# Feedback and Paradox

Ferenc András
ORCID iD: 0009-0002-9995-8173

January 29, 2025

Keywords: paradox, liar paradox, truth problem, finite automaton, logical circuits

## Abstract

Every truth function corresponds to an isomorphic digital circuit. Consequently, the logical structure of every proposition can be presented within the range of propositional logic as an equivalent digital circuit. Provided that the logical values 'true' and 'false' correspond to the 'high' and 'low' voltage levels, the output of a circuit being equivalent with contradiction is always low level for every input state, whereas the output of a circuit corresponding to a tautology is always high level, irrespective of the input states. On the other hand, the remaining propositions correspond to circuits whose output is high-level if and only if certain atomic components of the proposition are true, and the rest of the atomic components are false. The inputs equivalent to the atomic propositions are either high or low level. However, what logical circuits are equivalent to a circulating statement or argumentation?

The propositions are true or false irrespective of time, whereas the voltage level of the circuits can change over time. To be more precise, we can say that the input levels of the circuits are high or low depending on whether we evaluate the atomic formulae of the formula expressing the logical structure of the proposition as true or false. The voltage level of the output of the circuit corresponds to the truth value of the formula. I will call 'combinational automaton' the digital circuit that may model the formulae of propositional logic.

Formulae connected with truth functions yield formulae again. Although there are always corresponding automata for them, the situation is not simple in this case. We do not always get combined automata joined to each other, and in some cases, it is also possible that we do not get an automaton–an operating machine or circuit–at all.

There are digital circuits whose output is not a function of their input. The range of automata is wider than that of combinational automata. It includes machines whose input states do not unambiguously determine the output states; that is, the output is not a function of the input. This is because the circuit has feedback. Most digital circuits belong to this latter group. I will call them 'sequential automata'.

The question arises whether there is a logical structure of a circulating statement that corresponds to such a sequential automaton (or sequential circuit). In my view, the logical structure of the Liar Paradox coincides with the operation of a sequential automaton irrespective of the logical correctness of the paradox itself. The analysis also examines possibilities for further developing the model.

# 1 Finite State Machines

A Mealy automaton is a finite-state machine whose output values are determined simultaneously by the current inputs and its current internal state. A Mealy machine is a deterministic finite-state machine. In many cases, it is necessary to define the initial state of the automaton. It is assumed that, starting from some initial state and under the effect of certain input signals, all states of the automaton are reachable. Some automata have no input, but their initial state remains significant. I refer to such automata without input as generators.

- A deterministic finite-state machine is always in a unique, well-defined state and can only assume a finite number of states. The automaton operates in discrete time. The units of time are sometimes referred to as periods or moments. We use integers to represent discrete time units. For any time $t$, the subsequent time unit is $t+1$, whereas the preceding time unit is $t-1$. Sometimes, I will refer to the present moment as $t_0$, the subsequent moments as $t_1, t_2, \ldots$, and the preceding moments as $t_{-1}, t_{-2}, \ldots$.

- $A$ is the set of internal states, $X$ is the set of input states, and $Y$ is the set of output states. We investigate the output state of the automaton, which is uniquely determined by the internal and input states. In our framework, $Y$ is never an empty set.

- For a given internal state and an input or initial state, the automaton transitions to the next internal state at the next time step. All possible state transitions are described by the function $\delta$:

$$\text{next internal state} = \delta(\text{current input state}, \text{current internal state})$$

- For a given internal state and an input or initial state, the automaton produces an output state simultaneously.
  All possible output states are described by the function $\lambda$:

$$\text{current output state} = \lambda(\text{current input state}, \text{current internal state})$$

These imply that the Mealy finite automaton is an ordered quintuple $\langle A, X, Y, \delta, \lambda \rangle$, where

$$a_1, a_2, a_3, \ldots \in A;\ x_1, x_2, x_3, \ldots \in X;\ y_1, y_2, y_3, \ldots \in Y$$
$$a_w = \delta(x_u, a_v);\ y_i = \lambda(x_u, a_v)$$

Hereafter, a finite automaton is always understood to mean a Mealy-type finite automaton.

## 2 Combinational and sequential circuits

Every truth function corresponds to an isomorphic digital circuit. Consequently, the logical structure of every proposition can be presented within the range of propositional logic as an equivalent digital circuit. Provided that the logical values 'true' and 'false' correspond to the 'high' and 'low' voltage levels, the output of a circuit being equivalent to contradiction is always low level for every input state, whereas the output of a circuit corresponding to a tautology is always high level, irrespective of the input states. On the other hand, the remaining propositions correspond to circuits whose output is high-level if and only if a few of the atomic components of the proposition are true, and the rest of the atomic sentence is false. The inputs equivalent to the atomic propositions are either high or low level. However, what logical circuits are equivalent to a circulating statement or argumentation?

