International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 14 (2015) pp 34712-34716

Comparative analysis of jumping finite automata and classical finite automata

Anita Padyal
Master of engineering
Department of Computer Science and Engineering,
Chandigarh University, Gharuan
Mohali, Punjab
anitapadyal16@gmail.com

Rahul Kumar Singh
Assistant Professor,
Department of Computer Science and Engineering
Chandigarh University, Gharuan
Mohali, Punjab
rahulsinghcse25@gmail.com

Abstract: The present paper provides review on new research in automata theory called jumping finite automata. These automata work similar to classical finite automata except that it process information in discontinuous way that is, after reading a symbol, it can jump over a portion of the input tape in either direction and continue making moves from there. Once an occurrence of a symbol is read on the tape, it may not be re-read again later during computation of the JFA. Otherwise, it coincides with the standard notion of a finite automaton. The paper provides comparison between jumping finite automata and classical finite automata with suitable example. It also provides basis of its origin that is, occurrence of discontinuity in bag automata, on input revolving automata etc that lead to discovery of new automata called jumping finite automata.

Keywords: Classical finite automata**,** discontinuous tape reading, on input revolving automata, bag automata and nested word automata**.**

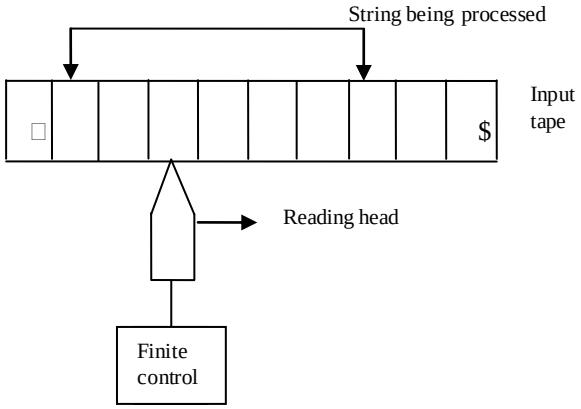
Introduction

Automata play a major role in theory of computation, compiler design, artificial intelligence, parsing etc. In past centuries, most classical computer methods were designed in such a way that processes information in a continuous manner [2]. For example, finite automata that are best known for capturing the family of regular languages works on words that represent information in strict continuous left to right symbol by symbol way. However, modern computer methods are designed in different way of information processing i.e. processes information in a discontinuous manner. For example, within a particular running process, may be particular computational step is performed somewhere in middle and next step may be performed far away from it. So to execute next computational step system may have to jump over large portion of information to desired location in order to complete execution and in this way discontinuity may arise in system [1, 2].

Therefore, classical information methods that process information continuously reflect discontinuous information processing of modern computer methods. Hence this discontinuity in information processing gives rise to idea of adapting classical computer model in discontinuous way. This new version of classical computer model that works discontinuously is known as jumping finite automata [2] which works similar to classical finite automata except that they read input discontinuously. The present paper investigates new version of classical finite automata and some of its closure properties are discussed.

Comparison

Classical finite automata M consist of three components: an input tape, reading head and finite state control [9, 10]. The input tape is divided into square, each square containing a single symbol from the input alphabet ∑. The end squares of tape contain the end marker □ and $ at left and right end respectively. The absence of end marker indicates that tape is of infinite length. The left to right sequence symbols between end marker is input string to be processed. The read head examines only one square at a time and it can move in either direction i.e. left or right. The symbol under read head is current input symbol. The finite state control is represented by a finite set of states together with set of computational rules. The input to finite control will be symbol under read head (say a) and the present state of machine (say q).



M computes by making a sequence of moves. Each move is made according to a computational rule that describe whether the current input symbol is read or not. If the symbol is read, the read head is shifted precisely one square to the right. M has one start state and some states designated as final states. If M can read say w (w is sequence of string to be processed) by making a sequence of moves from the start state to a final state, M accepts w; otherwise, M rejects w [10].

