IRAQI Academic Scientific Journals
Al-Rafidain Engineering Journal (AREJ) Vol.28, No.1, March 2023, pp. 230-238

The Influence of Temporal Logic on Finite Automata

Bilal Abdullateef Kareem
belalengineer53@gmail.com

Ahmad Senjari
ahmadsinjari@gmail.com

Electrical Engineering Department, College of Engineering, Salahaddin Universtiy, Erbil, Iraq

Received: 2022-11-08 Accepted: 2023-01-02

ABSTRACT

The theory of automata combines ideas from engineering, linguistics, mathematics, philosophy, etc. The Entscheidungsproblem asks if it is possible to design a series of steps that replaces a mathematician. An automaton is an abstract machine that processes data. C. Shannon's theory is today's most popular despite having no relationship with the other. The Kt system is called "minimal" because it makes no assumptions about the structure of time. In LKt, we have four monary temporal operators, F, P, G and H, which are mutually interdefinable. Interdefinability means that we will pass logic in the future is the same as saying I will never fail logic, interpreting not passing logic as failing logic. The minimal system syntax of temporal logic introduces operators that have the property of being defined in terms of others. Modal logic studies the reasoning that involves the use of expressions "necessarily" and "possibly". In this article, we will represent through a finite automaton the temporal logic formula Fp. It allows us to see an acceptance pattern for Fp by considering two variables: p and q. Kt's axiomatic system of time expresses the idea that both the present and the past are fixed, if it has always been in the past that it will be some time in the future that p is now. No philosophical argument supports deterministic time flow; the logic of time must be open.Temporal logic has revived many old problems, from the Megaric-Stoics to the minimal system of temporal logic. Our work suggests that the future operators of system Kt follow an evaluation pattern, but we must be cautious because this pattern can only apply to models whose time flow is based on instants and precedence relations.

Keywords:

Arithmetic and Automata, Formal logic, temporal logic, deterministic, non-deterministic.

1. INTRODUCTION

Converging disciplines create issues and problems in many fields. This is the case with the theory of automata, which combines ideas from engineering, linguistics, mathematics, philosophy, etc.[1] The theory of automata is a theoretical discipline that belongs to a broader field. Computer science specifies algorithmic processes using a formal language[2]. Next, we will describe[3] three fundamental contributions to the automata theory. First, Hilbert's formalist program and its relationship with Turing and Church's works define computing's limits. Second, formalizing an abstract machine. Third, automata and formal languages.D. Hilbert's (1982-1943)[4] Ents cheidungs problem asks if all mathematical truths can be derived from an axiomatic system[5]. The task would be to develop a system to decide the truth or falsity of each mathematical statement syntactically.

It resembles Leibniz's company, but symbolic logic was already available. The Entscheidungsproblem[6] asks if it is possible to design a series of steps that replaces a mathematician. Kurt Gödel[7] , Alan Turing[8], and Alonzo Church[6] stood up to him: (1903 - 1995). Gödel's proposal requires understanding consistency, completeness, and decidability. If a formula A has a formal proof, it is universally valid symbolically, according to consistency (1-A then 11-A). If a formula A is a tautology, then A has a proof in the system (11-A then 11-A). A formal system is decidable if we can decide in a finite number of steps if a formula is valid[9]. Classical logic is decidable because truth function.

Automatically determine if a formula is valid in a finite number of steps.After clarifying these concepts, we can list the authors' contributions. Gödel's incompleteness theorems show that any consistent recursive arithmetic theory is incomplete. Axiomatic theories can not prove their own consistency. First, in a recursive arithmetic system where all provable formulas are valid, there will be some valid formula that cannot be derived in the system (neither it nor its negation). Second, the second part of the incompleteness theorem is a negative answer to the Entscheidungsproblem. We can not tell if a theory's formula belongs to it. Turing and Church independently addressed a question in 1936. Church introduced calculus and Turing machines [10]. A Turing machine (TM) can recognize a formula but not determine its truth. Turing's work defined the limits of computation, or what problems an abstract machine can solve.

Church discusses the history of logic and automata in Logic, Arithmetic, and Automata. This article focuses on using formal logic beyond propositional calculus to describe an automaton, an abstract machine and second automata theory contribution. Church notes that Boolean algebra can be used to analyze combinational circuits[11].

C. Shannon (1916-2001)[12] best exemplified this idea, but it had different origins and destinations. V. Shestakov (1907-1987)[13] developed this model in 1934/35 but didn't publish it until 1941. In 1938, in two parts of the world, C. Shannon and Japanese A. Hanzawa[1]. Shannon's theory is today's most popular despite having no relationship with the other. In 1943, neural network behavior was analyzed logically. McCulloch and Pitts[14] introduced this idea in the given year to formally describe neural network behavior. J. von Neumann[15] applied McCulloch and Pitts' ideas to digital circuits in 1945, so S. Kleene[16] defines the first finite automaton in the representation of events in nerve nets and finite automata[17].

The section 3.2 defines automaton strictly. Automata accept external signals, process information, and respond. Dishwashers, mobile phones, lamps, and some daily activities can be considered automata. The simplest automaton is finite, while the Turing machine is the most complex. Stack and linearly bounded automata use different formal resources[18]. Automata types and Chomsky's hierarchy are similar. Chomsky's hierarchy[19] was introduced in 1956/59 in two articles that establish the correspondence between formal grammar and the type of automaton43. Just as there are four types of automata, the hierarchy has four levels that go according to complexity. Automata are abstract machines that process data.

