**REVIEW ARTICLES**

### Omega automata and its classes

**Natasha Singh** and **Ajay Kumar**

Thapar Institute of Engineering and Technology, Patiala 147 003, India

---

**Keywords:** Büchi automata, co-Büchi automata and Muller automata, Rabin automata, Streett automata.

In the theory of computation, infinite trees are used as a mathematical tool for representing the behaviour of systems such as protocols or circuits. Omega-automata are used to represent infinite strings (ω-word). ω-word is an infinite sequence of the symbol $a_1, a_2, a_3, \ldots$ where $a_i \in \Sigma$. Language generated by omega-automata is used for representing non-terminating computations. Omega automata are used for specifying and verifying reactive systems, operating systems, concurrent and distributed systems. Omega automata are also applied to solve the significant problem on temporal logic.

Research on ω-language began in 1960 when Büchi studied monadic second-order theories. In this paper, we review various types of omega automata and their applicability. Omega automata can be described by a finite number of states, an input alphabet, an initial state, transition function and an accepting condition. Regular ω-language is the most important ω-language and can be described by ω-regular expression, Büchi automata, Muller automata and other equivalent formalism.

Kupferman et al.1 studied three notions of typeness on ω-regular automata. These notions are useful for transitions between various types of omega automata. Further, they showed that the transitions from non-deterministic Büchi to non-deterministic co-Büchi is more complicated. Tao2 discussed the infinity problem of ω-automata. He showed $N$ logspace completeness if we can convert a generalized Büchi, Rabin, Muller or parity automata into an equivalent non-deterministic Büchi automata in logspace. Chen3 introduced the concept of ω-grammar and ω-automata. He provided a systematic study on the generative power of ω-grammar and described various types of ω-grammar. He showed that the generative power of ω-context-free grammar is strictly weaker than ω-pushdown automata, whereas the generative power of ω-context-sensitive grammar and ω-Turing machine is the same. Redziejowski4 proposed an improved construction of deterministic omega automata from ω-regular expression using the derivative. The proposed methodology was inspired by Safra’s method and produces omega automata with fewer states. Kupferman and Vardi5 described the optimal complementation construction for co-Büchi automata, Rabin and Streett automata.

Properties in model checking are explicitly specified using Linear-time Temporal Logic (LTL). Giannakopoulou and Lerda6 proposed an efficient approach for the conversion of LTL to Büchi automata. Their implementation was released as a part of Java Path Finder Software. Fritz and Wilke7 applied the quotientoing by simulation equivalence technique to alternating Büchi automata. They introduced minimax and semi-elective quotient and proved that these newly introduced quotients are not difficult to compute than non-deterministic Büchi automata quotient.

Carton and Michel8 introduced the concept of unambiguous Büchi automata for recognizing a rational set of ω-word. These automata start from an initial state and reach final states through exactly one path. Sheridan9 established a relation between star normal form (SNF) and alternating automata. He proposed a bounded model checking using alternating automata and explored the difference between alternating, Büchi automata and SNF in terms of their applicability.

Boker and Kupferman10 proposed an approach for conversion of non-deterministic Büchi word automata to deterministic and non-deterministic co-Büchi word automata. The obtained deterministic co-Büchi word automata have the upper bound $2^{O(n)}$, whereas the non-deterministic co-Büchi word automata have the upper bound $(n2^n)$ and lower bound $2^{O(n)}$. Further, they converted Rabin, Muller, Streett and Parity word automata into non-deterministic Co-Büchi word automata. Piterman11 described the construction of deterministic parity automata from non-deterministic Büchi and Streett automata with reduced states. The obtained automata have a variety of applications in satisfiability of CTL and reasoning of tree automata. The main advantage of parity acceptance condition allows more efficient algorithms than Rabin and Streett automata. Zeng and Tan12 proposed a specification-based methodology for testing reactive system by

---

*For correspondence. (e-mail: natasha3699@gmail.com)
specifying requirement using Büchi automata. They built a testing process for specifying the properties of a reactive system. The proposed testing approach detected bugs in the system and its requirement. Further, they evaluated their approach using cross-coverage comparison and fault sensitivity analysis.

