Speeding Up Chemical Database Searches Using a Proximity Filter Based on the Logical Exclusive OR

Pierre Baldi,*[,†,‡] Daniel S. Hirschberg,[†] and Ramzi J. Nasr[†]

Department of Computer Science, Institute for Genomics and Bioinformatics, School of Information and Computer Sciences, and Department of Biological Chemistry, University of California, Irvine, Irvine, California 92697-3435

Received March 2, 2008

In many large chemoinformatics database systems, molecules are represented by long binary fingerprint vectors whose components record the presence or absence in the molecular graphs of particular functional groups or combinatorial features, such as labeled paths or labeled trees. To speed up database searches, we propose to store with each fingerprint a small header vector containing primarily the result of applying the logical exclusive OR (XOR) operator to the fingerprint vector after modulo wrapping to a smaller number of bits, such as 128 bits. From the XOR headers of two molecules, tight bounds on the intersection and union of their fingerprint vectors can be rapidly obtained, yielding tight bounds on derived similarity measures, such as the Tanimoto measure. During a database search, every time these bounds are unfavorable, the corresponding molecule can be rapidly discarded with no need for further inspection. We derive probabilistic models that allow us to estimate precisely the behavior of the XOR headers and the level of pruning under different conditions in terms of similarity threshold and fingerprint density. These theoretical results are corroborated by experimental results on a large set of molecules. For a Tanimoto threshold of 0.5 (respectively 0.9), this approach requires searching less than 50% (respectively 10%) of the database, leading to typical search speedups of 2 to 3 times over the previous state-of-the-art.

1. INTRODUCTION

Modern chemoinformatics systems are undergoing two related expansions in terms of data and representations. In terms of data expansion, current databases of small molecules such as PubChem and ChemDB[1,2] contain many millions of compounds and continue to grow in size as new compounds are discovered or synthesized. There is no end in sight to this process and the drive toward searching and understanding the even larger chemical space of virtual compounds. In part to keep up with the data expansion, there has been an expansion in the representations, in particular, in the number of molecular descriptors that are associated with each molecule in the databases. In early systems, typically, a few dozen descriptors were associated primarily with the presence or absence of functional or structural groups derived from human chemical expertise. In modern systems, the number of descriptors is much larger and can vary in range from 10^3 to 10^6, with the emphasis shifting progressively toward the higher end of this spectrum, going from human-derived to machine-derived descriptors. The machine-derived, or spectral, descriptors are typically obtained in a combinatorial way, by indexing all of the possible labeled subgraphs of a given kind (e.g., paths, trees) and size (e.g., depth up to 3) of the molecular graphs, yielding the well-known fingerprint vector representations.[3–9]

The somewhat parallel expansions in data and representations create new opportunities and challenges and require the continued development of methods to store and search molecules in chemical databases in ways that can scale up with these expansions. One strategy[10] to speed up database searches based on molecular similarity is essentially a pruning strategy with two basic components. First, a small binary signature vector is extracted from each fingerprint and stored in the database together with the basic fingerprint vector. A signature is a relatively short value/vector associated with a large item such that it is unlikely, but not impossible, that different items have the same signature. In the literature, signatures have been used as an equality filter (with one-sided error). Also, signatures have been used in Bloom filters, to exclude items not in a subset (also with one-sided error).[11] Here, we present a new type of signature that can be used as a proximity filter. Second, during a database search, the signatures of the query molecule and of each molecule in the database can be used to rapidly compute an upper bound on the similarity between the query and each molecule. This upper bound provides criteria for rapidly discarding from the search molecules that are "not good enough", effectively pruning the search space and focusing computational resources on a small fraction of the database that may contain suitable candidates, without any loss of retrieval power. Key to this proximity-filter strategy is the kind of signature used to derive the bounds. The signature should be small and easy to derive and store. More importantly, it should yield similarity upper bounds that can be computed rapidly, in order to sift rapidly through a large number of molecules, and that are tight, in order to prune a large number of compounds. Only the most simple signature was used in ref 10 to derive the upper bounds, consisting of

* Corresponding author e-mail: pfbaldi@ics.uci.edu.
† Department of Computer Science.
‡ Department of Biological Chemistry.

10.1021/ci800076s CCC: $40.75
Published on Web 07/02/2008

a single number corresponding to the total number of 1-bits in the fingerprint vector. Even so, knowing the total number of 1-bits in the fingerprint yields simple bounds on standard similarity measures, such as the Tanimoto measure, which in turn can yield very significant amounts of pruning, considerably speeding up database searches.[10]

Here, we take this approach one step further by developing a new, more subtle, and complementary proximity filter, based on a signature derived using the exclusive-OR logical operator (XOR). The standard OR operator is used in several chemoinformatics systems, such as the Daylight, Avalon, and Unity systems, to lossly compress long fingerprints by application of the OR operator modulo n (e.g., 1024 bits) to derive short fingerprint representations. Here, we expand on this idea and use a different logical operator modulo of some short length n for purposes related to, but different from, compression. There is a total of 16 binary Boolean operators Ts, only eight of which are symmetric in the sense that $1\ T\ 0 = 0\ T\ 1$ to be consistent with the usual hypothesis of exchangeable fingerprint bits. From these eight operators, we can immediately remove the OR operator, since it has already been studied. The AND and FALSE (always 0) operators are also not suitable since they produce signatures that consist entirely, or mostly, of 0-bits and therefore are uninformative. The same is true of the TRUE operator (always 1) that produces constant representations containing only 1-bits. Of the remaining four operators, the negations of the OR and AND operators are not suitable for similar reasons, leaving only the XOR operator and its equivalent negation to be studied. Thus, by elimination, the XOR-derived signatures are the only ones worth considering. Furthermore, for a given compression length n, the XOR operator yields representations that are sparser than those obtained using the OR operator, which may be an advantage for very short signatures. While the OR-derived compressed representations are used to estimate the uncompressed similarity, in this paper, we show that the XOR-derived compact signatures can be used to compute an exact bound on the uncompressed similarity. We prove both analytically and empirically that the XOR-derived signature yields in general tighter upper bounds on the similarity measure, resulting in larger amounts of pruning for a computational cost that remains reasonable when properly managed.

2. BACKGROUND: FINGERPRINT REPRESENTATIONS, SIMILARITY MEASURES, AND DATA

2.1. Fingerprint Representations. We assume that each molecule **A** has an associated binary fingerprint vector $\vec{A} = (A_i)$ of length N. The binary component A_i records the presence or absence of a particular descriptor in molecule **A**. The nature of these descriptors is irrelevant for our purposes, but examples will be given in the Data section. N should be relatively large—in the applications considered here, N could be in the 10^4 to 10^6 range—but its exact value is not important for demonstrating the approach. It is possible to extend the approach to be presented here to the case of count fingerprints, where the number of occurrences of each descriptor is recorded, rather than its mere presence or absence. Here, we focus on binary fingerprints because these are the most widely used and also because, to the best of our knowledge, there is scant evidence that the information gain associated with count fingerprints is worth the corresponding additional computational costs. We let A denote the number of 1-bits in $\vec{A}$. More generally, given any two molecules **A** and **B** and any binary Boolean operator $*$, we let $A*B$ denote the number of 1-bits contained in the vector $\vec{A}*\vec{B}$, where $*$ is applied to each pair of components. For instance, $A \vee B = A \cup B$ denotes the number of 1-bits in the bitwise OR (union) of $\vec{A}$ and $\vec{B}$, and $A \wedge B = A \cap B$ denotes the number of 1-bits in the bitwise AND (intersection), with a slight abuse of notation since $\cup$ and $\cap$ are usually considered set operators. Of particular interest to this work is the XOR logical operator (exclusive-OR) denoted by $\oplus$.