As a result of Hilbert's formalist program, Turing and Church showed that these processes have limits. Now, the formal limits have been set, a formal definition of an automaton is needed. This was done by Kleene. Using the formal definition and processing limits, Chomsky gives us a hierarchy that defines the formal language that recogise our automaton. We can interpret many of the above problems as conceptual problems, so a good philosophical analysis can shed light on them. Rather than answer these questions, we've provided an overview of the disciplines involved in this field.

Interdisciplinarity has created several advances and challenges. Temporal logic shows how interdisciplinary study can be successful. Philosophy, ethics, logic, mathematics, languages, and theoretical informatics shaped its development. This work examines the influence of temporal logic and finite automata. In "Background," we'll briefly explore temporal logic and automata theory to achieve our goal. This section outlines the important contributions to both scientific fields. In "present state," we will discuss our main thesis. We divide the section into three parts . The first explores Kt's syntax, semantics, and axiom system. Second, finite automata are introduced. The third section shows how to build finite automata for Fp and Gp. Formulas and their explanations follow this pattern. In "discussion and positioning," we ponder philosophically on the temporal flow axioms and their link to the paper. In the section of «conclusions and open paths» , the findings of this work and some philosophical issues related to logic-automata relationships are discussed.

2. ACTUAL STATE

This section compares temporal logic to automata theory. A general definition and formal approach to temporal logic are given. The basic concepts of automata theory are introduced and defined very generally. The relationship between temporal logic and automata theory is established by building automata for F and G, explaining their operation and properties.

2.1. TEMPORAL LOGIC KT

Specific questions follow. A formal system consists of a formal language L, syntax, semantics, an axiomatic system, and rules of inference[20]. Natural language has an alphabet to form words and grammar to indicate well-formed sentences. Using the alphabet (a, b, c,... z), we can form the word «lamp». If we want to form a grammatically correct sentence like "the lamp is on," we would first check if each word is in the English vocabulary, then check gender and number agreement, etc. Formal languages specify primitive symbols and syntax. Vocabulary or the alphabet are primitive symbols. The vocabulary includes logical, non-logical, and punctuation symbols.

The non-logical symbols are the so-called propositional variables (p, q, r), and the logical symbols are the so-called logical connectors (¬, ∧, ∨,→, ↔), and the punctuation marks are usually parentheses ( ), braces [1], brackets [ ], commas, etc. Then, the rules must be taken into account to join the elements of the vocabulary, this set of rules is called formal grammar or syntax. These rules state that propositional variables are well-formed formulas (wff) and that certain compound formulas from propositional variables and logical connectors are wffs as well. For example, the following formulas are fbfs: p,(p ↔ r), a ∧ ¬a.

They would not be fbfs: ∧ ∨ r,(¬a →)p). A logical system can be represented as an axiomatic system. An axiom system is made up of a set of formulas (axioms) that are taken as given and that collect important characteristics of the system, from which all the other formulas (theorems) are proved. Therefore, a theorem would be that fbf that has a proof from the axioms of the system.

Interpreting logical and non-logical terms gives a formal system semantics. Tarski's[21] semantic concept of the truth underpins classical logic's standard interpretation. The Kt system is called "minimal" because it makes no assumptions about the structure of time[11]. In the next lines, we will define the minimum system of temporal logic's syntax, semantics, and axioms.

2.1.1. KT SYNTAX AND SEMANTICS

Temporal logic is a reinterpretation of modal logic in temporal terms. The next explanation must include this. Here are the syntax and main components of the minimum system of temporal logic. A formal system consists of a language, in this case, LKt, the minimum system of temporal logic.

According to the definition of formal language that we have given, LKt is made up of logical connectors and propositional variables. let φ the set of pro-positional variables of LKt, where φ = {p, q, r, . . .}, and ϕ any fbf. The logical connectors are the usual ones of the classical logic of propositions; conjunction ∧, disjunction ∨, negation ¬, implication → and co-implication ↔, in addition four monary temporal operators[22] are introduced: “it will be sometime in the future that ϕ” (Fϕ), “it was sometime in the past that ϕ” (Pϕ), “it will always be in the future that ϕ” (Gϕ) and “it has always been in the past that ϕ” (Hϕ). The time operators are mutually interdefinable: Fϕ =def ¬G¬ϕ and Pϕ =def ¬H¬ϕ. Interdefinability is very easy to explain. On the one hand ,we have the formula Fϕ, let's replace ϕ with «approve logic» the phrase then would be; "it will be sometime in the future that I will approve logic", which indicates that it will be sometime in the future that I will approve logic. On the other hand, we have the formula ¬G¬ϕ, which substituting means «that is, it will not happen at all times in the future that suspend logic» . That is, saying I will pass logic in the future is the same as saying I will never fail logic, interpreting not passing logic as failing logic. The following formula is similar. The formulas

are "I've passed logic before" and "I haven't always passed logic." In other words, I failed logic at some point in the past, which means I haven't always failed logic.