On the other hand, a jumping finite automata works just like classical finite automata except it does not read the input string in a symbol-by-symbol left-to-right way means continuously [2]. That is, after reading a symbol, M can jump over a portion of the tape in either direction and continue making moves from there. Once an occurrence of a symbol is read on the tape, it cannot be re-read again later during

computation of M. Otherwise; it coincides with classical finite automaton.

From above discussion, main difference between classical finite automata and jumping finite automata is that in finite automata the finite machine or system works according to string where as in jumping finite automata string is processed according to machine demand by making jumps wherever necessary. Therefore, in this way finite automation adapts the discontinuity of modern computer models.

TABLE 1. Comparison of FA and JFA

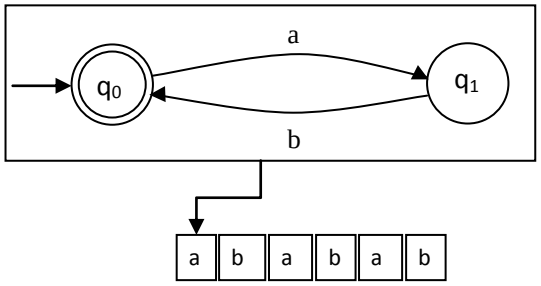
Finite automata(FA)	**Jumping finite automata (JFA)**
Works continuously	Works discontinuously
Symbol occurrence can be re-read again	Symbol occurrence cannot be re-read again
It can have more than one leaving nodes from any state	leaving nodes from any state is always less than one
It can jumps from one state to different states. **Note**: 1 to more than 1 mapping between states.	No jumps between states **Note**: 1 to 1 mapping between states.
$pb \rightarrow q \in R$ implies that $\|b\| \leq 1$ i.e. its degree is less than or equal to 1	$pb \rightarrow q \in R$ implies that $\|b\| \leq 1$ i.e. its degree is less than or equal to 1.

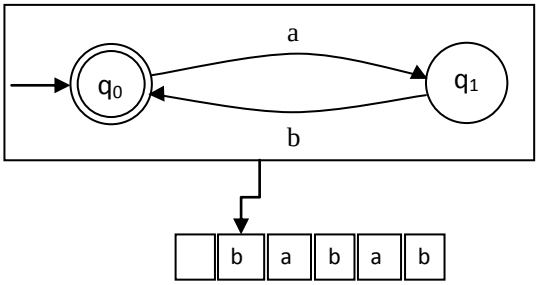
Preliminaries

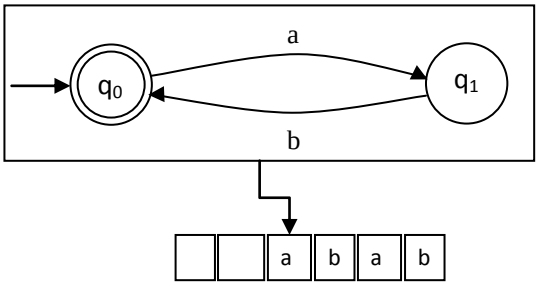
Definition 1: Finite automaton M can be represented by a 5-tuple $(Q, \sum, \delta, q_0, F)$ where

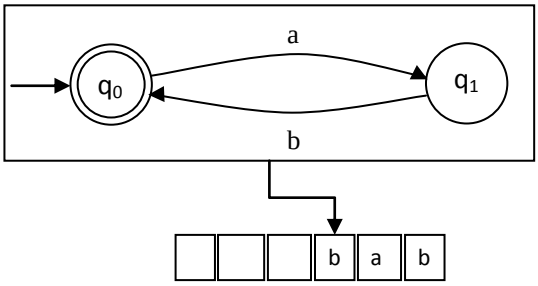
- Q is a finite nonempty set of states.
- $\sum$ is a finite nonempty set of inputs called the input alphabet.
- δ is a function which maps $Q \times \sum \times Q$ and is called the direct transition function which describes the change of states during the transition. This mapping is usually represented by a transition table or a transition diagram.
- $q_0 \in Q$ is the initial state.
- $F \subseteq Q$ is the set of final states [9].