Baier et al.\(^{13}\) applied the complementation operator on probabilistic Büchi automata (PBA). They proposed several algorithms for PBA. They showed that equivalence of PBA is an undecidable problem. Safra\(^{14}\) proposed the construction of deterministic Büchi automata from non-deterministic Büchi automata by a generalization of subset construction. The proposed method is simple and provides single exponent upper bound in the general case. His proposed approach can be applied for obtaining the complementation of Büchi automata. Giannakis and Andronikos\(^{15}\) described the query process of linked data using SPARQL and its verification is done using Büchi automata. Thiemann and Sulzmann\(^{16}\) extended the partial derivatives and Brzozowski derivatives\(^{17}\) for conversion of \(\omega\)-regular expressions to non-deterministic Büchi automata by introducing the concept of a linear factor.

Esik and Ivan\(^{18}\) introduced the concepts of context-free grammar with Büchi and Müller acceptance condition. Further, they proved that a context-free grammar generated using Müller acceptance condition is not a Büchi context-free language. Tsay et al.\(^{19}\) created an open repository on \(\omega\)-automata. Readers can refer Thomas\(^{20}\) for preliminaries related to \(\omega\)-automata.

After accessing the applications of \(\omega\)-automata in non-terminating systems, we realized the need for a systematic literature review. We summarized the existing research on \(\omega\)-automata and their classes using abstract and title-based approaches.

### Exploration of various types of omega automata

In this section, a brief overview of various classes of \(\omega\)-automata has been presented. Figure 1 represents various types of \(\omega\)-automata.

### Search inclusion and exclusion criterion

This review starts by identifying research papers related to \(\omega\)-automata and their applications from the Web of Science. Initially, we searched the paper using keyword omega automata, but it gave many irrelevant results related to automata. From the obtained search, we included the paper related to \(\omega\)-automata by looking at the title and abstract. Figure 2 represents the result obtained using keyword-based search followed by an exclusion criterion adopted.

Table 1 clearly shows that irrelevant papers are larger. Hence the keyword-based approach was discarded, and the abstract and title-based search approach was included.

#### Table 1. Results of keyword-based search and identified relevant paper

<table>
<thead>
<tr>
<th>Omega automata (keyword)</th>
<th>Relevant papers based on abstract and title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Omega automata</td>
<td>434</td>
</tr>
<tr>
<td>Büchi automata</td>
<td>352</td>
</tr>
<tr>
<td>Co-Büchi automata</td>
<td>21</td>
</tr>
<tr>
<td>Rabin automata</td>
<td>87</td>
</tr>
<tr>
<td>Muller automata</td>
<td>76</td>
</tr>
<tr>
<td>Streett automata</td>
<td>28</td>
</tr>
</tbody>
</table>

This table shows the number of papers and the number of relevant papers based on the abstract and title.

#### Figure 1. Classification of \(\omega\)-automata.

#### Figure 2. Criterion for selecting the paper relevant to omega automata.
Table 2. Research questions and their motivation

<table>
<thead>
<tr>
<th>Research question</th>
<th>Main motivation</th>
</tr>
</thead>
<tbody>
<tr>
<td>RQ 1: How many papers were published between 1989 and 2017 related to (\omega)-automata?</td>
<td>Identify the papers published after 1989 to till date</td>
</tr>
<tr>
<td>RQ 2: What are the different acceptance criteria for various types of (\omega)-automata?</td>
<td>Identify the major differences between the acceptance criterion of different types of (\omega)-automata</td>
</tr>
<tr>
<td>RQ 3: In which areas (\omega)-automata can be applied.</td>
<td>Identify the major applications of (\omega)-automata</td>
</tr>
<tr>
<td>RQ 4: What are the closure properties of various types of (\omega)-automata?</td>
<td>Identify the closure properties of various types of (\omega)-automata</td>
</tr>
<tr>
<td>RQ 5: To identify various researchers working in the field of (\omega)-automata.</td>
<td>Recognize the active researchers working in the area of (\omega)-automata</td>
</tr>
</tbody>
</table>

