State Complexity Advantages of Ultrametric Automata

Maksims Dimitrijevs

University of Latvia Faculty of Computing,
Raiņa bulvāris 19, Riga, LV-1586, Latvia

Abstract. Ultrametric automata have properties similar to the properties of probabilistic automata but the descriptional power of these types of automata can differ very much. In this paper, we compare ultrametric automata with deterministic, nondeterministic, probabilistic and alternating automata with various state complexities. We also show that two-way ultrametric automata can have a smaller state complexity than one-way ultrametric automata.

1 Introduction

p-adic numbers are widely used in chemistry [1], molecular biology [2] and physics [3]. The idea of using p-adic numbers in computer science as parameters in finite automata and Turing machines belongs to Rūsiņš Freivalds [4]. He proved that the use of p-adic numbers exposes new possibilities which do not inhere in deterministic or probabilistic approaches. Moreover, in 1916 Alexander Ostrowski proved that any non-trivial absolute value of the rational numbers $\mathbb{Q}$ is equivalent to either the usual real absolute value or a p-adic absolute value. Therefore using p-adic numbers was the only remaining possibility not yet explored [4].

In our research, we looked at previous results about state complexity of ultrametric automata [4, 6, 9, 10] and improved some of these results. To illustrate, we will exhibit a language that has exponential state complexity in the case of nondeterministic automata, but requires only 13 states for ultrametric automata. We will also compare the state complexity of ultrametric automata with the state complexity of deterministic, nondeterministic, probabilistic, and alternating automata.

After observing one-way ultrametric automata we will look at two-way ultrametric automata. We will then show some possible ways to reduce the number of states by making a two-way ultrametric automaton instead of a one-way ultrametric automaton.

2 p-adic Numbers and Ultrametric Automata

Ultrametric automata use p-adic numbers and therefore we will begin with definitions of p-adic numbers and related operations. A p-adic integer $(a_i)_{i \in N}$ is an infinite sequence of p-adic digits to the left side. A p-adic digit a_i is a natural number between 0 and $p-1$ where p is an arbitrary prime number. p-adic numbers are infinite on the left side but finite on the right side. For each natural number there exists a p-adic representation and only a finite number of p-adic digits are not zeroes. There also exist p-adic float numbers, which have a decimal point. For example, the p-adic number $1/p^2$ can be written as ...0000.01. p-adic numbers are infinite on the left side but finite on the right side.

We can add, subtract, multiply and divide p-adic numbers in the same way as natural numbers. For example we can take a 5-adic representation of $1/2$, which is ...22223. ...22223 + ...22223 = ...00001. The result is the same in the case of rational numbers: $1/2 + 1/2 = 1$. It is important to mention that the only division not available to p-adic integers is division by a positive degree of p (because the equation $x * p = 1$ cannot have the p-adic integer x as a solution). Despite these limitations, p-adic integers can represent any integer and most of rational numbers. The field of p-adic numbers is denoted by $\mathbb{Q}_p$. For those interested, much more about p-adic numbers and mathematical operations has been written by David A. Madore [5].

p-adic numbers cannot be linearly ordered so we will need the absolute value of the p-adic number. If p is a prime number then the p-adic ordinal of the rational number a, denoted by $ord_p a$, is the largest m such that p^m divides a.

Definition 1. *For any rational number x its p-norm (p-adic absolute value) is*

$$\|x\|_p = \begin{cases} 1/p^{ord_p x}, & \text{if } x \neq 0 \\ 0, & \text{if } x = 0. \end{cases}$$

Now we can show the basic definition of ultrametric automata. Ultrametric automata are defined by Rūsiņš Freivalds in [4].

Definition 2. *A one-way p-ultrametric finite automaton is a tuple $(Q, S, \delta, q_0, F, \Lambda)$ where*

- *Q is the finite set of states,*
- *S is the input alphabet,*
- *$\delta : Q \times S \times Q \to \mathbb{Q}_p$ is the transition function,*
- *$q_0 : Q \to \mathbb{Q}_p$ is the initial amplitude distribution,*
- *$F \subseteq Q$ is the set of accepting states,*
- *$\Lambda = (\lambda, \Diamond)$ is the acceptance condition where $\lambda \in \mathbb{R}$ is the acceptance threshold and $\Diamond \in \{\geq, \leq\}$.*