2.2. Similarity Measures. Given two molecules **A** and **B**, we use the Tanimoto measure

$$S(A, B) = S(\vec{A}, \vec{B}) = (A \cap B)/(A \cup B) \tag{1}$$

to measure molecular similarity. We use the Tanimoto similarity because it is the most widely used measure. However, the methods to be described can be extended immediately to other similarity measures,[10,12] such as the Tversky similarity measure[13,14]

$$S_{\alpha\beta}(\vec{A}, \vec{B}) = \frac{A \cap B}{\alpha A + \beta B + (1 - \alpha - \beta)(A \cap B)} \tag{2}$$

(where the parameters α and β can be used to tune the search toward substructures or superstructures of the query molecule). This is because most other measures of similarity can also be expressed in terms of $A \cap B$ and $A \cup B$ (as well as obvious terms such as A, B, and N),[10] and the present approach relies on providing bounds for these individual components shared by most similarity measures found in the literature.

2.3. Data. Although the methods to be presented do not depend on the details of any particular implementation, in the supporting experiments we use data from ChemDB,[1,2] which contains on the order of 5 million compounds. Whenever necessary for computational expediency, we use random subsets of up to 100 000 molecules to compute relevant statistics. In this case, experiments are repeated using several random samples of 100 000 molecules to make sure that the results are not sensitive with respect to sampling variability. The simulation results are obtained using fingerprints of length $N = \sim 60\,000$ on the basis of shallow labeled trees[9] of a depth up to 2, also called "circular" or "extended connectivity" fingerprints in the literature (see also Hert et al.,[15] Bender et al.,[16] and Hassan et al.[17]). Circular substructures are fully explored labeled trees of a particular depth, rooted at a particular vertex. All of the circular substructures of a molecule can be trivially listed using $O(Nd)$ steps, where d denotes the maximum tree depth and N denotes the number of atoms in the molecule. For the labeling, each vertex is labeled by the corresponding element (C, N, O, etc.) and degree (1, 2, 3, etc.) of the corresponding atom, and each edge is labeled by its type (single, double, triple, aromatic, and amide). The degree of a vertex is the number of edges incident to that vertex or, equivalently, the number of atoms bonded to the corresponding atom. For example, propane would be labeled as $C1sC2sC1$ and ethene would be labeled as $C1dC1$.

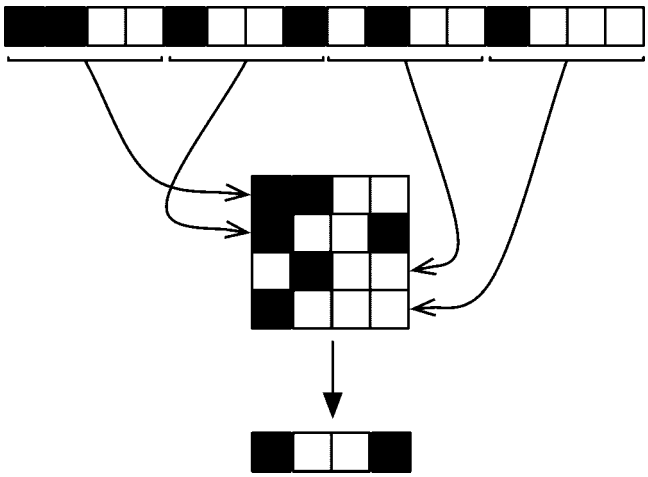

Figure 1. Illustration of the folding process with a binary vector of length $N = 16$ (1 1 0 0 1 0 0 1 0 1 0 0 1 0 0 0) folded into a binary vector of length $n = 4$ (1 0 0 1), modulo 4 using the XOR operator.

3. PREVIOUS APPROACH

In the basic version of the previous approach,[10] the value A is stored together with each fingerprint $\vec{A}$. Given two fingerprints $\vec{A}$ and $\vec{B}$, the Tanimoto measure can be bounded by writing

$$S(\vec{A},\vec{B}) = \frac{A \cap B}{A \cup B} \leq \frac{\min(A,B)}{\max(A,B)} = T(A,B) \qquad (3)$$

since $A \cap B \leq \min(A, B)$ and $A \cup B \geq \max(A, B)$. To see how this bound can be used to speed up molecular searches, consider a query molecule **A** and a similarity threshold $0 \leq t \leq 1$. If we are interested in retrieving only molecules that have a similarity greater than t to the query molecule, then any molecule **B** that satisfies $T(A, B) \leq t$ can be discarded from the search. Note that the bound $T(A, B)$ can be computed very rapidly from the additionally stored information (A and B) and does not depend on the details of the fingerprint vectors $\vec{A}$ and $\vec{B}$. In previous work,[10] we showed analytically and through simulations how this approach yields considerable savings in time and how it can be extended to other situations, including searches based on multiple-molecule queries, or searches aimed at retrieving the top K hits rather than the hits above some similarity threshold.

4. A NEW PROXIMITY FILTER: THE XOR APPROACH AND ITS SIMILARITY BOUNDS

We define a short fingerprint vector $\vec{a} = (a_i)$ of length n, where n is relatively small and typically a power of 2. In the simulations and in our implementation, we use $n = 128$. Vector $\vec{a}$ is derived from "folding" the long fingerprint vector $\vec{A}$ using the XOR operator applied modulo n ($\vec{a} = \oplus_n \vec{A}$). As a result, $a_i = 1$ if and only if the number of 1-bits contained in $\vec{A}$ at positions congruent with i modulo n is odd, and 0 otherwise. The diagram in Figure 1 illustrates the simple process of folding a binary fingerprint vector of size $N = 16$ into a fingerprint vector of size $n = 4$, using the XOR modulo operator.

Here, we propose to store for each molecule **A** in the database not only the fingerprint $\vec{A}$ but also additional information consisting of the shorter fingerprint $\vec{a}$, as well as the values a and A. The additional information can be viewed as a "header" preceding the vector $\vec{A}$, which will be used to derive useful bounds on similarity measures.

It is important to note that, in several chemoinformatics systems (e.g., Daylight, Avalon, and Unity), a similar folding approach is used in combination with the OR Boolean operator in order to compress long fingerprints into shorter fingerprints. In that lossy approach to compression, the length of the shorter fingerprints is typically 1024 or 2048, slightly above the value of n used here. This is related to a key difference in the last column of the truth table of the OR and XOR operators, namely, the XOR is a sparser operator: whenever the XOR gives a 1-bit, the OR operator gives also a 1-bit, but not vice versa. The OR-compressed representations are routinely used to estimate the similarity of the molecules, simply by computing the Tanimoto similarity of the compressed representations and using it as a proxy for the Tanimoto similarity of the uncompressed representations. We have shown[18] that this approach introduces a systematic bias in the estimation that can be corrected by deriving a better estimate of the uncompressed similarity from the OR-compressed representations. But, even this correction does not lead to any bounds or any pruning. Here, we are not proposing to use the XOR operator for lossy compression and to estimate the similarity values. Rather, and this is the key point, the short XOR fingerprints can be used to rapidly derive exact bounds on the similarity values that are, in general, tighter than those of eq 3.