Propositions are true or false irrespective of time, whereas the voltage level of the circuits can change over time. To be more precise, we can say that the input levels of the circuits are high or low depending on whether we evaluate the atomic formulae of the formula expressing the logical structure of the proposition as true or false. The voltage level of the output of the circuit and the truth value of the formula results from it. I will call 'combinational automaton' the digital circuit that may model the formulae of propositional calculus. Because the same atomic formulae can compose more complex formulae, one automaton can have multiple outputs, wherein every output corresponds to a complex formula.

Formulae connected with truth functions yield formulae again. Although there are always corresponding automata for them, the situation is not simple in this case. We do not get combined automata joined to each other in each case, and it is also possible that we do not get an automaton—operating machine or circuit—at all. To avoid this, it is sufficient to comply with the following rules:

(a) It is permitted to join any two inputs of any two automata.

(b) Arrange the automata on neighbouring levels and join the output of an automaton only with the input of another on a higher level.

At every point in time, the output state of a combinational automaton is unambiguously determined by the state of its inputs. In other words, the output of the machine is a function of its input. The two rules given above will guarantee this. If these were not the case, it would not be possible to simulate the automaton by logical formulae because the truth value of a logical formula is unambiguously determined by the truth value of its atomic formulae; correspondingly, the truth value of a proposition is the function of the truth value of its components. Therefore, the rules given in (a) and (b) are interpreted in the world of propositions in the following way: the truth value of a proposition can never be influenced by its own truth value (though it may be influenced by other attributes).

There are digital circuits whose output is not a function of their input. These are not combinational automata, but they can be constructed from components that are combinational automata themselves. They can be constructed using, for instance, AND, NOR, and XOR gates (Table 1).

| First input | Second input | AND | NOR | XOR |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 |

Table 1: Logic gates

The automaton below is based on $AND$, $NOR$, and $XOR$ gates. $Input1$ and $Input2$ are the inputs of an $XOR$ gate followed by two $AND$ gates, and $Output1$ and $Output2$ are the outputs of two $NOR$ gates. It operates in the following way: both input and output are either at a high level or low level at any point in time; the $Output1$ is high, or $Output2$ is low level at a point in time if $Input1$ is low but $Input2$ is high level. If the two inputs are at the same level, either both high or both low, the output remains in the previous $(t-1)$ state. This means that the output is not a function of its inputs at the same time. Notice the feedback between $Output1$ and $Output2$. These automata are very important digital circuits. It has two inputs: $\langle Input1, Input2\rangle$ and two outputs: $\langle Output1, Output2\rangle$. The input variables are $\langle 00\rangle, \langle 01\rangle, \langle 10\rangle, \langle 11\rangle$; the output variables are $\langle 00\rangle, \langle 01\rangle, \langle 10\rangle, \langle 11\rangle$ ($\langle 10\rangle$ means $Output1 = 1$ and $Output2 = 0$). This machine works in discrete time steps. Every time step period has a previous and next time step. The arrows show the direction of signals (Fig. 1).

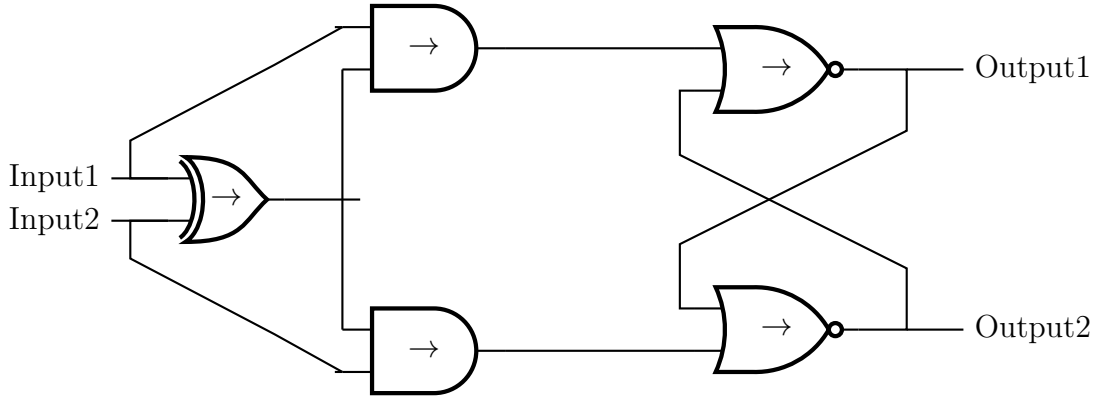


Figure 1: Flip-flop

The range of automata is broader than that of combinational automata. It includes machines whose input states do not unambiguously determine the output states. In other words, the output is not solely a function of the input. This is due to the circuit's feedback mechanism. Most digital circuits fall into this latter category. I will refer to them as sequential automata. Figure 1 illustrates an example of such an automaton, and I will provide further examples later. [1]

