Brief Announcement: Give Me Some Slack: Efficient Network Measurements

Ran Ben Basat
Technion, Haifa, Israel
sran@cs.technion.ac.il

Gil Einziger
Nokia Bell Labs, Kfar Saba, Israel
gil.einziger@nokia.com

Roy Friedman
Technion, Haifa, Israel
roy@cs.technion.ac.il

Abstract

Many networking applications require timely access to recent network measurements, which can be captured using a sliding window model. Maintaining such measurements is a challenging task due to the fast line speed and scarcity of fast memory in routers. In this work, we study the impact of allowing *slack* in the window size on the asymptotic requirements of sliding window problems. That is, the algorithm can dynamically adjust the window size between W and $W(1+\tau)$ where τ is a small positive parameter. We demonstrate this model's attractiveness by showing that it enables efficient algorithms to problems such as Maximum and General-Summing that require $\Omega(W)$ bits even for constant factor approximations in the exact sliding window model. Additionally, for problems that admit sub-linear approximation algorithms such as Basic-Summing and Count-Distinct, the slack model enables a further asymptotic improvement.

The main focus of our paper [4] is on the widely studied Basic-Summing problem of computing the sum of the last W integers from $\{0, 1 \ldots, R\}$ in a stream. While it is known that $\Omega(W \log R)$ bits are needed in the exact window model, we show that approximate windows allow an *exponential* space reduction for constant τ.

Specifically, for $\tau = \Theta(1)$, we present a space lower bound of $\Omega(\log(RW))$ bits. Additionally, we show an $\Omega(\log(W/\epsilon))$ lower bound for $RW\epsilon$ additive approximations and a $\Omega(\log(W/\epsilon) + \log\log R)$ bits lower bound for $(1+\epsilon)$ multiplicative approximations. Our work is the first to study this problem in the exact and additive approximation settings. For all settings, we provide memory optimal algorithms that operate in *worst case* constant time. This strictly improves on the work of [12] for $(1+\epsilon)$-multiplicative approximation that requires $O(\epsilon^{-1}\log(RW)\log\log(RW))$ space and performs updates in $O(\log(RW))$ worst case time. Finally, we show asymptotic improvements for the Count-Distinct, General-Summing and Maximum problems.

2012 ACM Subject Classification Networks → Network measurement

Keywords and phrases Streaming, Algorithms, Sliding window, Lower bounds

Digital Object Identifier 10.4230/LIPIcs.ICALP.2018.163

Related Version A full version of the paper is available at [4], https://arxiv.org/abs/1703.01166.

45th International Colloquium on Automata, Languages, and Programming (ICALP 2018).
Editors: Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella;
Article No. 163; pp. 163:1–163:5

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

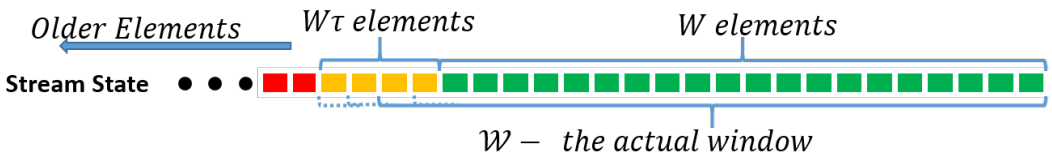

Figure 1 We need to answer each query with respect to a τ-slack window that must include the last W items, but may or may not consider a suffix of the previous $W\tau$ elements.

1 Introduction

Network algorithms in diverse areas such as traffic engineering, load balancing and quality of service [1, 8, 17, 20, 27] rely on timely link measurements. In such applications recent data is often more relevant than older data, motivating the notions of *aging* and *sliding window* [5, 10, 13, 21, 23]. For example, a sudden decrease in the average packet size on a link may indicate a SYN attack [22]. Additionally, a load balancer may benefit from knowing the current utilization of a link to avoid congestion [1].