Taking into account that temporal operators are mutually interdefinable, we can formally and recursively define the set of fbfs of temporal logic (Φ) as [23]:

$$\Phi = \varphi \mid \neg\phi \mid (\psi \vee \phi) \mid (\psi \wedge \phi) \mid (\psi \rightarrow \phi) \mid F\phi \mid P\phi \mid G\phi \mid H\phi$$

The set of fbfs called Φ, is composed of the set of variables propositions les φ, recall that φ = {p, q, r, ...} and ϕ/ψ, any fbf. The negation of a fbf is also a fbf (¬ϕ). If ϕ and ψ are fbfs, so are (ϕ ∨ ψ), (ψ ∧ ϕ), (ψ → ϕ). They are also fbfs Fϕ, Pϕ, Gϕ and Hϕ. The minimal system syntax of temporal logic introduces temporal operators that have the property of being defined in terms of others Note the link between temporal logic and modal logic in terms of their operators. Modal logic studies the reasoning that involves the use of expressions "necessarily" and "possibly"[24].

However, understanding the term modal logic in a broader way, it would be a family of logics with similar rules and a variety of symbols. In such a case, temporal logic would be a family of modal logic. F and P could be interpreted as operators of possibility "◊" for the future and past, just as the operators G and H can be interpreted as operators of necessity "Ω" for both the future and the past, respectively[2].

Prior introduces temporary operators to the Kt language and formation rules. All these symbols need meaning. Next, it is to explain Kt's semantics. Before tackling such work, we must review classical logic's semantics. Since temporal logic extends classical logic, its operators are the same. Classical logic semantics involve assigning true or false to each fbf [20]. Classical logic only allows 0 for false and 1 for true propositions. . An evaluation function v relates proposition variables to their truth values[25], so.:

$\nu(\neg\phi) = 1$ syss $\nu(\phi) = 0$,

$\nu(\neg\phi) = 0$ syss $1(\phi) = 1$,

$\nu(\phi \wedge \psi) = 1$ syss $\nu(\phi) = 1$ y $\nu(\psi) = 1$,

$\nu(\phi \vee \psi) = 1$ syss $\nu(\phi) = 1$ o $\nu(\psi) = 1$,

$\nu(\phi \rightarrow \psi) = 0$ syss $\nu(\phi) = 1$ y $\nu(\psi) = 0$,

- Where «syss» means if and only if, and «ϕ, ψ» are any propositional formula.

As we see, propositional formulas are interpreted as truth values; for example, the expression: «it rains and I get wet» is formalized by the propositional formula (ϕ∧ψ), where(ϕ) corresponds to «it rains» and (ψ)to «I get wet», can be true {1}or false {0}. The truth value is inductively determined by an evaluation function on all the operators of classical logic. A property of this semantic definition indicates that the truth value of a formula is fixed, in the case of (ϕ ∧ ψ) it will only be true when both propositional variables are true. In all other situations, (ϕ) true and (ψ) false, (ϕ) false and (ψ)true, (ϕ) false and (ψ) false, the evaluation of that formula is false. The rest of the interpretations follow the same scheme.

But if the value is fixed, how could we formalize expressions whose truth value depends on the time at which they have been stated? The idea is to associate the evaluation functions with a flow of time[26]. Formally, a time flow is a relational structure: T = (T, < is a binary relation on T, called a precedence relation. The elements of T are called points in time; if a pair (s, t) belongs to < , , we can say that s is prior to t "In the literature, any relation is normally represented as R. For practical reasons, we now adopt the < symbol for temp logic". The precedence relation has at least two basic properties: the irreflexive(The irreflexive property tells us that no element of the set is related to itself. Formally: $\forall a(a \in A : (a, a) \notin /<))$ and transitive property.If one element is related to another different element and this different element is related to a third party other than the two others, then the first is related to the third. Formally: $\forall a, b, c(a, b, c \in A : a < b \wedge b < c \rightarrow a < c)$ Therefore, an evaluation function ν over a time stream T assigns the value true or false to the set of propositional variables φ in the non-empty set of time instants T. It is formally defined as follows: $\nu : (T \rightarrow (\varphi \rightarrow \{0, 1\}))$ In this way, a model of temporal logic is a pair MKt = (T, ν) consisting of a time stream T and an evaluation function ν.

Now, with this definition, we can already interpret well-formed formulas at each point of the model. For example, we can say that the formula $p \wedge \neg q$ is true at time t, precisely if $\nu(p, t) = 1$ and $\nu(q, t) = 0$. Thus, we proceed to give the inductive definition of the notion of truth of a formula ϕ (semantics) at a time t in a model MKt = (T, <, ν):

$\nu(q, t) = 1$ o $\nu(q, t) = 0$,

$\nu(\neg\phi, t) = 1$ syss $\nu(\neg\phi, t) = 0$,

$\nu(\phi \rightarrow \psi, t) = 1$ syss $\nu(\phi, t) = 1$ y $\nu(\psi, t) = 1$,

$\nu(F\phi, t) = 1$ syss $\exists s \in T(t < s \wedge \nu(\phi, s) = 1)$

$\nu(P\phi, t) = 1$ syss $\exists s \in T(s < t \wedge \nu(\phi, s) = 1)$

For the operators G and H, the semantics would be defined as follows:

$\nu(G\phi, t) = 1$ syss $\forall s \in T(t < s \rightarrow \nu(\phi, s) = 1)$,

$\nu(H\phi, t) = 1$ syss $\forall s \in T(s < t \rightarrow \nu(\phi, s) = 1)$

A temporal logic formula ϕ is valid in the system Kt , if and only if $\nu(\phi, t) = 1$ is true at all time instants in all temporal models. A formula ϕ is true or satisfied if it is true at some instant of time in some temporal model. For example, consider the ordered set of natural numbers N, in which τ is an evaluation function that makes q true for all numbers greater than 1000 and r true for all odd numbers. With this evaluation function, it can be seen that the formula FGq is valid from point 0. The formula tells us: «it will be sometime in the future that it will always be in the future that q > 1000».

However, if we say "it will always be in the future that q > 1000", it is not valid, since if we start from point 0, it is not true that point 1 is greater than 1000, and so on. We can also see that the formula FGr is not valid from point 0, since it does not hold that "it will be sometime in the future that it will always be in the future that r"; in other words, imagine point 11, followed by point 12. From some time in the future (point 11) it does not follow that 'it will always be in the future that r is odd'. However the proposition GFr; "It will always be in the future that it will be some time in the future that r is an odd number" is valid, since if we start from any point, it is true that always in the future we will find an odd number.

Up to this point, we have described the syntax and semantics of Kt . Concerning semantics, we can say that it is closely related to the semantics of possible worlds[27]. To complete the explanation of the Kt system, we proceed to describe the axiomatic system.

2.1.2. AXIOMATIC SYSTEM

Kt's axioms are from classical propositional logic. Frege-Russell Hilbert[20] built the first axiomatic systems. Ax0 contains The proposed axiomatic system is the CHURCH system for the classical logic of propositions. classical logic axioms:

Ax0,1: $p \rightarrow (q \rightarrow p)$

Ax0,2: $(p \rightarrow (q \rightarrow r)) \rightarrow ((p \rightarrow q) \rightarrow (p \rightarrow r))$

Ax0,3: $(\neg q \rightarrow \neg p) \rightarrow (p \rightarrow q)$

Below are the axioms that constitute the minimum system of temporal logic. This system was created by Lemmon in 1965 and is called Kt[26].

Ax0: All the tautologies of the classical logic of propositions.

Ax1: G(p → q) → (Gp → Gq)

Ax2: H(p → q) → (Hp → Hq)

Ax3: p → HFp

Ax4: p → GPp

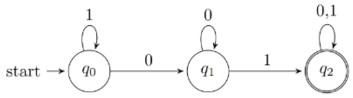
Ax5: Gq → GGq

Ax1 and Ax2 show temporary operators' distributive properties. Ax1 says "p q always implies p always implies q in the future." Ax2 says "p q implies p implies q is always in the past" Ax3 and Ax4 display time. Ax3 says "p implies that it is always been in the past that p" Ax4 says «p implies it will always be in the future that p was in the past. Ax5 shows the transitive property of time, it tells us "q implies that q implies that q implies that q implies".

The rules that operate in the Kt system are the following: On the one hand, we have the uniform substitution rule that tells us that if ϕ is a theorem, then so is ϕ[ψ/q]. On the other hand, the modus ponens (MP) tells us that if ϕ and ϕ → ψ are theorems, then so is ψ. And finally, the temporal generalization tells us that if ϕ is a theorem, then so are Gϕ and Hϕ. In summary, the minimum system of temporal logic consists of a formal language LKt, whose novelty at the syntactic level is the introduction of new temporal operators F, P, G and H, and at the semantic level the introduction of a time flow T = (T, <). With this, we have finished the presentation of the Kt system.

3. AUTOMATAS FINITOS (AF)

If we postmodern any process, we would agree it has states and transitions. A state is an instantaneous description of a system that gives all the relevant information to determine how it can evolve from a given point. Transitions are variations of states over time that can be influenced by external inputs or occur spontaneously. In the abstract, we assume transition states are instantaneous, but they usually take time. Starting an engine, computer circuits, traveling, elevators, Rubik's cubes, etc. are all examples of transition systems. A finite state transition



system consists of finite states and transitions. A finite automaton [28] defines this abstraction.

3.1 FINITE AUTOMATA BASICS (AF)

Finite automata recognize symbol sequences. A symbol could be 0-1-2-2-a-b-c-F. Alphabet is represented by. The binary alphabet, where =0,1, is a good example. We can create symbols from the alphabet. With the binary alphabet, we can create 01101. Given the Latin alphabet, we could create a string etc. A string over is a list where each element is. Languages are finite or infinite strings. Temporal logic uses a finite set of symbols, but their combinations are infinite. A language is a finite or infinite set of strings, and an alphabet's symbols form strings. Formally, a finite automaton A has Q non-empty states. represents a finite alphabet. A function that specifies when an automaton changes states. Initial state q0 and accepting states F, where F Q [5] .

Automaton A is defined by:

Σ: Alphabet

Q: Set of states

q0 ∈ Q: Initial state

$\delta : Q \times \Sigma \rightarrow Q$: Transition function

$F \subset Q$: Final state

There are two types of deterministic finite automata (DFA) and non-deterministic (NDFA). To understand the formal definition of an automaton we will propose two examples that explain the two types of automaton, first, we will see the DFA and then the NDFA.