This circuit can be considered a finite deterministic automaton (Mealy machine). The output states of such automata depend on the previous internal states and inputs. More precisely, every output and internal state is determined by the preceding input events and internal states. It is assumed that the automaton starts with an initial state, where

[1] In cyberspace, static formulas come to life. These operate over time, and their functionality can be tested. Please see: `https://sht.andrasek.hu/flip-flop.xlsx`

| $\delta$ | a | b |
|---|---|---|
| 00 | a | b |
| 01 | a | a |
| 10 | b | b |
| 11 | a | b |

Table 2: Int.-state transition function

| $\lambda$ | a | b |
|---|---|---|
| 00 | 10 | 01 |
| 01 | 10 | 10 |
| 10 | 01 | 01 |
| 11 | 10 | 01 |

Table 3: Output function

$output1 = 0$ and $output2 = 1$. Every subsequent event in the automaton is caused by its input events.

The machine has two internal states: $a$ and $b$. Here, $a = 10$ means $output1 = 1$ and $output2 = 0$, while $b = 01$ means $output1 = 0$, and $output2 = 1$. The automaton retains its previous internal state—either $a$ or $b$. Two functions, $\delta$ and $\lambda$, describe the automaton's operation. The left column of each table (Tables 2 and 3) represents the set of input states, while the top row represents the internal states.

Next internal state = $\delta$ (current internal state, current input state)

Current output state = $\lambda$ (current internal state, current input state)

Logical relationships between sentences often extend beyond propositional logic, corresponding to the operations of certain sequential logic circuits. What types of logical relationships can sequential circuits model? The following paragraphs will provide examples.

# 3 The Clock Generator as a Model of the Liar Paradox

The next digital circuit is simple, consisting of an inverter and a DELAY circuit. Notice the feedback again. The output of the inverter becomes high when the input is low, and vice versa. The inverter can be modelled as a negation operator if the logical values 'true' and 'false' correspond to the 'high' and 'low' signal levels, respectively. For instance, the input level represents the valuation of the formula '$p$', while the output level represents the valuation of the formula '$\sim p$.'

The DELAY unit operates such that its output state at every $t+1$ period is equivalent to the output state of the previous period, $t$. Thus, this automaton alternates between high and low signals over time. The feedback from the output to the input simulates a scenario where the output level—the valuation of $\sim p$—is a function of itself.[2]

To study this automaton, let us introduce the following notations:

[2]This is analogous to how an electric bell operates. The arm of the bell moves only when there is electric current in the coil, and conversely, current flows only when the arm is not drawn in. In short, the bell arm draws if and only if it does not draw in.

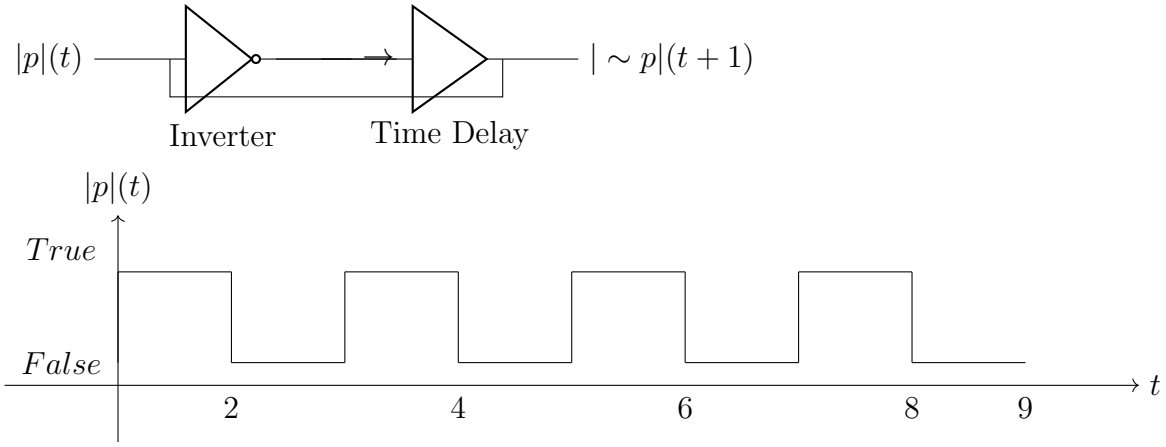


Figure 2: Liar paradox

$1 :=$ True

$0 :=$ False

$2 :=$ Not true and not false, i.e. value gap

$t + 1$ := the next period (time step) after $t$ period

$\sim p$ := Not $p$

$|p|(t)$ := $p$ formula valuation at $t$

$| \sim p|(t + 1)$ :=$\sim p$ formula valuation at $t + 1$

**Input** := $|p|(t)$

**Output** := $| \sim p|(t + 1)$

Figure 2 shows the operation and the output signal of this machine.

