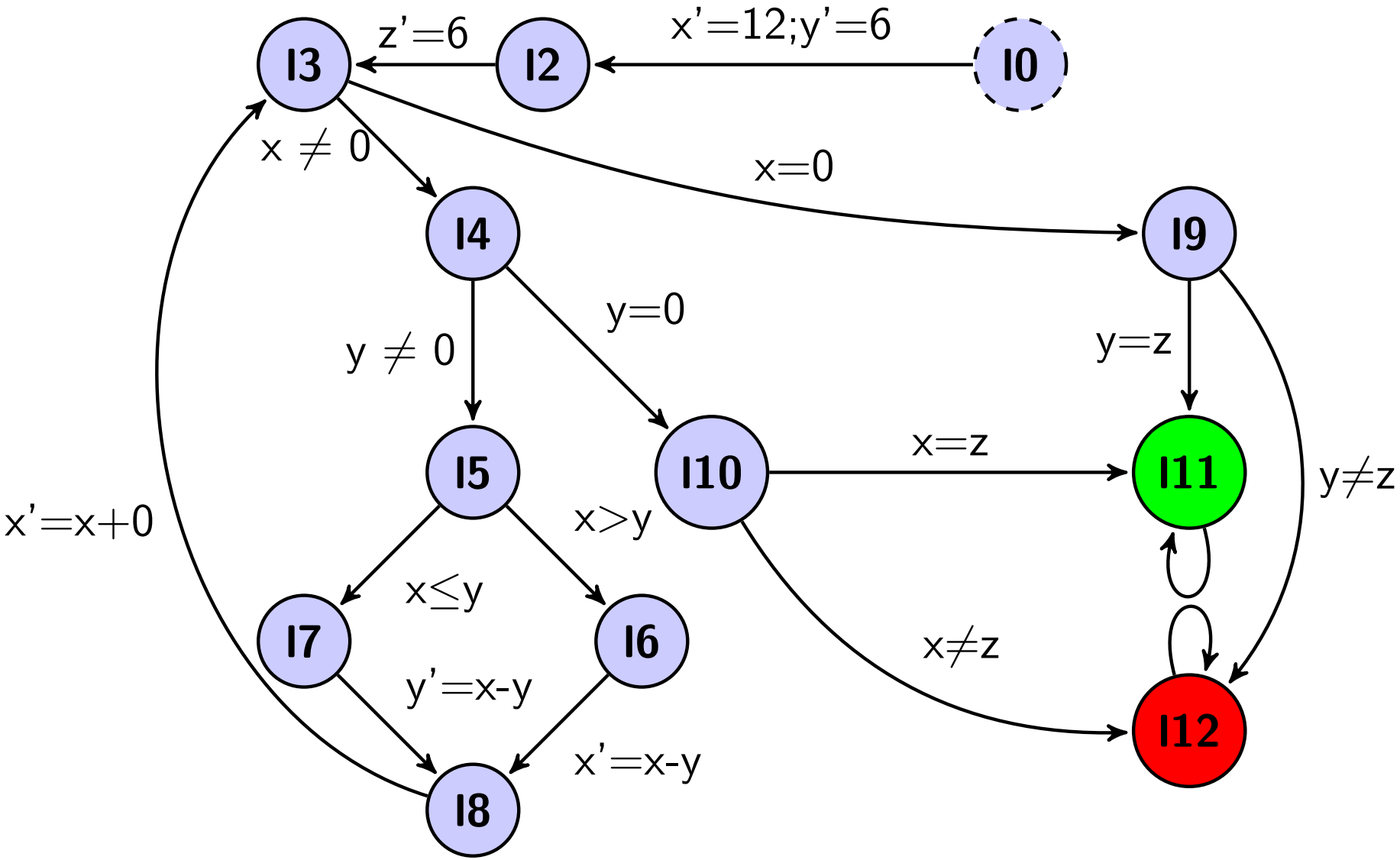
Example


l3
z'=6
l2
y'=6
l1
x'=12
l0
x ≠ 0
x=0
l4
l9
y=0
y ≠ 0
y=z
l5
l10
x=z
l11
y≠z
x'=x+0
x>y
x≤y
l7
l6
x≠z
y'=x-y
l12
x'=x-y
l8

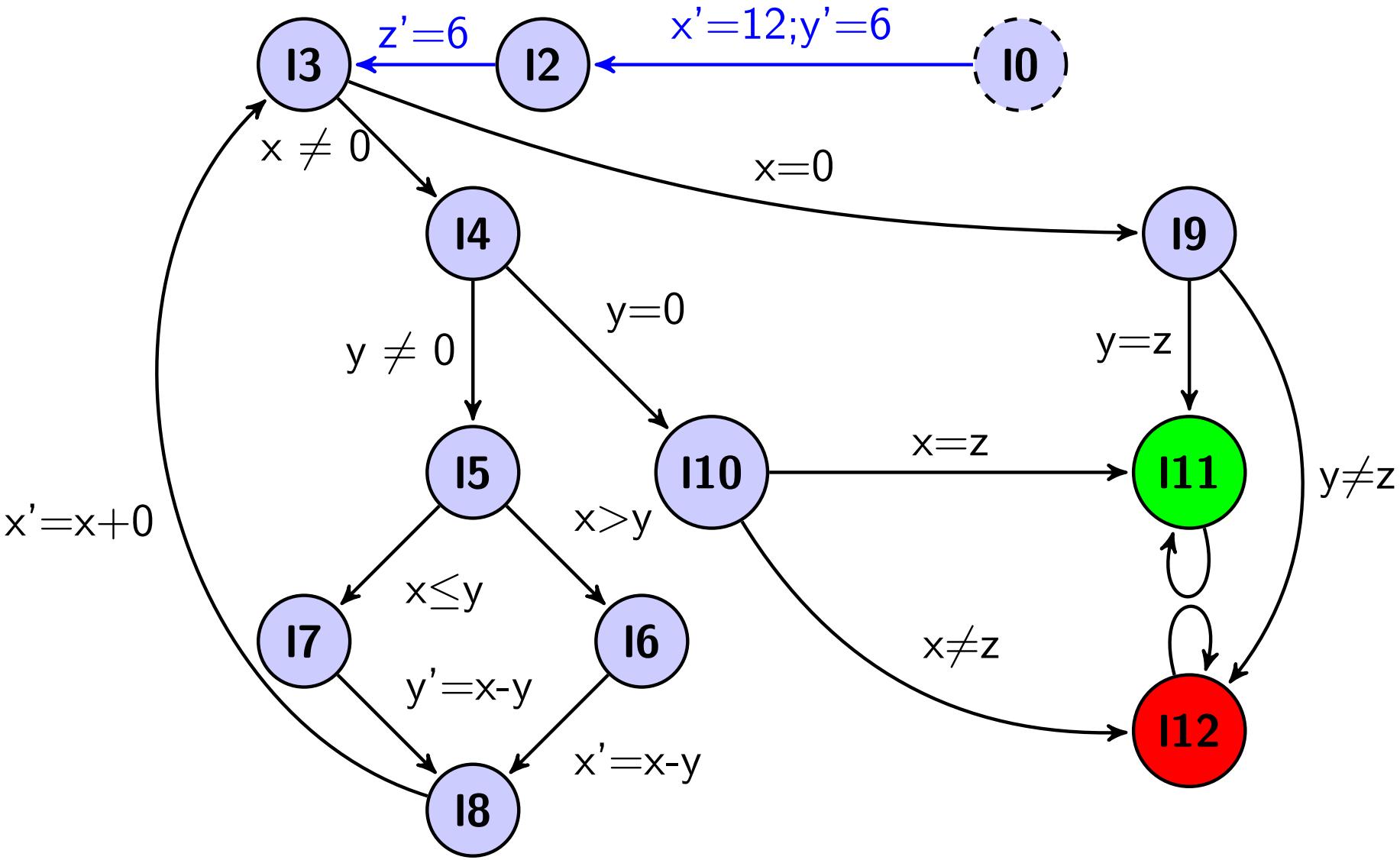
Example


l0
l2
l3
l4
l5
l6
l7
l8
l9
l10
l11
l12
x'=12;y'=6
z'=6
x ≠ 0
x=0
y=0
y ≠ 0
y=z
x=z
y≠z
x>y
x≤y
x≠z
y'=x-y
x'=x-y
x'=x+0

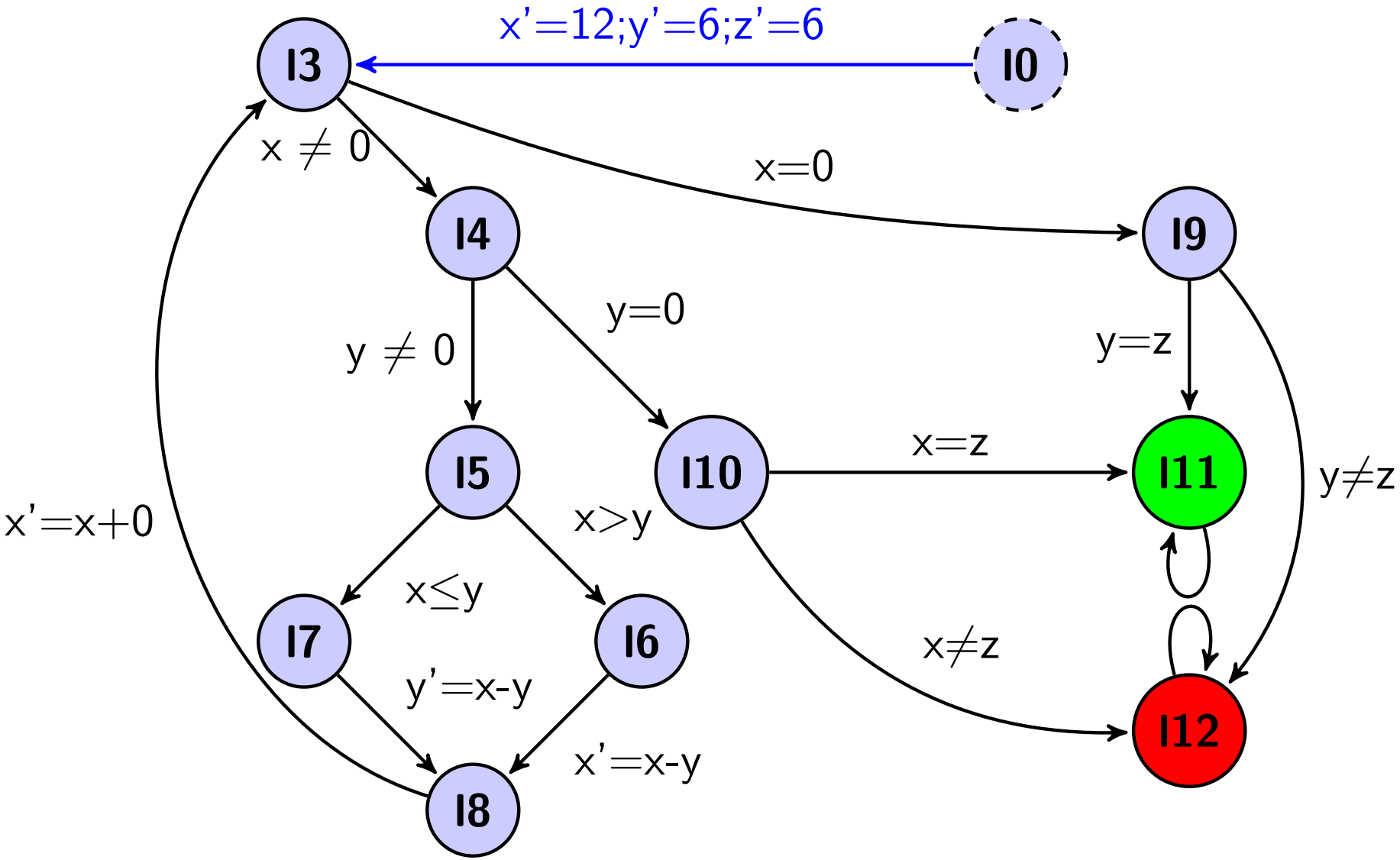
Example


l3
z'=6
l2
x'=12;y'=6
l0
x ≠ 0
x=0
l4
l9
y=0
y ≠ 0
y=z
l5
l10
x=z
l11
y≠z
x'=x+0
x>y
x≤y
l7
l6
x≠z
y'=x-y
l12
x'=x-y
l8

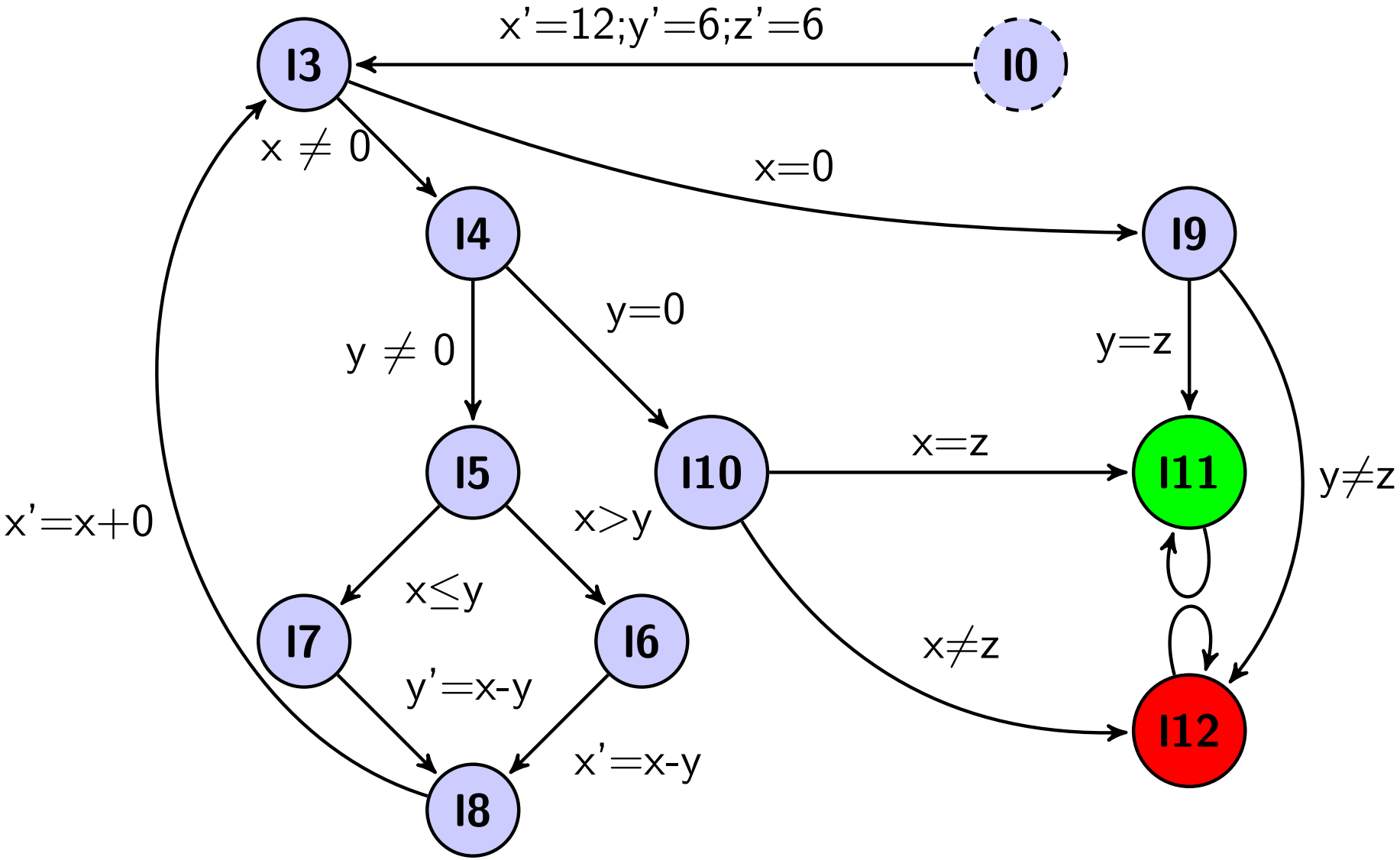
Example


l3
z'=6
l2
x'=12;y'=6
l0
x ≠ 0
x=0
l4
l9
y=0
y ≠ 0
y=z
l5
l10
x=z
l11
y≠z
x'=x+0
x>y
x≤y
l7
l6
x≠z
y'=x-y
l12
x'=x-y
l8

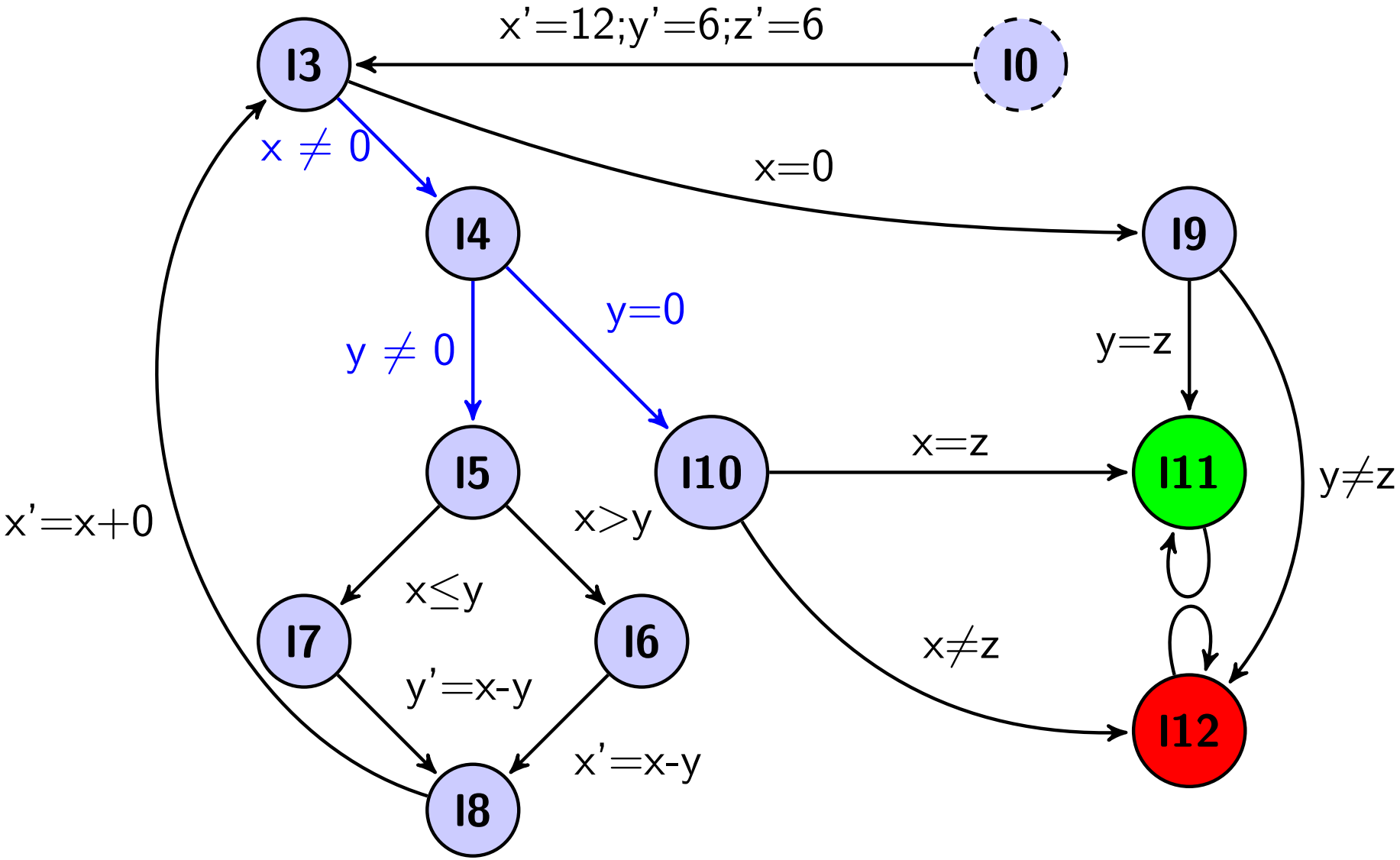
Example


x'=12;y'=6;z'=6
l0
l3
l4
l5
l6
l7
l8
l9
l10
l11
l12
x ≠ 0
x=0
y ≠ 0
y=0
y=z
x=z
y≠z
x>y
x≤y
x≠z
y'=x-y
x'=x-y
x'=x+0

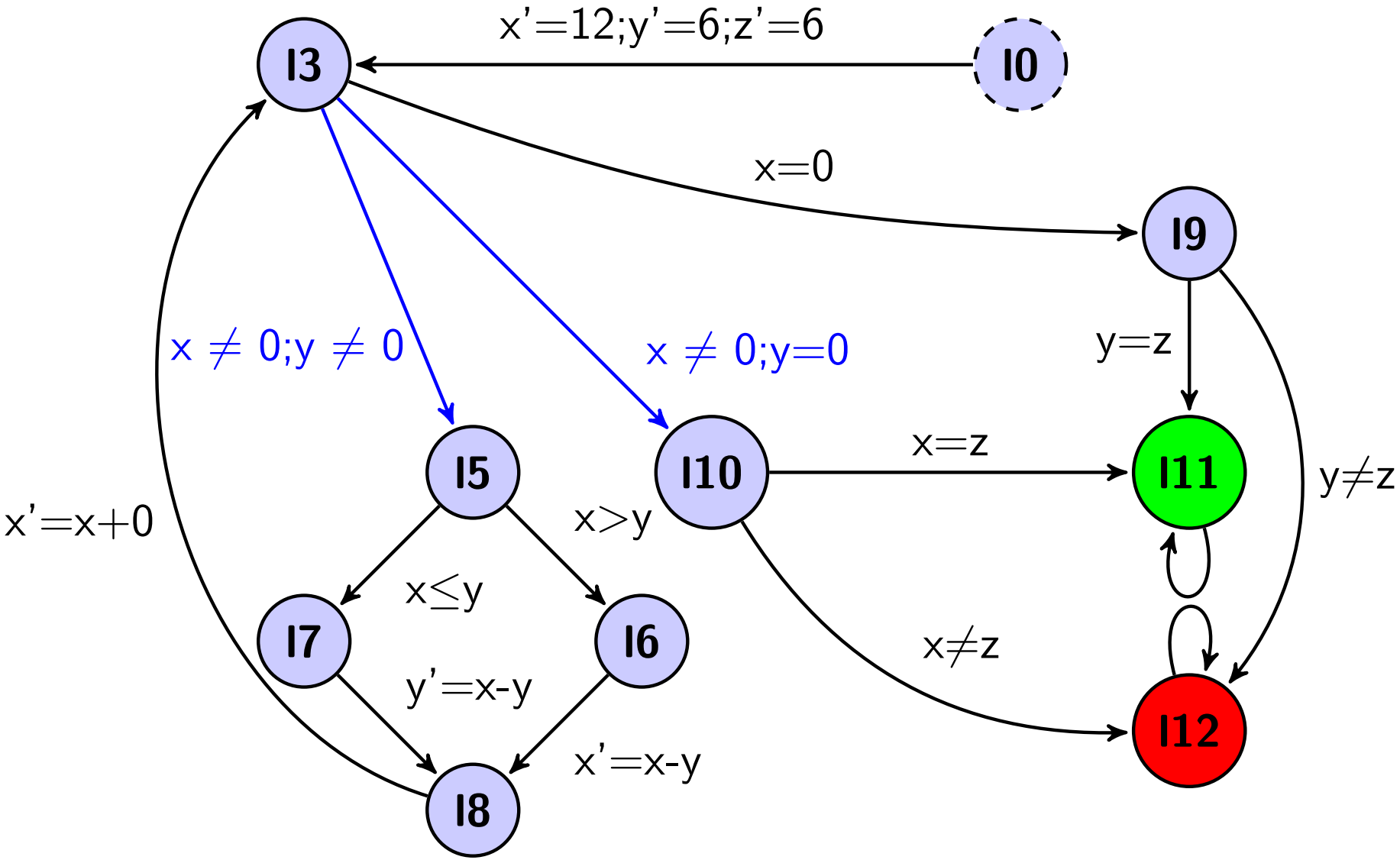
Example


l3
l0
x'=12;y'=6;z'=6
x ≠ 0
x=0
l4
l9
y=0
y ≠ 0
y=z
l5
l10
x=z
l11
y≠z
x'=x+0
x>y
x≤y
l7
l6
x≠z
y'=x-y
l12
x'=x-y
l8