To see this, first note that unions and intersections that enter into the similarity measure can be expressed using the XOR operator in the forms

$$A \cap B = \frac{1}{2}[A + B - (A \oplus B)] \qquad (4)$$

and

$$A \cup B = \frac{1}{2}[A + B + (A \oplus B)] \qquad (5)$$

The crucial property of the XOR folding is that

$$A \oplus B \geq a \oplus b \geq |a - b| \qquad (6)$$

As a result, we can bound the intersection and the union as follows:

$$A \cap B \leq \frac{1}{2}[A + B - (a \oplus b)] \leq \frac{1}{2}[A + B - |a - b|] \qquad (7)$$

and

$$A \cup B \geq \frac{1}{2}[A + B + (a \oplus b)] \geq \frac{1}{2}[A + B + |a - b|] \qquad (8)$$

Finally, we can combine these inequalities, to bound the Tanimoto similarity by

$$S(\vec{A},\vec{B}) = \frac{A \cap B}{A \cup B} \leq T_{\oplus}(A,B) = \frac{A + B - (a \oplus b)}{A + B + (a \oplus b)} \leq T_{ab}(A,B) = \frac{A + B - |a - b|}{A + B + |a - b|} \qquad (9)$$

For example, consider two molecules satisfying $A = 60$, $B = 50$, $A \cap B = 46$, $A \oplus B = 18$, and $a \oplus b = 16$. Then, the Tanimoto similarity is $S(\vec{A}, \vec{B}) = 46/74 = 0.71$. The simple bound from our previous work is $T(A, B) = 50/60 = 0.83$. The new tighter bound involving XOR is $T_{\oplus}(A, B) = 0.74$. Note that the XOR bound is not tighter than the previous bound in an absolute sense. Examples can be constructed

where, by chance, the XOR bound is weaker. For instance, if $A = 51$ and $B = 109$ and $a \oplus b = 40$, then the bit bound (eq 3) is 0.47, whereas the XOR bound is 0.6. However, as we shall see, from theoretical analysis and simulations, the XOR bound is much better statistically. Furthermore, both bounds can be used in succession to prune the database, the XOR bound being used last, so this problem never occurs in our implementation.

Given a query $\vec{A}$, if a molecule $\vec{B}$ satisfies

$$\frac{A + B - a \oplus b}{A + B + a \oplus b} \leq t \tag{10}$$

it should be rejected. Equivalently, it should be rejected if

$$A + B \leq \frac{1 + t}{1 - t}(a \oplus b) \tag{11}$$

or

$$\frac{1 - t}{1 + t}(A + B) \leq a \oplus b \tag{12}$$

The choice of n must achieve a tradeoff, which is in part implementation-dependent: larger values of n result in tighter bounds and greater amounts of pruning. Smaller values of n result in faster computation and smaller storage/memory requirements. We have experimented with $n = 32$, 64, 128, 256, and 512 to take advantage also of byte arithmetic. Empirically, and consistently with the theory to be reported, we find that $n = 128$ achieves an optimal compromise in the current computational environment. Thus, all simulation results are reported with $n = 128$.

The key questions that remain to be addressed are as follows: (1) How much pruning can be derived from the XOR bound, and how does it compare to the previous results? (2) Assuming that there is additional pruning resulting from the XOR bound, is it worth the additional computation cost? And if so, should the various pruning algorithms be combined, and how? These are primarily empirical questions that can be addressed by relatively simple simulations. In the next section, we show that these questions can also be treated analytically to some extent.

5. THEORETICAL RESULTS: STATISTICAL ANALYSIS OF XOR FOLDING AND PRUNING

5.1. Exchangeability. To begin with, consider the task of estimating a from A, in particular, the mean and variance of a given A. We can view $A = (A_i)$ as a vector random variable and the folding operation as a two-step operation where A balls (bits) are put into n boxes by modulo folding. We let Y_i denote the random variable associated with number of balls that end up in box i, for $i = 1,..., n$. We then apply the XOR operator and let X_i denote the corresponding binary random variable. Basically, $X_i = 1$ if Y_i is odd, and $X_i = 0$ otherwise. The fundamental statistical concept in this analysis, which can be viewed as a modeling assumption, is that the set of variables A_i is exchangeable, and so are the sets of variables Y_i and X_i. In simple terms, exchangeability means that any probability expression involving the A_i variables is invariant under any permutation of the indices. Exchangeability is a good model for large fingerprints where the descriptors are mapped more or less randomly to fingerprint components using a hash function, which is the case in many chemoinformatics systems. Note that with these notations $a = X = \sum_{i=1}^{n} X_i$. We thus have

$$E(X) = \sum_{i=1}^{n} E(X_i) = nE(X_1) = nP(X_1 = 1) = n[1 - P(X_1 = 0)] \tag{13}$$

which gives

$$E(X) = n[P(Y_1 = 1) + P(Y_1 = 3)...] = n[1 - P(Y_1 = 0) - P(Y_1 = 2)...] \tag{14}$$

Here, only box 1 is used as a result of the exchangeability. For the variance of X, we have

$$\mathrm{Var}X = \sum_{i=1}^{n} \mathrm{Var}X_i + \sum_{i=j} \mathrm{Cov}(X_i, X_j) = n\mathrm{Var}X_1 + 2\binom{n}{2} \mathrm{Cov}(X_1, X_2) \tag{15}$$

which finally gives

$$\mathrm{Var}X = n\mathrm{Var}X_1 + 2\binom{n}{2}[E(X_1X_2) - E(X_1)E(X_2)] = n\mathrm{Var}X_1 + 2\binom{n}{2}[E(X_1X_2) - (E(X_1))^2] \tag{16}$$

For the individual variance terms, we have

$$\mathrm{Var}X_1 = P(X_1 = 1)(1 - P(X_1 = 1)) \tag{17}$$

To compute the covariance terms, starting from terms corresponding to small ball counts, the term $E[X_1X_2]$ can be expanded as

$$E[X_1X_2] = P(X_1 = 1, X_2 = 1) = P(Y_1 = 1, Y_2 = 1) + 2P(Y_1 = 1, Y_2 = 3) + P(Y_1 = 3, Y_2 = 3) + ... \tag{18}$$

where again we have used the exchangeability to group the symmetric contributions $P(Y_1 = 1, Y_2 = 3) + P(Y_1 = 3, Y_2 = 1)$ into a single term.

Similar considerations can be applied to the compressed XOR vector conditioned on both A and B. Suppose that A and B are fixed and independent. Let Y_i^a and Y_i^b be the random variables associated with the number of balls in box i, and let X_i^a and X_i^b be the corresponding random variables after applying the XOR operator. Finally, let $X_i^{a \oplus b} = X_i^a \oplus X_i^b$ and $X^{a \oplus b} = \sum_i X_i^{a \oplus b}$. Applying the linearity of the expectation, the exchangeability of the random variables, and the independence of $\vec{a}$ and $\vec{b}$, we have

$$E(X^{a \oplus b}) = nE(X_1^{a \oplus b}) = nP(X_1^{a \oplus b} = 1) = n[P(X_1^a = 1)\, P(X_1^b = 0) + P(X_1^a = 0)\, P(X_1^b = 1)] \tag{19}$$

The last term can be further expanded as above in terms of the values of Y_1^a and Y_1^b. For instance, a second-order expansion would give

$$P(X_1^{a \oplus b} = 1) \approx P(Y_1^a = 1)\, P(Y_1^b = 0) + P(Y_1^a = 0)\, P(Y_1^b = 1) + P(Y_1^a = 1)\, P(Y_1^b = 2) + P(Y_1^a = 2)\, P(Y_1^b = 1) \tag{20}$$

Because of the exchangeability, the variance of $X^{a \oplus b}$ can be written again in the same form as eq 16 with all of the X variables superscripted with $a \oplus b$. The variance term is given by

$$\mathrm{Var}X_1^{a \oplus b} = P(X_1^{a \oplus b} = 1)[1 - P(X_1^{a \oplus b} = 1)] \tag{21}$$

Finally, we need to compute the covariance term

$$E(X_1^{a\oplus b}X_2^{a\oplus b}) = P(X_1^{a\oplus b}=1, X_2^{a\oplus b}=1) = \sum_S P(Y_1^a=y_1^a, Y_1^b=y_1^b, Y_2^a=y_2^a, Y_2^b=y_2^b) \quad (22)$$

where the set S corresponds to all sets of integer values for $(y_1^a, y_1^b, y_2^a, y_2^b)$ with $0 \le y_1^a, y_2^a \le \inf(n, A)$, $0 \le y_1^b, y_2^b \le \inf(n, B)$, y_1^a and y_1^b of opposite parity, and y_2^a and y_2^b of opposite parity. Given the independence of $\vec{A}$ and $\vec{B}$, this yields

$$E(X_1^{a\oplus b}X_2^{a\oplus b}) = P(X_1^{a\oplus b}=1, X_2^{a\oplus b}=1) = \sum_S P(Y_1^a=y_1^a, Y_2^a=y_2^a)\, P(Y_1^b=y_1^b, Y_2^b=y_2^b) \quad (23)$$

