Semi-Group Range Sum Revisited: Query-Space Lower Bound Tightened

Xiaocheng Hu† Yufei Tao† Yi Yang‡ Shuigeng Zhou‡

†Chinese University of Hong Kong
{xchu,taoyf}@cse.cuhk.edu.hk

‡Fudan University
{yyang1,sgzhou}@fudan.edu.cn

Abstract

Let $\mathcal{D}$ be a set of n elements $e_1, ..., e_n$ drawn from a commutative semigroup. Given two integers x, y satisfying $1 \leq x \leq y \leq n$, a *range sum query* returns the sum of the $y - x + 1$ elements e_x, e_{x+1}, ... , e_y. The goal of indexing is to store $\mathcal{D}$ in a data structure so that all such queries can be answered efficiently in the worst case.

This paper proves a new lower bound in the semigroup model on the tradeoff between space and query time for the above problem. We show that, if the query time needs to be at most an integer t, a structure must use

$$\begin{cases} \Omega\left(n \log_{\overbrace{*...*}^{\lfloor (t-2)/2 \rfloor \text{ stars}}} n\right) & \text{for } t \geq 4 \\ \Omega(n \log \log n) & \text{for } t = 3 \\ \Omega(n \log n) & \text{for } t = 2 \end{cases}$$

space. The bound is asymptotically tight for *every* $t \geq 2$, and is matched by an existing structure. Previously, the best lower bounds either had a substantially smaller non-linear factor [Yao, 1982], or were tight only for constant t [Alon and Schieber, 1987].

Our lower bound is asymptotically tight *bidirectionally*, namely, it also answers the following question: if the space needs to be bounded by an integer m, what is the best query time achievable? The techniques behind our lower bound are drastically different from those of [Yao, 1982] and [Alon and Schieber, 1987], and reveal new insight on the characteristics of the problem.

Keywords: Range Sum Queries, Semi-Group, Lower Bound, Length Decomposition Property

Corresponding Author
Yufei Tao
taoyf@cse.cuhk.edu.hk
Tel: +852-39438437
Fax: +852-26035024

1 Introduction

Let $\mathcal{D}$ be a set of n elements $e_1, ..., e_n$ drawn from a commutative semigroup $(G, \oplus)$. Given two integers x, y satisfying $1 \leq x \leq y \leq n$, a *range sum query*, denoted as $[x, y]$, returns the sum

$$e_x \oplus e_{x+1} \oplus ... \oplus e_y.$$

The goal of indexing is to store $\mathcal{D}$ in a data structure so that all such queries can be answered efficiently in the worst case. This is a fundamental problem in computer science with a huge variety of applications, and has received considerable attention (see Section 1.1 for a list of representative works in theoretical computer science).

We study the problem in the standard *semigroup model* [5]. In this model, a *partial sum* is a pair (D, s), where D is a subset of the input set $\mathcal{D}$, and $s = \oplus_{e \in D} e$ is the sum of those elements in D. A *data structure T of space m and query time t* is defined to be a set of m partial sums satisfying the following condition: for any range sum query $[x, y]$, T has no more than t partial sums $(D_1, s_1), ..., (D_t, s_t)$ such that:

- $D_1, ..., D_t$ are disjoint
- the union of $D_1, ..., D_t$ is exactly $\{e_x, ..., e_y\}$.

The above two properties allow the structure to answer the query by returning $s_1 \oplus s_2 \oplus ... \oplus s_k$.

In this paper, we aim to understand the tradeoff between space m and query time t. Specifically, we want to derive:

1. *Space lower bound Space*(t): the minimum amount of space required to guarantee query time t.

2. *Query lower bound Query*(m): the minimum achievable query time by using m space.

Note that a lower bound of either type implies a lower bound of the other type. For example, the space lower bound *Space*(t) immediately implies that *Query*(m) is at least the smallest t satisfying *Space*$(t) \leq m$.

A Math Convention. Denote by $\mathbb{N}$ the set of natural numbers. Given a function $f : \mathbb{N} \to \mathbb{N}$, we define $f^{(0)}(x) = x$, and $f^{(i+1)}(x) = f(f^{(i)}(x))$ for integer $i \geq 0$. Also, define $f^*(x)$ to be the smallest h such that $f^{(h)}(x) \leq 256$. Finally, for each integer $k \geq 1$, define $f^{\overbrace{*...*}^{k+1 \text{ stars}}}(x) = g^*(x)$ where $g(x) = f^{\overbrace{*...*}^{k \text{ stars}}}(x)$.

1.1 Previous Results

To answer query $[x, x]$ $(1 \leq x \leq n)$, there is no other way but to use the partial sum $(\{e_x\}, e_x)$. Hence, trivially, any structure must store at least n partial sums, regardless of query time. On the other hand, to ensure query time $t = 1$, a structure must store the results of all the $\binom{n}{2}$ possible queries directly, necessitating $\binom{n}{2}$ space.

The query-space tradeoff for $t \geq 2$ is much more elusive. In a classic paper [11], Yao proved a query lower bound *Query*(m), and showed the asymptotic tightness of his bound by describing a structure that uses m space (for any $m \geq n$), and ensures $O(\textit{Query}(m))$ query time.