Ultrametric automata are similar to probabilistic automata. A probabilistic automaton has transition probabilities that are real numbers. In the case of p-ultrametric automata, transitions are done with amplitudes, which are p-adic numbers. Therefore, we can assume that for p-ultrametric automata, a prime number p is also a parameter. Every state of ultrametric automaton has a beginning amplitude (the amplitude may also be zero). The final amplitudes of the states are calculated the same way as probabilities for probabilistic automata. To get the result after reading the input word, the amplitude of every accepting state is transformed into p-norm, and the word is accepted if and only if the p-norm sum of accepting states satisfies the acceptance condition. We can also assume that the input word is followed by an end-marker $\dashv$ [4].

There exist ultrametric automata with more restricted definitions. General definitions allow us to use all possible p-adic numbers. This gives them the capability to recognize nonrecursive languages [4] and this is one of the reasons why more restricted definitions have been introduced.

Definition 3. *A finite p-ultrametric automaton is called integral if all the p-adic numbers in its initial distribution and transition function are p-adic integers.*

Definition 4. *A state of a p-ultrametric automaton is called regulated if there exist constants λ, c such that for every input word the p-norm of amplitude γ of this state is bounded by $\lambda - c < \|\gamma\|_p < \lambda + c$. A finite p-ultrametric automaton is called regulated if all of its states are regulated.*

Ultrametric integral automata do not have examples of recognizing nonrecursive languages. Ultrametric regulated automata can recognize only regular languages, but still can have great state complexity advantages over deterministic finite automata [6].

We need to note that if there is only one accepting state, then the possible amplitudes of acceptance are discrete values $0, p^1, p^{-1}, p^2, p^{-2}, p^3, ...$ Hence, there is no natural counterpart of isolated cut-point or bounded error for ultrametric machines [4].

3 Reducing the Number of States with Ultrametric Automata

We have researched the state complexity advantages of ultrametric automata. In [6] we considered a regular language $L_{k,m}$. Let $w = (w_1, w_2, ..., w_m) \in \{0, 1, ..., k-1\}^m$, and consider the following two operations:

1. a cyclic shift: $f_a(w_1, w_2, ..., w_m) = (w_m, w_1, w_2, ..., w_{m-1})$;
2. increasing the first element: $f_b(w_1, w_2, ..., w_m) = ((w_1+1) mod k, w_2, ..., w_m)$.

Let $x \in \{a, b\}^*$. We define $f_{x_1 x_2 \dots x_n}(w) = f_{x_n}(...f_{x_2}(f_{x_1}(w))...)$. The considered language is $L_{k,m} = \{x \in \{a, b\}^* | f_x(0^m) = 0^m\}$. We have proven that a deterministic automaton requires at least k^m states. For all prime numbers p, an ultrametric automata can recognize $L_{k,m}$ with $k * m$ states. In this case we can construct a regulated ultrametric automaton. Stronger results are achieved for every prime $p > m$: p-ultrametric automata can recognize $L_{p,m}$ with $m+1$ state [6]. In this case amplitudes and p-norms of the states are not bounded, and the automaton is integral, but not regulated.

We will show that the results achieved in [6] for ultrametric regulated automata can also be achieved for other models, which can recognize only regular languages.

Theorem 1. *A one-way alternating automaton can recognize* $L_{k,m}$ *with* $k*m+1$ *states.*

Proof. To recognize the language $L_{k,m}$ we will construct an alternating automaton with one beginning universal state and with ε-transitions it will get into all accepting states. The other $k*m$ states are existence states. The constructed automaton is shown on Fig 1. The automaton works like in the proof for regulated ultrametric automata in [6]. w_s^t denotes that s-th element of w has value t. This means that in the beginning all w elements have value 0, and because all transitions were done from the universal state, they should return into accepting states, that is, after all the operations with w all elements should have value 0, otherwise the automaton will reject the input word. □