3.2. DETERMINISTIC FINITE AUTOMATA (DFA)

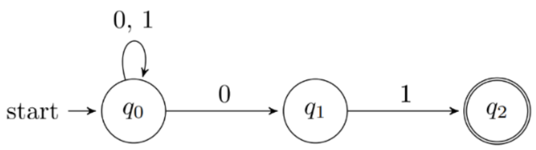
Automata are symbols-recognition machines. Graphs or sequence tables can represent these automata's transition functions. A deterministic finite automaton has one state for each input. We will explain a deterministic finite automaton's operation using its graphical representation.

At first glance, we see a series of circles and arrows. We will say that the circles are the states of the automaton, and the arrows indicate the transitions of the automaton that allow us to go from one state to another and occur when the automaton receives certain symbols as input, in this case {0, 1}, these symbols constitute the alphabet Σ of the automaton. We can identify the set of states as Q = {q0, q1, q2}, where q0 represents the initial state and where q2 the final state F or also called the acceptance state, graphically represented by a double circle. The transitions are represented by the arrows, and the direction of the arrows indicates the location of the next state.

A transition function takes as input parameters a state and a symbol of the alphabet and returns the following state $\delta : Q \times \Sigma \rightarrow Q$. Next, we detail all the transition functions of the automaton. The state q0 has two possible functions: $\delta(q0, 1) = q0$ and $\delta(q0, 0) = q1$, the first one indicates that being in state q0, the symbol 1 is received as input, then it remains in state q0 , but if a 0 is received as input, then state q1 is passed. State q1 has two other possible functions: $\delta(q1, 0) = q1$ and $\delta(q1, 1) = q2$, the first one indicates that being in state q1, the symbol 0 is received as input, so it remains in state q1. state q1, but if a 1 is received as input then it goes to the acceptance state q2. In state q2, we finally have functions $\delta(q2, 0) = q2$ and $\delta(q2, 1) = q2$, which indicate that whether a 0 or a 1 is received, it will remain in the accepting state. If the automaton, responding to the input symbols, manages to reach the acceptance state, then the set of symbols or strings received is a valid formula for the automaton. In this example, valid strings are 100101, 10010, 10011, 1001, 101, and 01.

3.3. NON-DETERMINISTIC FINITE AUTOMATA (AFND)

Use a graphic to explain non-deterministic finite automata (AFND).



At first glance, we see that it is a finite automaton, whose states are q0, q1, q2 and whose alphabet Σ is made up of {0, 1}. So far, nothing different from the AFDs. However, we can notice that state q0 has two different responses to input 0; if a 0 is introduced as the first term of the chain, it can stay in state q0 or advance to state q1. Now the question is: How is an AFND able to recognize a string of symbols? The answer lies in the chain itself. The automaton considers the chain that it has to analyze and based on it, it will decide if it has to go one way or another. If the path takes us to a state whose transition function does not take us anywhere, then it is not a chain accepted by the automaton. To clarify the explanation let's consider chain 01. The first transition function tells us that $\delta(q0, 0) = \{q0, q1\}$. This means that we have two options which we will call option a and option b, where a means to remain in state q0 and b means to advance to state q1. If we decide to consider option a, we will remain in state q0 and therefore, we have to examine the next symbol in the chain, in this case, 1. Being in q0, we have the function $\delta(q0, 1) = q0$ which tells us that if we introduce a 1 so we stay at q0. This shows us that option a does not lead to the state of acceptance. Let us, therefore, consider option b which tells us that we are in state q1 after receiving the symbol 0 as input. Well, the function $\delta(q1, 1) = q2$, tells us that receiving the symbol 1 and being in state q1, leads us to the acceptance state q2.

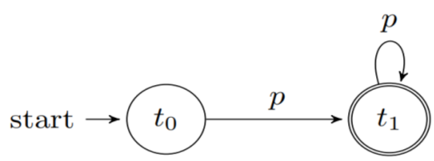
As we can see, the essence of AFNDs lies in their transition function, where before we had a single state to go to; and now, we have a set of states that we could go to in response to an input. Summarizing, an AFND is defined by an alphabet Σ, a nonempty finite set of states Q, an initial state q0, acceptance states $F \subset Q$, and the transition function $\delta = \Sigma \times Q = P(Q)$, where P(Q) is the power set of a given set is another set formed by all the subsets of it. It is denoted P(Q) or 2^Q. set of the non-empty finite set of states of the automaton.

4. TEMPORAL LOGIC AND FINITE AUTOMATA

So far, we have explained the foundations of both temporal logic and finite automata. In what follows, we will focus solely on their relationship, which is a temporal logic model can be interpreted as an automaton that makes a formula true or false. Reviewing: a formula ϕ is true or is satisfied if it is true at some instant of time in some temporal model MKt = (T , ν), where T is a relational structure or flow of time and ν the transition function. The time flow T , in turn, is composed of a set of time instants T and a relation between the time instants <. Relational structures can also be interpreted as finite transition systems[29]. Since finite automata are finite transition systems defined by a quintuple $A = \{\Sigma, Q, q0, \delta, F\}$, then we can construct an automaton that make a temporal logic formula true or false. To illustrate the relationship we will proceed to build the automata for Gp and Fp.

