Subsets and Supermajorities: Optimal Hashing-based Set Similarity Search [*]

Thomas D. Ahle
BARC, IT University of Copenhagen
thdy@itu.dk

Jakob B. T. Knudsen
BARC, University of Copenhagen
jakn@di.ku.dk

April 15, 2020

Abstract

We formulate and optimally solve a new generalized Set Similarity Search problem, which assumes the size of the database and query sets are known in advance. By creating polylog copies of our data-structure, we optimally solve any symmetric Approximate Set Similarity Search problem, including approximate versions of Subset Search, Maximum Inner Product Search (MIPS), Jaccard Similarity Search and Partial Match.

Our algorithm can be seen as a natural generalization of previous work on Set as well as Euclidean Similarity Search, but conceptually it differs by optimally exploiting the information present in the sets as well as their complements, and doing so asymmetrically between queries and stored sets. Doing so we improve upon the best previous work: MinHash [J. Discrete Algorithms 1998], SimHash [STOC 2002], Spherical LSF [SODA 2016, 2017] and Chosen Path [STOC 2017] by as much as a factor $n^{0.14}$ in both time and space; or in the near-constant time regime, in space, by an arbitrarily large polynomial factor.

Turning the geometric concept, based on Boolean supermajority functions, into a practical algorithm requires ideas from branching random walks on $\mathbb{Z}^2$, for which we give the first non-asymptotic near tight analysis.

Our lower bounds follow from new hypercontractive arguments, which can be seen as characterizing the exact family of similarity search problems for which supermajorities are optimal. The optimality holds for among all hashing based data structures in the random setting, and by reductions, for 1 cell and 2 cell probe data structures. As a side effect, we obtain new hypercontractive bounds on the directed noise operator $T_\rho^{p_1 \to p_2}$.

[*] A previous version of this manuscript has appeared on arXiv.org under the title "Subsets and Supermajorities: Unifying Hashing-based Set Similarity Search".

Contents

1 Introduction

Set Similarity Search (SSS) is the problem of indexing sets (or sparse boolean data) to allow fast retrieval of sets, similar under a given similarity measure. The sets may represent one-hot encodings of categorical data, “bag of words” representations of documents, or “visual/neural bag of words” models, such as the Scale-invariant feature transform (SIFT), that have been discretized. The applications are ubiquitous across Computer Science, touching everything from recommendation systems to gene sequences comparison. See [26, 40] for recent surveys of methods and applications.

Set similarity measures are any function, s that takes two sets and return s value in $[0,1]$. Unfortunately, most variants of Set Similarity Search, such as Partial Match, are hard to solve assuming popular conjectures around the Orthogonal Vectors Problem [67, 5, 1, 25], which roughly implies that the best possible algorithm is to not build an index, and “just brute force” scan through all the data, on every query. A way to get around this is to study Approximate SSS: Given a query, q, for which the most similar set y has similarity$(q,y) \geq s_1$, we are allowed to return any set y' with similarity$(q,y') > s_1$, where $s_2 < s_1$. In practice, even the best *exact* algorithms for similarity search use such an (s_1,s_2)-approximate[1] solution as a subroutine [29].

[1] By classical reductions [38] we can assume s_1 is known in advance.

Euclidean Similarity Search, where the data is vectors $x \in \mathbb{R}^d$ and the measure of similarity is "Cosine", has recently been solved optimally — at least in the model of hashing based data structures [11, 9]. Meanwhile, the problem on sets has proven much less tractable. This is despite that the first solutions date back to the seminal MinHash algorithm (a.k.a. min-wise hashing), introduced by Broder et al. [21, 20] in 1997 and by now boasting thousands of citations. In 2014 MinHash was shown to be near-optimal for set intersection *estimation* [55], but in a surprising recent development, it was shown not to be optimal for similarity *search* [28]. The question thus remained: What *is* the optimal algorithm for Set Similarity Search?

The question is made harder by the fact that previous algorithms study the problem under different similarity measures, such as Jaccard, Cosine or Braun-Blanquet similarity. The only thing those measures have in common is that they can be defined as a function f of the sets sizes, the universe size and the intersection size. In other words, similarity$(q, y) = f(|q|, |y|, |q \cap y|, |U|)$ where $|U|$ is the size of the universe from which the sets are taken. In fact, any symmetric measure of similarity for sets must be defined by those four quantities.

Hence, to fully solve Set Similarity Search, we avoid specifying a particular similarity measure, and instead define the problem solely from those four parameters. This generalized problem is what we solve optimally in this paper, for all values of the four parameters:

Definition 1 (The (w_q, w_u, w_1, w_2)-GapSS problem)**.** *Given some universe U and a collection $Y \subseteq \binom{U}{w_u|U|}$ of $|Y| = n$ sets of size $w_u|U|$, build a data structure that for any query set $q \in \binom{U}{w_q|U|}$: either returns $y' \in Y$ with $|y' \cap q| > w_2|U|$; or determines that there is no $y \in Y$ with $|y \cap q| \geq w_1|U|$.*

For the problem to make sense, we assume that $w_q|U|$ and $w_u|U|$ are integers, that $w_q, w_u \in [0, 1]$, and that $0 < w_2 < w_1 \leq \min\{w_q, w_u\}$. Note that $|U|$ may be very large, and as a consequence the values w_q, w_u, w_1, w_2 may all be very small.

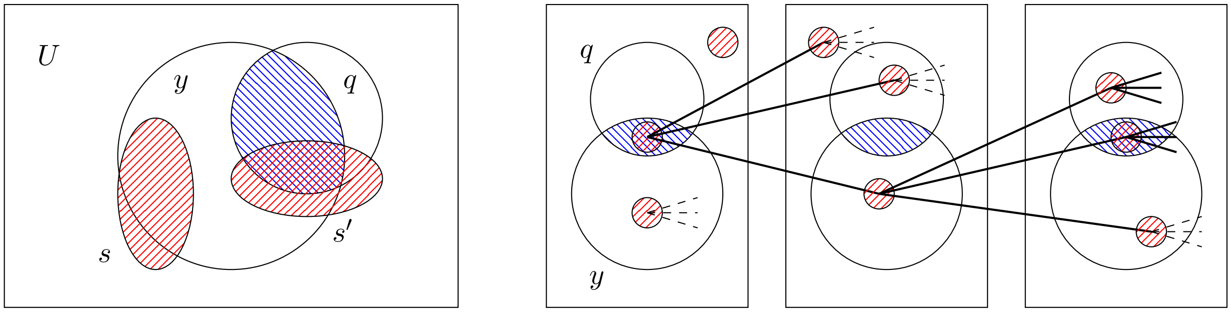
At first sight, the problem may seem easier than the version where the sizes of sets may vary. However, the point is that making polylog(n) data-structures for sets and queries of progressively bigger sizes,[2] immediately yields data structures for the original problem. Similarly, any algorithm assuming a specific set similarity measure also yields an algorithm for (w_q, w_u, w_1, w_2)-GapSS, so our lower bounds too hold for all previously studied SSS problems.

Example 1 As an example, assume we want to solve the Subset Search Problem, in which we, given a query q, want to find a set y in the database, such that $y \subseteq q$. If we allow a two-approximate solution, GapSS includes this problem by setting $w_1 = w_u$ and $w_2 = w_1/2$: The overlap between the sets must equal the size of the stored sets; and we are guaranteed to return a y' such that at least $|q \cap y'| \geq |y|/2$.

Example 2 In the (j_1, j_2)-Jaccard Similarity Search Problem, given a query, q, we must find y such that the Jaccard Similarity $|q \cap y|/|q \cup y| > j_2$ given that a y' exists with similarity at least j_1. After partitioning the sets by size, we can solve the problem using GapSS by setting $w_1 = \frac{j_1(w_q+w_u)}{1+j_1}$ and $w_2 = \frac{j_2(w_q+w_u)}{1+j_2}$. The same reduction works for any other similarity measure with polylog(n) overhead.

The version of this problem where $w_2 = w_q w_u$ is similar to what is in the literature called *"the random instance"* [56, 44, 10]. To see why, consider generating $n - 1$ sets independently at random with size $w_u|U|$, and a "planted" pair, (q, y), with size respectively $w_q|U|$ and $w_u|U|$ and with

[2] For details, see [28] Section 5. A similar reduction, called "norm ranging", was recently shown at NeurIPS to give state of the art results for Maximum Inner Product Search in $\mathbb{R}^d$ [69], suggesting it is very practical.



(a) Two cohorts, y and q with a large intersection (blue). The first representative set, s, favours y, while the second, s', favours both y and q.

(b) Branching random walk run on two cohorts q and y. The bold lines illustrate paths considered by sets, while the dashed lines adorn paths only considered by only one of x or y. Here q has a higher threshold ($t_q = 2/3$) than y ($t_u = 1/2$), so q only considers paths starting with two favourable representatives.

Figure 1: The representative sets, coloured in red, are scattered in the universe to provide an efficient space partition for the data.

intersection $|q \cap y| = w_1|U|$. Insert the size $w_u|U|$ sets into the database and query with q. Since q is independent from the $n-1$ original sets, its intersection with those is strongly concentrated around the expectation $w_q w_u |U|$. Thus, if we parametrize GapSS with $w_2 = w_q w_u + o(1)$, the query for q is guaranteed to return the planted set y.

There is a tradition in the Similarity Search literature for studying such this independent case, in part because *it is expected that one can always reduce to the random instance*, for example using the techniques of "data-dependent hashing" [8, 11]. However, for such a reduction to make sense, we would first need an optimal "data-independent" algorithm for the $w_2 = w_q w_u$ case, which is what we provide in this paper. We discuss this further in the Related Work section.

For generality we still define the problem for all $w_2 \in (0, w_1)$, our upper bound holds in this general setting and so does the lower bound Theorem 2.

We give our new results in Section 1.2 and our new lower bounds in Section 1.3, but first we would like to sketch the algorithm and some probabilistic tools used in the theorem statement.

1.1 Supermajorities

In Social Choice Theory a supermajority is when a fraction strictly greater than 1/2 of people agree about something.[3] In the analysis of Boolean functions a t-supermajority function $f : \{0,1\}^n \to \{0,1\}$ can be defined as 1, if a fraction $\geq t$ of its arguments are 1, and 0 otherwise. We will sometimes use the same word for the requirement that a fraction $\leq t$ of the arguments are 1.[4]

The main conceptual point of our algorithm is the realization that an optimal algorithm for Set Similarity Search must take advantage of the information present in the given sets, as well as that present in their complement. A similar idea was leveraged by Cohen et al. [30] for Set Similarity *Estimation*, and we show in Section 4.2 that the classical MinHash algorithm can be seen as an average of functions that pull varying amounts of information from the sets and their complements.

[3] "America was founded on majority rule, not supermajority rule. Somehow, over the years, this has morphed into supermajority rule, and that changes things." – Kent Conrad.

[4] It turns out that defining everything in terms of having a fraction $t \pm o(1)$ of 1's is also sufficient. This is similar to Dubiner [33].

In this paper, we show that there is a better way of combining this information, and that doing so results in an optimal hashing based data structure for the entire parameter space of random instance GapSS.

This way of combining this information is by supermajority functions. While on the surface they will seem similar to the threshold methods applied for time/space trade-offs in Spherical LSF [9], our use of them is very different. Where [9] corresponds to using small $t = 1/2 + o(1)$ thresholds (essentially simple majorities) our t may be as large as 1 (corresponding to the AND function) or as small as 0 (the NOT AND function). This way they are a sense as much a requirement on the complement as it is on the sets themselves.

The algorithm (idealized): While our data structure is technically a tree with a carefully designed pruning rule, the basic concept is very simple.

We start by sampling a large number of "representative sets" $R \subseteq \binom{U}{k}$. Here roughly $|R| \approx n^{\log n}$ and $k \approx \log n$. Given family $Y \subseteq \binom{U}{w_u|U|}$ of sets to store, which we call "cohorts", we say that $r \in R$ "t-favours" the cohort y if $|y \cap r|/|r| \geq t$. Representing sets as vectors in $\{0,1\}^d$, this is equivalent to saying $f_t(r \cap y) = 1$, where f_t is the t-supermajority function. (If t is less than w_u, the expected size of the overlap, we instead require $|y \cap r|/|r| \leq t$.)

Given the parameters $t_q, t_u \in [0,1]$, the data-structure is a map from elements of R to the cohorts they t_u-favour. When given a query $q \in \binom{U}{w_q|U|}$, (a $w_q|U|$ sized cohort), we compare it against all cohorts y favoured by representatives $r \in R$ which t_q-favour q (that is $|q \cap r|/|r| \geq t_q$). This set $R_{t_q}(q)$ is much smaller than $|R|$ (we will have $|R_{t_q}(q)| \approx n^{\varepsilon}$ and $E[|R_{t_u}(y) \cap R_{t_q}(q)|] \approx n^{\varepsilon-1}$), so the filtering procedure greatly reduces the number of cohorts we need to compare to the query from n to n^{ε} (where $\varepsilon = \rho_q < 1$ is defined later.)

The intuition is that while it is quite unlikely for a representative to favour a given cohort, and it is *very* unlikely for it to favour two given cohorts (q and y). So if it does, the two cohorts probably have a substantial overlap. Figure 1a has a simple illustration of this principle.

In order to fully understand supermajorities, we want to understand the probability that a representative set is simultaneously in favour of two distinct cohorts given their overlap and representative sizes. This paragraph is a bit technical, and may be skipped at first read. Chernoff bounds in $\mathbb{R}$ are a common tool in the community, and for iid. $X_i \sim \text{Bernoulli}(p) \in \{0,1\}$ the sharpest form (with a matching lower bound) is $\Pr[\sum X_i \geq tn] \leq \exp(-n\,\mathrm{d}(t \,\|\, p))$,[5] which uses the binary KL-Divergence $\mathrm{d}(t \,\|\, p) = t \log \frac{t}{p} + (1-t) \log \frac{1-t}{1-p}$. The Chernoff bound for $\mathbb{R}^2$ is less common, but likewise has a tight description in terms of the KL-Divergence between two discrete distributions: $\mathrm{D}(P \,\|\, Q) = \sum_{\omega \in \Omega} P(\omega) \log \frac{P(\omega)}{Q(\omega)}$ (summing over the possible events). In our case, we represent the four events that can happen as we sample an element of U as a vector $X_i \in \{0,1\}^2$. Here $X_i = \left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$ means the ith element hit both cohorts, $X_i = \left[\begin{smallmatrix}1\\0\end{smallmatrix}\right]$ means it hit only the first and so on. We represent the distribution of each X_i as a matrix $P = \left[\begin{smallmatrix} w_1 & w_q - w_1 \\ w_u - w_1 & 1 - w_q - w_u + w_1 \end{smallmatrix}\right]$, and say $X_i \sim \text{Bernoulli}(P)$ iid. such that $\Pr[X_i = \left[\begin{smallmatrix}1-j\\1-k\end{smallmatrix}\right]] = P_{j,k}$. Then $\Pr[\sum X_i \geq \left[\begin{smallmatrix}t_q\\t_u\end{smallmatrix}\right] n] \approx \exp(-n\,\mathrm{D}(T \,\|\, P))$ where $T = \left[\begin{smallmatrix} t_1 & t_q - t_1 \\ t_u - t_1 & 1 - t_q - t_u + t_1 \end{smallmatrix}\right]$ and $t_1 \in [0, \min\{t_u, t_q\}]$ minimizes $\mathrm{D}(T \,\|\, P)$. (Here the notation $\left[\begin{smallmatrix}x\\y\end{smallmatrix}\right] \geq \left[\begin{smallmatrix}t_u\\t_q\end{smallmatrix}\right]$ means $x \geq t_u \wedge y \geq t_q$.)

The optimality of Supermajorities for Set Similarity Search is shown using a certain correspondence we show between the Information Theoretical quantities described above, and the hypercontractive inequalities that have been central in all previous lower bounds for similarity search.

These bounds above would immediately allow a cell probe version of our upper bound Theo-

[5] A special case of Hoeffding's inequality is obtained by $\mathrm{d}(p + \varepsilon \,\|\, p) \geq 2\varepsilon^2$, Pinsker's inequality.

rem 1, e.g. a query would require $n^{\frac{\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_q \,\|\, w_q)}{\mathrm{D}(T_2 \,\|\, P_2) - \mathrm{d}(t_q \,\|\, w_q)}}$ probes, where $P_i = \left[\begin{smallmatrix} w_i & w_q - w_1 \\ w_u - w_i & 1 - w_q - w_u + w_i \end{smallmatrix}\right]$ and T_i defined accordingly. The algorithmic challenge is that, for optimal performance, $|R|$ must be in the order of $\Omega(n^{\log n})$, and so checking which representatives favour a given cohort takes super polynomial time!

The classical approach to designing an oracle to efficiently yield all such representatives, , is a product-code or "tensoring trick". The idea, (used by [27, 16]), is to choose a smaller $k' \approx \sqrt{k}$, make k/k' different R_i' sets of size $n^{\sqrt{\log n}}$ and take R as the product $R_1' \times \cdots \times R_{k/k'}'$. As each R' can now be decoded in $n^{o(1)}$ time, so can R. This approach, however, in the case of Supermajorities, has a big drawback: Since $t_q k'$ and $t_u k'$ must be integers, t_q and t_u have to be rounded and thus distorted by a factor $1+1/k'$. Eventually, this ends up costing us a factor $w_1^{-k/k'}$ which can be much larger than n. For this reason, we need a decoding algorithm that allows us to use supermajorities with as large a k as possible!

We instead augment the above representative sampling procedure as follows: Instead of independent sampling sets, we (implicitly) sample a large, random height k tree, with nodes being elements from the universe. The representative sets are taken to be each path from the root to a leaf. Hence, some sets in R share a common prefix, but mostly they are still independent. We then add the extra constraint that *each of the prefixes of a representative has to be in favour of a cohort*, rather than only having this requirement on the final set. This is the key to making the tree useful: Now given a cohort, we walk down tree, pruning any branches that do not consistently favour a supermajority of the cohort. Figure 1b has a simple illustration of this algorithm and Algorithm 1 has a pseudo-code implementation. This pruning procedure can be shown to imply that we only spend time on representative sets that end up being in favour of our cohort, while only weakening the geometric properties of the idealized algorithm negligibly.

While conceptually simple and easy to implement (modulo a few tricks to prevent dependency on the size of the universe, $|U|$), the pruning rule introduces dependencies that are quite tricky to analyze sufficiently tight. The way to handle this will be to consider the tree as a "branching random walk" over $\mathbb{Z}_+^2$ where the value represents the size of the representative's intersection with the query and a given set respectively. The paths in the random walk at step i must be in the quadrant $[t_q i, i] \times [t_u i, i]$ while only increasing with a bias of $\left[\begin{smallmatrix} w_q \\ w_u \end{smallmatrix}\right]$ per step. The branching factor is carefully tuned to just the right number of paths survive to the end.

The "history" aspect of the pruning is a very important property of our algorithm, and is where it conceptually differs from all previous work.

Previous Locality Sensitive Filtering, LSF, algorithms [28, 10] can be seen as trees with pruning, but their pruning is on the individual node level, rather than on the entire path. This makes a big difference in which space partitions can be represented, since pruning on node level ends up representing the intersection of simple partitions, which can never represent Supermajorities in an efficient way. In [16] a similar idea was discussed heuristically for Gaussian filters, but ultimately tensoring was sufficient for their needs, and the idea was never analyzed.

1.2 Upper Bounds

As discussed, the performance of our algorithm is described in terms of KL-divergences. To ease understanding, we give a number of special cases, in which the general bound simplifies. The bounds in this section assume w_q, w_u, w_1, w_2 are constants. See Section 2.1 for a version without this assumption.

Theorem 1 (Simple Upper Bound). *For any choice of constants $w_q, w_u \ge w_1 \ge w_2 \ge 0$ and $1 \ge t_q, t_u \ge 0$ we can solve the (w_q, w_u, w_1, w_2)-GapSS problem over universe U with query time $\tilde{O}(n^{\rho_q} + w_q|U|) + n^{o(1)}$ and auxiliary space usage $\tilde{O}(n^{1+\rho_u})$, where*

$$\rho_q = \frac{\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_q \,\|\, w_q)}{\mathrm{D}(T_2 \,\|\, P_2) - \mathrm{d}(t_q \,\|\, w_q)}, \quad \rho_u = \frac{\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_u \,\|\, w_u)}{\mathrm{D}(T_2 \,\|\, P_2) - \mathrm{d}(t_q \,\|\, w_q)}.$$

and T_1, T_2 are distributions with expectation $\left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]$ minimizing respectively $\mathrm{D}(T_1 \,\|\, P_1)$ and $\mathrm{D}(T_2 \,\|\, P_2)$, as described in Section 1.1.

The two bounds differ only in the $\mathrm{d}(t_q \,\|\, w_q)$ and $\mathrm{d}(t_u \,\|\, w_u)$ terms in the numerator. The thresholds t_q and t_u can be chosen freely in $[0,1]^2$. Varying them compared to each other allows a full space/time trade-off with $\rho_q = 0$ in one end and $\rho_u = 0$ (and $\rho_q < 1$) in the other. Note that for a given GapSS instance, there are many (t_q, t_u) which are not optimal anywhere on the space/time trade-off. Using Lagrange's condition $\nabla \rho_q = \lambda \nabla \rho_u$ one gets a simple equation that all optimal (t_q, t_u) trade-offs must satisfy. As we will discuss later, it seems difficult to prove that a solution to this equation is unique, but in practice it is easy to solve and provides an efficient way to optimize ρ_q given a space budget $n^{1+\rho_u}$. Figure 2 and Figure 3 provides some additional intuition for how the ρ values behave for different settings of GapSS.

Regarding the other terms in the theorem, we note that the $\tilde{O}$ hides only $\log n$ factors, and the additive $n^{o(1)}$ term grows as $e^{O(\sqrt{\log n \log\log n})}$, which is negligible unless $\rho_q = 0$. We also note that there is no dependence on $|U|$, other than the need to store the original dataset and the additive $w_q|U|$, which is just the time it takes to receive the query. The main difference between this theorem and the full version, is that the full theorem does not assume the parameters (w_q, w_u, w_1, w_2) are constants, but consider them potentially very small. In this more realistic scenario it becomes very important to limit the dependency on factors like w_1^{-1}, which is what guides a lot of our algorithmic decisions.

Example 1: Near balanced ρ values. As noted, many pairs (t_q, t_u) are not optimal on the trade-off, in that one can reduce one or both of ρ_q, ρ_u by changing them. The pairs that are optimal are not always simple to express, so it is interesting to study those that are. One such particularly simple choice on the Lagrangian is $t_q = 1 - w_u$ and $t_u = 1 - w_q$.[6] This point is special because the values of t_q and t_u depend only on w_u and w_q, while in general they will also depend on w_1 and w_2. In this setting we have $T_i = \left[\begin{smallmatrix} 1-w_q-w_u+w_i & w_u-w_i \\ w_u-w_i & w_i \end{smallmatrix}\right]$, which can be plugged into Theorem 1.

In the case $w_q = w_u = w$ we get the balanced ρ values $\rho_q = \rho_u = \log(\frac{w_1}{w}\frac{1-w}{1-2w+w_1}) / \log(\frac{w_2}{w}\frac{1-w}{1-2w+w_2})$ in which case it is simple to compare with Chosen Path's ρ value of $\log(\frac{w_1}{w}) / \log(\frac{w_2}{w})$. Chosen Path on balanced sets was shown in [28] to be optimal for w, w_1, w_2 small enough, and we see that Supermajorities do indeed recover this value for that range.

We give a separate lower bound in Section 3.4 showing that this value is in fact optimal when $w_2 = w_q w_u$.

Example 2: Subset/superset queries. If $w_1 = \min\{w_u, w_q\}$ and $w_2 = w_u w_q$ we can take $t_q = \frac{\alpha}{w_q - w_u} + \frac{w_q(1-w_u)}{w_q - w_u}$ and $t_u = \frac{w_u(1-w_u)w_q(1-w_q)}{w_q - w_u}\alpha^{-1} - \frac{w_u(1-w_q)}{w_q - w_u}$ for any $\alpha \in [w_1 - w_q w_u,\ \max\{w_u, w_q\} -$

[6] To make matters complicated, this *is* a simple choice *and* on the Lagrangian, but that doesn't prove another point on the Lagrangian won't reduce both ρ_q and ρ_u and thus be better. That we have a matching lower bound for the algorithm doesn't help, since it only matches the upper bound for (t_q, t_u) minimal in Theorem 1. In the case $w_q = w_u$ we can, however, prove that this t_q, t_u pair is optimal.

$w_q w_u]$. Theorem 1 then gives data structures with

$$\rho_q = \frac{t_q \log \frac{1-t_u}{1-w_u} - t_u \log \frac{1-t_q}{1-w_q}}{\mathrm{d}(t_u \,\|\, w_u)} \qquad \rho_u = \frac{(1-t_u)\log \frac{t_q}{w_q} - (1-t_q)\log \frac{t_u}{w_u}}{\mathrm{d}(t_u \,\|\, w_u)} \qquad \text{if } w_1 = w_u,$$

$$\rho_q = \frac{-(1-t_u)\log \frac{t_q}{w_q} + (1-t_q)\log \frac{t_u}{w_u}}{\mathrm{d}(t_u \,\|\, w_u)} \qquad \rho_u = \frac{-t_q \log \frac{1-t_u}{1-w_u} + t_u \log \frac{1-t_q}{1-w_q}}{\mathrm{d}(t_u \,\|\, w_u)} \qquad \text{if } w_1 = w_q.$$

This represents one of the cases where we can solve the Lagrangian equation to get a complete characterization of the t_q, t_u values that give the optimal trade-offs. Note that when $w_1 = w_u$ or $w_1 = w_q$, the P matrix as used in the theorem has 0's in it. The only way the KL-divergence $\mathrm{D}(T \,\|\, P)$ can then be finite is by having the corresponding elements of T be 0 and use the fact that $0 \log \frac{0}{q}$ is defined to be 0 in this context.

Example 3: Linear space/constant time. Setting t_1 in $T_1 = \begin{bmatrix} t_1 & t_q - t_1 \\ t_u - t_1 & 1 - t_q - t_u + t_1 \end{bmatrix}$ such that either $\frac{t_1}{w_1} = \frac{t_q - t_1}{w_q - w_1}$ or $\frac{t_1}{w_1} = \frac{t_u - t_1}{w_u - w_1}$ we get respectively $\mathrm{D}(T_1 \,\|\, P_1) = \mathrm{d}(t_q \,\|\, w_q)$ or $\mathrm{D}(T_1 \,\|\, P_1) = \mathrm{d}(t_u \,\|\, w_u)$. Theorem 1 then yields algorithms with either $\rho_q = 0$ or $\rho_u = 0$ corresponding to either a data structure with $\approx e^{\tilde{O}(\sqrt{\log n})}$ query time, or with $\tilde{O}(n)$ auxiliary space. Like [9] we have $\rho_q < 1$ for any parameter choice, even when $\rho_u = 0$. For very small w_q and $w_u < \exp(-\sqrt{\log n})$ there are some extra concerns which are discussed after the main theorem.

1.3 Lower Bounds

Results on approximate similarity search are usually phrased in terms of two quantities: (1) The "query exponent" $\rho_q \in [0,1]$ which determines the query time by bounding it by $O(n^{\rho_q})$; (2) The "update exponent" $\rho_u \in [0,1]$ which determines the time required to update the data structure when a point is inserted or deleted in Y and is given by $O(n^{\rho_u})$. The update exponent also bounds the space usage as $O(n^{1+\rho_u})$. Given parameters (w_q, w_u, w_1, w_2), the important question is for which pairs of (ρ_q, ρ_u) there exists data structures. E.g. given a space budget imposed by ρ_u, we ask how small can one make ρ_q?

Since the first lower bounds on Locality Sensitive Hashing [49], lower bounds for approximate near neighbours have split into two kinds: (1) Cell probe lower bounds [57, 58, 9] and (2) Lower bounds in restricted models [52, 13, 9, 28]. The most general such model for data-independent algorithms was formulated by [9] and defines a type of data structure called "list of points":

Definition 2 (List-of-points)**.** *Given some universes, Q, U, a similarity measure $S : Q \times U \to [0,1]$ and two thresholds $1 \geq s_1 > s_2 \geq 0$,*

1. *We fix (possibly random) sets $A_i \subseteq \{-1,1\}^d$, for $1 \leq i \leq m$; and with each possible query point $q \in \{-1,1\}^d$, we associate a (random) set of indices $I(q) \subseteq [m]$;*
2. *For a given dataset P, we maintain m lists of points $L_1, L_2, \ldots, L_m$, where $L_i = P \cap A_i$.*
3. *On query q, we scan through each list L_i for $i \in I(q)$ and check whether there exists some $p \in L_i$ with $S(q,p) \geq s_2$. If it exists, return p.*

The data structure succeeds, for a given $q \in Q, p \in P$ with $S(q,p) \geq s_1$, if there exists $i \in I(q)$ such that $p \in L_i$. The total space is defined by $S = m + \sum_{i \in [m]} |L_i|$ and the query time by $T = |I(q)| + \sum_{i \in I(q)} |L_i|$.

The List-of-points model contains all known Similarity Search data structures, except for the so-called "data-dependent algorithms". It is however conjectured [10] that data-dependency does not help on random instances (recall this corresponds to $w_2 = w_q w_u$), which is the setting of Theorem 3.

We show two main lower bounds: (1) That requires $w_q = w_u$ and $\rho_q = \rho_u$ and (2) That requires $w_2 = w_q w_u$. The second type is tight everywhere, but quite technical. The first type meanwhile is quite simple to state, informally:

Theorem 2. *If $w_q = w_u = w$ and $\rho_u = \rho_q = \rho$, any data-independent LSF data structure must use space $n^{1+\rho}$ and have query time n^{ρ} where $\rho \geq \log(\frac{w_1 - w^2}{w(1-w)}) / \log(\frac{w_2 - w^2}{w(1-w)})$.*

The LSF Model defined in [16, 28] generalizes [49, 54], but is slightly stronger than list-of-points. It is most likely that they are equivalent, so we defer its definition till Definition 4. We will just note that previous bounds of this type [54, 28] were only asymptotic, whereas our lower bound holds over the entire range of $0 < w_2 < w_1 < w < 1$. By comparison with $\rho = \log(\frac{w_1(1-w)}{w(1-2w+w_1)}) / \log(\frac{w_2(1-w)}{w(1-2w+w_2)})$ from Example 1 in the Upper Bounds section, we see that the lower bound is sharp when $w, w_1, w_2 \to 0$[7] and also for $w_1 \to w$, since $w(1 - 2w + w_1) = w(1-w) - w(w - w_1)$. However, for $w_2 = w^2$ (the random instance), Theorem 2 just says $\rho \geq 0$, which means it tells us nothing.