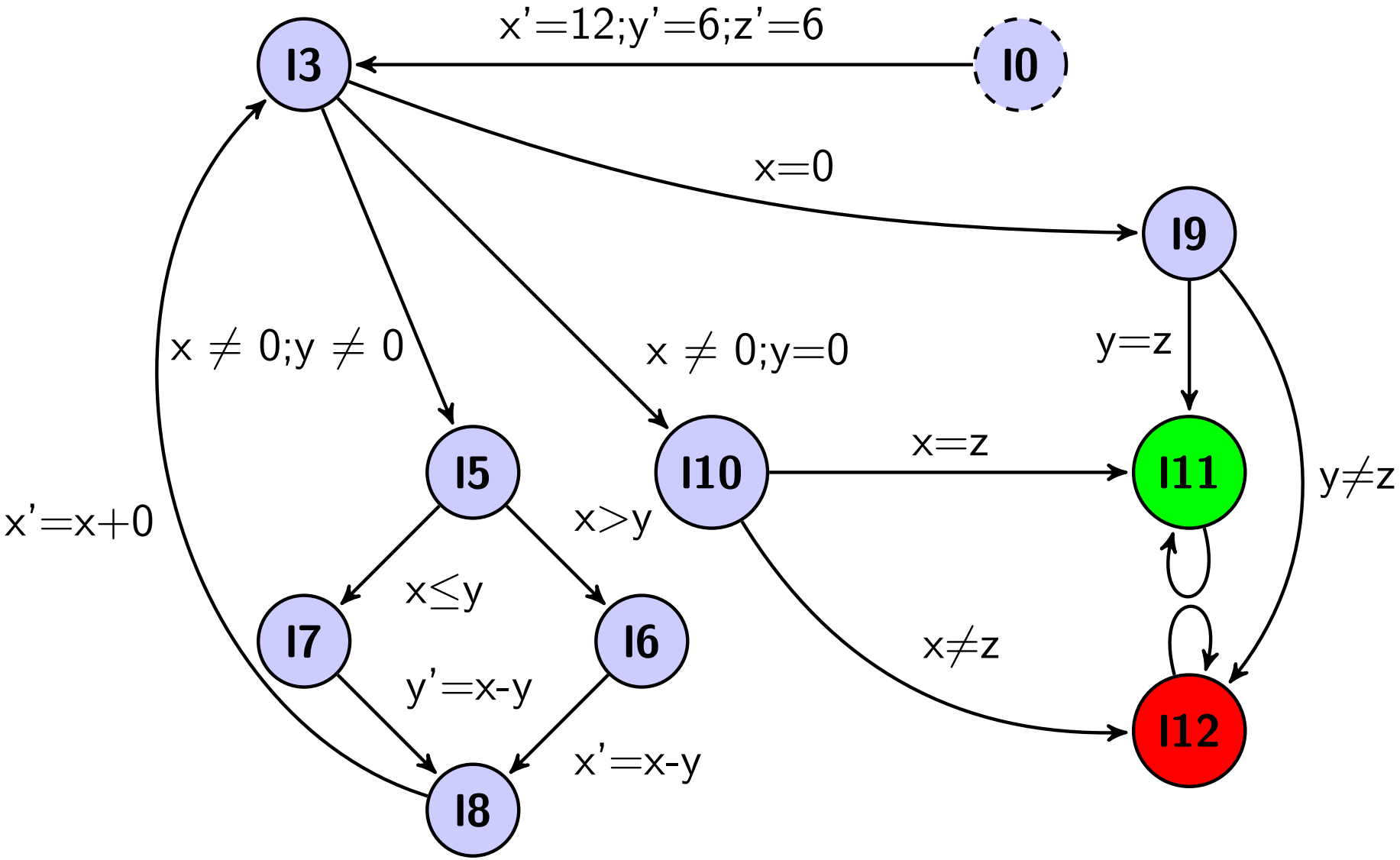
Example


l3
l0
x'=12;y'=6;z'=6
x ≠ 0
x=0
l4
l9
y=0
y ≠ 0
y=z
l5
l10
x=z
l11
y≠z
x'=x+0
x>y
x≤y
l7
l6
x≠z
y'=x-y
l12
x'=x-y
l8

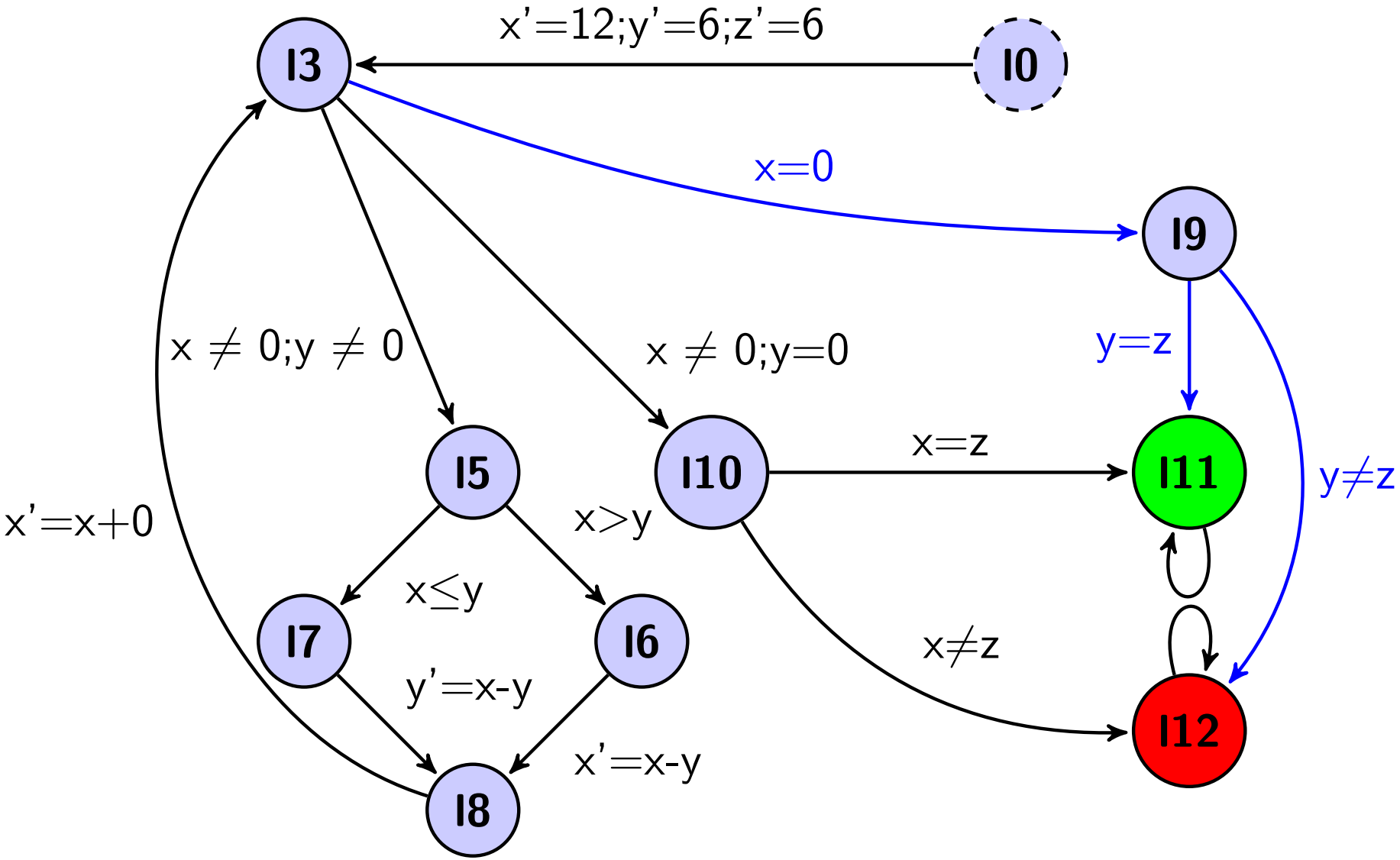
Example


l3
x'=12;y'=6;z'=6
l0
x=0
l9
x ≠ 0;y ≠ 0
x ≠ 0;y=0
y=z
l5
l10
x=z
l11
y≠z
x'=x+0
x>y
x≤y
l7
l6
x≠z
y'=x-y
l12
x'=x-y
l8

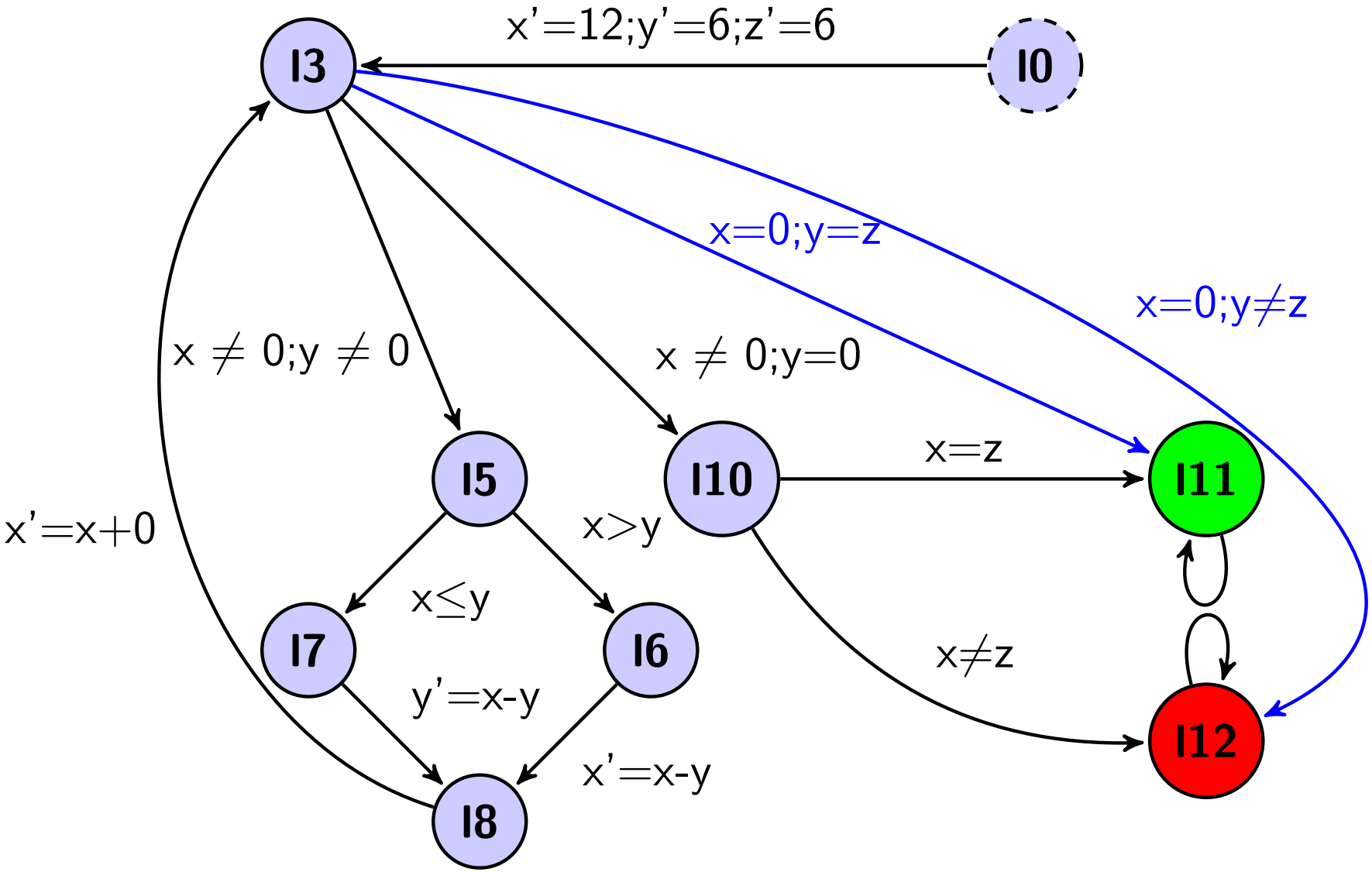
Example


x'=12;y'=6;z'=6
l3
l0
x=0
l9
x ≠ 0;y ≠ 0
x ≠ 0;y=0
y=z
l5
l10
x=z
l11
y≠z
x'=x+0
x>y
x≤y
l7
l6
x≠z
y'=x-y
l12
x'=x-y
l8

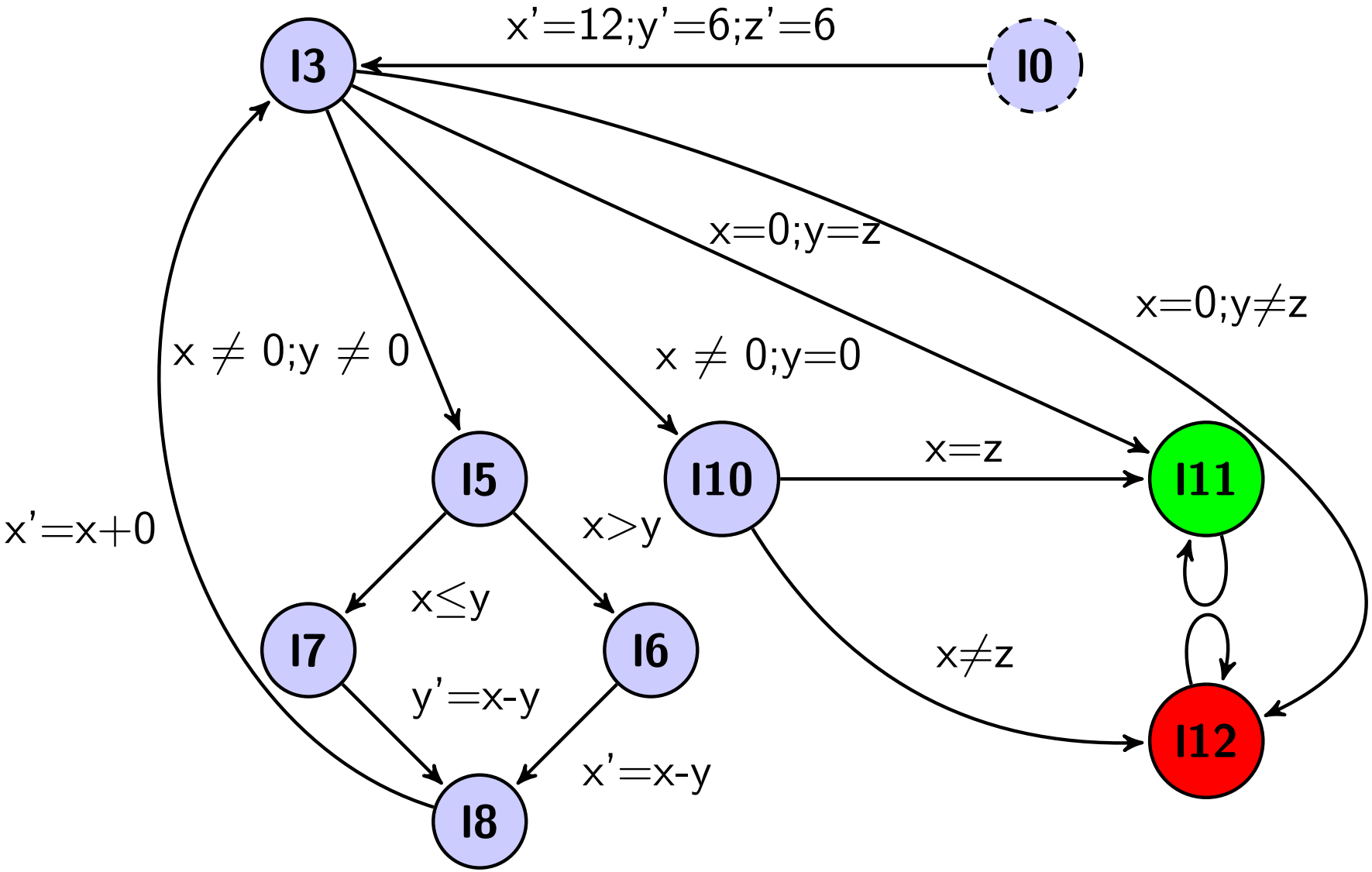
Example


x'=12;y'=6;z'=6
l3
l0
x=0
l9
x ≠ 0;y ≠ 0
x ≠ 0;y=0
y=z
l5
l10
x=z
l11
y≠z
x'=x+0
x>y
x≤y
l7
l6
x≠z
y'=x-y
l12
x'=x-y
l8

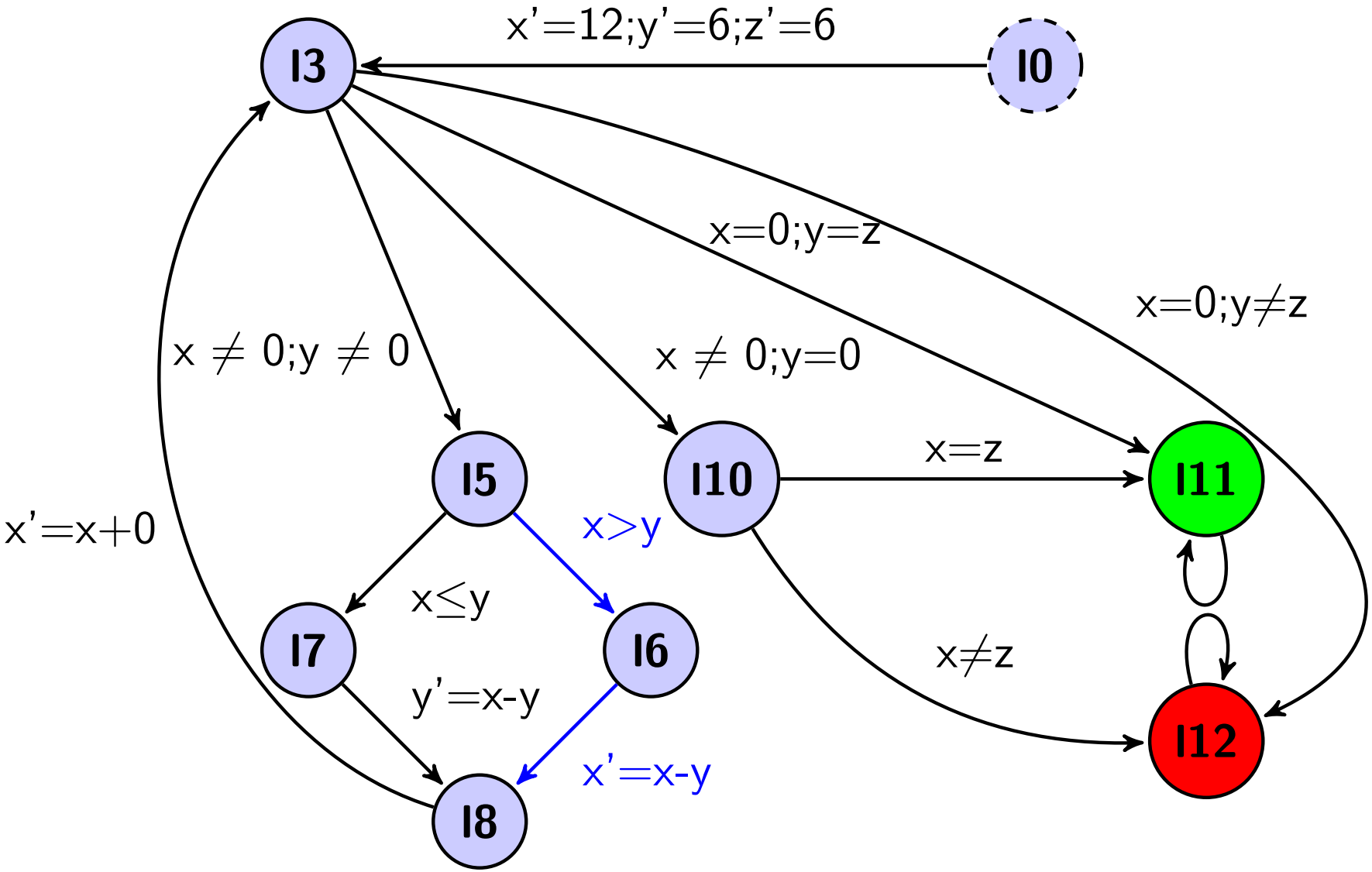
Example


l3
l0
x'=12;y'=6;z'=6
x=0;y=z
x=0;y≠z
x ≠ 0;y ≠ 0
x ≠ 0;y=0
l5
l10
l11
x=z
x'=x+0
x>y
x≤y
l7
l6
x≠z
y'=x-y
l12
x'=x-y
l8

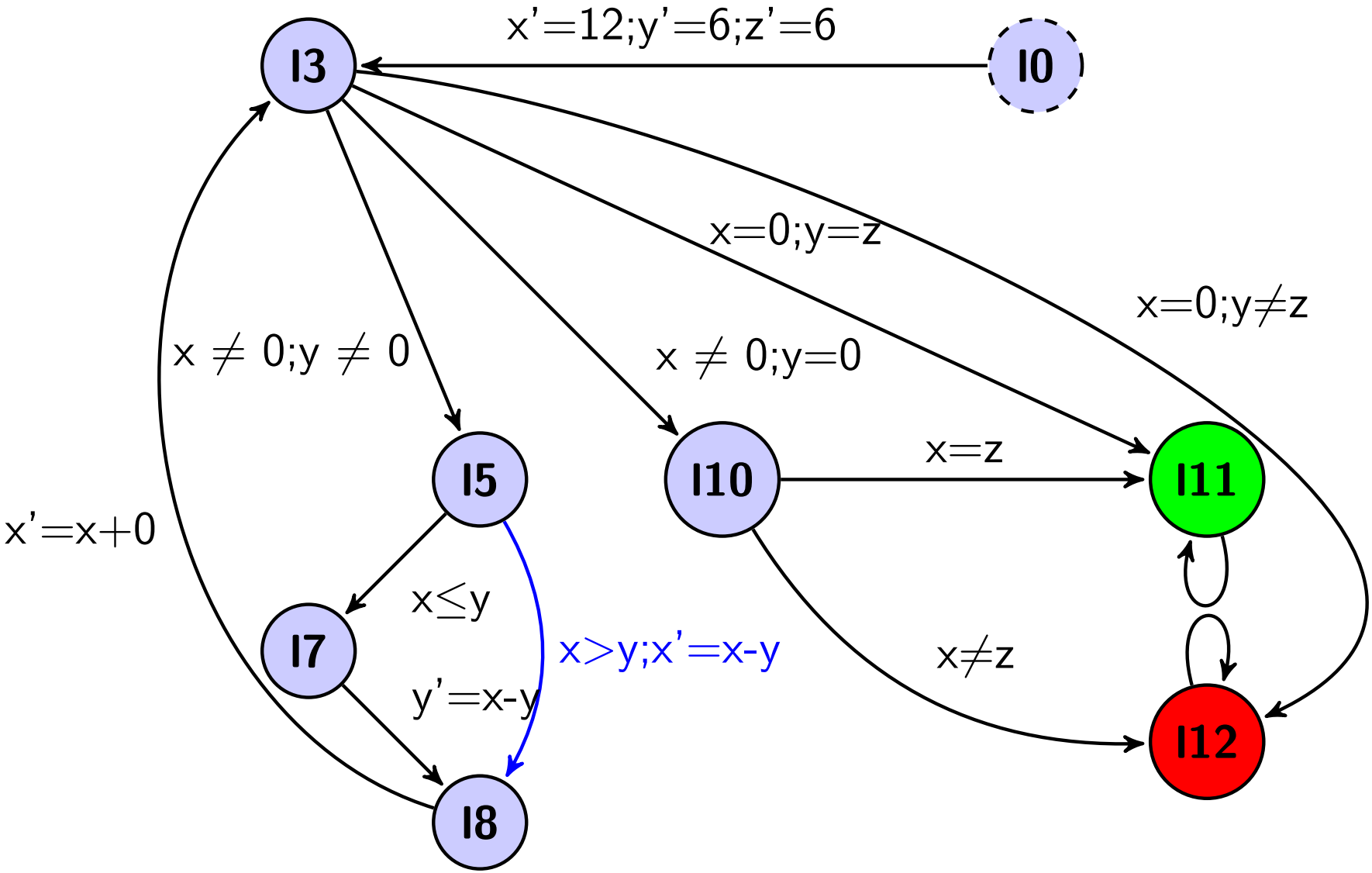
Example


l3
l0
x'=12;y'=6;z'=6
x=0;y=z
x=0;y≠z
x ≠ 0;y ≠ 0
x ≠ 0;y=0
l5
l10
x=z
l11
x'=x+0
x>y
x≤y
l7
l6
x≠z
y'=x-y
l12
x'=x-y
l8