For instance, the first-order expansion gives

$$\begin{aligned} E(X_1^{a\oplus b}X_2^{a\oplus b}) = P(X_1^{a\oplus b}=1, X_2^{a\oplus b}=1) = \\ \sum_S P(Y_1^a=0, Y_2^a=0)P(Y_1^b=1, Y_2^b=1) + \\ P(Y_1^a=0, Y_2^a=1)P(Y_1^b=1, Y_2^b=0) + \\ P(Y_1^a=1, Y_2^a=0)P(Y_1^b=0, Y_2^b=1) + \\ P(Y_1^a=1, Y_2^a=1)P(Y_1^b=0, Y_2^b=0) \end{aligned} \quad (24)$$

We now consider a simple approach for further estimating these quantities under a simple binomial model where the components of the long fingerprints, hence also of the short XOR fingerprints, are independent, so that the covariance terms are equal to zero. A more accurate, but more complicated, derivation which does not assume independence is given in the Appendix.

5.2. Binomial Model. In this section, we assume that the fingerprints are produced by a coin-flipping process. Given A, we assume that query fingerprint vectors $\vec{A}$ are produced using the binomial $\mathbf{B}(N, p_A)$ where $p_A = A/N$. These queries are used to search the fingerprints $\vec{B}$ in the background database, and these fingerprints are produced using the binomial $\mathbf{B}(N, p_B)$. Note that p_A and p_B are not necessarily equal, especially if the queries tend to have a skewed distribution. In any case, the typical framework one ought to consider is one where N is large and p_A and p_B are very small. Given $N = nM$, we let $M = 2L$ or $M = 2L + 1$, so that $L = \lfloor M/2 \rfloor$. In this case, we have

$$P(X_1=0) = P(Y_1 \text{ is even}) = \sum_{i=0}^{L} P(Y_1=2i) = \sum_{i=0}^{L} \binom{M}{2i} p^{2i}(1-p)^{M-2i} \quad (25)$$

and

$$P(X_1=1) = P(Y_1 \text{ is odd}) = \sum_{i=0}^{L} P(Y_1=2i+1) = \sum_{i=0}^{L} \binom{M}{2i+1} p^{2i+1}(1-p)^{M-2i-1} \quad (26)$$

Thus, with this binomial approximation, we can write

$$E(a|A) \approx E(a|p_A) = nP(X_1=1) = np_a \quad (27)$$

so that we can view the short XOR fingerprints of the queries as being produced by a binomial $\mathbf{B}(n, p_a)$ with

$$p_a = P(X_1=1) = P(X_1=1|N, n, A) \quad (28)$$

When the number of balls is typically small relative to the number of boxes, the low-order terms in the sums of eqs 25 and 26 dominate. In this case, taking for instance only the first two terms in the sums, we have the approximations

$$n[P(Y_1=1) + P(Y_1=3)] \le E(a|p_A) \le n[1 - P(Y_1=0) - P(Y_1=2)] \quad (29)$$

The variance of this new binomial is given by $np_a(1 - p_a) = nP(X_1 = 1)P(X_1 = 0)$. Again, by taking low-order terms which are dominant in our regime, this leads to the approximations

$$\mathrm{Var}(a|p_A) \approx n[P(Y_1=1) + P(Y_1=3)][1 - P(Y_1=1) - P(Y_1=3)] \quad (30)$$

or

$$\mathrm{Var}(a|p_A) \approx n[1 - P(Y_1=0) - P(Y_1=2)][P(Y_1=0) + P(Y_1=2)] \quad (31)$$

or

$$\mathrm{Var}(a|p_A) \approx n[P(Y_1=1) + P(Y_1=3)][P(Y_1=0) + P(Y_1=2)] \quad (32)$$

In any specific case, it is easy to see which expectation and variance approximations are better and what kinds of bounds can be derived. We expect these approaches to yield slight overestimates because, in reality, the variance includes covariance terms that are negative, as discussed in the next section.

Under the binomial assumptions, we have the simple estimate

$$E(a \cap b|p_A, p_B) \approx np_ap_b = np_{ab} \quad (33)$$

and

$$E(a\oplus b|p_A, p_B) \approx n(p_a + p_b - p_ap_b) = np_{a\oplus b} \quad (34)$$

with $p_{a\oplus b} = p_a + p_b - p_ap_b$. Thus, using eq 12 with the binomial approximation, given p_A and p_B, the probability of pruning rejection is given by the binomial tail

$$P\left(\frac{1-t}{1+t}(A+B) \le a\oplus b\right) \approx \sum_{i=f(A,B,t)}^{n} \binom{n}{i} p_{a\oplus b}^i (1-p_{a\oplus b})^{n-i} \quad (35)$$

with $f(A, B, t) = \lceil (1 - t)/(1+t)(A+B) \rceil$. The fraction of the database that needs to be searched is equal to

$$P\left(\frac{1-t}{1+t}(A+B) \ge a\oplus b\right) \approx \sum_{i=0}^{g(A,B,t)} \binom{n}{i} p_{a\oplus b}^i (1-p_{a\oplus b})^{n-i} \quad (36)$$

where $g(A, B, t) = \lfloor (1 - t)/(1 + t)(A+B) \rfloor$. Approximating the Binomial distribution with a Gaussian yields a Gaussian integral. Thus, the fraction $1 - f(A, B)$ of the database that needs to be searched is approximately given by

$$1 - f(A,B) \approx \int_{-\infty}^{\frac{1-t}{1+t}(A+B)} N(x;\mu,\sigma^2)\,dx = \int_{-\infty}^{\frac{1-t}{1+t}(A+B)} \frac{1}{\sqrt{2}\pi\sigma} e^{-\frac{1}{2}\frac{(x-np_{a\oplus b})^2}{\sigma^2}} dx \quad (37)$$

with $\mu = np_{a\oplus b}$ and $\sigma^2 = np_{a\oplus b}(1 - p_{a\oplus b})$. These expressions can be further integrated with respect to B to derive the average fraction $1 - f(A)$ of the database to be searched, given A. In previous work,[10] we have seen that a reasonable approximation to the distributions of B in a database such as ChemDB is a Normal distribution $\mathbf{N}(\mu_D, \sigma_D^2)$. Finally, these expressions can also be integrated over the queries A to yield the fraction $1 - f$ of the database to be searched, averaged over the queries. The

distribution over the queries can be taken to be normal $\mathbf{N}(\mu_Q, \sigma_Q^2)$ but can be different from the distribution over all molecules.[10] For instance, as t approaches 1, the sum in eq 36 contains only one term equal to $(1 - p_{a\oplus b})^n$, so that

$$1 - f(A) \approx \int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi}\sigma_D} e^{-\frac{1}{2}\frac{(B-\mu_B)^2}{\sigma_B^2}} (1 - p_{a\oplus b})^n \, dB \quad (38)$$

In the proper regime, for n being sufficiently large, this can be further approximated by using $(1 - p_{a\oplus b})^n \approx 1 - np_{a\oplus b}$.