For the random instances, we give an even stronger lower bound, which gets rid of the restrictions $w_q = w_u$ and $\rho_q = \rho_u$. This lower bound is tight for any $0 < w_q w_u < w_1 < \min\{w_q, w_u\}$ in the list-of-points model.

Theorem 3. *Consider any list-of-point data structure for the $(w_q, w_u, w_1, w_q w_u)$-GapSS problem over a universe of size d of n points with $w_q w_u d = \omega(\log n)$, which uses expected space $n^{1+\rho_u}$, has expected query time $n^{\rho_q - o_n(1)}$, and succeeds with probability at least 0.99. Then for every $\alpha \in [0,1]$ we have that*

$$\alpha\rho_q + (1-\alpha)\rho_u \geq \inf_{\substack{t_q, t_u \in [0,1] \\ t_u \neq w_u}} \left(\alpha \frac{\mathrm{D}(T \,\|\, P) - \mathrm{d}(t_q \,\|\, w_q)}{\mathrm{d}(t_u \,\|\, w_u)} + (1-\alpha) \frac{\mathrm{D}(T \,\|\, P) - \mathrm{d}(t_u \,\|\, w_u)}{\mathrm{d}(t_u \,\|\, w_u)} \right) ,$$

where $P = \begin{bmatrix} w_1 & w_q - w_1 \\ w_u - w_1 & 1 - w_q - w_u + w_1 \end{bmatrix}$ *and* $T = \operatorname*{arg\,inf}_{T \ll P,\ \underset{X\sim T}{E}[X] = \left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]} \mathrm{D}(T \,\|\, P).$

Note that for $w_2 = w_q w_u$, the term $\mathrm{D}(T_2 \,\|\, P_2)$, in Theorem 1, splits into $\mathrm{d}(t_q \,\|\, w_q) + \mathrm{d}(t_u \,\|\, w_u)$, and so the upper and lower bounds perfectly match. This shows that for any linear combination of ρ_q and ρ_u our algorithm obtains the minimal value. By continuity of the terms, this equivalently states as saying that no list-of-points algorithm can get a better query time than our Theorem 1, given a space budget imposed by ρ_u. [8]

Example 1: Choices for t_q and t_u. As in the upper bounds, it is not easy to prove that a particular choice of t_q and t_u minimizes the lower bound. One might hope that having corresponding lower and upper bounds would help in this endeavour, but alas both results have a minimization.

[7] As $w, w_1, w_2 \to 0$ we recover the lower bound $\rho \geq \log(\frac{w_1}{w}) / \log\left(\frac{w_2}{w}\right)$ obtained for Chosen Path in [28].

[8] It is easy to see that $\rho_u = 0$ minimizes $\alpha\rho_q + (1-\alpha)\rho_u$ when $\alpha = 0$, and similarly $\rho_u = \rho_{max}$ minimizes $\alpha\rho_q + (1-\alpha)\rho_u$ when $\alpha = 1$, where ρ_{max} is the minimal space usage when $\rho_q = 0$. Furthermore, we note that when we change α from 0 to 1, then ρ_u will continuously and monotonically go from 0 to ρ_{max}. This shows that for every $\rho_u \in [0, \rho_{max}]$ there exists an α such that $\alpha\rho_q + (1-\alpha)\rho_u$ is minimized, where ρ_q is best query time given the space budget imposed by ρ_u.

E.g. setting $t_q = 1 - w_u$ and $t_u = 1 - w_q$ the expression in Theorem 3 we obtain the same value as in Theorem 1, however it could be (though we strongly conjecture not) that another set of values would reduce both the upper and lower bound.

The good news is that the hypercontractive inequality by Oleszkiewicz [53], can be used to prove certain optimal choices on the space/time trade-off.[9] In particular we will show that for $w_q = w_u = w$ the choice $t_q = t_u = 1 - w$ is optimal in the lower bound, and matches exactly the value $\rho = \log\left(\frac{w_1(1-w)}{w(1-2w+w_1)}\right) / \log(\frac{w_2(1-w)}{w(1-2w+w^2)})$ from Example 1 in the Upper Bounds section.

Example 2: Cell probe bounds Panigrahy et al. [57, 58, 42] created a framework for showing cell probe lower bounds for problems like approximate near-neighbour search and partial match based on a notion of "robust metric expansion". Using the hypercontractive inequalities shown in this paper with this framework, as well (as the extension by [9]), we can show, unconditionally, that no data structure, which probes only 1 or 2 memory locations[10], can improve upon the space usage of $n^{1+\rho_u}$ obtained by Theorem 1 as we let $\rho_q = 0$. In particular, this shows that the near-constant query time regime from Example 3 in the Upper Bounds is optimal up to $n^{o(1)}$ factors in time and space.

1.4 Technical Overview

The contributions of the paper are conceptual as well as technical. To a large part, what enables tight upper and lower parts is defining the right problem to study. The second part is realizing which geometry is going to work and proving it in a strong enough model. Lastly, a number of tricky algorithmic problems arise, requiring a novel algorithm and a new analysis of 2-dimensional branching random walks of exponentially tilted variables.

Supermajorities – why do they work? Representing sets $x \subseteq U$ as a vector $x \in \{0,1\}^{|U|}$ and scaling by $1/\sqrt{|x|}$, we get $\|x\|_2 = 1$, and it is natural to assume the optimal Similarity Search data structure for data on the unit-sphere — Spherical LSF — should be a good choice. Unfortunately this throws away two key properties of the data: that the vectors are sparse, and that they are non-negative. Algorithms like MinHash, which are specifically designed for this type of data, take advantage of the sparsity by entirely disregarding the remaining universe, U. This is seen by the fact that adding new elements to U never changes the MinHash of a set. Meanwhile Spherical LSF takes the inner product between x and a Gaussian vector scaled down by $1/\sqrt{|U|}$, so each new element added to U, in a sense, lowers the "sensitivity" to x.

In an alternative situation we might imagine $|x|$ being nearly as big as $|U|$. In this case we would clearly prefer to work with $U \setminus x$, since information about an element that is left out, is much more valuable than information about an element contained in x. What Supermajorities does can be seen as balancing how much information to include from x with how much to include from $U \setminus x$. A very good example of this is in Section 4.2, which shows how to view MinHash as an average of simple algorithms that sample a specific amount from each of x and $U \setminus x$. Supermajorities, however, does this in a more clever way, that turns out to be optimal. A crucial advantage is the knowledge of the size of x, as well as the future queries, which allow us to use different thresholds on the storage and query side, each which is perfectly balanced to the problem instance.

[9] The generalizations by Wolff [68] could in principle expand this range, but they are only tight up to a constant in the exponent.

[10] For 1 probe, the word size can be $n^{o(1)}$, whereas for the 2 probe argument, the word size can only be $o(\log n)$ for the lower bound to hold.

As an interesting side effect, the extra flexibility afforded by our approach allows balancing the time required to perform queries with the size of the database. It is perhaps surprising that this simple balancing act is enough to be optimal across all hashing algorithms as well as 1 cell and 2 cell probe data structures.

The results turn out to be best described in terms of the KL-divergences $\mathrm{D}(T \,\|\, P) - \mathrm{d}(t_q \,\|\, w_q)$ and $\mathrm{D}(T \,\|\, P) - \mathrm{d}(t_u \,\|\, w_u)$, which are equivalent to $\mathrm{D}(T_{XY} \,\|\, P_{Y|X} T_X)$ and $\mathrm{D}(T_{XY} \,\|\, P_{X|Y} T_Y)$. Here P_{XY} is the distribution of a coordinated sample from both a query and a dataset, P_X and P_Y are the marginals, and T_{XY} is roughly the distribution of samples conditioned on having a shared representative set. Intuitively these describe the amount of information gained when observing a sample from T_{XY} given a belief that X (resp. Y) is distributed as T and Y (resp. X) is distributed as P. In this framework, Supermajorities can be seen as a continuation of the Entropy LSH approach by [56].

Branching Random Walks Making Supermajorities a real algorithm (rather than just cell probe), requires, as discussed in the introduction, an efficient decoding algorithm of which representative sets overlap with a given cohort. Previous LSF methods can be seen as trees, with independent pruning in each leaf, going back to the LSH forest in 2005 [15, 12]. Our method is the first to significantly depart from this idea: While still a tree, our pruning is highly dependent across the levels of the tree, carrying a state from the root to the leaf which needs be considered by the pruning as well as the analysis. In "branching random walk", the state is represented in the "random walk", while the tree is what makes it branching. While considered heuristically in [16], such a stateful oracle has not before been analysed, partly because it wasn't necessary. For Supermajorities, meanwhile, it is crucially important. The reason is that failure of the "tensoring trick" employed previously in the literature, when working with thresholds.

The approach from [7, 16, 9] when applied to our scheme would correspond to making our representatives have size just $\sqrt{k}$ (so there are only $|R'| \approx e^{\tilde{O}(\sqrt{\log n})}$ of them,) and then make $R'^{\otimes\sqrt{k}}$ our new R. Since R' can be decoded in $n^{o(1)}$ time, and the second step can be made to take only time proportional to the output, this works well for some cases. This approach has two main issues: (1) There is a certain overhead that comes from not using the optimal filters, but only an approximation. However, this gives only a factor $e^{\tilde{O}(\sqrt{\log n})}$, which is usually tolerated. Worse is (2): Since the thresholds $t_q k$ and $t_u k$ have to be integral, using representative sets of size $\sqrt{k}$ means we have to "repair" them by a multiplicative distortion of approximately $1 \pm 1/\sqrt{k}$, compared to $1 \pm 1/k$ for the "real" filters. This turns out to cost as much as $w_1^{-\sqrt{k}}$ which can easily be much larger than the polynomial cost in n. In a sense, this shows that supermajority functions must be applied to measure the entire representative part of a cohort at once! This makes tensoring not well fit for our purposes.

A pruned branching random walk on the real line can be described in the following way. An initial ancestor is created with value 0 and form the zeroth generation. The people in the ith generation give birth Δ times each and independently of one another to form the $(i{+}1)$th generation. The people in the $(i+1)$th generation inherit the value, v, of their parent plus an independent random variable X. If ever $v + X < 0$, the child doesn't survive. After k generations, we expect by linearity $\Delta^k \Pr[\forall_{i \le k} \sum_{j \in [i]} X_i \ge 0]$ people to be alive, where X_i are iid. random variables as used in the branching. A pruned 2d-branching random walk is simply one using values $\in \mathbb{R}^2$.

Branching random walks have been analysed before in the Brownian motion literature [63]. They are commonly analysed using the second-moment method, however, as noted by Bramson [18]: "an immediate frontal assault using moment estimates, but ignoring the branching structure of the process, will fail." The issue is that the probability that a given pair of paths in the branching process survives is too large for standard estimates to succeed. If the lowest common ancestor of two nodes manages to accumulate much more wealth than expected, its children will have a much too high chance of surviving. For this reason we have to *counterintuitively add extra pruning when proving the lower bound* that a representative set survives. More precisely, we prune all the paths that accumulate much more than the expected value. We show that this does not lower the probability that a representative set is favour by much, while simultaneously decreasing the variance of the branching random walk a lot. Unfortunately, this adds further complications, since ideally, we would like to prune every path that gets below the expectation. Combined with the upper bound this would trap the random walks in a band to narrow to guarantee the survival of a sufficient number of paths. Hence instead, we allow the paths to deviate by roughly a standard deviation below the expectation.

Exponential Tilting and Non-asymptotic Central Limit Lemmas for Random Walks To analyse our algorithm, we need probability bounds for events such as "survival of k generations" that are tight up to polynomial factors. This contrast with many typical analysis approaches in Computer Science, such as Chernoff bounds, which only need to be tight up to a constant in the exponent. We also can't use Central Limit type estimates, since they either are asymptotic (which correspond to assuming w_q and w_u are constants) or too weak (such as Berry Esseen) or just don't apply to random walks.

The technical tool we employ is "Exponential Tilting", which allows coupling the real pruned branching random walk to one that is much more well behaved. This can be seen as a nicer way of conditioning the random walk on succeeding. This nicer random walk then needs to be analysed for properties such as "probability that the path is always above the mean." This is shown using a rearrangement lemma, known as the Truck Driver's Lemma: Assume a truck driver must drive between locations $l_1, l_2, \ldots, l_n, l_1$. At stop i they pick up g_i gas, and between stop i and $i+1$ they expand e_i gas. The lemma say, that if the sum of $g_i - e_i$ is non-negative, then there is a starting position $j \in \{1, \ldots, n\}$ so that the driver's gas level never goes below 0.

This lemma gives an easy proof that a random walk on $\mathbb{R}_+$ of n identically distributed steps, must be always non-negative with probability at least $1/n$ times the probability that it is eventually non-negative. That's because, if the location is eventually non-negative, and all arrangements of steps happen with the same probability, then we must hit the "always non-negative" rotation with probability $\geq 1/n$.

Extending this argument to two dimensions turns out to require a few extra conditions, such as a positive correlation between the coordinates, but as a surprisingly key result, we manage to show Lemma 2.5, which says that for $k \in \mathbb{Z}_+$ and $p, p_1, p_2 \in [0,1]$, such that, pk, p_1k, and p_2k are integers and $p \geq p_1p_2$. Let $X^{(i)} \in \{0,1\}^2$ be independent identically distributed variables. We then get that

$$\Pr\left[\forall l \leq k : \sum_{i\in[k]} X^{(i)} \geq \left[\begin{smallmatrix} p_1 \\ p_2 \end{smallmatrix}\right] l \;\middle|\; \sum_{i\in[k]} X^{(i)} = \left[\begin{smallmatrix} p_1 \\ p_2 \end{smallmatrix}\right] k \wedge \sum_{i\in[k]} X_1^{(i)} X_2^{(i)} = pk\right] \geq k^{-3}\,.$$

Output-sensitive set decoding In our algorithm we are careful to not have factors of $|U|$ and $|X|$ (the size of the sets) on our query time and space bounds. When sampling our tree, at each level we must pick a certain number, Δ, of elements from the universe and check which of them are contained in the set being decoded. This is an issue, since Δ may be much bigger than $X \cap \Delta$, and so we need an "output-sensitive" sampling procedure. We do this by substituting random sampling with a two-independent hash function $h : U^k \to [q]$, where q is a prime number close to $|U|$. The sampling criterion is then $h(r \circ x) \le \Delta$, where $\circ$ is string concatenation. The function $h(r)$ can be taken to be $\sum_{i=1}^k a_i x_i + b \pmod q$ for random values $a_1, \dots, a_k, b \in [q]$, so we can expand $h(r \circ x)$ as $h(r) + a_k x \pmod q$.

Now

$$\begin{aligned}\{x \in X \mid (h(r \circ x) \mod q) < \Delta\} &= \{x \in X \mid (h(r) + a_k x \mod q) < \Delta\} \\ &= \cup_{i=0}^{\Delta-1} \{x \in X \mid a_k x \equiv \Delta - h(r) \mod q\} \\ &= \{x \in X \mid (a_k x \mod q) \in [-h(r), \Delta - h(r)] \mod q\},\end{aligned}$$

where the last equation is adjusted in case $(-h(r) \mod q) > (\Delta - h(r) \mod q)$. By pre-computing $\{a_k x \mod q \mid x \in X\}$ (just has to be done one for each of roughly $\log n$ levels in the tree), and storing the result in a predecessor data-structure (or just sorting it), the sampling can be done it time proportional to the size of its output.

Lower Bounds and Hypercontractivity The structure of our lower bounds is by now standard: We first reduce our lower bound to random instances by showing that with high probability the random instances are in fact an instance of our problem. For this to work, we need $w_w |U| = \omega(\log n)$ and in particular $|U| = \omega(\log n)$, so we get concentration around the mean. This requirement is indeed known to be necessary, since the results of [16, 22] break the known lower bounds in the "medium dimension regime" when $|U| = O(\log n)$.

The main difference compared to previous bounds is that we study Boolean functions on so-called p-biased spaces, where the previous lower bounds used Boolean functions on unbiased spaces. This is necessary for us to lower bound every parameter choice for GapSS. In particular we are interested in tight hypercontractive inequalities on p-biased spaces. We say that a distribution $\mathcal{P}_{XY}$ on a space $\Omega_X \times \Omega_Y$ is (r, s)-hypercontractive if

$$\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[f(X)g(Y)] \le \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X}[f(X)^r]^{1/r} \mathop{\mathrm{E}}_{Y\sim\mathcal{P}_Y}[g(Y)^s]^{1/s} ,$$

for all functions $f : \Omega_X \to \mathbb{R}$ and $g : \Omega_Y \to \mathbb{R}$, where $\mathcal{P}_X$ and $\mathcal{P}_Y$ are the marginal distributions on the spaces Ω_X and Ω_Y respectively. On unbiased spaces, the classic Bonami-Beckner inequality [19, 17] gives a complete understanding of the hypercontractivity. Unfortunately, this is not the case for p-biased spaces where the hypercontractivity is much less understood, with [53] and [68] being state of the art. We sidestep the issue of finding tight hypercontractive inequalities by instead showing an equivalence between hypercontractivity and KL-divergence, which is captured in the following lemma:[11]

Lemma 1.1. *Let $\mathcal{P}_{XY}$ be a probability distribution on a space $\Omega_X \times \Omega_Y$ and let $\mathcal{P}_X$ and $\mathcal{P}_Y$ be the marginal distributions on the spaces Ω_X and Ω_Y respectively. Let $s, r \in [1, \infty)$, then the following is equivalent*

1. *For all functions $f : \Omega_X \to \mathbb{R}$ and $g : \Omega_Y \to \mathbb{R}$,*

$$\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[f(X)g(Y)] \le \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X}[f(X)^r]^{1/r} \mathop{\mathrm{E}}_{X\sim\mathcal{P}_Y}[g(Y)^s]^{1/s} .$$

2. *For all probability distributions $\mathcal{Q}_{XY} \ll \mathcal{P}_{XY}$,*

$$\mathrm{D}(\mathcal{Q}_{XY} \,\|\, \mathcal{P}_{XY}) \ge \frac{\mathrm{D}(\mathcal{Q}_X \,\|\, \mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y \,\|\, \mathcal{P}_Y)}{s} ,$$

where $\mathcal{Q}_X$ and $\mathcal{Q}_Y$ be the marginal distributions on the spaces Ω_X and Ω_Y respectively

[11] It appears that one might prove a similar result using [50] and [36].

The main technical argument needed for proving Lemma 1.1 is that, for all probability distributions $\mathcal{P}, \mathcal{Q}$, where $\mathcal{Q}$ is absolutely continuous with respect to $\mathcal{P}$, and all functions ϕ,

$$\mathrm{D}(\mathcal{Q} \,\|\, \mathcal{P}) + \log \mathop{\mathrm{E}}_{X\sim\mathcal{P}} [\exp(\phi(X))] \geq \mathop{\mathrm{E}}_{X\sim\mathcal{Q}} [\phi(X)] \,.$$

This can be seen as a version of Fenchel's inequality, which says that $f(x) + f^*(p) \geq xp$ for all convex functions f, f^*, where f^* is convex conjugate of f, and all $x, p \in \mathbb{R}$.

We use Lemma 1.1 together with the "Two-Function Hypercontractivity Induction Theorem" [52], which shows that if $\mathcal{P}_{XY}^{\otimes n}$ is (r, s)-hypercontractive if and only if $\mathcal{P}_{XY}$ is (r, s)-hypercontractive. This implies that $\mathrm{E}_{(X,Y)\sim\mathcal{P}_{XY}^{\otimes n}} [f(X)g(Y)] \leq \mathrm{E}_{X\sim\mathcal{P}_X^{\otimes n}} [f(X)^r]^{1/r} \, \mathrm{E}_{X\sim\mathcal{P}_Y^{\otimes n}} [g(Y)^s]^{1/s}$ for all functions f, g if and only $\mathrm{D}(\mathcal{Q}_{XY} \,\|\, \mathcal{P}_{XY}) \geq \frac{\mathrm{D}(\mathcal{Q}_X \,\|\, \mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y \,\|\, \mathcal{P}_Y)}{s}$ for all probability distributions $\mathcal{Q}_{XY}$. In the proof of Theorem 3 we have $\mathcal{P}_{XY} = \begin{bmatrix} w_1 & w_q - w_1 \\ w_u - w_1 & 1 - w_q - w_u + w_1 \end{bmatrix}$ and consider all the probability distributions of the form $\mathcal{Q}_{XY} = \mathop{\arg\inf}_{\mathcal{Q}_{XY} \ll \mathcal{P}_{XY}, \mathop{E}_{X\sim\mathcal{Q}_{XY}}[X] = [\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}]} \mathrm{D}(\mathcal{Q}_{XY} \,\|\, \mathcal{P}_{XY})$ for $t_q, t_u \in [0, 1]$.

The obtained inequalities can be used directly with the framework by Panigrahy et al. [57] to obtain bounds on "Robust Expansion", which has been shown to give lower bounds for 1-cell and 2-cell probe data structures, with word size $n^{o(1)}$ and $o(\log n)$ respectively.

The Directed Noise Operator We extend the range of our lower bounds further, by studying a recently defined generalization of the p-biased noise operator [4, 2, 45, 43]. This "Directed Noise Operator", $T_\rho^{p_1\to p_2} : L_2(\{0,1\}^d, \pi_{p_1}^{\otimes d}) \to L_2(\{0,1\}^d, \pi_{p_2}^{\otimes d})$ has the property $\widehat{T_\rho^{p_1\to p_2} f}^{(p_2)}(S) = \rho^{|S|} \hat{f}^{(p_1)}(S)$ for any $S \subseteq [d]$, where $\hat{f}^{(p)}(S)$ denotes the p-biased Fourier coefficient of f. Just like the Ornstein Uhlenbeck operator, we show that $T_\sigma^{p_2\to p_3} T_\rho^{p_1\to p_2} = T_{\rho\sigma}^{p_1\to p_3}$ and that $T_\rho^{p_2\to p_1}$ is the adjoint of $T_\rho^{p_1\to p_2}$. By connecting this operator to our hypercontractive theorem, we can integrate the results by Oleszkiewicz and obtain provably optimal points on the (t_q, t_u) trade-off.

We show that for p-biased distributions over $\{0,1\}^n$, we can add the following line to the list of equivalent statements in Lemma 1.1:

3. For all functions $f : \{0,1\}^n \to \mathbb{R}$ it holds $\|T_\rho^{p_1\to p_2} f\|_{L_{s'}(p_1)} \leq \|f\|_{L_r(p_2)}$.

The operator allows us to prove some optimal choices for r and s in Lemma 1.1 (and by effect for t_q and t_u.) Following [4] we use Pareseval's identity, to write $\|T_\rho^{p_1\to p_2} f\|_{L_2(p_2)}^2$ as

$$\widehat{T_\rho^{p_1\to p_2} f}^{(p_2)}(\emptyset)^2 + \widehat{T_\rho^{p_1\to p_2} f}^{(p_2)}(\{1\})^2 = \widehat{f}^{(p_1)}(\emptyset)^2 + \rho^2 \widehat{f}^{(p_1)}(\{1\})^2 = \|T^{p_1\to p_1} f\|_{L_2(p_1)}^2 \leq \|f\|_{L_r(p_1)}^2 \,,$$

where r is perfectly determined by Oleszkiewicz in [53]. It is possible to prove further lower bounds using Hölder's inequality on T, however the bounds obtained this way turn out to be optimal only in the case $s = 2$ or $r = 2$ that also follow from Parseval. A particular simple case is $r = s =$, $w_q = w_u = w$, and $w_2 = w^2$, in which case the arguments above gives the lower bound $\rho \geq \log(\frac{w_1(1-w)}{w(1-2w+w_1)}) / \log(\frac{1-w}{w})$ mentioned in Example 1 in the Upper Bounds section.

Another use of T is in proving lower bounds outside of the random instance $w_2 = w_q w_u$ regime. Using the power means inequality over p-biased Fourier coefficients, we show the relation

$$\left(\langle T_\alpha^{p\to p} f, f\rangle_{L_2(p)} / \|f\|_{L_2(p)}^2\right)^{1/\log(1/\alpha)} \leq \left(\langle T_\beta^{p\to p} f, f\rangle_{L_2(p)} / \|f\|_{L_2(p)}^2\right)^{1/\log(1/\beta)} .$$

which is allows comparing functions under two different noise levels. This is stronger than hypercontractivity, even though we can prove it in fewer instances. The proof can been seen as a variation of [54] and we get a lower bound with a similar range, but without asymptotics and for Set Similarity instead of Hamming space Similarity Search.

1.5 Related Work

For the reasons laid out in the introduction, we will compare primarily against approximate solutions. The best of those are all able to solve GapSS, thus making it easy to draw comparisons. The guarantees of these algorithms are listed in Table 1 and we provide plots in Figure 2 and Figure 3 for concreteness.

The methods known as Bit Sampling [39] and SimHash (Hyperplane rounding) [24], while sometimes better than MinHash[21] and Chosen Path [28] are always worse (theoretically) that Spherical LSF, so we won't perform a direct comparison to those.

It should be noted that both Chosen Path and Spherical LSF both have proofs of optimality in the restricted models. However these proofs translated to only a certain region of the (w_q, w_u, w_1, w_2) space, and so they may nearly always be improved.

Arguably the largest break-through in Locality Sensitive Hashing, LSH, based data structures was the introduction of *data-dependent* LSH [8, 11, 12]. It was shown how to reduce the general case of α, β similarity search as described above, to the case $(\alpha, \beta) \mapsto (\frac{\alpha-\beta}{1-\beta}, 0)$, in which many LSH schemes work better. Using those data structures on GapSS with $w_2 > w_q w_u$ will often yield better performance than the algorithms described in this paper. However, in the "random instance" case $w_2 = w_q w_u$, which is the main focus of this paper, data-dependency has no effect, and so this issue won't show up much in our comparisons.

We note that even without a reduction to the random instance, for many practical uses, it is natural to assume such "independence" between the query and most of the dataset. Arguably this is the main reason why approximate similarity search algorithms have gained popularity in the first place. In practice, some algorithms for Set Similarity Search take special care to handle "skew" data distributions [61, 70, 47], in which some elements of the Universe are heavily over or under-represented. By special casing those elements, those algorithms can be seen as reducing the remaining dataset to the random instance. Curiously, even the early research on Partial Match by Ronald Rivest in his PhD thesis [62], studied the problem on random data.

Many of the algorithms, based on the LSH framework, all had space usage roughly $n^{1+\rho}$ and query time n^ρ for the same constant ρ. This is known as the "balanced regime" or the "LSH regime". Time/space trade-offs are important, since $n^{1+\rho}$ can sometimes be too much space, even for relatively small ρ. Early work on this was done by Panigrahy [56] and Kapralov [41] who gave smooth trade-offs ranging from space $n^{1+o(1)}$ to query time $n^{o(1)}$. A breakthrough was the use of LSF (rather than LSH), which allowed time/space trade-offs with sublinear query time even for near linear space and small approximation [44, 27, 10].

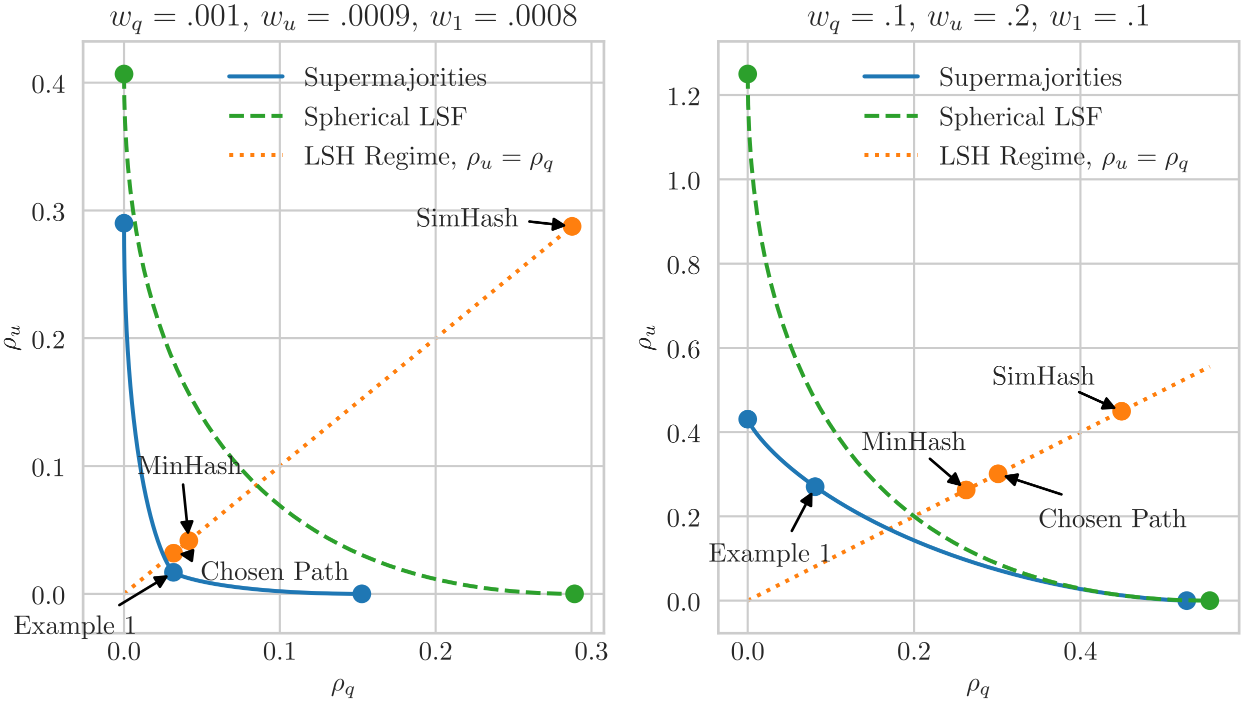
We finally compare our results to the classical literature on Partial Match and Super-/Subset search, which has some intriguing parallels to the work presented here.