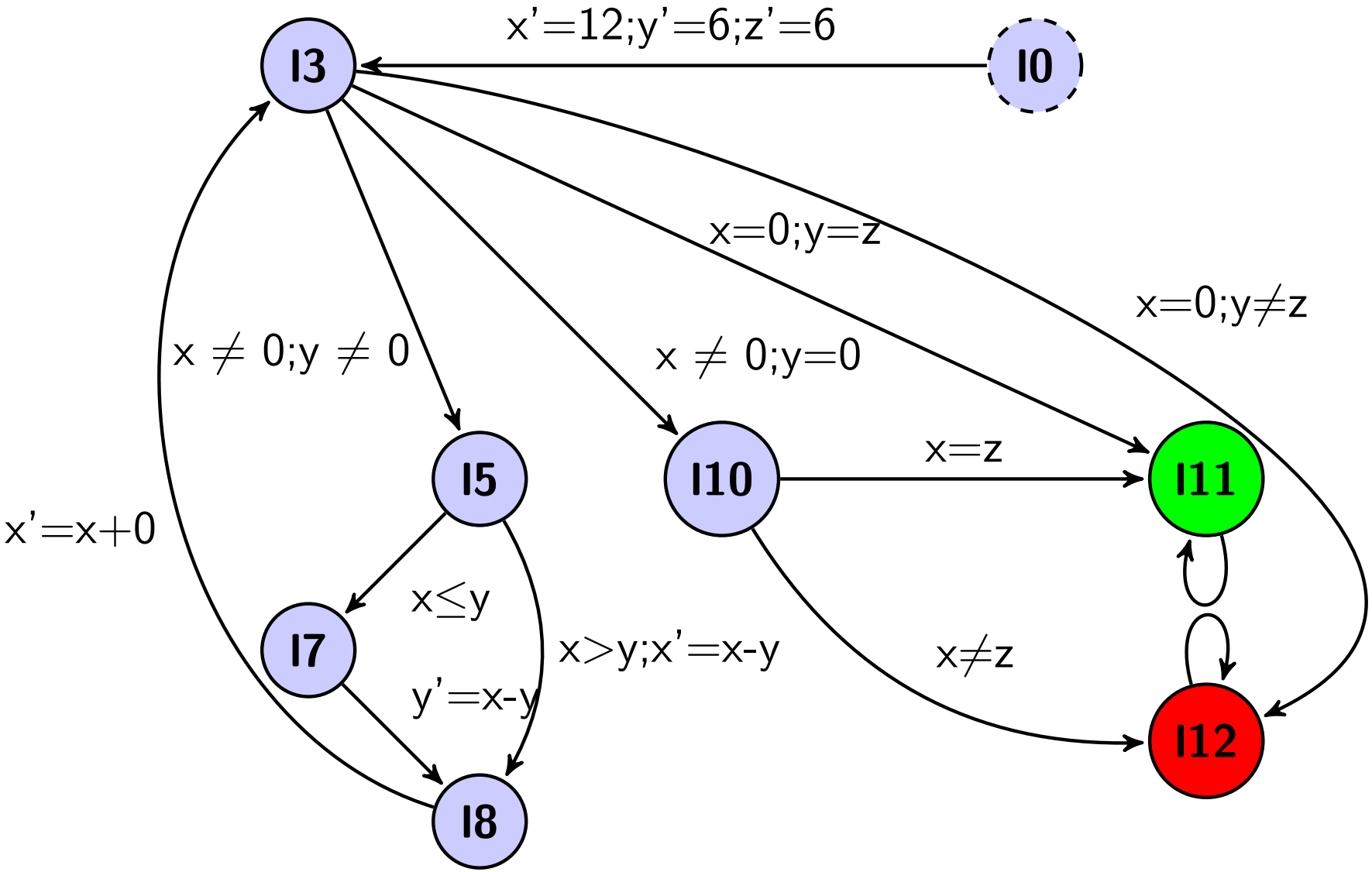
Example


l3
l0
x'=12;y'=6;z'=6
x=0;y=z
x=0;y≠z
x ≠ 0;y ≠ 0
x ≠ 0;y=0
l5
l10
x=z
l11
x'=x+0
x>y
x≤y
l7
l6
x≠z
y'=x-y
l12
x'=x-y
l8

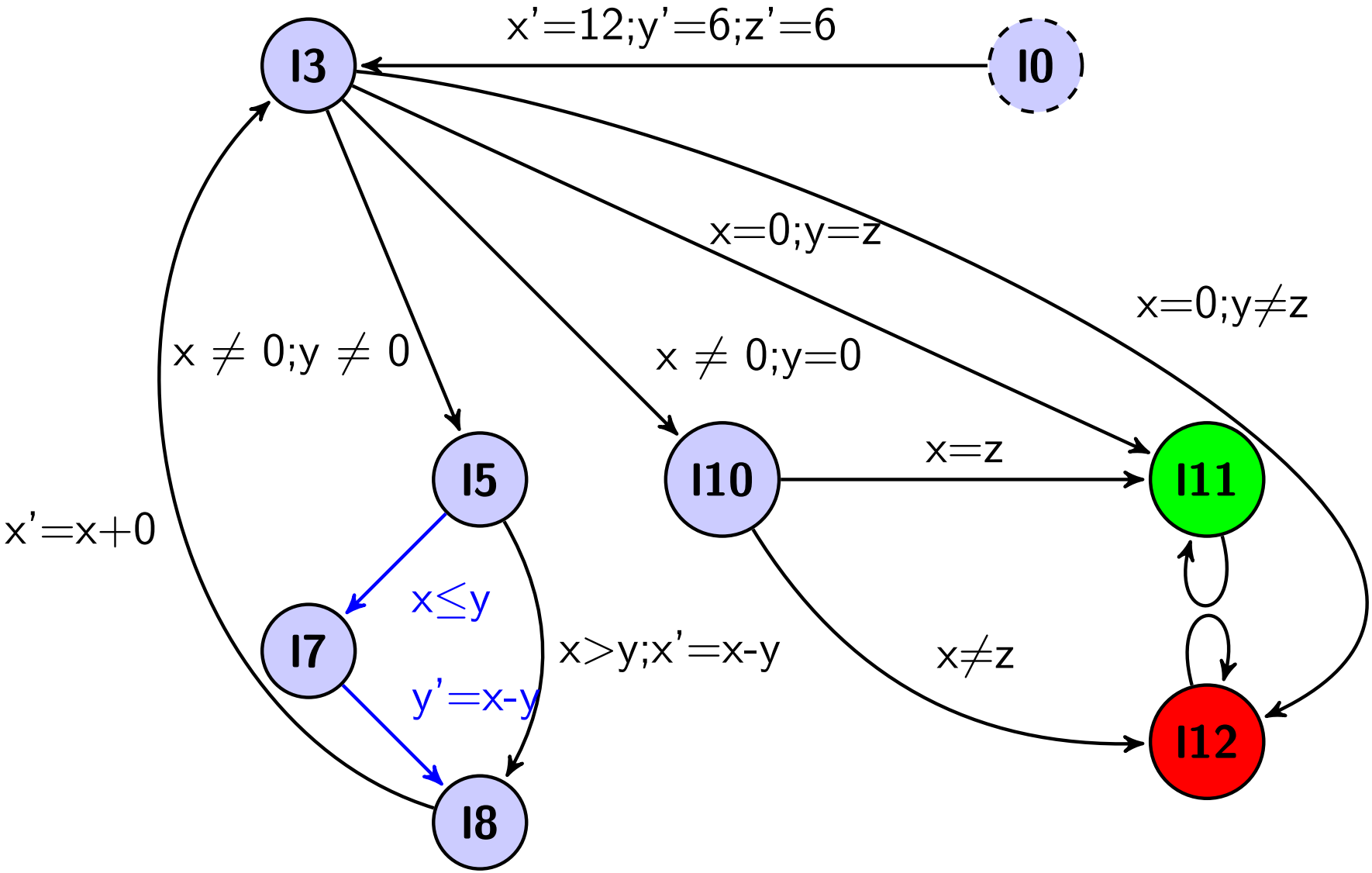
Example


x'=12;y'=6;z'=6
l3
l0
x=0;y=z
x=0;y≠z
x ≠ 0;y ≠ 0
x ≠ 0;y=0
l5
l10
x=z
l11
x'=x+0
x≤y
l7
x>y;x'=x-y
x≠z
y'=x-y
l12
l8

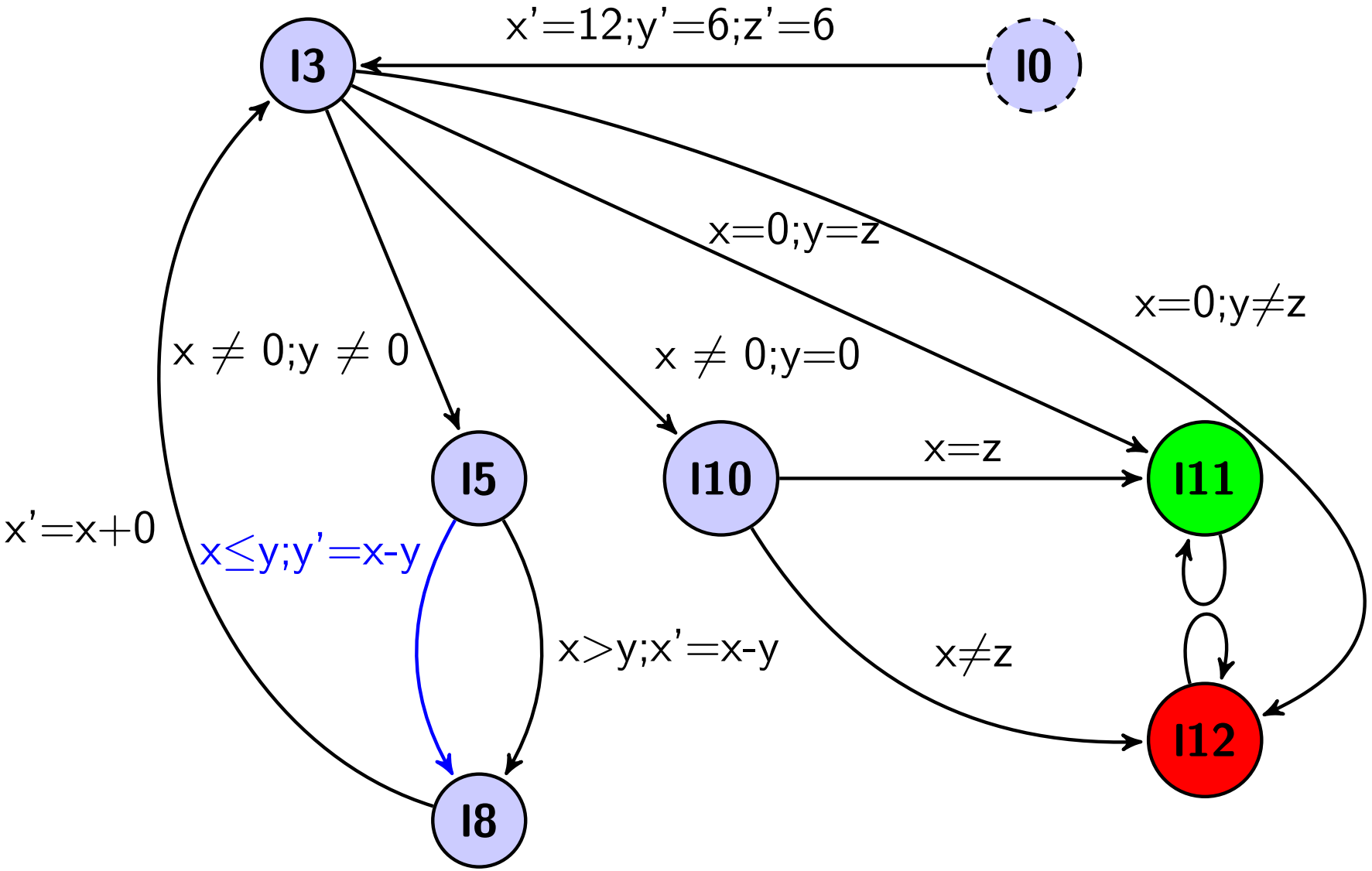
Example


l3
x'=12;y'=6;z'=6
l0
x=0;y=z
x=0;y≠z
x ≠ 0;y ≠ 0
x ≠ 0;y=0
l5
l10
x=z
l11
x'=x+0
x≤y
l7
x>y;x'=x-y
x≠z
y'=x-y
l12
l8

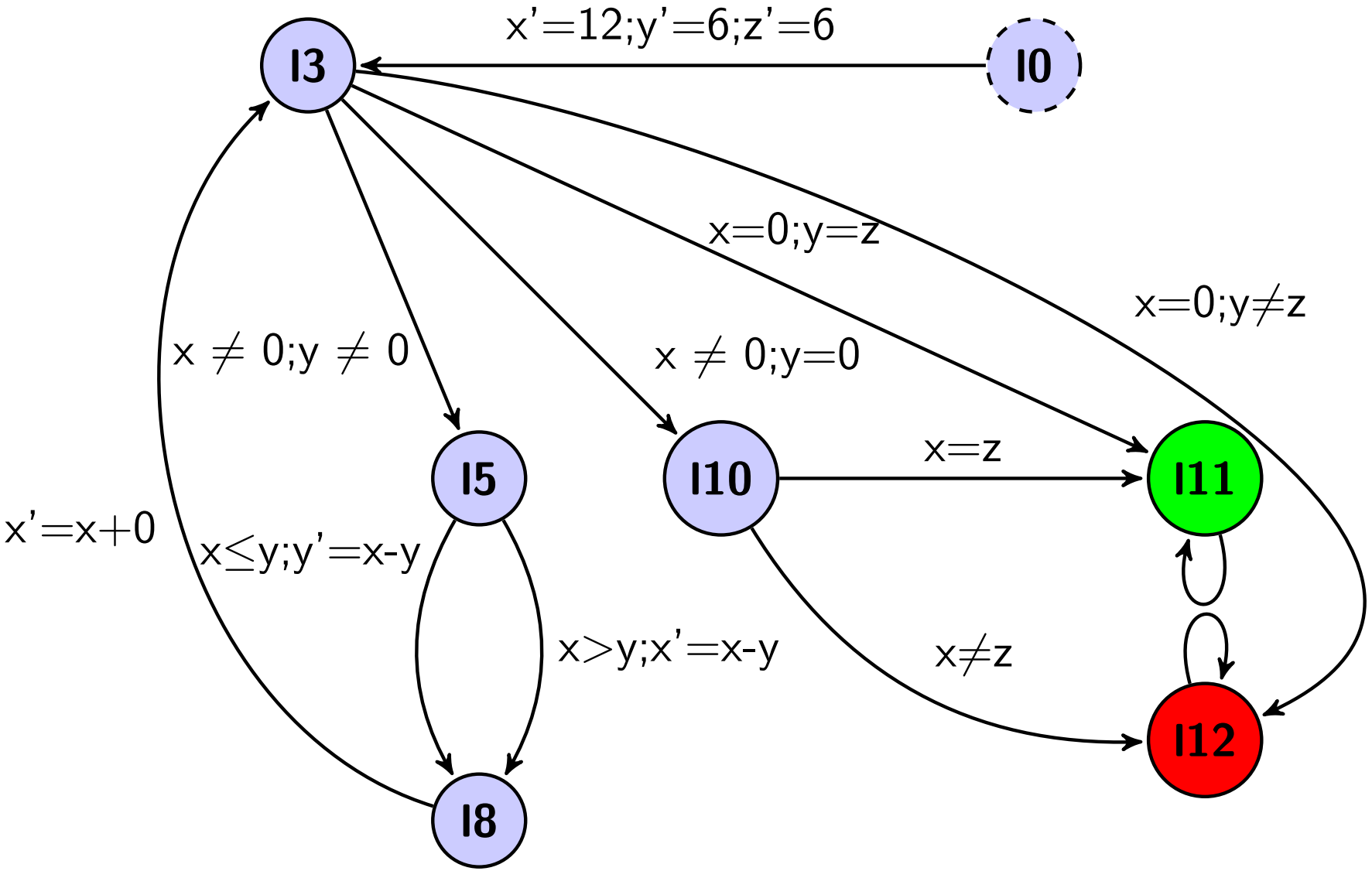
Example


l3
l0
x'=12;y'=6;z'=6
x=0;y=z
x=0;y≠z
x ≠ 0;y ≠ 0
x ≠ 0;y=0
l5
l10
x=z
l11
x'=x+0
x≤y
l7
x>y;x'=x-y
x≠z
y'=x-y
l12
l8

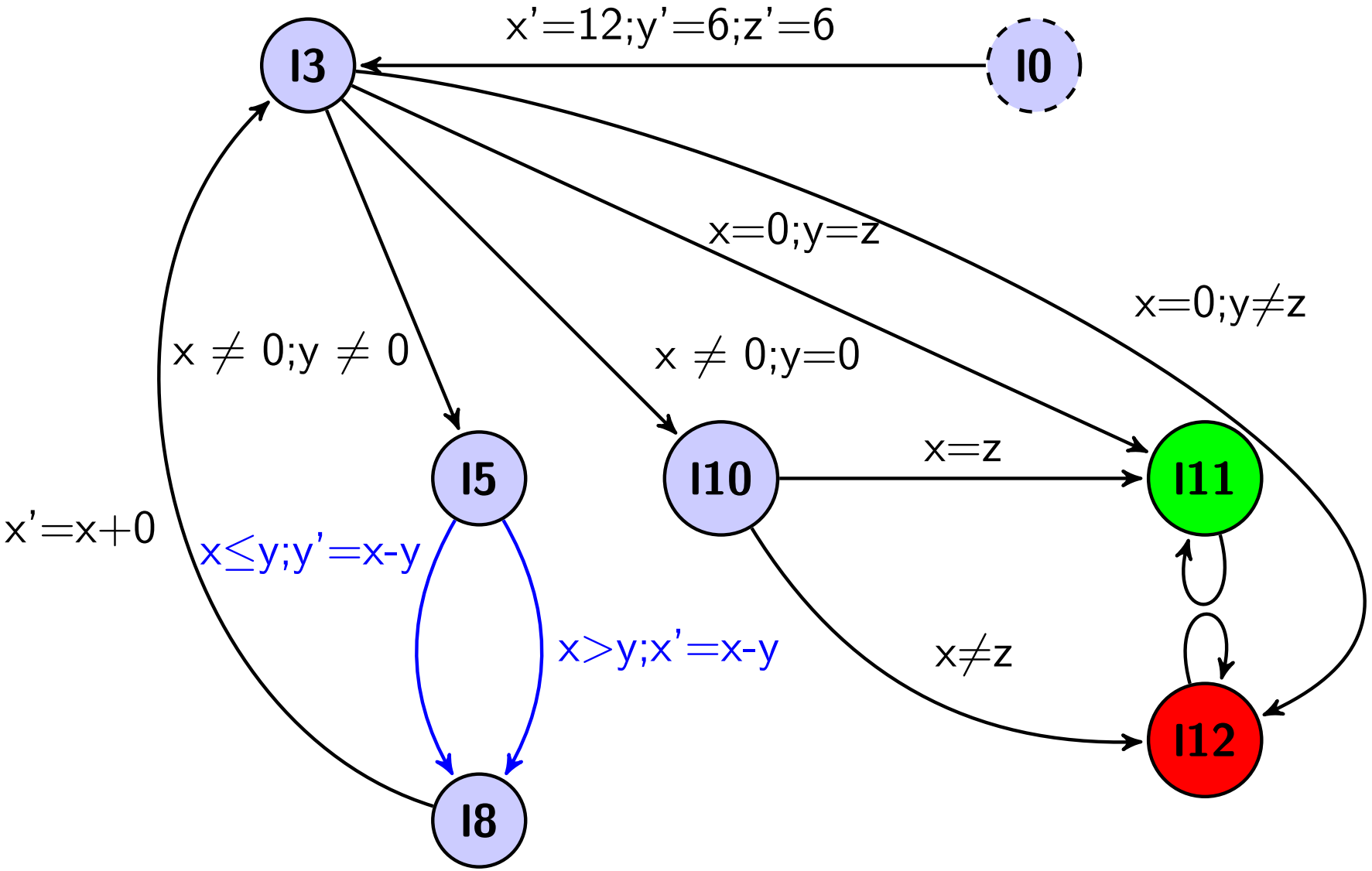
Example


l3
x'=12;y'=6;z'=6
l0
x=0;y=z
x=0;y≠z
x ≠ 0;y ≠ 0
x ≠ 0;y=0
l5
l10
x=z
l11
x'=x+0
x≤y;y'=x-y
x>y;x'=x-y
x≠z
l12
l8

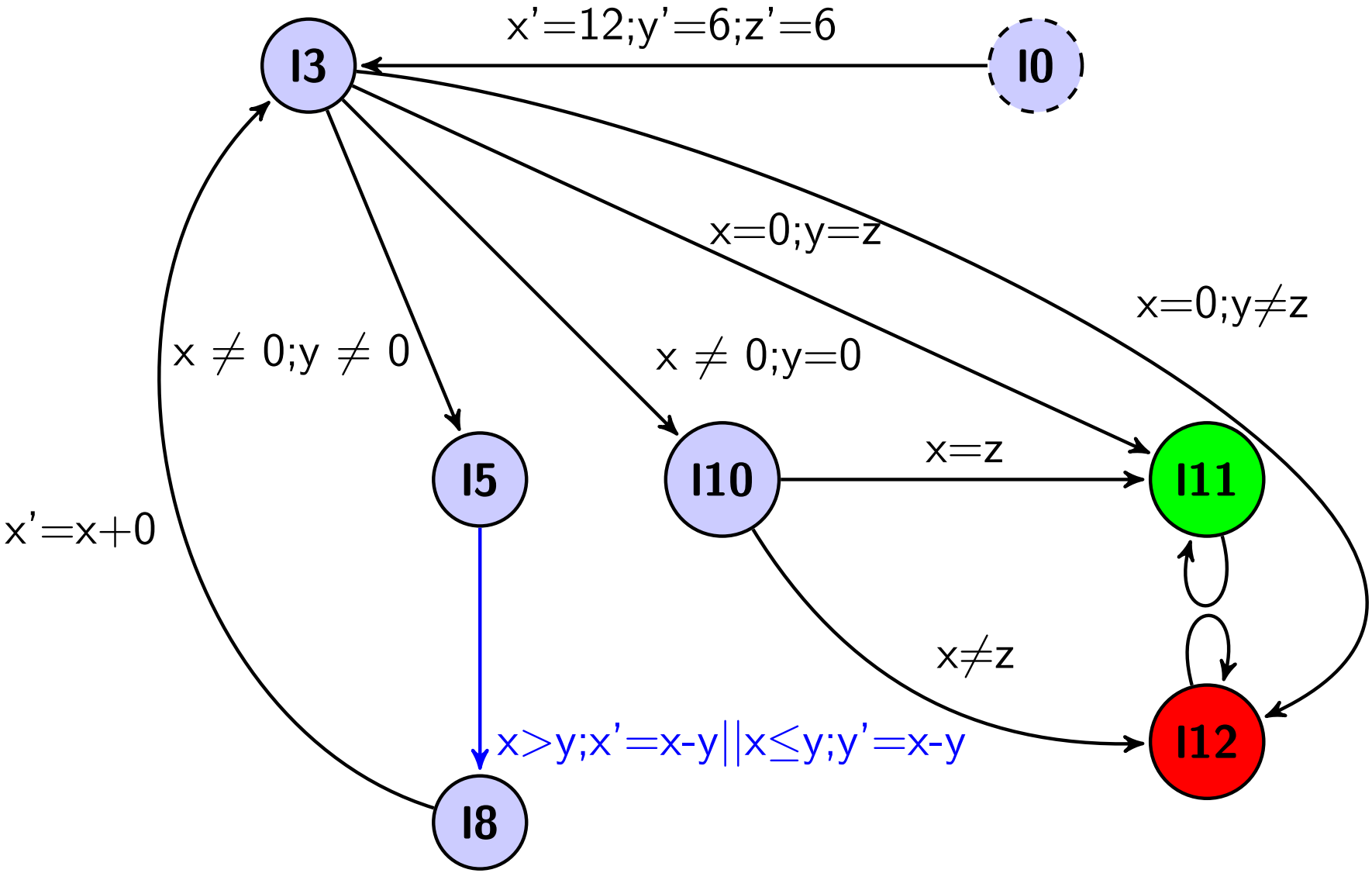
Example


l3
l0
x'=12;y'=6;z'=6
x=0;y=z
x=0;y≠z
x ≠ 0;y ≠ 0
x ≠ 0;y=0
l5
l10
x=z
l11
x'=x+0
x≤y;y'=x-y
x>y;x'=x-y
x≠z
l12
l8