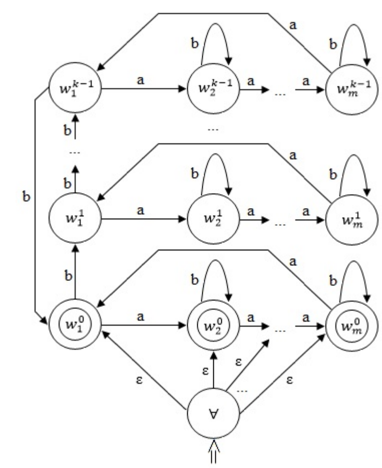

Fig. 1. An alternating automaton recognizing $L_{k,m}$

To conclude about the state complexity advantages of ultrametric regulated automata over deterministic automata, we have to mention the two following results. There is a proof that for any arbitrary prime number p there is a constant c_p such that if a language M is recognized by a regulated p-ultrametric finite automaton with k states, then there is a deterministic finite automaton with $(c_p)^{k*logk}$ states recognizing the language M [6]. Second, there is a proof that such a difference in state complexity is obtainable: for any arbitrary prime p there is a language, which is recognized by a p-ultrametric regulated automaton with $p+2$ states, and this language requires at least $p! = c^{p*logp}$ states for the deterministic automaton to recognize this language [4].

Ultrametric integral automata have better capabilities than regulated ultrametric automata and we can expect greater state complexity advantages. We consider a language $L_n = \{awbwa | w \in \{0,1\}^* \, and \, |w| = n\}$. One-way deterministic and even nondeterministic automata require at least 2^n states [7].

Theorem 2. *For every prime number p language L_n can be recognized by integral p-ultrametric finite automaton with constant state complexity.*

Proof. To prove this theorem we will construct a p-ultrametric automaton with 13 states. The constructed automaton is shown in Fig. 2. $l|m$ denotes that a transition is made for letter l with amplitude m. The automaton has four logical parts. After reading letter b, the amplitude of the state $q3$ will be $\sum_{i=0}^{n-1} x_i * p^i$ where n is the number of letters before b, x_i is the i-th symbol of the first part w, where x_i is either 0 or 1. Analagously, after reading the second part w, $\sum_{i=0}^{m-1} x_i * p^i$ is subtracted from the amplitude of the state $q3$, where m is the number of symbols between b and the second letter a. The amplitude of $q3$ will be equal to zero if and only if the positions of the 1s are the same in both word parts w.

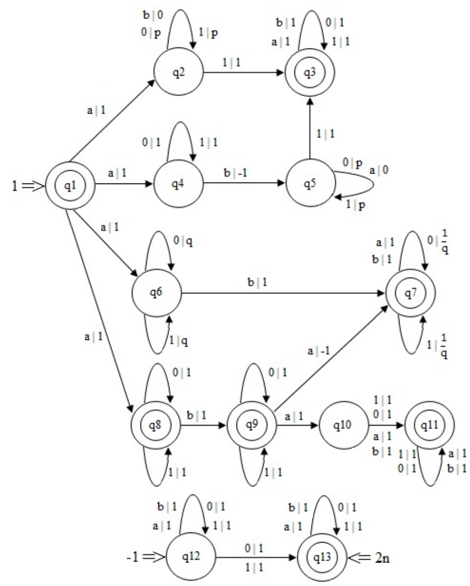

Fig. 2. A p-ultrametric automaton recognizing L_n

When the input word is read, the amplitude of $q7$ will be $q^{|w_1|}/q^{|w_2|} - 1$, where $|w_1|$ is the length of the first part w and $|w_2|$ the length of second part w. Therefore, the amplitude of state $q7$ will be zero if and only if the lengths of both parts w are equal.

States $q8$, $q9$ and $q11$ will have an amplitude of zero if and only if the input word has a correct structure: b after a, the second letter a after b, and no letters present after the second letter a. If this condition does not hold one of the mentioned states will have amplitude 1.

States $q12$ and $q13$ ensure the check for equality $|w| = n$. In the beginning, state $q13$ has amplitude $2n$ and when the symbols zero or one are read, with the help of state $q12$, 1 is subtracted from the amplitude of state $q13$. An acceptable input word should have two parts w, each having n symbols, and therefore state $q13$ will have amplitude zero if both parts w have length n.

A constructed automaton has four checks, and an input word belongs to L_n if and only if all four conditions are satisfied and the amplitudes of all the accepting states are equal to zero. In this case, the sum of the p-norms of all the accepting states will be zero. Otherwise, it will be greater than zero. All the amplitudes of the constructed automaton are p-adic integers (it doesn't have divisions by p) and therefore it is an integral ultrametric automaton. □

By constructing the aforementioned automaton, we have shown that there exists a language that requires exponential state complexity for nondeterministic automata and constant state complexity for p-ultrametric automata for every prime number p. We can improve this result by increasing the base of exponent.