What is interesting is the flip side of the coin. To achieve query time t, the aforementioned structure of Yao's demands space

$$\begin{cases} O\left(n \log \overbrace{*\ldots*}^{\lfloor (t-2)/2 \rfloor \text{ stars}} n\right) & \text{for } t \geq 4 \\ O(n \log\log n) & \text{for } t = 3 \\ O(n \log n) & \text{for } t = 2 \end{cases} \tag{1}$$

On the other hand, Yao's query lower bound implies the following space lower bound for $t \geq 2$:

$$\mathit{Space}(t) \quad = \quad \begin{cases} \Omega\left(n \log \overbrace{*\ldots*}^{t-2 \text{ stars}} n\right) & \text{for } t \geq 3 \\ \Omega(n \log\log n) & \text{for } t = 2 \end{cases}$$

Note that there is a (large) gap in the above space lower and upper bounds. Later, Alon and Schieber [1] tightened the lower bound for *constant* t:

$$\mathit{Space}(t) \quad = \quad \begin{cases} \Omega\left(n \log \overbrace{*\ldots*}^{\lfloor (t-2)/2 \rfloor \text{ stars}} n\right) & \text{for } \underline{\underline{\textbf{constant}}}\ t \geq 4 \\ \Omega(n \log\log n) & \text{for } t = 3 \\ \Omega(n \log n) & \text{for } t = 2 \end{cases}$$

Till now, a tight space bound for non-constant t has remained open, in spite of the fact that no structure better than Yao's has ever been proposed.

Range sum queries can be naturally extended to higher dimensions, where the dataset consists of n points in d-dimensional space (for some fixed $d \geq 2$); and each point is associated with a *weight* from a commutative semigroup. Given an axis-parallel rectangle, a query returns the sum of the weights of the points falling in the rectangle. Using a range tree with fanout $\log^{\epsilon} n$ for some small constant $\epsilon > 0$, it is easy to achieve $n \log^{O(1)} n$ space and $O((\log n / \log\log n)^{d-1})$ query time. This already matches a lower bound proven by Chazelle [4].

The range sum problem has also been studied in the dynamic settings. A range tree can be used in this scenario to ensure $O((\log n)^d)$ time per insertion, deletion, and query. From the lower bound side, Fredman [5] showed that, for any structure that supports deletions, the maximum of its insertion, deletion, and query times must be $\Omega((\log n)^d)$. In the semi-dynamic case where only insertions and queries are allowed, Chazelle [4] proved that the maximum of the insertion and query times must be $\Omega((\log n / \log\log n)^d)$. Hampapuram and Fredman [6] improved the lower bound to $\Omega(\log n)$ for the special case $d = 1$, which is tight.

The problem becomes harder when the query range is a d-dimensional half-space or a simplex, where a structure with linear or near linear space typically has query time polynomial in n. See [3, 9] for upper bounds and [2] for lower bounds. Recent studies have also touched upon other computation models, e.g., the group model in [7, 10] and the cell-probe model in [8, 10].

1.2 Our Results

The main result of this paper (Theorem 1) is a new space lower bound (for the range-sum problem) for *all* $t \geq 2$:

$$Space(t) \quad = \quad \begin{cases} \Omega\left(n \log \overbrace{*\ldots*}^{\lfloor (t-2)/2 \rfloor \text{ stars}} n\right) & \text{for } t \geq 4 \\ \Omega(n \log\log n) & \text{for } t = 3 \\ \Omega(n \log n) & \text{for } t = 2 \end{cases} \tag{2}$$

This improves both of the existing space lower bounds [1, 11], and matches the space upper bound as shown in (1).

Furthermore, we prove (Theorem 3) that our space lower bound in (2) also implies the query lower bound (up to a constant factor) established in [11], as long as $2n$ space is provided. In other words, our space-query tradeoff is the first one that is tight *bidirectionally*. As a consequence, now the range sum problem has been fully understood in the semi-group model—both the space and query lower bounds match the guarantees of the structure in [11].

Main Technical Novelty. Our techniques are drastically different from both Yao's [11] and Alon and Schieber's [1]. In fact, our derivation reveals a somewhat surprising property—which we call the *length decomposition property*—inherent in the problem's characteristics:

> **The Length Decomposition Property.** For a partial sum (D, s), let i (or j) be the smallest (or largest, resp.) integer k such that $e_k \in D$; we refer to $j - i + 1$ as the *length* of the partial sum. We show that, for any $t \in [2, n]$, it is *always* possible to divide the length range $[1, n]$ into a set $\mathcal{I}$ of
>
> $$\begin{cases} \Theta\left(\log \overbrace{*\ldots*}^{\lfloor (t-2)/2 \rfloor \text{ stars}} n\right) & \text{for } t \geq 4 \\ \Theta(\log\log n) & \text{for } t = 3 \\ \Theta(\log n) & \text{for } t = 2 \end{cases}$$
>
> disjoint intervals such that a structure must store, for each interval $I \in \mathcal{I}$, $\Omega(n)$ partial sums whose lengths are in I.

The space lower bound in (2) is an immediate corollary of this property. We believe that the property captures the essence of the range sum problem, and is the key reason why our space-query tradeoff is tighter than the previous ones [1, 11].

2 Preliminaries

2.1 A Variant of Ackermann Function and Its Inverse

Let us first define a variant of the Ackermann function. For each integer $t \geq 2$, define function $A_t : \mathbb{N} \to \mathbb{N}$ as

$$A_t(x) \quad = \quad \begin{cases} 0 & \text{if } x = 0 \\ 256 & \text{if } x = 1 \\ 16^{x+1} & \text{if } t = 2 \text{ and } x \geq 2 \\ 4^{4^x} & \text{if } t = 3 \text{ and } x \geq 2 \\ A_{t-2}(A_t(x-1)) & \text{otherwise} \end{cases} \tag{3}$$

It is fundamental to show that, for $t \geq 2$ and $x \geq 0$:

- $A_t(x) \leq A_{t+1}(x)$;
- $A_t(x) < A_t(x+1)$;
- If $x \geq 1$, then $A_t(x) > x$.