4.1. Finite Automata for Gp

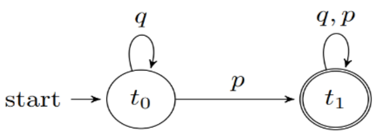
We will represent through a finite automaton the temporal logic formula Gp. The formal semantic definition is as follows: $\nu(Gp, t) = 1$ syss $\forall s \in T(t < s \rightarrow \nu(p, s) = 1)$, and tells us that a Gp formula that is read as "will always be in the future that p", is true ($\nu(G\phi, t) = 1$) if and only if, for every moment s, which belongs to the non-empty finite set of time instants T, t is prior to s then p at moment s is true.



The described automaton is an abstract representation of the behavior of the elements of the automaton are the states t0 and t1m where the first is the initial state and the second the acceptance state, both of which are part of the set of states Q. The automaton alphabet Σ is composed of the symbol {p} and the transition functions which are as follows: The instant t0 has a single transition δ(t0, p) = t1, the instant t1 has a single transition δ(t1, p) = t1. The strings p, pp, and ppp would be examples of strings leading to the acceptance time t1. The automaton allows us to see an acceptance pattern for Gp, where considering the variable, p, allows us to recognize sequences of the type: (m0), p (m1), p (m2), p, where (m0), (m1), and (m2) are instants of time and, p the variables that we considered before. In this way, in short, Gp represented by the proposed automaton tells us that in an initial state p may or may not appear, but later, for the evaluation to be fulfilled, p must always appear.

4.2. Finite Automaton for Fp

We will represent through a finite automaton the temporal logic formula Fp «sometime in the future, it will be that p», whose semantic definition is given by ν(Fp, t) = 1 sys ∃s ∈ T(t < s ∧ ν (p, s) = 1) and tells us that an Fp is true if and only if there exists a moment s belonging to T, the finite non-empty set of time instants, such that t is before s and p at moment s It is true.



The described automaton is an abstract representation of the behavior of the formula Fp. The elements of the proposed automaton are two states t0 and t1, the first is the initial state and the second is the acceptance state, both belonging to the set Q of states of the automaton. The automaton alphabet Σ is composed of {p, q} and the transition functions are as follows: The state t0 has two transitions δ(t0, q) = t0 and δ(t0, p) = t1. The state t1 has two transitions δ(t1, p) = t1 and δ(t1, q) = t1. The strings p, qp, qpq, and qpp would be examples of strings that lead to the acceptance state t1. The automaton allows us to see an acceptance pattern for Fp, where considering two variables, p and q for example, allows us to recognize sequences of the type: (m0), q (m1), q (m2), p (m0), q (m1), p (m2), q (m0), q (m1), q (m2), p where (m0), (m1), (m2), (m3), (m4) and (m5) are instants of time and, p and q, the variables that we considered before. In this way, summing up, Fp represented by the proposed automaton shows us that the temporal logic formula Fp is true if it responds to the pattern that indicates that p appears at least once in the sequence.

5. Discussion and Positioning

5.1. About Ax3 and Ax4, and Flow of Time

The Kt system is said to be minimal because it does not have any assumptions about the nature of time, however, it does have elements worth discussing. We have chosen one of the many problems proposed by van Benthem, J. in Tense Logic and Time[9]. To explain it, we follow the line of argument of Müller, T. (2011) and then we relate the criticism to the thesis of this work. The topic is about Ax3 and Ax4, and the flow of time.

An axiomatic system is intended to reflect the most important features of a system. One of the determining elements of temporal logic is temporal flow. Ax3 and Ax4 represent the time flow in the Kt axiomatic system. Ax3 : p → HF tells us that «p implies that it has always been in the past that it will be sometime in the future p» and Ax4 : p → GPp that «p implies that it will always be in the future that it once was in the past p». Considering Ax3, we see that p is true now; "it is that it has always been in the past that sometime in the future p" represents a very indeterminate idea of what is true now. That in reality in the past, it was already planned that p was going to be now anyway.

Considering Ax4, we see that it expresses the idea that both the present and the past are fixed, «p is now true, if it has always been in the past that it will be sometime in the future that p. «If the operator of the future has as its mission to imitate the use of the future tense in natural language or to provide a basis for philosophical discussions about time, the symmetrical nature of the Ax3 and Ax4 of Kt can be taken as inappropriate»[30].

Müller argues that we can show the formula is philosophically appropriate by assuming linear time and interpreting F as "it will be the case sometime in the future that." Once attention is limited to linear frames, the problem disappears, leaving only a chain of future moments. If both axioms had the same status, there would be no philosophical issue. Computer science studies linear frames for many applications.

We experience time linearly[11] and assign dates, names, etc. to it, so this is the most studied model of time. This is just a concept of time, according to which we experience one moment at a time, remembering the unreflective property that no moment is before or after itself. No philosophical argument supports deterministic time flow. The logic of time must be open.