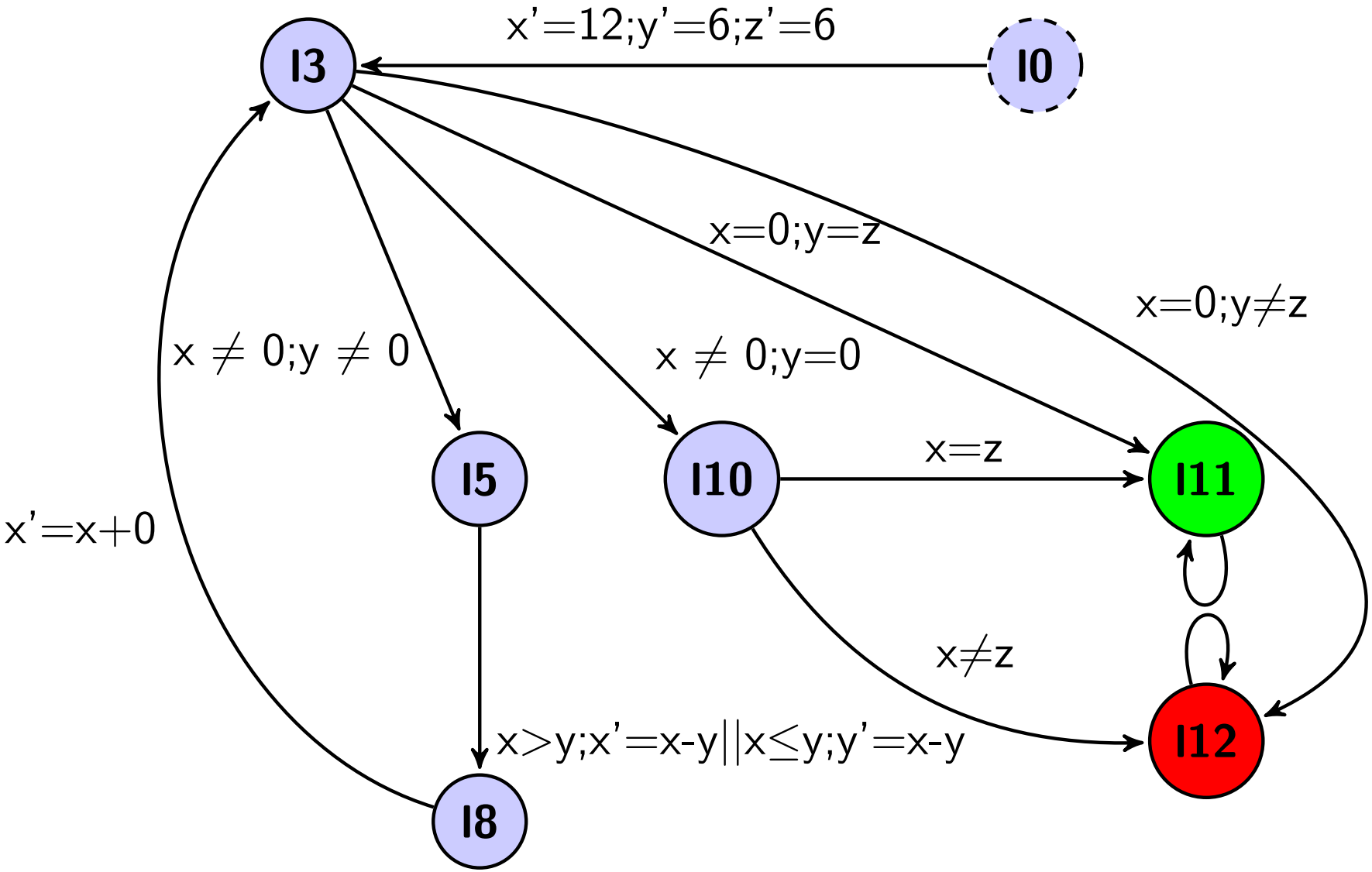
Example


l3
l0
x'=12;y'=6;z'=6
x=0;y=z
x=0;y≠z
x ≠ 0;y ≠ 0
x ≠ 0;y=0
l5
l10
x=z
l11
x'=x+0
x≤y;y'=x-y
x>y;x'=x-y
x≠z
l12
l8

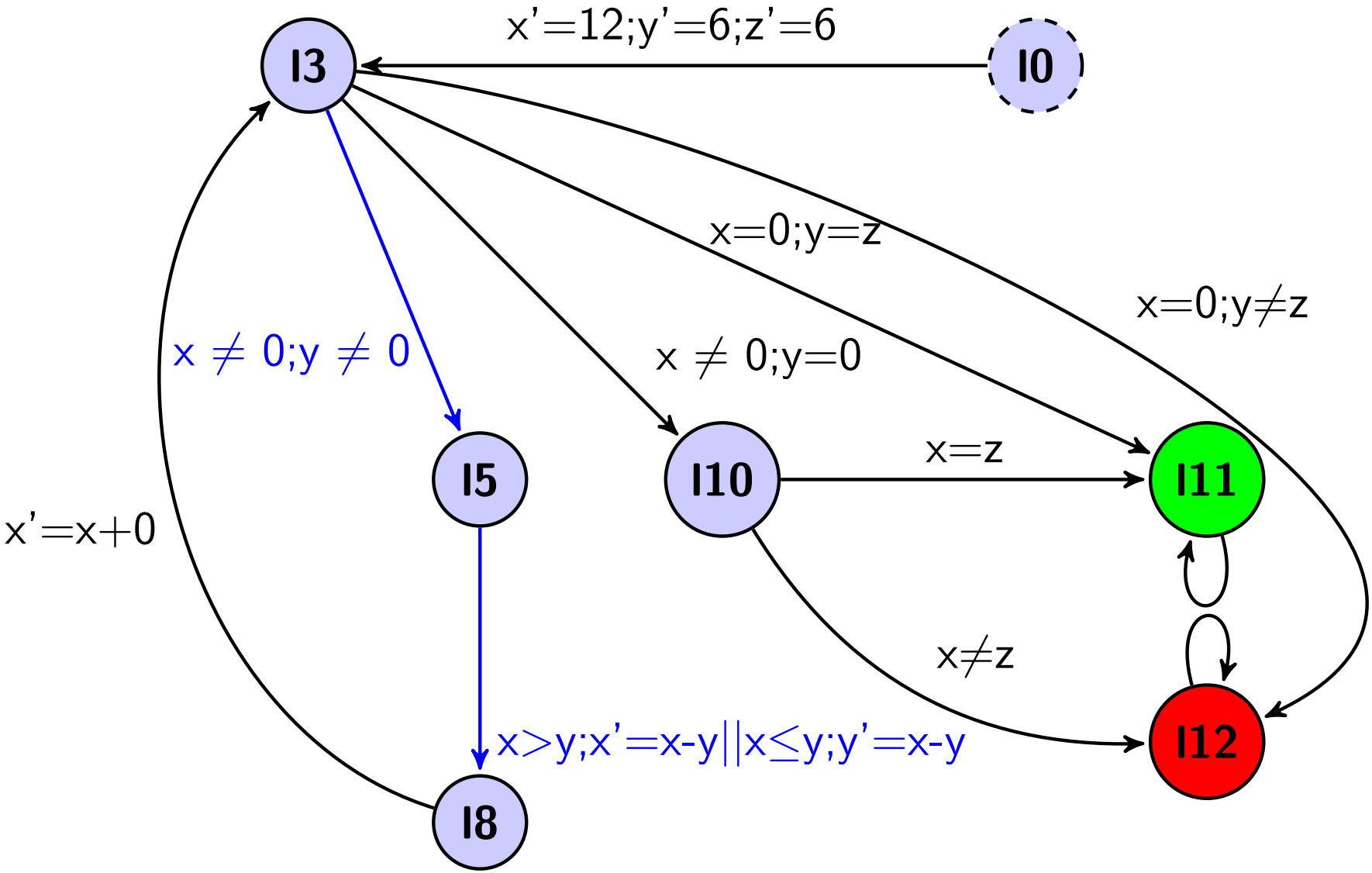
Example


l3
l0
x'=12;y'=6;z'=6
x=0;y=z
x=0;y≠z
x ≠ 0;y ≠ 0
x ≠ 0;y=0
l5
l10
x=z
l11
x'=x+0
x≠z
l12
x>y;x'=x-y||x≤y;y'=x-y
l8

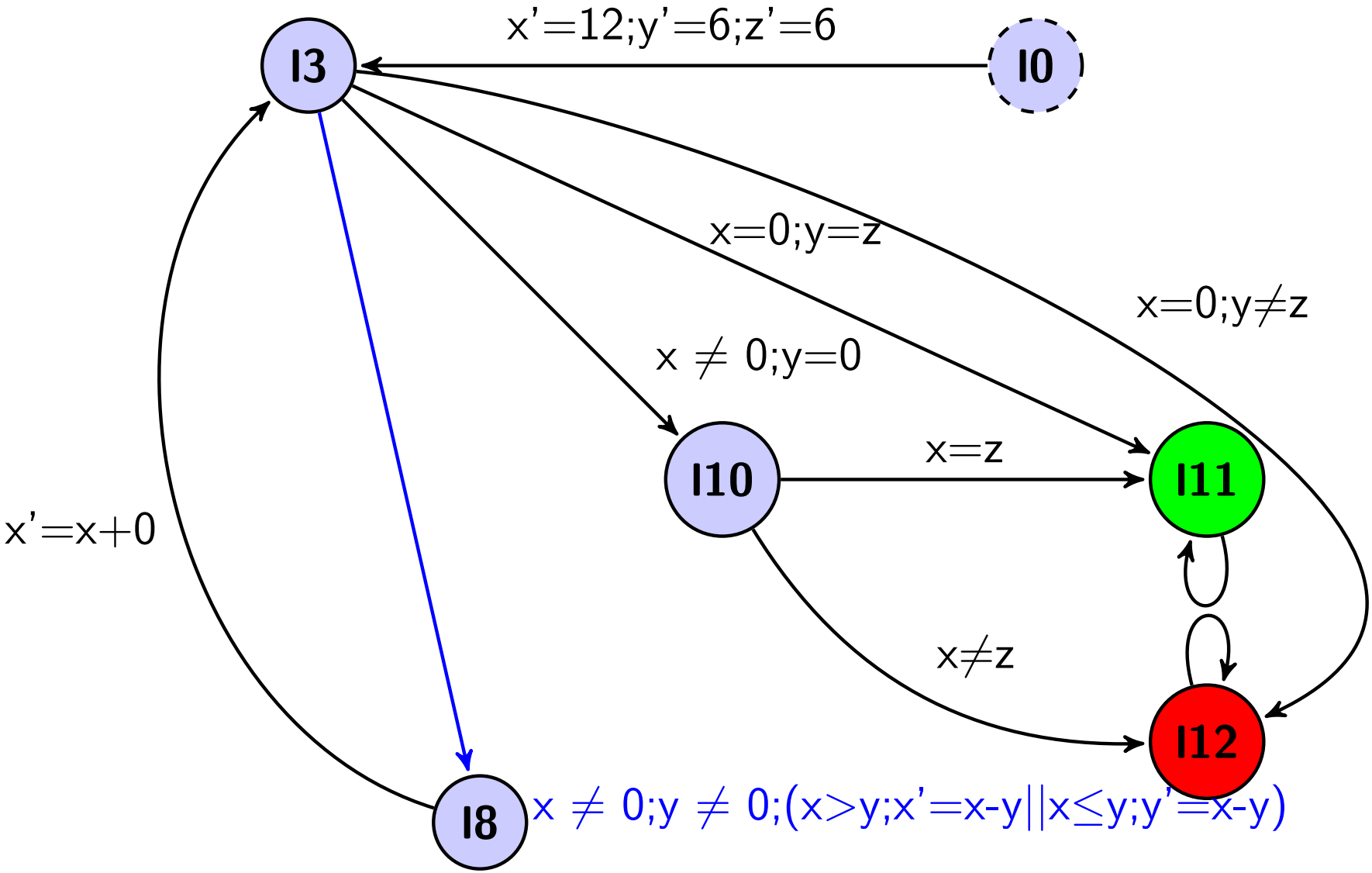
Example


l3
l0
x'=12;y'=6;z'=6
x=0;y=z
x=0;y≠z
x ≠ 0;y ≠ 0
x ≠ 0;y=0
l5
l10
x=z
l11
x'=x+0
x≠z
l12
x>y;x'=x-y||x≤y;y'=x-y
l8

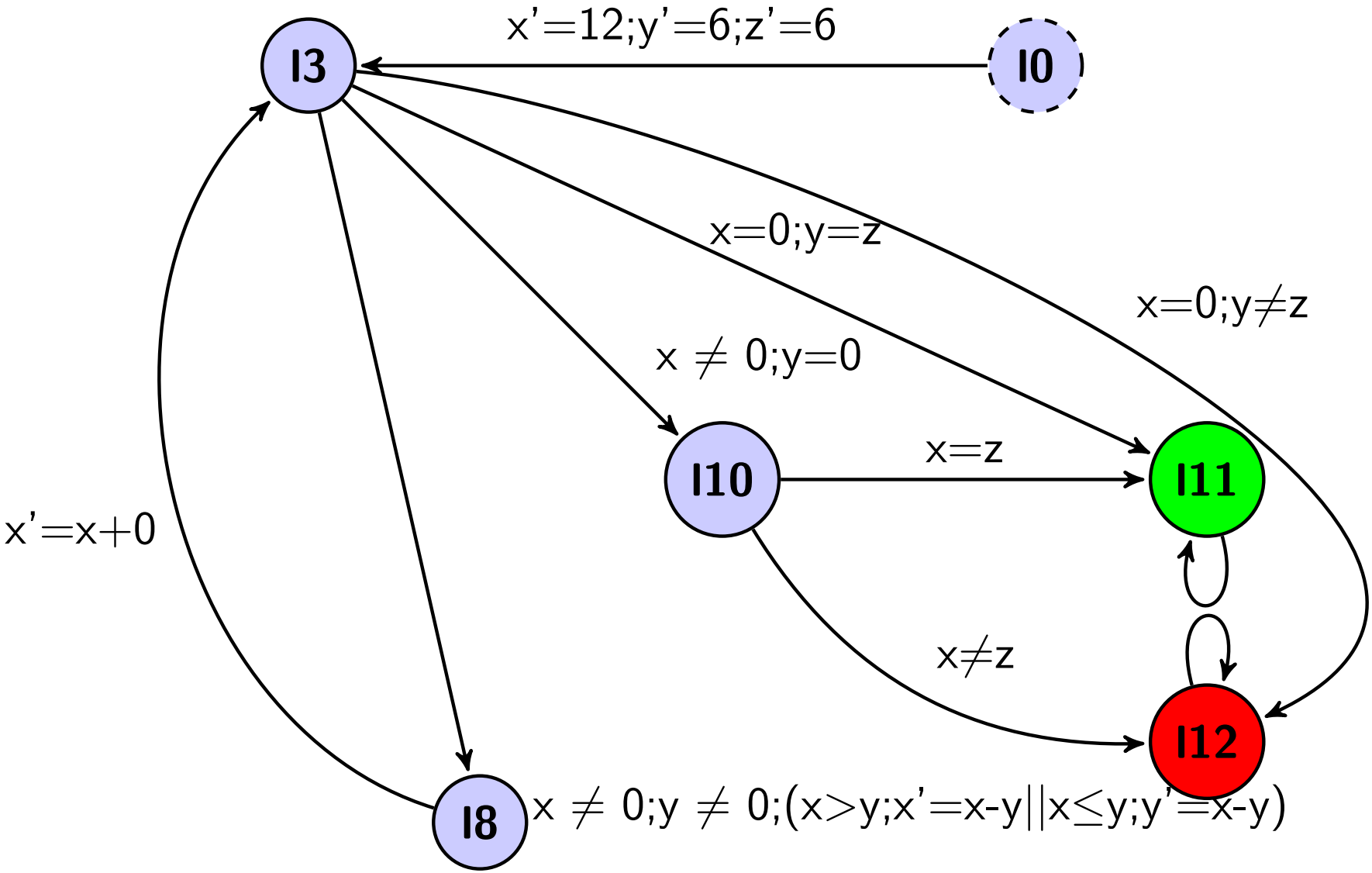
Example


l3
l0
x'=12;y'=6;z'=6
x=0;y=z
x=0;y≠z
x ≠ 0;y ≠ 0
x ≠ 0;y=0
l5
l10
x=z
l11
x'=x+0
x≠z
l12
x>y;x'=x-y||x≤y;y'=x-y
l8

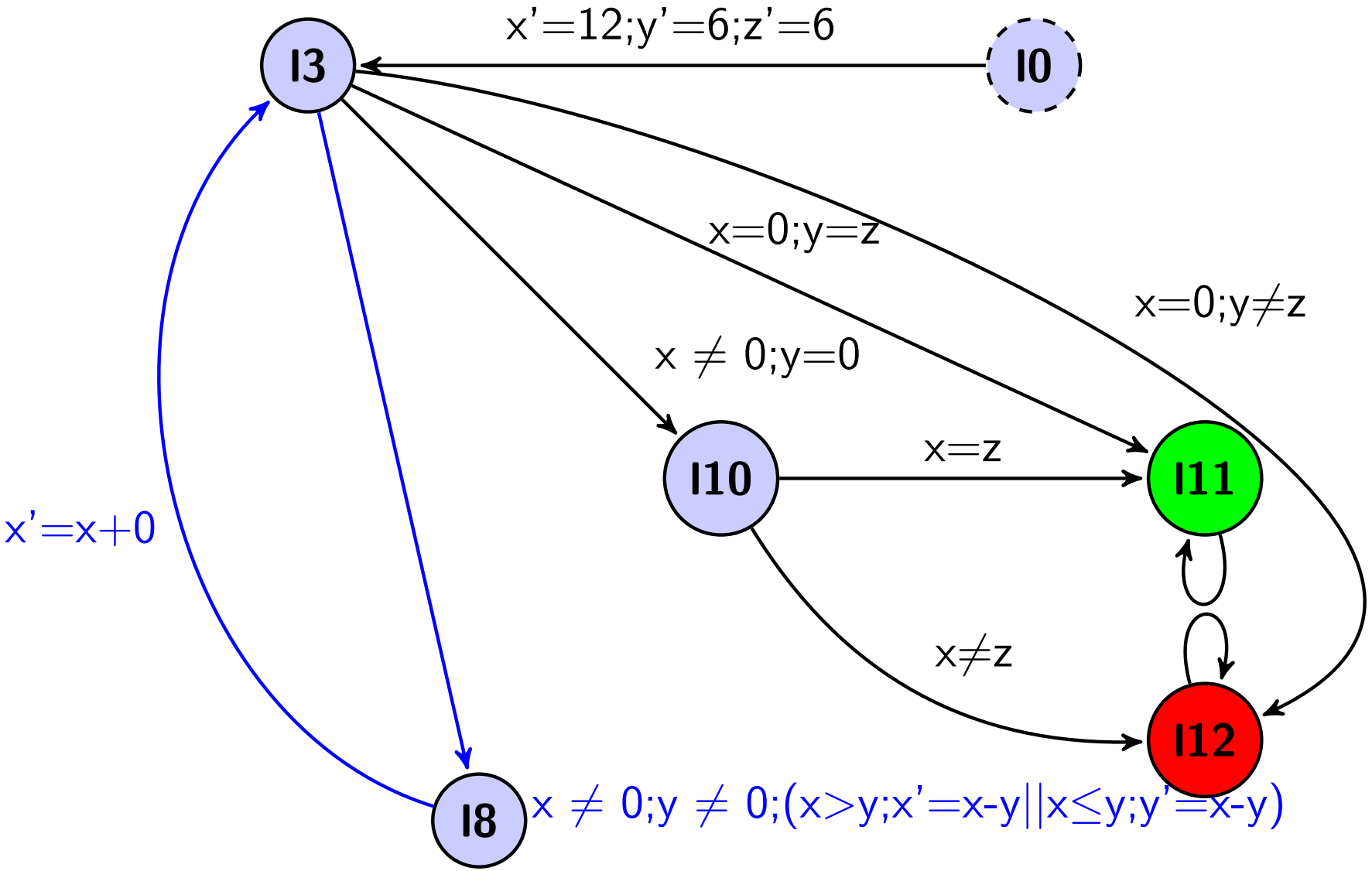
Example


l3
l0
x'=12;y'=6;z'=6
x=0;y=z
x=0;y≠z
x ≠ 0;y=0
l10
x=z
l11
x'=x+0
x≠z
l12
l8
x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y)

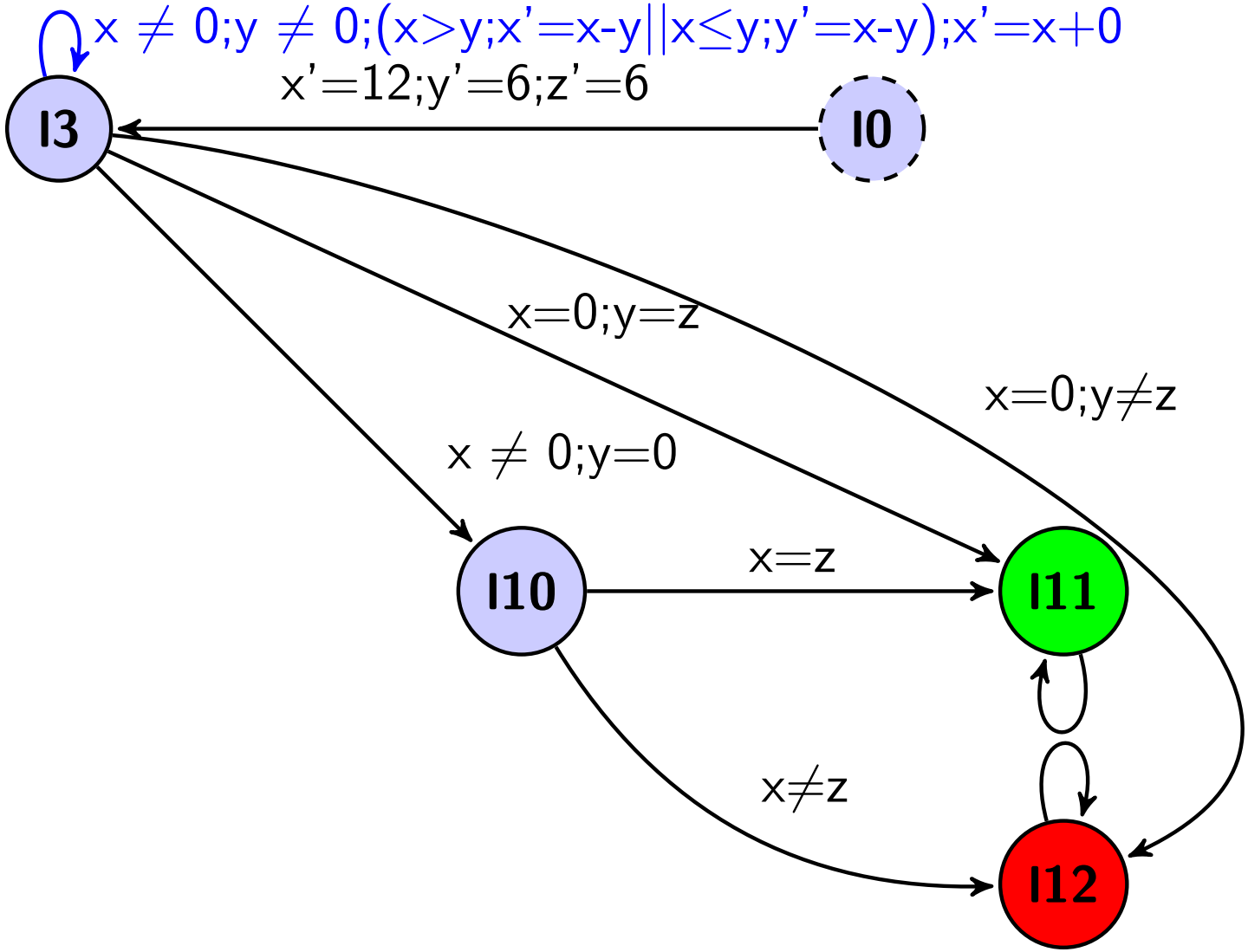
Example


l3
l0
x'=12;y'=6;z'=6
x=0;y=z
x=0;y≠z
x ≠ 0;y=0
l10
x=z
l11
x'=x+0
x≠z
l12
l8
x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y)