Comparison to Spherical LSF We use "Spherical LSF" as a term for the algorithms [16] and [44], but in particular section 3 of [9], which has the most recent version. The algorithm solves the (r, cr)-Approximate Near Neighbour problem, in which we, given a dataset $Y \subseteq \mathbb{R}^d$ and a query $q \in \mathbb{R}^d$ must return $y \in Y$ such that $\|q - y\| < cr$ or determinate that there is no $y' \in Y$ with $\|y - q\| \leq r$.

The algorithm is a tree over the points, P. At each node they sample T i.i.d. Gaussian d-dimensional vectors $z_1, \ldots, z_T$ and split the dataset up into (not necessarily disjoint) "caps" $P_i = \{p \in P \mid \langle z_i, p\rangle \geq t_u\}$. They continue recursively and independently until the expected number of leaves shared between two points at distance $\geq cr$ is $\approx n^{-1+\varepsilon}$.

Method	**Balanced** $\rho_q = \rho_u$	**Space/time trade-offs**
Spherical LSF [66, 44, 27, 9]	$\frac{1-\alpha}{1+\alpha}\frac{1+\beta}{1-\beta}$	$\rho_q = \frac{(1-\alpha^{1+\lambda})^2}{1-\alpha^2}\frac{1-\beta^2}{(1-\alpha^\lambda\beta)^2}^{(***)}$ $\rho_u = \frac{(1-\alpha^{1+\lambda})^2}{1-\alpha^2}\frac{1-\beta^2}{(1-\alpha^\lambda\beta)^2}$
MinHash [21]	$\frac{\log \frac{w_1}{w_q+w_u-w_1}}{\log \frac{w_2}{w_q+w_u-w_2}}$	Same as above$^{(*)}$ with $\alpha = \frac{w_1}{w_q+w_u-w_1}, \beta = \frac{w_2}{w_q+w_u-w_2}$
Chosen Path [28]	$\frac{\log \frac{w_1}{\max\{w_q,w_u\}}}{\log \frac{w_2}{\max\{w_q,w_u\}}}$	N/A
Supermajorities (This paper)	Theorem 1, Example 1	Theorem 1
Data-Dependent LSF [11, 9]	$\frac{1-\alpha}{1+\alpha-2\beta}$	$\sqrt{\rho_q} + \alpha'\sqrt{\rho_u} = \sqrt{1-\alpha'^2}$ where $\alpha' = 1 - \frac{1-\alpha}{1-\beta}$
SimHash [24]	$\frac{\log(1-\arccos(\alpha)/\pi)}{\log(1-\arccos(\alpha)/\pi)}$	N/A$^{(**)}$
Bit Sampling [39]	$\frac{\log(1-w_q-w_u+2w_1)}{\log(1-w_q-w_u+2w_2)}$	N/A$^{(**)}$

Table 1: Time and space exponents for the best similarity search data-structures. For Spherical LSF and SimHash, α and β are the inner products between sets represented as vectors, and can by Lemma 4.1 be taken to be $\alpha = \frac{w_1-w_qw_u}{\sqrt{w_q(1-w_q)w_u(1-w_u)}}$ and $\beta = \frac{w_2-w_qw_u}{\sqrt{w_q(1-w_q)w_u(1-w_u)}}$.
$(*)$: Space/time trade-offs for MinHash can be obtained using MinHash as an embedding for Spherical LSF. $(**)$: Some space/time trade-offs can be obtained for LSH using Multi-probing [46]. $(***)$: $\lambda \in [-1, 1]$ controls the space/time trade-off.



(a) Example of search with very small sets.

(b) Example with larger sets of different sizes.

Figure 2: Comparison to Spherical LSF: Plots of the achievable ρ_q (time exponent) and ρ_u (space exponent) achievable with Theorem 1. Note that using our optimal spherical embedding from Lemma 4.1 is critical to achieve the exponents shown for Spherical LSF. The plots are drawn in the "random setting", $w_2 = w_q w_u$ where Spherical LSF and Data-Dependent LSH coincide.

The real algorithm also samples includes some caps that are dependent on an analysis of the dataset. This allows obtaining a query time of $n^{1/(2c^2-1)}$, for all values of r, rather than only in the "random instance", which, for data on the sphere, corresponds to $r = 1/(\sqrt{2}c)$. (To see this, notice that $rc = 1/\sqrt{2}$, which is the expected distance between two orthogonal points on a sphere.)

Whether we analyse the data-independent algorithm or not, however, a key property of Spherical LSF is that each node in the tree is independent of the remaining nodes. This allows a nice inductive analysis. In comparison, in our algorithm, the nodes are not independent. Whether a certain node gets pruned, depends on which elements from the universe were sampled at all the previous nodes along the path from the root. One could imagine doing Spherical LSF with a running total of inner products along each path, which would make the space partition more smooth, and possible better in practice. Something along these lines was indeed suggested in [16], however it wasn't analysed, as for Spherical LSF *the inner products at each node are continuous, and the thresholds can be set at any precision.*

It is clear that Spherical LSF can solve GapSS – one simply needs an embedding of the sets onto the sphere. An obvious choice is $x \mapsto x/\|x\|_2$. This was used in [28] when comparing Chosen Path to Spherical LSF. However it is also clear that the choice of embedding matters on the performance one gets out of Spherical LSF. Other authors have considered $x \mapsto (2x-1)/\sqrt{d}$ and various asymmetric embeddings [64].

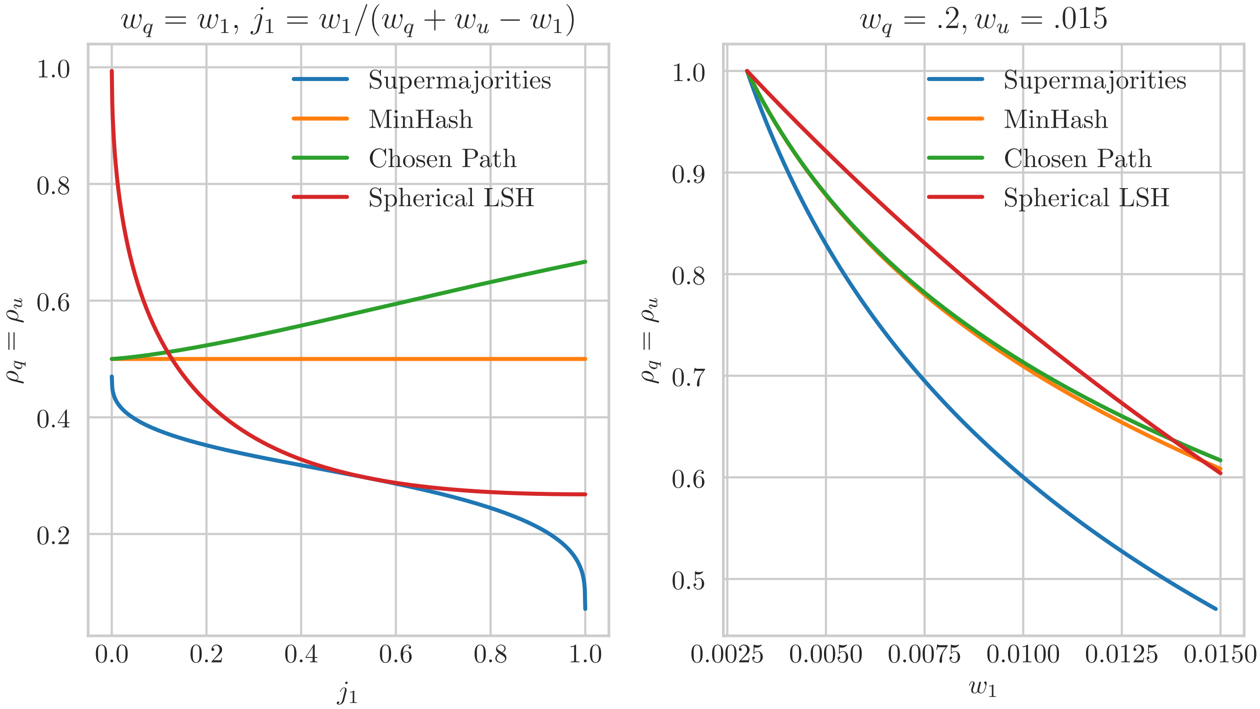
We would like to find the most efficient embedding to get a fair comparison. However, we don't know how to do this optimally over all possible embeddings, which include using MinHash and possibly somehow emulating Supermajorities.[12] We instead find the most efficient *affine* embedding, which turns out to be surprisingly simple, and which encompasses all previously suggested approaches. In Lemma 4.1 we prove a general result, implying that the embedding is optimal for Spherical LSF as well as other spherical data structures like SimHash. In Figure 2 and Figure 3 the ρ-values of Spherical LSF are obtained using this optimal embedding.

From the figures, we see the two main cases in which Spherical LSF is suboptimal. As the sets get very small ($w_q, w_u, w_1 \to 0$) the ρ value in the LSH regime goes to 1, whereas Supermajorities (as well as MinHash and Chosen Path) still obtain good performance. Similarly in the asymmetric case $w_q \neq w_u$, as we make ρ_q very small, the performance gap between Supermajorities and Spherical LSF can grow to arbitrarily large polynomial factors.

Comparison to MinHash Given a random function $h : \mathcal{P}(\{1,\dots,d\}) \to [0,1]$, the MinHash algorithm hashes a set $x \subseteq \{1,\dots,d\}$ to $m_h(x) = \arg\min_{i\in x} h(i)$. One can show that $Pr[m_h(x) = m_h(y)] = J(x,y) = \frac{|x\cap y|}{|x\cup y|}$. Using the LSH framework by Indyk and Motwani [39] this yields a data structure for Approximate Set Similarity Search over Jaccard similarity, J, with query time dn^ρ and space usage $n^{1+\rho} + dn$, where $\rho = \frac{\log j_1}{\log j_2}$ and j_1 and j_2 define the gap between "good" and "bad" search results. As Jaccard similarity is a set similarity measures, it is clear that MinHash yields a solution to the GapSS problem with $\rho_q = \rho_u = \log \frac{w_1}{w_q+w_u-w_1} \big/ \log \frac{w_2}{w_q+w_u-w_2}$. Similarly, and that any solution to GapSS can yield a solution to Approximate SSS over Jaccard similarity.

MinHash has been very popular, since it gives a good, all-round algorithm for Set Similarity Search, that is easy to implement. In Figure 3 we see how MinHash performant for different settings of GapSS. In particular we see that when solving the Superset Search problem, which is a common use case for MinHash, our new algorithm obtains quite a large polynomial improvement, except when the Jaccard similarity between the query and the sought after superset is nearly 0 (which is hardly an interesting situation.)

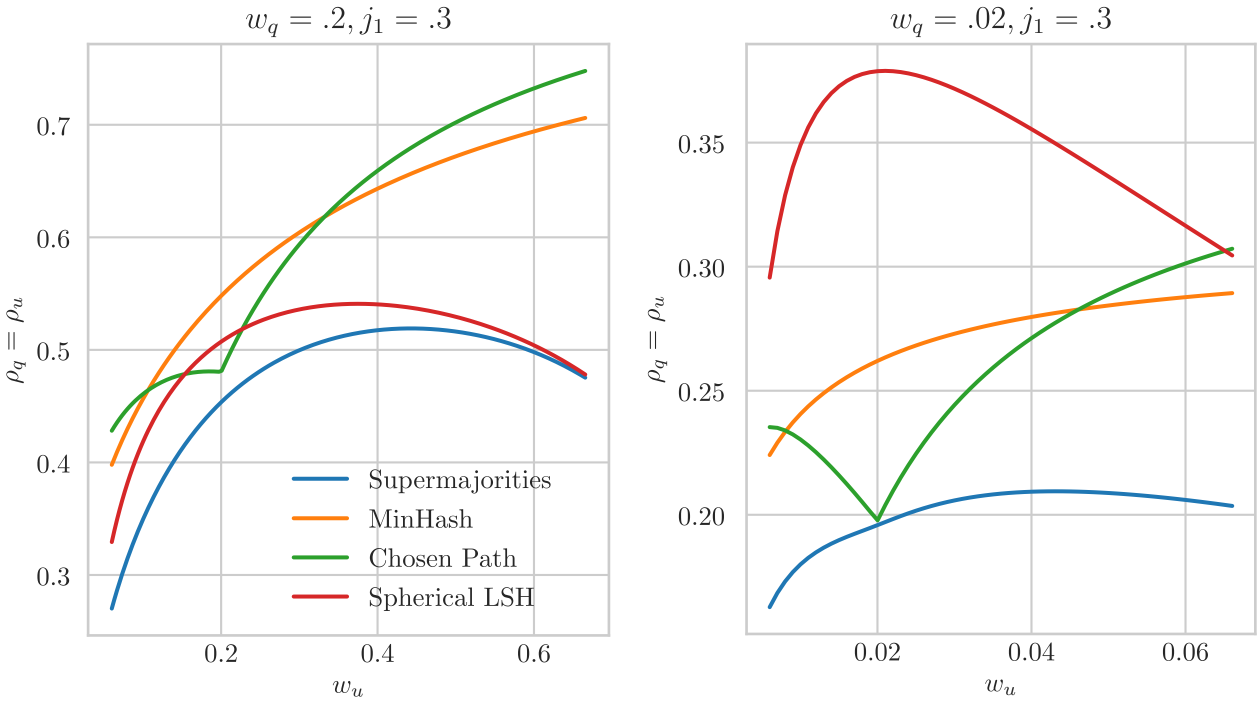
[12] We would also need some sort of limit on how much time the embedding takes to perform.



(a) The plot shows the effect on the balanced exponent of Supermajorities as we the Jaccard similarity while fixing the exponent of MinHash at $\rho = .5$.

(b) Varying the overlap w_1 among close sets while fixing the query and database set sizes. At $w_1 = .002$ there is no gap between close and far sets.

Figure 3: Comparison to MinHash: Varying different parameters while searching on a background of random sets ($w_2 = w_q w_u$), Supermajorities regularly get substantially better time and space exponents. The plots are drawn in the "random setting", $w_2 = w_q w_u$ and use the optimal embedding for Spherical LSF.



(a) In the plot, Chosen Path never matches Supermajorities, even at $w_q = w_u$ since the sets are relatively large.

(b) In the plot, Chosen Path nearly matches Supermajorities when $w_u = w_q$ as the sets are relatively small.

Figure 4: Comparison to Chosen Path: Fixing w_q and the Jaccard similarity so $w_1 = \frac{j_1}{1+j_1}(w_q+w_u)$, we vary w_u to see the performance of different algorithms at different levels of asymmetry in the set sizes. The plots are drawn in the "random setting", $w_2 = w_q w_u$ and use the optimal embedding for Spherical LSF.

It is possible to use MinHash as an embedding (or densification) of sets into Hamming space or onto the Sphere. We can then use Spherical LSF to get space/time trade-offs. We have not plotted those, but we can notice that in the balanced case, $\rho_q = \rho_u$, this would give $\rho = \frac{1-j_1}{1+j_1}\frac{1+j_2}{1-j_2}$, which is worse than $\rho = \log j_1 / \log j_2$ obtained by the direct algorithm.

MinHash is quite different from the other algorithms considered in this section. For some more intuition of why MinHash is not optimal for Approximate Set Similarity Search, we show in Section 4.2 that MinHash can be seen as an average of a family of Chosen Path like algorithms. We also show that an average is always worse than simply using the best family member, which implies that MinHash is never optimal.

Comparison to Chosen Path The Chosen Path algorithm of [28], is virtually identical to Supermajorities, when parametrized with $t_q = t_u = 1$. Similar to Spherical LSF and our decoding algorithm, they build a tree on the datasets. For each node they sample iid. Elements $x_1, x_2, \dots \in U$ from the universe, and split the data into (not necessarily disjoint) subsets $P_i = \{p \in P \mid x_i \in p\}$. They again continue recursively and independently until the expected number of leaves shared between two dissimilar points is sufficiently small.

The case $t_q = t_u = 1$ however, turns out to be a very special case of our algorithm, because one can decide which leaves of the tree to prune, without knowledge of what happened previously on the path from the root to the node. This allows a nice inductive analysis of Chosen Path based on second moments, which is a classic example literature on branching processes. Meanwhile, for our general algorithm, we need to analyse the resulting branching random walk, a conceptually much different beast.

Doing the analysis, one gets a data structure for Approximate Set Similarity Search over Braun-Blanquet similarity, $B(x, y) = \frac{|x \cap y|}{\max\{|x|,|y|\}}$, with query time $|q|n^{\rho}$ and auxiliary space usage $n^{1+\rho}$, where $\rho = \frac{\log b_1}{\log b_2}$ and b_1 and b_2 define the gap between "good" and "bad" search results. Since $t_q = t_u = 1$ is sometimes the optimal choice for Supermajorities, it is clear that we must sometimes coincide in performance with Chosen Path. In particular, this happens as $w_q = w_u$ and $w_q, w_u, w_1 \to 0$. This is also one of the case where our lower bound Theorem 2 is sharp, which confirms, in addition to the lower bound in [28] that both algorithms are sharp for LSF data structures in this setting. Figure 2a shows how Chosen Path does nearly as well as Supermajorities on very small sets.

In the case $w_q = w_u$ the ρ value of Chosen Path can be equivalently written in terms of Jaccard similarities as $\log \frac{2j_1}{1+j_1} \big/ \log \frac{2j_2}{1+j_2}$, which is always smaller than the $\log j_1 \big/ \log j_2$ obtained by MinHash. (This value, $2j/(1 + j)$, is also known as the Sørensen-Dice coefficient of two sets.) However, in the case $w_q \neq w_u$ Chosen Path can be much worse than MinHash, as seen in Figure 2b and Figure 3a. In [28] it was left as an open problem whether MinHash could be improved upon in general. It is a nice result that the balanced ρ value of Supermajorities (when $\rho_q = \rho_u$) can be shown (numerically) to always be less than or equal to $\log \frac{2j_1}{1+j_1} \big/ \log \frac{2j_2}{1+j_2}$, even when $w_q \neq w_u$. It is a curious problem for which similarity measure, S, so the balanced ρ value of Supermajorities equal $\log s_1 \big/ \log s_2$.

Partial Match (PM) and Super-/Subset queries (SQ) Partial Match asks to pre-process a database D of n points in $\{0, 1\}^d$ such that, for all query of the form $q \in \{0, 1, *\}^d$, either report a point $x \in D$ matching all non-$*$ characters in q or report that no such x exists. A related problem is Super-/Subset queries, in which queries are on the form $q \in \{0, 1\}^d$, and we must either report a point $x \in D$ such that $x \subseteq q$ (resp. $q \subseteq x$) or report that no such x exists.

The problems are equivalent to the subset query problem by the following folklore reductions: (PM → SQ) Replace each $x \in D$ by the set $\{(i, p_i) : i \in [d]\}$. Then replace each query q by $\{(i, q_i) : q_i = *\}$. (SQ → PM) Keep the sets in the database as vectors and replace in each query each 0 by an $*$.

The classic approach, studied by Rivest [62], is to split up database strings like `supermajority` and file them under `s`, `u`, `p` etc. Then when given query like `set` we take the intersection of the lists `s`, `e`, `t`. Sometimes this can be done faster than brute force searching each list. He also considered the space heavy solution of storing all subsets, and showed that when $d \leq 2 \log n$, the trivial space bound of 2^d can be somewhat improved. Rivest finally studied approaches based on tries and in particular the case where most of the database was random strings. The latter case is in some ways similar to the LSH based methods we will describe below.

Indyk, Charikar and Panigrahy [23] also studied the exact version of the problem, and gave, for each $c \in [n]$, an algorithm with $O(n/2^c)$ time and $n2^{(O(d \log^2 d \sqrt{c/\log n})}$ space, and another with $O(dn/c)$ query time and nd^c space. Their approach was a mix between the shingling method of Rivest, building a look-up table of size $\approx 2^{\Omega(d)}$, and a brute force search. These bounds manage to be non-trivial for $d = \omega(\log n)$, however only slightly. (e.g. $n/\operatorname{poly}(\log n)$ time with polynomial space.)

There has also been a large number of practical papers written on Partial Match / Subset queries or the equivalent batch problem of subset joins [60, 48, 37, 3, 34]. Most of these use similar

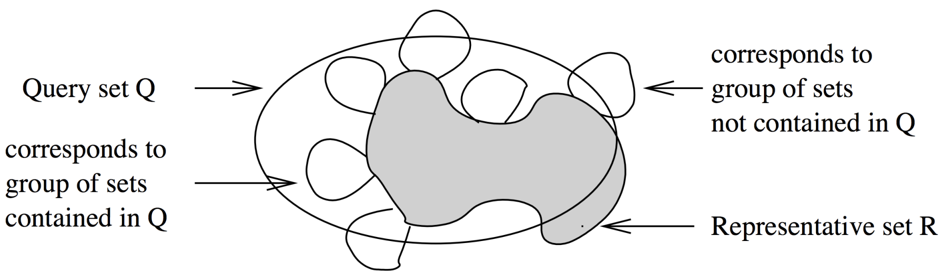

Query set Q
corresponds to group of sets contained in Q
corresponds to group of sets not contained in Q
Representative set R

Figure 5: This figure from the Partial Match algorithm of [23] shares some of the same geometrical intuition visible in our own figure 1a.

methods to the above, but save time and space in various places by using bloom filters and sketches such as MinHash [21] and HyperLogLog [35].

Maximum Inner Product (MIPS) is the Similarity Search problem with $S(x, y) = \langle x, y\rangle$ — the Euclidean inner product. For exact algorithms, most work has been done in the batch version (n data points, n queries). Here Alman et al. [6] gave an $n^{2-1/\tilde{O}(\sqrt{k})}$ algorithm, when $d = k \log n$.

An approximative version can be defined as: Given $c > 1$, pre-process a database D of n points in $\{0, 1\}^d$ such that, for all query of the form $q \in \{0, 1\}^d$ return a point $x \in D$ such that $\langle q, x\rangle \geq \frac{1}{c} \max_{x' \in D}\langle q, x'\rangle$. Here [5] gives a data structure with query time $\approx \tilde{O}(n/c^2)$, and [25] solves the batch problem in time $n^{2-1/O(\log c)}$ (both when d is $n^{o(1)}$.)

There are a large number of practical papers on this problem as well. Many are based on the Locality Sensitive Hashing framework (discussed below) and have names such as SIMPLE-LSH [51] and L2-ALSH [64]. The main problem for these algorithms is usually that no hash family of functions $h : \{0, 1\}^d \times \{0, 1\}^d \to [m]$ such that $\Pr[h(q) = h(x)] = \langle q, x\rangle/d$ [5] and various embeddings and asymmetries are suggested as solutions.

The state of the art is a paper from NeurIPS 2018 [69] which suggests partitioning data by the vector norm, such that the inner product can be more easily estimated by LSH-able similarities such as Jaccard. This is curiously very similar to what we suggest in this paper.

We will not discuss these approaches further since, for GapSS, they all have higher exponents than the three LSH approaches we study next.

2 The Algorithm

We now describe the full algorithm that gives Theorem 1. We state the full version of the theorem, discuss it and prove it. The section ends with an involved analysis of the survival probabilities of the branching random walk.

Notationally we define $[n] = \{1, \dots, n\}$ and let $(\cdot \circ \cdot) : A^{l_1} \times A^{l_2} \to A^{l_1+l_2}$ be the concatenation operator for any set A and integers l_1, l_2. We will use the Iversonian bracket, defined by $[P] = 1$ if P and 0 otherwise. For R and U sets, we have $R \times U = \{r \circ u \mid r \in R, u \in U\}$ $\mathcal{P}(U)$ is the power set of U.

The first step is to set up our assumptions. For $w_q, w_u, w_1, w_2, t_q, t_u \in [0, 1]$ given, we can assume $\min\{w_q, w_u\} \geq w_1 > w_2$ and $t_q \neq w_q, t_u \neq w_u$. We are also given a universe U and a family $Y \subseteq \binom{U}{w_u|U|}$ of size $|Y| = n$.

It will be nice to assume $|U| = q$ where q is a prime number. This can always be achieved by adding at most $|U|^{0.525}$ elements to U large enough[13] [14]. Hence we only distort each of

[13] It is an open conjecture by Harald Cramér that $(\log |U|)^2$ suffices as well. [31]

w_q, w_u, w_1, w_2 by roughly a factor $1 + O(|U|^{-1/2})$, which is insignificant for $|U| = \Omega(\log n)^2$, and we can always increase $|U|$ without changing the problem parameters by duplicating the set elements.

Let $k \in \mathbb{Z}_+$ be defined later. For all $i \in [k]$ we define $h_i(r) : [q]^i \to [q]$ by $h_i(r) = \sum_{j\in[i]} a_{i,j} r_j + b_i \mod q$ for some sequences of random numbers $a_{i,j} \in [q] \setminus \{0\}, b_i \in [q]$, such that each h_i is a 2-independent random function. (That means $\Pr[h_i(r) = h_i(r')] \le 1/q$ for $r \ne r'$.)

Finally two sequences $(\Delta_i \in \mathbb{Z}_+)_{i\in[k]}$ and $(c_\ell \in \mathbb{R}^2)_{\ell\in[k]}$ to be specified later. We can now define the sets $R_i = \{r \circ x \in R_{i-1} \times U \mid h_i(r \circ x) < \Delta_i\}$, as well as the decoding functions

$$\mathcal{R}_i(X, t) = \left\{ r \in R_i \,\middle|\, \forall \ell \le i : \sum_{j\in[\ell]} [r_j \in X] \ge t\ell - c_\ell \right\}$$

Intuitively R_i are our representative sets at level i in the tree, such that R_k is a close to iid. uniform sample from U^k. The decoding function takes a set $X \subseteq U$ and a value $t \in [0, 1]$, and returns all $r \in R_i$ such that all prefixes r' of r "$(t - \varepsilon)$-favours" X (as defined by $|r \cap X|/|r| \ge t - \varepsilon$ in the introduction), where $\varepsilon = \frac{c_{|r'|}}{|r'|}$ is some slack that helps ensure survival of at least one representative set. The slack won't be the same on each coordinate, but scaled by their variance. The algorithm is shown below as pseudo-code in Algorithm 1.

Algorithm 1: Pseudocode for the decoding function $\mathcal{R}$.

```
Input: Universe $U$, Set $X \subseteq U$, Threshold $t \in [0,1]$
Result: Set $P_k \subseteq U^k$ of paths
$R_0 \leftarrow \{((), 0)\}$                    // These $R_i$ values contain the paths and scores
for $i = 1$ to $k$ do
    $R_i \leftarrow \{\}$
    for $(r, s) \in R_{i-1}$ do
        for $x \in U$ st. $h_i(r \circ x) < \Delta_i$ do                 // Sample the universe
            $s' \leftarrow s + [x \in X]$
            if $s' \ge it - c_i$ then                              // Trim to promising paths
                $R_i \leftarrow R_i \cup \{(r \circ x, s')\}$
            end
        end
    end
end
```

Our data structure now builds a hash-table M of lists of pointers and store each set $y \in Y$ in $M[r]$ for every $r \in R_k(y, t_u)$. One can think of this as storing the elements at the leafs of the tree represented by the sets R_i. On a new query $q \in \binom{U}{w_q|U|}$ we look at every list $M[r]$ for $r \in R_k(q, t_q)$. For each y in such a list, we compute the intersection with q and return y if $|q \cap y|/|U| \ge w_2$. This takes time $\min\{w_u, w_q\}|U|$, which would be a large multiplicative factor on our query time, so we may instead choose to sample just

$$O(\min\{w_q, w_u\} w_2^{-1} \log n) \tag{1}$$

elements, which suffices as a test with high probability.

This describes the entire algorithm, exception for an optimization for the "Sample the universe" step above, which naively implemented would take time $|X|$. This optimization is the reason $|U|$ was chosen to be a prime number.

An optimization In the "Sample the universe" step of Algorithm 1 a naive implementation spends time $|X|$ hashing all possible elements and comparing their value to Δ_i. We now show how to make this step output sensitive, using only time equal to the number of values for which the condition is true. [14]

The requirement $s' \geq it - c_i$ we call the "trimming condition". This allows us to trim away most prefix paths which would be very unlikely to ever reach our requirement for the final path. To speed up finding all $x \in U$ such that $h_i(r \circ x) < \Delta_i$ we note that there are two cases relevant to the trimming condition, depending on s in the algorithm: (1) s' has to be $s+1$ or (2) $s' = s$ suffices. In the first case we are only interested in x values in X, while in the second case, all $x \in U$ values are relevant.

We have $h_i(r \circ x) = \eta + ax \mod q$ for some values η, a and b where $a > 0$. In case (2) the relevant x are simple $\{a^{-1}(v - \eta) \mod q \mid v \in [\Delta_i]\}$, where a^{-1} exists because q is prime. For the case (1) where x must be in X, we pre-process X by storing $ax \mod q$ for $x \in X$ in a sorted list. Using a single binary search, we can then find the relevant values with a time overhead of just $\lg |X|$. Using a more advanced predecessor data structure, this overhead can be reduced. See Algorithm 2 for a pseudocode version of this idea.

```
Algorithm 2: Output sensitive sample
  Input        : $r \in [q]$, $\Delta \in [q]$
  Pre-process: $s = \text{sorted}\{h(x) \mid x \in X\} \in [q]^{|X|}$ and $\kappa \in X^{|X|}$ st. $h(\kappa[i]) = s[i]$.
  Result: $R = \{x \in X \mid (h(x) + r \mod q) < \Delta\}$
  $i \leftarrow \min\{i \in [|X|] \mid s[i-1] < q - r \leq s[i]\}$          // We assume $s[i] = -\infty$ for $i < 0$
  $R \leftarrow \{\}$
  while $(s[i \mod |X|] + r \mod q) < \Delta$ do
      $R \leftarrow R \cup \{\kappa[i \mod |X|]\}$
      $i \leftarrow i + 1$
  end
```

2.1 Full Theorem

We state the full version of Theorem 1 and a discussion of the differences between it and the idealized version in the introduction.