The question arises whether there exists a proposition corresponding to such an automaton. In my view, the Liar paradox can serve as a corresponding proposition. Although the Liar paradox is not a proposition in the strict sense, its self-referential nature–where the truth value depends on itself–mirrors the feedback mechanism of the automaton. The truth value of the Liar sentence changes over time, just as the automaton alternates its output. Therefore, the logical structure of the Liar sentence coincides with the operation of this automaton, regardless of the logical validity of the paradox itself.

# 4 $T$ schema

Let the object language sentence be 'The snow is white.' Denoting this by $p_0$, the object language is represented as $L_0$. The meta-language translation of this sentence is $p_1 :=$ The snow is white. Here, the meta-language $L_1$ translates $p_0$ in $L_0$ into a statement about itself. Let $x$ in $L_1$ refer to the sentence $p_0$. Then, the following equivalences hold in $L_1$:

$(T)$ $x$-is $True_1$ iff the snow is white; $AND$ $x$-is $False_1$ iff the snow is not white.

In general form: $x = \ulcorner p_1 \urcorner$

$(T)$ $x$ is $True_1$ iff $p_1$; $AND$ $x$ is $False_1$ iff not-$p_1$.

The extensions of the meta-linguistic predicates $True_1$ and $False_1$ include the names of sentences in language $L_0$. For simplicity, the meta-linguistic sentence $p_1$ will be referred to as $p$ in the following paragraphs.

The principles of bivalence are as follows:

$(B1)$ There is no sentence $x$, such that $x$ is $True_1$, and $x$ is $False_1$.

$(B2)$ There is no sentence $x$, such that $x$ is neither $True_1$ nor $False_1$.

$(B3)$ $x$ is not $True_1$ iff $x$-is $False_1$; AND $x$-is not $False_1$ iff $x$-is $True_1$.

$\{(B1), (B2)\} \Rightarrow (B3)$

# 5 Versions of Liar paradox interpretations

To highlight the specific characteristics of each interpretation, I will apply Kleene's strong three-valued logic and simulate semantic valuations using finite automata. Each version of the Liar paradox is represented by a clock generator where the state corresponds to the sentence's truth value. Each version differs in the internal state transition functions.

## 5.1 Classical Liar paradox

$p :=$ This sentence is false.

Considering the sentence $p$, let $x$ denote $p$ and assume $x = \ulcorner p \urcorner$. Without language levels, a paradox arises. Applying the $T$ schema, if $x$ is false, then not-$p$. Since $p = \ulcorner x$-is-false.$\urcorner$, it follows that not-$\ulcorner x$-is-false.$\urcorner$. By bivalence, $x$ must then be true. Applying the $T$ schema again, if $x$ is true, then $p$, which implies $x$ is false. This cycle repeats indefinitely.[3]

If we suspend bivalence at the meta-language level and consider $x$ meaningless—neither true nor false—then $x$ is not false. However, $p$ asserts that $x$ is false, leading to a contradiction. This again initiates an infinite cycle.

The automaton model simulates this behaviour. Regardless of the initial state, the automaton alternates between true and false values without reaching a constant state.

The digital circuit model (see Table 4).

The automaton model is shown in Table 5 and 6. The first row of each table represents the internal states, while the first column lists the input values.

[3]To see how the application of language levels blocks the paradox, see my paper "What is truth?". `https://ferenc.andrasek.hu/papers/what-is-truth6b.pdf`

| | In | Out | |
|---|---|---|---|
| | 0 | 1 | |
| $\|p\|(t)$ | 1 | 0 | $\| \sim p\|(t+1)$ |
| | 2 | 0 | |
| | | | Feedback |

Table 4: Classical Liar paradox

| $\delta$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 |
| 2 | 1 | 0 | 0 |

Table 5: Int. state transition function

| $\lambda$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 1 | 2 |
| 1 | 0 | 1 | 2 |
| 2 | 0 | 1 | 2 |

Table 6: Output function

The operation does not depend on the initial state. Whether the initial state is 1, 0, or 2, the output alternates between 1 and 0, repeating indefinitely. This result differs from the Kripke model or the strengthened Liar due to differences in the internal state transition functions.

## 5.2 The strengthened Liar

$p :=$ This sentence is false or has no truth value at all, because meaningless.

In this version, we suspend $(B2)$ on the object language level. Consider the sentence $p$ that the sentence named $x$ is false or has no truth value at all, because meaningless. If sentence $x$ is false, then it is the case as sentence $p$ claims, so $p$ is true. If the sentence $x$ is true, then $p$ is not true, because it does not correspond to reality; therefore $p$ is false. If sentence $x$ is neither true nor false because it is meaningless, then $x$ is true because it corresponds to reality, which implies that $p$ is true. A paradox arises if $x$ is the name of sentence $p$. Then the truth value of the sentence depends on itself, expressed by a feedback. The operation of the automaton simulates that no matter how we evaluate sentence $p$, the automaton does not reach a constant state, alternating between true and false values.