The problem of elaborating adequate semantics for the temporal operator was central to the pre-development of temporal logic. Consider Peirce's criticism of attempts to describe time algebraically or the problem of futures. Aristotle's army. Our work suggests that the future operators of system Kt follow an evaluation pattern, but we must be cautious because this pattern can only be applied to the model of operators F and G whose time flow is based on instants and precedence relations. It does not say how a future operator will behave with interval-based or branched time.

6. CONCLUSION AND FUTURE WORK

The harmonious link that exists between automata and temporal logic is something to be admired. Both of these stories' twists in the storyline, as well as her fertile nature, speak for themselves. The development of temporal logic and automata theory hints at the existence of conceptual challenges that are the same from the beginning. The role of temporal logic as a technical language that gives exact notions and specifies communication is brought into focus with regard to this particular use of logic. In this approach, temporal logic has resurrected a great deal of previously solved issues. We have examined the relationship between Logic and time throughout history, beginning with the Megaric-Stoics and ending with the simplest system of temporal logic.

Our comprehension of temporal logic has significantly improved as a result. While everything was going on, we worked on automata theory, which included everything from Hilbert's Entscheidungs problem to formal grammar. The theory of automata uses logical language as a foundation to improve conceptual analysis and issue tracking.

The minimum temporal logic system Kt may be thought of as a formal language LKt that introduces additional temporal operators F, P, G, and H at the syntactic level and a flow of time T = (T, T,<) at the semantic level. What sort of primal notions are acceptable, intervals or instants? Is only one example of the philosophical debates centered on this system's problematic conceptual machinery? Or rather, can we condense periods into their constituent moments? Our investigation leads us to the conclusion that the Kt system possesses characteristics that may be either built upon or condemned. That in no way negates the inquiry, but rather prompts more thought and the generation of new issues; for example, is it possible to reconcile two logically distinct conceptions of time (intervals, instants)?

The establishment of this connection between temporal logic and automata is made possible by the advent of finite automata. Because of their inherent abstract nature, automata allow us to understand the composition of a temporal logic formula. Connectivity between the F and G operator models was proven by developing automated representations of each. Given the suggested premise, we are compelled to pursue both of these lines of inquiry. Finding out if there is an automaton that verifies the models of past-time operators would be the first step. Also, would the operators F, P, G, and H be sufficient to define a linear time sequence if we were to avoid the philosophical difficulty of the indeterminacy of the future operator? Alternately, might the need arise for the creation of novel, more evocative operators? Reasoning for these answers comes from theoretical computer science's potential use of temporal logic. Specifically, the operators: δ , Ω, ◊, and U are new to linear temporal logic. As with the fundamental operators of Kt, they are read as "in the next state," "always," "eventually," and "until" and are understood in the same temporal frame.

REFERENCES

[1] R. S. Stanković, J. T. Astola, A. A. Shalyto, and A. V. Strukov, "Reprints from the Early Days of Information Sciences", 2016.

[2] M. A. Orgun and W. Ma, "An overview of temporal and modal logic programming," in *Temporal Logic: First International Conference, ICTL'94 Bonn, Germany, July 11–14, 1994 Proceedings*, 2005, pp. 445–479.

[3] A. Meduna, *Automata and languages: theory and applications*. Springer Science & Business Media, 2012.

[4] C. Reid and H. Weyl, "David Hilbert and His Mathematical Work," *Hilbert*, pp. 245–283, 1970.

[5] W. Thomas, *Languages, automata, and logic*. Springer, 1997.

[6] A. Church, "A note on the Entscheidungsproblem," *The journal of symbolic logic*, vol. 1, no. 1, pp. 40–41, 1936.

[7] K. Gödel, "Collected Works: Volume I: Publications 1929-1936," *Oxford University Press*, vol. 1, USA, 1986.

[8] Bowen, J.P.J.T.S.A.E.o.D.T. and Hudson, *Alan Turing.*, p. 270-275, 2012.

[9] J. van Benthem, "Tense logic and time.," *Notre Dame Journal of Formal Logic*, vol. 25, no. 1, pp. 1–16, 1984.

[10] J. R. Shoenfield, *Mathematical logic*. CRC Press, 2018.

[11] N. Rescher and A. Urquhart, *Temporal logic*, vol. 3. Springer Science & Business Media, 2012.

[12] T. D. Schneider, "Claude Shannon: Biologist [information theory used in biology]," *IEEE Engineering in Medicine and Biology Magazine*, vol. 25, no. 1, pp. 30–33, 2006.

[13] M. K. Kerimov, "Brief history of the Dorodnicyn Computing Center, Russian Academy of Sciences (dedicated to the 50th anniversary of its foundation)," *Computational Mathematics and Mathematical Physics*, vol. 46, pp. 1086–1120, 2006.

[14] S. Chakraverty, D. M. Sahoo, N. R. Mahato, S. Chakraverty, D. M. Sahoo, and N. R. Mahato, "McCulloch–Pitts neural network model," *Concepts of soft computing: fuzzy and ANN with programming*, pp. 167–173, 2019.

[15] J. Von Neumann, *Mathematical foundations of quantum mechanics: New edition*, vol. 53. Princeton university press, 2018.

[16] S. C. Kleene, "On the interpretation of intuitionistic number theory," *The journal of symbolic logic*, vol. 10, no. 4, pp. 109–124, 1945.