Theorem 1 (Full version)**.** *Let $w_q, w_u \geq w_1 \geq w_2 \geq 0$ be given with $w_1 \geq w_q w_u$ and $1 \leq t_q, t_u \leq 0$. Set k to be the smallest even integer greater than or equal to $\frac{\log n}{\mathrm{D}(T_2 \,\|\, P_2) - \mathrm{d}(t_q \,\|\, w_q)}$ and assume that $t_q k/2$ and $t_u k/2$ are integers. The (w_q, w_u, w_1, w_2)-GapSS problem over a universe U can be solved with expected query time*

$$\begin{aligned} \textit{query time} \quad & O\big(\varsigma_q\, k^{28}\, n^{\rho_q} + k w_q\, |U| + \big(\tfrac{t_q(1-w_q)}{(1-t_q)w_q}\big)^{\sqrt{t_q(1-t_q)k \cdot 6.5 \log(3k))}}\big), \\ \textit{space usage} \quad & O(\varsigma_u\, k^{28}\, n^{1+\rho_u} + n w_u\, |U|) \\ \textit{and update time} \quad & O\big(\varsigma_u\, k^{28}\, n^{\rho_u} + k w_u\, |U| + \big(\tfrac{t_u(1-w_u)}{(1-t_u)w_u}\big)^{\sqrt{t_u(1-t_u)k \cdot 6.5 \log(3k))}}\big), \end{aligned}$$

[14] The subroutine is inspired by personal communications with Rasmus Pagh and Tobias Christiani.

$$\textit{where}\quad \rho_q = \frac{\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_q \,\|\, w_q)}{\mathrm{D}(T_2 \,\|\, P_2) - \mathrm{d}(t_q \,\|\, w_q)} \quad \textit{and} \quad \rho_u = \frac{\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_u \,\|\, w_u)}{\mathrm{D}(T_2 \,\|\, P_2) - \mathrm{d}(t_q \,\|\, w_q)},$$

$$\textit{and}\quad \varsigma_q = \tfrac{\min\{w_q, w_u\}}{w_2} e^{2(\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_q \,\|\, w_q))}, \quad \varsigma_u = e^{2(\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_u \,\|\, w_u))}.$$

We stress that all previous Locality Sensitive algorithms with time/space trade-offs had $n^{o(1)}$ factors on n^{ρ_q} and n^{ρ_u}. These could be as large as $\exp(\sqrt{\log n})$ or even $\exp((\log n)/(\log\log n))$. In contrast, our algorithm is the first that only loses $k \approx \log(n)$ multiplicative factors!

In the statement of Theorem 1 we have taken great effort to make sure that any dependence on $w_q, w_u, w_1, w_2, t_q, t_u$ is visible and only truly universal constants, like 4, are hidden in the $O(\cdot)$.

The main thing we do lose is the additive $(\frac{t_q(1-w_q)}{(1-t_q)w_q})^{\tilde{O}(\sqrt{t_q(1-t_q)k})}$. We may note the bound $(\frac{t_q}{1-t_q})^{\sqrt{t_q(1-t_q)}} \le 2$, so the main eyesore is the $1/w_q$. For $w_q > e^{-\tilde{O}(\sqrt{\log n})}$ this is dominated by the main term, but for very small sets it could potentially be an issue. However, it turns out that as w_q and w_u get small, the optimal choices of t_q and t_u move towards 0 or 1. Since this effect is exponentially stronger we get that $(1/w_q)^{\sqrt{t_q(1-t_q)}}$ is usually never more than a small constant. It also means that we recover the performance of Chosen Path in the case $t_q = 1$, $t_u = 1$, which has no $\Omega(e^{\sqrt{\log n}})$ terms. [15]

In case w_q^{-1} is large, but w_2 is not too small, we can reduce w_q^{-1} to $\frac{w_u}{w_2}k$ by hashing! Sketch: Define a hash function $h : U \to [m]$ where $m = O(\frac{w_q w_u}{w_2}|U|k)$ and map each set y to $\{i \in [m] \mid \exists e \in y : h(e) = i\}$, that is the OR of the hashed values. With high probability this only distorts the size of the sets and their inner products by a factor $(1 + 1/k)$ which doesn't change ρ.

The constants of the size ς_q and ς_u are standard in all other similar algorithms since [39], as they come from the requirement that k is an integer. The terms $\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_q \,\|\, w_q)$ and $\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_u \,\|\, w_u)$ in ς_q and ς_u may be bounded by $\log\frac{w_u}{w_1}$ and $\log\frac{w_q}{w_1}$ respectively. The factor of 2 on those terms come from the tensoring step done on paths of length $k/2$. This can be removed at the cost of making the ratio-of-odds term multiplicative in the bounds above. The factor $\min\{w_q, w_u\}/w_2$ in ς_q comes from equation (1) and is the time it takes to verify a candidate identified by the filtering. Note that this factor would exist even in a brute force $O(n)$ algorithm and exists in any data structures known for similar problems. In fact, for small n, it is necessary due to communication complexity bounds.

Proof of Theorem 1. Let $\mathcal{T}_q$ and $\mathcal{T}_u$ be the time it takes to compute $R_k(x, t_q)$ and $R_k(y, t_u)$ on given sets. When creating the data structure, decoding each $y \in Y$ takes time $n\mathcal{T}_u$ and uses $n\,\mathrm{E}\left[|R_k(Y, t_u)|\right]$ words of memory for space equivalent. When querying the data structure we first use time $\mathcal{T}_q$ to decode q, then $\mathrm{E}\left[|R_k(X, t_q)|\right]$ time to look in the buckets, and finally $\frac{\min\{w_q, w_u\}\log n}{w_2}$ time on each of $\mathrm{E}\left[|R_k(X, t_q) \cap R_k(Y, t_u)|\right] n$ expected collisions with far sets (the worst case is that we never find any y with $y \cap q > w_2|U|$ so we can't return early.)

The key to proving the theorem is thus bounding the above quantities. We do this using the following lemma, which we prove at the end of the section:

Lemma 2.1. *In Algorithm 1 let $k \in \mathbb{Z}_+$ and let $w_q, w_u, w_1, w_2 \in [0, 1]$ be the Gap-SS parameters such that $w_1 \ge w_q w_u$. Now let $t_q, t_u \in [0, 1]$ be the thresholds such that $t_q k$ and $t_u k$ are integers, and let $\Delta > 0$ be the branching factor. Given a query set X, with $|X| = w_q\,|U|$, and data set $Y \subseteq U$, with $|Y| = w_u\,|U|$, then running Algorithm 1 with $c_\ell = \left[\frac{\sqrt{t_q(1-t_q)}}{\sqrt{t_u(1-t_u)}}\right] \cdot \sqrt{6.5\ell\log(3k)}$ for $\ell < k$ and*

[15] The authors know of a way to reduce the error term further, so it only appears in the $\rho_q = 0$ case, and only as $\exp((\log 1/w_q)^{2/3}k^{1/3})$ which is $o(n)$ for any $w_q = \omega(1/n)$.

$c_k = \left[\begin{smallmatrix}0\\0\end{smallmatrix}\right]$, *gives that*

$$\begin{aligned}
\mathrm{E}\left[|R_k(X,t_q)|\right] &\le 2\Delta^k \exp(-k\,\mathrm{d}(t_q \,\|\, w_q)) \,. && (2)\\
\mathrm{E}\left[|R_k(Y,t_u)|\right] &\le 2\Delta^k \exp(-k\,\mathrm{d}(t_u \,\|\, w_u)) \,. && (3)\\
\Pr\left[|R_k(X,t_q) \cap R_k(Y,t_u)| \ge 1\right] &\ge 7^{-8}k^{-14}\Delta^k \exp(-k\,\mathrm{D}(T_1 \,\|\, P_1)) \qquad \textit{if } |X \cap Y| \ge w_1\,|U| \,. && (4)\\
\mathrm{E}\left[|R_k(X,t_q) \cap R_k(Y,t_u)|\right] &\le 2\Delta^k \exp(-k\,\mathrm{D}(T_2 \,\|\, P_2)) \qquad \textit{if } |X \cap Y| \le w_2\,|U| \,. && (5)
\end{aligned}$$

where $P_j = \left(\begin{smallmatrix} w_j & w_q - w_j \\ w_u - w_j & 1 - w_q - w_u + w_j \end{smallmatrix}\right)$, $T_j = \left(\begin{smallmatrix} t_j & t_q - t_j \\ t_u - t_j & 1 - t_q - t_u + t_j \end{smallmatrix}\right)$, $t_j = \arg\inf \mathrm{D}(T_j \,\|\, P_j)$ *for* $j \in \{1,2\}$.

Finally the expected running times, $\mathcal{T}_q$ *and* $\mathcal{T}_u$*, it takes to compute* $R_k(X,t_q)$ *and* $R_k(Y,t_u)$ *respectively are bounded by*

$$\begin{aligned}
\mathrm{E}\left[\mathcal{T}_q\right] &\le O\big(k\,|X| + k(k+\log(n))\Delta^k \exp(-k\,\mathrm{d}(t_q \,\|\, w_q)) \left(\tfrac{t_q(1-w_q)}{(1-t_q)w_q}\right)^{(c_k)_1}\big) \,.\\
\mathrm{E}\left[\mathcal{T}_u\right] &\le O\big(k\,|Y| + k(k+\log(n))\Delta^k \exp(-k\,\mathrm{d}(t_u \,\|\, w_u)) \left(\tfrac{t_u(1-w_u)}{(1-t_u)w_u}\right)^{(c_k)_2}\big) \,.
\end{aligned} \qquad (6)$$

We define $\Delta = \exp(\mathrm{D}(T_1 \,\|\, P_1))$, and let k be the smallest even integer at least $\frac{\log n}{\mathrm{D}(T_2 \,\|\, P_2) - \mathrm{d}(t_q \,\|\, w_q)}$. Define the sequence $\Delta_i = 2^{l_i}$ for some $l_i \in \mathbb{Z}_{\ge 0}$ such that $\prod_{j=1}^{i} \Delta_j \le \Delta^i < 2\prod_{j=1}^{i} \Delta_j$ for all $i \in [k]$.

We make 2 initiations of Algorithm 1, M_1, M_2, with height $k/2$. Δ and c_ℓ are adjusted correspondingly. In we have $c_{k/2} = c_k = \left[\begin{smallmatrix}0\\0\end{smallmatrix}\right]$.

For each instance we have

$$\begin{aligned}
\mathrm{E}\left[\left|R_{k/2}(X,t_q)\right|\right] &\le 2\exp(k/2\ \mathrm{D}(T_1 \,\|\, P_1) - k/2\ \mathrm{d}(t_q \,\|\, w_q))\\
&\le 2\exp\left(\left(\frac{\log n}{\mathrm{D}(T_2 \,\|\, P_2) - \mathrm{d}(t_q \,\|\, w_q)} + 2\right)\frac{(\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_q \,\|\, w_q))}{2}\right)\\
&= 2n^{\frac{1}{2}\frac{\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_q \,\|\, w_q)}{\mathrm{D}(T_2 \,\|\, P_2) - \mathrm{d}(t_q \,\|\, w_q)}}(\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_q \,\|\, w_q)).
\end{aligned}$$

similarly we get

$$\mathrm{E}\left[\left|R_{k/2}(X,t_q)\right|\right] \le 2n^{\frac{1}{2}\frac{\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_u \,\|\, w_u)}{\mathrm{D}(T_2 \,\|\, P_2) - \mathrm{d}(t_q \,\|\, w_q)}}(\mathrm{D}(T_1 \,\|\, P_1) - \mathrm{d}(t_u \,\|\, w_u)).$$

We combine the two data instances M_1 and M_2 by taking as representative sets returned the product of the sets returned by each of them. In particular, this means we successfully find a near set, if $\left|R_{k/2}(X,t_q) \cap R_{k/2}(Y,t_u)\right| \ge 1$ for both instances, which happens with probability at least

$$(7^{-8}k^{-14}\Delta^k \exp(-k\,\mathrm{D}(T_1 \,\|\, P_1)))^2 = (7^{-8}k^{-14})^2.$$

hence, repeating the algorithm Ck^{28} times, for some C, we can boost this probability to 99%.

Putting it all together now yields the full version of Theorem 1 contingent on Lemma 2.1.

□

2.2 Bounds on Branching

It now remains to prove Lemma 2.1. The inequalities (2), (3) and (5) are all simple calculations based on linearity of expectation. The time bound (6) is also fairly simple, but we have to take the decoding optimization described above into account. We also need to bound the number of paths alive at some point during the decoding process, which requires being more careful about the trimming conditions.

Finally the proof of the probability lower bound (4) is the main star of the section. We do this using essentially a second-moment method, but a number of tricks are needed in order to squeeze out acceptable bounds, taking into account that any of w_q, w_u, w_1, w_2 may be $o(1)$, which among other things forbid the use of many Central Limit Theorem type results.

Proof of (2) *and* (3). We only provide the proof for (2) since the proof (3) is analogous.

Let $r \in R_k$ be a representative string and define the random variables $\mathcal{X}^{(i)} = [r_i \in X]$ for $i \in [k]$, because the hash functions h_i used at each level of the tree are independent, so are the $(\mathcal{X}^{(i)})_{i\in[k]}$ independent.

We use linearity of expectation, and completely throw away the fact that some branches may have been cut early. Throwing away extra cuts of course only increases the probability of survival. Meanwhile, we do not expect to gain more than factors of k this way, compared to a sharp analysis, since the whole point of the algorithm is to efficiently approximate cuts done only at the leaf level.

$$\begin{aligned} \mathrm{E}\left[|R_k(X,t_q)|\right] &\le |R_k| \Pr\left[\forall \ell \le k : \sum_{i\in[\ell]} \mathcal{X}^{(i)} \ge t_q\ell - (c_\ell)_1\right] \\ &\le |R_k| \Pr\left[\sum_{i\in[k]} \mathcal{X}^{(i)} \ge t_q k\right] \\ &\le |R_k| \exp(-k\, \mathrm{d}(t_q \,\|\, w_q)). \end{aligned}$$

The final bound is the entropy Chernoff bound we use everywhere. Since $|R_k| = \prod_{i=1}^k \Delta_i \le 2\Delta^k$ we get the bound. □

Proof of (5). This is similar to the proof of (2) and (3), but two dimensional. Like in the those proofs we consider a single representative string $r \in R_k$ and define the random variables $\mathcal{X}^{(i)} = \left[\begin{smallmatrix} [r_i\in X] \\ [r_i\in Y]\end{smallmatrix}\right]$ for $i \in [k]$. By definition of Algorithm 1 $(\mathcal{X}^{(i)})_{i\in[k]}$ are independent.

We then bound using linearity of expectation:

$$\begin{aligned} \mathrm{E}\left[|R_k(X,t_q) \cap R_k(Y,t_u)|\right] &\le |R_k| \Pr\left[\forall \ell \le k : \sum_{i\in[\ell]} \mathcal{X}^{(i)} \ge \left[\begin{smallmatrix} t_q \\ t_u\end{smallmatrix}\right]\ell - c_\ell\right] \\ &\le 2\Delta^k \Pr\left[\sum_{i\in[k]} \mathcal{X}^{(i)} \ge \left[\begin{smallmatrix} t_q \\ t_u\end{smallmatrix}\right]k\right] \\ &\le 2\Delta^k \exp(-\,\mathrm{D}(T_2 \,\|\, P_2)) \end{aligned}$$

□

Proof of (6). As a preprocessing stage we make k sorted lists of $(a_i x)_{x\in X}$ where a_i is the coefficient in $h_i(p \circ x) = h'_i(p) + a_i x \mod q$, this takes $O(k\,|X|)$ time.

We will argue that at each level of tree that we only use $O(k + \log|X|) = O(k + \log n)$ amortized time per active path. More precisely, at level l we use $O((k + \log n)\,|R_\ell(X,t_q)|)$ amortized time.

Let $\ell \in [k]$ be fixed and consider an active path $r \in R_\ell(X,t_q)$. If $\sum_{i\in[\ell]}[r_i \in X] \ge t_q(l+1) - (c_{l-1})_1$ then every one of its children will be active. So we need to find $\{x \in U \mid h_\ell(p\circ x) < \Delta_\ell\} = h_\ell^{-1}([\Delta_\ell])$. Now $h_\ell(p\circ x) = h'_\ell(p) + ax \mod q$ where $a \ne 0 \mod q$ and $s = h'_\ell(p)$ can be computed in $O(k)$ time. We then get that $h_\ell^{-1}([\Delta_\ell]) = \left\{a^{-1}(i-s) \,\middle|\, i \in [\Delta_\ell]\right\}$, this we can find in time proportional with the number of active children, so charging the cost to them gives the result.

If $\sum_{i\in[\ell]}[r_i \in X] < t_q(l+1) - (c_{l-1})_1$ then only the children $r \circ x \in R_{l+1}$ where $x \in X$ will be active. So we need to find $\{x \in X \mid h_\ell(p \circ x) < \Delta_\ell\}$. Again using that $h_\ell(p \circ x) = h'_\ell(p) + ax \mod q$ where $a \neq 0 \mod q$ and $s = h'_\ell(p)$ can be computed in $O(k)$ time, we have reduced the problem to finding $h_\ell^{-1}([\Delta_\ell]) = \{x \in X \mid s + ax \mod q < \Delta_\ell\}$. This we note we can rewrite as $h_\ell^{-1}([\Delta_\ell]) = \{x \in X \mid s \le ax \vee ax < \Delta + s - q\}$, so using our sorted list this can be done in $O(\log n)$ time plus time proportional with the number of active children, so charging this cost to them gives the result.

We bound the expected number of active paths on a level $\ell \in [k]$. Let $r \in R_\ell$ be a representative string and define the random variables $\mathcal{X}^{(i)} = [r_i \in X]$ for $i \in [k]$, by definition of Algorithm 1 $(\mathcal{X}^{(i)})_{i\in[k]}$ are independent. We then bound

$$\begin{aligned}
\Pr\left[\sum_{i\in[l]} \mathcal{X}^{(i)} \ge t_q\ell - (c_\ell)_1\right] &\le \Pr\left[\forall j \le \ell : \sum_{i\in[j]} \mathcal{X}^{(i)} \ge t_q j - (c_j)_1\right] \\
&\le \Pr\left[\sum_{i\in[\ell]} \mathcal{X}^{(i)} \ge t_q\ell - (c_\ell)_1\right] \\
&\le \exp(-l \,\mathrm{d}(t_q - c_\ell/l \,\|\, w_q)) \\
&\le \exp(-l \,\mathrm{d}(t_q \,\|\, w_q)) \left(\tfrac{t_q(1-w_q)}{w_q(1-t_q)}\right)^{(c_\ell)_1} .
\end{aligned}$$

The crucial step here was using the identity

$$\mathrm{d}(t_q - \varepsilon \,\|\, w_q) = \mathrm{d}(t_q \,\|\, w_q) - \varepsilon \log \tfrac{t_q(1-w_q)}{w_q(1-t_q)} + \mathrm{d}(t_q - \varepsilon \,\|\, t_q)$$

from which we can ignore the $\mathrm{d}(t_q - \varepsilon \,\|\, t_q)$ term, since it is positive.

Using linearity of expectation we get that

$$\begin{aligned}
\mathrm{E}\left[|R_\ell(X, t_q)|\right] &\le |R_\ell| \exp(-\ell \,\mathrm{d}(t_q \,\|\, w_q)) \left(\tfrac{t_q(1-w_q)}{w_q(1-t_q)}\right)^{(c_\ell)_1} \\
&\le 2\Delta^\ell \exp(-\ell \,\mathrm{d}(t_q \,\|\, w_q)) \left(\tfrac{t_q(1-w_q)}{w_q(1-t_q)}\right)^{(c_\ell)_1} .
\end{aligned}$$

Now the expected cost of the tree becomes

$$\begin{aligned}
\mathrm{E}\left[\sum_{\ell\in[k]} O((k+\log(n))\,|R_\ell(X, t_q)|)\right] &= O((k+\log(n)) \sum_{\ell\in[k]} \mathrm{E}\left[|R_\ell(X, t_q)|\right]) \\
&\le O(k(k+\log(n))\Delta^k \exp(-k \,\mathrm{d}(t_q \,\|\, w_q)) \left(\tfrac{t_q(1-w_q)}{w_q(1-t_q)}\right)^{(c_\ell)_1}) .
\end{aligned}$$

□

Note that we throw away some leverage here by bounding the size of each level by the final level. We might have defined c_ℓ such that $\ell\Delta - \ell \,\mathrm{d}(t_q \,\|\, w_q) + c_\ell \log \frac{t_q(1-w_q)}{w_q(1-t_q)} - \ell \,\mathrm{d}(t_q - c_\ell/\ell \,\|\, t_q) = k\Delta - k \,\mathrm{d}(t_q \,\|\, w_q)$ and still used the same bound. The only later requirement we set the c_ℓ is that $\sum_{\ell\in[k]} \exp(-\ell \,\mathrm{d}(t_q - c_\ell/\ell \,\|\, t_q))$ sum to $1/\operatorname{poly}(k)$.

Making this change could potentially kill the $\left(\frac{t_q(1-w_q)}{w_q(1-t_q)}\right)^{(c_\ell)_1}$ factor, which is a bit of an eye sore. However in the near-constant query time case, which is really when this factor (or term once we using the tensoring trick) is relevant, this trick wouldn't work, since we then have exactly $\Delta = \mathrm{d}(t_q \,\|\, w_q)$.

For the final proof we need the following lemma, which bounds the probability that an unbiased Bernoulli $2d$ random walk stays entirely in the negative quadrant. A lemma like this is an exercise to show using the Central Limit Theorem and convergence to Brownian motion. However, our bound is non-asymptotic, making no assumptions about the relationship between the probability distribution of X_i and the size of n. There are non-asymptotic CLT bounds, like Berry Esseen, but unfortunately multivariate Berry Esseen bounds for random walks are not very developed.

Lemma 2.2 (The probability that a random walk stays in a quadrant)**.** *Let $X_1, \dots, X_k \in \{0,1\}^2$ be iid. Bernoulli 2d-random variables with probability matrix $\left[\begin{smallmatrix} p & p_1-p \\ p_2-p & 1-p_1-p_2+p \end{smallmatrix}\right]$. Assume that the coordinates are correlated, that is $p \geq p_1 p_2$, and assume $p_q k$ and $p_2 k$ are integers.*

Let $S_\ell = \sum_{i\in[\ell]} X_i$ be the associated random walk. Then

$$\Pr[\forall \ell \in [k] : S_\ell \leq 0] \geq \frac{1}{400\, k^{6.5}}.$$

The proof of this is in Section 2.3.

Proof of (4)*.* We will prove this bound using the second moment method. For this to work, it is critical that we restrict our representative strings further and consider

$$S = \left\{ r \in R_k \;\middle|\; \forall \ell \leq k : \left[\begin{smallmatrix} [r_i \in X] \\ [r_i \in Y] \end{smallmatrix}\right] \ell - c_\ell \leq \sum_{i\in[\ell]} \left[\begin{smallmatrix} [r_i \in X] \\ [r_i \in Y] \end{smallmatrix}\right] \leq [\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}]\ell \right\} ,$$

It is easy to check that $S \subseteq R_k(X, t_q) \cap R_k(Y, t_u)$, thus we have that

$$\Pr\left[|R_k(X,t_q) \cap R_k(Y,t_u)| \geq 1\right] \geq \Pr\left[|S| \geq 1\right] \geq \mathrm{E}\left[|S|\right]^2 / \mathrm{E}\left[|S|^2\right] ,$$

where the last bound is Paley-Zygmund's inequality. We then need to do two things: 1) Lower bound $\mathrm{E}\left[|S|\right]$, and 2) Upper bound $\mathrm{E}\left[|S|^2\right]$.

Lower bounding $\mathrm{E}\left[|S|\right]$. Let $r \in R_k$ be a representative string and define the random variables $\mathcal{X}^{(i)} = \left[\begin{smallmatrix} [r_i \in X] \\ [r_i \in Y] \end{smallmatrix}\right]$ for $i \in [k]$. Each one has distribution $P = \left[\begin{smallmatrix} w_1 & w_q - w_1 \\ w_u - w_q & 1-w_q-w_u+w_1 \end{smallmatrix}\right]$. We then introduce variables $\tilde{\mathcal{X}}^{(i)}$ with law $T = \left[\begin{smallmatrix} t_1 & t_q - t_1 \\ t_u - t_q & 1-t_q-t_u+t_1 \end{smallmatrix}\right]$, where t_1 minimizes $\mathrm{D}(T \,\|\, P)$ as defined in the algorithm.

We then use the following variation on Sanov's theorem:

Lemma 2.3. *For any set $A \subseteq \mathbb{R}^{2\times n}$ we have*

$$\Pr\left[(\mathcal{X}^{(i)})_{i\in[k]} \in A\right] = \exp(-k\,\mathrm{D}(T \,\|\, P)) \Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in A\right]$$

Proof. Define the logarithmic moment generating function $\Lambda(\lambda) = \log \mathrm{E}\left[\exp(\langle \lambda, \mathcal{X} \rangle)\right]$, and let $z = (\nabla_x \Lambda^*)(t)$. By a standard correspondence, (see e.g. [59] Chapter 14 or [32] Chapter 6.2), we have that

$$\mathrm{d}T(x) = \exp(\langle z, x\rangle - \Lambda(z))\mathrm{d}P(x)$$

for Radon–Nikodym derivates $\mathrm{d}T$ and $\mathrm{d}P$. Now using the exponential change of measure, we get that

$$\begin{aligned}\Pr\left[(\mathcal{X}^{(i)})_{i\in[k]} \in A\right] &= \int_{(x^{(i)})_{i\in[k]}\in A} \mathrm{d}P^{\otimes k}\\ &= \int_{(\tilde{x}^{(i)})_{i\in[k]}\in A} \exp\left(k\Lambda(z) - \left\langle z, \sum_{i\in[k]} \tilde{x}^{(i)}\right\rangle\right) \mathrm{d}T^{\otimes k}\\ &= \exp(-k\,\mathrm{D}(T\,\|\,P)) \int_{(\tilde{x}^{(i)})_{i\in[k]}\in A} \exp\left(-\left\langle z, \sum_{i\in[k]} \left(\tilde{x}^{(i)} - \left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]\right)\right\rangle\right) \mathrm{d}T^{\otimes k}\\ &= \exp(-k\,\mathrm{D}(T\,\|\,P)) \Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in A\right],\end{aligned}$$

where the last inequality follows from the fact that if $(\tilde{x}^{(i)})_{i\in[k]} \in A$ then $\sum_{i\in[k]} \tilde{x}^{(i)} = \left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]k$. □

For convenience we will sometimes write $T = \left[\begin{smallmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{smallmatrix}\right]$. Note that by assumption $t_q k = (t_{12} + t_{11})k$ and $t_u k = (t_{21} + t_{11})k$ are integers, but values such as $t_{11}k$ and $t_{22}k$ need not be. This will

We define the sets

$$U = \left\{ (x^{(i)})_{i\in[k]} \in \mathbb{R}^{2\times k} \,\middle|\, \forall \ell \le k : \sum_{i\in[\ell]} x^{(i)} \le \left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]\ell \right\}$$
$$\text{and} \quad L = \left\{ (x^{(i)})_{i\in[k]} \in \mathbb{R}^{2\times k} \,\middle|\, \forall \ell \le k : \sum_{i\in[\ell]} x^{(i)} \ge \left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]\ell - c_\ell \right\}$$

such that $U \cap L$ are all sequences satisfying our path requirement. In other words $E|S| = \exp(-k\,\mathrm{D}(T\,\|\,P)) \Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in U \cap L\right]$. Using a union bound we split up:

$$\Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in U \cap L\right] \ge \Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in U\right] - \Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in L\right].$$

The term is bounded by Lemma 2.2 from the Appendix. Once we notice that $w_1 \ge w_q w_u$ implies that $t_1 \ge t_q t_u$. One way to see this is that t_1 minimizing $\mathrm{D}(T\,\|\,P)$ gives rise to the equation $\frac{w_1(1-w_q-w_u+w_1)}{(w_q-w_1)(w_u-w_1)} = \frac{t_1(1-t_q-t_u+t_1)}{(t_q-t_1)(t_u-t_1)} = \frac{t_1+t_1(t_1-t_q-t_u)}{t_q t_u+t_1(t_1-t_q-t_u)}$. If $w_1 \ge w_q w_u$ the left hand side is ≥ 1, and so we must have $t_1 \ge t_q t_u$.

Lemma 2.2 then gives us

$$\Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in U\right] \ge \frac{1}{400}k^{-3.5}.$$

This is a pretty small value, so for the union bound to work we need an even smaller probability for the lower bound.

We bound each coordinate individually. The cases are symmetric, so we only consider the first coordinate. Using another union bound and Bernstein's inequality we get

$$\begin{aligned}\Pr\left[\exists \ell \leq k : \sum_{i=1}^{\ell} \tilde{\mathcal{X}}_1^{(i)} \leq t_q\ell - (c_\ell)_1\right] &= \sum_{l \leq k} \Pr\left[\sum_{i=1}^{\ell} \tilde{\mathcal{X}}_1^{(i)} \leq t_q\ell - (c_\ell)_1\right] \\ &\leq \sum_{l \leq k} \exp\left(\frac{-(c_\ell)_1^2/2}{(1-t_q)t_q\ell + (1-2t_q)(c_\ell)_1/3}\right) \\ &\leq \frac{1}{1200}k^{-6.5}.\end{aligned}$$

since $(c_\ell)_1 = \Omega(\sqrt{t_q(1-t_q)l\log l} + |1-2t_q|\log l)$.