Table 3. Classification of papers

<table>
<thead>
<tr>
<th>Property</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year</td>
<td>1989–2017</td>
</tr>
<tr>
<td>Evaluation criteria</td>
<td>Based on title and abstract</td>
</tr>
<tr>
<td>Classification of papers</td>
<td>Papers related to omega, Büchi, co-Büchi, Muller, Rabin and Streett automata</td>
</tr>
<tr>
<td>Publication type</td>
<td>Journal article, book chapter and conference article</td>
</tr>
</tbody>
</table>

Table 4. Analysis of publication type referred in the survey

<table>
<thead>
<tr>
<th>Publication type</th>
<th>No. of papers referred</th>
</tr>
</thead>
<tbody>
<tr>
<td>Journal</td>
<td>63</td>
</tr>
<tr>
<td>Book</td>
<td>2</td>
</tr>
<tr>
<td>Conference article</td>
<td>64</td>
</tr>
</tbody>
</table>

Classification of papers

After identifying the relevant papers, it was observed that \(\omega\)-automata research papers can be categorized based on the research questions in Table 2.

We included papers in our review based on the following classification criteria as shown in Table 3. Publication type analysis of the relevant papers referred in this survey is shown in Table 4.

Results

Year-wise status of publications (RQ 1)

After searching the research papers from Web of Science using keyword, followed by finding the relevant paper using title and abstract the figures are shown in Figure 3. Figure 4 represents the pie chart for the number of papers published since 1989 related to \(\omega\)-automata. Figures 5–10 represent the graph plot of various research papers taken from various resources related to \(\omega\)-automata.

Büchi Automata (RQ 2)

Julius Richard Büchi\(^{21}\) proposed in 1962 the concept of Büchi automata which works on \(\omega\)-strings. If some final state is visited infinitely often for running a \(\omega\)-string, then the string is said to be accepted. It also represents \(\omega\)-regular languages. Nitsche\(^{22}\) proposed a power set construction for Büchi automata. Colcombet and Zdanowski\(^{23}\) found tight lower bound for conversion of Büchi automata into deterministic Rabin automata. Shan et al.\(^{24}\) proposed an approach for conversion of LTL to Büchi automata. Figure 11 represents two different types of Büchi automata.

Büchi automata are classified into unambiguous and ambiguous Büchi automata. Unambiguous Büchi automata start with an initial state and on reading \(\omega\)-word, visit some final state through exactly one path. Ambiguous Büchi automata\(^{8}\) on reading \(\omega\)-string visit some final states infinitely often with more than one path.

Co-Büchi automata (RQ 2)

Co-Büchi automaton\(^{25}\) is a variant of Büchi automata with different accepting conditions. A \(\omega\)-word is accepted by co-Büchi automata if all the states occurring infinitely often in the run belong to the set of final states. Wang et al.\(^{26}\) worked on synthesis of co-Büchi automata specification.

Rabin automata (RQ 2)

Rabin automata\(^{27}\) \(M(Q, \Sigma, \delta, q_0, \Omega)\) with accepting component \(\Omega = \{(E_1, F_1), (E_2, F_2), \ldots, (E_n, F_n) | E_i, F_i \subseteq Q \}\) and a run of \(\omega\)-word is accepted if for \(\forall i \in [1, \ldots, n]\) satisfy conditions \(\text{Inf}(r) \cap E = \emptyset\) and \(\text{Inf}(r) \cap F \neq \emptyset\).

Müller automata (RQ 2)

Muller automata\(^{28}\) \(M(Q, \Sigma, \delta, q_0, \Omega)\) with accepting component \(F \subseteq \text{Pow}(Q)\) and a run of \(\omega\)-word is accepted if \(\text{Inf}(r) \in F\).
Streett automata (RQ 2)

Streett automata \( M = (Q, \Sigma, \delta, q_0, \Omega) \) with accepting component \( \Omega = \{ (E_1, F_1), (E_2, F_2), \ldots, (E_n, F_n) \mid E_i, F_i \subseteq Q \} \) and a run of \( \omega \)-word is accepted if for \( \forall i \in [1, \ldots, n] \) satisfy conditions \( \text{Inf}(r) \cap E_i = \emptyset \) or \( \text{Inf}(r) \cap F_i \neq \emptyset \).