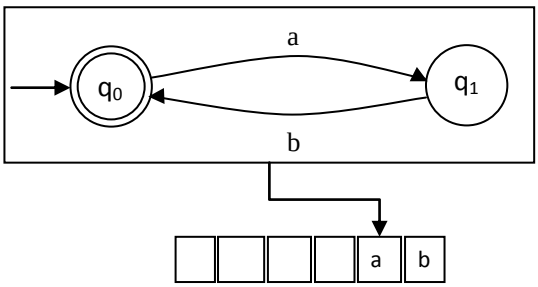
For example, transition diagram for finite automata is given below:

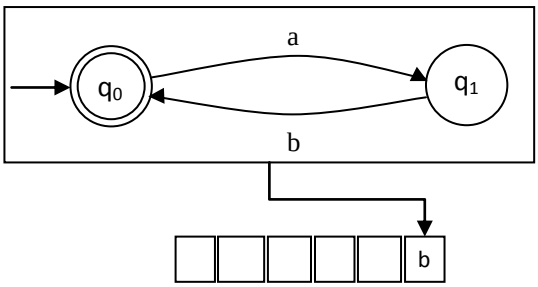


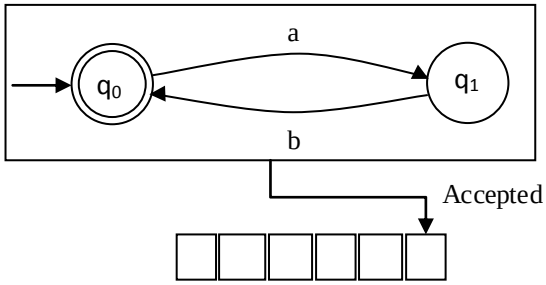












Accepted language: $\{ab\}^*$

Definition 2: A general jumping finite automaton (GJFA) is a quintuple M = (Q, Σ, R, s, F) where

- Q is a finite set of states.
- Σ is the input alphabet.
- R is a finite set of rules of the form $py \rightarrow q$ ($p, q \in Q, y \in \Sigma^*$).
- s is the start state.
- $F \subseteq Q$ is a set of final states [1, 3].

The binary jumping relation symbolically denoted by $\curvearrowright$ is defined as follows:
If $x, z, x', z', y \in \Sigma^*$ such that $xz = x'z'$ and $py \rightarrow q \in R$, then M makes a jump from xpyz to x'qz', symbolically written as

$$xpyz \curvearrowright x'qz'$$

In formal way,
$\curvearrowright^n$ is a sequence of n jumps ($n \geq 0$); mathematically, the nth power of $\curvearrowright$.
$\curvearrowright^+$ is a sequence of jumps (not empty); mathematically, the transitive closure of $\curvearrowright$.
$\curvearrowright^*$ is a sequence of jumps (possibly empty); mathematically, the reflexive-transitive closure of $\curvearrowright$.