While conceptually simple, conveying the necessary information to network algorithms is a difficult challenge due to current memory technology limitations. Specifically, DRAM memory is abundant but too slow to cope with the line rate while SRAM memory is fast enough but has a limited capacity [9, 11, 25]. Online decisions are therefore realized through space efficient data structures [6, 7, 14, 15, 3, 19, 24, 26] that store measurement statistics in a concise manner. For example, [14, 24] utilize probabilistic counters that only require $O(\log\log N)$ bits to approximately represent numbers up to N. Others conserve space using variable sized counter encoding [15, 19] and monitoring only the frequent elements [5].

Basic-Summing is one of the most basic textbook examples of such approximated sliding window stream processing problems [12]. In this problem, one is required to keep track of the sum of the last W elements, when all elements are non-negative integers in the range $\{0, 1, \ldots, R\}$. The work in [12] provides a $(1+\epsilon)$-multiplicative approximation of this problem using $O\left(\frac{1}{\epsilon}\cdot\left(\log^2 W + \log R\cdot(\log W + \log\log R)\right)\right)$ bits. The amortized time complexity is $O(\frac{\log R}{\log W})$ and the worst case is $O(\log W + \log R)$. In contrast, we previously showed an $RW\epsilon$-additive approximation with $\Theta\left(\frac{1}{\epsilon} + \log W\epsilon\right)$ bits [2].

Sliding window counters (approximated or accurate) require asymptotically more space than plain stream counters. Such window counters are prohibitively large for networking devices which already optimize the space consumption of plain counters.

This paper explores the concept of *slack*, or approximated sliding window, bridging this gap. Figure 1 illustrates a "window" in this model. Here, each query may select a *τ-slack window* whose size is between W (the green elements) and $W(1+\tau)$ (the green plus yellow elements). The goal is to compute the sum with respect to this chosen window.

Slack windows were also considered in previous works [12, 23] and we call the problem of maintaining the sum over a slack window Slack Summing. Datar et al. [12] showed that constant slack reduces the required memory from $O(\frac{1}{\epsilon}\cdot\left(\log^2 W + \log R\cdot(\log W + \log\log R)\right))$ to $O(\epsilon^{-1}\log(RW)\log\log(RW))$. For τ-slack windows they provide a $(1+\epsilon)$-multiplicative approximation using $O(\epsilon^{-1}\log(RW)(\log\log(RW) + \log\tau^{-1}))$ bits.

Our Contributions

Our paper [4] studies the space and time complexity reductions that can be attained by allowing *slack* – an error in the window size. Our results demonstrate exponentially smaller and asymptotically faster data structures compared to various problems over exact windows.

Table 1 Comparison of Basic-Summing algorithms. Our contributions are in bold. All algorithms process elements in constant time except for the rightmost column where both update in $O(\log(RW))$ time. We present matching lower bounds to all our algorithms.

	Exact Sum	Additive Error	Multiplicative Error	
$\tau = \Theta(1)$	$\mathbf{\Theta(\log(RW))}$	$\mathbf{\Theta(\log(W/\epsilon))}$	$\mathbf{\Theta(\log(W/\epsilon) + \log\log R)}$	$O(\epsilon^{-1}\log(RW)\log\log(RW))$ [12]
Exact Window	$\Theta(W \log R)$	$\Theta(\epsilon^{-1} + \log W)$ [2]	$O(\epsilon^{-1}\log^2(RW))$ [18]	$O(\epsilon^{-1}\log RW \log(W \log R))$ [12]

We start with deriving lower bounds for three variants of the Basic-Summing problem – when computing an exact sum over a slack window, or when combined with an additive and a multiplicative error in the sum. We present algorithms that are based on dividing the stream into $W\tau$-sized blocks. Our algorithms sum the elements within each block and represent each block's sum in a cyclic array of size τ^{-1}. We use multiple compression techniques during different stages to drive down the space complexity. The resulting algorithms are space optimal, substantially simpler than previous work, and reduce update time to $O(1)$.