Henzinger and Telle used lockup search for solving non-emptiness of Streett automata.

Table 5 represents the classification of various types of \( \omega \)-automata based on their acceptance condition. Rabin and Streett automata accepting conditions are complement to each other.

The classification table presenting the various classes of omega automata, their acceptance conditions, comparison analysis and the transition modes is depicted in Table 5. The closure properties of various types of \( \omega \)-automata are shown in Table 6.

Applications of omega automata (RQ 3)

The applications of various classes of omega automata are described with the help of Table 7.
Table 5. Classification of $\omega$-automata

<table>
<thead>
<tr>
<th>$\omega$-automata</th>
<th>Year</th>
<th>Acceptance condition</th>
<th>Comparison</th>
</tr>
</thead>
<tbody>
<tr>
<td>Büchi automata</td>
<td>1962</td>
<td>Inf($r$) $\cap$ $F \neq \phi$</td>
<td>There exists at least one final state that is visited infinitely often during $\omega$-run</td>
</tr>
<tr>
<td>Co-Büchi automata</td>
<td>1965</td>
<td>Inf($r$) $\subseteq F$</td>
<td>All states visiting infinitely often belongs to set $F$</td>
</tr>
<tr>
<td>Rabin automata</td>
<td>1969</td>
<td>Inf($r$) $\cap E = \phi$ and Inf($r$) $\cap F = \phi$</td>
<td>Accepting component $\Omega = {(E_1, F_1), (E_2, F_2), \ldots, (E_n, F_n)</td>
</tr>
<tr>
<td>Muller automata</td>
<td>1963</td>
<td>Inf($r$) $\in F$</td>
<td>Infinitely visited states belong to the accepting component</td>
</tr>
<tr>
<td>Streett automata</td>
<td>1982</td>
<td>Inf($r$) $\cap E_i = \phi$ or Inf($r$) $\cap F_i = \phi$</td>
<td>Accepting component $\Omega = {(E_1, F_1), (E_2, F_2), \ldots, (E_n, F_n)</td>
</tr>
</tbody>
</table>

Minimization of non-deterministic Büchi, co-Büchi, Rabin, Muller and Streett automata is PSPACE-comp. Expressive power of Büchi, deterministic Rabin, Müller and Streett automata are same, whereas co-Büchi automata expressive power is less than Büchi automata.

Figure 6. Graph plot of the number of research papers related to Büchi automata.

Figure 7. Graph plot of the number of research papers related to co-Büchi automata.

Figure 8. Graph for various research papers related to Rabin automata.

Figure 9. Graph plot for various research papers related to Muller automata.

Figure 10. Graph plot for various research papers related to Streett automata.

Figure 11. Types of Büchi automata.

**Literature survey on $\omega$-automata (RQ 5)**

McNaughton and Yamada proposed algorithms (supported by theorems) for converting state graph into a regular expression and vice versa. Ginzburg devised a method for checking equivalence between two regular expressions using derivative of regular expressions and state graph. Kumar and Verma determined the state complexity of non-deterministic finite automata from regular expressions. Garhwal and Jiwari introduced the concept of fuzziness in parallel regular expression. Yan determined the lower bound for the complementation of generalized Büchi automata. He showed that the
Table 6. Closure properties of various classes of $\omega$-automata (RQ 4)

<table>
<thead>
<tr>
<th>Types of $\omega$-automata</th>
<th>Closed under operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Büchi automata</td>
<td>Union, concatenation, complement, intersection and omega closure</td>
</tr>
<tr>
<td>Co-Büchi automata</td>
<td>Union, intersection, projection, determinization and omega closure</td>
</tr>
<tr>
<td>Rabin automata</td>
<td>Union, intersection, omega closure and not closed under negation</td>
</tr>
<tr>
<td>Muller automata</td>
<td>Boolean operation and complementation</td>
</tr>
<tr>
<td>Streett automata</td>
<td>Determinization, union, intersection and omega closure</td>
</tr>
</tbody>
</table>