Since $A_t(x)$ is strictly ascending with x, we can define its inverse function. For each integer $t \geq 2$, define function $\alpha_t : \mathbb{N} \to \mathbb{N}$ as

$$\alpha_t(x) \quad = \quad \min\{h \in \mathbb{N} \mid A_t(h) \geq x\}.$$

The following facts are also fundamental for $t \geq 2$:

- For any $x \in \mathbb{N}$, $\alpha_t(x) \leq \alpha_t(x+1)$;
- If $x > 0$, then $\alpha_t(x) > 0$;
- If $x \geq 256$, then $\alpha_t(x) < x$.
- $\alpha_t(x)$ has the following (more intuitive) form:

$$\alpha_t(x) \quad = \quad \begin{cases} \Theta\left(\log \overbrace{*\ldots*}^{\lfloor (t-2)/2 \rfloor \text{ stars}} x\right) & \text{for } t \geq 4 \\ \Theta(\log \log x) & \text{for } t = 3 \\ \Theta(\log x) & \text{for } t = 2 \end{cases}$$

Also, there is an important connection between functions $\alpha_{t+2}(x)$ and $\alpha_t^*(x)$:

Lemma 1. *For each $t \geq 2$ and $x > 0$, $\alpha_t^*(x) + 1 \leq \alpha_{t+2}(x) \leq \alpha_t^*(x) + 2$.*

Proof. See appendix. □

2.2 Interval Covers

We say that a set S of intervals *covers* an interval $[x, y]$, if the union of the intervals in S is exactly $[x, y]$. For example, $S = \{[5, 8], [6, 11], [10, 12]\}$ covers $[5, 12]$.

Set n be an integer at least 1. Let $\mathcal{I}$ be a set of integer intervals in the domain from 1 to n. $\mathcal{I}$ is an (n, t)-*cover* if, for every interval $[x, y]$ where x, y are integers satisfying $1 \leq x \leq y \leq n$, there exists a subset $\mathcal{I}(x, y)$ of $\mathcal{I}$ such that (i) $\mathcal{I}(x, y)$ covers $[x, y]$, and (ii) $|\mathcal{I}(x, y)| \leq t$. If multiple subsets exist, then $\mathcal{I}(x, y)$ can be designated to be any of them. We refer to $|\mathcal{I}|$ as the *space* of $\mathcal{I}$.

If there is a range sum structure T on n elements with space m and query time t, then there must exist an (n, t)-cover of space at most m. To see this, simply construct a set $\mathcal{I}$ of intervals from T as follows. For every partial sum $(D, s) \in T$, suppose that i (or j) is the smallest (or largest, resp.) integer k such that element $e_k \in D$; we add the interval $[i, j]$ to $\mathcal{I}$. It is not hard to see that $\mathcal{I}$ is an (n, t)-cover with space at most m.

3 A New Space Lower Bound for Range Sum

This section serves as a proof for our first main result:

Theorem 1. *For any range sum structure on n elements with space m and query time t, where $n \geq 1$ and $t \geq 2$, it holds that $m \geq n \cdot \alpha_t(n)/256$.*[1]

We will instead establish a lower bound on interval covers. Denote by $S_t(n)$ the minimum space of an (n, t)-cover for $t \geq 2$ and $n \geq 1$. We will prove the following lemma, which implies Theorem 1 by the discussion in Section 2.2:

Lemma 2. *For $t \geq 2$ and $n \geq 1$, $S_t(n) \geq \frac{1}{256} n \cdot \alpha_t(n)$.*

Clearly, the only way to cover the query interval $[x, x]$ $(1 \leq x \leq n)$ is to use a singleton set $\{[x, x]\}$. Hence, any (n, t)-cover must contain all the n singleton sets, regardless of t. Therefore, $S_t(n) \geq n$ always holds, making Lemma 2 trivially true when $\alpha_t(n) \leq 256$. Assuming $\alpha_t(n) > 256$, we will first prove the lemma for the base cases $t = 2$ and 3 in Sections 3.1 and 3.2, respectively. Then, we will prove the general case $t \geq 4$ in Section 3.3.

It is worth pointing out that, for constant t, Alon and Schieber [1] have given a lower bound on $S_t(n)$ that is different from ours in Lemma 2 by up to a constant factor (which depends on t). Our proof, however, aims to establish the length decomposition property stated in Section 1.2, for which purpose we must resort to more sophisticated ideas (the argument in [1] fails to prove the property). As we will see in Section 3.3, the property is imperative for proving Theorem 1 at arbitrary (in particular, non-constant) t. Therefore, Sections 3.1 and 3.2 offer the additional benefit of allowing the reader to "warm-up" with simpler cases of $t = 2, 3$ in her/his journey towards understanding the property for any t. Indeed, the property follows from Lemma 3 (for $t = 2$), Lemma 4 (for $t = 3$), and Lemma 6 (for $t \geq 4$).

Terminology. Let us introduce some concepts to facility the presentation. We refer to each integer in $[1, n]$ as a *position*. Given an integer interval $[i, j]$, we define its *length* to be $j - i + 1$, and call i (or j) its *starting* (or *ending*, resp.) position. Recall that, for every interval $[x, y]$ satisfying $1 \leq x \leq y \leq n$, an (n, t)-cover $\mathcal{I}$ must designate a subset $\mathcal{I}(x, y)$ that covers $[x, y]$. We refer to $[x, y]$ as a *query*, and say that the query *consumes* the intervals in $\mathcal{I}(x, y)$.

