On Greedy Trie Execution

Zbigniew Gołębiewski and Filip Zagórski[†]

Faculty of Fundamental Problems of Technology, Wrocław University of Technology

In the paper "How to select a looser" Prodinger was analyzing an algorithm where n participants are selecting a leader by flipping fair coins, where recursively, the 0-party (those who i.e. have tossed heads) continues until the leader is chosen.

We give an answer to the question stated in the Prodinger's paper – what happens if not a 0-party is recursively looking for a leader but always a party with a smaller cardinality. We show the lower bound on the number of rounds of the greedy algorithm (for fair coin).

Keywords: Trie, leader election

1 Introduction

Incomplete $Trie$ is a structure that corresponds to executions of the following leader election algorithm (in the paper we will call it $Trie$ algorithm). A group of n players flip coins, and recursively, the 0-party (those who i.e. have tossed heads) continues until the leader is chosen (there is only one player who tosses heads).

In [7], Prodinger analyzed i.e. the expected number of rounds of the $Trie$ algorithm for the unbiased coins (i.e. the average depth of a random incomplete trie). He showed that if at the beginning of the protocol there were n players then the process stops on average after $\log_2 n + \frac{1}{2} - \delta_2(\log_2 n)$ rounds, where $\delta_2(\log_2 n)$ is the periodic function (and very small amplitude). That paper started a series of works that analyze various modifications and properties of tries ([2, 3, 4, 5, 6]).

Our result This work focuses on the analysis of the *Greedy*$Trie$ algorithm – the algorithm starts with n active players, in each round all active players toss a coin. If all players throw heads (or all players flip tails) then all players remain active and participate in the next round. If it is not the case then all members of a group with a larger cardinality becomes inactive. If both groups have the same cardinality, then those who have tossed tails become inactive. Algorithm is completed when there is only one active player left.

Let us notice that *Greedy*$Trie$ algorithm corresponds to the greedy process of walking from a root of a binary trie towards its leaves – only a branch with smaller weight is developed.

One can observe that the natural lower bound of the running time of *Greedy*$Trie$ algorithm is the length of the shortest path from root to leaf in binary trie. Devroye showed in [1] that the length of the shortest path equals to $\lg n - \lg \log n + O(1)$.

[†]Email: `{Zbigniew.Golebiewski, Filip.Zagorski}@pwr.wroc.pl`

We answer to the question formulated in [7] – what is the expected number of rounds of the greedy execution of $Trie$ algorithm (expected running time of *Greedy*$Trie$). We show that *Greedy*$Trie$ gains only a constant number of rounds (in terms of average running time) when compared to $Trie$.

2 Analysis

Let $X_k(n)$ be an indicator of a random variable corresponding to a number of participants of the next round, assuming that at the beginning of a current round there are n players. Let T_n be a random variable denoting a running time of the *Greedy*$Trie$ algorithm starting with n participants, then $T_1 = 0$ and $T_n = \sum_{k=1}^{\lfloor \frac{n}{2} \rfloor} X_k(n)\,(T_k + 1) + X_n(n)(T_n + 1)$ and therefore

$$\mathbf{E}\left[T_n\right] = \frac{1}{1 - \mathbf{E}\left[X_n\right]} \left(\sum_{k=1}^{\lfloor \frac{n}{2} \rfloor} \mathbf{E}\left[X_k\right] \mathbf{E}\left[T_k\right] + 1 \right) . \tag{1}$$

Theorem 1 *Let T_n be a random variable denoting a running time of the Greedy$Trie$ algorithm starting with n participants then*

$$\mathbf{E}\left[T_n\right] \geq \lg(n) - C\ , \tag{2}$$

where $C \in \mathbf{R}$.

The analysis of the lower bound of $\mathbf{E}\left[T_n\right]$ is based on elementary technique, i.e. inductive proof with usage of some technical lemmas, but it works here well. It is worthwhile to mention that precise analysis of the obtained recurrence equation 1 seems to be very hard because the recurrence sum that one have to handle is a "half" binomial recurrence sum i.e. upper limit of the sum equals $\lfloor \frac{n}{2} \rfloor$. It seems that the complex analysis tools like the "depoissonization", the mellin transform and other methods based on the generating functions fails here.

Acknowledgements

The paper was supported by grant N N206 369739 of the Polish Ministry of Science and Higher Education.

References

[1] Luc Devroye. A note on the probabilistic analysis of patricia trees. *Random Struct. Algorithms*, 3(2):203–214, 1992.

[2] James Allen Fill, Hosam M. Mahmoud, and Wojciech Szpankowski. On the distribution for the duration of a randomized leader election algorithm. *Ann. Appl. Probab*, 6:1260–1283, 1996.

[3] Svante Janson and Wojciech Szpankowski. Analysis of an asymmetric leader election algorithm. *Electr. J. Comb.*, 4(1), 1997.

[4] Charles Knessl. A note on the asymptotic behavior of the depth of tries. *Algorithmica*, 22(4):547–560, 1998.

[5] Guy Louchard and Helmut Prodinger. The asymmetric leader election algorithm: Another approach. *Annals of Combinatorics*, 12(4):449–478, apr 2009.

[6] Conrado Martinez, Guy Louchard, and Helmut Prodinger. The swedish leader election protocol: Analysis and variations. In *ANALCO*, pages 127–134, 2011.

[7] Helmut Prodinger. How to select a loser. *Discrete Mathematics*, 120(1-3):149–159, 1993.