Table 7. Application of Büchi automata in various interdisciplinary fields (RQ 3)

<table>
<thead>
<tr>
<th>Types of omega automata</th>
<th>Application areas</th>
</tr>
</thead>
<tbody>
<tr>
<td>Büchi automata</td>
<td>Developing decision procedures for temporal logic 64</td>
</tr>
<tr>
<td></td>
<td>Bounded model checking of LTL formulae 65</td>
</tr>
<tr>
<td></td>
<td>Modelling of discrete event systems 66</td>
</tr>
<tr>
<td></td>
<td>Querying linked data using Büchi automata 15</td>
</tr>
<tr>
<td></td>
<td>Testing reactive systems 67</td>
</tr>
<tr>
<td></td>
<td>Formalization of digital forensic theory 68</td>
</tr>
<tr>
<td></td>
<td>Generation of profile trees for determinization 69</td>
</tr>
<tr>
<td></td>
<td>Generation of automata games 69</td>
</tr>
<tr>
<td></td>
<td>Generation of emptiness checking algorithms 70</td>
</tr>
<tr>
<td></td>
<td>Generation of safety and reachability games 71</td>
</tr>
<tr>
<td></td>
<td>Biological processes 72</td>
</tr>
<tr>
<td>Co-Büchi automata</td>
<td>Game theory 71,73</td>
</tr>
<tr>
<td></td>
<td>Sequential synthesis of finite state machines using co-Büchi specifications 74</td>
</tr>
<tr>
<td></td>
<td>Puzzle games 73</td>
</tr>
<tr>
<td></td>
<td>Formal verification and synthesis 75</td>
</tr>
<tr>
<td>Muller automata</td>
<td>Modelling asynchronous circuits and real time systems 75,76</td>
</tr>
<tr>
<td></td>
<td>Extension of existential monadic second-order logic 77</td>
</tr>
<tr>
<td>Rabin automata</td>
<td>Game theory 75</td>
</tr>
<tr>
<td>Streett automata</td>
<td>Game theory 63,78</td>
</tr>
<tr>
<td></td>
<td>Verification 79</td>
</tr>
</tbody>
</table>

complementation of Büchi and generalized Büchi automata have the lower bounds $\Omega((0.76n)^n)$ and $(\Omega(nk))^n$ respectively.

Loding and Thomas 36 established the relation between monadic second-order logic and alternating weak automata on $\omega$-strings. Miyano and Hayashi 37 introduced the concept of alternating $\omega$-automata. They characterized the classes of alternating automata into four categories based on acceptance conditions. One of the acceptance conditions increases its power in comparison to non-deterministic. Baier and Grosser 38 introduced the concept of Probabilistic Büchi Automata (PBA). They showed that PBA is more expressive power than non-deterministic Büchi automata. They applied the concept of PBA in the verification of a Markov chain.

Park 39 modelled fair concurrency using the concept of $\omega$-regular languages. He proved that $\omega$-regular languages are closed under the operator fair concurrency. Staiger 40 proposed the concept of finite state $\omega$-languages based on structural properties. The proposed finite state $\omega$-languages are closely related to finite state automata. Study of $\omega$-languages is a subset of topological space. Redziejowski 41 proposed a topology which results in a strongly-zero dimensional and completely regular topological space. Tsay et al. 42 designed a tool Graphical tool for Omega Automata and Logics (GOAL). This tool can be used for checking the equivalence of Büchi automata, conversion of linear temporal logic (LTL) to Büchi automata, design and testing of Büchi automata.