5.3. Exact Model with Indistinguishable Balls. In the binomial approximation, starting from A, we assume that all of the bits are independent and fingerprints are derived using the binomial $\mathbf{B}(N, p_A)$. These fingerprints in general do not have exactly A 1-bits but include a certain amount of extra fluctuations. A more rigorous calculation can be obtained by assuming that, given A, fingerprints are drawn uniformly over all vectors of length N with exactly A 1-bits. In this model, the components are still exchangeable but not independent of each other: for instance, if a ball goes into box 1, only $A - 1$ balls are left to be distributed among the n boxes. To simplify the analysis, in this section, we assume that $M \geq A$ so that there cannot be any "overflow" from any box. In this case, eqs 13–18 are still true. What we need is to update the values of $P(Y_1 = k)$ under the fixed-size model.

The total number of ways of distributing the A bits among the $N = Mn$ slots is given by

$$\binom{N}{A} = \binom{Mn}{A} \quad (39)$$

The number of ways of having k balls associated with the first box is given by $\binom{M}{k}\binom{M(n-1)}{(A-k)}$, which gives the probability

$$P(Y_1 = k) = \frac{\binom{M}{k}\binom{M(n-1)}{(A-k)}}{\binom{N}{A}} \quad (40)$$

This probability is of course 0 for $k > M$ or $k > A$. Similarly, for pairwise terms needed to compute the covariances, we have

$$P(Y_1 = k, Y_2 = l) = \frac{\binom{M}{k}\binom{M}{l}\binom{M(n-2)}{(A-k-l)}}{\binom{N}{A}} \quad (41)$$

which again requires $k \leq \inf(A, M)$, $l \leq \inf(A, M)$, and $k + l \leq A$ to be nonzero. From here, we can proceed as described in the general case to derive exact expressions, or use the lower-order terms to derive approximations when the number of balls is relatively small compared to the number of boxes. Note that, from these equations, we can easily compute all of the quantities in eqs 19–24 for the XOR conditioned on the values of A and B. In particular, eq 40 is exactly what is needed to compute the covariance terms of eq 23.

For completeness, two other approximate ball models are described in the Appendix. These can be used for some approximations but in general are less accurate than the Exact model.

6. SIMULATION RESULTS

6.1. Assessment of the Statistical Models: Convergence. We investigate first the number of terms needed in the expansion of the mean (eq 14) and the variance (eq 18) for the two main models, the Binomial model and the Exact model. The answer to this question is provided in Figure 2 for the mean and Figure 3 for the variance, in the case of $n = 128$, which is used in all of the simulations.

Figure 2 shows that, as far as the convergence of the mean is concerned, the Binomial and Exact models behave similarly in terms of approximations. Taking only one term in the expansion of eq 14 gives essentially indistinguishable results from more elaborate expansions only for the lower range of values of A, up to 50 or so. Taking two or more terms in eq 14 gives results that are close to the correct value on the entire range of A, and virtually identical to the exact value when using three or four terms. In any case, for the expectation, all of the terms in the expansion of eq 14 can be taken into account for both the Binomial and the Exact models. Thus, in the rest of the paper, we use the exact values corresponding to the full expansion for the expectation values of the Binomial and Exact models. Note from the zoomed-in regions of Figure 2 that the expectation values for the two models are slightly different.

Figure 3 shows a similar behavior for the variance of the Binomial model. Because the variance of the Binomial model does not contain any covariance terms, it can be computed exactly using the full expansion of $P(X_1 = 1)$ associated with eq 14 and the related equations. The first-order expansion ($Y_1 = 1$) gives a reasonable approximation to the variance only for values of A up to 40 or so. Second-order expansion ($Y_1 = 1$ or $Y_1 = 3$) and beyond gives a good approximation over the entire range of A with results practically indistinguishable from the values obtained with the full expansion as soon as at least three terms are used in the expansion ($Y_1 = 1$ or $Y_1 = 3$ or $Y_1 = 5$). In the rest of the paper, for the Binomial model, we use the full expansion of the variance since it can be computed exactly. On the other hand, for the Exact model, the covariance terms are not equal to zero, and the full expansion cannot be computed exactly. Thus, for the Exact model, the expansion is computed using all of the odd terms up to $2m - 1$ with $m = 1-4$. For instance, for $m = 3$, we include the following pairs of (Y_1, Y_2) values in the expansion of the covariance (eq 18): (1,1), (1,3), (3,1), (3,3), (1,5), (5,1), (3,5), (5,3), and (5,5). We call this the "third-order" expansion of the covariance. Note that the third- and fourth-order expansions give very similar answers. The first- and second-order expansions, however, give unrealistic results as soon as A is greater than 40 or so, leading even to erroneous negative variance estimates. Thus, to compromise between accuracy and efficiency, in the rest of the paper, we use the third-order expansion to estimate the variance terms of the Exact model.

For the other two approximate models given in the Appendix, it is possible to use all of the terms in the expansion of the expectation, as in the case of the Binomial and Exact models. For the variance of the two additional models given in the Appendix, we have been able to use up to $m = 3$ for the model with distinguishable balls, and up to $m = 5$ for the model with indistinguishable balls (see the Appendix).

6.2. Assessment of the Statistical Models: Accuracy. We address how the Binomial and Exact models fit real data by considering first the prediction of the mean and variance of the number a of 1-bits in the XOR-compressed vector of a fingerprint containing A 1-bits. As can be seen in Figure 4,

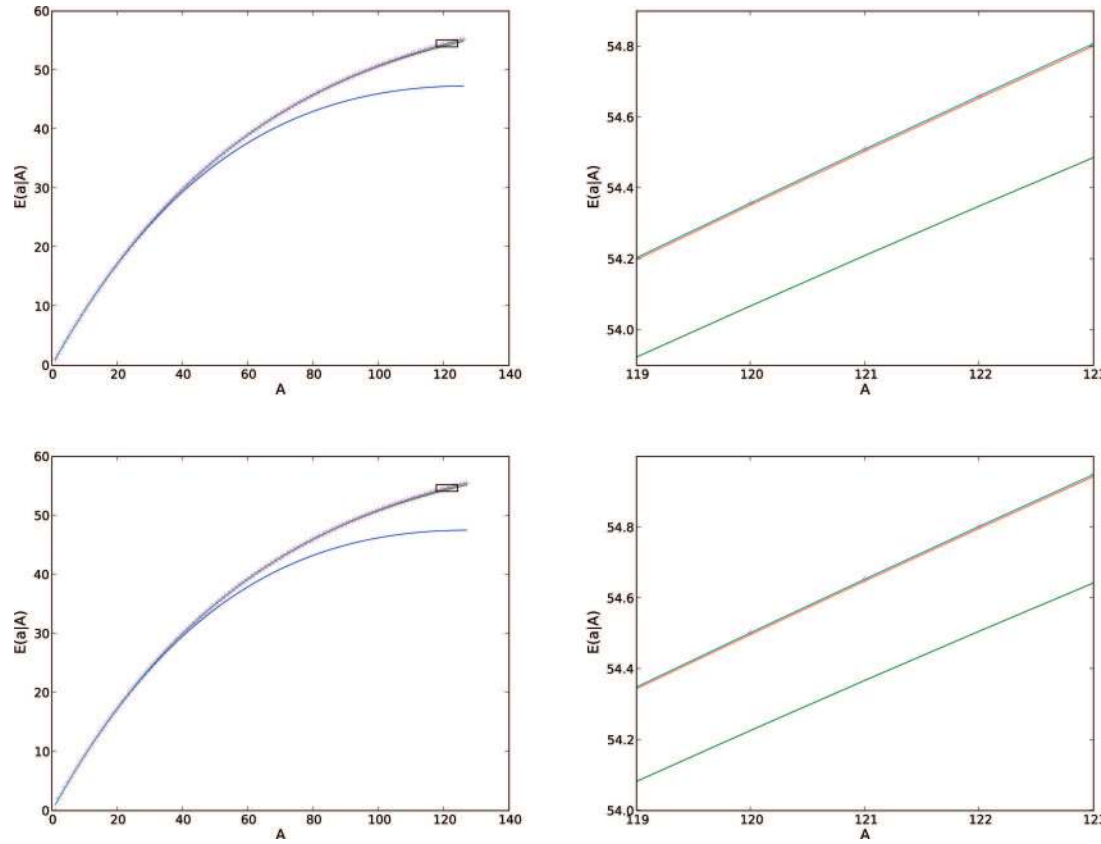

Figure 2. (Upper left) Expected value of the number a of 1-bits in the XOR-folded fingerprints given the number A of 1-bits in the unfolded fingerprint computed under the Binomial model. The blue, green, red, and cyan curves correspond to using 1, 2, 3, and 4 terms, respectively, in the complete expansion of the expectation (eq 26). Purple crosses correspond to the full expansion. (Upper right) Magnified view of the box insert from upper left. (Lower left) Expected value of the number a of 1-bits in the XOR-folded fingerprints given the number A of 1-bits in the unfolded fingerprint computed under the Exact model. The blue, green, red, and cyan curves correspond to using 1, 2, 3, and 4 terms, respectively, in the complete expansion of the expectation (eq 14). Purple crosses correspond to the full expansion. (Lower right) Magnified view of the box insert.