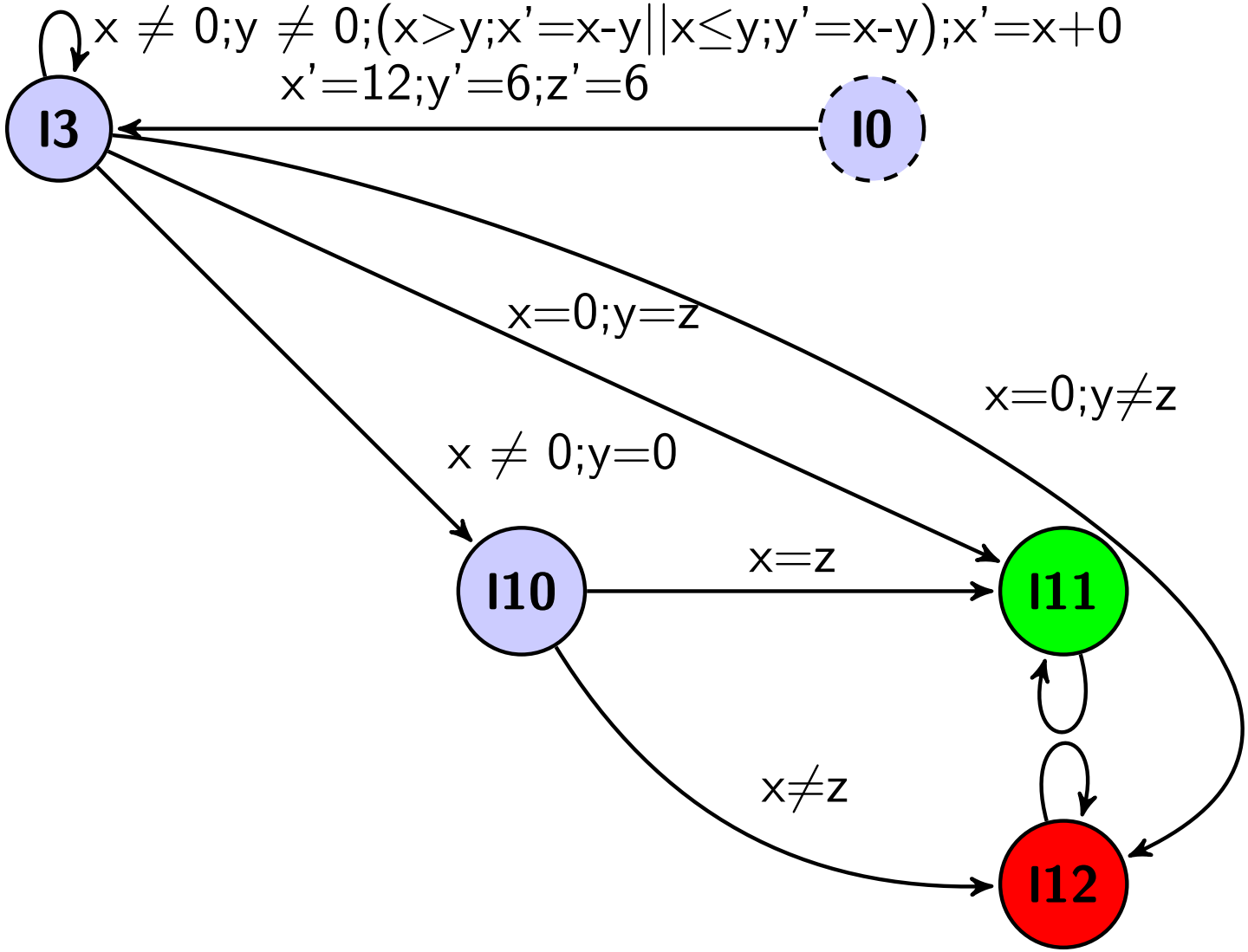
Example


l3
l0
x'=12;y'=6;z'=6
x=0;y=z
x=0;y≠z
x ≠ 0;y=0
l10
x=z
l11
x'=x+0
x≠z
l12
l8
x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y)

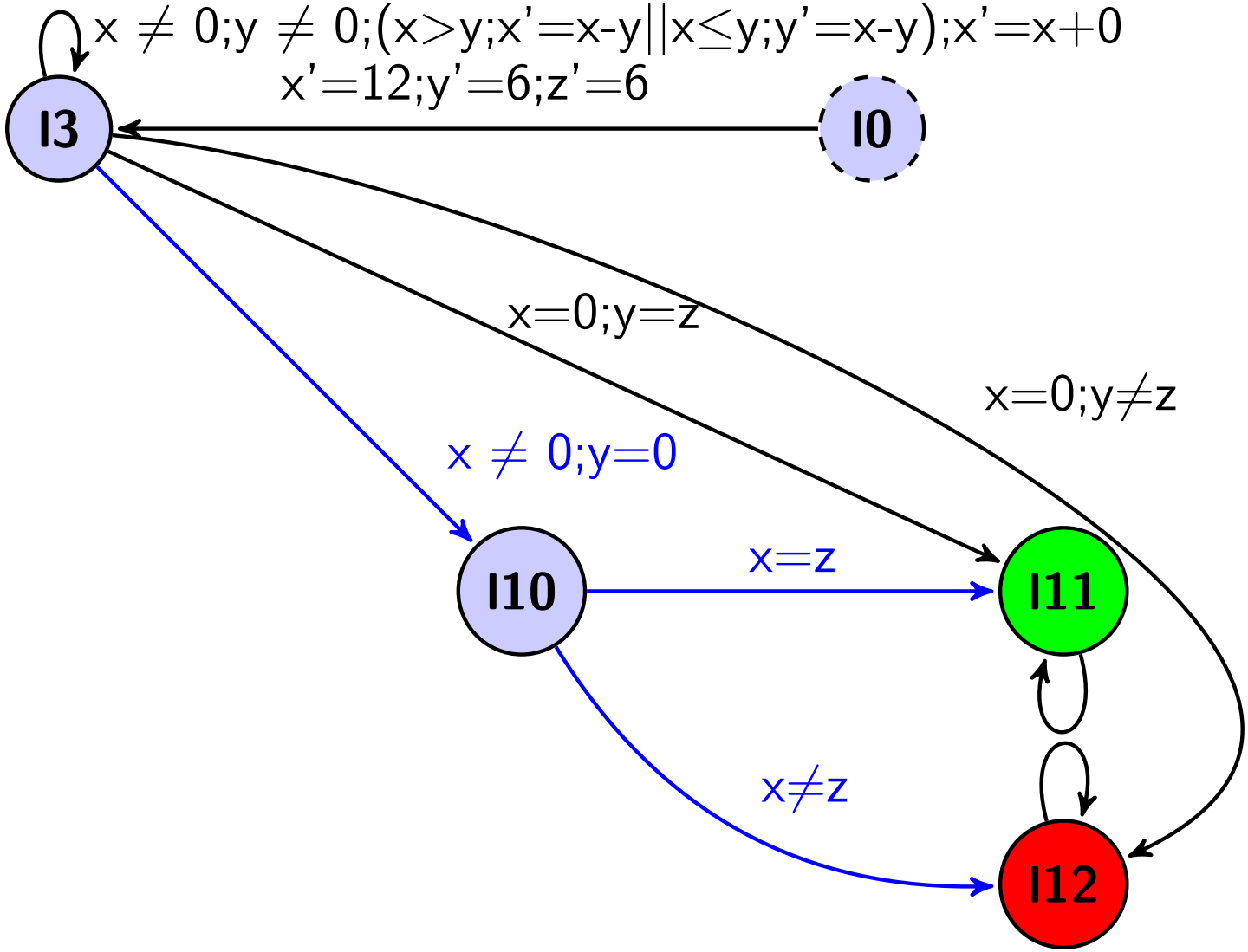
Example


x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y);x'=x+0
x'=12;y'=6;z'=6
l3
l0
x=0;y=z
x=0;y≠z
x ≠ 0;y=0
l10
x=z
l11
x≠z
l12

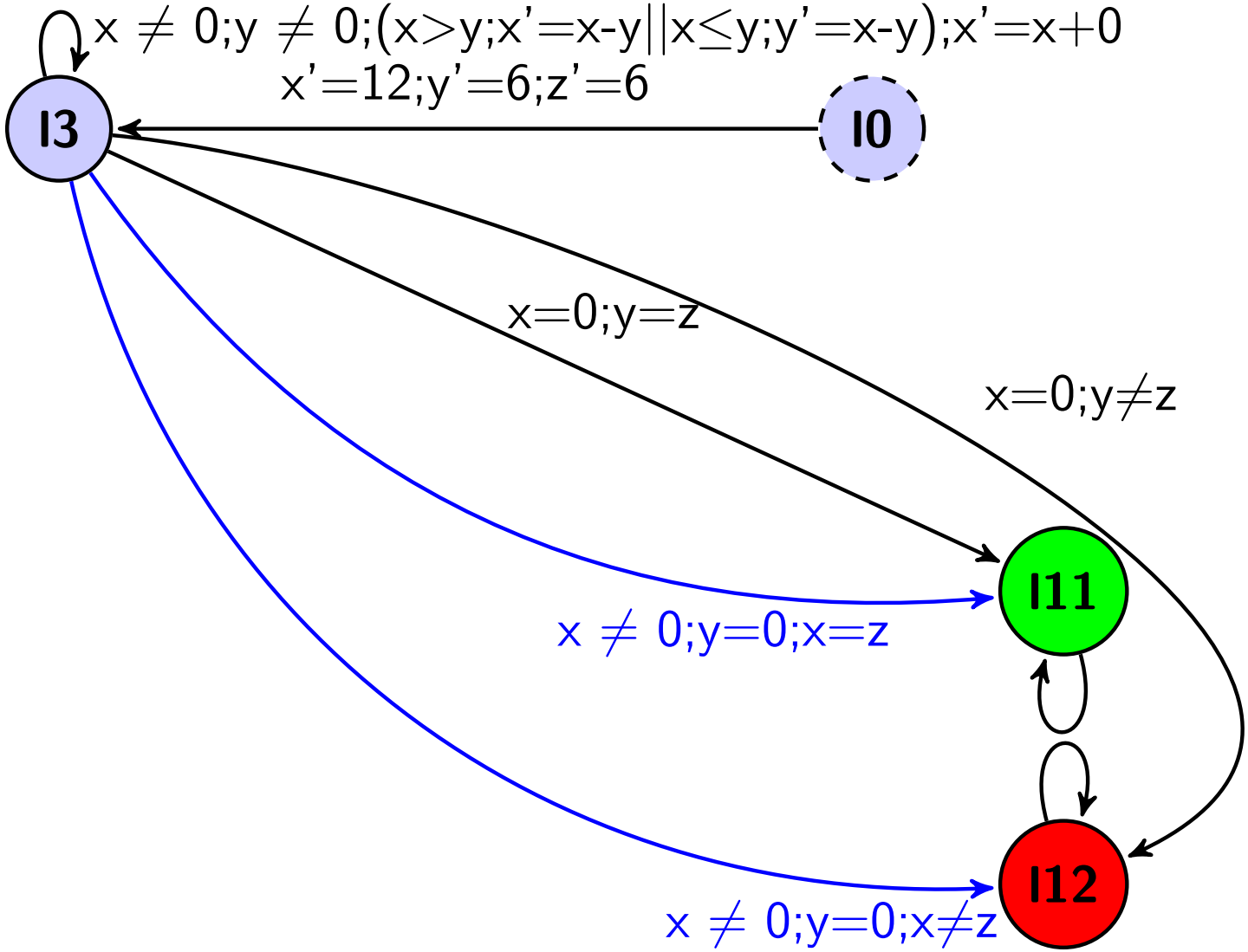
Example


x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y);x'=x+0
x'=12;y'=6;z'=6
l3
l0
x=0;y=z
x=0;y≠z
x ≠ 0;y=0
l10
x=z
l11
x≠z
l12

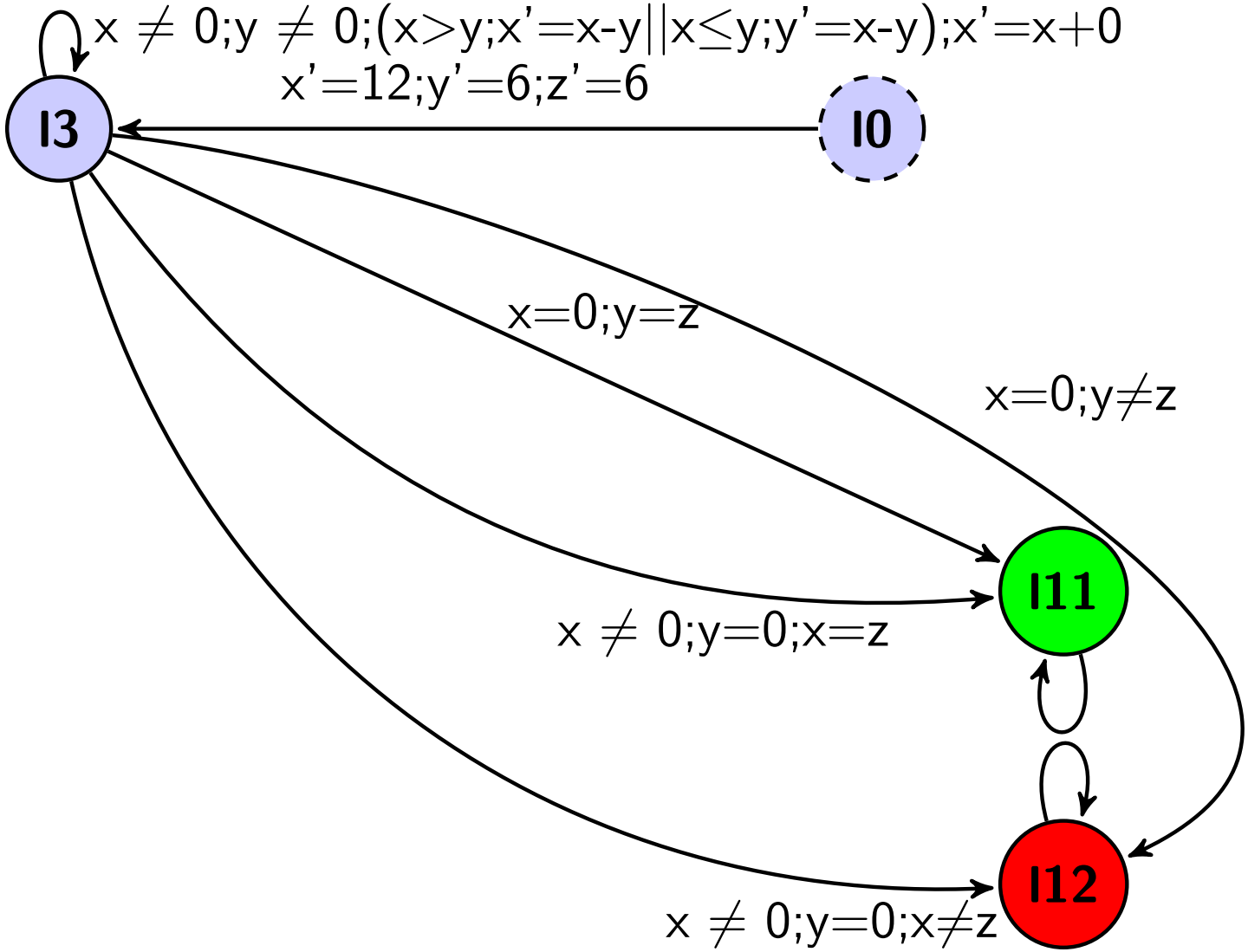
Example


x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y);x'=x+0
x'=12;y'=6;z'=6
l3
l0
x=0;y=z
x=0;y≠z
x ≠ 0;y=0
l10
x=z
l11
x≠z
l12

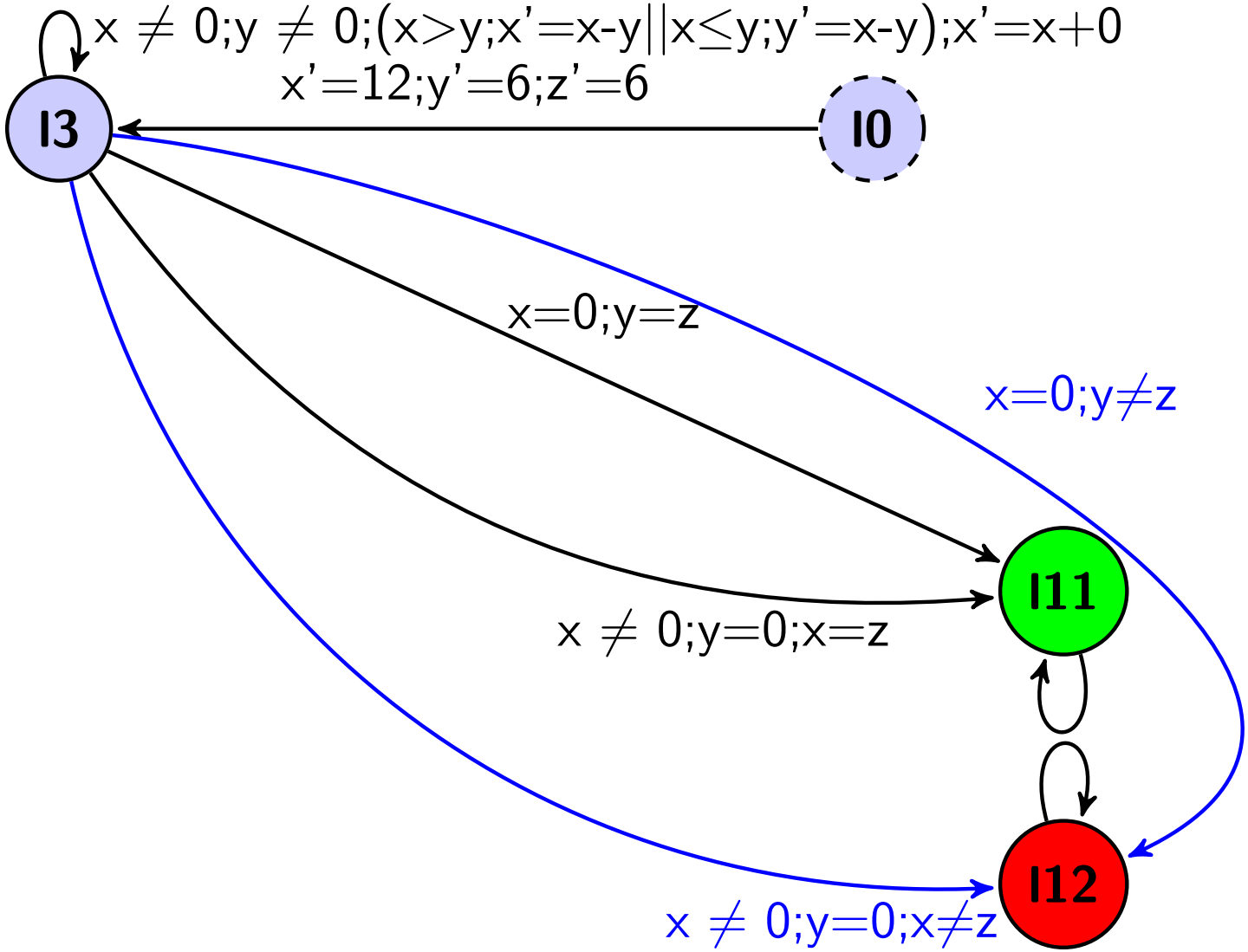
Example


x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y);x'=x+0
x'=12;y'=6;z'=6
l3
l0
x=0;y=z
x=0;y≠z
l11
x ≠ 0;y=0;x=z
l12
x ≠ 0;y=0;x≠z

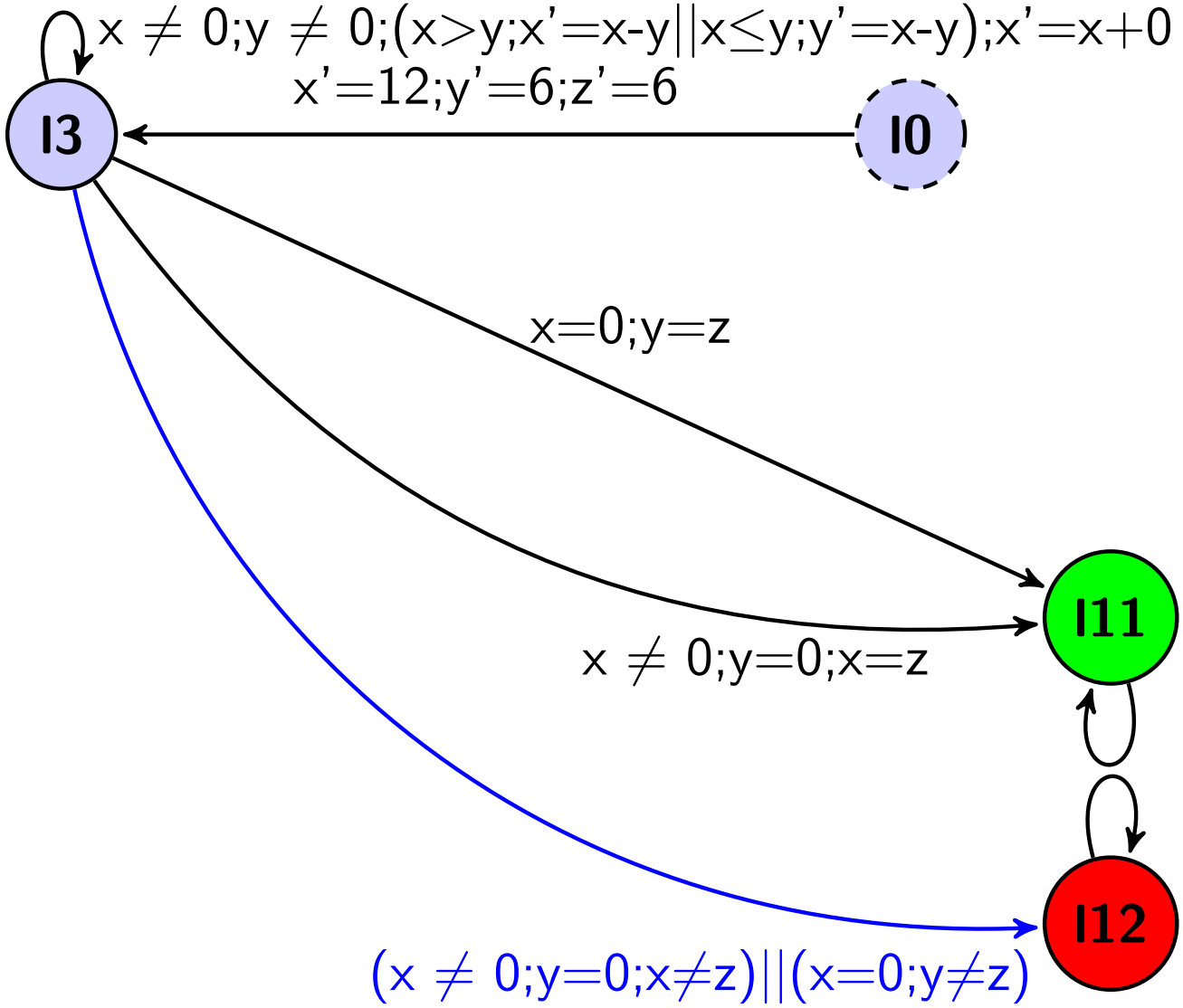
Example


x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y);x'=x+0
x'=12;y'=6;z'=6
l3
l0
x=0;y=z
x=0;y≠z
l11
x ≠ 0;y=0;x=z
l12
x ≠ 0;y=0;x≠z