[1] The constant factor 256 is chosen to simplify the proof. Shrinking its value is possible but out of the scope of this paper.

3.1 Base Case $t = 2$

Let us first prove a crucial lemma:

Lemma 3. *Let $\mathcal{I}$ be an $(n, 2)$-cover where $\alpha_2(n) > 256$, and l an integer in $[1, n/4]$. Then, $\mathcal{I}$ contains at least $n/8$ intervals whose lengths are in the range $[l, 2l - 1]$.*

Proof. We say that an interval $[i, j] \in \mathcal{I}$ is a *target interval* if its length is in $[l, 2l - 1]$; otherwise, $[i, j]$ is a *non-target interval*. A non-target interval $[i, j] \in \mathcal{I}$ is said to be *long* if its length is at least $2l$, or *short* otherwise. For each position $p \in [1, n]$, if there is a target interval starting or ending at p, we say that p is a *marked position*; otherwise, p is an *unmarked position*.

Let $[x, y]$ be a query of length $2l-1$. $|\mathcal{I}(x, y)| \leq 2$ because $t = 2$. On the one hand, $\mathcal{I}(x, y)$ does not contain any long interval, because such an interval is already longer than $[x, y]$. On the other hand, $\mathcal{I}(x, y)$ cannot contain only short intervals, because the total length of two short intervals is at most $2l - 2$. It follows that $\mathcal{I}(x, y)$ contains at least one target interval. Since $|\mathcal{I}(x, y)| \leq 2$, this target interval either starts at x or ends at y. Therefore, either x or y must be a marked position.

The total number of queries of length $2l - 1$ is $n - (2l - 1) + 1 \geq n/2$. Since (i) every such query starts/ends at a marked position but (ii) each marked position can be an endpoint of at most two such queries, there are at least $n/4$ marked positions. As each target interval marks only 2 positions, $\mathcal{I}$ contains at least $n/8$ target intervals. □

To complete the proof of Lemma 2 for $t = 2$, notice that when $\alpha_2(n) > 256$, $\alpha_2(n) = \lceil \log_{16} n \rceil - 1$. Set $h = \lfloor \log_2 n \rfloor - 1$, which is at least $\alpha_2(n)/32$. For each $i \in [0, h - 1]$, define $l_i = 2^i$, which falls in $[1, n/4]$. By Lemma 3, any $(n, 2)$-cover $\mathcal{I}$ must contain at least $n/8$ intervals whose lengths are in the range $[l_i, 2l_i - 1]$, for each $i \in [0, h - 1]$. Since these ranges are disjoint, $\mathcal{I}$ must contain at least $nh/8 \geq n \cdot \alpha_2(n)/256$ intervals.

3.2 Base Case $t = 3$

The next lemma is the counterpart of Lemma 3, but requires a more sophisticated argument.

Lemma 4. *Let $\mathcal{I}$ be an $(n, 3)$-cover where $\alpha_3(n) > 256$, and l an odd integer in $[3, n^{1/4}]$. Then, there are at least $n/32$ intervals in $\mathcal{I}$ whose lengths are in the range $[l, l^2 - 1]$.*

Proof. Set $r = (l - 1)/2 \geq 1$. For each $i \in [1, r]$, let $\mathcal{I}_i$ be the set of the intervals in $\mathcal{I}$ whose lengths are in the range $[(2i - 1)l, (2i + 1)l - 1]$. Also, let $\mathcal{I}_0$ be the set of intervals in $\mathcal{I}$ strictly shorter than length l. Denote by $m_1, ..., m_r$ the sizes of $\mathcal{I}_1, ..., \mathcal{I}_r$, respectively. Since $\mathcal{I}_1, ..., \mathcal{I}_r$ are disjoint, it suffices to show that $\sum_{i=1}^{r} m_i \geq n/32$.

We first show a lower bound on m_1. Let Q_1 be the set of all the queries with length $3l - 1$. Clearly, $|Q_1| = n - (3l - 1) + 1 \geq n/2$. Each query $[x, y] \in Q_1$ must be covered by a set $\mathcal{I}(x, y)$ of at most 3 intervals in $\mathcal{I}_0 \cup \mathcal{I}_1$, since all the other intervals are longer than the query. Since the query has length $3l - 1$, at least one interval in $\mathcal{I}(x, y)$ must have length at least l, meaning that $[x, y]$ consumes at least one interval in $\mathcal{I}_1$. However, as each query in Q_1 has length $3l - 1$ and each interval in $\mathcal{I}_1$ has length at least l, an interval in $\mathcal{I}_1$ can be consumed by at most $(3l - 1) - l + 1 = 2l$ queries of Q_1. Therefore,

$$m_1 \geq \frac{|Q_1|}{2l} \geq \frac{n}{4l}. \tag{4}$$

For each $i \in [2, r]$, let Q_i be the set of all the queries with length $(2i + 1)l - 1$. It holds that $|Q_i| \geq n/2$. Each query $[x, y] \in Q_i$ must be covered by a set $\mathcal{I}(x, y)$ of at most 3 interval in

$\mathcal{I}_0 \cup ... \cup \mathcal{I}_i$. Let Q'_i be the subset of the queries in Q_i that do not consume any intervals from $\mathcal{I}_i$, namely:

$$Q'_i = \{[x, y] \in Q_i \mid \mathcal{I}(x, y) \cap \mathcal{I}_i = \emptyset\}.$$

An argument similar to the earlier one on Q_1 shows that, every query in $Q_i \setminus Q'_i$ consumes an interval in $\mathcal{I}_i$, and every interval in $\mathcal{I}_i$ can be consumed by at most $2l$ queries in $Q_i \setminus Q'_i$. Hence,

$$m_i \geq \frac{|Q_i| - |Q'_i|}{2l} \geq \frac{n/2 - |Q'_i|}{2l}. \tag{5}$$

Next, we give an upper bound on $|Q'_i|$, which will lead to a lower bound for m_i. For each integer $p \in [1, n]$, define p as a *marked position* if an interval in $\mathcal{I}_1 \cup ... \cup \mathcal{I}_{i-1}$ starts or ends at p; otherwise, p is an *unmarked position*.