both the Binomial model and the Exact model give excellent predictions of the expectation of a given A, both in simulated data as well as in real data extracted from ChemDB. As shown in the right subfigure of Figure 4, the Approximate model in the Appendix with distinguishable balls significantly underestimates the expected value of a.

When one looks at the variance of a given A in Figure 5, one sees that the Binomial still provides reasonable predictions but tends to overestimate the variance, as predicted by the theory since negative covariance terms are ignored. The Exact model, on the other hand, provides excellent predictions of the variance across the entire range of values of A. The Approximate model in the Appendix with distinguishable balls yields predictions that are better than those of the Binomial model.

In short, the Exact model gives very good predictions of real behavior for both the mean and the variance. The simpler Binomial model can be used to approximate the behavior accurately for the mean, but much less so for the variance, which tends to be overestimated by the Binomial model.

6.3. Assessment of Database Pruning. Heat maps for the average fraction of the database being pruned using the XOR bounds are given in Figure 6 under different models. The upper-left heat map corresponds to the empirical data in ChemDB. It shows that overall the XOR approach is very effective at pruning the search space. Even in the more difficult case of queries with the highest observed density ($A = 128$ or so), most of the database can be eliminated from the search for Tanimoto thresholds as low as 0.6 or so. A similar behavior, with a slight overall decrease in the degree of pruning, is seen in the upper-right heat map corresponding to randomly generated fingerprints using independent Bernoulli trials, with a different probability p_i of producing a 1-bit for each component i. The values p_i are set to the empirical ChemDB frequencies. The difference in behavior with respect to the real data is due to the correlations between the components of the ChemDB data. The two lower heat maps correspond to the Binomial and Exact models. The overall level of pruning is slightly better for the Exact model, presumably due again to its nonzero correlations. Both the Exact model and the Binomial model provide a good approximation of the behavior of ChemDB. The Exact model is slightly more accurate, especially for larger values of A. However, to assess pruning levels, the Binomial model can also be used as a reasonable approximation to the overall behavior and is computationally less demanding than the Exact model.

Slices through these heatmaps for threshold values of 0.4, 0.5, 0.6, and 0.7 are shown in Figure 7. The Exact model approximates the ChemDB data better than the Binomial model (green curve) at all thresholds, but especially for low thresholds ($t = 0.4$). The Exact model (blue curve) tends to overapproximate the pruning of the ChemDB data (red solid curve), especially for low thresholds, because of its inherent assumption of uniformly distributed 1-bits in the fingerprints. 1-bits in ChemDB fingerprints are not uniformly distributed

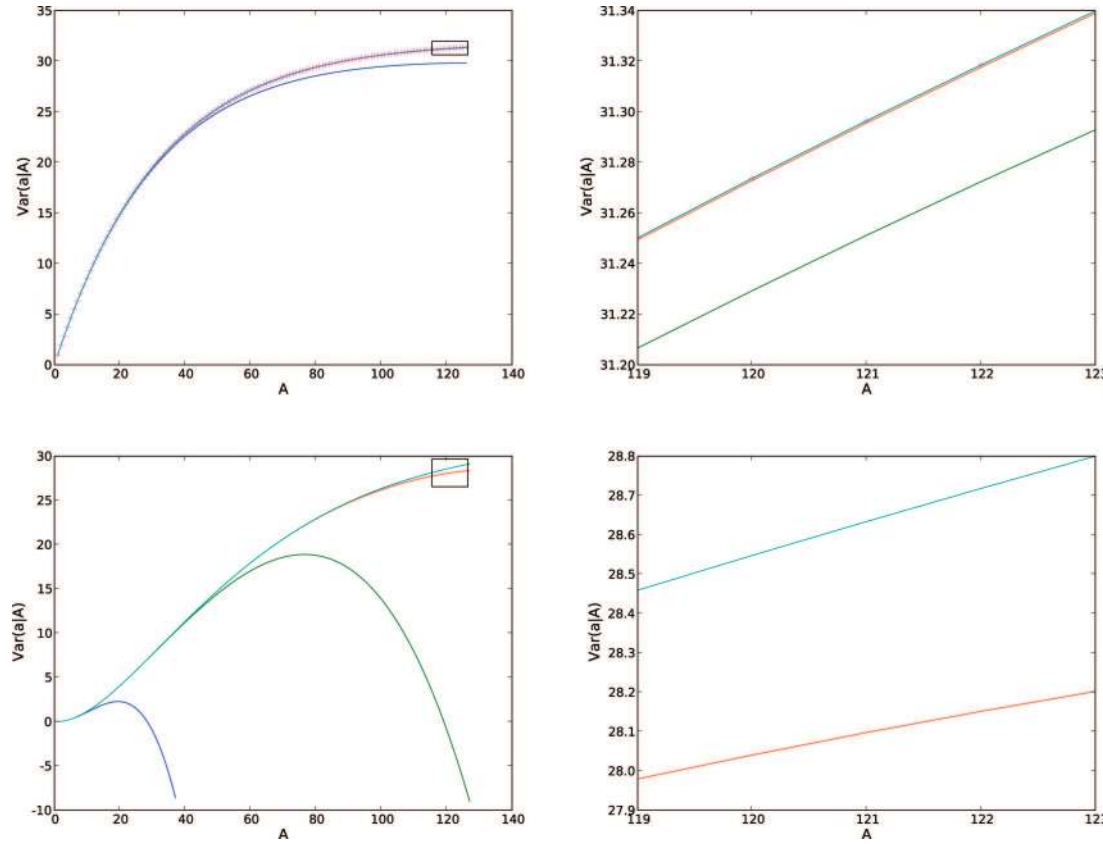

Figure 3. (Upper left) Variance of the number a of 1-bits in the XOR-folded fingerprints given the number A of 1-bits in the unfolded fingerprint computed under the Binomial model. The blue, green, red, and cyan curves correspond to using 1, 2, 3, and 4 terms respectively in the complete expansion associated with the variance (eq 26, since all covariance terms are 0). Purple crosses correspond to the full expansion. (Upper right) Magnified view of the box insert. (Lower left) Variance of the number a of 1-bits in the XOR-folded fingerprints given the number A of 1-bits in the unfolded fingerprint computed under the Exact model. The blue, green, red, and cyan curves correspond to using $m = 1$, 2, 3, and 4 terms, respectively, in the expansion of the variance (eq 18). Note that, for $m = 1$ (blue curve) and $m = 2$ (green), the errors resulting from omitting higher-order terms result in unrealistic (negative) variance estimates. (Lower right) Magnified view of the box insert.