Similarly, we upper bound $\Pr\left[\exists \ell \leq k : \sum_{i=1}^{\ell} \tilde{\mathcal{X}}_2^{(i)} \leq t_u\ell - (c_\ell)_2\right] \leq \frac{1}{1200}k^{-6.5}$. Putting it all together we get

$$\Pr\left[(\mathcal{X}^{(i)})_{i\in[k]} \in A\right] \geq \frac{1}{1200}\exp(-k\,\mathrm{D}(T_1 \,\|\, P_1))k^{-6.5} \ ,$$

so by linearity of expectation we get that

$$\mathrm{E}\left[S\right] \geq |R_k|\,\frac{1}{1200}k^{-6.5}\exp(-k\,\mathrm{D}(T \,\|\, P)) \geq \frac{1}{1200}k^{-6.5}\Delta^k\exp(-k\,\mathrm{D}(T \,\|\, P)) \ .$$

Upper bounding $\mathrm{E}\left[|S|^2\right]$

Consider two representative strings $r, r' \in R_k$ and let $q \in R_\ell$ be their common prefix, hence l is the length of their common prefix. Define the random variables $\mathcal{X}^{(i)} = \begin{bmatrix}[r_i\in X]\\ [r_i\in Y]\end{bmatrix}$, $\mathcal{Y}^{(j)} = \begin{bmatrix}[r'_j\in X]\\ [r'_j\in Y]\end{bmatrix}$, and $\mathcal{Z}^{(h)} = \begin{bmatrix}[q_h\in X]\\ [q_h\in Y]\end{bmatrix}$ for $i, j \in [k]\setminus[\ell]$ and $h \in [l]$. We then get that

$$\begin{aligned}\Pr\left[p, p' \in S\right] &\leq \Pr\left[\sum_{h\in[\ell]}\mathcal{Z}^{(h)} + \sum_{i\in[k]\setminus[l]}\mathcal{X}^{(i)} \geq tk \wedge \sum_{h\in[\ell]}\mathcal{Z}^{(h)} + \sum_{j\in[k]\setminus[l]}\mathcal{Y}^{(j)} \geq tk \wedge \sum_{h\in[\ell]}\mathcal{Z}^{(h)} \leq tl\right] \\ &\leq \Pr\left[\sum_{h\in[\ell]}\mathcal{Z}^{(h)} + \sum_{i\in[k]\setminus[l]}\mathcal{X}^{(i)} + \sum_{j\in[k]\setminus[l]}\mathcal{Y}^{(j)} \geq (2k-\ell)t\right] \ .\end{aligned}$$

Now $\sum_{h\in[\ell]}\mathcal{Z}^{(h)} + \sum_{i\in[k]\setminus[l]}\mathcal{X}^{(i)} + \sum_{j\in[k]\setminus[l]}\mathcal{Y}^{(j)}$ is almost a sum of independent random variable. We have that $\mathcal{X}^{(k-\ell+1)}$ and $\mathcal{Y}^{(k-\ell+1)}$ are correlated since they are chosen by sampling without replacement, but this implies that

$$\mathrm{E}\left[\exp(\langle\lambda, \mathcal{X}^{(k-\ell+1)} + \mathcal{Y}^{(k-\ell+1)}\rangle)\right] \leq \mathrm{E}\left[\exp(\langle\lambda, \mathcal{X}^{(k-\ell+1)}\rangle)\right]\mathrm{E}\left[\exp(\langle\lambda, \mathcal{Y}^{(k-\ell+1)}\rangle)\right]$$

We can then use a 2-dimensional Entropy-Chernoff bound and get that

$$\Pr\left[\sum_{h\in[\ell]}\mathcal{Z}^{(h)} + \sum_{i\in[k]\setminus[l]}\mathcal{X}^{(i)} + \sum_{j\in[k]\setminus[l]}\mathcal{Y}^{(j)} \geq (2k-\ell)t\right] \leq \exp(-(2k-\ell)\,\mathrm{D}(T \,\|\, P)) \ ,$$

Using this we can upper bound $\mathrm{E}\left[|S|^2\right] = \mathrm{E}\left[\sum_{r,r' \in R_k} [r, r' \in S]\right]$ by splitting the sum by the length of their common prefix.

$$
\begin{aligned}
\mathrm{E}\left[|S|^2\right] &= \mathrm{E}\left[\sum_{r,r' \in S_k} [r, r' \in S]\right] \\
&\leq \sum_{i=1}^{k} \left(\prod_{j=1}^{i} \Delta_j\right) \binom{\Delta_{i+1}}{2} \left(\prod_{j=i+2}^{k} \Delta_j\right) \exp(-(2k-i)\,\mathrm{D}(T \,\|\, P)) \\
&\leq \left(\prod_{j=1}^{k} \Delta_j\right)^2 \exp(-2k\,\mathrm{D}(T \,\|\, P)) \sum_{i=1}^{k} \exp(i\,\mathrm{D}(T \,\|\, P)) \left(\prod_{j=1}^{i} \Delta_j\right)^{-1} \\
&\leq \left(\prod_{j=1}^{k} \Delta_j\right)^2 \cdot \exp(-2k\,\mathrm{D}(T \,\|\, P)) \cdot k \cdot \exp(k\,\mathrm{D}(T \,\|\, P)) \cdot \Delta^{-k} \\
&\leq 4k\Delta^k \exp(-k\,\mathrm{D}(T \,\|\, P))
\end{aligned}
$$

Finishing the proof

Having lower bounded $\mathrm{E}\left[|S|\right]$ and upper bounded $\mathrm{E}\left[|S|^2\right]$ we can finish the proof.

$$
\begin{aligned}
\Pr\left[|R_k(X, t_q) \cap R_k(Y, t_u)| \geq 1\right] &\geq \Pr\left[|S| \geq 1\right] \\
&\geq \frac{\mathrm{E}\left[|S|\right]^2}{\mathrm{E}\left[|S|^2\right]} \\
&\geq \frac{\frac{1}{1200^2} k^{-13} \Delta^{2k} \exp(-2k\,\mathrm{D}(T \,\|\, P))}{4k\Delta^k \exp(-k\,\mathrm{D}(T \,\|\, P))} \\
&= \frac{1}{2400^2} k^{-14} \Delta^{\ell} \exp(-k\,\mathrm{D}(T \,\|\, P)) \,.
\end{aligned}
$$

□

2.3 Central Random Walks

The main goal of this section is to prove Lemma 2.2, which polynomially in k lower bounds the probability that a biased random walk on $\mathbb{Z}^2$ always stays below its means. Asymptotically, this can be done in various ways using the Central Limit Theorem for Brownian Motion, but as far as we know there are no standard ways to prove such a result in a quantitative way.

What we would really want is a Multidimensional Berry Esseen for Random Walks. Instead we prove something specifically for walks where each iid. step $X_1, \ldots, X_k \in \{0,1\}^2$ be is a Bernoulli $2d$-random variables with probability matrix $\left[\begin{smallmatrix} p & p_1 - p \\ p_2 - p & 1 - p_1 - p_2 + p \end{smallmatrix}\right]$. We need the further restrictions that the coordinates are correlated ($p \geq p_1 p_2$), and that $p_1 k$ and $p_2 k$ are integers.

We will start by proving some partial results, simply bounding the probability that the final position of the random walk hits a specific value. We then prove the lemma conditioned on hitting those values, and finally put it all together.

Lemma 2.4. *Let $k \in \mathbb{Z}_+$ and $p_1, p_2 \in [0,1]$, such that, both $p_1 k$ and $p_2 k$ are integers. Choose $p \in [0,1]$, such that, $p \geq p_1 p_2$. Let $X^{(i)} \in \mathbb{R}^2$ be independent identically distributed 2-dimensional Bernoulli variables, where their probability matrix is $P = \left[\begin{smallmatrix} p & p_1 - p \\ p_2 - p & 1-p_1-p_2+p \end{smallmatrix}\right]$. We then get that*

$$\Pr\left[\sum_{i\in[k]} X^{(i)} = \left(\begin{smallmatrix} p_1 \\ p_2 \end{smallmatrix}\right) k \wedge \sum_{i\in[k]} X_1^{(i)} X_2^{(i)} = \lceil pk \rceil\right] \geq \frac{1}{400} k^{-3.5} \ .$$

In the proof we will be using the Stirling's approximation

$$\sqrt{2\pi n} n^n e^{-n} \leq n! \leq e\sqrt{n} n^n e^{-n} \ .$$

This implies the following useful bounds on the binomial and multinomial coefficients.

$$\begin{aligned}\binom{n}{an} &\geq \frac{\sqrt{2\pi}}{e^2} \cdot \frac{\sqrt{n}}{\sqrt{an(1-a)n}} \frac{n^n}{(an)^{an}((1-a)n)^{(1-a)n}} \\ &\geq \frac{\sqrt{2\pi}}{e^2} n^{-0.5} a^{-an} (1-a)^{-(1-a)n} \ .\end{aligned} \tag{7}$$

$$\begin{aligned}\binom{n}{an, bn, cn} &\geq \frac{\sqrt{2\pi}}{e^4} \cdot \frac{\sqrt{n}}{\sqrt{anbncn(1-a-b-c)n}} \frac{n^n}{(an)^{an}(bn)^{bn}(cn)^{cn}((1-a-b-c)n)^{(1-a-b-c)n}} \\ &\geq \frac{\sqrt{2\pi}}{e^4} n^{-1.5} a^{-an} b^{-bn} c^{-cn} (1-a-b-c)^{-(1-a-b-c)n} \ .\end{aligned} \tag{8}$$

Proof. If $p_1 = 1$ then $p = p_2$ and we get that

$$\Pr\left[\sum_{i\in[k]} X_2^{(i)} = p_2 k\right] = \binom{k}{p_2 k} p_2^{p_2 k} (1-p_2)^{(1-p_2)k} \geq \frac{\sqrt{2\pi}}{e^2} k^{-\frac{1}{2}} \ ,$$

where we have used eq. (7). We get the same bound when $p_1 = 0$, $p_2 = 1$, or $p_2 = 0$.

Now assume that $p_1, p_2 \notin \{0, 1\}$, we then have that $\frac{1}{k} \leq p_1 \leq 1 - \frac{1}{k}$ and $\frac{1}{k} \leq p_2 \leq 1 - \frac{1}{k}$. We first note that

$$\begin{aligned}&\Pr\left[\sum_{i\in[k]} X^{(i)} = (p_1 k, p_2 k) \wedge \sum_{i\in[k]} X_1^{(i)} X_2^{(i)} = \lceil pk \rceil\right] \\ &= \binom{k}{\lceil pk \rceil, p_1 k - \lceil pk \rceil, p_2 k - \lceil pk \rceil} p^{\lceil pk \rceil} (p_1 - p)^{p_1 k - \lceil pk \rceil} (p_2 - p)^{p_2 k - \lceil pk \rceil} (1 - p_1 - p_2 + p)^{k - p_1 k - p_2 k + \lceil pk \rceil} \\ &\geq \frac{\sqrt{2\pi}}{e^4} \cdot \frac{1}{k^{3/2}} \exp\Big(- \lceil pk \rceil \log \frac{\lceil pk \rceil}{pk} - (p_1 k - \lceil pk \rceil) \log \frac{p_1 k - \lceil pk \rceil}{p_1 k - pk} \\ &\qquad - (p_2 k - \lceil pk \rceil) \log \frac{p_2 k - \lceil pk \rceil}{p_2 k - pk} - (k - p_1 k - p_2 k + \lceil pk \rceil) \log \frac{k - p_1 k - p_2 k + \lceil pk \rceil}{k - p_1 k - p_2 k + pk} \Big)\end{aligned}$$

where we have used eq. (8). We will bound each of the terms $\lceil pk \rceil \log \frac{\lceil pk \rceil}{pk}$, $(p_1 k - \lceil pk \rceil) \log \frac{p_1 k - \lceil pk \rceil}{p_1 k - pk}$, $(p_2 k - \lceil pk \rceil) \log \frac{p_2 k - \lceil pk \rceil}{p_2 k - pk}$, and $(k - p_1 k - p_1 k + \lceil pk \rceil) \log \frac{k - p_1 k - p_2 k + \lceil pk \rceil}{k - p_1 k - p_2 k + pk}$ individually.

Using that $p \geq p_1 p_2 \geq \frac{1}{k^2}$ we get that

$$\lceil pk \rceil \log \frac{\lceil pk \rceil}{pk} = (1 + pk) \log\left(1 + \frac{1}{pk}\right) \leq 1 + \log(1 + k) \ .$$

Now using that $1 - p_1 - p_2 + p \geq (1-p_1)(1-p_2) \geq \frac{1}{k^2}$ we get that

$$\begin{aligned}(k - p_1k - p_1k + \lceil pk \rceil) \log \frac{k - p_1k - p_2k + \lceil pk \rceil}{k - p_1k - p_2k + pk} &\leq (k(1 - p_1 - p_2 + p) + 1) \log \left(1 + \frac{1}{k(1 - p_1 - p_2 + p)}\right) \\ &\leq 1 + \log(1 + k) \ .\end{aligned}$$

We easily get that

$$(p_1k - \lceil pk \rceil) \log \frac{p_1k - \lceil pk \rceil}{p_1k - pk} \leq (p_1k - \lceil pk \rceil) \log \frac{p_1k - pk}{p_1k - pk} = 0 \ .$$

Similarly, we get that $(p_2k - \lceil pk \rceil) \log \frac{p_2k - \lceil pk \rceil}{p_2k - pk} = 0$.

Combining all this we get that

$$\begin{aligned}\Pr\left[\sum_{i\in[k]} X^{(i)} = (p_1k, p_2k) \wedge \sum_{i\in[k]} X_1^{(i)} X_2^{(i)} = \lceil pk \rceil\right] &\geq \frac{\sqrt{2\pi}}{e^4} k^{-1.5} \exp(-(1 + \log(1+k)) - (1 + \log(1+k))) \\ &\geq \frac{1}{400} k^{-3.5} \ .\end{aligned}$$

□

We now prove a result for the random walk, conditioned on the final position. In the last result of this section, we will remove those restrictions.

Lemma 2.5. *Let $k \in \mathbb{Z}_+$ and $p, p_1, p_2 \in [0,1]$, such that, pk, p_1k, and p_2k are integers and $p \geq p_1p_2$. Let $X^{(i)} \in \{0,1\}^2$ be independent identically distributed variables. We then get that*

$$\Pr\left[\forall l \leq k : \sum_{i\in[k]} X^{(i)} \geq \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} l \,\middle|\, \sum_{i\in[k]} X^{(i)} = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} k \wedge \sum_{i\in[k]} X_1^{(i)} X_2^{(i)} = pk \right] \geq k^{-3} \ .$$

In the proof we will use the folklore result.

Lemma 2.6. *Let $k \in Z_+$ and $(a_i)_{i\in[k]}$ numbers such that $\sum_{i\in[k]} a_i \geq 0$ then there exists a $s \in [k]$ such that $\sum_{i\in[l]} a_{(s+i) \mod k} \geq 0$ for every $l \leq k$.*

Proof of Lemma 2.5. Using Lemma 2.6 we get that $\sum_{i\in[l]} X_1^{(i)} \geq p_1 l$ for every $l \leq k$ with probability at least k^{-1} since every variable identically distributed. Fixing $(X_1^{(i)})_{i\in[k]}$ and using Lemma 2.6 2 times we get that $\sum_{i\in[l]} X_1^{(i)} X_2^{(i)} \geq \frac{p}{p_1} \sum_{i\in[l]} X_1^{(i)}$ and $\sum_{i\in[l]} (1 - X_1^{(i)}) X^{(i)} \geq \frac{p_2 - p}{1 - p_1} \sum_{i\in[l]} X_1^{(i)}$ for every $l \leq k$ with probability at least k^{-2}. If all these three events happens then for every $l \leq k$ we get that

$$\begin{aligned}\sum_{i\in[l]} X_2^{(i)} &= \sum_{i\in[l]} X_1^{(i)} X_2^{(i)} + \sum_{i\in[l]} (1 - X_1^{(i)}) X_2^{(i)} \\ &\geq \frac{p}{p_1} \sum_{i\in[l]} X_1^{(i)} + \frac{p_2 - p}{1 - p_1} \sum_{i\in[l]} X_1^{(i)} \\ &= \frac{p - p_1p_2}{p_1(1 - p_1)} \sum_{i\in[l]} X_1^{(i)} + \frac{p_2 - p}{1 - p_1} l \\ &\geq \frac{p - p_1p_2}{p_1(1 - p_1)} p_1 l + \frac{p_2 - p}{1 - p_1} l \\ &= p_2 l \ .\end{aligned}$$

So we conclude that with probability at least k^{-3} then $\sum_{i\in[l]} X^{(i)} \geq \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} l$ for every $l \leq k$ which finishes the proof. □

All that remains is proving Lemma 2.2. We restate it and then prove it.

Lemma 2.2. *Let $X_1, \ldots, X_k \in \{0,1\}^2$ be iid. Bernoulli 2d-random variables with probability matrix $\left[\begin{smallmatrix} p & p_1-p \\ p_2-p & 1-p_1-p_2+p \end{smallmatrix}\right]$. Assume that the coordinates are correlated, that is $p \geq p_1 p_2$, and assume $p_q k$ and $p_2 k$ are integers.*

Let $S_\ell = \sum_{i\in[\ell]} X_i$ be the associated random walk. Then

$$\Pr[\forall \ell \in [k] : S_\ell \leq 0] \geq \frac{1}{400\, k^{6.5}}.$$

Proof. We define the set $U = \left\{ (x^{(i)})_{i\in[k]} \in \mathbb{R}^{2\times k} \,\middle|\, \forall \ell \leq k : \sum_{i\in[\ell]} x^{(i)} \leq \left[\begin{smallmatrix} p_1 \\ p_2 \end{smallmatrix}\right] \ell \right\}$ of all sequences satisfying our path requirement. In other words $\Pr[\forall k \in [n] : S_k \leq 0] = \Pr\left[(\mathcal{X}^{(i)})_{i\in[k]} \in U\right]$. We then add even more restrictions by defining

$$A' = \left\{ (x^{(i)})_{i\in[k]} \in \mathbb{R}^{2\times k} \,\middle|\, \sum_{i\in[k]} x^{(i)} = \left[\begin{smallmatrix} p_1 \\ p_2 \end{smallmatrix}\right] k \wedge \sum_{i\in[k]} (1 - x_1^{(i)})(1 - x_2^{(i)}) = \lceil p_{22} k \rceil \right\}.$$

That is, we require the last final value of the path to completely match its expectation, rounded up. By monotonicity we have $\Pr\left[(\mathcal{X}^{(i)})_{i\in[k]} \in U\right] \geq \Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in U \cap A'\right]$.

We want to use Lemma 2.4 and Lemma 2.5 and to ease the notation we introduce the negated random variables $\mathcal{Y}^{(i)} = 1 - \tilde{\mathcal{X}}^{(i)}$. Define $p_{22} = 1 - p_1 - p_2 + p$. We then have that $\mathrm{E}\left[\mathcal{Y}^{(i)}\right] = \left[\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}\right]$ and $\Pr\left[\mathcal{Y}^{(i)} = \left(\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right)\right] = p_{22} = 1 - p_1 - p_2 + p \geq (1-p_1)(1-p_2)$ by the assumption of correlation.

We can then rewrite using $\mathcal{Y}^{(i)}$:

$$\Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in U \cap A'\right] = \Pr\left[\forall \ell \leq k : \sum_{i\in[k]} \mathcal{Y}^{(i)} \geq \left[\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}\right] \ell \wedge \sum_{i\in[k]} \mathcal{Y}^{(i)} = \left[\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}\right] k \wedge \sum_{i\in[k]} \mathcal{Y}_1^{(i)} \mathcal{Y}_2^{(i)} = \lceil p_{22} k \rceil \right]$$

Now using Lemma 2.4 we have that

$$\Pr\left[\sum_{i\in[k]} \mathcal{Y}^{(i)} = \left[\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}\right] k \wedge \sum_{i\in[k]} \mathcal{Y}_1^{(i)} \mathcal{Y}_2^{(i)} = \lceil p_{22} k \rceil \right] \geq \frac{1}{400} k^{-3.5}\ .$$

Combining this with Lemma 2.5 we get that

$$\begin{aligned}
&\Pr\left[\forall \ell \leq k : \sum_{i\in[k]} \mathcal{Y}^{(i)} \geq \left[\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}\right] \ell \wedge \sum_{i\in[k]} \mathcal{Y}^{(i)} = \left[\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}\right] k \wedge \sum_{i\in[k]} \mathcal{Y}_1^{(i)} \mathcal{Y}_2^{(i)} = \lceil p_{22} k \rceil \right] \\
&\quad = \Pr\left[\sum_{i\in[k]} \mathcal{Y}^{(i)} = \left[\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}\right] k \wedge \sum_{i\in[k]} \mathcal{Y}_1^{(i)} \mathcal{Y}_2^{(i)} = \lceil p_{22} k \rceil \right] \\
&\qquad \cdot \Pr\left[\forall \ell \leq k : \sum_{i\in[k]} \mathcal{Y}^{(i)} \geq \left[\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}\right] \ell \,\middle|\, \sum_{i\in[k]} \mathcal{Y}^{(i)} = \left[\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}\right] k \wedge \sum_{i\in[k]} \mathcal{Y}_1^{(i)} \mathcal{Y}_2^{(i)} = \lceil p_{22} k \rceil \right] \\
&\quad \geq \frac{1}{400} k^{-6.5}\ .
\end{aligned}$$

□

3 Lower Bounds

Our lower bounds all assume that $w_2 d = \omega(\log n)$, where d is the size of the universe. As discussed in the introduction is both standard and necessary.

We proceed to define the hard distributions for all further lower bounds.

1. A query $x \in \{0,1\}^d$ is created by sampling d random independent bits with Bernoulli(w_q) distribution.

2. A dataset $P \subseteq \{0,1\}^d$ is constructed by sampling $n-1$ vectors with random independent bits from such that $y_i \sim$ Bernoulli(w_2/w_q) if $x_i = 1$ and $y_i \sim$ Bernoulli($(w_u - w_2)/(1-w_q)$) otherwise, for all $y \in P$.

3. A 'close point', $y' \in \{0,1\}^d$, is created by $y'_i \sim$ Bernoulli(w_1/w_q) if $x_i = 1$ and $y'_i \sim$ Bernoulli($(w_u - w_1)/(1-w_q)$) otherwise. This point is also added to P.

The values are chosen such that $\mathrm{E}\left[|x|\right] = w_q d$, $\mathrm{E}\left[|z|\right] = w_u d$ for all $z \in P$, $\mathrm{E}\left[|x \cap y'|\right] = w_1 d$, and $\mathrm{E}\left[|x \cap y|\right] = w_2 d$ for all $y \in P \setminus \{y'\}$. By a union bound over P, the actual values are within factors $1+o(1)$ of their expectations with high probability. Changing at most $o(\log n)$ coordinates we ensure the weights of queries/database points is exactly their expected value, while only changing the inner products by factors $1+o(1)$. Since the changes do not contain any new information, we can assume for lower bounds that entries are independent. Thus any $(w_q, w_u, w_1(1-o(1)), w_2(1+o(1)))$-GapSS data structure on P must thus be able to return y' with at least constant probability when given the query x.

Model Our lower bounds are shown in slightly different models. The first lower bound follows the framework of O'Donnell et al. [54] and Christiani [27] and directly lower bound the quantity $\frac{\log(p_1/\min\{p_u,p_q\})}{\log(p_2/\min\{p_u,p_q\})}$ which lower bounds ρ_u and ρ_q in Definition 4. This lower bound holds for all $w_2 \neq w_q w_u$, i.e., it gives a lower bound when we are not considering a random instance, and it only gives a lower bound in the case where $\rho_q = \rho_u$.

For the second lower bound we follow the framework of Andoni et al. [10] and give a lower bound in the "list-of-points"-model (see Definition 2). This is a slightly more general model, though it is believed that all bounds for the first model can be shown in the list-of-points model as well. Our lower bound shows that our upper bound is tight in the full time/space trade-off when $w_2 = w_q w_u$, i.e., when we are given a random instance.

The second bound can be extended to show cell probe lower bounds by the arguments in [57].

3.1 p-biased Analysis

We first give some preliminaries on b-biased Boolean analysis, and then introduce the directed noise operator.

3.1.1 Preliminaries

We want analyse Boolean functions $f : \{0,1\}^d \to \{0,1\}$ but as is common, it turns out to be beneficial to consider a more general class of functions $f : \{0,1\}^d \to \mathbb{R}$.

The probability distribution π_p is defined on $\{0,1\}$ by $\pi_p(1) = p$ and $\pi_p(0) = 1-p$, and we define $\pi_p^{\otimes d}$ to be the product probability distribution on $\{0,1\}^d$. We write $L_2(\{0,1\}^d, \pi_p^{\otimes d})$ for the inner product space of functions $f : \{0,1\}^d \to \mathbb{R}$ with inner product

$$\langle f, g \rangle_p = \mathop{\mathrm{E}}_{x \sim \pi_p^{\otimes d}} [f(x)g(x)] \ .$$

We will define the norm $\|f\|_{L_q(p)} = \left(\mathrm{E}_{x\sim\pi_p^{\otimes d}}[f(x)^q]\right)^{1/q}$.

We define the p-biased Fourier coefficients for a function $f : L_2(\{0,1\}^d, \pi_p^{\otimes d})$ by

$$\hat{f}^{(p)}(S) = \mathop{\mathrm{E}}_{x\sim\pi_p^{\otimes d}}\left[f(x)\phi_S^{(p)}(x)\right] \ ,$$

for every $S \subseteq [d]$ and where we define

$$\phi^{(p)}(x) = \frac{x-p}{\sqrt{p(1-p)}} \qquad\qquad \phi_S^{(p)}(x) = \prod_{i\in S}\phi^{(p)}(x_i) \ .$$

The Fourier coefficients have the nice property that

$$f(x) = \sum_{S\subseteq[d]} \hat{f}^{(p)}(S)\phi_S^{(p)}(x) \ .$$

The Fourier coefficients satisfy the Parseval-Plancherel identity, which says that for any $f, g \in L_2(\{0,1\}^d, \pi_p^d)$ we have that

$$\langle f, g \rangle_p = \sum_{S\subseteq[d]} \hat{f}^{(p)}(S)\hat{g}^{(p)}(S) \ .$$

In particular we have that $\mathrm{E}_{x\sim\pi_p^{\otimes d}}\left[f(x)^2\right] = \|f\|_{L_2(p)}^2 = \sum_{S\subseteq[d]}\hat{f}^{(p)}(S)^2$. For Boolean functions $f : \{0,1\}^d \to \{0,1\}$ this is particularly useful since we get that

$$\begin{aligned}
\mathop{\Pr}_{x\sim\pi_p^{\otimes d}}[f(x)=1] &= \mathop{\mathrm{E}}_{x\sim\pi_p^{\otimes d}}[f(x)] = \mathop{\mathrm{E}}_{x\sim\pi_p^{\otimes d}}\left[\sum_{S\subseteq[n]}\hat{f}^{(p)}(S)\phi_S^{(p)}(x)\right] = \hat{f}^{(p)}(\emptyset) \\
\mathop{\Pr}_{x\sim\pi_p^{\otimes d}}[f(x)=1] &= \mathop{\mathrm{E}}_{x\sim\pi_p^{\otimes d}}[f(x)] = \mathop{\mathrm{E}}_{x\sim\pi_p^{\otimes d}}\left[f(x)^2\right] = \sum_{S\subseteq[d]}\hat{f}^{(p)}(S)^2.
\end{aligned}$$

If we think of f as a filter in a Locality Sensitive data structure, $\Pr_{x\sim\pi_p^{\otimes d}}[f(x)=1]$ is the probability that the filter accepts a random point with expected weight p ($d \cdot p$ of the coordinates being 1).

3.1.2 Noise

For $\rho \in [-1,1]$, $p_1, p_2 \in (0,1)$, and $x \in \{0,1\}^d$ we write $y \sim N_\rho^{p_1\to p_2}(x)$ when $y \in \{0,1\}^d$ is randomly chosen such that for each $i \in [d]$ independently, we have that if $x_i \sim \pi_{p_2}$ then $y_i \sim \pi_{p_1}$ and (x_i, y_i) are ρ-correlated. We note that if $x \sim \pi_{p_2}^{\otimes d}$ and $y \sim N_\rho^{p_1\to p_2}$ then we also have that $y \sim \pi_{p_1}^{\otimes d}$ and $x \sim N_\rho^{p_2\to p_1}(y)$.

For $\rho \in [-1,1]$ and $p_1, p_2 \in (0,1)$ we define the *directed noise operator* $T_\rho^{p_1 \to p_2} : L_2(\{0,1\}^d, \pi_{p_1}^{\otimes d}) \to L_2(\{0,1\}^d, \pi_{p_2}^{\otimes d})$ by

$$T_\rho^{p_1 \to p_2} f(x) = \mathop{\mathrm{E}}_{y \sim N_\rho^{p_1 \to p_2}} [f(y)] \ .$$

When $p_1 = p_2 = p$ then $T_\rho^{p \to p}$ is the *usual noise operator* on p-biased spaces and we denote it $T_\rho^{(p)}$. $T_\rho^{(p)}$ has the nice property that $\widehat{T_\rho^{(p)} f}^{(p)}(S) = \rho^{|S|} \hat{f}^{(p)}(S)$ for any $S \subseteq [d]$, and hence $T_\rho^{(p)}$ satisfies the semigroup property $T_\rho^{(p)} T_\sigma^{(p)} = T_{\rho\sigma}^{(p)}$. The following lemma shows that we have similar properties for $T_\rho^{p_1 \to p_2}$.

Lemma 3.1. *For $\rho \in [-1,1]$, $p_1, p_2 \in (0,1)$ and $f \in L_2(\{0,1\}^d, \pi_{p_1}^{\otimes})$ we have that*

$$\widehat{T_\rho^{p_1 \to p_2} f}^{(p_2)}(S) = \rho^{|S|} \hat{f}^{(p_1)}(S) \ ,$$

for any $S \subseteq [d]$. Furthermore, for any $\sigma \in [-1,1]$ and $p_3 \in [0,1]$ we have that $T_\sigma^{p_2 \to p_3} T_\rho^{p_1 \to p_2} = T_{\rho\sigma}^{p_1 \to p_3}$ and $T_\rho^{p_2 \to p_1}$ is the adjoint of $T_\rho^{p_1 \to p_2}$.