[17] D. Giri, K.-K. R. Choo, S. Ponnusamy, W. Meng, S. Akleylek, and S. P. Maity, *Proceedings of the Seventh International Conference on Mathematics and Computing: ICMC 2021*, vol. 1412. Springer Nature, 2022.

[18] A. Salomaa, *Theory of automata*. Elsevier, 2014.

[19] J G. Jäger and J. Rogers, “Formal language theory: refining the Chomsky hierarchy,” *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 367, no. 1598, pp. 1956–1970, 2012.

[20] G. Priest, *An introduction to non-classical logic: From if to is*. Cambridge University Press, 2008.

[21] S. I. Hayakawa, “Semantics, General Semantics,” *ETC: A Review of General Semantics*, pp. 161–170, 1947.

[22] A. N. Prior, “The syntax of time-distinctions,” *Franciscan studies*, vol. 18, no. 2, pp. 105–120, 1958.

[23] D. M. Gabbay, I. Hodkinson, and M. A. Reynolds, “Temporal logic: mathematical foundations and computational aspects,” 1994.

[24] M. R. Genesereth and N. J. Nilsson, *Logical foundations of artificial intelligence*. Morgan Kaufmann, 2012.

[25] L. M. Tabajara and M. Y. Vardi, “Linear Temporal Logic–From Infinite to Finite Horizon,” in *Automated Technology for Verification and Analysis: 19th International Symposium, ATVA 2021, Gold Coast, QLD, Australia, October 18–22, 2021, Proceedings*, 2021, pp. 3–12.

[26] L. Floridi, *The Blackwell guide to the philosophy of computing and information*. John Wiley & Sons, 2008.

[27] J. Van Benthem, J. F. van Benthem, J. F. van Benthem, I. Mathématicien, and J. F. van Benthem, *Modal logic for open minds*. Center for the Study of Language, 2010.

[28] D. C. Kozen, *Automata and computability*. Springer Science & Business Media, 2007.

[29] C. Areces and P. Blackburn, *Bringing them all together*, vol. 11. Oxford University Press, 2001, pp. 657–669.

[30] T. Müller, *Tense or temporal logic*. 2011.

أثير المنطق المؤقت على الأتمتة المحدودة

بلال عبد اللطيف كريم
belalengineer53@gmail.com

احمد سنجاري
ahmadsinjari@gmail.com

قسم الهندسة الكهربائية - كلية الهندسة - جامعة صلاح الدين - أربيل - العراق

الملخص

تجمع نظرية الأوتوماتا أفكارًا من الهندسة ، واللغويات ، والرياضيات ، والفلسفة ، وما إلى ذلك. تسأل Entscheidungsproblem إذا كان من الممكن تصميم سلسلة من الخطوات التي تحل محل عالم الرياضيات. الإنسان الآلي هو آلة مجردة تعالج البيانات. تعتبر نظرية جيم شانون الأكثر شيوعًا اليوم على الرغم من عدم وجود علاقة مع الأخرى. اقترضنا نظام Kt اسم "الحد الأدنى" لأنه لا يقدم أي افتراضات حول هيكل الوقت. وفي نظام LKt لدينا أربعة عوامل مؤقتة أحادية ، Fو P و G و H و H ، والتي يمكن تعريفها بشكل متبادل. القابلية للتعريف تعني ان النظام المنطقي سوف ينجح في المستقبل هو نفس القول بأنني لن أفشل أبدًا في النظام المنطقي ، وأن أفسر عدم نجاح خطوة او فقرة في النظام على أنه منطق فاشل في تحقيق الهدف المنشود لهذه الخطوة او الفقرة . يقدم الحد الأدنى من بناء جملة النظام للمنطق الزمني للمعاملات المختارة خاصية تعريفها من خلال عوامل اخرى مفترضة او مثبتة غير قابلة للاختلاف .

المنطق الشرطي دراسة المنطق التي تنطوي على استخدام التعبيرات "بالضرورة" و "ممكن". في هذه المقالة سوف نمثل من خلال صيغة منطقية وقت Fp تلقائية محدودة. تتكون الأبجدية الآلية من {p، q} ووظائف الانتقال كما يلي. يسمح لنا برؤية نمط قبول لـ Fp من خلال النظر في متغيرين p و q.
يعبر نظام الزمن البديهية لـ Kt عن فكرة أن كل من الحاضر والماضي ثابتان ، إذا كان دائمًا في الماضي أنه سيكون في المستقبل في بعض الوقت أن p الآن. لا توجد حجة فلسفية لدعم التدفق الزمني الحتمي. يجب أن يكون منطق الوقت مفتوحًا ، فقد أعاد المنطق الزمني إحياء العديد من المشاكل القديمة ، من ميغان-رواقيون إلى نظام الحد الأدنى من المنطق الزمني. يشير عملنا إلى أن العاملين المشغلين المستقبليين لنظام Kt يتبعون نمط تقييم ، ولكن يجب أن نكون حذرين لأن هذا النمط لا يمكن تطبيقه إلا على النماذج التي يعتمد تدفقها الزمني على اللحظات وعلاقات الأسبقية.

الكلمات الدالة :

الحساب والأوتوماتا ، المنطق الرسمي ، المنطق الزمني ، الحتمية ، غير القطع