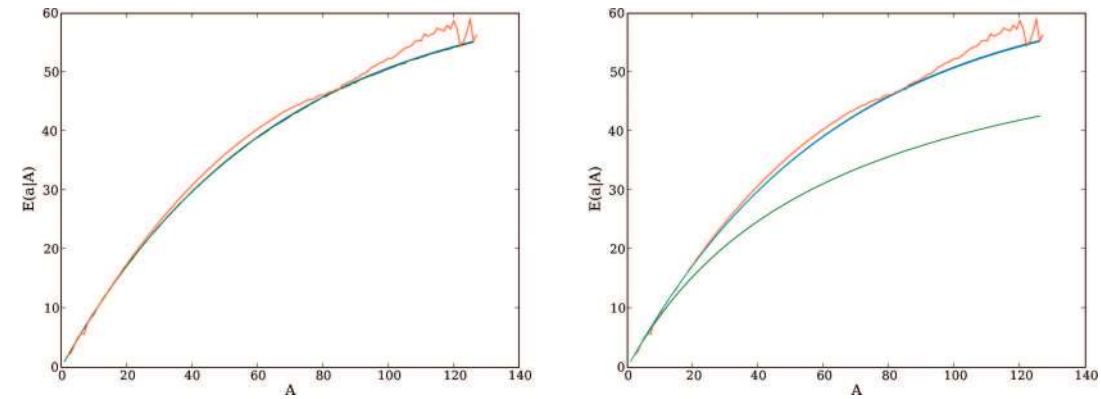

Figure 4. (Left) Expected value of the number a of 1-bits in the XOR-folded fingerprints given the number A of 1-bits in the unfolded fingerprint. The red curves correspond to the empirical average computed over the ChemDB database. The blue curve corresponds to the expectation computed using the Binomial model (eqs 27–29). The green curve corresponds to a Monte Carlo simulation of the Binomial model with 100 000 trials (essentially indistinguishable from the blue curve). (Right) Similar plot, but here the blue curve corresponds to the expectation computed using the Exact model (eq 14 evaluated using eq 40). The green curve corresponds to the Approximate model with distinguishable balls in the Appendix (eqs 45–47). The cobalt curve corresponds to a Monte Carlo simulation of the Exact model with 100 000 trials dropping a fixed number of balls A into $N = 128$ boxes (uniform distribution of configurations for a fixed value of A).

and as a result tend to produce on average a smaller value for $a \oplus b$, which in turn results in fewer molecules being pruned (see Equation 12).

It is also worthwhile to compare the level of pruning provided by the XOR bound, corresponding to eq 9, to the level of pruning previously described[10] using the bit bound, corresponding to eq 3. The very significant improvement associated with the XOR bound is clearly visible in Figure 8 by comparing the green and blue areas. Note also a small further improvement by combining both approaches, shown in red, corresponding to the relatively rare cases of molecules that are pruned using the bit bound but not the XOR bound. Because the bit bound is simpler and faster to compute, in our database implementation, we first filter molecules using

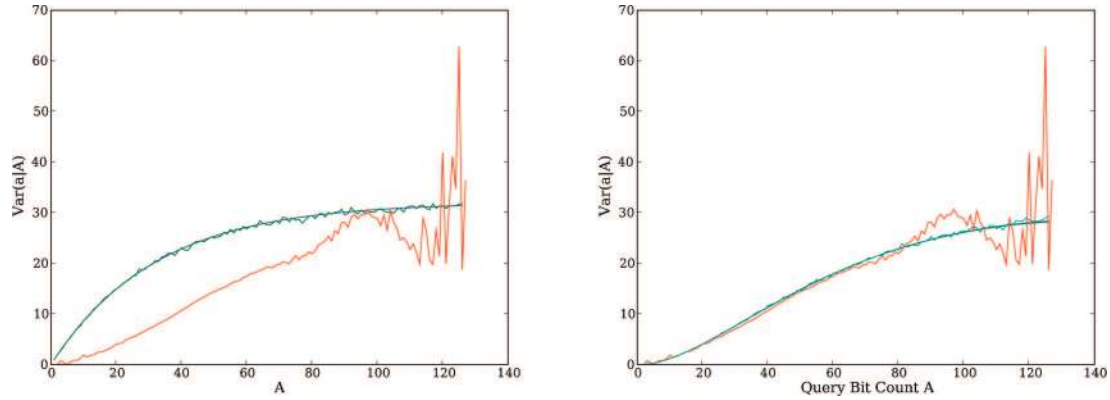

Figure 5. (Left) Same as Figure 4, left, but curves are for the variance of a given A (eqs 24–32). (Right) Same as Figure 4, right, but curves are for the variance of a given A (eqs 15–18) computed using eqs 40 and 41.

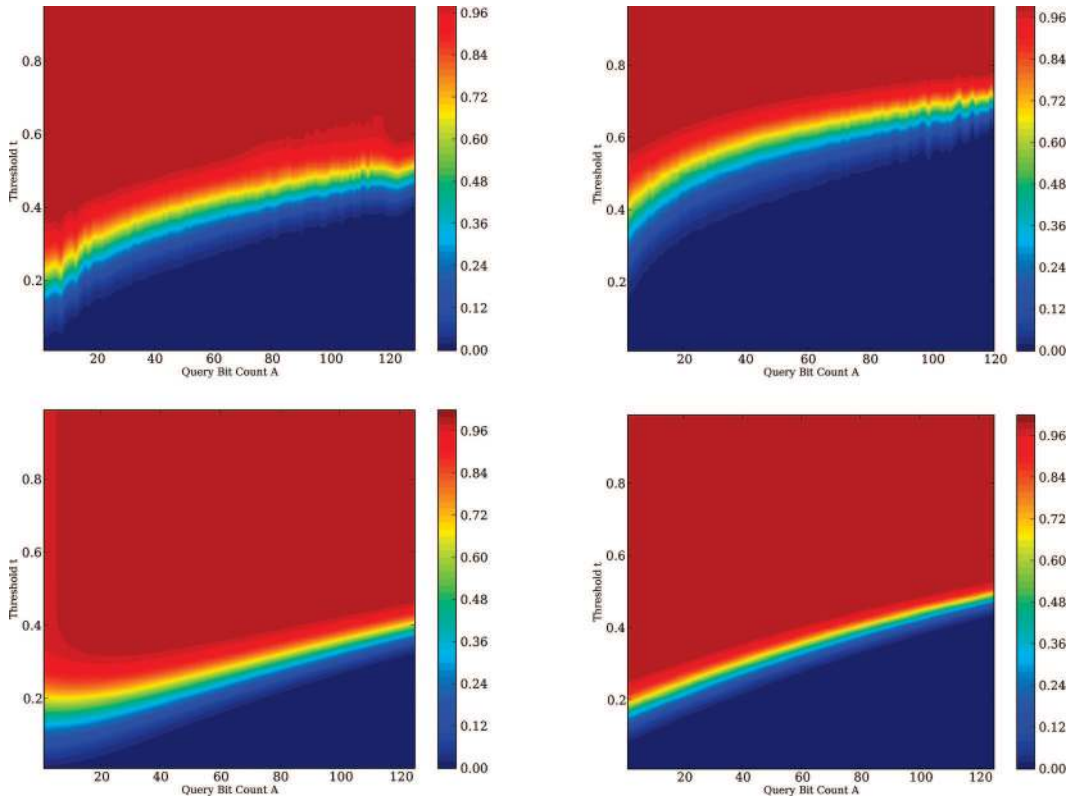

Figure 6. (Upper left) Average fraction of database pruned using a random sample of 100 000 fingerprints from ChemDB, as a function of the query bit count A and the threshold t. (Upper right) Average fraction of database pruned using a sample of 100 000 randomly generated fingerprints as a function of the query bit count A and the threshold t. The fingerprints are randomly generated by using for each component i a Bernoulli trial with probability p_i. The values p_i are set to the empirical values found in ChemDB. (Lower left) Probability of pruning predicted as a function of the query bit count A and the threshold t using the Binomial model (eq 35). The values of B in eq 35 are obtained by sampling from a Gaussian distribution of the number B of 1-bits in the fingerprints in ChemDB. (Lower right) Probability of pruning predicted as a function of the query bit count A and the threshold t using the Exact model. The values of B are obtained as above.

the bit bound, followed by application of the $a - b$ bound (eq 9, right bound) and the XOR bound (eq 9, middle bound) (see Assessment of Speedup).