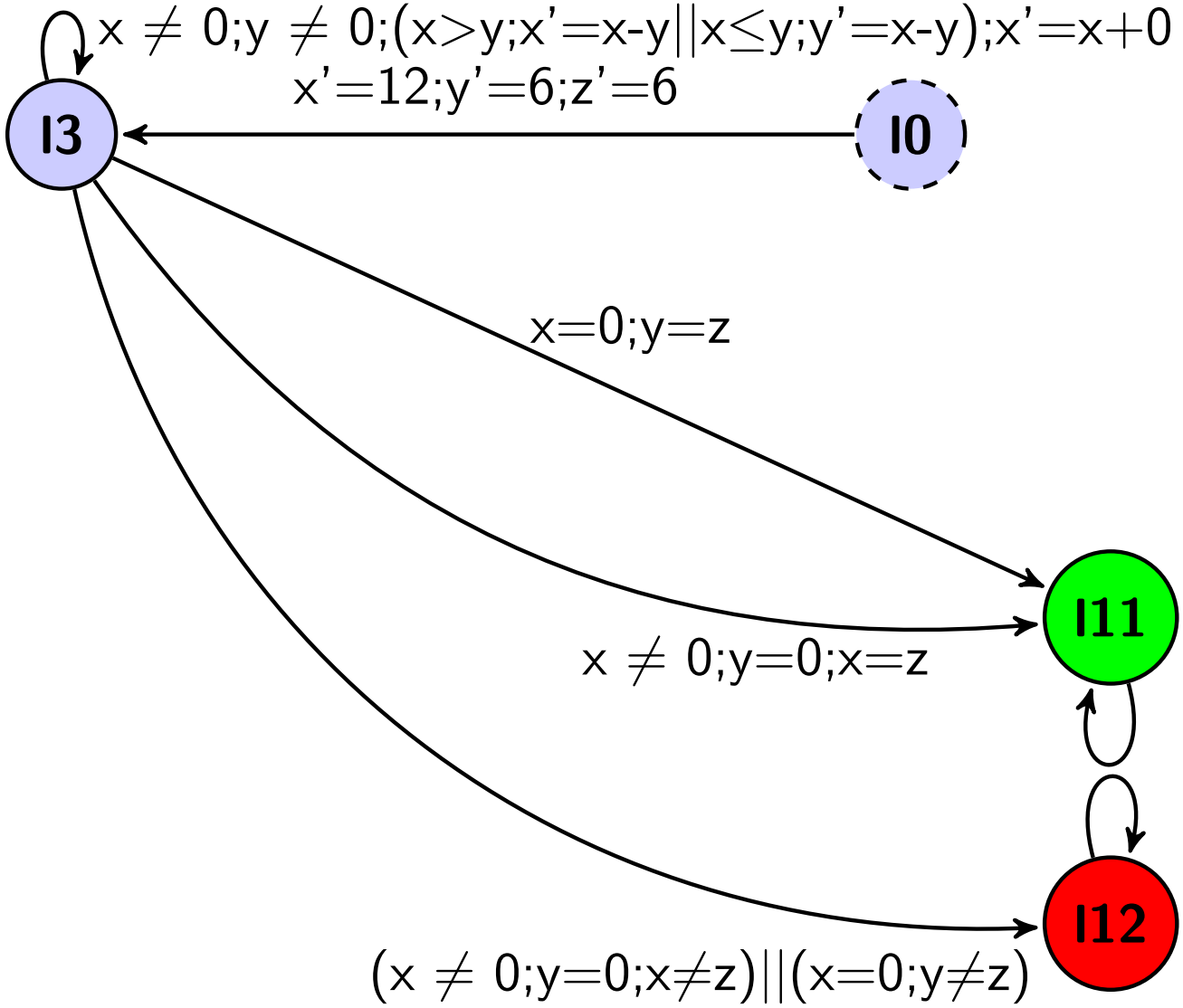
Example


x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y);x'=x+0
x'=12;y'=6;z'=6
l3
l0
x=0;y=z
x=0;y≠z
l11
x ≠ 0;y=0;x=z
l12
x ≠ 0;y=0;x≠z

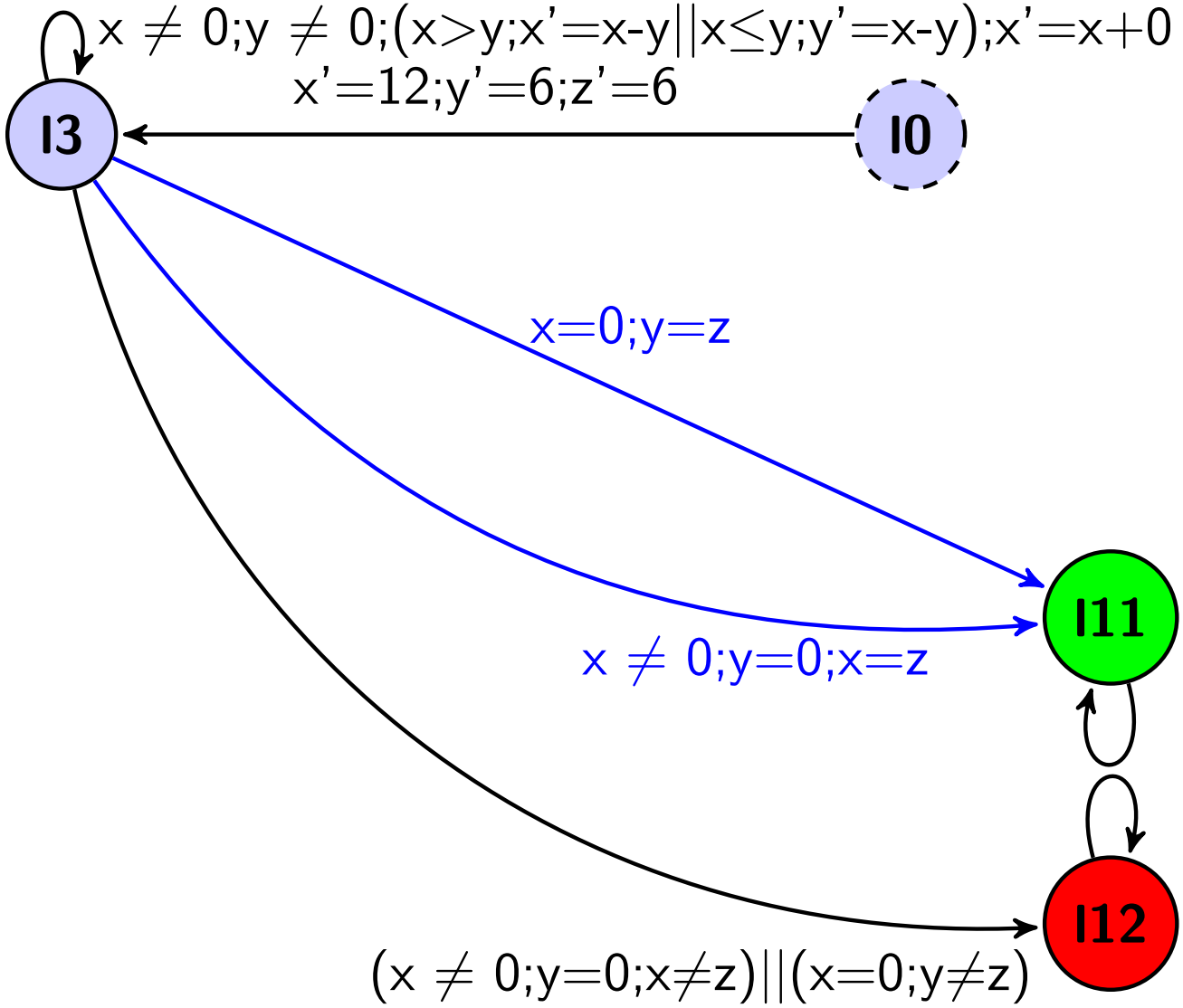
Example


x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y);x'=x+0
x'=12;y'=6;z'=6
l3
l0
x=0;y=z
l11
x ≠ 0;y=0;x=z
l12
(x ≠ 0;y=0;x≠z)||(x=0;y≠z)

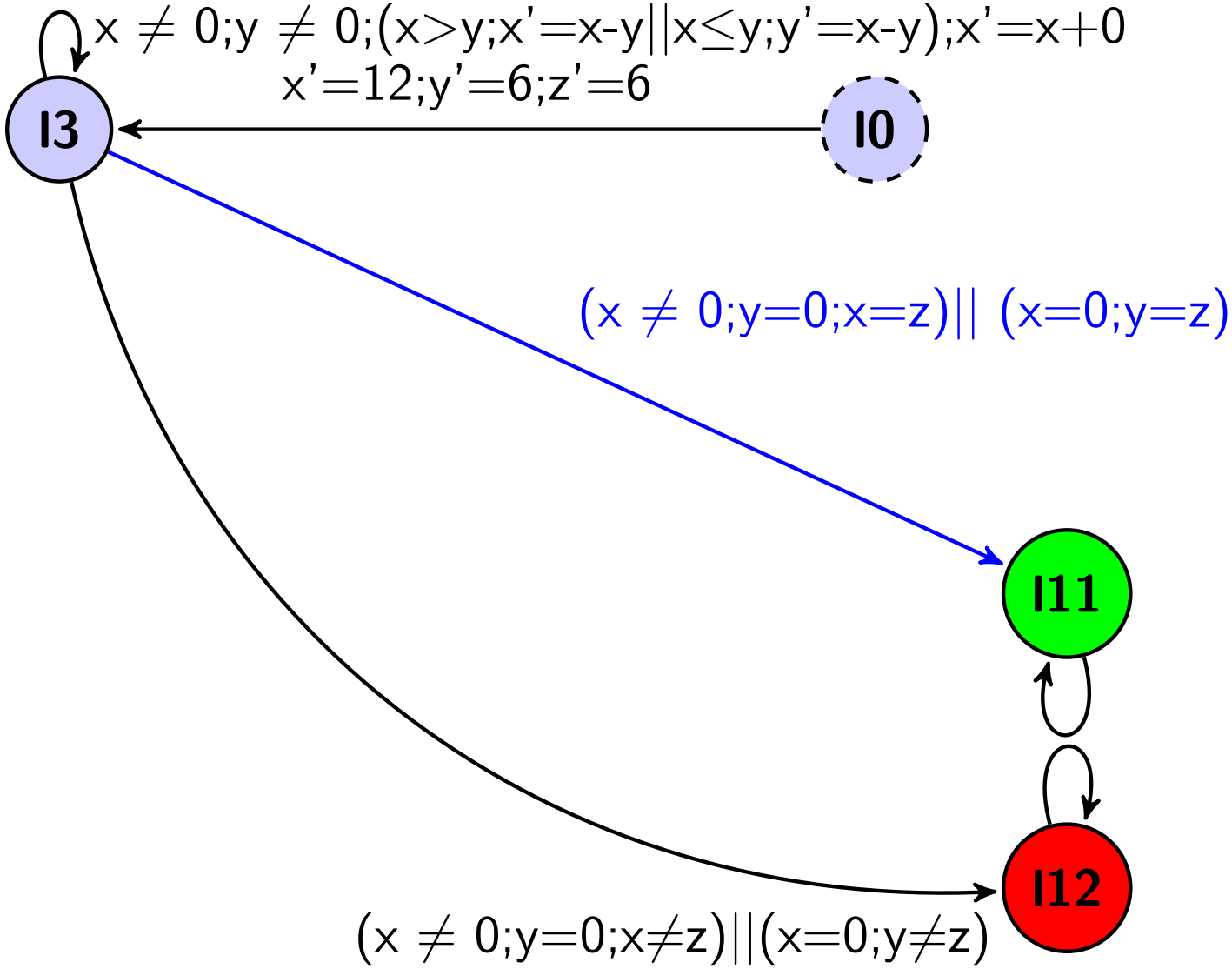
Example


x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y);x'=x+0
x'=12;y'=6;z'=6
l3
l0
x=0;y=z
l11
x ≠ 0;y=0;x=z
l12
(x ≠ 0;y=0;x≠z)||(x=0;y≠z)

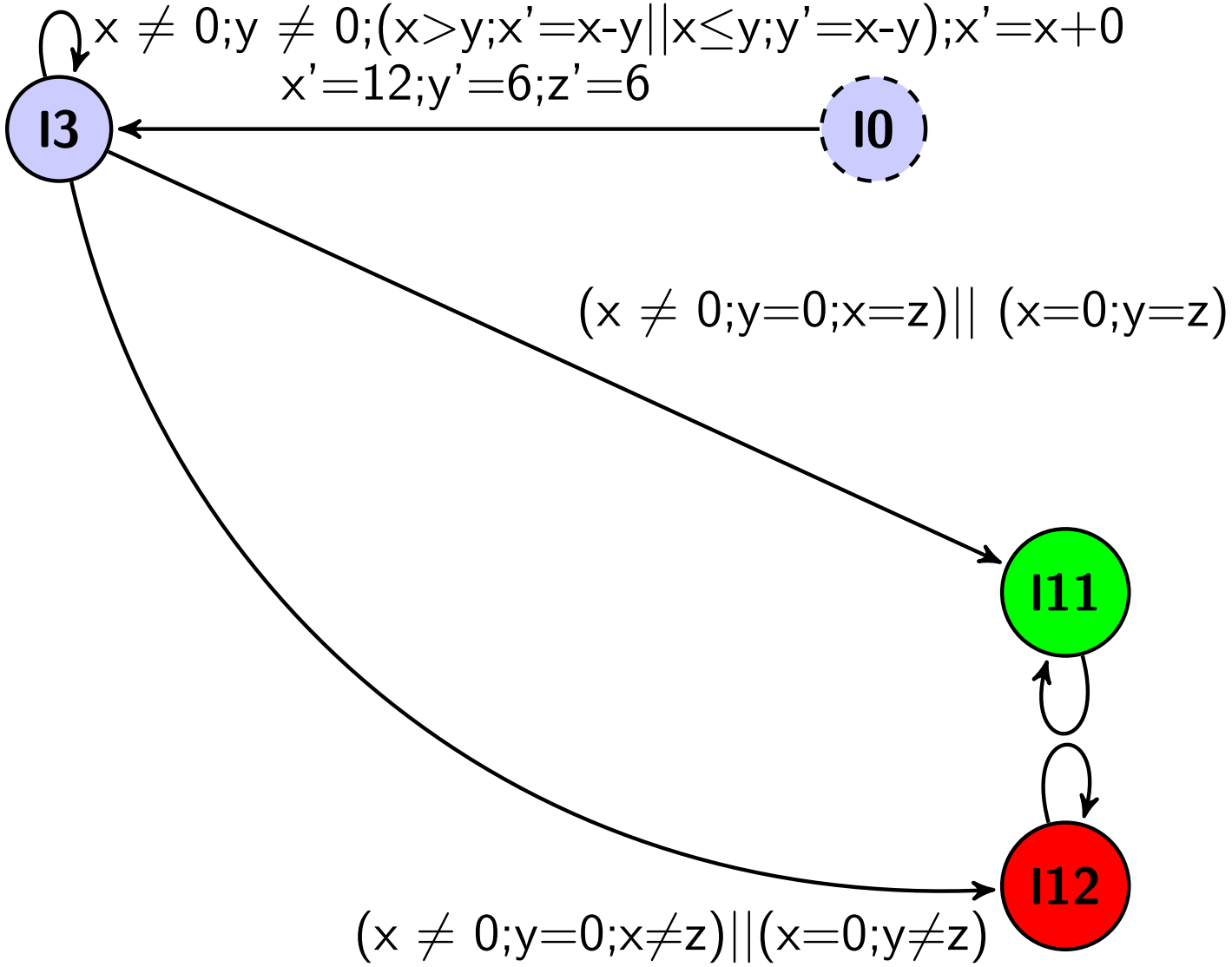
Example


x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y);x'=x+0
x'=12;y'=6;z'=6
l3
l0
x=0;y=z
l11
x ≠ 0;y=0;x=z
l12
(x ≠ 0;y=0;x≠z)||(x=0;y≠z)

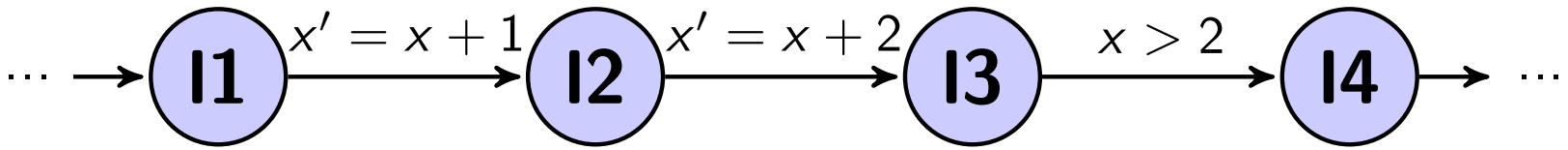
Example


x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y);x'=x+0
x'=12;y'=6;z'=6
l3
l0
(x ≠ 0;y=0;x=z)|| (x=0;y=z)
l11
l12
(x ≠ 0;y=0;x≠z)||(x=0;y≠z)

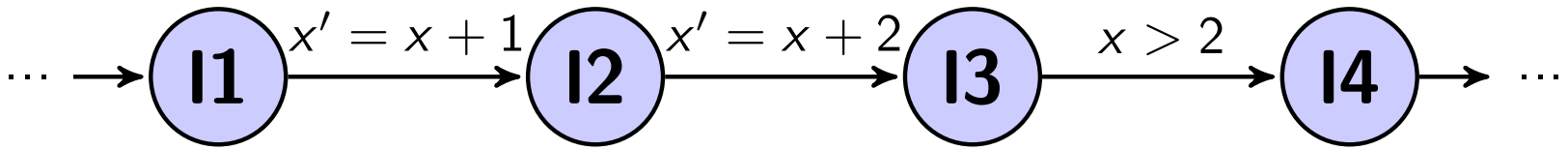
Example


x ≠ 0;y ≠ 0;(x>y;x'=x-y||x≤y;y'=x-y);x'=x+0
x'=12;y'=6;z'=6
l3
l0
(x ≠ 0;y=0;x=z)|| (x=0;y=z)
l11
l12
(x ≠ 0;y=0;x≠z)||(x=0;y≠z)

Variable Substitution

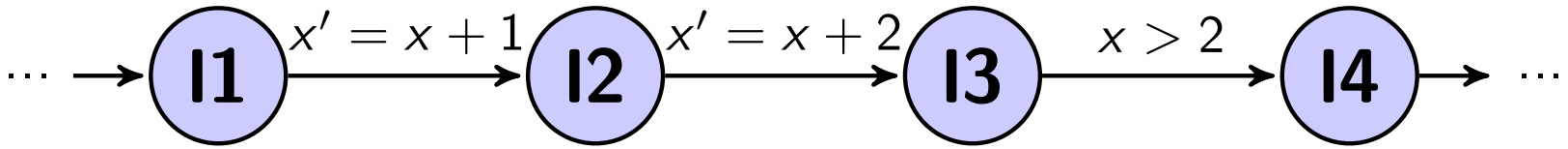


Variable Substitution

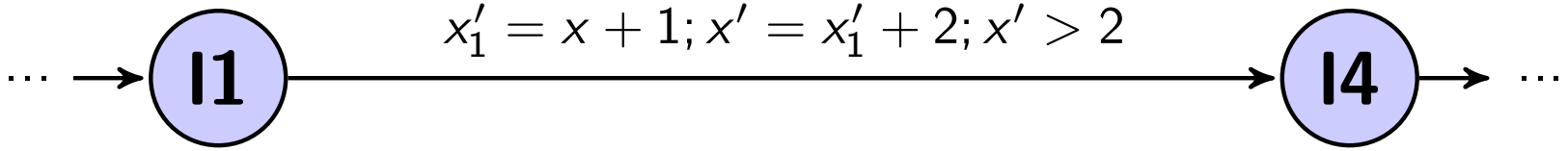


Model Checker interprets an assignment as $f : VAR \rightarrow VAR'$.

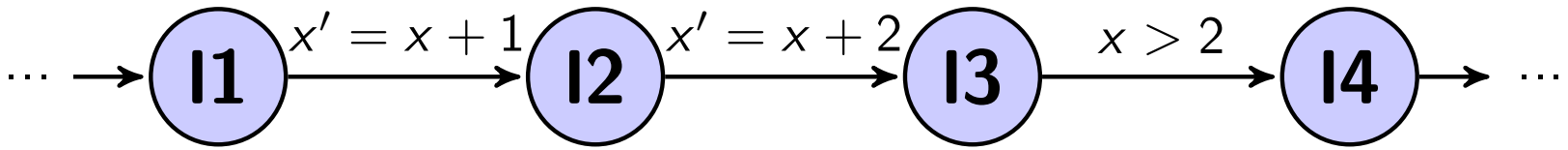
Variable Substitution



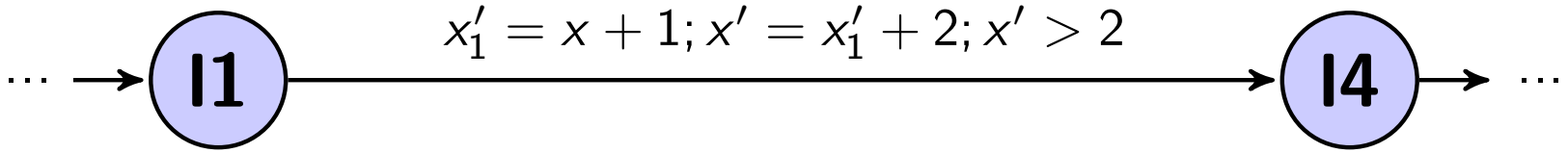
Model Checker interprets an assignment as $f : VAR \rightarrow VAR'$.



Variable Substitution

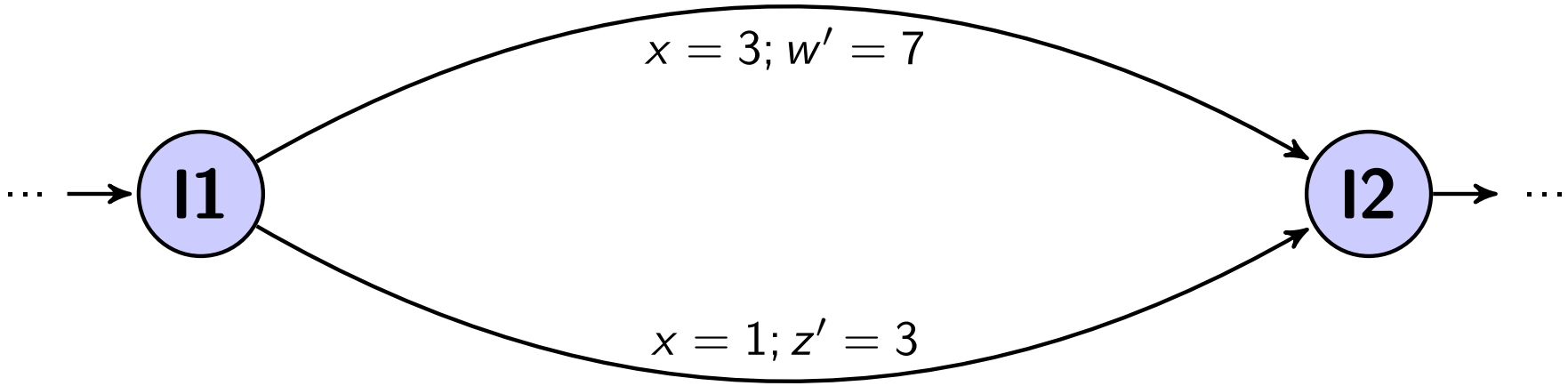


Model Checker interprets an assignment as $f : VAR \rightarrow VAR'$.