Proof. We fix $S \subseteq [d]$ and get that

$$\begin{aligned}
\widehat{T_\rho^{p_1 \to p_2} f}^{(p_2)}(S) &= \mathop{\mathrm{E}}_{x \sim \pi_{p_2}^{\otimes d}} \left[T_\rho^{p_1 \to p_2} f(x) \phi_S^{(p_2)}(x) \right] \\
&= \mathop{\mathrm{E}}_{x \sim \pi_{p_2}^{\otimes d}} \left[\mathop{\mathrm{E}}_{y \sim N_\rho^{p_1 \to p_2}(x)} [f(y)] \, \phi_S^{(p_2))}(x) \right] \\
&= \mathop{\mathrm{E}}_{x \sim \pi_{p_2}^{\otimes d}} \left[\mathop{\mathrm{E}}_{y \sim N_\rho^{p_1 \to p_2}(x)} \left[\sum_{T \subseteq [d]} \widehat{f}^{(p_1)}(T) \phi_T^{(p_1)}(y) \right] \phi_S^{(p_2)}(x) \right] \\
&= \mathop{\mathrm{E}}_{x \sim \pi_{p_2}^{\otimes d}} \left[\mathop{\mathrm{E}}_{y \sim N_\rho^{p_1 \to p_2}(x)} \left[\widehat{f}^{(p_1)}(S) \phi_S^{(p_1)}(y) \phi_S^{(p_2)}(x) \right] \right] \\
&= \widehat{f}^{(p_1)}(S) \prod_{i \in S} \mathop{\mathrm{E}}_{x_i \sim \pi_{p_2}} \left[\mathop{\mathrm{E}}_{y_i \sim N_\rho^{p_1 \to p_2}} \left[\phi_i^{(p_1)}(y_i) \phi_i^{(p_2)}(x_i) \right] \right] \\
&= \rho^{|S|} \widehat{f}^{(p_1)}(S) \ ,
\end{aligned}$$

where the last line uses that $\phi_i^{(p)}(x) = \frac{x-p}{\sqrt{p(1-p)}}$, which proves the first claim. For the second claim we note that

$$(\widehat{T_\sigma^{p_2 \to p_3} T_\rho^{p_1 \to p_2} f})^{(p_3)}(S) = \sigma^{|S|} \widehat{T_\rho^{p_1 \to p_2} f}^{(p_2)}(S) = (\rho\sigma)^{|S|} \widehat{f}^{(p_1)}(S) = \widehat{T_{\rho\sigma}^{p_1 \to p_2} f}^{(p_3)}(S) \ ,$$

for any $f \in f \in L_2(\{0,1\}^d, \pi_{p_1}^{\otimes})$ and any $S \subseteq [d]$ which proves the second claim. For the last claim we use the Plancherel-Parseval identity and get that

$$\langle T_\rho^{p_1 \to p_2} f, g \rangle_{L_2(p_2)} = \sum_{S \in [d]} \rho^{|S|} \widehat{f}^{(p_1)} \widehat{g}^{(p_2)} = \langle f, T_\rho^{p_2 \to p_1} g \rangle_{L_2(p_1)} \ ,$$

for any $f \in L_2(\{0,1\}^d, \pi_{p_1}^{\otimes d})$ and any $g \in L_2(\{0,1\}^d, \pi_{p_2}^{\otimes d})$ which shows that $T_\rho^{p_2 \to p_1}$ is the adjoint of $T_\rho^{p_1 \to p_2}$. □

We say that $(T_\rho^{p_1 \to p_2})_{\rho > 0}$ is (s, r)-hypercontractive if there exists $\rho_0 > 0$ such that for every $\rho \geq \rho_0$ and every $f \in L_r(\{0,1\}^d, \pi_{p_1}^d)$

$$\left\| T_\rho^{p_1 \to p_2} f \right\|_{L_s(p_2)} \leq \|f\|_{L_r(p_1)} \ .$$

We define $\sigma_{s,r}(p_1, p_2)$ to be the smallest possible ρ_0 We are interested in the hypercontractivity of $T^{p_1 \to p_2}$

3.2 Symmetric Lower bound

The most general, but sadly least tractable, approach to our lower bounds, is to bound the noise operator T_α in terms of a different level of noise, T_β. We do however manage to show one bound on this type, following an spectral approach first used by O'Donnell et al. [54] to prove the first optimal LSH lower bounds of $\rho \geq 1/c$ for data-independent hashing. Besides handling the case of set similarity with filters rather than hash functions, we slightly generalize the approach a big by using the power-means inequality rather than log-concavity. [16]

We will show the following inequality

$$\left(\frac{\Pr_{x,y',f}[f(x) = 1, f(y') = 1]}{\Pr_{x,f}[f(x) = 1]} \right)^{1/\log\alpha} \leq \left(\log \frac{\Pr_{x,y,f}[f(x) = 1, f(y) = 1]}{\Pr_{x,f}[f(x) = 1]} \right)^{1/\log\beta}$$

where $\alpha = \frac{w_1 - w^2}{w(1-w)}$ and $\beta = \frac{w_2 - w^2}{w(1-w)}$, and y' and y are sampled as respectively a close and a far point (see the top of the section). By rearrangement, this directly implies a lower bound in the LSF model as defined in Definition 4.

First we prove a general lemma about Boolean functions, which contains the most important arguments.

Lemma 3.2. *Let $f : \{0,1\}^n \to \mathbb{R}$ be a function and $p \in (0,1)$. Then for any $1 > \alpha \geq \beta > 0$ we have that*

$$\left(\frac{\langle T_\alpha^{(p)} f, f \rangle_{L_2(p)}}{\|f\|_{L_2(p)}^2} \right)^{1/\log(1/\alpha)} \leq \left(\frac{\langle T_\beta^{(p)} f, f \rangle_{L_2(p)}}{\|f\|_{L_2(p)}^2} \right)^{1/\log(1/\beta)} .$$

[16] This widens the range in which the bound is applicable – the O'Donnell bound is only asymptotic for $r \to 0$. However the values we obtain outside this range, when applied to Hamming space LSH, aren't sharp against the upper bounds.

Proof. We use the Parseval-Plancherel identity and the power-mean inequality to get that

$$\begin{aligned}
\left(\frac{\langle T_\alpha^{(p)} f, f\rangle_{L_2(p)}}{\|f\|_{L_2(p)}^2}\right)^{1/\log(1/\alpha)} &= \left(\frac{\sum_{S\subseteq[n]} \alpha^{|S|} \hat{f}^{(p)}(S)^2}{\sum_{S\subseteq[n]} \hat{f}^{(p)}(S)^2}\right)^{1/\log(1/\alpha)} \\
&= \left(\sum_{k=0}^{n} \frac{\sum_{\substack{S\subseteq[n]\\|S|=k}} \hat{f}^{(p)}(S)^2}{\sum_{S\subseteq[n]} \hat{f}^{(p)}(S)^2} \left(e^{-k}\right)^{\log(1/\alpha)}\right)^{1/\log(1/\alpha)} \\
&\le \left(\sum_{k=0}^{n} \frac{\sum_{\substack{S\subseteq[n]\\|S|=k}} \hat{f}^{(p)}(S)^2}{\sum_{S\subseteq[n]} \hat{f}^{(p)}(S)^2} \left(e^{-k}\right)^{\log(1/\beta)}\right)^{1/\log(1/\beta)} \\
&= \left(\frac{\sum_{S\subseteq[n]} \beta^{|S|} \hat{f}^{(p)}(S)^2}{\sum_{S\subseteq[n]} \hat{f}^{(p)}(S)^2}\right)^{1/\log(1/\beta)} \\
&= \left(\frac{\langle T_\beta^{(p)} f, f\rangle_{L_2(p)}}{\|f\|_{L_2(p)}^2}\right)^{1/\log(1/\beta)} .
\end{aligned}$$

The first and the last equality follows from the Parseval-Plancherel identity and the inequality follows from the power-mean inequality since $\log(1/\alpha) \le \log(1/\beta)$. □

The proof of Theorem 2 is then simply a few a rearrangements such that we can use Lemma 3.2.

Corollary 1. *Any data-independent LSF data structure for the (w, w, w_1, w_2)-GapSS problem with expected query time n^{ρ_q} and expected space usage $n^{1+\rho_u}$ where $\rho_q = \rho_u = \rho$ must have*

$$\rho \ge \log\left(\frac{w_1 - w^2}{w(1-w)}\right) \Big/ \log\left(\frac{w_2 - w^2}{w(1-w)}\right) .$$

Proof. Let $\mathcal{F}$ be any fixed LSF-family and let $f : \{0,1\}^n \to \{0,1\}$ be a random function such that $f^{-1}(1) = Q$ for $Q \sim \mathcal{F}$. Now we define the deterministic function $\overline{f} : \{0,1\}^n \to \mathbb{R}$ by $\overline{f}(x) = \sum_{S\subseteq[d]} \sqrt{\mathrm{E}_f\left[\hat{f}^{(w)}(S)^2\right]}\phi_S(x)$. Using the Parseval-Plancherel identity we get that

$$\mathop{\mathrm{E}}_f\left[\langle T_\rho^{(w)} f, f\rangle_{L_2(w)}\right] = \sum_{S\subseteq[d]} \rho^{|S|} \mathop{\mathrm{E}}_f\left[\hat{f}^{(w)}(S)^2\right] = \langle T_\rho^{(w)} \overline{f}, \overline{f}\rangle_{L_2(w)} .$$

for every ρ. We set $\alpha = \frac{w_1-w^2}{w(1-w)}$ and $\beta = \frac{w_2-w^2}{w(1-w)}$ and note that $\Pr_{x,y',f}[f(x) = 1, f(y') = 1] = \mathrm{E}_f\left[\langle T_\alpha^{(w)} f, f\rangle_{L_2(w)}\right]$, $\Pr_{x,y,f}[f(x) = 1, f(y) = 1] = \mathrm{E}_f\left[\langle T_\beta^{(w)} f, f\rangle_{L_2(w)}\right]$, and $\Pr_{x,f}[f(x) = 1] =$

$\mathrm{E}_f\left[\|f\|^2_{L_2(w)}\right]$. Then using Lemma 3.2 we get that

$$\begin{aligned}
\left(\frac{\Pr_{x,y',f}[f(x)=1, f(y')=1]}{\Pr_{x,f}[f(x)=1]}\right)^{1/\log 1/\alpha} &= \left(\frac{\mathrm{E}_f\left[\langle T_\alpha^{(w)} f, f\rangle_{L_2(w)}\right]}{\mathrm{E}_f\left[\|f\|^2_{L_2(w)}\right]}\right)^{1/\log 1/\alpha} \\
&= \left(\frac{\langle T_\alpha^{(w)} \overline{f}, \overline{f}\rangle_{L_2(w)}}{\|\overline{f}\|^2_{L_2(w)}}\right)^{1/\log 1/\alpha} \\
&\le \left(\frac{\langle T_\beta^{(w)} \overline{f}, \overline{f}\rangle_{L_2(w)}}{\|\overline{f}\|^2_{L_2(w)}}\right)^{1/\log 1/\beta} \\
&= \left(\frac{\Pr_{x,y,f}[f(x)=1, f(y)=1]}{\Pr_{x,f}[f(x)=1]}\right)^{1/\log 1/\beta} .
\end{aligned}$$

By rearrangement this implies that

$$\rho = \frac{\log \frac{\Pr_{x,y',f}[f(x)=1, f(y')=1]}{\Pr_{x,f}[f(x)=1]}}{\log \frac{\Pr_{x,y,f}[f(x)=1, f(y)=1]}{\Pr_{x,f}[f(x)=1]}} \ge \frac{\log\alpha}{\log\beta} .$$

□

As noted the bound is sharp against our upper bound when w_u, w_q, w_1, w_2 are all small. Also notice that $\log\alpha/\log\beta \le \frac{1-\alpha}{1+\alpha}\frac{1-\beta}{1+\beta}$ is a rather good approximation for α and β close to 1. Here the right hand side is the ρ value of Spherical LSH with the batch-normalization embedding discussed in Section 4.1.

Note that the lower bound becomes 0 when we get close to the random instance, $w_2 \to w_q w_u$. In the next sections we will remedy this, by showing a lower bound tight exactly when $w_2 = w_q w_u$.

3.3 General Lower Bound

Our second lower bound will be proven in the "list-of-points" model. We follow and expand upon the approach by Andoni et al. [10]. The main idea is to lower bound random instances with planted points. If the random instances correspond to a Similarity Search problem with high probability then we have a lower bound for the Similarity Search problem. We formalize the notion of random instances in the following general definition.

Definition 3 (Random instance)**.** *For spaces Q and U we describe a distribution of dataset-query pairs (P, q) where $P \subseteq U$ and $q \in Q$. Let $\mathcal{P}_{QU}$ be a probability distribution on $Q \times U$, a $\mathcal{P}_{QU}$-random instance is a dataset-query pair drawn from the following distribution.*

1. *A dataset $P \subseteq U$ is constructed by sampling n points where $p \sim \mathcal{P}_U$ for all $p \in P$.*
2. *A dataset point $p' \in P$ is fixed and a $q \in Q$ is sampled such that $(q, p') \sim \mathcal{P}_{QU}$.*
3. *The goal of the data structure is to preprocess P such that it recovers p' when given the query point q.*

We can then generalize the result by Andoni et al. [10], who proved a result specifically for random Hamming instances, to general random instances. We defer the proof to Appendix A.

Lemma 3.3. *Let Q and U be some spaces and $\mathcal{P}_{QU}$ a probability distribution on $Q \times U$. Consider any list-of-points data structure for $\mathcal{P}_{QU}$-random instances of n points, which uses expected space $n^{1+\rho_u}$, has expected query time $n^{\rho_q - o_n(1)}$, and succeeds with probability at least 0.99. Let $r, s \in [1, \infty]$ satisfy*

$$\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{QU}} [f(X)g(Y)] \le \|f(X)\|_{L_r(\mathcal{P}_Q)} \|g(Y)\|_{L_s(\mathcal{P}_U)} \ ,$$

for all functions $f : Q \to \mathbb{R}$ and $g : U \to \mathbb{R}$. Then

$$\frac{1}{r}\rho_q + \frac{1}{r'}\rho_u \ge \frac{1}{r} + \frac{1}{s} - 1 \ ,$$

where $r' = \frac{r}{r-1}$ is the convex conjugate of r.

This gives a good way to lower bound random instances when one has tight hypercontractive inequalities. Unfortunately, for most probability distributions this is not the case but we can amplify the power of Lemma 3.3 by combining it with Lemma 1.1 which we recall from the introduction.

Lemma 1.1. *Let $\mathcal{P}_{XY}$ be a probability distribution on a space $\Omega_X \times \Omega_Y$ and let $\mathcal{P}_X$ and $\mathcal{P}_Y$ be the marginal distributions on the spaces Ω_X and Ω_Y respectively. Let $s, r \in [1, \infty)$, then the following is equivalent*

1. *For all functions $f : \Omega_X \to \mathbb{R}$ and $g : \Omega_Y \to \mathbb{R}$ we have*

$$\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}} [f(X)g(Y)] \le \|f(X)\|_{L_r(\mathcal{P}_X)} \|g(Y)\|_{L_s(\mathcal{P}_Y)} \ . \tag{9}$$

2. *For all probability distributions $\mathcal{Q}_{XY}$ which are absolutely continuous with respect to $\mathcal{P}_{XY}$ we have*

$$\mathrm{D}(\mathcal{Q}_{XY} \,\|\, \mathcal{P}_{XY}) \ge \frac{\mathrm{D}(\mathcal{Q}_X \,\|\, \mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y \,\|\, \mathcal{P}_Y)}{s} \ . \tag{10}$$

We defer the proof to the end of the section and instead start by focusing on the effects of combining Lemma 3.3 and Lemma 1.1. First of all we can prove the following general lower bound for random instances.

Theorem 4. *Let Q and U be some spaces and $\mathcal{P}_{QU}$ a probability distribution on $Q \times U$. Consider any list-of-points data structure for $\mathcal{P}_{QU}$-random instances of n points, which uses expected space $n^{1+\rho_u}$, has expected query time $n^{\rho_q - o_n(1)}$, and succeeds with probability at least 0.99. Then for every $r \in [1, \infty]$ we have that*

$$\frac{1}{r}\rho_q + \frac{1}{r'}\rho_u \ge \inf_{\mathcal{Q}_{QU}} \left(\frac{1}{r} \frac{\mathrm{D}(\mathcal{Q}_{QU} \,\|\, \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_Q \,\|\, \mathcal{P}_Q)}{\mathrm{D}(\mathcal{Q}_U \,\|\, \mathcal{P}_U)} + \frac{1}{r'} \frac{\mathrm{D}(\mathcal{Q}_{QU} \,\|\, \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_U \,\|\, \mathcal{P}_U)}{\mathrm{D}(\mathcal{Q}_U \,\|\, \mathcal{P}_U)} \right) \ ,$$

where $r' = \frac{r}{r-1}$ is the convex conjugate of r and the infimum is over every probability distribution $\mathcal{Q}_{QU}$ with $\mathcal{Q}_U \neq \mathcal{P}_U$ and which is absolutely continuous with respect to $\mathcal{P}_{QU}$.

Proof. Let $r \in [1, \infty]$ and choose $s = \arg\inf \{s \in [1, \infty] \mid \mathcal{P}_{QU} \text{ is } (r, s)\text{-hypercontractive}\}$. Lemma 3.3 give us that

$$\frac{1}{r}\rho_q + \frac{1}{r'}\rho_u \ge \frac{1}{r} + \frac{1}{s} - 1 \ . \tag{11}$$

Lemma 1.1 give us that

$$\mathrm{D}(\mathcal{Q}_{XY} \,\|\, \mathcal{P}_{XY}) \geq \frac{\mathrm{D}(\mathcal{Q}_X \,\|\, \mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y \,\|\, \mathcal{P}_Y)}{s}$$

for every $\mathcal{Q}_{QU}$ with $\mathcal{Q}_U \neq \mathcal{P}_U$ and which is absolutely continuous with respect to $\mathcal{P}_{QU}$. We can rewrite this as

$$\frac{1}{r}\mathrm{D}(\mathcal{Q}_{QU} \,\|\, \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_Q \,\|\, \mathcal{P}_Q) + \frac{1}{r'}\mathrm{D}(\mathcal{Q}_{QU} \,\|\, \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_U \,\|\, \mathcal{P}_U) \geq \left(\frac{1}{s} - \frac{1}{r'}\right)\mathrm{D}(\mathcal{Q}_U \,\|\, \mathcal{P}_U) \quad \Leftrightarrow$$

$$\frac{1}{r}\frac{\mathrm{D}(\mathcal{Q}_{QU} \,\|\, \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_Q \,\|\, \mathcal{P}_Q)}{\mathrm{D}(\mathcal{Q}_U \,\|\, \mathcal{P}_U)} + \frac{1}{r'}\frac{\mathrm{D}(\mathcal{Q}_{QU} \,\|\, \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_U \,\|\, \mathcal{P}_U)}{\mathrm{D}(\mathcal{Q}_U \,\|\, \mathcal{P}_U)} \geq \frac{1}{s} - \frac{1}{r'} = \frac{1}{s} + \frac{1}{r} - 1$$

Now the minimality of s give us that

$$\inf_{\mathcal{Q}_{QU}}\left(\frac{1}{r}\frac{\mathrm{D}(\mathcal{Q}_{QU} \,\|\, \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_Q \,\|\, \mathcal{P}_Q)}{\mathrm{D}(\mathcal{Q}_U \,\|\, \mathcal{P}_U)} + \frac{1}{r'}\frac{\mathrm{D}(\mathcal{Q}_{QU} \,\|\, \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_U \,\|\, \mathcal{P}_U)}{\mathrm{D}(\mathcal{Q}_U \,\|\, \mathcal{P}_U)}\right) = \frac{1}{s} + \frac{1}{r} - 1\,, \tag{12}$$

where the infimum is over every probability distribution $\mathcal{Q}_{QU}$ with $\mathcal{Q}_U \neq \mathcal{P}_U$ and which is absolutely continuous with respect to $\mathcal{P}_{QU}$. Now combining (11) and (12) give us the result. □

Combining the lemma with the "Hypercontractive Induction Theorem" [52] we can prove Theorem 3.

Lemma 3.4. *Let $\mathcal{P}_{XY}$ be a probability distribution on a space $\Omega_X \times \Omega_Y$ and $\mathcal{P}_{XY}^{\otimes n}$ be a probability distribution consisting n independent copies of $\mathcal{P}_{XY}$. Then $\mathcal{P}_{XY}$ is (r,s)-hypercontractive if and only if $\mathcal{P}_{XY}^{\otimes n}$ is (r,s)-hypercontractive.*

We restate Theorem 3 and prove it.

Theorem 3. *Consider any list-of-point data structure for the $(w_q, w_u, w_1, w_q w_u)$-GapSS problem over a universe of size d of n points with $w_q w_u d = \omega(\log n)$, which uses expected space $n^{1+\rho_u}$, has expected query time $n^{\rho_q - o_n(1)}$, and succeeds with probability at least 0.99. Then for every $\alpha \in [0,1]$ we have that*

$$\alpha\rho_q + (1-\alpha)\rho_u \geq \inf_{\substack{t_q,t_u \in [0,1]\\ t_u \neq w_u}}\left(\alpha\frac{\mathrm{D}(T \,\|\, P) - \mathrm{d}(t_q \,\|\, w_q)}{\mathrm{d}(t_u \,\|\, w_u)} + (1-\alpha)\frac{\mathrm{D}(T \,\|\, P) - \mathrm{d}(t_u \,\|\, w_u)}{\mathrm{d}(t_u \,\|\, w_u)}\right)\,,$$

where $P = \left[\begin{smallmatrix} w_1 & w_q - w_1 \\ w_u - w_1 & 1 - w_q - w_u + w_1\end{smallmatrix}\right]$ *and* $T = \underset{T \ll P,\, \underset{X\sim T}{E}[X] = \left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]}{\arg\inf} \mathrm{D}(T \,\|\, P)$.

Proof. From the discussion at the beginning of the section it is enough to lower bound the $P^{\otimes d}$-random instance where $P = \text{Bernoulli}(\left[\begin{smallmatrix} w_1 & w_q - w_1 \\ w_u & 1 - w_q - w_u + w_1\end{smallmatrix}\right])$, since this will imply a lower bound for the $(w_q, w_u, w_1, w_q w_u)$-GapSS problem. Combining Lemma 3.4 and Lemma 1.1 we get that $P^{\otimes d}$ is (r,s)-hypercontractive if and only if $\mathrm{D}(T \,\|\, P) \geq \frac{\mathrm{d}(t_q \,\|\, w_q)}{r} + \frac{\mathrm{d}(t_u \,\|\, w_u)}{s}$ where $T = \underset{T \ll P,\, \underset{X\sim T}{E}[X] = \left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]}{\arg\inf} \mathrm{D}(T \,\|\, P)$. Now repeating the proof of Theorem 4 give us the result. □

Proof of Lemma 1.1 We now turn to the proof Lemma 1.1. The main argument needed in the proof of is contained in the following lemma, which can be seen as a variation of Fenchel's inequality.

Lemma 3.5. *Let P be a probability distribution on a space Ω, Q a probability which is absolutely continuous with respect to P, and $\phi : \Omega \to \mathbb{R}$ a function such that $\mathrm{E}_P\left[\exp(\phi(X))\right] \leq \infty$. Then*

$$\mathrm{D}(Q \,\|\, P) + \log \mathop{\mathrm{E}}_{X\sim P}\left[\exp(\phi(X))\right] \geq \mathop{\mathrm{E}}_{X\sim Q}\left[\phi(X)\right] \ .$$

and we have equality if and only if $\frac{dQ}{dP}(x) = \frac{\exp(\phi(x))}{\mathrm{E}_{X\sim P}[\exp(\phi(X))]}$.

Proof. To ease notation we write $p = \frac{dQ}{dP}$. We note that

$$\mathrm{D}(Q \,\|\, P) = \mathop{\mathrm{E}}_{X\sim Q}\left[\log p(X)\right] = \mathop{\mathrm{E}}_{X\sim Q}\left[\log \frac{p(X)}{\exp(\phi(X))}\right] + \mathop{\mathrm{E}}_{X\sim Q}\left[\phi(X)\right] = \mathop{\mathrm{E}}_{X\sim Q}\left[\phi(X)\right] - \mathop{\mathrm{E}}_{X\sim Q}\left[\log \frac{\exp(\phi(X))}{p(X)}\right] \ .$$

Using Jensen's inequality we get that

$$\mathop{\mathrm{E}}_{X\sim Q}\left[\log \frac{\exp(\phi(x))}{p(X)}\right] \leq \log \mathop{\mathrm{E}}_{X\sim Q}\left[\exp(\phi(x))\frac{dP}{dQ}(x)\right] = \log \mathop{\mathrm{E}}_{X\sim P}\left[\exp(\phi(x))\right] \ .$$

Combining these two equations give us the inequality. Now we note that we have equality if and only if $e^{\phi(x)}\frac{dP}{dQ}(x)$ is constant, and since Q is a probability distribution this is equivalent with $\frac{dQ}{dP}(x) = \frac{\exp(\phi(x))}{\mathrm{E}_{X\sim P}[\exp(\phi(X))]}$. □

We are now ready to prove Lemma 1.1.

Proof of Lemma 1.1. (9) ⇒ (10). Let $\mathcal{Q}_{XY}$ be a probability distribution which is absolutely continuous with respect to $\mathcal{P}_{XY}$. We set $\exp(\phi_X(x)) = \frac{d\mathcal{Q}_X}{d\mathcal{P}_X}(x)$ and $\exp(\phi_Y(y)) = \frac{d\mathcal{Q}_Y}{d\mathcal{P}_Y}(y)$. From this we see that $\mathrm{E}_{X\sim\mathcal{P}_X}\left[\exp(\phi_X(X))\right] = \mathrm{E}_{X\sim\mathcal{P}_x}\left[\frac{d\mathcal{Q}_X}{d\mathcal{P}_X}(X)\right] = \mathrm{E}_{X\sim\mathcal{Q}_X}\left[1\right] = 1$ and similarly that $\mathrm{E}_{Y\sim\mathcal{P}_Y}\left[\exp(\phi_X(X))\right] = 1$, hence we have that $\frac{d\mathcal{Q}_X}{d\mathcal{P}_X}(x) = \frac{\exp(\phi_X(x))}{\mathrm{E}_{X\sim\mathcal{P}_X}[\exp(\phi_X(X))]}$ and $\frac{d\mathcal{Q}_Y}{d\mathcal{P}_Y}(y) = \frac{\exp(\phi_Y(y))}{\mathrm{E}_{Y\sim\mathcal{P}_Y}[\exp(\phi_X(Y))]}$. Using (9) we get that

$$\begin{aligned}
&\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}\left[\exp(\phi_X(X) + \phi_Y(Y)\right] \leq \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X}\left[\exp(r\phi_X(X))\right]^{1/r} \mathop{\mathrm{E}}_{Y\sim\mathcal{P}_Y}\left[\exp(s\phi_Y(Y)\right]^{1/s} \qquad \Leftrightarrow \\
&\log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}\left[\exp(\phi_X(X) + \phi_Y(Y)\right] \leq \frac{\log \mathrm{E}_{X\sim\mathcal{P}_X}\left[\exp(r\phi_X(X))\right]}{r} + \frac{\log \mathrm{E}_{Y\sim\mathcal{P}_Y}\left[\exp(s\phi_Y(Y)\right]}{s} \ .
\end{aligned}$$

Using Lemma 3.5 3 times we have that

$$\begin{aligned}
\log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}\left[\exp(\phi_X(X) + \phi_Y(Y))\right] &\geq \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{Q}_{XY}}\left[\phi_X(X) + \phi_Y(Y)\right] - \mathrm{D}(\mathcal{Q}_{XY} \,\|\, \mathcal{P}_{XY}) \\
\log \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X}\left[\exp(\phi_X(X))\right] &= \mathop{\mathrm{E}}_{X\sim\mathcal{Q}_X}\left[\phi_X(X)\right] - \mathrm{D}(\mathcal{Q}_X \,\|\, \mathcal{P}_X) \\
\log \mathop{\mathrm{E}}_{X\sim\mathcal{P}_Y}\left[\exp(\phi_Y(Y))\right] &= \mathop{\mathrm{E}}_{Y\sim\mathcal{Q}_Y}\left[\phi_Y(Y)\right] - \mathrm{D}(\mathcal{Q}_Y \,\|\, \mathcal{P}_Y) \ ,
\end{aligned}$$

where the equalities hold since $\frac{d\mathcal{Q}_X}{d\mathcal{P}_X}(x) = \frac{\exp(\phi_X(x))}{\mathrm{E}_{X\sim\mathcal{P}_X}[\exp(\phi_X(X))]}$ and $\frac{d\mathcal{Q}_Y}{d\mathcal{P}_Y}(y) = \frac{\exp(\phi_Y(y))}{\mathrm{E}_{Y\sim\mathcal{P}_Y}[\exp(\phi_X(Y))]}$. We then get that

$$\begin{aligned}
&\log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}} [\exp(\phi_X(X) + \phi_Y(Y)] \le \frac{\log \mathrm{E}_{X\sim\mathcal{P}_X} [\exp(r\phi_X(X))]}{r} + \frac{\log \mathrm{E}_{Y\sim\mathcal{P}_Y} [\exp(s\phi_Y(Y)]}{s} && \Rightarrow \\
&\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{Q}_{XY}} [\phi_X(X) + \phi_Y(Y)] - \mathrm{D}(\mathcal{Q}_{XY} \,\|\, \mathcal{P}_{XY}) \\
&\quad \le \mathop{\mathrm{E}}_{X\sim\mathcal{Q}_X} [\phi_X(X)] - \mathrm{D}(\mathcal{Q}_X \,\|\, \mathcal{P}_X) + \mathop{\mathrm{E}}_{Y\sim\mathcal{Q}_Y} [\phi_Y(Y)] - \mathrm{D}(\mathcal{Q}_Y \,\|\, \mathcal{P}_Y) && \Leftrightarrow \\
&\mathrm{D}(\mathcal{Q}_{XY} \,\|\, \mathcal{P}_{XY}) \ge \frac{\mathrm{D}(\mathcal{Q}_X \,\|\, \mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y \,\|\, \mathcal{P}_Y)}{s} \ ,
\end{aligned}$$

which proves that (9) ⇒ (10).