We claim that, for each query $[x, y] \in Q'_i$, either x or y must be marked. Suppose that this is not true. Consider the (at most 3) intervals in $\mathcal{I}(x, y)$. One of them must start at x, and must be from $\mathcal{I}_0$ (by the fact that x is unmarked). Similarly, $\mathcal{I}(x, y)$ also has an interval in $\mathcal{I}_0$ that ends at y. However, the total length of these two intervals is at most $2l - 2$. To cover $[x, y]$, which has length $(2i + 1)l - 1$, the length of the remaining interval in $\mathcal{I}(x, y)$ must be at least $(2i - 1)l + 1$. So $\mathcal{I}(x, y)$ must contain an interval in $\mathcal{I}_i$, which is a contradiction.

The number of marked positions is at most $2|\mathcal{I}_1 \cup ... \cup \mathcal{I}_{i-1}| = 2\sum_{j=1}^{i-1} m_j$. As each marked position can be an endpoint of at most two queries in Q'_i, it follows that

$$|Q'_i| \leq 4 \sum_{j=1}^{i-1} m_j. \tag{6}$$

By combining (4), (5) and (6), we have:

$$\begin{aligned}
& \sum_{i=1}^{r} m_i \;\geq\; \sum_{i=1}^{r} \frac{n/2 - 4(\sum_{j=1}^{i-1} m_j)}{2l} \;\geq\; \frac{nr}{4l} - \frac{2r}{l} \sum_{i=1}^{r} m_i \\
\Rightarrow \quad & (1 + 2r/l) \sum_{i=1}^{r} m_i \;\geq\; \frac{nr}{4l} \\
\Rightarrow \quad & \sum_{i=1}^{r} m_i \;\geq\; \frac{nr}{4l(1 + 2r/l)}.
\end{aligned}$$

By plugging in $r = \frac{1}{2}(l - 1) \in [l/4, l/2]$, we have that $\sum_{i=1}^{r} m_i \geq \frac{nl/4}{4l(1+2(l/2)/l)} \geq n/32$. □

To complete the proof of Lemma 2 for $t = 3$. When $\alpha_3(n) > 256$, $\alpha_3(n) = \lceil \log_4 \log_4 n \rceil$. Define integer $h = \lfloor \log_2 \log_3 n \rfloor - 1$, which is at least $\alpha_3(n)/8$. For each $i \in [0, h - 1]$, let $l_i = 3^{2^i}$, which is an odd integer in the range $[3, n^{1/4}]$. By Lemma 4, any $(n, 3)$-cover contains at least $n/32$ intervals whose lengths are in $[l_i, l_i^2 - 1]$. Since these ranges are disjoint, it follows that any $(n, 3)$-cover contains at least $nh/32 \geq n \cdot \alpha_3(n)/256$ intervals.

3.3 General Case $t \geq 4$

We will now prove the correctness of Lemma 2 for $t \geq 4$ by induction. Suppose that the lemma holds for some $t \geq 2$, namely, for every $n > 0$, $S_t(n) \geq \frac{1}{256} n \cdot \alpha_t(n)$. Our objective is to prove:

$$\forall n > 0,\; S_{t+2}(n) \geq \frac{1}{256} n \cdot \alpha_{t+2}(n). \tag{7}$$

Our proof uses the same strategy as deployed in the proofs of Lemmas 3 and 4. We will break $[1, n]$ into $h = \Omega(\alpha_{t+2}(n))$ disjoint intervals, and argue that, for each interval I, any $(n, t+2)$-cover must contain $\Omega(n)$ intervals whose lengths are in I. The following technical lemma is crucial in our construction:

Lemma 5. *For any integers $t \geq 2$ and $l \geq 256$, $6l \cdot A_t(64l) < A_t(A_t(l))$.*

Proof. See appendix. □

We will focus on $\alpha_{t+2}(n) > 256$ because otherwise (7) is trivially true. Set $h = \lfloor (\alpha_t^*(n) - 16)/2 \rfloor$, which is at least 100 by Lemma 1. For each $i \in [0, h]$, define integer $l_i = A_t^{(2i)}(256)$. In our setup, it holds that $l_0 = 256$, and $l_h \leq n/2$. Thus, by Lemma 5, we have for each $i \in [0, h-1]$:

$$6l_i \cdot A_t(64l_i) < A_t(A_t(l_i)) = l_{i+1}.$$

Hence, we can produce h disjoint ranges: $[l_i, 6l_i \cdot A_t(64l_i)]$ for each $i \in [0, h-1]$. We will prove the following lemma in the next subsection:

Lemma 6. *Let t, n, l, λ be integers such that $t \geq 2$, $\alpha_{t+2}(n) > 256$, $l \geq 256$ and $\lambda = 6l \cdot A_t(64l) < n/2$. Then, any $(n, t+2)$-cover must contain at least $n/64$ intervals whose lengths are in range $[l, \lambda]$.*

By the above lemma, any $(n, t+2)$ cover contains at least $n/64$ intervals in each of the h ranges produced. Hence, $S_{t+2}(n) \geq \frac{1}{64} nh$. Then (7) follows from the fact that when $\alpha_{t+2}(n) > 256$, $\alpha_{t+2}(n) \leq \alpha_t^*(n) + 2 \leq 4h$.