Theorem 3. *For every positive integer k there exists a language, which consists of words of length $O(n)$, requires at least k^n states to be recognized by a nondeterministic finite automaton, but for every prime number p, an integral p-ultrametric finite automaton can recognize this language with constant state complexity.*

Proof. We will slightly improve the language $L_n = \{awbwa | w \in \{0,1\}^* \; and \; |w| = n\}$, this time w will be in the k letter alphabet. Similar to the description in [7], the new language will require at least k^n states for a nondeterministic automaton. For the ultrametric automaton in the proof of Theorem 2 we will need to change only the first logical part. One of the possibilities is to duplicate the states $q2$, $q3$, $q4$ and $q5$ for 3-rd, 4-th, ..., k-th letter of the alphabet. In this case, we will have a p-ultrametric automaton with $13 + (k - 2) * 4 = 4 * k + 5$ states, which is a constant that does not depend on the length of the input word. □

Ultrametric integral automata can have smaller state complexities than probabilistic automata. We consider the following language, where p is a prime number: $L_p = \{1^n | n \; is \; divisible \; by \; p\}$. In [8] there is a proof that any one-way probabilistic finite automaton recognizing L_p with probability $1/2+\varepsilon$, for a fixed $\varepsilon > 0$, has at least p states.

Theorem 4. *Language L_p can be recognized by an integral p-ultrametric automaton with two states.*

Proof. The constructed p-ultrametric automaton is shown in Fig. 3. After reading an input word of length n, the amplitude of the accepting state will be n. If n is divisible by p, the p-norm of n will be p^{-c}, where c is a positive integer. If $n = 0$, the p-norm will be 0. Otherwise, the p-norm will be 1. So, the acceptance condition for the p-norm will be $\leq p^{-1}$. □

In [9] another example is shown, where probabilistic automata require more states than p-ultrametric automata. Here the language is quite simple: $C_n =$

$\{1^n\}$. A probabilistic automaton requires 3 states, while a p-ultrametric automaton requires 2 states. There exists a probabilistic automaton with 3 states and a p-ultrametric automaton with 2 states, both recognizing C_n. The strength of this result is in the fact that it is proven for all prime numbers p as parameters of p-ultrametric automaton.

We can improve another result achieved in [10]. Authors show that language $L_N = \{1^n | n \neq N\}$ can be recognized by a p-ultrametric finite automaton for an arbitrary odd prime number p, having $O((logN)^2 * loglogN)$ states.

Theorem 5. *For every prime number p, for every natural number N, language L_N can be recognized by a p-ultrametric automaton with two states.*

Proof. The constructed p-ultrametric automaton is shown in Fig. 4. After reading N letters, both accepting states will have an amplitude of 1. So, the p-norm sum of amplitudes will be equal to 2. Otherwise, the p-norm sum will be $p^c + p^{-c}$, where $c \neq 0$. We can see that for every prime number p, if $c \neq 0$, $p^c + p^{-c} > 2$ (the smallest possible value is $2^1 + 2^{-1} = 2.5$). Therefore, the acceptance condition for the p-norm sum will be ≥ 2.5. ☐

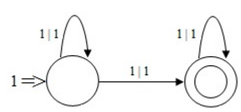

Fig. 3. A p-ultrametric automaton recognizing L_p

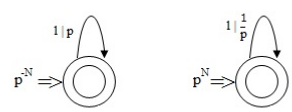

Fig. 4. A p-ultrametric automaton recognizing L_N

The situation with the language L_N is similar to the situation of the language C_n. Both languages can be recognized by p-ultrametric automata with two states and both languages require at least three states to be recognized by probabilistic automata (this can be concluded from [9]).

In [11] the authors considered a language L_m with the alphabet $\{a_1, a_2, ..., a_m\}$ and consisting of all words that contain each of the letters $a_1, a_2, ..., a_m$ exactly m times. There exists a probabilistic finite automaton with isolated cutpoint, which accepts L_m and has $O(m*(logm)^2/loglogm)$ states. A deterministic finite automaton requires at least $(m+1)^m$ states to recognize this language [11].