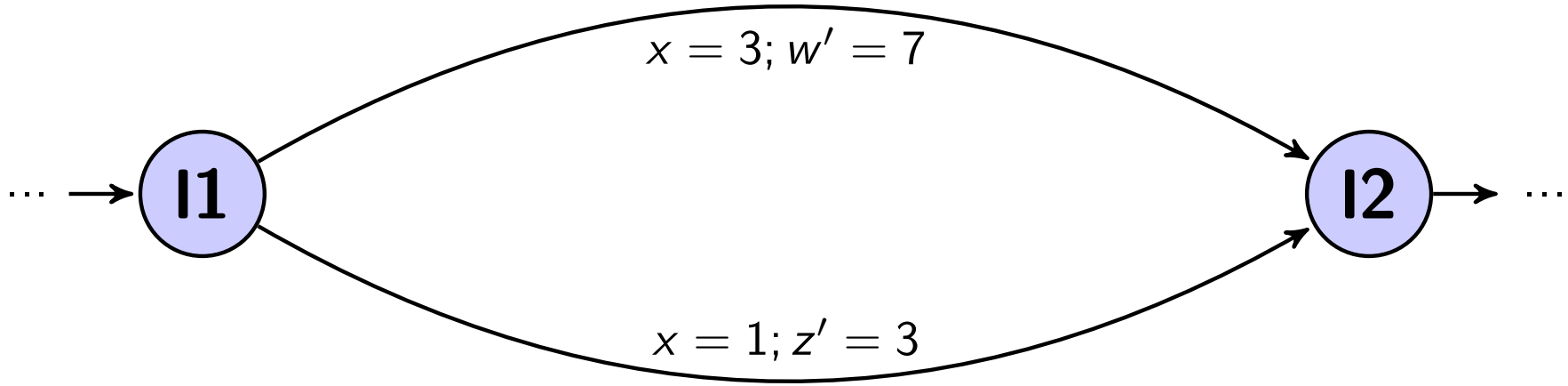


A sequential excecution of assignments looks as follow: $f_2(f_1(x))$

Variable Propagation

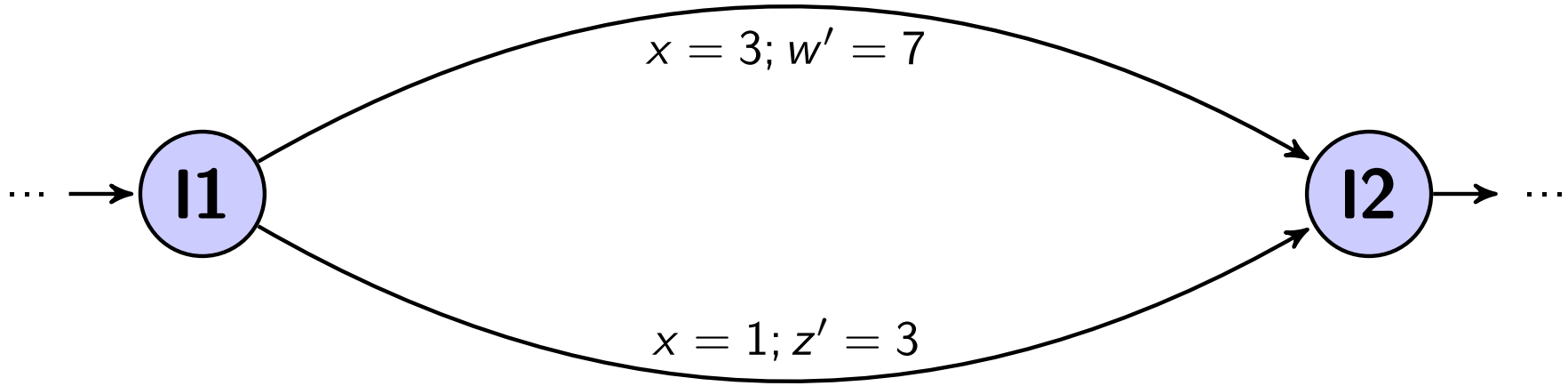


Variable Propagation

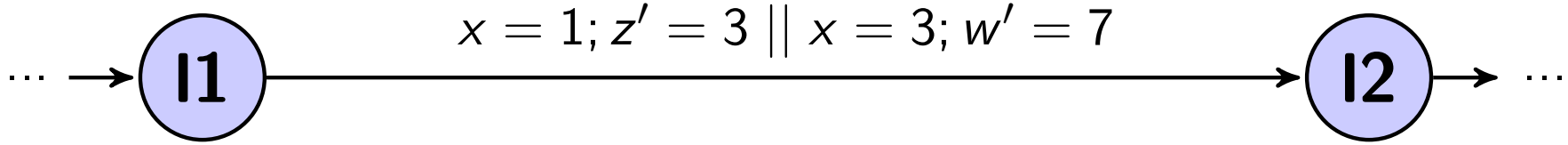


Ensure that model checker does not skip a variable assignment

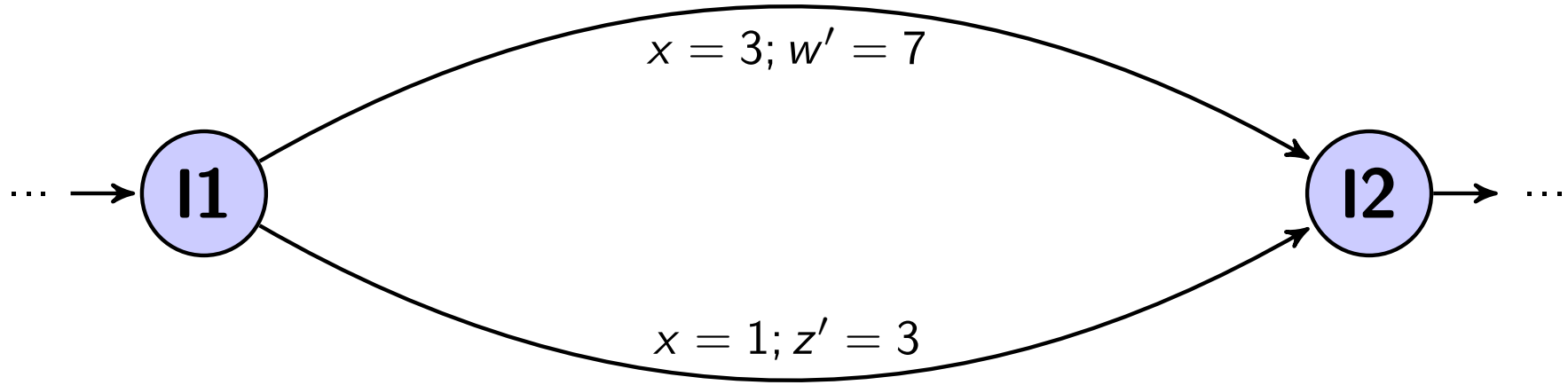
Variable Propagation



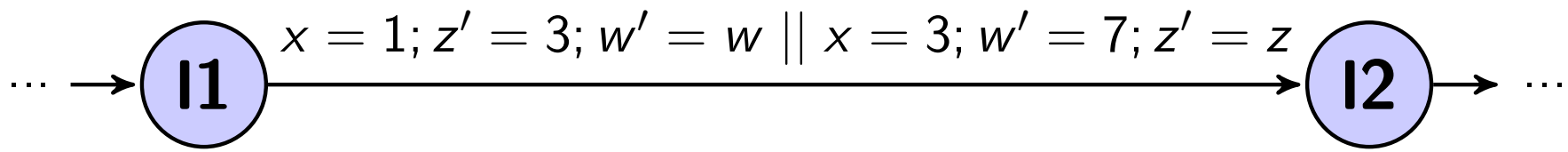
Ensure that model checker does not skip a variable assignment



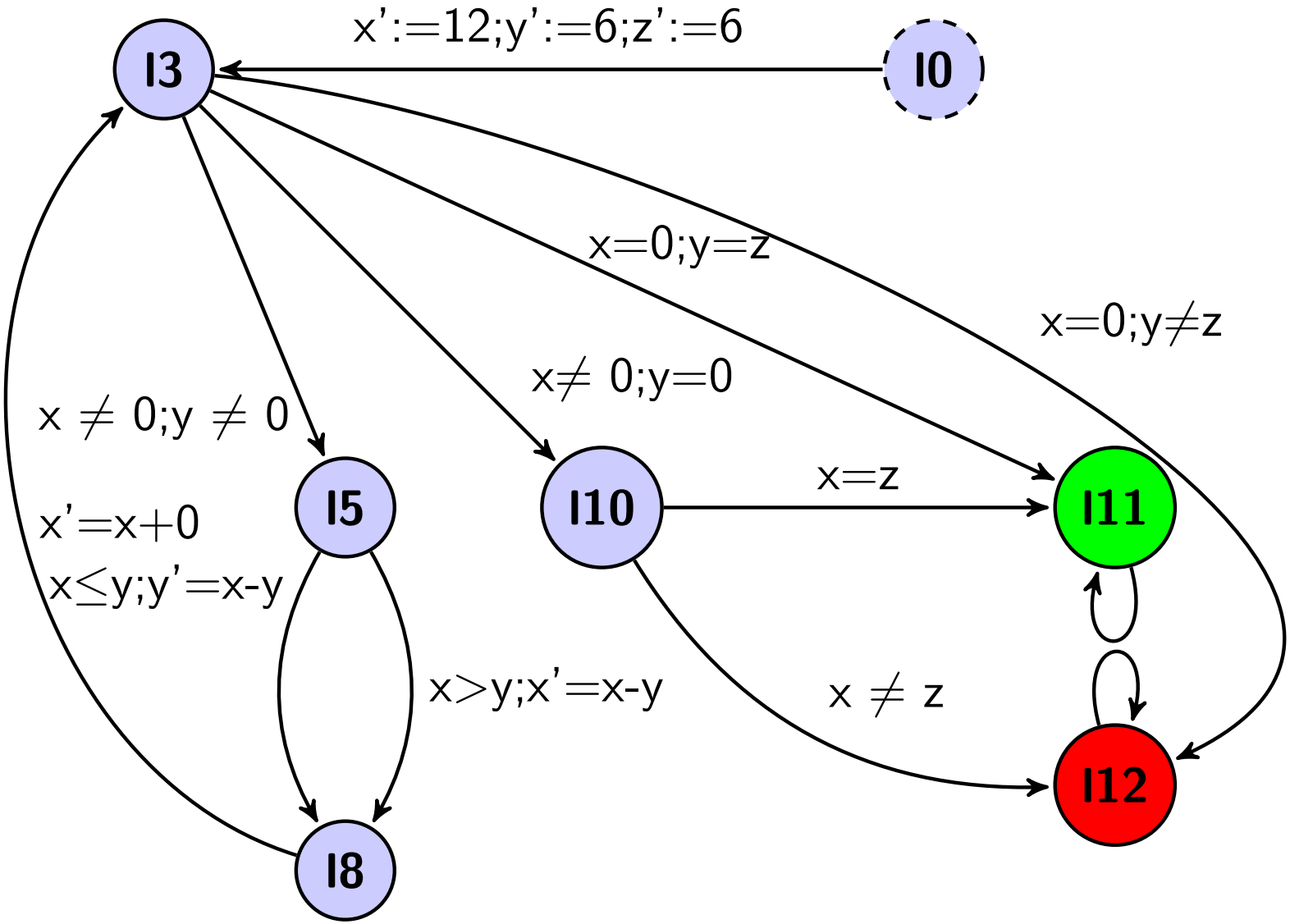
Variable Propagation



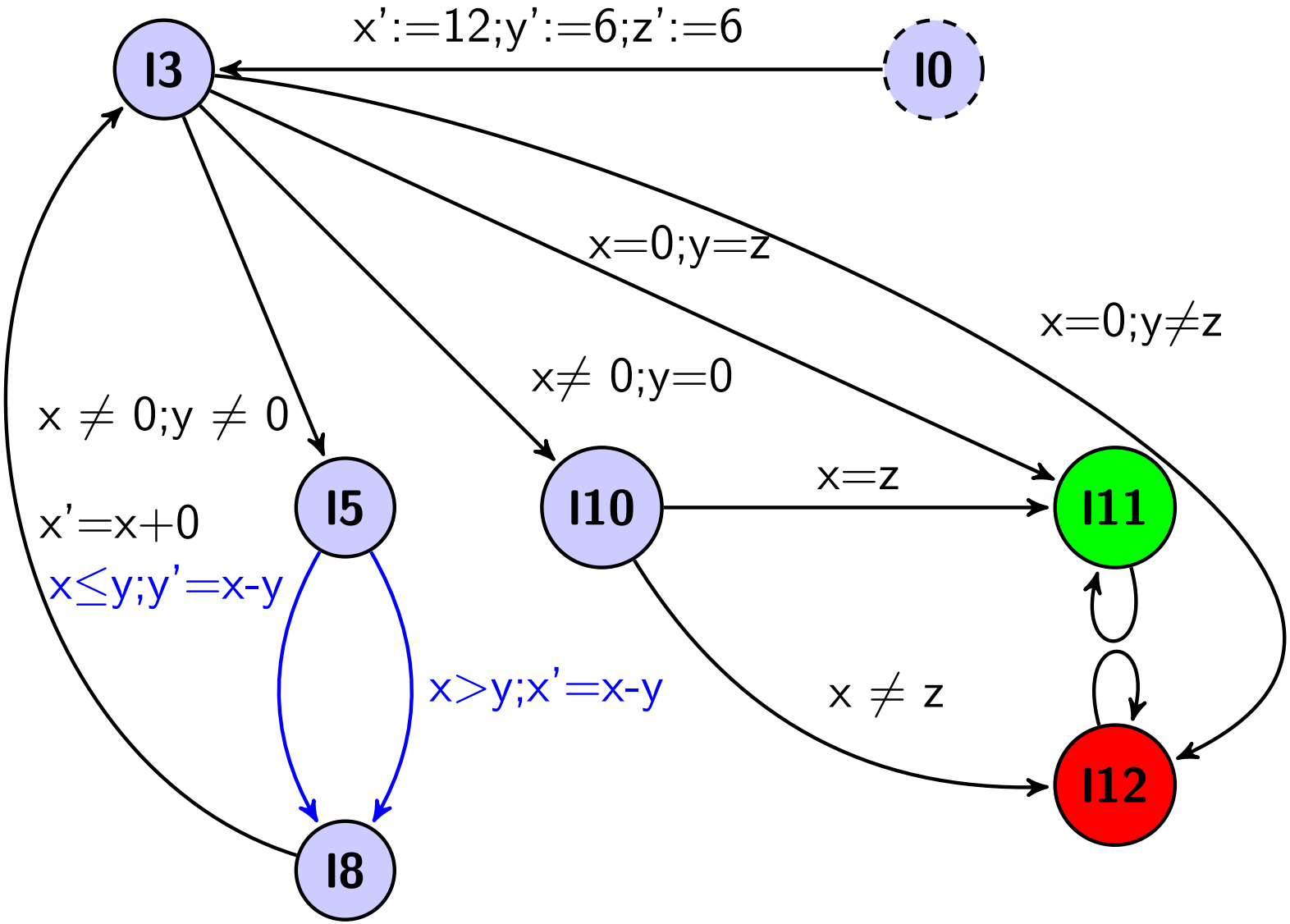
Ensure that model checker does not skip a variable assignment



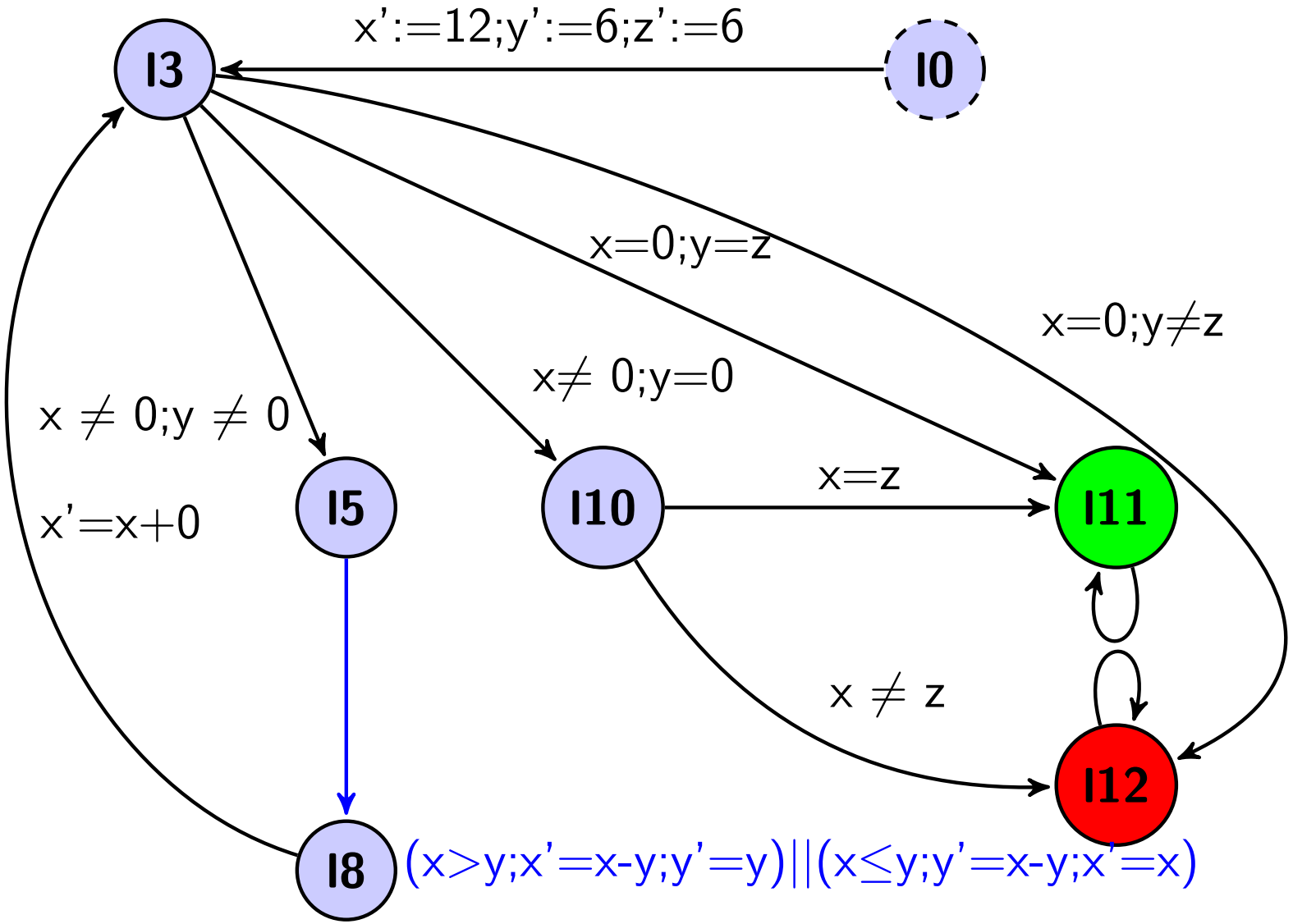
Example


l3
x':=12;y':=6;z':=6
l0
x=0;y=z
x=0;y≠z
x≠ 0;y=0
x ≠ 0;y ≠ 0
l5
l10
x=z
l11
x'=x+0
x≤y;y'=x-y
x>y;x'=x-y
x ≠ z
l12
l8

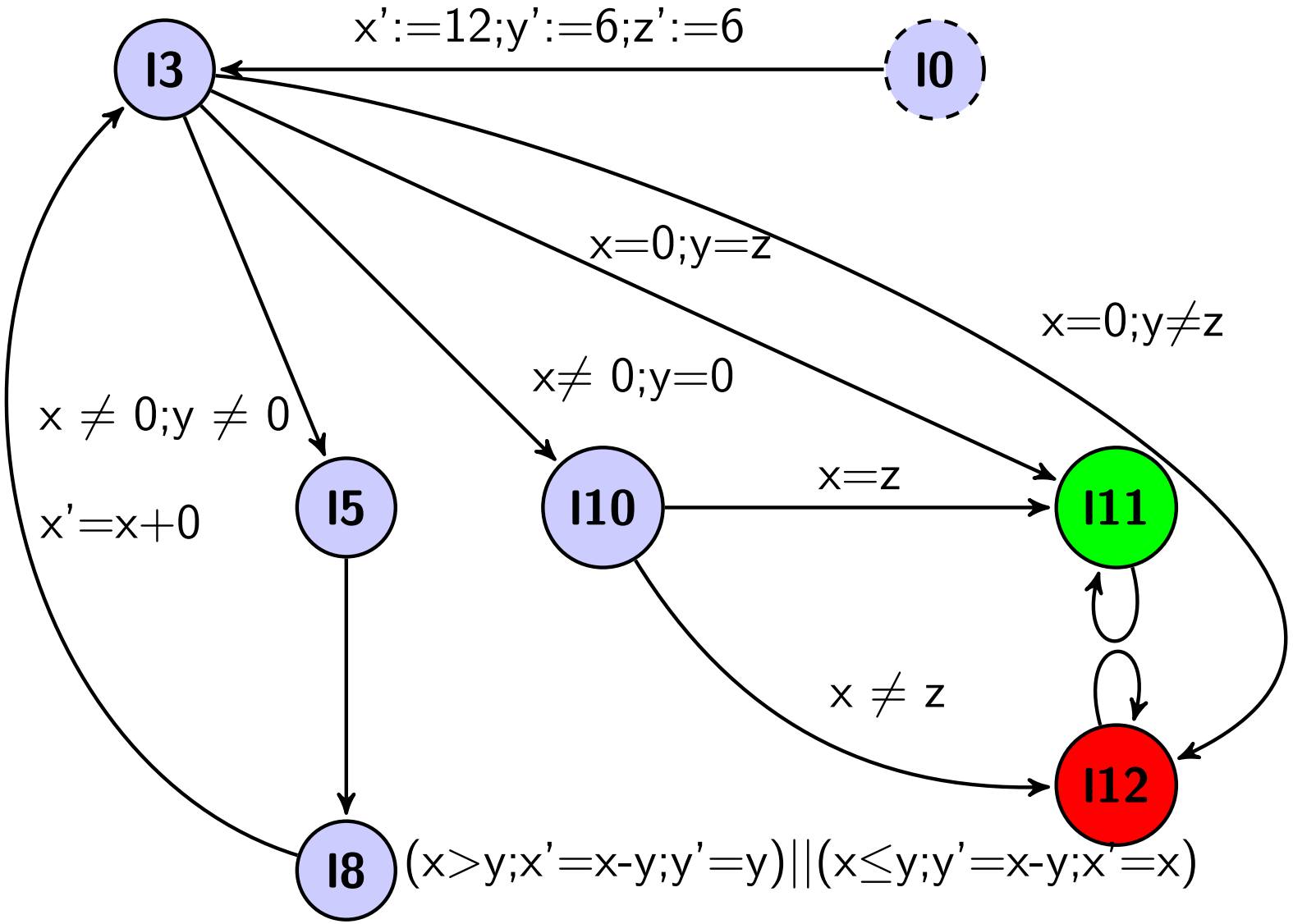
Example


l3
x':=12;y':=6;z':=6
l0
x=0;y=z
x=0;y≠z
x≠ 0;y=0
x ≠ 0;y ≠ 0
l5
l10
x=z
l11
x'=x+0
x≤y;y'=x-y
x>y;x'=x-y
x ≠ z
l12
l8