(10) ⇒ (9). Fix the functions $f : \Omega_X \to \mathbb{R}$ and $g : \Omega_Y \to \mathbb{R}$. We note that $\mathrm{E}_{(X,Y)\sim\mathcal{P}_{XY}} [f(X)g(Y)] \le \mathrm{E}_{(X,Y)\sim\mathcal{P}_{XY}} [|f|(X)\,|g|(Y)]$ hence we can assume that f and g are non-negative. We define $\phi_X(x) = \log(f(x))$ and $\phi_Y(x) = \log(g(x))$[17]. Then (9) is equivalent with

$$\begin{aligned}
&\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}} [\exp(\phi_X(X) + \phi_Y(Y)] \le \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X} [\exp(r\phi_X(X))]^{1/r} \mathop{\mathrm{E}}_{Y\sim\mathcal{P}_Y} [\exp(s\phi_Y(Y)]^{1/s} && \Leftrightarrow \\
&\log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}} [\exp(\phi_X(X) + \phi_Y(Y)] \le \frac{\log \mathrm{E}_{X\sim\mathcal{P}_X} [\exp(r\phi_X(X))]}{r} + \frac{\log \mathrm{E}_{Y\sim\mathcal{P}_Y} [\exp(s\phi_Y(Y)]}{s} \ .
\end{aligned}$$

We define the probability distribution $\mathcal{Q}_{XY}$ by $\frac{d\mathcal{Q}_{XY}}{d\mathcal{P}_{XY}}(x,y) = \frac{\exp(\phi_X(X)+\phi_Y(Y)}{\mathrm{E}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y))]}$. It is easy to see that $\mathcal{Q}_{XY}$ is indeed a probability distribution. Using (10) we get that

$$\mathrm{D}(\mathcal{Q}_{XY} \,\|\, \mathcal{P}_{XY}) \ge \frac{\mathrm{D}(\mathcal{Q}_X \,\|\, \mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y \,\|\, \mathcal{P}_Y)}{s} \ .$$

Using Lemma 3.5 3 times we have that

$$\begin{aligned}
\mathrm{D}(\mathcal{Q}_{XY} \,\|\, \mathcal{P}_{XY}) &= \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{Q}_{XY}} [\phi_X(X) + \phi_Y(Y)] - \log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}} [\exp(\phi_X(X) + \phi_Y(Y))] \\
\mathrm{D}(\mathcal{Q}_X \,\|\, \mathcal{P}_X) &\ge \mathop{\mathrm{E}}_{X\sim\mathcal{Q}_X} [\phi_X(X)] - \log \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X} [\exp(\phi_X(X))] \\
\mathrm{D}(\mathcal{Q}_Y \,\|\, \mathcal{P}_Y) &\ge \mathop{\mathrm{E}}_{Y\sim\mathcal{Q}_Y} [\phi_Y(Y)] - \log \mathop{\mathrm{E}}_{X\sim\mathcal{P}_Y} [\exp(\phi_Y(Y))] \ ,
\end{aligned}$$

where the equality holds since $\frac{d\mathcal{Q}_{XY}}{d\mathcal{P}_{XY}}(x,y) = \frac{\exp(\phi_X(X)+\phi_Y(Y)}{\mathrm{E}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y))]}$. We then get that

$$\begin{aligned}
&\mathrm{D}(\mathcal{Q}_{XY} \,\|\, \mathcal{P}_{XY}) \ge \frac{\mathrm{D}(\mathcal{Q}_X \,\|\, \mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y \,\|\, \mathcal{P}_Y)}{s} && \Rightarrow \\
&\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{Q}_{XY}} [\phi_X(X) + \phi_Y(Y)] - \log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}} [\exp(\phi_X(X) + \phi_Y(Y))] \\
&\quad \ge \mathop{\mathrm{E}}_{X\sim\mathcal{Q}_X} [\phi_X(X)] - \log \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X} [\exp(\phi_X(X))] + \mathop{\mathrm{E}}_{Y\sim\mathcal{Q}_Y} [\phi_Y(Y)] - \log \mathop{\mathrm{E}}_{X\sim\mathcal{P}_Y} [\exp(\phi_Y(Y))] && \Leftrightarrow \\
&\log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}} [\exp(\phi_X(X) + \phi_Y(Y)] \le \frac{\log \mathrm{E}_{X\sim\mathcal{P}_X} [\exp(r\phi_X(X))]}{r} + \frac{\log \mathrm{E}_{Y\sim\mathcal{P}_Y} [\exp(s\phi_Y(Y)]}{s} \ ,
\end{aligned}$$

which proves that (10) ⇒ (9). □

[17] We define $\log(0) = -\infty$ and $\exp(-\infty) = 0$.

3.4 Explicit Hypercontractive Bounds

In this section we show how to relate the directed noise operator to the lower bounds of Oleszkiewicz [53], thereby giving direct lower bounds for a number of cases for s and r. By Theorem 3 and Lemma 3.3 this is the dual to proving optimal values (t_q, t_u) in our upper bound.

We start by with a standard lemma which shows that hypercontractivity of an operator implies hypercontractivity of its adjoint.

Lemma 3.6. *Let $T : L_2(\Omega, \pi) \to L_2(\Omega, \pi')$ be an operator with $T^* : L_2(\Omega, \pi') \to L_2(\Omega, \pi)$ being its adjoint, and let $1 \leq r, s < \infty$ with r', s' being their convex conjugates. Then*

$$\|Tf\|_{L_{s'}(\pi')} \leq \|f\|_{L_r(\pi)}$$

holds for all $f \in L_2(\Omega, \pi)$, if and only if

$$\langle Tf, g\rangle_{L_2(\pi')} = \langle f, T^* g\rangle_{L_2(\pi)} \leq \|f\|_{L_r(\pi)} \|g\|_{L_s(\pi')}$$

holds for all $f \in L_2(\Omega, \pi)$ and all $g \in L_2(\Omega, \pi')$, if and only if

$$\|T^* g\|_{L_{r'}(\pi)} \leq \|g\|_{L_s(\pi')}$$

holds for all $g \in L_2(\Omega, \pi')$.

Proof. We assume that $\|Tf\|_{L_{s'}(\pi')} \leq \|f\|_{L_r(\pi)}$ holds for all $f \in L_2(\Omega, \pi)$. Let $f \in L_2(\Omega, \pi)$ and $g \in L_2(\Omega, \pi')$ then by Hölder's inequality we have that

$$\langle Tf, g\rangle_{L_2(\pi')} \leq \|Tf\|_{L_{s'}(\pi')} \|g\|_{L_s(\pi')} \leq \|f\|_{L_r(\pi)} \|g\|_{L_s(\pi')} \ .$$

Similarly, we assume that $\|T^* g\|_{L_{r'}(\pi)} \leq \|g\|_{L_s(\pi')}$ holds for all $g \in L_2(\Omega, \pi')$. Let $f \in L_2(\Omega, \pi)$ and $g \in L_2(\Omega, \pi')$ then by Hölder's inequality we have that

$$\langle f, T^* g\rangle_{L_2(\pi)} \leq \|f\|_{L_r(\pi')} \|T^* g\|_{L_{r'}(\pi')} \leq \|f\|_{L_r(\pi)} \|g\|_{L_{s'}(\pi')} \ .$$

Finally, we assume that $\langle Tf, g\rangle_{L_2(\pi')} \leq \|f\|_{L_r(\pi)} \|g\|_{L_s(\pi')}$ holds for all $f \in L_2(\Omega, \pi)$ and all $g \in L_2(\Omega, \pi')$. Let $f \in L_2(\Omega, \pi)$ then using that $L_s(\pi')$ is the dual norm of $L_{s'}(\pi')$ we get that

$$\|Tf\|_{L_{s'}(\pi')} = \sup_{\|g\|_{L_s(\pi')}=1} \langle Tf, g\rangle_{L_2(\pi')} \leq \sup_{\|g\|_{L_s(\pi')}=1} \|f\|_{L_s(\pi)} \|g\|_{L_r(\pi')} = \|f\|_{L_r(\pi)} \ .$$

Similarly, let $g \in L_2(\Omega, \pi')$ then using that $L_r(\pi)$ is the dual norm of $L_{r'}(\pi)$ we get that

$$\|T^* g\|_{L_{r'}(\pi)} = \sup_{\|f\|_{L_r(\pi)}=1} \langle f, T^* g\rangle_{L_2(\pi)} \leq \sup_{\|f\|_{L_r(\pi')}=1} \|f\|_{L_r(\pi)} \|g\|_{L_s(\pi')} = \|g\|_{L_s(\pi')} \ ,$$

which finishes the proof. □

Our hypercontractive results will be based on the tight hypercontractive inequality by Oleszkiewicz [53].

Theorem 5 ([53]). *Let $p \in (0, \frac{1}{2}) \cup (\frac{1}{2}, 1)$ and $1 \leq r \leq 2$ then for any function $f \in L_2(\{0,1\}^d, \pi_p^{\otimes d})$ we have that*

$$\left\| T_\rho^{(p)} f \right\|_{L_2(p)} \leq \|f\|_{L_r(p)} \ , F$$

where $\rho = p^{-1/2}(1-p)^{-1/2} \sqrt{\frac{(1-p)^{2-2/r} - p^{2-2/r}}{p^{-2/r} - (1-p)^{-2/r}}}$ which is best possible.

From this we get following tight hypercontractive inequalities for $T_\rho^{p_1\to p_2}$.

Corollary 2. *Let $p_1, p_2 \in (0, \frac{1}{2}) \cup (\frac{1}{2}, 1)$ and $1 \le r \le 2$ then for any function $f \in L_2(\{0,1\}^d, \pi_{p_1}^{\otimes d})$ we have that*

$$\left\|T_\rho^{p_1\to p_2} f\right\|_{L_2(p_2)} \le \|f\|_{L_r(p_1)} \ ,$$

where $\rho = p_1^{-1/2}(1-p_1)^{-1/2}\sqrt{\frac{(1-p_1)^{2-2/r}-p_1^{2-2/r}}{p_1^{-2/r}-(1-p_1)^{-2/r}}}$ which is best possible.

Proof. Using the Parseval-Plancherel identity and Lemma 3.1 we get that

$$\left\|T_\rho^{p_1\to p_2} f\right\|_{L_2(p_2)}^2 = \sum_{S\subseteq[d]} \widehat{T_\rho^{p_1\to p_2} f}^{(p_2)}(S)^2 = \sum_{S\subseteq[d]} \rho^{2|S|}\widehat{T_\rho^{p_1\to p_2} f}^{(p_2)}(S)^2 = \left\|T^{(p_1)} f\right\|_{L_2(p_1)}^2 \ ,$$

hence the result follows from Theorem 5. □

Corollary 3. *Let $p_1, p_2 \in (0, \frac{1}{2}) \cup (\frac{1}{2}, 1)$ and $1 \le s \le 2$ with s' the convex conjugate of s, then for any function $f \in L_2(\{0,1\}^d, \pi_{p_1}^{\otimes d})$ we have that*

$$\left\|T_\rho^{p_1\to p_2} f\right\|_{L_{s'}(p_2)} \le \|f\|_{L_2(p_1)} \ ,$$

where $\rho = p_2^{-1/2}(1-p_2)^{-1/2}\sqrt{\frac{(1-p_2)^{2-2/s}-p_2^{2-2/s}}{p_2^{-2/s}-(1-p_2)^{-2/s}}}$ which is best possible.

Proof. By Lemma 3.6 we get that the result is true if and only if

$$\left\|T_\rho^{p_2\to p_1} g\right\|_{L_2(p_1)} \le \|g\|_{L_s(p_2)} \ ,$$

for all functions $g \in L_2(\{0,1\}, \pi_{p_2})$. Now the result follows by using Corollary 2. □

We also get a hypercontractive inequality for the standard noise operator $T_\rho^{(p)}$.

Corollary 4. *Let $p \in (0, \frac{1}{2}) \cup (\frac{1}{2}, 1)$ and $1 \le r \le 2$ with convex conjugate r', then for any function $f \in L_2(\{0,1\}^d, \pi_p^{\otimes d})$ we have that*

$$\left\|T_\rho^{(p)} f\right\|_{L_{r'}(p)} \le \|f\|_{L_r(p)} \ ,$$

where $\rho = p(1-p)\frac{(1-p)^{2-2/r}-p^{2-2/r}}{p^{-2/r}-(1-p)^{-2/r}}$ which is best possible.

Proof. Using Lemma 3.6 we get the result holds if and only if

$$\langle T_\rho^{(p)} f, g\rangle_{L_2(p)} \le \|f\|_{L_r(p)} \|g\|_{L_r(p)} \ ,$$

holds for all $f, g \in L_2(\{0,1\}^d, \pi_p^{\otimes d})$. First we note that the result is true by using Cauchy-Schwartz and Theorem 5

$$\langle T_\rho^{(p)} f, g\rangle_{L_2(p)} = \langle T_{\sqrt{\rho}}^{(p)} f, T_{\sqrt{\rho}}^{(p)} g\rangle_{L_2(p)} \le \left\|T_{\sqrt{\rho}}^{(p)} f\right\|_{L_2p} \left\|T_{\sqrt{\rho}}^{(p)} g\right\|_{L_2p} \le \|f\|_{L_rp} \|f\|_{L_rp}$$

Now to see that ρ is best possible we set $g = f$ which give us that

$$\left\|T_{\sqrt{\rho}}^{(p)} f\right\|_{L_2p}^2 = \langle T_\rho^{(p)} f, f\rangle_{L_2(p)} \le \|f\|_{L_rp}^2 \ ,$$

so Theorem 5 gives that ρ is best possible. □

We will use Corollary 4 to show that setting $(t_q, t_u) = (1-w, 1-w)$ is an optimal threshold for the (w, w, w_1, w^2)-GapSS problem. First of we note that $\rho = p(1-p)\frac{(1-p)^{2-2/r}-p^{2-2/r}}{p^{-2/r}-(1-p)^{-2/r}}$ can be rewritten as $r = \frac{2\log\tau}{\log\frac{\rho+\tau}{\rho+\tau^{-1}}}$ where $\tau = \frac{1-p}{p}$. Using Lemma 3.3 we get that $\rho = \frac{w_1-w^2}{w(1-w)}$, $\tau = \frac{1-w}{w}$, and that

$$\frac{1}{r}\rho_q + \frac{1}{r'}\rho_u \geq \frac{2}{r} - 1 \; .$$

Now we have that $\rho_q = \rho_u = \frac{\log\frac{w(w_1-2w+1)}{w_1(1-w)}}{\log\frac{w}{1-w}}$, and we find that

$$\frac{2}{r} - 1 = \frac{\log\frac{\rho+\tau}{\rho+\tau^{-1}}}{\log\tau} - 1 = \frac{\log\tau^{-1}\frac{\rho+\tau}{\rho+\tau^{-1}}}{\log\tau} \; .$$

It is then easy to check that $\tau^{-1}\frac{\rho+\tau}{\rho+\tau^{-1}} = \frac{w_1(1-w)}{w(w_1-2w+1)}$, which then shows that $(t_q, t_u) = (1-w, 1-w)$ is an

4 Other Algorithms

We show two results that, while orthogonal to Supermajorities, help us understand them and how they fit within the space of Similarity Search algorithms.

The first result is an optimal affine embedding of sets onto the sphere. This result is interesting in its own right, as it results in an algorithm that is in many cases better than the state of the art, and which can be implemented very easily in systems that can already solve Euclidean or Spherical Nearest Neighbours. The result gives a simple, general condition a Spherical LSH scheme must meet for the embedding to be optimal, and we show that both SimHash and Spherical LSH meets it.

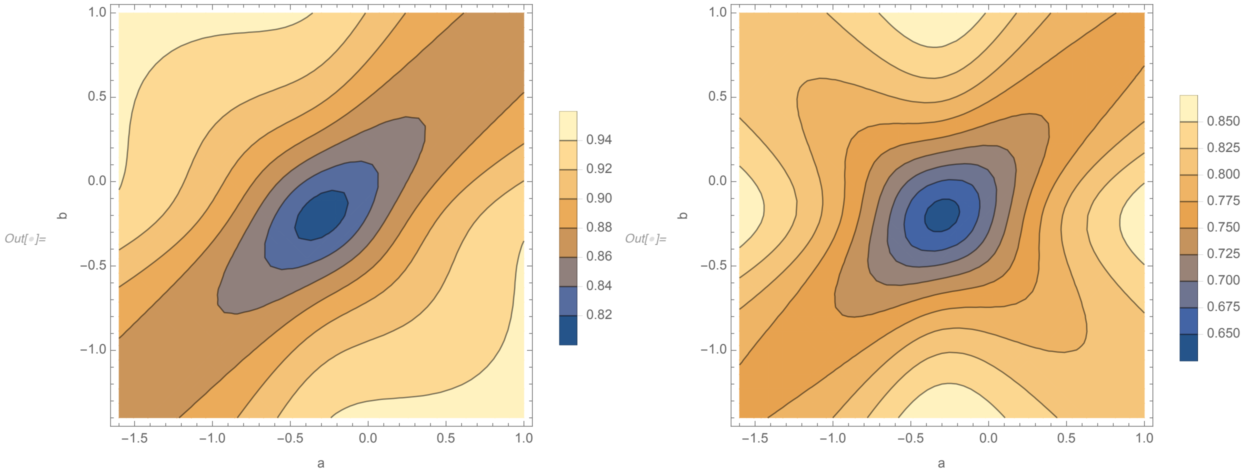
The second result is also a new algorithm. In particular, it is a mix between Chosen Path and MinHash, which always achieves ρ values lower than both of them. It is in a sense a simple answer to the open problem in [28] about how to beat MinHash consistently. More interesting though, is that it sheds light on what makes Supermajorities work: It balances the amount of information pulled from sets vs. their complements. The proof is also conceptually interesting, since it proves that it is never advantageous to combine multiple Locality Sensitive Filter families.

4.1 Embedding onto the Sphere

We show, that if an algorithm has exponent $\rho(\alpha, \beta) = f(\alpha)/f(\beta)$ where α is the cosine similarity between good points and β is the similarity between bad points on the sphere; then assuming some light properties on f, which contain both Spherical and Hyperplane LSH, two affine embedding of sets $x \in \{0,1\}^d$ to S^{d-1} that minimizes ρ once the new cosine similarities are calculated, is $x \mapsto (x-w)/\sqrt{w(1-w)}$ where $w = |x|/d$. While the mapping is allowed to depend on any of the GapSS parameters, it curiously only cares about the weight of the set itself. For fairness, all our plots, such as Figure 2, uses this embedding when comparing Supermajorities to Spherical LSH.

Lemma 4.1 (Embedding Lemma)**.** *Let $g, h : \{0,1\}^d \to \mathbb{R}^d$ be function on the form $g(x) = a_1 x + b_1$ and $h(y) = a_2 y + b_2$. Let $\rho(x, y, y') = f(\alpha(x, y))/f(\alpha(x, y'))$ where $\alpha(x, y) = \langle x, y\rangle/\|x\|\|y\|$ be such that*

$$f(z) \geq 0, \quad \tfrac{d}{dz}\left((\pm 1 - z)\tfrac{d}{dz}\log f(z)\right) \geq 0 \quad \textit{and} \quad \tfrac{d^3}{dz^3}\log f(z) \leq 0$$



(a) Query time/space exponent, ρ, for the SimHash algorithm [23].

(b) Query time/space exponent, ρ, for the Spherical LSH algorithm [66].

Figure 6: Given a GapSS instance with $w_q = .3$ and $w_u = .2$, the optimal affine embedding of the data (represented as vectors $x \in \{0,1\}^{|U|}$) onto the sphere, turns out to be normalizing the "mean" and "variance". That is, before scaling down to $\|x\|_2 = 1$, we subtract respectively w_q and w_u from all coordinates. The plot shows the "ρ-value" achieved by different spherical algorithms as the among subtracted is varied: The x-axis, a, is the amount subtracted from queries and the y-axis, b, is the amount subtracted from datasets.

for all $z \in [-1,1]$. *Assume we know that* $\|x\|_2^2 = w_q d$, $\|y\|_2^2 = w_u d$, $\langle x, y' \rangle = w_1 d$ *and* $\langle x, y \rangle = w_2 d$, *then* $\arg\min_{a_1,a_2,b_1,b_2} \rho(g(x), h(y), h(y')) = (1, 1, -w_q, -w_u)$.

In this section we will show that Hyperplane [24] and Spherical [10] LSH both satisfy the requirements of the lemma. Hence we get two algorithms with ρ-values:

$$\rho_{\text{hp}} = \frac{\log(1 - \arccos(\alpha)/\pi)}{\log(1 - \arccos(\beta)/\pi)}, \quad \rho_{\text{sp}} = \frac{1-\alpha}{1+\alpha}\frac{1+\beta}{1-\beta}.$$

where $\alpha = \frac{w_1 - w_q w_u}{\sqrt{w_q(1-w_q)w_u(1-w_u)}}$ and $\beta = \frac{w_2 - w_q w_u}{\sqrt{w_q(1-w_q)w_u(1-w_u)}}$, and space/time trade-offs using the ρ_q, ρ_u values in [27]. [18] Figure 6 shows how ρ varies with different translations a, b.

Taking $t_q = w_q(1+o(1))$ and $t_u = w_u(1+o(1))$ in theorem 1 recovers ρ_{sp} by standard arguments. This implies that theorem 1 dominates Spherical LSH (for binary data).

Lemma 4.2. *The functions* $f(z) = (1-z)/(1+z)$ *for Spherical LSH and* $f(z) = -\log(1 - \arccos(z)/\pi)$ *for Hyperplane LSH satisfy lemma 4.1.*

Proof. For Spherical LSH we have $f(z) = (1-z)/(1+z)$ and get

$$\begin{aligned} \tfrac{d}{dz}\left((\pm 1 - z)\tfrac{d}{dz}\log f(z)\right) &= 2/(1 \pm z)^2 \geq 0, \\ \tfrac{d^3}{dz^3}\log f(z) &= -4(1+3z^2)/(1-z^2)^3 \leq 0. \end{aligned}$$

[18] Unfortunately the space/time aren't on a form applicable to lemma 4.1. From numerical experiments we however still conjecture that the embedding is optimal for those as well.

For Hyperplane LSH we have $f(z) = -\log(1 - \arccos(z)/\pi)$ and get

$$\tfrac{d}{dz}\left((\pm 1 - z)\tfrac{d}{dz}\log f(z)\right) = \frac{(\arccos(z) \mp \sqrt{1-z^2} - \pi)\log(1 - \arccos(z)/\pi) \mp \sqrt{1-z^2}}{(1 \pm z)\sqrt{1-z^2}(\pi - \arccos(z))^2 \log(1 - \arccos(z)/\pi)^2}.$$

In both cases the denominator is positive, and the numerator can be shown to be likewise by applying the inequalities $\sqrt{1-z^2} \leq \arccos(z)$, $\sqrt{1-z^2} + \arccos(z) \leq \pi$ and $x \leq \log(1+x)$.

The $\frac{d^3}{dz^3}\log f(z) \leq 0$ requirement is a bit trickier, but a numerical optimization shows that it's in fact less than -1.53. □

Finally we prove the embedding lemma:

Proof of lemma 4.1. We have

$$\alpha = \frac{\langle x + a, y + b\rangle}{\|x + a\|\|y + b\|} = \frac{w_1 + w_q b + w_u a + ab}{\sqrt{(w_q(1+a)^2 + (1-w_q)a^2)(w_u(1+b)^2 + (1-w_u)b^2)}}$$

and equivalent with w_2 for β. We'd like to show that $a = -w_q$, $b = -w_u$ is a minimum for $\rho = f(\alpha)/f(\beta)$.

Unfortunately the f's we are interested in are usually not convex, so it is not even clear that there is just one minimum. To proceed, we make the following substitution $a \to (c+d)\sqrt{w_q(1-w_q)} - w_q$, $b \to (c-d)\sqrt{w_u(1-w_u)} - w_u$ to get

$$\alpha(c,d) = \frac{cd + \frac{w_1 - w_q w_u}{\sqrt{w_q(1-w_q)w_u(1-w_u)}}}{\sqrt{(1+c^2)(1+d^2)}}.$$

We can further substitute $cd \mapsto rs$ and $\sqrt{(1+c^2)(1+d^2)} \mapsto r+1$ or $r \geq 0$, $-1 \leq s \leq 1$, since $1 + cd \leq \sqrt{(1+c^2)(1+d^2)}$ by Cauchy Schwartz, and $(cd, \sqrt{(1+c^2)(1+d^2)})$ can take all values in this region.

The goal is now to show that $h = f\left(\frac{rs+x}{r+1}\right)/f\left(\frac{rs+y}{r+1}\right)$, where $1 \geq x \geq y \geq -1$, is increasing in r. This will imply that the optimal value for c and d is 0, which further implies that $a = -w_q$, $b = -w_u$ for the lemma.

We first show that h is quasi-concave in s, so we may limit ourselves to $s = \pm 1$. Note that $\log h = \log f\left(\frac{rs+x}{r+1}\right) - \log f\left(\frac{rs+y}{r+1}\right)$, and that $\frac{d^2}{ds^2}\log f\left(\frac{rs+x}{r+1}\right) = \left(\frac{r}{1+r}\right)^2 \frac{d^2}{dz^2}\log f(z)$ by the chain rule. Hence it follows from the assumptions that h is log-concave, which implies quasi-concavity as needed.

We now consider $s = \pm 1$ to be a constant. We need to show that $\frac{d}{dr}h \geq 0$. Calculating,

$$\frac{d}{dr}f\left(\frac{rs+x}{r+1}\right)/f\left(\frac{rs+y}{r+1}\right) = \frac{(s-x)f\left(\frac{rs+y}{r+1}\right)f'\left(\frac{rs+x}{r+1}\right) - (s-y)f\left(\frac{rs+x}{r+1}\right)f'\left(\frac{rs+y}{r+1}\right)}{(1+r)^2 f\left(\frac{rs+y}{r+1}\right)^2}.$$

Since $f \geq 0$ it suffices to show $\frac{d}{dx}(s-x)f'\left(\frac{rs+x}{r+1}\right)/f\left(\frac{rs+x}{r+1}\right) \geq 0$. If we substitute $z = \frac{rs+x}{r+1}$, $z \in [-1,1]$, we can write the requirement as $\frac{d}{dz}(s-z)f'(z)/f(z) \geq 0$ or $\frac{d}{dz}\left((\pm 1 - z)\frac{d}{dz}\log f(z)\right) \geq 0$. □

4.2 A MinHash Dominating Family

Consider the classical MinHash scheme: A permutation $h : [d] \to [d]$ is sampled at random, and $y \subseteq \{0,1\}^d$ is placed in bucket $i \in [m]$ if $h(i) \in y$ and $\forall_{j<i} h(j) \notin y$. The probability for a collision between two sets q, y is then $|q \cap y|/(|q| + |y| - |q \cap y|)$ by a standard argument which implies an exponent of $\rho_{\text{mh}} = \log \frac{w_1}{w_q+w_u-w_1} \big/ \log \frac{w_2}{w_q+w_u-w_2}$.

Now consider building multiple independent such MinHash tables, but *keeping only the kth bucket in each one.* That gives a Locality Sensitive Filter family, which we will analyse in this section.

The Locality Sensitive Filter approach to similarity search is an extension by Becker et al. [16] to the Locality Sensitive Hashing framework by Indyk and Motwani [39]. We will use the following definition by Christiani [27], which we have slightly extended to support separate universes for query and data points:

Definition 4 (LSF)**.** *Let X and Y be some universes, let $S : X \times Y \to \mathbb{R}$ be a similarity function, and let $\mathcal{F}$ be a probability distribution over $\{(Q, U) \mid Q \subseteq X, U \subseteq Y\}$. We say that F is $(s_1, s_2, p_1, p_2, p_q, p_u)$-sensitive if for all points $x \in X, y \in Y$ and (Q, U) sampled randomly from $\mathcal{F}$ the following holds:*

1. *If $S(x, y) \geq s_1$ then $\Pr[x \in Q, y \in U] \geq p_1$.*
2. *If $S(x, y) \leq s_2$ then $\Pr[x \in Q, y \in U] \leq p_2$.*
3. *$\Pr[x \in Q] \leq p_q$ and $\Pr[x \in U] \leq p_u$.*

We refer to (Q, U) as a filter and to Q as the query filter and U as the update filter.

We first state the LSF-Symmetrization lemma implicit in [28]:

Lemma 4.3 (LSF-Symmetrization)**.** *Given a (p_1, p_2, p_q, p_u)-sensitive LSF-family, we can create a new family that is $(p_1 \frac{q}{p}, p_2 \frac{q}{p}, q, q)$-sensitive, where $p = \max\{p_q, p_u\}$ and $q = \min\{p_q, p_u\}$.*

For some values of p_1, p_2, p_q, p_u this will be better than simply taking $\max(\rho_u, \rho_q)$. In particular when symmetrization may reduce ρ_u by a lot by reducing its denominator.

Proof. W.l.o.g. assume $p_q \geq p_u$. When sampling a query filter, $Q \subseteq U$, pick a random number $\varrho \in [0, 1]$. If $\varrho > p_u/p_q$ use $\emptyset$ instead of Q. The new family then has $p_q' = p_q \cdot p_u/p_q$ and so on giving the lemma. □

Getting back to MinHash, we note that the "keeping only the ith bucket" family discussed above, corresponds sampling a permutation s of Y and taking the filter

$$U = \{x \mid s_i \in x \wedge s_0 \notin x \wedge \cdots \wedge s_{i-1} \notin x\}.$$

That is, the collection of x such that the first $i - 1$ values of s are not in x (since then x would have been put in that earlier bucket), but the ith element of s is in x (since otherwise x would have been put in a later bucket.)

Using just one of these families, combined with symmetrization, gives the ρ value:

$$\rho_i = \log \frac{(1 - w_q - w_u + w_1)^i w_1}{\max\{(1 - w_q)^i w_q,\, (1 - w_u)^i w_u\}} \bigg/ \log \frac{(1 - w_q - w_u + w_2)^i w_2}{\max\{(1 - w_q)^i w_q,\, (1 - w_u)^i w_u\}}.$$

This scheme is a generalization of Chosen Path, since taking $i = 0$ recovers exactly that algorithm. However, as we increase i, we see that the weight gradually shifts from the *present* elements (symbolized by w_1, w_2, w_q and w_u) to the *absent* elements (symbolized by $(1-w_q-w_u+w_q)$, etc.).