For exact Slack Summing, we present a lower bound of $\Omega(\tau^{-1}\log(RW\tau))$ bits. For $(1+\epsilon)$ multiplicative approximations we prove an $\Omega\big(\log(W/\epsilon) + \tau^{-1}(\log(\tau/\epsilon) + \log\log(RW))\big)$ bits bound when $\tau = \Omega\left(\frac{1}{\log RW}\right)$. We show that $\Omega(\tau^{-1}\log\lfloor 1+\tau/\epsilon\rfloor + \log(W/\epsilon))$ bits are required for $RW\epsilon$ additive approximations.

Next, we introduce algorithms for the Slack Summing problem, which asymptotically reduce the required memory compared to the sliding window model. For the exact and additive error versions of the problem, we provide memory optimal algorithms. In the multiplicative error setting, we provide an $O\big(\tau^{-1}\left(\log\epsilon^{-1} + \log\log(RW\tau)\right) + \log(RW)\big)$ space algorithm. This is asymptotically optimal when $\tau = \Omega(\log^{-1} W)$ and $R = \text{poly}(W)$. It also asymptotically improves [12] when $\tau^{-1} = o(\epsilon^{-1}\log(RW))$. We further provide an asymptotically optimal solution for constant τ, even when $R = W^{\omega(1)}$. All our algorithms are deterministic and operate in worst case constant time. In contrast, the algorithm of [12] works in $O(\log RW)$ worst case time.

To exemplify our results, consider monitoring the average bandwidth (in bytes per second) passed through a router in a 24 hours window, i.e., $W \triangleq 86400$ seconds. Assuming we use a 100GbE fiber transceiver, our stream values are bounded by $R \approx 2^{34}$ bytes. If we are willing to withstand an error of $\epsilon = 2^{-20}$ (i.e., about $16KBps$), the work of [2] provides an additive approximation over the sliding window and requires about 120KB. In contrast, using a 10 minutes slack ($\tau \triangleq \frac{1}{144}$), our algorithm for **exact** Slack Summing requires only 800 bytes, 99% less than approximate summing over exact sliding window. For the same slack size, the algorithm of [12] requires more space than our **exact** algorithm even for a large 3% error. Further, if we also allow the same additive error ($\epsilon = 2^{-20}$), we provide an algorithm that requires only 240 bytes - a reduction of more than 99.8% !

Table 1 compares our results for the important case of constant slack with [12]. As depicted, our *exact* algorithm is faster and more space efficient than the multiplicative approximation of [12]. Comparing our multiplicative approximation algorithm to that of [12], we present *exponential* space reductions in the dependencies on ϵ^{-1} and R, with an asymptotic reduction in W as well. We also improve the update time from $O(\log(RW))$ to $O(1)$.

Finally, we apply the slack window approach to multiple streaming problems, including Maximum, General-Summing, Count-Distinct and Standard-Deviation. We show that, while some of these problems cannot be approximated on an exact window in sub-linear space (e.g. Maximum and General-Summing), we can easily do so for slack windows. In the count distinct problem, a constant slack yields an asymptotic space reduction over [10, 16]. The full version of our paper appears in [4].

References

1 Mohammad Alizadeh, Tom Edsall, Sarang Dharmapurikar, Ramanan Vaidyanathan, Kevin Chu, Andy Fingerhut, Vinh The Lam, Francis Matus, Rong Pan, Navindra Yadav, and George Varghese. Conga: Distributed congestion-aware load balancing for datacenters. In *ACM SIGCOMM 2014 Conference, SIGCOMM'14, Chicago, IL, USA, August 17-22, 2014*, ACM SIGCOMM 2014, 2014. doi:10.1145/2619239.2626316.

2 Ran Ben Basat, Gil Einziger, Roy Friedman, and Yaron Kassner. Efficient Summing over Sliding Windows. In *SWAT*, 2016.

3 Ran Ben Basat, Gil Einziger, Roy Friedman, Marcelo Caggiani Luizelli, and Erez Waisbard. Constant time updates in hierarchical heavy hitters. In *ACM SIGCOMM*, 2017.

4 R. Ben Basat, G. Einziger, and R. Friedman. Give Me Some Slack: Efficient Network Measurements. *ArXiv e-prints*, 2017. arXiv:1703.01166.