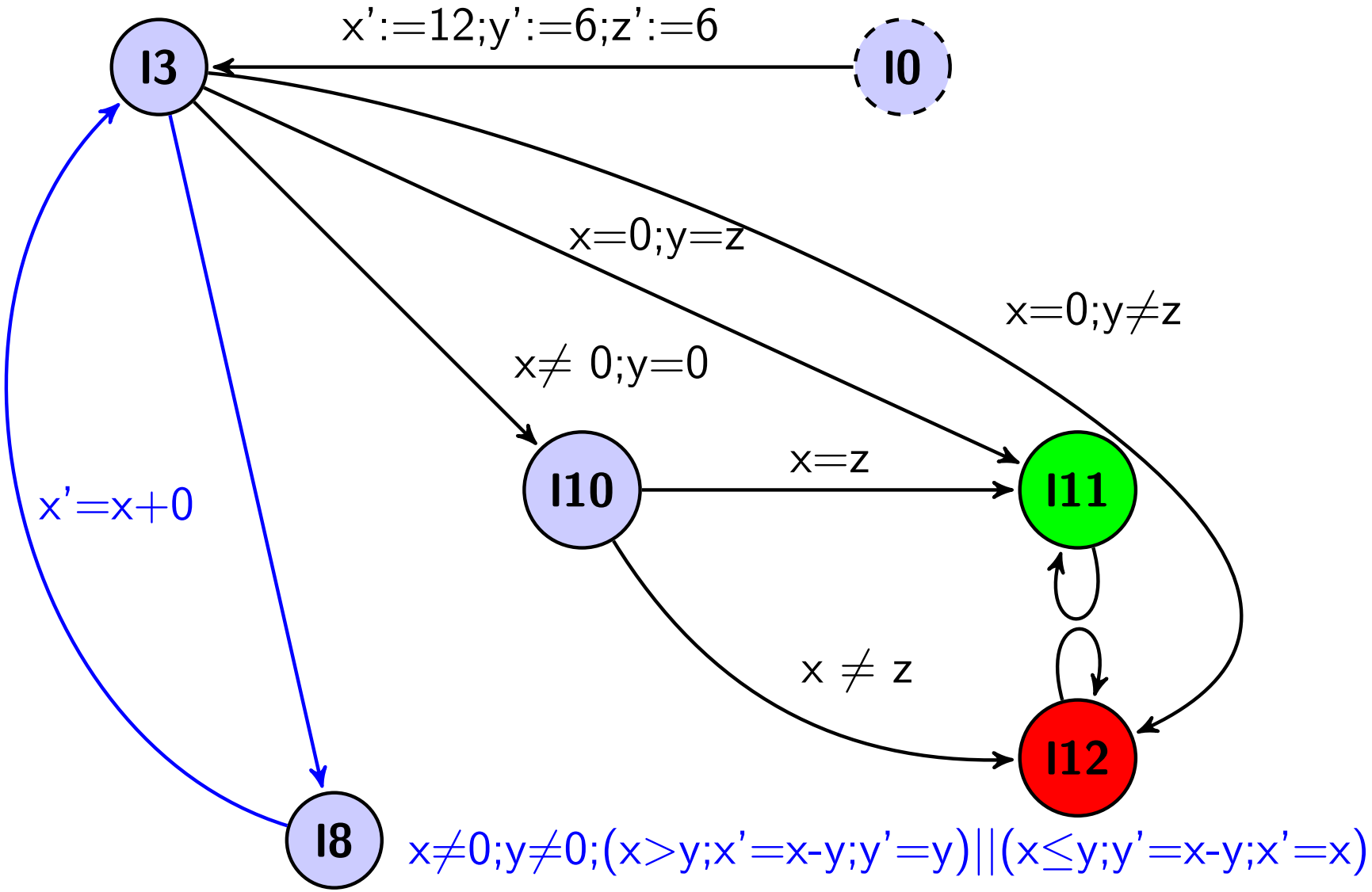
Example


x':=12;y':=6;z':=6
l3
l0
x=0;y=z
x=0;y≠z
x≠ 0;y=0
x ≠ 0;y ≠ 0
x'=x+0
l5
l10
x=z
l11
x ≠ z
l12
l8
(x>y;x'=x-y;y'=y)||(x≤y;y'=x-y;x'=x)

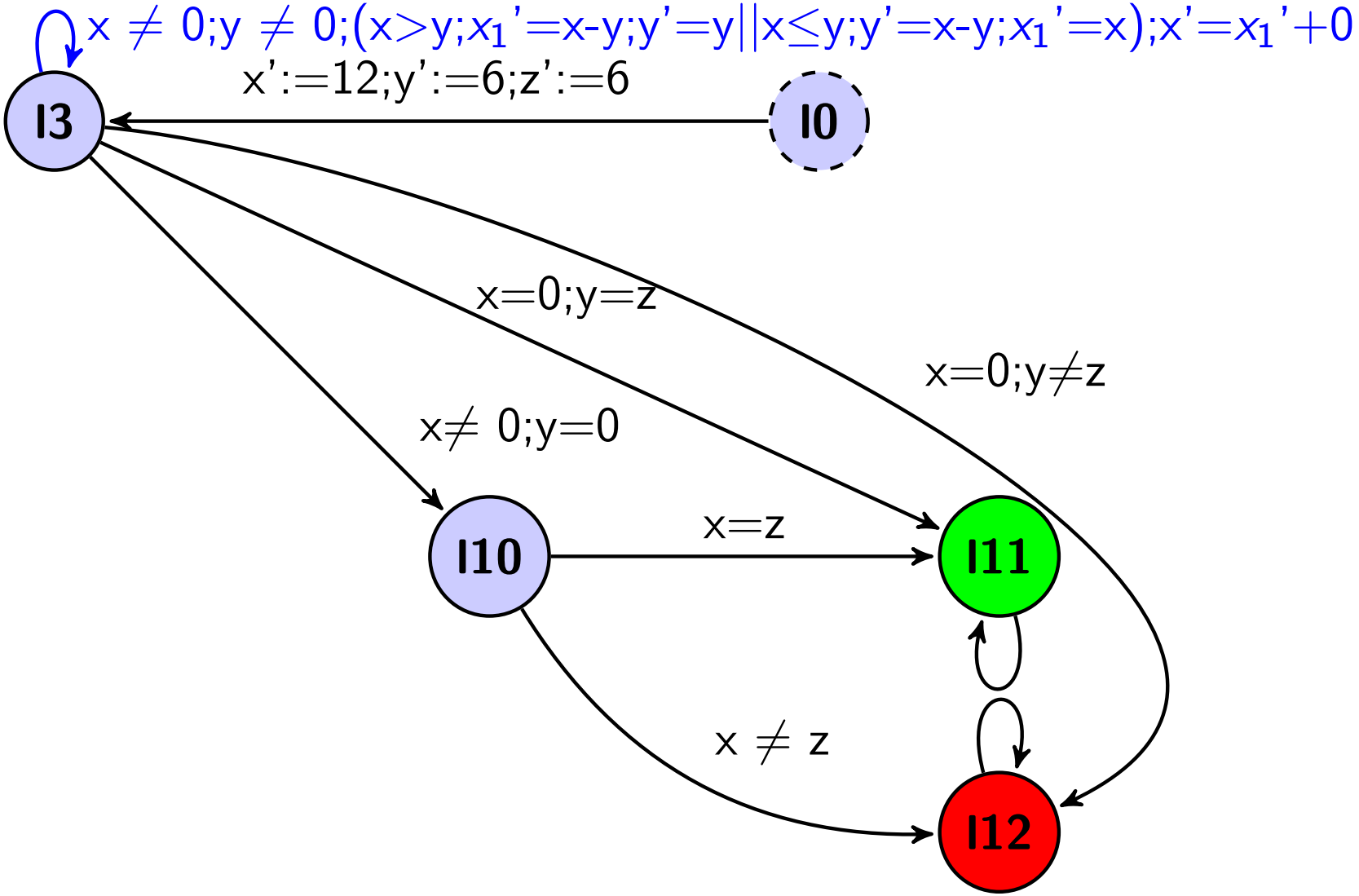
Example


l3
x':=12;y':=6;z':=6
l0
x=0;y=z
x=0;y≠z
x≠ 0;y=0
x ≠ 0;y ≠ 0
x'=x+0
l5
l10
x=z
l11
x ≠ z
l12
l8
(x>y;x'=x-y;y'=y)||(x≤y;y'=x-y;x'=x)

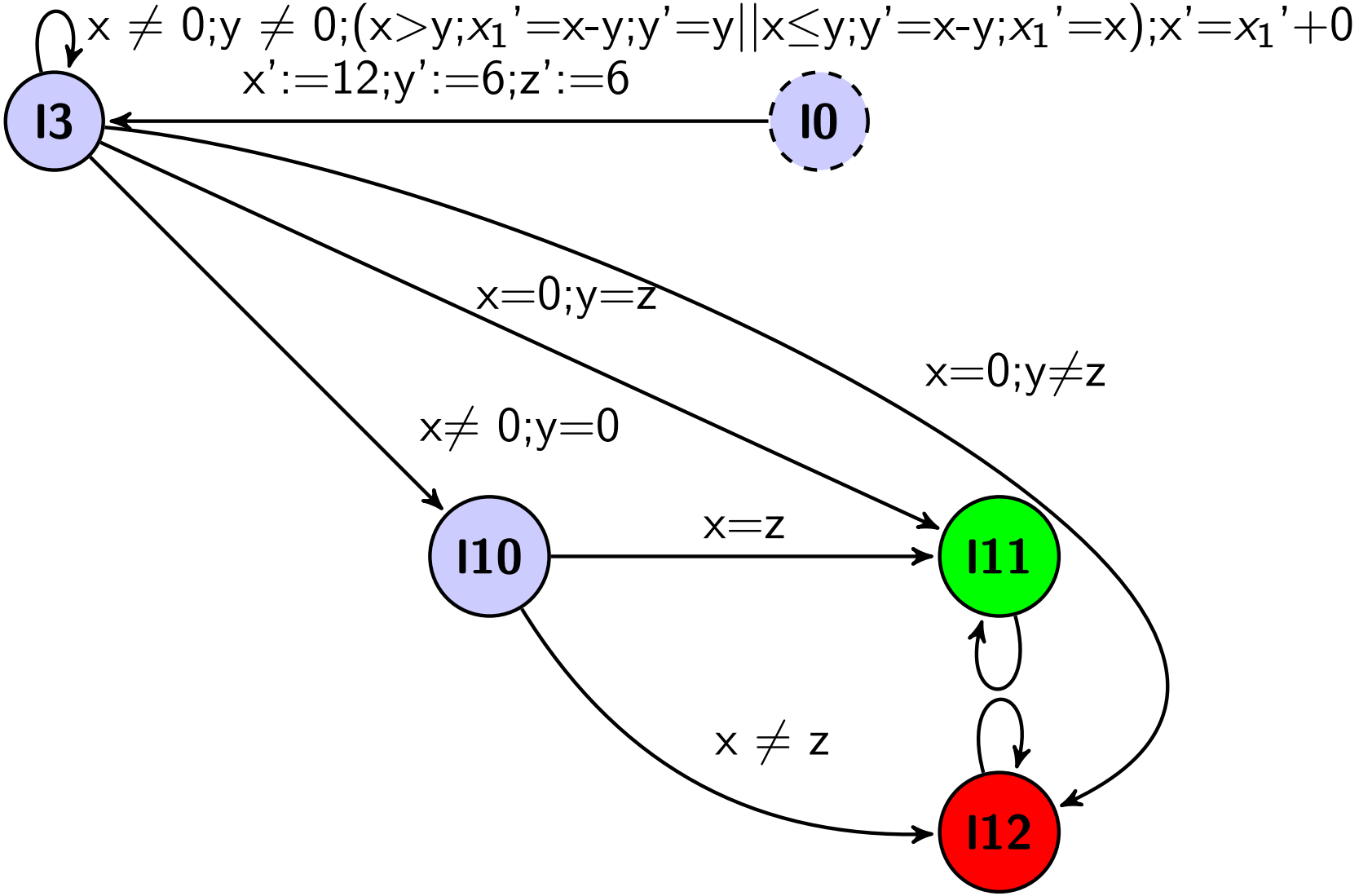
Example


l3
l0
x':=12;y':=6;z':=6
x=0;y=z
x=0;y≠z
x≠ 0;y=0
l10
x=z
l11
x'=x+0
x ≠ z
l12
l8
x≠0;y≠0;(x>y;x'=x-y;y'=y)||(x≤y;y'=x-y;x'=x)

Example


x ≠ 0;y ≠ 0;(x>y;x1'=x-y;y'=y||x≤y;y'=x-y;x1'=x);x'=x1'+0
x':=12;y':=6;z':=6
l3
l0
x=0;y=z
x=0;y≠z
x≠ 0;y=0
l10
x=z
l11
x ≠ z
l12

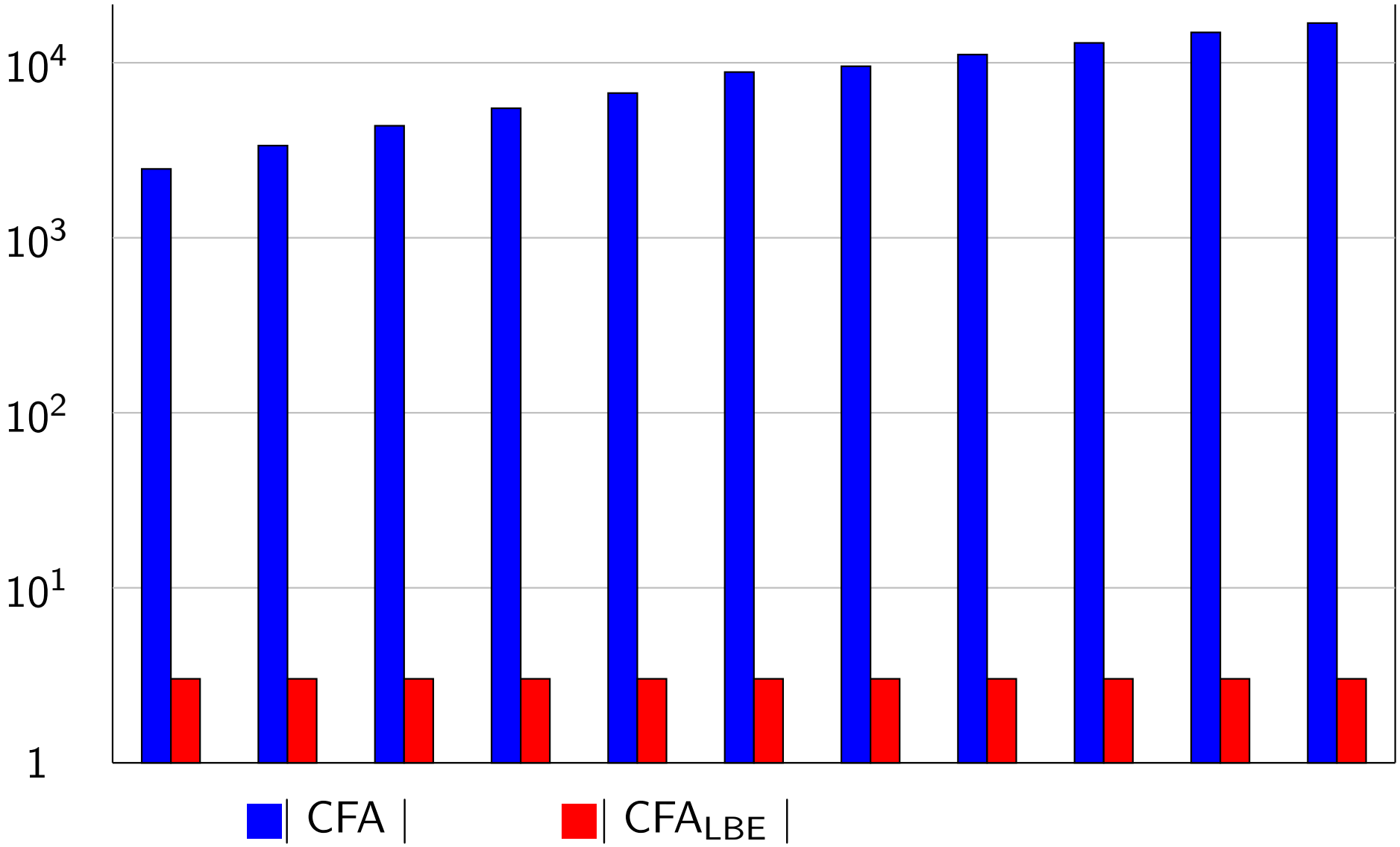
Example


x ≠ 0;y ≠ 0;(x>y;x1'=x-y;y'=y||x≤y;y'=x-y;x1'=x);x'=x1'+0
x':=12;y':=6;z':=6
l3
l0
x=0;y=z
x=0;y≠z
x≠ 0;y=0
l10
x=z
l11
x ≠ z
l12

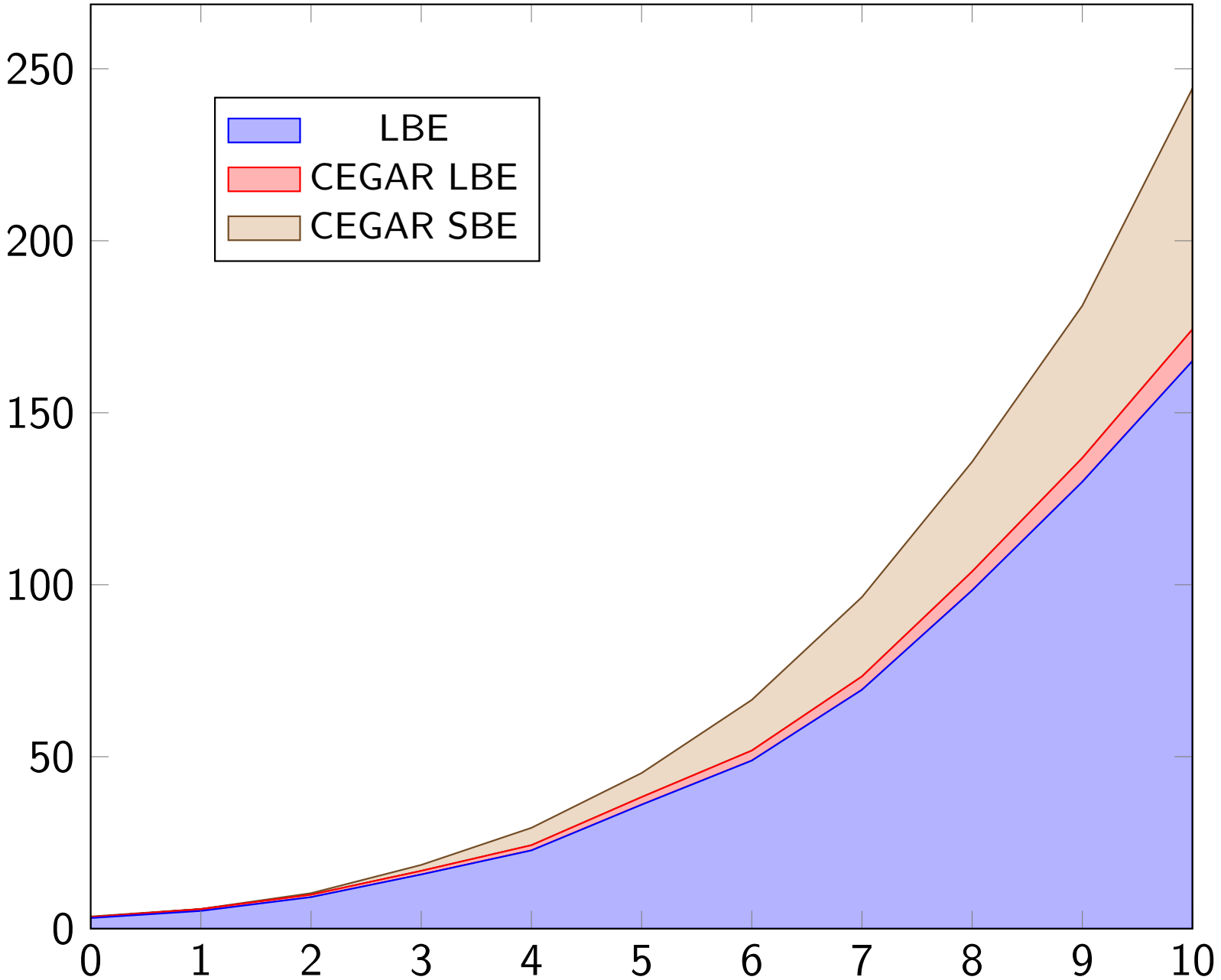
Outline

1. Motivation
2. Background
   - Program and Control-Flow Automaton
   - Model Checker
3. Large-Block Encoding
   - Summarization
   - Post-processing
4. Evaluation
5. Conclusion

CFA Reduction



Runtime Analysis (CEGAR)

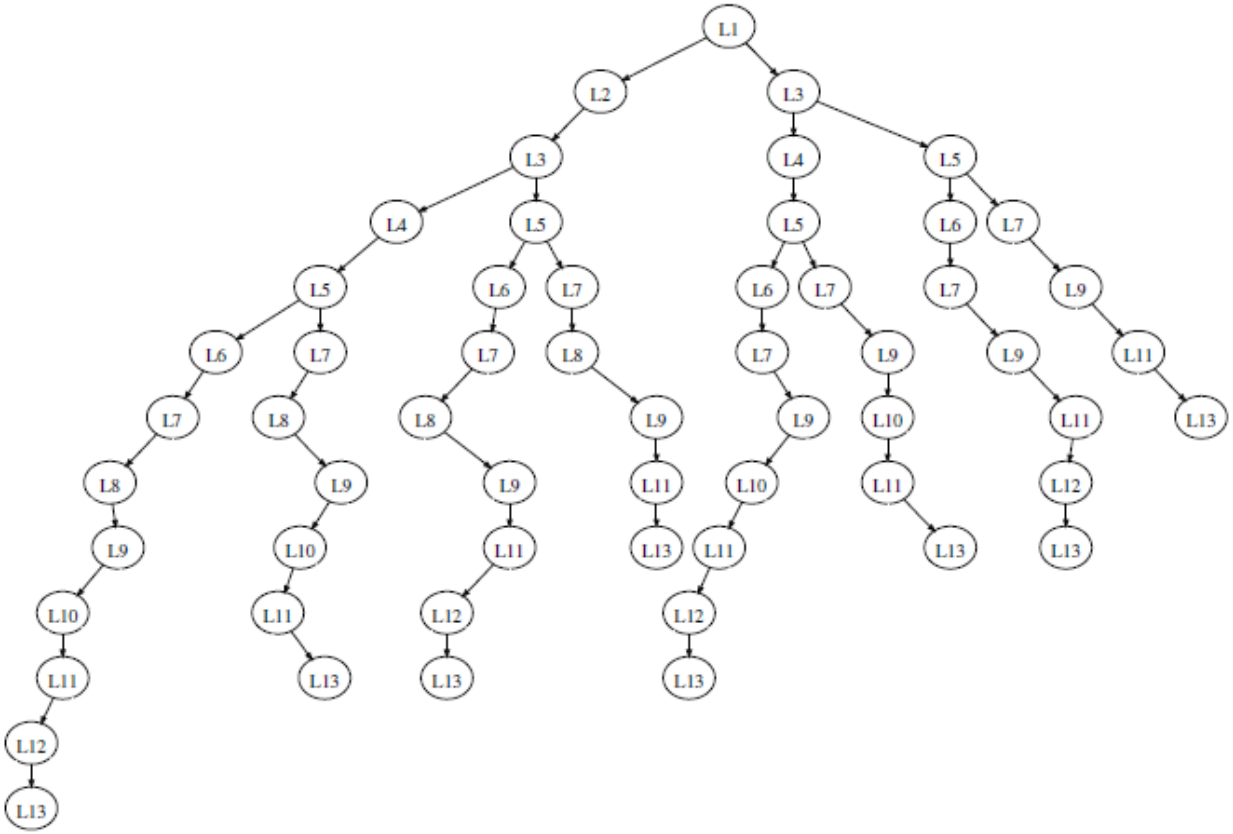


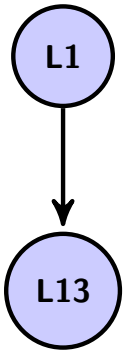
Outline

1. Motivation
2. Background
   - Program and Control-Flow Automaton
   - Model Checker
3. Large-Block Encoding
   - Summarization
   - Post-processing
4. Evaluation
5. Conclusion

Conclusion

- Factor of minimization up to exponentially
- CFA can be summarized in polynomial time





Questions?