Theorem 6. *For every prime number p, language L_m can be recognized by an integral p-ultrametric automaton with two states.*

Proof. We will take m different prime numbers $p_1, p_2, ..., p_m$, all of them different from p. The beginning amplitude of an accepting state will be $p_1^m * p_2^m * ... * p_m^m$. After reading the symbol a_i the amplitude of an accepting state will be multiplied by p_i^{-1}. If each of the letters $a_1, a_2, ..., a_m$ was present exactly m times, the amplitude of an accepting state will be 1. Otherwise, it will be different. Using a second state we can subtract 1 from the amplitude of an accepting state after the input word has been read. Therefore, the amplitude of an accepting state will be zero if and only if the input word belongs to L_m. Therefore, the acceptance condition for p-norm will be ≤ 0. The constructed automaton does not have amplitudes that are not p-adic integers. □

The proven theorem shows how we can reduce the number of states in ultrametric automata by "hiding" different counters in one particular state. We cannot achieve smaller state complexity than in Theorem 5 and Theorem 6 because a p-ultrametric automaton requires at least two states to recognize the aforementioned languages. The idea of this proof will be similar to that of the proof of Theorem 10 in [9]. In the case of one state, as the length of the word increases, the norm increases or decreases monotonically or does not change. Therefore, in the case of Theorem 5 and Theorem 6 we are not able to decrease the state complexity for ultrametric automata from two states to one state.

Consider the language, defined for all integers $k > 0$: $EVENODD^k_{yes} = \{a^{j2^k} | j \;\; is \;\; a \;\; nonnegative \;\; even \;\; integer\}$. It is known that language $EVENODD^k_{yes}$ requires at least 2^{k+1} states to be recognized by a one-way probabilistic automaton with a probability of at least $1/2 + \varepsilon$ (for a fixed $\varepsilon > 0$) [12]. A two-way nondeterministic automaton also requires at least 2^{k+1} states [13]. We can reduce the number of states by half with a regulated ultrametric automaton.

Theorem 7. *For every prime number p, the language $EVENODD^k_{yes}$ can be recognized by a regulated p-ultrametric automaton with $2^k + 1$ states.*

Proof. We will use an amplitude to reduce the number of states by half compared to 2^{k+1} states in the deterministic approach. The automaton can be seen in Fig. 5.

The constructed automaton has a cycle of length 2^k. When an amplitude has reached an accepting state, that means that the automaton has received a number of symbols a which is divisible by 2^k. The amplitude is then multiplied by -1 before reaching the accepting state. The accepting state has an amplitude of 1 if and only if the number of symbols a is divisible by 2^{k+1}. If the number of symbols a is divisible by 2^k, but not divisible by 2^{k+1}, the accepting state will have an amplitude of -1. If the number of symbols a is not divisible by 2^k, the accepting state will have an amplitude of zero.

One more state is required to subtract 1 from the amplitude of the accepting state after reading the input word. The accepting state will have amplitude zero if the number of symbols a was divisible by 2^{k+1}, otherwise it will be -1 or -2. Therefore, the input word will be accepted if the p-norm of the accepting state will satisfy condition ≤ 0.

The constructed automaton is regulated. Any of the states can have the amplitudes -2, -1, 0 or 1, but not any other. The automaton will work equally with all prime numbers p. Therefore, we have constructed a regulated p-ultrametric automaton with $2^k + 1$ states to recognize $EVENODD^k_{yes}$. □

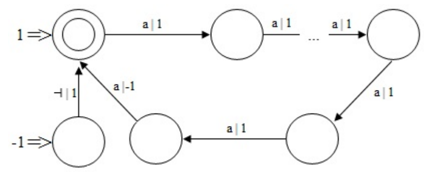

Fig. 5. A regulated p-ultrametric automaton recognizing $EVENODD^k_{yes}$

We have obtained a regulated ultrametric automaton with two times fewer states than a probabilistic or two-way nondeterministic automaton. A deterministic automaton with one counter can recognize $EVENODD^k_{yes}$ with nearly the same state complexity advantages. We can use the same cycle of length 2^k, increase the value of the counter to 1 as it passes the beginning of the cycle, and reduce the value back to zero if it was 1. After reading the input word, we can determine whether the automaton has gone through the cycle an even number of times or not.

We can do this job much better with non-regulated ultrametric automata.