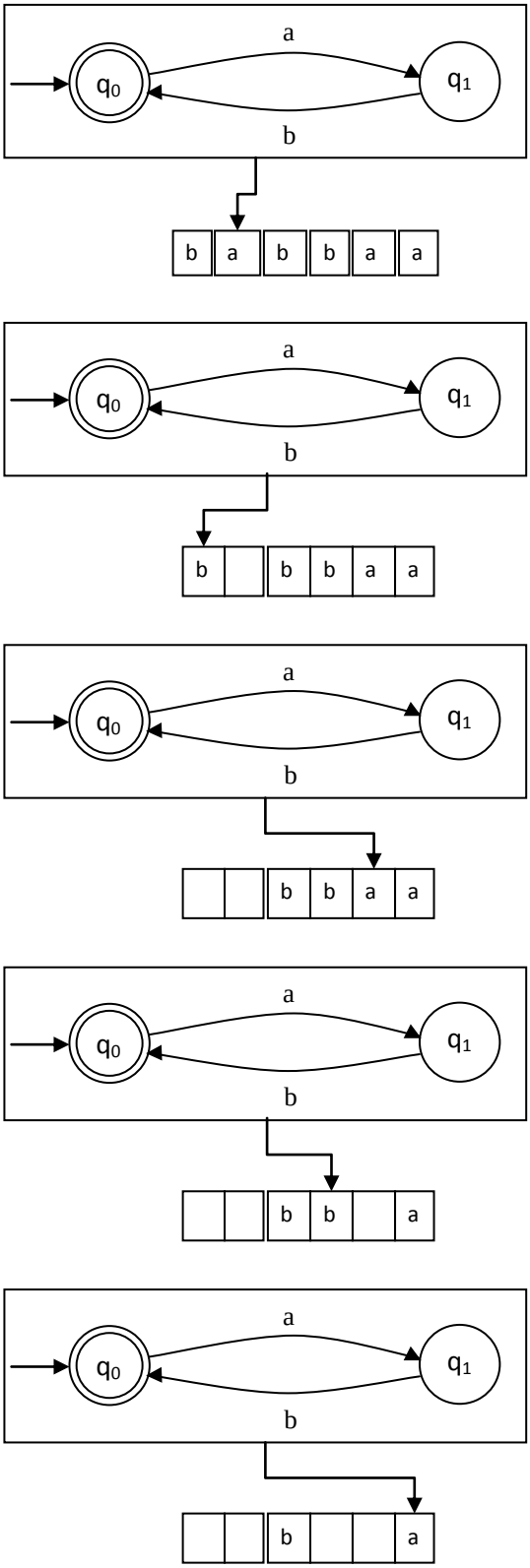
The language accepted by M, denoted by L(M), is defined as $L(M) = \{ uv : u, v \in \Sigma^*, usv \curvearrowright^* f, f \in F \}$. Let $w \in \Sigma^*$ then, M accepts w if and only if $w \in L(M)$. M rejects w if and only if $w \in \Sigma^* - L(M)$. Two GJFAs M and M′ are said to be equivalent if and only if L(M) = L(M′).

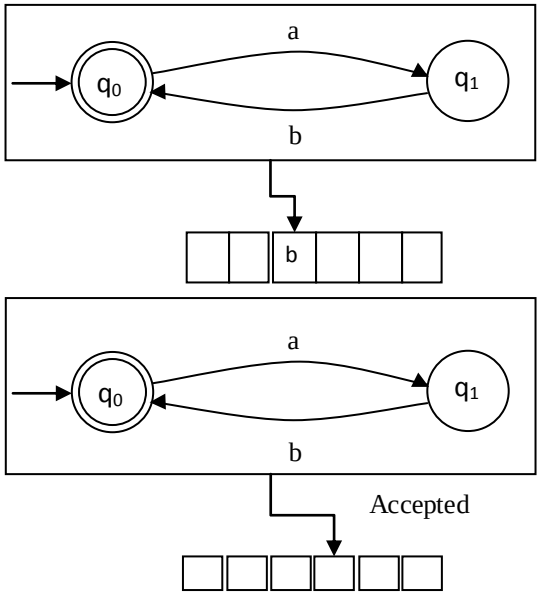
A GJFA M = (Q, Σ, R, s, F) is of degree n, where $n \geq 0$, if $py \rightarrow q \in R$ implies that $|y| \leq n$.

Example: The GJFA M = ({s, p, f}, {a, b, c}, R, s, {f}) with R = {sabc →p, pcc →f, fa →f} is of degree 3.