The digital circuit model:

| | In | Out | |
|---|---|---|---|
| | 0 | 1 | |
| $\|p\|(t)$ | 1 | 0 | $\| \sim p\|(t+1)$ |
| | 2 | 0 | |
| | | | Feedback |

Table 7

Automaton model:

| $\delta$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 |
| 2 | 1 | 0 | 1 |

Table 8: Int. state transition function

| $\lambda$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 1 | 2 |
| 1 | 0 | 1 | 2 |
| 2 | 0 | 1 | 2 |

Table 9: Output function

The operation does not depend on the initial state. The initial state can be 1 or 0 or 2, the output will alternate between 1 and 0, and it repeats without end. There is no fixed point. No matter what the initial state of the automaton is, that is, no matter whether the strengthened Liar sentence is considered true, false, or value-gap, the automaton does not reach a constant state. Note that $\delta$ function. The table describing the function differs from the Classical Liar Paradox (Table 5, 6).

## 5.3 Kripke's approach

$p :=$'This sentence is false' sentence is ungrounded because it has no independent input that determines the truth value of the sentence.

The digital circuit model:

| | In | Out | |
|---|---|---|---|
| | 0 | 1 | |
| $2 = \|p\|(t)$ | 1 | 0 | $2 = \| \sim p\|(t+1)$ |
| | 2 | 2 | |
| | | | Feedback |

Table 10

Automaton model:

| $\delta$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 |
| 2 | 2 | 2 | 2 |

Table 11: Int. state transition function

| $\lambda$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 1 | 2 |
| 1 | 0 | 1 | 2 |
| 2 | 0 | 1 | 2 |

Table 12: Output function

The first row of the tables shows the internal states; the first column on the left shows the initial values. The operation depends on the initial state. If at time $t$, the value of $p$ is 1 (i.e., true), then next time $t+1$ value is 0 (i.e., false), and vice versa. It repeats without end. The situation changes if the initial value is 2. If at time $t$, the value of $p$ is 2 (i.e., value gap), then next time $t+1$ value also is 2 (i.e., value gap). The value gap is the fixed point of the function $\delta$ and $\lambda$. With this 2 (value gap) initial value, the automaton reaches a stable state. This result is characteristic of the Kripke model. Notice again the characteristic internal state transition functions that are different from previous versions of the paradox (Table 5, 8, 11).

## 5.4 Failed Liar

p:=This sentence is not true or false.

Considering the meta-language sentence $p$, the sentence $x$ is not true or false, but meaningless. Whether $x$ denotes a true or false sentence, it follows that $p$ is false because $p$ asserts the opposite. If $x$ is the name of a meaningless sentence, then we must conclude that $p$ is true because it correctly describes reality. However, this evaluation holds only temporarily.
In the specific case where $x$ is the name of the sentence $p$, it follows that $p$ is true, which in turn negates what $p$ asserts; hence, $p$ is false. Regardless of the initial valuation, the automaton reaches a stable state, and this stable state is 0.

The digital circuit model:

| | In | Out | |
|---|---|---|---|
| | 0 | 0 | |
| $0 = \|p\|(t)$ | 1 | 0 | $0 = \| \sim p\|(t+1)$ |
| | 2 | 1 | |
| | | | Feedback |

Table 13

Automaton model:

| $\delta$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 |

Table 14: Int. state transition function

| $\lambda$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 1 | 2 |
| 1 | 0 | 1 | 2 |
| 2 | 0 | 1 | 2 |

Table 15: Output function

Therefore, assuming that the sentence $p$ is either true or false, the output of the automaton, independent of its internal state, is 0 (i.e., false). If we assume that the sentence has a value gap (i.e., 2), the automaton produces true in the first step and false in the next, eventually stabilizing in this constant state. Thus, regardless of how the sentence is evaluated—true, false, or value gap—the output of the automaton is always 0 (i.e., the sentence is false).

# 6 Other examples

## 6.1 The Truth-Teller is ungrounded

$p :=$ 'This sentence is true'
The sentence $p$ is ungrounded because it has no independent input that determines its truth value.

The digital circuit model:

| | In | Out | |
|---|---|---|---|
| | 0 | 0 | |
| $1 = \|p\|(t)$ | 1 | 1 | $1 = \|p\|(t+1)$ |
| | 2 | 2 | |
| | | | Feedback |

Table 16

Automaton model:

| $\delta$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 2 | 0 | 1 | 2 |

Table 17: Int. state transition function

| $\lambda$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 1 | 2 |
| 1 | 0 | 1 | 2 |
| 2 | 0 | 1 | 2 |

Table 18: Output function

If the initial state is true, the automaton stabilizes in a true constant output state. If the initial state is false, the automaton stabilizes in a false constant output state. If we assume that the Truth-Teller sentence is meaningless and has no truth value, the automaton produces a value-gap output state. Thus, the automaton has three possible constant output states: true, false, or value gap.

