Abstract. The computation width (a.k.a. tree width, a.k.a. leaf size) of a nondeterministic finite automaton (NFA) $A$ counts the number of branches in the computation tree of $A$ on a given input. The deviation number of $A$ on a given input counts the number of nondeterministic paths that branch out from the best accepting computation. Deviation number is a best-case nondeterminism measure closely related to the guessing measure of Goldstine, Kintala and Wotschke (Inform. Comput. 86, 1990, 179–194). We consider the descriptional complexity of NFAs with similar given deviation number and with computation width.

1 Introduction

Different ways to quantify and measure the amount of nondeterminism in finite automata have been considered in the literature. The degree of ambiguity counts the number of accepting computations and is possibly the most well studied measure \[7\]. Other measures are based on the amount of nondeterminism used in all (accepting as well as non-accepting) computations, and further distinctions arise depending on whether the measure is a best-case or a worst-case measure \[4,12\].

The computation width of an NFA measures the width (i.e., the number of leaves) of the computation tree on a given input. This measure has been previously studied under the name ‘tree width’ \[11\] or ‘leaf size’ \[2,7\]. On the other hand, the guessing measure of an NFA \[5\] counts the number of bits the automaton needs to make the nondeterministic choices on the “best” accepting path, that is, the path using the least amount of nondeterminism.

The computation width measures the amount of nondeterminism in all possible computations, while guessing is a best case measure that limits the amount of nondeterminism only on a best accepting computation. If the computation width of an NFA $A$ is $k$ then also the guessing of $A$ is at most $k - 1$ but, in general, the guessing of $A$ can be much smaller. In particular, it is possible that an NFA $A$ has finite guessing but the computation width of $A$ is unbounded.

In this paper we study the descriptional complexity trade-offs between NFAs of finite computation width and NFAs where the amount of nondeterminism of only the best accepting computation is bounded. If the minimal NFA for a regular language $L$ has to make a sequence of binary nondeterministic choices where always one of the choices leads to failure (without further nondeterminism),
then the computation width is equivalent to a best case computation measure. We provide an example where this situation occurs. Our main goal is to provide a construction of the opposite situation where a given limit $k$ on a best case computation measure allows to have a much smaller NFA than the same limit $k$ on the computation width.

As a best case computation measure we consider deviation number that counts the number of nondeterministic paths that branch out from a best accepting computation. Deviation number is closely related to the guessing measure [5] and, if all nondeterministic steps of an NFA $A$ have exactly two choices, then the deviation number of $A$ is always exactly the guessing of $A$. For a descriptional complexity comparison with computation width, we feel that deviation number is a more natural best case measure than guessing. For example, the guessing (as defined in [5]) involved in one transition involving three possible choices is less than the guessing of a computation where we achieve three choices by first making a binary choice between states $q_1$ and $q_2$ and then another binary choice in state $q_2$.

The definition of the guessing measure is natural and it guarantees that the guessing of an NFA $A$ is always the logarithm of the multiplicative branching measure of $A$ [5], however, for our purposes deviation number is more suitable as it directly counts the number of branches the best accepting computation has.

We note that there are known examples of NFAs with finite deviation number where determinization causes a super-polynomial size blow-up. This follows from Theorem 5.4 of [5] by observing that the guessing of an NFA is finite if and only if its deviation number is finite. At first sight the above could seem to yield an example where an NFA with given deviation number is significantly smaller than the minimal equivalent NFA with the same computation width because it is known that determinizing an NFA with finite computation width causes only a polynomial size blow-up [11]. However, the above does not directly imply a size difference between NFAs with finite deviation number and finite computation width, because in the latter result the degree of the polynomial depends on the computation width (and, in fact, the NFAs used in [5] for the super-polynomial size blow-up have the same finite computation width and deviation number). It seems possible that for the language family $L_n$, $n \geq 1$, used in the proof of Theorem 5.4 of [5] one could construct NFAs with suitably chosen finite deviation number that are smaller than any NFAs of the same computation width, but determining the size difference would require similar, and likely more complicated, estimations than what we use below in Section 4.

In our construction, to obtain worst-case size blow-up from an NFA with given deviation number to an NFA with the same computation width we want to have a regular language $L$ such that any minimal NFA has to make all nondeterministic choices in the beginning. In this case by making the initial nondeterministic choices as a balanced binary tree we obtain an NFA with deviation number $\ell$ (where $\ell$ is roughly the logarithm of the number of nondeterministic choices) that is not much larger than the minimal NFA for $L$ while, at least intuitively,