Angluin and Fisman 43 designed three variant of algorithms for finding unknown $\omega$-regular expression based on learning from membership and equivalence queries. Chaturvedi et al. 44 established the relation between infinity games and $\omega$-languages. They represented the winning strategies using *-languages which can be converted into $\omega$-languages. Selivanov 45 introduced the concept of regular aperiodic $\omega$-languages and characterized their Wadge degrees. d’Amorim and Roşu 46 proposed a technique for transforming Büchi automata to statistically optimal non-deterministic finite state machine with reduced size. Finkel 47 studied the closure properties of locally finite omega languages and proved that these languages are not closed under intersection and complement.

Thomas 48 described the relation of $\omega$-words (represented by tree languages) using monadic second-order logic and showed that the $k$-ary tree is undecidable...
Table 8. Key findings related to $\omega$-automata

<table>
<thead>
<tr>
<th>Author</th>
<th>Notable findings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chen(^{3})</td>
<td>Discussed the generative power of $\omega$-grammar</td>
</tr>
<tr>
<td>Redziejowski(^{55})</td>
<td>Conversion of $\omega$-regular expressions to deterministic omega automata using the concept of derivatives</td>
</tr>
<tr>
<td>Giannakopoulou and Lerda(^{6})</td>
<td>Proposed tableau-based translation algorithm for constructing generalized Büchi automata</td>
</tr>
<tr>
<td>Carton and Michel(^{8})</td>
<td>Introduced the concept of unambiguous Büchi automata</td>
</tr>
<tr>
<td>Piterman(^{11})</td>
<td>Conversion of nondeterministic Büchi and Streett automata to deterministic parity automata</td>
</tr>
<tr>
<td>Safra(^{14})</td>
<td>Discussed the complexity of complementation problem for various types of $\omega$-automata</td>
</tr>
<tr>
<td>Antimirov(^{57})</td>
<td>Introduced the concept of partial derivatives for regular expressions</td>
</tr>
<tr>
<td>Thiemann and Sulzmann(^{16})</td>
<td>Extended the conversion of omega-regular expressions to non-deterministic Büchi automata by introducing the concept of a linear factor in the partial derivative</td>
</tr>
</tbody>
</table>

Figure 12. Various approaches for conversion of $\omega$-regular expressions to Büchi automata.

...in monadic second-order logic. Thistle and Wonham\(^{49}\) found that automata can be controlled by imposing certain allowable restriction on the set of symbols. Thistle and Wonham\(^{50}\) developed a fixpoint characterization of controllability subset for a deterministic Rabin automata. Using this fixpoint characterization, automata can be controlled to satisfy the acceptance criterion. Maler and Staiger\(^{51}\) proposed several notions for $\omega$-languages of syntactic congruence. They established a relation between syntactic congruence and its infinitary refinement. Thistle and Malhame\(^{52}\) worked on deadlock-free control of finite automata with respect to the specification specified in the Rabin acceptance conditions. State fairness condition implies that the controllability set can be computed in polynomial time.

Berry and Sethi\(^{53}\) proposed the conversion of regular expressions to finite automata using the derivatives of regular expressions. Owens et al.\(^{54}\) re-examined the regular expression derivative and reported their experiences with two functional languages. Redziejowski\(^{55}\) applied the concept of partial derivatives for the conversion of $\omega$-regular expressions to deterministic $\omega$-automata. Antimirov\(^{56}\) discussed the containment problem in algebra by applying algebraic specifications and term-rewriting methods. Antimirov\(^{57}\) introduced the concept of partial derivatives for the conversion of regular expressions to non-deterministic finite automata based on the generalization of Brzozowski’s derivatives. Caron et al.\(^{58}\) extended the Antimirov’s partial derivatives for the conversion of regular expressions (with complementation and intersection operators) to finite automata. Kumar and Verma\(^{59}\) proposed a direct conversion from parallel regular expressions to deterministic finite automata.

Redziejowski\(^{4}\) proposed an approach for the construction of deterministic automata from $\omega$-regular expression using the derivative concept. Their proposed approach is inspired by Piterman’s improvement. Brzozowski and Leiss\(^{60}\) proposed the equations corresponding to Boolean automata. Their proposed equation was used to determine the language accepted by the sequential network.