3.4 Proof of Lemma 6

We will first prove the following fact:

Lemma 7. *Let t, l and λ be integers such that $t \geq 2$, $l \geq 256$ and $\lambda = 6l \cdot A_t(64l)$. Then, any $(\lambda, t+2)$-cover must contain at least $\lambda/32$ intervals of lengths at least l.*

Proof. Let $\mathcal{I}$ be a $(\lambda, t+2)$-cover. We call an interval in $\mathcal{I}$ a *target interval* if its length is at least l, or a *dwarf* otherwise. Cut the range $[1, \lambda]$ into $r = \lambda/3l$ disjoint intervals—called *segments*—of length $3l$, namely, the first segment is $[1, 3l]$, the next one is $[3l+1, 6l]$, and so on (note that λ is a multiple of $3l$). Each segment is further divided into 3 parts:

- *Left buffer*, which consists of the first l positions in the segment;
- *Right buffer*, which consists of the last l positions in the segment;
- *Pivot region*, which consists of the remaining l positions in the middle.

A position $p \in [1, \lambda]$ is *marked* if there exists a target interval in $\mathcal{I}$ that starts or ends at p; otherwise, p is *unmarked*. We say that a segment is *saturated* if all the l positions in its pivot region are marked; otherwise, the segment is *unsaturated*. We separate two cases depending on the number of saturated segments.

In the easy case where there are at least $3r/16$ saturated segments, since each saturated segment contains at least l marked positions, there are at least $3rl/16$ marked positions in total. Each target interval can mark only 2 positions, and hence, there are at least $\frac{3}{32} rl = \frac{1}{32}\lambda$ target intervals.

Now consider the hard case where less than $3r/16$ segments are saturated. Let n' be the number of unsaturated segments, which is at least $13r/16 > \lambda/4l$. We will handle this case by inducing from $\mathcal{I}$ an (n', t)-cover $\mathcal{I}'$, on which our induction hypothesis can be applied. Formally, name all the unsaturated segments from left to right as $\sigma_1, ..., \sigma_{n'}$. Starting with an empty $\mathcal{I}'$, we inspect each interval $[i, j] \in \mathcal{I}$ as follows:

- if $[i, j]$ intersects at least one unsaturated interval, add $[i', j']$ to $\mathcal{I}'$ (if it is not already there), where i' (or j') is such that $\sigma_{i'}$ (or $\sigma_{j'}$, resp.) is the leftmost (or rightmost, resp.) unsaturated segment intersecting $[i, j]$. In this case, we say that $[i', j']$ is *spawned* by $[i, j]$.
- otherwise, do nothing for $[i, j]$.

We will prove in Lemma 8 later that $\mathcal{I}'$ must be an (n', t)-cover.

By our inductive hypothesis, $|\mathcal{I}'| \geq S_t(n') \geq \frac{1}{256} n' \cdot \alpha_t(n')$. As a dwarf interval in $\mathcal{I}$ can intersect at most two segments, the interval in $\mathcal{I}'$ spawned by a dwarf has length either 1 or 2. Therefore, at most $2n'$ intervals in $\mathcal{I}'$ are spawned by dwarf intervals; all the remaining ones must be spawned by target intervals. Hence, the number of target intervals in $\mathcal{I}$ is at least

$$\begin{aligned}
|\mathcal{I}'| - 2n' &\geq \frac{1}{256} n' \cdot (\alpha_t(n') - 512) \\
&\geq \frac{1}{256} \cdot \frac{\lambda}{4l} \cdot \left(\alpha_t \left(\left\lfloor \frac{\lambda}{4l} \right\rfloor \right) - 512 \right) \\
&\geq \frac{\lambda}{1024l} \cdot (\alpha_t(A_t(64l)) - 512) \\
&= \frac{\lambda}{1024l} \cdot (64l - 512) \\
&\geq \frac{1}{32} \lambda.
\end{aligned}$$

□

Lemma 8. *The $\mathcal{I}'$ constructed in the proof of Lemma 7 is an (n', t)-cover.*

Proof. Apparently, $\mathcal{I}'$ contains all the singleton intervals $[1, 1], ..., [n', n']$ (every singleton interval in $\mathcal{I}$ spawns a singleton interval in $\mathcal{I}'$). It remains to show that every query $[x', y']$ $(1 \leq x' < y' \leq n')$ is covered by at most t invertals in $\mathcal{I}'$. Since $\sigma_{x'}$ is an unsaturated segment, there exists an unmarked position x in the pivot region of $\sigma_{x'}$. Similarly, there exists an unmarked position y in the pivot region of $\sigma_{y'}$. Since $\mathcal{I}$ is an $(n, t+2)$-cover, $[x, y]$ can be covered by a set $\mathcal{I}(x, y)$ of no more than $t + 2$ intervals in $\mathcal{I}$. Note that every interval in $\mathcal{I}(x, y)$ has spawned at most one interval in $\mathcal{I}'$. Denote by $\mathcal{I}'(x', y')$ the set of those (at most $t + 2$) spawned intervals. It is clear that $\mathcal{I}'(x', y')$ covers $[x', y']$.