We will now show that for a given set of (w_q, w_u, w_1, w_2) there is always an optimal i which is better than using all of the i, which is what MinHash does. The exact goal is to show

$$\rho_{\mathrm{mh}} = \log \frac{w_1}{w_q + w_u - w_1} \Big/ \log \frac{w_2}{w_q + w_u - w_2} \geq \min_{i \geq 0} \rho_i.$$

For this we show the following lemma, which intuitively says that it is never advantageous to combine multiple filter families:

Lemma 4.4. *The function* $f(x, y, z, t) = \log(\max\{x, y\}/z)/\log(\max\{x, y\}/t)$*, defined for* $\min\{x, y\} \geq z \geq t > 0$*, is quasi-concave.*

This means in particular that

$$\frac{\log(\max\{x+x', y+y'\}/(z+z'))}{\log(\max\{x+x', y+y'\}/(t+t'))} \geq \min\left\{\frac{\log(\max\{x, y\}/z)}{\log(\max\{x, y\}/t)}, \frac{\log(\max\{x', y'\}/z')}{\log(\max\{x', y'\}/t')}\right\},$$

when the variables are in the range of the lemma.

Proof. We need to show that the set

$$\{(x, y, z, t) : \log(\max\{x, y\}/z)/\log(\max\{x, y\}/t) \geq \alpha\} = \{(x, y, z, t) : \max\{x, y\}^{1-\alpha} t^{\alpha} \geq z\}$$

is convex for all $\alpha \in [0, 1]$ (since $z \geq t$ so $f(x, y, z, t) \in [0, 1]$). This would follow if $g(x, y, t) = \max\{x, y\}^{1-\alpha} t^{\alpha}$ would be quasi-concave itself, and the eigenvalues of the Hessian of g are exactly 0, 0 and $-(1-\alpha)\alpha t^{\alpha-2} \max\{x, y\}^{-\alpha-1} \left(\max\{x, y\}^2 + t^2\right)$ so g is even concave! □

We can then show that MinHash is always dominated by one of the filters described, as

$$\rho_{\mathrm{mh}} = \frac{\log \frac{w_1}{w_q+w_u-w_1}}{\log \frac{w_2}{w_q+w_u-w_2}} = \frac{\log \frac{\sum_{i\geq 0}(1-w_q-w_u+w_1)^i w_1}{\max\{\sum_{i\geq 0}(1-w_q)^i w_q, \sum_{i\geq 0}(1-w_u)^i w_u\}}}{\log \frac{\sum_{i\geq 0}(1-w_q-w_u+w_2)^i w_2}{\max\{\sum_{i\geq 0}(1-w_q)^i w_q, \sum_{i\geq 0}(1-w_u)^i w_u\}}} \geq \min_{i\geq 0} \frac{\log \frac{(1-w_q-w_u+w_1)^i w_1}{\max\{(1-w_q)^i w_q,(1-w_u)^i w_u\}}}{\log \frac{(1-w_q-w_u+w_2)^i w_2}{\max\{(1-w_q)^i w_q,(1-w_u)^i w_u\}}},$$

where the right hand side is exactly the symmetrization of the "only bucket i" filters. By monotonicity of $(1-w_q)^i w_q$ and $(1-w_u)^i w_u$ we can further argue that it is even possible to limit ourselves to one of $i \in \{0, \infty, \log(w_q/w_u)/\log((1-w_q)/(1-w_u))\}$, where the first gives Chosen Path, the second gives Chosen Path on the complemented sets, and the last gives a balanced trade-off where $(1-w_q)^i w_q = (1-w_u)^i w_u$.

5 Conclusion and Open Problems

For a long time there was a debate [65] about why MinHash worked so well for sets, compared to other more general methods, like SimHash. It was a mystery why this method, so foreign to the frameworks of Spherical LSF and Chosen Path could still do so much better. For asymmetric problems like Subset Search, it was entirely open how far ρ could be reduced.

This paper finally solves the mystery of MinHash and unifies the ideas and frameworks of Euclidean and Set Similarity Search.

By showing that supermajorities indeed solve the general problem optimally, we not only unify and explain the performance of the previous literature, but also recover major performance improvements, space/time trade-offs, and the ability to solve Set Similarity Search for any similarity measure.

We propose the following open problems for future research:

LSH with polylog time When parametrized accordingly, we get a data structure with $e^{\tilde{O}(\sqrt{\log n})}$ query time and $n^{O(1)}$ space. Using Spherical LSH one can get similar runtime, though with a higher polynomial space usage. Employing a tighter analysis of our algorithm, the query time can be reduced to $e^{\tilde{O}((\log n)^{1/3})}$, which by comparison with we conjecture is tight for the approach. A major open question is whether one can get $\tilde{O}(1)$?

Data-dependent Data-dependent LSH is able to reduce approximate similarity search problems to the case where far points are as far away as had they been random. For (w_q, w_u, w_1, w_2)-GapSS this corresponds to the case $w_2 = w_q w_u$. This would finally give the "optimal" algorithm for GapSS without any "non-data-dependent" disclaimers.

Sparse, non-binary data Our lower bounds really hold for a much larger class of problems, including cosine similarity search on sparse data in $\mathbb{R}^d$. However, our upper bounds currently focus on binary data only. It would be interesting to generalize our algorithm to this and other types of data for which Supermajorities are also optimal.

Sketching We have shown that Supermajorities can shave large polynomial factors of space and query time in LSH. Can they be used to give similar gains in the field of sketching sets under various similarity measures? Can one expand the work of [55] and show optimality of some intersection sketching scheme?

5.1 Acknowledgements

We would like to thank Rasmus Pagh and Tobias Christiani on suggesting the problem discussed in this paper, as well as for reviews of previous manuscripts. Much of the initial work in the paper was done while the first author was visiting Eric Price at the University of Texas, and we would like to thank people there for encouragement and discussions on Boolean functions. Many people have helped reading versions of the manuscript, and we would like to thank Morgan Mingle, Morten Stöckel, John Kallaugher, Ninh Pham, Evangelos Kipouridis, Peter Rasmussen, Anders Aamand, Mikkel Thorup and everyone else who has given feedback. Finally, we really appreciate the comprehensive comments from the anonymous reviewers!

Thomas D. Ahle and Jakob B. T. Knudsen are partly supported by Thorup's Investigator Grant 16582, Basic Algorithms Research Copenhagen (BARC), from the VILLUM Foundation.

References

[1] Amir Abboud, Aviad Rubinstein, and Ryan Williams. Distributed pcp theorems for hardness of approximation in p. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 25–36. IEEE, 2017.

[2] Amirali Abdullah and Suresh Venkatasubramanian. A directed isoperimetric inequality with application to bregman near neighbor lower bounds. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 509–518, 2015.

[3] Parag Agrawal, Arvind Arasu, and Raghav Kaushik. On indexing error-tolerant set containment. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 927–938. ACM, 2010.

[4] Daniel Ahlberg, Erik Broman, Simon Griffiths, and Robert Morris. Noise sensitivity in continuum percolation. *Israel Journal of Mathematics*, 201(2):847–899, 2014.

[5] Thomas Dybdahl Ahle, Rasmus Pagh, Ilya Razenshteyn, and Francesco Silvestri. On the complexity of inner product similarity join. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 151–164. ACM, 2016.

[6] Josh Alman, Timothy M Chan, and Ryan Williams. Polynomial representations of threshold functions and algorithmic applications. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 467–476. IEEE, 2016.

[7] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 459–468. IEEE, 2006.

[8] Alexandr Andoni, Piotr Indyk, Huy L Nguyen, and Ilya Razenshteyn. Beyond locality-sensitive hashing. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1018–1028. Society for Industrial and Applied Mathematics, 2014.

[9] Alexandr Andoni, Thijs Laarhoven, Ilya Razenshteyn, and Erik Waingarten. Optimal hashing-based time-space trade-offs for approximate near neighbors. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 47–66. Society for Industrial and Applied Mathematics, 2017.

[10] Alexandr Andoni, Thijs Laarhoven, Ilya Razenshteyn, and Erik Waingarten. Optimal hashing-based time-space trade-offs for approximate near neighbors. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 47–66. SIAM, 2017.

[11] Alexandr Andoni and Ilya Razenshteyn. Optimal data-dependent hashing for approximate near neighbors. In *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing*, pages 793–801. ACM, 2015.

[12] Alexandr Andoni, Ilya Razenshteyn, and Negev Shekel Nosatzki. Lsh forest: Practical algorithms made theoretical. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 67–78. SIAM, 2017.

[13] Alexandr Andoni and Ilya Razensteyn. Tight lower bounds for data-dependent locality-sensitive hashing. In *32nd International Symposium on Computational Geometry (SoCG 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

[14] Roger C Baker, Glyn Harman, and János Pintz. The difference between consecutive primes, ii. *Proceedings of the London Mathematical Society*, 83(3):532–562, 2001.

[15] Mayank Bawa, Tyson Condie, and Prasanna Ganesan. Lsh forest: self-tuning indexes for similarity search. In *Proceedings of the 14th international conference on World Wide Web*, pages 651–660, 2005.

[16] Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 10–24. SIAM, 2016.

[17] William Beckner. Inequalities in fourier analysis. *Annals of Mathematics*, pages 159–182, 1975.

[18] John D Biggins. Martingale convergence in the branching random walk. *Journal of Applied Probability*, 14(1):25–37, 1977.

[19] Aline Bonami. 'E study of the fourier coefficients of the functions of $l\ p(g)$. In *Annals of the Fourier Institute*, volume 20, pages 335–402, 1970.

[20] Andrei Z Broder. On the resemblance and containment of documents. In *Compression and Complexity of Sequences 1997. Proceedings*, pages 21–29. IEEE, 1997.

[21] Andrei Z Broder, Steven C Glassman, Mark S Manasse, and Geoffrey Zweig. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, 29(8-13):1157–1166, 1997.

[22] Timothy M Chan. Orthogonal range searching in moderate dimensions: kd trees and range trees strike back. In *33rd International Symposium on Computational Geometry (SoCG 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[23] Moses Charikar, Piotr Indyk, and Rina Panigrahy. New algorithms for subset query, partial match, orthogonal range searching, and related problems. In *International Colloquium on Automata, Languages, and Programming*, pages 451–462. Springer, 2002.

[24] Moses S Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM Symposium on Theory of Computing*, pages 380–388. ACM, 2002.

[25] Lijie Chen and Ryan Williams. An equivalence class for orthogonal vectors. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 21–40. SIAM, 2019.

[26] Seung-Seok Choi, Sung-Hyuk Cha, and Charles C Tappert. A survey of binary similarity and distance measures. *Journal of Systemics, Cybernetics and Informatics*, 8(1):43–48, 2010.

[27] Tobias Christiani. A framework for similarity search with space-time tradeoffs using locality-sensitive filtering. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 31–46. SIAM, 2017.

[28] Tobias Christiani and Rasmus Pagh. Set similarity search beyond minhash. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 1094–1107, 2017. URL: `https://doi.org/10.1145/3055399.3055443`, `doi:10.1145/3055399.3055443`.

[29] Tobias Christiani, Rasmus Pagh, and Mikkel Thorup. Confirmation sampling for exact nearest neighbor search. *arXiv preprint arXiv:1812.02603*, 2018.

[30] Edith Cohen and Haim Kaplan. Leveraging discarded samples for tighter estimation of multiple-set aggregates. *ACM SIGMETRICS Performance Evaluation Review*, 37(1):251–262, 2009.

[31] Harald Cramér. On the order of magnitude of the difference between consecutive prime numbers. *Acta Arithmetica*, 2:23–46, 1936.

[32] Ian H. Dinwoodie. Large deviations techniques and applications (amir dembo and ofer zeitouni). *SIAM Review*, 36(2):303–304, 1994. URL: `https://doi.org/10.1137/1036078`, `doi:10.1137/1036078`.

[33] Moshe Dubiner. Bucketing coding and information theory for the statistical high-dimensional nearest-neighbor problem. *IEEE Transactions on Information Theory*, 56(8):4166–4179, 2010.

[34] Raul Castro Fernandez, Jisoo Min, Demitri Nava, and Samuel Madden. Lazo: A cardinality-based method for coupled estimation of jaccard similarity and containment. In *ICDE*, 2019.

[35] Philippe Flajolet, Éric Fusy, Olivier Gandouet, and Frédéric Meunier. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. In *Discrete Mathematics and Theoretical Computer Science*, pages 137–156. Discrete Mathematics and Theoretical Computer Science, 2007.

[36] Ehud Friedgut. An information theoretic proof of a hypercontractive inequality. *Entropy*, (1/29), 2015.

[37] Ashish Goel and Pankaj Gupta. Small subset queries and bloom filters using ternary associative memories, with applications. *ACM SIGMETRICS Performance Evaluation Review*, 38(1):143–154, 2010.

[38] Sariel Har-Peled, Piotr Indyk, and Rajeev Motwani. Approximate nearest neighbor: Towards removing the curse of dimensionality. *Theory of computing*, 8(1):321–350, 2012.

[39] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM, 1998.

[40] Lianyin Jia, Lulu Zhang, Guoxian Yu, Jinguo You, Jiaman Ding, and Mengjuan Li. A survey on set similarity search and join. *International Journal of Performability Engineering*, 14(2), 2018.

[41] Michael Kapralov. Smooth tradeoffs between insert and query complexity in nearest neighbor search. In *Proceedings of the 34th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 329–342. ACM, 2015.

[42] Michael Kapralov and Rina Panigrahy. Nns lower bounds via metric expansion for l_∞ and emd. In *International Colloquium on Automata, Languages, and Programming*, pages 545–556. Springer, 2012.

[43] Peter Keevash, Noam Lifshitz, Eoin Long, and Dor Minzer. Hypercontractivity for global functions and sharp thresholds. *arXiv preprint arXiv:1906.05568*, 2019.

[44] Thijs Laarhoven. Tradeoffs for nearest neighbors on the sphere. *arXiv preprint arXiv:1511.07527*, 2015.

[45] Noam Lifshitz. Hypergraph removal lemmas via robust sharp threshold theorems. *arXiv preprint arXiv:1804.00328*, 2018.

[46] Qin Lv, William Josephson, Zhe Wang, Moses Charikar, and Kai Li. Multi-probe lsh: efficient indexing for high-dimensional similarity search. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, pages 950–961. VLDB Endowment, 2007.

[47] Samuel McCauley, Jesper W Mikkelsen, and Rasmus Pagh. Set similarity search for skewed data. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 63–74, 2018.

[48] Sergey Melnik and Hector Garcia-Molina. Adaptive algorithms for set containment joins. *ACM Transactions on Database Systems (TODS)*, 28(1):56–99, 2003.

[49] Rajeev Motwani, Assaf Naor, and Rina Panigrahi. Lower bounds on locality sensitive hashing. In *Proceedings of the twenty-second annual symposium on Computational geometry*, pages 154–157. ACM, 2006.

[50] Chandra Nair. Equivalent formulations of hypercontractivity using information measures. In *International Zurich Seminar on Communications*, page 42, 2014.

[51] Behnam Neyshabur and Nathan Srebro. On symmetric and asymmetric lshs for inner product search. In *International Conference on Machine Learning*, pages 1926–1934, 2015.

[52] Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

[53] Krzysztof Oleszkiewicz. On a nonsymmetric version of the khinchine-kahane inequality. In *Stochastic inequalities and applications*, pages 157–168. Springer, 2003.

[54] Ryan O'Donnell, Yi Wu, and Yuan Zhou. Optimal lower bounds for locality-sensitive hashing (except when q is tiny). *ACM Transactions on Computation Theory (TOCT)*, 6(1):5, 2014.

[55] Rasmus Pagh, Morten Stöckel, and David P Woodruff. Is min-wise hashing optimal for summarizing set intersection? In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 109–120. ACM, 2014.

[56] Rina Panigrahy. Entropy based nearest neighbor search in high dimensions. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 1186–1195. Society for Industrial and Applied Mathematics, 2006.

[57] Rina Panigrahy, Kunal Talwar, and Udi Wieder. A geometric approach to lower bounds for approximate near-neighbor search and partial match. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 414–423. IEEE, 2008.

[58] Rina Panigrahy, Kunal Talwar, and Udi Wieder. Lower bounds on near neighbor search via metric expansion. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 805–814. IEEE, 2010.

[59] Yury Polyanskiy and Yihong Wu. Lecture notes on information theory. *Lecture Notes for ECE563 (UIUC) and*, 6(2012-2016):7, 2014.

[60] Karthikeyan Ramasamy, Jignesh M. Patel, Jeffrey F. Naughton, and Raghav Kaushik. Set containment joins: The good, the bad and the ugly. In *VLDB*, 2000.

[61] Cyrus Rashtchian, Aneesh Sharma, and David P Woodruff. Lsf-join: Locality sensitive filtering for distributed all-pairs set similarity under skew. *arXiv preprint arXiv:2003.02972*, 2020.

[62] Ronald L Rivest. Partial-match retrieval algorithms. *SIAM Journal on Computing*, 5(1):19–50, 1976.

[63] Zhan Shi. *Branching random walks.* Springer, 2015.

[64] Anshumali Shrivastava and Ping Li. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). In *Advances in Neural Information Processing Systems*, pages 2321–2329, 2014.

[65] Anshumali Shrivastava and Ping Li. In defense of minhash over simhash. In *Artificial Intelligence and Statistics*, pages 886–894, 2014.

[66] Kengo Terasawa and Yuzuru Tanaka. Spherical lsh for approximate nearest neighbor search on unit hypersphere. In *Workshop on Algorithms and Data Structures*, pages 27–38. Springer, 2007.

[67] Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theoretical Computer Science*, 348(2):357–365, 2005.

[68] Paweł Wolff. Hypercontractivity of simple random variables. *Studia Mathematica*, 3(180):219–236, 2007.

[69] Xiao Yan, Jinfeng Li, Xinyan Dai, Hongzhi Chen, and James Cheng. Norm-ranging lsh for maximum inner product search. In *Advances in Neural Information Processing Systems*, pages 2956–2965, 2018.

[70] Yong Zhang, Xiuxing Li, Jin Wang, Ying Zhang, Chunxiao Xing, and Xiaojie Yuan. An efficient framework for exact set similarity search using tree structure indexes. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pages 759–770. IEEE, 2017.

A Proof of Lemma 3.3

This proof in this section mostly follows [9], with a few changes to work with separate spaces Q and U.

Lemma 3.3. *Let Q and U be some spaces and $\mathcal{P}_{QU}$ a probability distribution on $Q \times U$. Consider any list-of-points data structure for $\mathcal{P}_{QU}$-random instances of n points, which uses expected space $n^{1+\rho_u}$, has expected query time $n^{\rho_q - o_n(1)}$, and succeeds with probability at least 0.99. Let $r, s \in [1, \infty]$ satisfy*

$$\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{QU}}[f(X)g(Y)] \le \|f(X)\|_{L_r(\mathcal{P}_Q)} \|f(Y)\|_{L_s(\mathcal{P}_U)} ,$$

for all functions $f : Q \to \mathbb{R}$ and $g : U \to \mathbb{R}$. Then

$$\frac{1}{r}\rho_q + \frac{1}{r'}\rho_u \ge \frac{1}{r} + \frac{1}{s} - 1 ,$$

where $r' = \frac{r}{r-1}$ is the convex conjugate of r.

Proof. Fix a data structure D, where A_i specifies which dataset points are placed in L_i. Additionally, we define $B_i = \{v \mid i \in I(v)\}$ to the set of query points which scan L_i. We sample a random dataset point u and then a random query point v from the neighborhood of u. Let

$$\gamma_i = \Pr\left[v \in B_i \mid u \in A_i\right]$$

represent the probability that query v scans the list L_i conditioned on u being in L_i. The query time for D is given by the following expression

$$\begin{aligned} T &= \sum_{i \in [m]} [v \in B_i] \left(1 + \sum_{j \in [n]} [u_j \in A_i]\right) \\ \mathrm{E}\left[T\right] &= \sum_{i \in [m]} \Pr\left[v \in B_i\right] + \sum_{i \in [m]} \gamma_i \Pr\left[u \in A_i\right] + (n-1) \sum_{i \in [m]} \Pr\left[u \in A_i\right] \Pr\left[v \in B_i\right] \ . \end{aligned}$$

We want to lower bound $\Pr\left[v \in B_i\right]$, so let $1 \leq r, s$ be any values such that $\mathcal{P}_{QU}$ is (r,s)-hypercontractive. We then get that

$$\begin{aligned} \gamma_i \Pr\left[u \in A_i\right] &= \Pr\left[u \in A_i \wedge v \in B_i\right] \\ &= \mathrm{E}\left[[u \in A_i][v \in B_i]\right] \\ &\leq \left\|[u \in A_i]\right\|_{L_s(p_1)} \left\|[v \in B_i]\right\|_{L_r(p_2)} \\ &= \Pr\left[u \in A_i\right]^{1/s} \Pr\left[v \in B_i\right]^{1/r} \end{aligned}$$

Hence we get that $\Pr\left[v \in B_i\right] \geq \gamma_i^r \Pr\left[u \in A_i\right]^{r/s'}$. We define $\tau_i = \Pr\left[u \in A_i\right]$ and get that

$$\mathrm{E}\left[T\right] \geq \sum_{i \in [m]} \gamma_i^r \tau_i^{r/s'} + \sum_{i \in [m]} \gamma_i \tau_i + (n-1) \sum_{i \in [m]} \gamma_i^r \tau_i^{1+r/s'} \ .$$

Since the data structure succeeds with probability γ we have that

$$\sum_{i \in [m]} \tau_i \gamma_i \geq \Pr\left[\exists i \in [m] : v \in B_i, u \in A_i\right] = \gamma \ .$$

Since D uses at most S space we have that

$$m + \sum_{i \in [m]} |A_i| \leq S \ \Rightarrow \ \sum_{i \in [m]} \tau_i \leq \frac{S}{n} \ .$$

We then get that we want to minimize

$$\mathrm{E}\left[T\right] \geq \sum_{i \in [m]} \gamma_i^r \tau_i^{r/s'} + \sum_{i \in [m]} \gamma_i \tau_i + (n-1) \sum_{i \in [m]} \gamma_i^r \tau_i^{1+r/s'} \geq \sum_{i \in [m]} \gamma_i^r \tau_i^r (\tau_i^{-r/s} + (n-1) \tau_i^{1-r/s}) \ ,$$

given the constraints

$$\begin{aligned} \sum_{i \in [m]} \tau_i \gamma_i &\geq \gamma \\ \sum_{i \in [m]} \tau_i &\leq \frac{S}{n} \ . \end{aligned}$$

First we fix $(\tau_i)_{i\in[m]}$ and minimize the function with respect to $(\gamma_i)_{i\in[m]}$. Using Lagrange multipliers this is equivalent to minimizing the function

$$f((\gamma_i)_{i\in[m]},\lambda,\nu)=\sum_{i\in[m]}\gamma_i^r\tau_i^r(\tau_i^{-r/s}+(n-1)\tau_i^{1-r/s})-\lambda(\sum_{i\in[m]}\tau_i\gamma_i-\gamma-\nu^2)$$

We find the critical points $\nabla f=0$:

$$\begin{aligned} r\gamma_i^{r-1}\tau_i^r(\tau_i^{-r/s}+(n-1)\tau_i^{1-r/s}) &= \lambda\tau_i \\ \sum_{i\in[m]}\tau_i\gamma_i &= \gamma+\nu^2 \\ 2\lambda\nu &= 0 \end{aligned}$$

for all $i\in[m]$. We note that since $\gamma>0$ then $\lambda>0$ and hence $\nu=0$. The first inequality can be rewritten as

$$\begin{aligned} \gamma_i^{r-1}\tau_i^{r-1} &= \frac{\lambda}{r(\tau_i^{-r/s}+(n-1)\tau_i^{1-r/s})} && \Leftrightarrow \\ \gamma_i\tau_i &= \left(\frac{\lambda}{r(\tau_i^{-r/s}+(n-1)\tau_i^{1-r/s})}\right)^{r'/r} \end{aligned}$$

Combining this with $\sum_{i\in[m]}\tau_i\gamma_i=\gamma$ give us that

$$\begin{aligned} \sum_{i\in[m]}\left(\frac{\lambda}{r(\tau_i^{-r/s}+(n-1)\tau_i^{1-r/s})}\right)^{r'/r} &= \gamma && \Leftrightarrow \\ \lambda^{r'/r} &= \frac{\gamma}{\sum_{i\in[m]}\left(\frac{1}{r(\tau_i^{-r/s}+(n-1)\tau_i^{1-r/s})}\right)^{r'/r}} \end{aligned}$$

We define $t_i=\left(\frac{1}{\tau_i^{-r/s}+(n-1)\tau_i^{1-r/s}}\right)^{r'/r}$ and get that

$$\begin{aligned} \gamma_i\tau_i &= \gamma\frac{t_i}{\sum_{i\in[m]}t_i} && \Leftrightarrow \\ \gamma_i^r\tau_i^r &= \gamma^r\frac{t_i^r}{(\sum_{i\in[m]}t_i)^r} \end{aligned}$$

We then get that our original function becomes

$$\gamma^r\sum_{i\in[m]}\frac{t_i^r}{(\sum_{i\in[m]}t_i)^r}t_i^{-r/r'}=\gamma^r\sum_{i\in[m]}\frac{t_i}{(\sum_{i\in[m]}t_i)^r}=\gamma^r(\sum_{i\in[m]}t_i)^{-(r-1)}=\gamma^r(\sum_{i\in[m]}t_i)^{-r/r'}$$

So we want to maximize

$$\sum_{i\in[m]}t_i=\sum_{i\in[m]}\left(\frac{1}{\tau_i^{-r/s}+(n-1)\tau_i^{1-r/s}}\right)^{r'/r}=\sum_{i\in[m]}\frac{\tau_i^{r'/s}}{(1+(n-1)\tau_i)^{r'/r}}$$

We now consider two different cases.

Case 1. $r > s$**.** We know that $\tau_i \leq 1$ so we get that

$$\sum_{i\in[m]} \frac{\tau_i^{r'/s}}{(1+(n-1)\tau_i)^{r'/r}} \leq \sum_{i\in[m]} \frac{\tau_i^{r'(1/s-1/r)}}{n^{r'/r}}$$

Since $r'(1/s - 1/r) > 0$ then we can use the power-mean inequality to get

$$\begin{aligned}
\sum_{i\in[m]} \frac{\tau_i^{r'(1/s-1/r)}}{n^{r'/r}} &\leq \frac{m}{n^{r'/r}} \left(\frac{\sum_{i\in[m]}\tau_i}{m}\right)^{r'(1/s-1/r)} \\
&\leq \frac{m^{r'-r'/s}}{n^{r'/r}} \left(\frac{S}{n}\right)^{r'(1/s-1/r)} \\
&= \frac{m^{r'/s'}}{n^{r'/s}} S^{r'(1/s-1/r)} \\
&\leq \frac{S^{r'/s'+r'(1/s-1/r)}}{n^{r'/s}} \\
&= \frac{S}{n^{r'/s}}
\end{aligned}$$

where we have used that $\max\left\{m, n\sum_{i\in[m]}\tau_i\right\} \leq S$.

Case 2. $r \leq s$ We find the derivatives

$$\begin{aligned}
&\frac{\frac{r'}{s} t_i^{r'/s-1}(1+(n-1)\tau_i)^{r'/r} - (n-1)\frac{r'}{r}(1+(n-1)\tau_i)^{r'/r-1} t_i^{r'/s}}{(1+(n-1)\tau_i)^{2r'/r}} \\
&= \frac{r' t_i^{r'/s-1}}{(1+(n-1)\tau_i)^{r'/r+1}} \left(\frac{1}{s}(1+(n-1)\tau_i) - (n-1)\frac{1}{r}\tau_i\right)
\end{aligned}$$

Case 2.1. $r < s$ We note that the function is maximized when we set $\tau_i = \frac{\frac{1}{s}}{(n-1)(\frac{1}{r}-\frac{1}{s})} = \frac{r}{(n-1)(s-r)}$. This give us that

$$\sum_{i\in[m]} \frac{\tau_i^{r'/s}}{(1+(n-1)\tau_i)^{r'/r}} \leq m \frac{\left(\frac{r}{(n-1)(s-r)}\right)^{r'/s}}{\left(1+\frac{r}{s-r}\right)^{r'/r}} \leq \frac{S}{n^{r'/s}} \frac{\left(\frac{2r}{s-r}\right)^{r'/s}}{\left(\frac{s}{s-r}\right)^{r'/r}}$$

where we have used that $m \leq S$ and $n \geq 2$.

$$m\frac{r}{(n-1)(s-r)} \leq \frac{S}{n} \Rightarrow m \leq S\frac{(n-1)(s-r)}{nr}$$

Case 2.2. $r = s$ We note that the function is increasing in τ_i so it is maximized when $\tau_i = 1$. Then we get that

$$\sum_{i\in[m]} \frac{\tau_i^{r'/s}}{(1+(n-1)\tau_i)^{r'/r}} \leq \frac{m}{n^{r'/r}} = \frac{S}{n^{r'/r}} = \frac{S}{n^{r'/s}}$$

where we have used that $m \leq S$ and $r = s$.

From this we note that if we set $K = \max\left\{1, \frac{\left(\frac{2r}{s-r}\right)^{r'/s}}{\left(\frac{s}{s-r}\right)^{r'/r}}\right\}$ then $\sum_{i\in[m]} \frac{\tau_i^{r'/s}}{(1+(n-1)\tau_i)^{r'/r}} \leq \frac{S}{n^{r'/s}}K$. Now we can give the final lower bound on $\mathrm{E}\left[T\right]$:

$$\mathrm{E}\left[T\right] \geq \gamma^r (\sum_{i\in[m]} t_i)^{-r/r'} \geq \gamma^r \left(\frac{S}{n^{r'/s}}K\right)^{-r/r'} = \gamma^r K^{-r/r'} S^{-r/r'} n^{r/s}$$

From this we get the result we want

$$\rho_q \geq -\frac{r}{r'}(1+\rho_u) + \frac{r}{s} - o_n(1) \Leftrightarrow$$
$$\frac{1}{r}\rho_q + \frac{1}{r'}\rho_u \geq \frac{1}{s} - \frac{1}{r'} - o_n(1)$$

□