Caron et al.\(^{61}\) surveyed for the accepting conditions and equivalence of various $\omega$-automata. Further, they studied the work of prophetic automata. Recently, Singh and Kumar\(^{62}\) initiated the work for modelling various stages of cancer using Büchi automata.

Thiemann and Sulzmann\(^{16}\) extended Brzozowski derivatives and partial derivatives for the construction of non-deterministic Büchi automata from $\omega$-regular expressions by introducing the concept of a linear factor. This is shown with the help of Figure 12. The linear factor is...
defined as 3-tuples \(\langle \alpha, \beta, \delta \rangle\) where \(\alpha, \beta\) and \(\delta\) represent current symbol read, remaining \(\omega\)-regular expression to be processed and the current state which is final \((\delta = 1)\) or non-final \((\delta = 0)\) respectively.

Considering \(\omega\)-regular expression \(r = a^* b^* c^\omega\). On applying Thiemann and Sulzmann approach\(^{16}\) for finding a linear factor, we obtain linear factor (LF) of \(r\) as

\[
\langle a, a^* b^* c^\omega, 0, 0, c^\omega, 1 \rangle = Q = Q_0, \\
\delta(\langle a, a^* b^* c^\omega, 0, 0, c^\omega, 1 \rangle, a) = LF(r) = Q, \\
\delta(\langle a, a^* b^* c^\omega, 0, 0, c^\omega, 1 \rangle, b) = \langle \rangle, \\
\delta(\langle a, a^* b^* c^\omega, 0, 0, c^\omega, 1 \rangle, c) = \langle \rangle, \\
\delta(\langle b, b^* c^\omega, 0, 0, c^\omega, 1 \rangle, a) = \langle \rangle, \\
\delta(\langle b, b^* c^\omega, 0, 0, c^\omega, 1 \rangle, b) = \langle b, b^* c^\omega, 0, 0, c^\omega, 1 \rangle, \\
\delta(\langle b, b^* c^\omega, 0, 0, c^\omega, 1 \rangle, c) = \langle \rangle, \\
\delta(\langle c, c^\omega, 1, b \rangle = \langle c, c^\omega, 1 \rangle. \\
\]

Accepting state is \(F = \langle c, c^\omega, 1 \rangle\).

Table 8 summarizes the key research finding in the area of \(\omega\)-automata.

Open research challenges

The following are the research directions on which future work can be carried out: (i) Work can be carried out on \(\omega\)-regular expression with an additional operator such as shuffle. (ii) Thiemann and Sulzmann\(^{16}\) proposed the concept of a linear factor in the construction of Büchi automata. Work can be carried out for reducing the number of states by combining linear factors. (iii) A systematic comparative study can be carried out for generating Büchi automata from \(\omega\)-regular expressions using existing approaches. (iv) Equivalence problem for \(\omega\)-deterministic pushdown automata is still an open problem. (v) The concept of visibly pushdown automata and infinity games can be explored in the near future. (vi) Work can be carried out for extending Büchi automata with the involvement of quantum and establishing its relation with quantum logic. (vii) New approaches can be proposed for construction of Büchi automata from \(\omega\)-regular expressions.

Conclusion

This paper reviewed and explored \(\omega\)-automata and its types based on 76 relevant papers. Various classes of \(\omega\)-automata have been investigated, characterized by their acceptance criteria and various application areas explored. In addition, this survey paper also reviewed various approaches used for conversion of \(\omega\)-regular expressions to Büchi automata. Finally, this paper also facilitates readers with various open research challenges that future researchers can explore.

15. Giannakis, K. and Andronikos, T., Querying linked data and Büchi automata. 9th International Workshop on Semantic and Social Media Adaptation and Personalization, 9th International Workshop of IEEE, Corfu, Greece, 6–7 November 2014, pp. 110–114.
REVIEW ARTICLES


49. Thistle, J. G. and Wonham, W., Control of omega-automata, Church’s problem, and the emptiness problem for tree omega-automata. 5th Workshop on Computer Science Logic (CSL’91), 1991, 367–381.


Received 6 October 2017; revised accepted 8 August 2018
doi: 10.18520/cs/v115/i11/2042-2051