5 Ran Ben-Basat, Gil Einziger, Roy Friedman, and Yaron Kassner. Heavy hitters in streams and sliding windows. In *IEEE INFOCOM*, 2016.

6 Ran Ben-Basat, Gil Einziger, Roy Friedman, and Yaron Kassner. Optimal elephant flow detection. In *IEEE INFOCOM*, 2017.

7 Ran Ben-Basat, Gil Einziger, Roy Friedman, and Yaron Kassner. Randomized admission policy for efficient top-k and frequency estimation. In *IEEE INFOCOM*, 2017.

8 Theophilus Benson, Ashok Anand, Aditya Akella, and Ming Zhang. Microte: Fine grained traffic engineering for data centers. In *ACM CoNEXT*, 2011.

9 Flavio Bonomi, Michael Mitzenmacher, Rina Panigrahy, Sushil Singh, and George Varghese. An improved construction for counting bloom filters. In *ESA*, 2006.

10 Y. Chabchoub and G. Hebrail. Sliding hyperloglog: Estimating cardinality in a data stream over a sliding window. In *2010 IEEE ICDM Workshops*, 2010.

11 Min Chen and Shigang Chen. Counter tree: A scalable counter architecture for per-flow traffic measurement. In *IEEE ICNP*, 2015.

12 Mayur Datar, Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Maintaining stream statistics over sliding windows. *SIAM Journal of Computing*, 2002.

13 G. Einziger and R. Friedman. TinyLFU: A highly efficient cache admission policy. In *PDP 2014*. IEEE Computer Society, 2014. doi:10.1109/PDP.2014.34.

14 Gil Einziger, Benny Fellman, and Yaron Kassner. Independent counter estimation buckets. In *IEEE INFOCOM*, 2015.

15 Gil Einziger and Roy Friedman. Counting with TinyTable: Every Bit Counts! In *ICDCN 2016*. ACM, 2016. doi:10.1145/2833312.2833449.

16 Éric Fusy and Frécéric Giroire. Estimating the number of active flows in a data stream over a sliding window. In *ANALCO*, 2007.

17 Pedro Garcia-Teodoro, Jesús E. Díaz-Verdejo, Gabriel Maciá-Fernández, and E. Vázquez. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers and Security*, 2009.

18 Phillip B. Gibbons and Srikanta Tirthapura. Distributed streams algorithms for sliding windows. In *SPAA*, 2002. doi:10.1145/564870.564880.

19 Nan Hua, Bill Lin, Jun (Jim) Xu, and Haiquan (Chuck) Zhao. Brick: A novel exact active statistics counter architecture. In *ACM/IEEE ANCS*, 2008.

20 Abdul Kabbani, Mohammad Alizadeh, Masato Yasuda, Rong Pan, and Balaji Prabhakar. Af-qcn: Approximate fairness with quantized congestion notification for multi-tenanted data centers. In *IEEE HOTI*, 2010.

21 Yang Liu, Wenji Chen, and Yong Guan. Near-optimal approximate membership query over time-decaying windows. In *IEEE INFOCOM*, 2013.

22 B. Mukherjee, L.T. Heberlein, and K.N. Levitt. Network intrusion detection. *Network, IEEE*, 1994.

23 Moni Naor and Eylon Yogev. Sliding bloom filters. In *ISAAC*. Springer, 2013. doi:10.1007/978-3-642-45030-3_48.

24 Erez Tsidon, Iddo Hanniel, and Isaac Keslassy. Estimators also need shared values to grow together. In *IEEE INFOCOM*, 2012.

25 Hao Wang, H. Zhao, Bill Lin, and Jun Xu. Dram-based statistics counter array architecture with performance guarantee. *IEEE/ACM Transactions on Networking*, 2012.

26 Li Yang, Wu Hao, Pan Tian, Dai Huichen, Lu Jianyuan, and Liu Bin. Case: Cache-assisted stretchable estimator for high speed per-flow measurement. In *IEEE INFOCOM*, 2016.

27 L. Ying, R. Srikant, and X. Kang. The power of slightly more than one sample in randomized load balancing. In *IEEE INFOCOM*, 2015.