## 6.2 The Truth-Teller is Either True or False

p:= 'This sentence is true'
The sentence $p$ is either true or false.

The digital circuit model:

| | In | Out | |
|---|---|---|---|
| | 0 | 0 | |
| $1 = \|p\|(t)$ | 1 | 1 | $1 = \|p\|(t+1)$ |
| | 2 | 0 | |
| | | | Feedback |

Table 19

Automaton model:

If the initial state is true, the automaton stabilizes in a true constant output state. If the initial state is false, the automaton stabilizes in a false constant output state. If we assume that the Truth-Teller sentence is meaningless and has no truth value, the automaton eventually produces a false output state as well. Thus, the automaton has two constant output states: true or false.[4]

[4]You can test these simulations in practice. Please see `https://sht.andrasek.hu/liar-versions4.xlsx`

| $\delta$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 2 | 0 | 1 | 0 |

Table 20: Int. state transition function

| $\lambda$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 1 | 2 |
| 1 | 0 | 1 | 2 |
| 2 | 0 | 1 | 2 |

Table 21: Output function

# 7 Kripke's counter-example

## 7.1 Introduction to the problem

In his seminal study, Saul Kripke addresses the following problem:[5] How do we evaluate the truth of the following statements, given that Jones's statements $(s2)$ and $(s6)$ and Nixon's statements $(s3)$, $(s4)$, and $(s5)$ assert the following?

According to Jones:

- $(s2)$ Most of Nixon's assertions about the Watergate scandal (i.e., more than half) are false.

Additionally, Jones makes an assertion labelled as 'sentence6', referred to as $(s6)$, with an associated truth value $|s6|$.

According to Nixon:

- $(s3)$ All of Jones's assertions about the Watergate scandal are true.

Nixon also makes two additional assertions, labelled as sentence4' and sentence5,' referred to as $(s4)$ and $(s5)$, with their respective truth values $|s4|$ and $|s5|$. These three independent assertions (sentences 4, 5, and 6) do not directly or indirectly reference statements about the story itself—that is, they do not refer to either $(s2)$ or $(s3)$.

To summarize, the independent assertions made by Jones and Nixon can be outlined as follows:

- Nixon made the assertions 'sentence4' and 'sentence5'.
- Jones made the assertion 'sentence6'.

$d2 = 1$: Jones made the assertion $(s2)$.

$e2 = 0$: Nixon did not make the assertion $(s2)$.

$d3 = 0$: Jones did not make the assertion $(s3)$.

$e3 = 1$: Nixon made the assertion $(s3)$.

...

Table 22 summarizes who made which assertion:

[5] "Most (i.e., a majority) of Nixon's assertions about Watergate are false ... Everything Jones says about Watergate is true." Kripke 1975.

| Sentence-Person Relation | Jones | Nixon |
|---|---|---|
| ($s2$) Most of Nixon's assertions about the Watergate scandal (more than half) are false. | $d2 = 1$ | $e2 = 0$ |
| ($s3$) All of Jones's assertions about the Watergate scandal are true. | $d3 = 0$ | $e3 = 1$ |
| ($s4$) sentence4 | $d4 = 0$ | $e4 = 1$ |
| ($s5$) sentence5 | $d5 = 0$ | $e5 = 1$ |
| ($s6$) sentence6 | $d6 = 1$ | $e6 = 0$ |

Table 22: Sentence-Person Relation

## 7.2 The basic valuation concept

The valuation of sentences ($s4$), ($s5$), and ($s6$) is an independent fact; it does not depend on the assertions made by either Jones or Nixon. Note that both ($s2$) and ($s3$) use the predicates 'true' or 'false', and the truth of each sentence is somehow contingent on the truth of the other. Therefore, the truth value of sentences ($s2$) and ($s3$) depends solely on the truth values of sentences 'sentence4', 'sentence5', and 'sentence6', as well as their mutual reference to the truth values ($|s2|$) and ($|s3|$).

However, we can freely evaluate the truth of 'sentence4', 'sentence5', and 'sentence6', but not the truth values of ($s2$) and ($s3$). The truth value of sentences ($s2$) and ($s3$) is, to some extent, determined by the evaluations of the other three sentences, although we have yet to establish a precise method for their determination. It is clear that the truth values of ($s4$), ($s5$), and ($s6$) will influence whether the truth value of sentence ($s2$) (made by Jones) and sentence ($s3$) (made by Nixon) is true, false, or alternating. This gives rise to a total of eight possible variations, which we will explore below.

The question then arises: In what language, and using what method, can we evaluate this? How can we determine the truth value of Jones's assertion ($s2$) and Nixon's assertion ($s3$)? Let's investigate this in more detail.

