Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Erratum

Erratum to "The computational power of Benenson automata" [Theoret. Comput. Sci. 344 (2005) 279–297]

David Soloveichik[a,*], Erik Winfree[b,1]

[a] University of Washington, AC101, Paul G. Allen Center, Box 352350, 185 Stevens Way, Seattle, WA 98195, United States

[b] California Institute of Technology, MC 136-93 1200, E California Blvd, Pasadena, CA 91125, United States

ARTICLE INFO

Article history:
Received 21 February 2011
Accepted 25 April 2011
Communicated by G. Rozenberg

(i)

We erroneously claim that Lemma 5.1 applies to both non-deterministic and deterministic Benenson automata, while it applies only to deterministic ones. The following corrected lemma holds for both deterministic and non-deterministic Benenson automata. It also exponentially improves on the dependence on D for both for circuit depth and size compared to the paper version. Note that compared with the paper version, the circuit depth has an extra logarithmic dependence on n, and the circuit size has an extra linear dependence on n. Theorem 3.2 follows from Lemma 5.1(corrected) as before, by taking $D = O(1)$, $S = O(\log n)$, and $L = \text{poly}(n)$.

Lemma 5.1 (*Corrected*)**.** *A function $f : \{0, 1\}^n \to \{0, 1\}$ computed, possibly non-deterministically, by a Benenson automaton $(S, D, L, \Sigma, n, \sigma, \mathcal{R})$ can be computed by a $O(\log(L/D)\log D + \log^2 D + \log n)$ depth, $O(LD^2 \log D + LDn)$ size circuit.*

The proof of Lemma 5.1(corrected) proceeds as in the paper with the following exceptions. In contrast to what is stated in the paper, when the given Benenson automaton is non-deterministic, in order to generate the initial series of matrices $T_{1,2}(x), \ldots, T_{Q-1,Q}(x)$, each gadget A_q may require more than $2D$ bits of input. Indeed, whether any cutting rule applies at the exposed sticky end may depend on all n inputs. In matrix $T_{q,q'}(x)$ the bit at row j, column h (both 0-indexed) is 1 iff $\sigma[(q-1)D + j] \to_x^* \sigma[(q'-1)D + h]$.[2] For non-deterministic Benenson automata we can construct the initial matrices $T_{1,2}(x), \ldots, T_{Q-1,Q}(x)$ as follows. In each gadget A_q, first we construct a $D \times D$ binary matrix M_q which captures one-step cuts only: the bit at row j, column h (both 0-indexed) is 1 iff $\sigma[(q-1)D + j] \to_x \sigma[(q-1)D + h]$. To compute a bit of M_q requires a $O(\log n)$-depth and $O(n)$-size tree of ORs since there could be $O(n)$ relevant cutting rules for non-deterministic automata. Now note that using matrix multiplication where $+$ is logical OR and $\cdot$ is logical AND, and I is the identity matrix, $(I + M_q)^t$ is the reachability matrix after up to t cuts. Thus $T_{q,q+1} = (I + M_q)^{2D}$ since at most $2D$ cuts are possible to get from the beginning of one segment to the end of the following one. We can compute $(I + M_q)^{2D}$ by squaring $O(\log D)$ times using the same construction as for gadgets B, each of which has depth $O(\log D)$ and size $O(D^3)$. This results in gadgets A of depth $O(\log^2 D + \log n)$ and size $O(D^3 \log D + D^2 n)$. With the rest of the construction as in the paper, the total circuit depth is $O(\log(L/D)\log D + \log^2 D + \log n)$ and size $O(LD^2 \log D + LDn + LD^2) = O(LD^2 \log D + LDn)$ as desired.

DOI of original article: 10.1016/j.tcs.2005.07.027.

* Corresponding author. Tel.: +1 310 293 9594; fax: +1 206 543 3842.
E-mail addresses: dsolov@u.washington.edu (D. Soloveichik), winfree@caltech.edu (E. Winfree).

1 Tel.: +1 626 395 6246; fax: +1 626 584 0630.

2 There is a typographical error on page 294 in the expression defining $T_{q,q'}$, neglecting to subtract 1 from q, q'. The correct expression is as given here.

doi:10.1016/j.tcs.2011.04.044

(ii)

Section 5 makes two claims which should not be interpreted more broadly than is justified by the proofs in the paper; specifically: "...allowing non-determinism does not increase the computational power...", and "...increasing the sticky end size to be larger than $O(\log n)$ does not increase computational power". The correct sense is as expressed in Section 3: Benenson automata with sticky end size $S = O(\log n)$, cutting range $D = O(1)$, and state string size $L = \text{poly}(n)$ compute the same class of families of functions as $O(\log n)$-depth circuits, and allowing non-determinism or larger sticky ends does not expand this class. However, it may be possible that non-determinism or larger sticky ends add a lot of computational power if we allow a cutting range $D = O(n)$, for example.

Acknowledgement

We thank Semih Salihoglu for bringing these issues to our attention.