Theorem 8. *There exists a language, which requires at least $k+1$ state for alternating automata, but can be recognized by 2-ultrametric integral automaton with two states.*

Proof. We can construct an automaton like that in the proof of Theorem 4. The accepting state will have a beginning amplitude of zero and reading letter a will add 1 to the amplitude. The amplitude will be divisible by 2^{k+1} if and only if the input word belongs to $EVENODD^k_{yes}$. An input word will be accepted if 2-norm does not exceed $1/2^{k+1}$. An empty input word will be accepted, because 2-norm zero is less than $1/2^{k+1}$.

The language $EVENODD^k_{yes}$ requires at least $k+1$ states to be recognized by an alternating automaton [14]. □

4 Two-way Ultrametric Automata

Two-way ultrametric finite automata can go through the input word in both directions. This gives them the ability to read an input word many times and to read an input word from right to left. Before the beginning of the input word there is a left end-marker ⊢ on the input tape and after the end of the input word there is a right end-marker ⊣. Two-way finite automata are like Turing machines with one tape containing an input word. The aforementioned tape is read-only [15].

The ability to read an input word in both directions makes it easier to create an algorithm and to understand such an algorithm. For example, to check if the word is a palindrome, we will not need to check the positions of the input letters and their reverse positions. We can just compare the positions of the symbols reading word in one and another direction. We will look at palindromes in a binary alphabet.

Theorem 9. *For every prime number p palindromes of a binary alphabet can be recognized by a one-way integral p-ultrametric automaton with 4 states.*

Proof. The constructed p-ultrametric automaton is shown in Fig. 6, q is a prime number, and $q \neq p$. Assume that the automaton has read an input word of length n. The top two states ensure that $\sum_{i=0}^{n-1} a_i * q^{2*i-n+1}$ will be added to the amplitude of the accepting state, where $a_i = 1$ if i-th input symbol was 1. The bottom two states ensure that $\sum_{i=0}^{n-1} a_i * q^{n-1-2*i}$ will be subtracted from the amplitude of the accepting state. Both values will be equal if and only if the input word is a palindrome. Therefore, the amplitude of an accepting state will be equal to zero if the input word is a palindrome. Therefore, the acceptance condition for p-norm will be ≤ 0. □

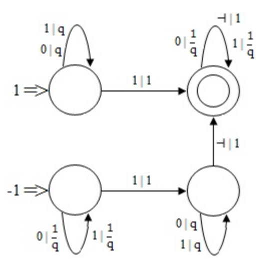

Fig. 6. A one-way p-ultrametric automaton recognizing palindromes

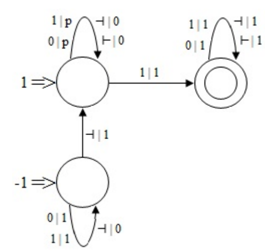

Fig. 7. A two-way p-ultrametric automaton recognizing palindromes

Now we will reduce the number of states by allowing the automaton to read the input word in both directions.

Theorem 10. *For every prime number p palindromes of a binary alphabet can be recognized by a two-way integral p-ultrametric automaton with 3 states.*

Proof. We will construct an automaton like that in the proof of Theorem 9, but this time we will save one state. The constructed p-ultrametric automaton is shown in Fig. 7. Here we obtain the following check: $\sum_{i=0}^{n-1} a_i * p^i = \sum_{i=0}^{n-1} a_i * p^{n-1-i}$. It is a more direct and natural check: the input word should be the same reading it in both directions. We have obtained a two-way p-ultrametric automaton with 3 states. □

To conclude that two-way ultrametric automata require fewer states to recognize binary palindromes than one-way ultrametric automata, we have to prove that one-way ultrametric automata cannot have less than 4 states.

Theorem 11. *For every prime number p, a one-way p-ultrametric automaton cannot recognize binary palindromes with less than 4 states.*

Proof (idea). It is not enough to have one state to remember the sequence of input symbols. In [9] it is said that with the increase of the number of input symbols, the p-norm of one state monotonically increases or decreases, or does not change. This does not give us the ability to remember the positions of the symbols, only the number of symbols. Therefore, to remember the positions of the required symbols at least two states are required.