According to Nixon, every assertion made by Jones is true. Since Jones makes two assertions in total, Nixon claims: Jones's assertions about the Watergate scandal are true if and only if the majority (more than half) of Nixon's assertions about the Watergate scandal are false, and 'sentence6' is true. Using sentence notation, we can more clearly see the logical relationships involved.

Nixon: ($s3$) is true if and only if ($s2$) is true and ($s6$) is true.

Jones makes a more complex statement regarding Nixon's assertions. According to Jones, most of Nixon's assertions are false. Nixon made three assertions, and the majority of these can be false if and only if just one of the three is true. But what happens if, rather than the majority, all of Nixon's assertions are false? And what if Jones's sentence ($s6$) is false? How can we calculate all of these possibilities? In what language can the task be formulated so that all potential outcomes are easily calculable?

Table 23 below summarizes the above and presents the valuations used in the following argumentation. In the right-hand column, $e2 \times |s2| = 1$ if $|s2|$ is true and Jones makes

the assertion $s2$; $e3 \times |s3| = 1$ if $|s3|$ is true and Jones makes the assertion $s3$, etc. The second line is $|s3| = 1$, precisely when all assertions made by Jones are true. Nixon's total number of assertions is $e9$.

| Sentence | Valuation |
|---|---|
| (s2) Most of Nixon's assertions about the Watergate scandal (more than half) are false. | $\|s2\|$ = If $((0.5 > (e2 \times \|s2\| + e3 \times \|s3\| + e4 \times \|s4\| + e5 \times \|s5\| + e6 \times \|s6\|)/e9)$; then $\|s2\| = 1$; otherwise $\|s2\| = 0$) |
| (s3) All of Jones's assertions about the Watergate scandal are true. | $\|s3\|$ = (If $d6 = 1$; then $\|s6\|$; otherwise 1) × (If $d5 = 1$; then $\|s5\|$; otherwise 1) × (If $d4 = 1$; then $\|s4\|$; otherwise 1) × (If $d3 = 1$; then $\|s3\|$; otherwise 1) × (If $d2 = 1$; then $\|s2\|$; otherwise 1) |
| (s4) sentence4 | $\|s4\| = 0$ or 1 |
| (s5) sentence5 | $\|s5\| = 0$ or 1 |
| (s6) sentence6 | $\|s6\| = 0$ or 1 |

Table 23

## 7.3 Limitations to the classical truth theory

According to Alfred Tarski's basic insights, we lack guidelines for applying the concept of truth used by Jones at the first meta-language level and the concept of truth used by Nixon at the higher, second meta-language level—or vice versa. It is clear that since each concept of truth refers to the other, the two meta-language truth predicates cannot be placed on the same meta-language level. This point is correctly highlighted by Kripke in his famous study.

In the formal languages governed by Tarski, the above argumentation about Jones's and Nixon's assertions, as part of the language, cannot be formulated using any Tarskian concept of truth. This is a valid observation made by Kripke. (An additional question is whether this limitation is a virtue or a flaw in the classical concept of truth. I believe it to be a virtue.)

However, neither Kripke nor anyone else who has developed an alternative concept of truth is correct in claiming that, within the usual Tarskian concept of truth, the phenomenon we encounter in semantic paradoxes cannot be described. In this case, we perceive the paradox as a phenomenon—a series of events—rather than as an argument within a logical language framework that is considered correct.

In the following, I present models that simulate the phenomenon we encounter in semantic paradoxes. Rather than constructing another formal language, I create models that describe the phenomenon at the meta-language level, using semantic valuation functions, while remaining within the classical logic framework. These models simulate the semantic phenomenon without using the terms 'true' and 'false', but instead using the corresponding numerals '1' and '0'. In some cases, these numerals are mere characters; in others, they are treated as numbers, especially when used as arithmetic operators.

## 7.4 Arithmetic transformation

$|s|$ denotes the truth value of any sentence $s$. The function $\zeta$ assigns numbers to the truth values: 0 for false, and 1 for true.

(4.1) $s2$ if and only if $0.5 > (\zeta|s3| + \zeta|s4| + \zeta|s5|)/3$

(4.2) $s3$ if and only if $s2$ and $s6$

(4.3) $s2$ if and only if $0.5 > (\zeta|s2| \times \zeta|s6| + \zeta|s4| + \zeta|s5|)/3$ (4.2)

Introducing the obvious definitions: $|s2| := \zeta|s2|$; $|s3| := \zeta|s3|$; $|s4| := \zeta|s4|$; $|s5| := \zeta|c5|$; $|s6| := \zeta|s6|$; this is what we obtain:

(4.4) $|s2| = \zeta|0.5 > (|s2| \times |s6| + |s4| + |s5|)/3)|$
Equation (4.4) is most easily solved using spreadsheet programs, which yields the following truth table (Table 24). It is important to note that $|s2|$ represents the truth value of Jones's assertion, while $|s3|$ represents the truth value of Nixon's assertion. The value of $|s3|$ can be easily calculated using $|s2|$ and $|s6|$: $|s3| = |s2| \times |s6|$. When there is no solution—i.e., when there is no fixed point—the spreadsheet shows no constant value, and the values fluctuate between 0 and 1.

| sentence6 | sentence5 | sentence4 | Jones's sentence | Nixon's sentence |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | There is no fixed point | There is no fixed point |
| 1 | 1 | 0 | There is no fixed point | There is no fixed point |
| 1 | 1 | 1 | 0 | 0 |

Table 24

## 7.5 Finite State Machine model

Table 25 illustrates how the logical relationships between sentences can also be simulated with digital circuits. According to Table 25, the output state of the circuits depends on their own previous states. This feedback simulates the semantic circularity of truth values. Such feedback circuits are called sequential networks, as their operation cannot be described purely as a function of the input signals. The language of truth functions is not suitable to describe them, so we need a different mathematical tool.

Based on this, we can define a Mealy finite automaton to simulate the problem. The output of the automaton represents the truth values of Jones's and Nixon's sentences, while the input represents the values of the other sentences. The machine has two internal states that simulate the valuation situations of the sentences.

(5.1) output = $\lambda$(input, internal state) – output function

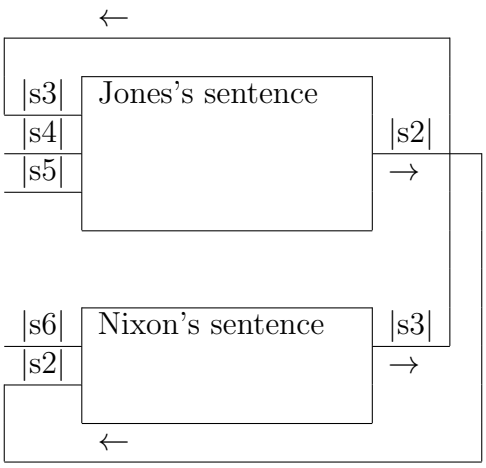


Table 25

(5.2) next internal state = $\delta$(input, internal state) – state transition function

(5.3) internal state (situation) = $\{0, 1\}$

(5.4) input = $\langle |s6|, |s5|, |s4| \rangle$ – three independent sentences

(5.5) output = $\langle |s2|, |s3| \rangle$ – Jones's and Nixon's sentences

The operation of the automaton—its $\delta$ and $\lambda$ functions—is defined in Table 26.

| $\delta$ | 0 | 1 |
|---|---|---|
| 000 | 0 | 1 |
| 001 | 0 | 1 |
| 010 | 0 | 1 |
| 011 | 0 | 1 |
| 100 | 0 | 0 |
| 101 | 1 | 0 |
| 110 | 1 | 0 |
| 111 | 0 | 1 |

| $\lambda$ | 0 | 1 |
|---|---|---|
| 000 | 10 | 10 |
| 001 | 10 | 10 |
| 010 | 10 | 10 |
| 011 | 00 | 00 |
| 100 | 11 | 11 |
| 101 | 11 | 00 |
| 110 | 11 | 00 |
| 111 | 00 | 00 |

Table 26

The finite automaton model also works in cyberspace, generating output values for any given input. In cases where there is no solution—i.e., where a contradiction arises (no fixed point)—there is no stable (constant) state of the automaton, and the output alternates between 1 (true) and 0 (false). In this way, multiple versions of the liar paradox can be simulated using finite automatons. The simulation of semantic values via the finite automaton can be applied in all cases where the domain of discourse is finite.[6]

[6]The working model can be downloaded here:
`https://sht.andrasek.hu/kripke-s-counterexample.xlsx`

# 8 Summary

Feedback logic circuits can simulate a significant fraction of semantic paradoxes (an exception being the Yablo paradox). Feedback simulates the self-dependence of truth values. Such sequential automata can be considered as finite automata. Finite automaton models can distinguish different types of semantic self-reference in truth values. They employ the classical, Tarskian notion of truth.
Their operation can be modelled using spreadsheet software, which allows the adequacy of these simulations to be verified in practice. This is because spreadsheet software can perform computations, unlike static, non-interactive representations such as written text.[7]

# 9 References


- Givone, Donald D. (2002) *Digital Principles and Design*, McGraw Hill India.
- Kripke, Saul (1975) "Outline of a Theory of Truth". *The Journal of Philosophy*, Vol. 72, No. 19, p. 691.
- Mealy, George H. (1955) "A Method for Synthesizing Sequential Circuits". *The Bell System Technical Journal*, vol. 34. 1045–1079.


[7] I formulated the basic ideas back in 1985. Since then, I have rewritten and developed them in several versions.