Definition 3 Let M = (Q, Σ, R, s, F) be a GJFA. M is an ε-free GJFA if $py \rightarrow q \in R$ implies that $|y| \geq 1$. M is of degree n, where $n \geq 0$, if $py \rightarrow q \in R$ implies that $|y| \leq n$ i.e. if all rules $py \rightarrow q \in R$ satisfy $|y| \leq 1$(for short, if its degree is 1)., then M is a jumping finite automation (JFA) [1,2,3].

For example, transition diagram is shown below:





Accepted language: $\{w \in \{a, b\}^*, |w|_a = |w|_b\}$

Related work

A. On input-revolving finite automata

In 2009, Bensch *et. al.* [4] investigate new technique known as on input revolving finite automata that works similarly to ordinary finite automata except that they can make three special operations on input words — left revolving, right revolving, and circular interchanging. For example, let x be the current input word on the tape. Left revolving moves the rightmost symbol of x at the beginning of x, right revolving moves the current input symbol to the very end of x. Finally, circular interchanging exchanges the current input symbol and the rightmost symbol of x. In this way, these three special operations modify x in a discontinuous way which leads to discontinuous information processing.

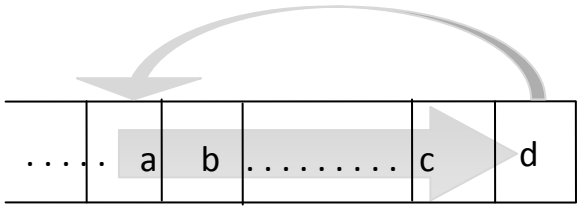


Figure: 3(a) Left revolving

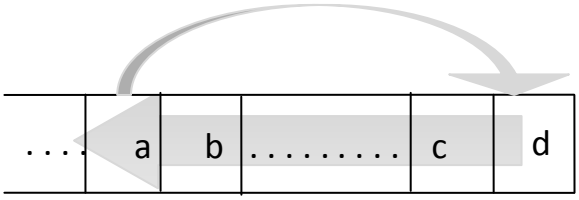


Figure: 3(b) Right revolving

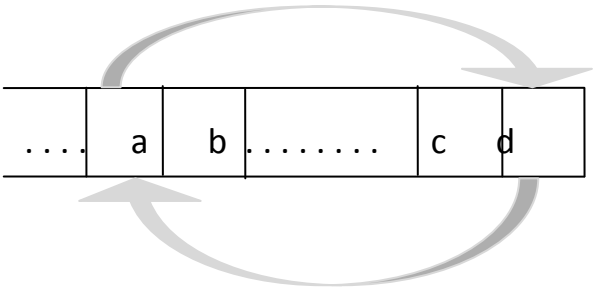


Figure: 3(c) Circular interchanging

B. Bag automata

M.Daley *et. a.l* [5] describes bag automata that works just like pushdown automata except that instead of pushdown lists they use multisets. During a move, these automata can modify their multisets by inserting or deleting a symbol therein. A modification like this can be performed anywhere in the multiset which leads to discontinuity in system; in other words, the multisets are modified discontinuously.

C. Nested word automata

R. Alur *et. al* [6, 7] propose nested word automata that work similar to ordinary finite automata, which read their input words in a left-to-right symbol-by-symbol way. However, as their name indicates, nested word automata work on nested words rather than ordinary words. These nested words are enriched by additional information specifying linear sequencing as well as measures of nestedness. Assume that these automata read a symbol preceded by two predecessors, one of which occurs there because of linear sequencing and the other is there because of nestedness; under this assumption, the next state depends on the states of the automata when they worked on these two predecessors.

Conclusion

This paper, suggested the study of jumping finite automata as a new investigation area in automata theory. It also provides comparison between classical formal models and modern information processing methods. Several related automata like on input revolving automata, bag automata, nested word automata etc have been discussed in this paper that gives idea of processing information in a discontinuous way. This new theory of automata offers an idea of adapting classical formal models in a discontinuous way.