We have to remember the positions of the input word's symbols and the reverse positions of the input word's symbols, because we can read the input word only once in one direction. Otherwise, an incorrect word could be accepted. To remember the positions of the symbols in one direction we need at least two states. To remember the reverse positions of symbols we also require two states and these states cannot be the same as the previously mentioned two, otherwise we could distract the remembrance of the positions of symbols. That makes four states together. The aforementioned restrictions do not depend on the chosen prime number p and even on the fact that the automaton can use not only p-adic integers. □

Theorem 11 allows us to say that two-way integral ultrametric automata require fewer states to recognize binary palindromes than one-way unresricted ultrametric automata.

5 Summary

Regulated ultrametric automata can have exponential state complexity advantages over deterministic automata. However, some results used only abilities that are available to other types of automata, like alternating and counter automata. We have obtained new results about the complexity advantages of ultrametric automata. There exists a language, whose recognition by nondeterministic automata requires an exponential number of states (any arbitrary positive integer can be chosen for the base of exponent), while an ultrametric integral automaton can have constant state complexity. Another language has shown exponential state complexity for deterministic automata while an integral ultrametric automaton has only two states. Two states of integral ultrametric automata are able to recognize languages that require linear state complexity for probabilistic and two-way nondeterministic automata or logarithmic state complexity for alternating automata.

Two-way ultrametric automata can increase the advantages of the state complexity of ultrametric automata. It is possible to show that in some cases two-way ultrametric automata require fewer states than one-way ultrametric automata. In the case of the language of binary palindromes, it is three states instead of four.

References

1. Khrennikov, A.Y.: NonArchimedean Analysis: Quantum Paradoxes, Dynamical Systems and Biological Models. Kluwer Academic Publishers (1997)
2. Dragovich, B., Dragovich, A.: A p-adic Model of DNA Sequence and Genetic Code. In: p-adic Numbers, Ultrametric Analysis, and Applications, vol. 1(1), pp. 34–41 (2009)
3. Vladimirov, V.S., Volovich, I.V., Zelenov, E.I.: P-Adic Analysis and Mathematical Physics, World Scientific (1995)
4. Freivalds, R.: Ultrametric automata and Turing machines. In: Voronkov, A. (ed.) Turing-100. EPiC Series, vol. 10, pp. 98–112, EasyChair (2012)
5. Madore D.A.: A first introduction to p-adic numbers (2000) `http://www.madore.org/~david/math/padics.pdf`
6. Balodis, K., Beriņa, A., Cīpola, K., Dimitrijevs, M., Iraids, J. et al.: On the State Complexity of Ultrametric Finite Automata. In: 39th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2013), Proceedings, vol. 2, pp. 1–9 (2013)
7. Damanik, D.: Finite automata with restricted two-way motion. Master's thesis (in German), J. W. Goethe-Universitaet Frankfurt (1996)
8. Ambainis A., Freivalds R.: 1-Way Quantum Finite Automata: Strengths, Weaknesses and Generalizations. In: Proc. 39th FOCS, pp. 332–341 (1998)
9. Balodis, K.: Counting with Probabilistic and Ultrametric Finite Automata. In: Essays Dedicated to Jozef Gruska on the Occasion of His 80th Birthday, LNCS, pp. 3–16 (2014)
10. Cīpola, K., Pakulis, A., Freivalds, R.: Experiments in complexity of probabilistic and ultrametric automata. In: 41st International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2015), Czech Republic (2015)
11. Ambainis, A.: The complexity of probabilistic versus deterministic finite automata. In: Asano, T., Igarashi, Y., Nagamochi, H., Miyano, S., Suri, S. (eds.) Algorithms and Computation, Lecture Notes in Computer Science, vol. 1178, pp. 233–238, Springer Berlin Heidelberg (1996) `http://dx.doi.org/10.1007/BFb0009499`
12. Ambainis, A., Yakaryilmaz, A.: Superiority of exact quantum automata for promise problems. In: Information Processing Letters, vol. 112(7), pp. 289–291 (2012)
13. Cem Say, A.C., Yakaryilmaz, A.: Quantum finite automata: A modern introduction. In: Essays Dedicated to Jozef Gruska on the Occasion of His 80th Birthday, LNCS, pp. 208–222 (2014)
14. Geffert, V., Yakaryilmaz, A.: Classical Automata on Promise Problems. In: Descriptional Complexity of Formal Systems (DCFS 2014), LNCS, pp. 126–137 (2014)
15. Naseem, H.: Two-way Deterministic Finite Automata. (2DFA) `http://www.slideshare.net/Hafsa.Naseem/twoway-deterministic-finite-automata`