It suffices to show that, there are two redundant intervals in $\mathcal{I}'(x', y')$, such that even if they are removed, the remaining (at most t) intervals still cover $[x', y']$. There must be an interval $[x, z]$ (for some z) in $\mathcal{I}(x, y)$ that starts at x. Since x is an unmarked position, $[x, z]$ must be a dwarf (whose length is less than l). As x is in the pivot region of $\sigma_{x'}$, position $z + 1$ must be also in $\sigma_{x'}$; otherwise, $[x, z]$ should span the right buffer of $\sigma_{x'}$ and have length at least l, which is impossible for a dwarf. Thus, there must be another interval in $\mathcal{I}(x, y)$ that starts inside $\sigma_{x'}$, and contains position $z + 1$. Now consider their spawned intervals in $\mathcal{I}'(x', y')$. Both spawned intervals must start at x'; and furthermore, the one created from $[x, z]$ must be $[x', x']$. So $[x', x']$ is redundant for covering $[x', y']$. With a similar argument, we can find another redundant interval $[y', y']$ in $\mathcal{I}'(x', y')$ (recall that $x' < y'$), which completes the proof. □

Completing the Proof of Lemma 6. Let $\mathcal{I}$ be an $(n, t+2)$-cover. Cut the range $[1, n]$ into $g = \lfloor \frac{n}{\lambda} \rfloor$ disjoint *chunks* of length λ, namely, the first chunk is $[1, \lambda]$, the next chunk is $[\lambda+1, 2\lambda]$, and so on[2]. For each chunk, let $\mathcal{I}'$ be the set of intervals in $\mathcal{I}$ that are completely inside the chunk. Clearly, $\mathcal{I}'$ has to be a $(\lambda, t+2)$-cover. By Lemma 7, $\mathcal{I}'$ contains at least $\frac{1}{32}\lambda$ intervals whose lengths are in $[l, \lambda]$. Therefore, the number of intervals in $\mathcal{I}$ whose lengths are in $[l, \lambda]$ is at least

$$g \cdot \frac{1}{32}\lambda \geq \frac{n}{2\lambda} \cdot \frac{1}{32}\lambda = \frac{1}{64}n.$$

4 Implying Yao's Query Lower Bound from Theorem 1

As explained in Section 1, a space lower bound implies a query lower bound, and vice versa. In [11], Yao presented a tight query lower bound which, however, implies a loose space lower bound. In this section, we will show that our space lower bound (Theorem 1) actually implies Yao's query lower bound. This means that our space-query tradeoff is tight bidirectionally, as claimed in Section 1.2.

Let us first introduce Yao's query lower bound in [11]. His variant of Ackermann function is $\mathbf{A} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ where:

$$\mathbf{A}(x, y) = \begin{cases} 2y & \text{if } x = 0 \\ 0 & \text{if } y = 0 \\ 2 & \text{if } y = 1 \\ \mathbf{A}(x-1, \mathbf{A}(x, y-1)) & \text{otherwise} \end{cases}.$$

There is a relationship between his variant and ours as defined in (3):

Lemma 9. *For any integers $t \geq 2$ and $x \geq 8$, $\mathbf{A}(4t, x) \geq A_t(64x)$.*

Proof. See appendix. □

Let $\mathbb{Z}^+$ be the set of positive integers. The inverse of $\mathbf{A}$ is $\mathbf{a} : \mathbb{Z}^+ \times \mathbb{Z}^+ \to \mathbb{Z}^+$ where

$$\mathbf{a}(m, n) = \min\{h \in \mathbb{Z}^+ \mid \mathbf{A}(h, 4\lceil m/n \rceil) > \log_2 n\}.$$

Yao proved the following query lower bound:

Theorem 2 ([11])**.** *For any range sum structure on n elements with space m, where $n > 0$ and $m \geq n$, it holds that $Query(m) = \Omega(\mathbf{a}(m, n) + n/(m-n+1))$.*

Note that the term $n/(m-n+1)$ is effective only if m is very close to n. For $m \geq 2n$, the lower bound becomes $Query(m) = \Omega(\mathbf{a}(m, n))$.

Theorem 3. *Theorem 1 implies Theorem 2 when $m \geq 2n$.*

Proof. By Theorem 1, the query time t satisfies that

$$\alpha_t(n) \leq 256m/n \leq 256\lceil m/n \rceil.$$

By definition, $A_t(\alpha_t(n)) \geq n$. Hence,

$$A_t(256\lceil m/n \rceil) \geq n.$$

Since $m \geq 2n$, $4\lceil m/n \rceil \geq 8$. By Lemma 9, $\mathbf{A}(4t, 4\lceil m/n \rceil) \geq n > \log_2 n$. By the definition of $\mathbf{a}(m, n)$, we have that $\mathbf{a}(m, n) \leq 4t$, which completes the proof. □

[2]The last $n - g\lambda$ positions are not in any chunk if n does not divide λ.

5 Conclusions

Range sum queries are one of the most fundamental and important type of queries in computer science. There has been considerable research aiming to understand the tradeoff between the space consumption and the query cost of a data structure. Previously, there was a gap between the upper and lower bounds in the space consumption when the objective query cost is non-constant (which is fairly realistic in practice). In this paper, we close the gap by deriving the complete tradeoff. In particular, our lower bounds are asymptotically tight bidirectionally, namely, it gives both (i) the lowest space cost against any target query time, and (ii) the lowest query cost against any target space budget.

References

[1] Noga Alon and Baruch Schieber. Optimal preprocessing for answering on-line product queries. Technical Report TR 71/87, Tel-Aviv University, 1987.