References

[1] A. Meduna "Automata and Languages: Theory and Applications", *Springer*, London, 2000.

[2] A. meduna and P. zemek "Jumping Finite Automata", *International Journal of Foundations of Computer Science* Vol. 23, No. 7 ,1555–1578, 2012.

[3] A. Meduna, Lukas Vrabel, and Petr Zemek "Jumping Finite Automata pdf", *Brno University of Technology, Faculty of Information Technology,* 2012.

[4] S. Bensch, H. Bordihn, M. Holzer, and M. Kutrib "On input-revolving deterministic and nondeterministic finite automata", *Information and Computation*, 207(11):1140–1155, 2009.

[5] M. Daley, M. Eramian, and I. McQuillan "Bag automata and stochastic retrieval of biomolecules in solution", *In O. H. Ibarra and Z. Dang, editors, Implementation and Application of Automata, Eighth International Conference* CIAA 2003, Santa Barbara, CA, July 16-18, 2003, Springer-Verlag, 2003.

[6] R. Alur and P. Madhusudan "Adding nesting structure to words", *Journal of the ACM*, 56.

[7] R. Alur and P. Madhusudan "Visibly pushdown languages", *In Annual ACM Symposium on Theory of Computing*, Chicago, IL, USA, 2004 (3):16:1–16:43, 2009.

[8] Daniel I. A. Cohen "Introduction to computer theory", *Hunter College City University of New York, John Wiley & Sons, Inc*., 1986.

[9] KI.P. Mishra and N. Chandrasekaran "Theory of computer science automata, languages and computation", 3rd edition, *Prentice Hall of India Private limited,* New Delhi - 110 '001, 2008.

[10] Peter Linz "An Introduction to Formal Languages and Automata", 3rd *Edition University of California at Davis,* Jones and Bartlett Publisher, 2001.

[11] Alur, R. 2007 "Marrying words and trees", *In Proceedings of the 26th ACM Symposium on Principles of Database Systems*, 233–242.

[12] N. Baudru "Compositional Synthesis of Asynchronous Automata", *Theoretical Computer Science,* vol. 412, no. 29, pp. 3701-3716, 2011.

[13] A. Meduna "Deep pushdown automata", *Acta Informatica*, 2006(98):114–124, 2006.

[14] D. Wood "Theory of Computation: A Primer", Addison-Wesley, Boston, 1987.

[15] C. Moore and J. Crutchfield "Quantum automata and quantum grammars", *Theoretical Computer Science,* 237(1-2):275–306, 2000.

[16] S. Bensch, H. Bordihn, M. Holzer, M. Kutrib "Deterministic input-reversal and input-revolving finite automata, in: Language and Automata Theory and Applications", vol. 5196 of LNCS, pp. 113–124, Springer, 2008,.

[17] H. Bordihn, M. Holzer, M. Kutrib, "Hybrid extended finite automata, in: Implementation and Application of Automata", vol. 4094 of LNCS, pp. 34–45, Springer, 2006.

[18] H.Bordihn, M.Holzer, M.Kutrib "Input reversals and iterated pushdown automata: a new characterization of Khabbaz geometric hierarchy of languages in: Developments in Language Theory", vol. 3340 of LNCS, pp. 102–113, Springer, 2004.

[19] Bordihn, M. Holzer, M. Kutrib "Revolving-input finite automata, in: Developments in Language Theory", vol. 3572 of LNCS, pp. 168–179, Springer, 2005.

[20] A.K. Chandra, D.C. Kozen, L.J. Stockmeyer"Alternatio", Journal of the ACM, 114–133, 1981.

[21] N. Chomsky "Formal properties of grammars: Handbook of Mathematic Psychology", Wiley & Sons, New York, vol. 2, pp. 323–418, 1962.

[22] C. Culy "Formal properties of natural language and linguistic theories, Linguistics and Philosophy", pp. 599–617, 1996.