[2] Sunil Arya, David M. Mount, and Jian Xia. Tight lower bounds for halfspace range searching. *Discrete & Computational Geometry*, 47(4):711–730, 2012.

[3] Timothy M. Chan. Optimal partition trees. In *SoCG*, pages 1–10, 2010.

[4] Bernard Chazelle. Lower bounds for orthogonal range searching II. the arithmetic model. *JACM*, 37(3):439–463, 1990.

[5] Michael L. Fredman. A lower bound on the complexity of orthogonal range queries. *JACM*, 28(4):696–705, 1981.

[6] Haripriyan Hampapuram and Michael L. Fredman. Optimal biweighted binary trees and the complexity of maintaining partial sums. *SIAM J. of Comp.*, 28(1):1–9, 1998.

[7] Kasper Green Larsen. On range searching in the group model and combinatorial discrepancy. In *FOCS*, pages 542–549, 2011.

[8] Kasper Green Larsen. The cell probe complexity of dynamic range counting. In *STOC*, pages 85–94, 2012.

[9] Jiri Matousek. Range searching with efficient hiearchical cutting. *Discrete & Computational Geometry*, 10:157–182, 1993.

[10] Mihai Patrascu. Lower bounds for 2-dimensional range counting. In *STOC*, pages 40–46, 2007.

[11] Andrew Chi-Chih Yao. Space-time tradeoff for answering range queries (extended abstract). In *STOC*, pages 128–136, 1982.

Appendix

A Proof of Lemma 1

Let $h = \alpha_{t+2}(n)$, which is at least 1. Our objective is to show $\alpha_t^*(n) \in [h-2, h-1]$. It holds by definition that

$$A_{t+2}(h-1) < n \leq A_{t+2}(h).$$

We first show that $\alpha_t^*(n) \leq h-1$. By expanding the recursive definition of $A_{t+2}(h)$, we get

$$A_{t+2}(h) = A_t(A_{t+2}(h-1)) = ... = A_t^{(h-1)}(A_{t+2}(1)) = A_t^{(h-1)}(256) \geq n.$$

Therefore,

$$\alpha_t^{(h-1)}(n) \leq \alpha_t^{(h-1)}(A_t^{(h-1)}(256)) = 256,$$

which implies that $\alpha_t^*(n) \leq h-1$.

Now we show that $\alpha_t^*(n) > h-3$, which is trivial if $h \leq 2$. If $h \geq 3$, by definition we have that

$$A_t^{(h-3)}(\alpha_t^{(h-3)}(n)) \geq n > A_{t+2}(h-1) = A_t^{(h-2)}(256) > A_t^{(h-3)}(256).$$

Therefore, $\alpha_t^{(h-3)}(n) > 256$, which implies that $\alpha_t^*(n) > h-3$.

B Proof of Lemma 5

When $t = 2$ and $t = 3$, the correctness of the lemma is straightforward. Assume $t \geq 4$. It is easy to show that $6l \leq A_t(64l)$ for any $l \geq 256$. Now consider the inequality in the lemma. For the left hand side, we have that

$$6l \cdot A_t(64l) \leq (A_t(64l))^2 = 16^{\log_4 A_t(64l)} \leq 16^{A_t(64l)}.$$

For the right hand side,

$$\begin{aligned} A_t(A_t(l)) &= A_{t-2}(A_t(A_t(l)-1)) \geq A_2(A_t(A_t(l)-1)) \\ &= 16^{A_t(A_t(l)-1)+1} > 16^{A_t(A_t(l)-1)}. \end{aligned}$$

Therefore, it suffices to show that

$$\begin{aligned} 16^{A_t(64l)} \leq 16^{A_t(A_t(l)-1)} &\Leftarrow A_t(64l) \leq A_t(A_t(l)-1) \\ &\Leftarrow 64l \leq A_t(l)-1, \end{aligned}$$

which is true as long as $t \geq 4$ and $l \geq 256$.

C Proof of Lemma 9

For each integer $t \geq 2$, define propositions $p(t)$ and $q(t)$ as follows:

$p(t)$: for each integer $x \geq 8$, $A_t(64x) \leq \mathbf{A}(4t, x)$
$q(t)$: for each integer $x \geq 256$, $A_t(x) \leq \mathbf{A}(4t, x)$.

It suffices to show that $p(t)$ and $q(t)$ are true for all $t \geq 2$.

The following statements directly follow from the definitions of the two variants of the Ackermann functions:

- $\mathbf{A}(x, y)$ is non-descending with x and strictly ascending with y;
- $128x \leq \mathbf{A}(8, x-1) \leq \mathbf{A}(4t+8, x-1)$, for all $t \geq 2$;
- $p(2)$, $q(2)$, $p(3)$ and $q(3)$ are true.

Assume inductively that $p(t)$ and $q(t)$ are true for some $t \geq 2$. It suffices to prove $p(t+2)$ and $q(t+2)$. For $p(t+2)$, we have that

$$\begin{aligned}
A_{t+2}(64x) &= A_t^{(64x-1)}(256) \\
&\leq \underbrace{\mathbf{A}(4t, \mathbf{A}(4t, ..., \mathbf{A}(4t, 256)))}_{64x-1 \text{ instances of } \mathbf{A}} \\
&= \mathbf{A}(4t+1, 64x+254) \\
&\leq \mathbf{A}(4t+7, 128x) \\
&\leq \mathbf{A}(4t+7, \mathbf{A}(4t+8, x-1)) \\
&= \mathbf{A}(4t+8, x).
\end{aligned}$$

$q(t+2)$ can be proved analogously.