6.4. Assessment of Speedup. While the amount of pruning is a fundamental consideration, speed benchmarks must be conducted to make sure that the additional costs involved with computing the XOR signatures and the related bounds do not offset the pruning gains. In this regard, one must note that the XOR signatures can be precomputed offline for all of the molecules in the database. Only the XOR signature of the query molecule needs to be computed at query time, but this introduces only a very small overhead compared to the rest of the operations. To address the tradeoff between the time wasted in computing the XOR bounds versus the time saved by the corresponding pruning, we benchmarked the time taken by each algorithm by running many queries on a single machine, corresponding to an Intel Pentium 4, 2.20 GHz CPU with 1 GB of RAM (Figure 9) using a random subset of 100 000 molecules. Although 100 000 is a fairly large sample size, experiments were repeated six times, using six different samples of 100 000 molecules to check for robustness. The standard deviations of the results across the experiments did not exceed a few percentage points (6.5% in the worst case).

For a Tanimoto threshold of 0.8, the best algorithm is about 5.5 times faster than a full linear search of the database,

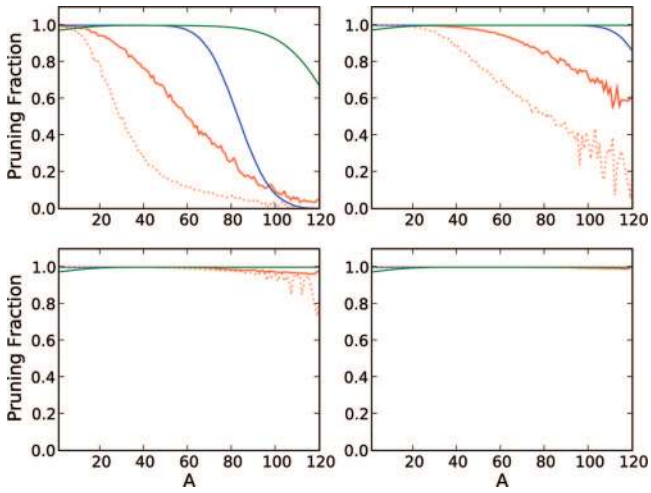

Figure 7. Curves representing the fraction of molecules discarded from a given search as a function of the size A of the binary query fingerprint and the similarity threshold $t = 0.4$ (upper left), $t = 0.5$ (upper right), $t = 0.6$ (lower left), and $t = 0.7$ (lower right). The solid red line corresponds to pruning values computed using a random sample of 100 000 molecules from the ChemDB database, and the dotted red line corresponds to pruning values computed using randomly generated fingerprints by using for each component i a Bernoulli trial with probability p_i. The values p_i are set to the empirical values found in ChemDB. The solid blue line corresponds to pruning values predicted by the Exact model, and the green line corresponds to pruning values predicted by the Binomial model.

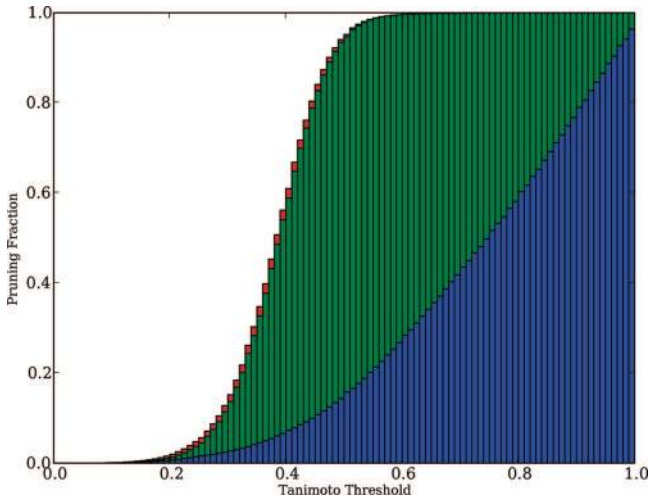

Figure 8. Fraction of the database pruned by different methods as a function of the Tanimoto threshold used in the search. The values reflect the average pruning ratio over 100 randomly sampled query molecules. The database consists of 100 000 randomly sampled molecules from ChemDB. Results obtained using the bit bound approach (eq 3) are shown in blue, while results for the XOR approach are shown in green. Results in red are obtained by combining both methods.

and about 2.4 times faster than the previous state-of-the-art based on the bit bound alone. Overall, the combination of the bit bound with the XOR bound, or even the slightly better combination of the bit bound, followed by the $|a - b|$ bound, followed by the XOR bound, leads to a robust 2−3-fold speedup over the bit bound alone across the entire range of possible thresholds. Figure 10 shows in more detail the fold speedup difference between the bit bound pruning approach and the bit bound approach followed by the XOR pruning approach. Depending on the Tanimoto threshold, the fold speedup varies from 1.5 to 4.5. Slight variations are seen

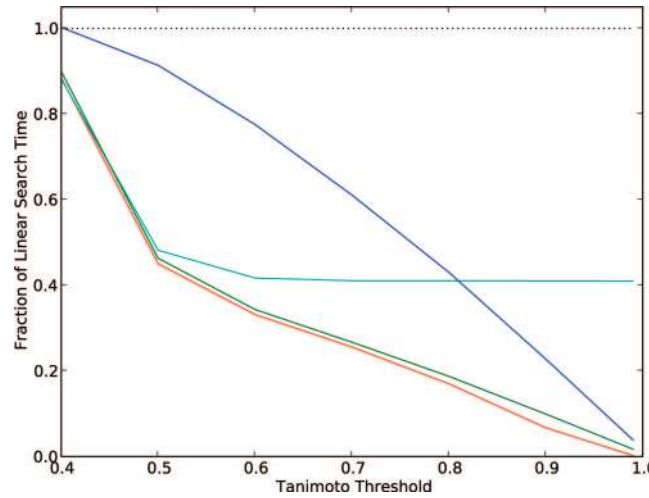

Figure 9. Timing benchmarks using different pruning methods as a function of the Tanimoto threshold used in the search. Each measurement reported corresponds to the average time to search 100 randomly sampled molecules from a database of 100 000 randomly selected molecules on an Intel Pentium 4, 2.20 GHz CPU with 1 GB of RAM. Six different samples of 100 000 molecules were used to test for robustness, and for each sample, 10 random subsets of 100 molecules were used to establish query statistics. The standard deviations of the results over different query or database samples did not exceed a few percentage points (6.5% in the worst case). The unit on the y axis (and the dotted black curve) correspond to a full search of the database, molecule by molecule, with no pruning. The blue curve corresponds to pruning using the previous bit bound approach (eq 3). The cyan curve correspond to pruning with the XOR approach (eq 9, middle bound). The green curve corresponds to pruning using the bit bound approach first, followed by the XOR approach. The red curve corresponds to pruning by using the bit bound approach first, followed by pruning using the difference $|a - b|$ of the number of 1-bits in the XOR-compressed vectors (eq 9, right bound), followed by pruning using the XOR approach (eq 9, middle bound).

depending on whether the queries are randomly sampled from the ChemDB database or randomly sampled from the queries that are typically submitted to the ChemDB database.

Similar speedup results are seen with queries aimed at retrieving the top K molecules, rather than the molecules above a fixed similarity threshold. The algorithm to retrieve the top K molecules proceeds as in our previous work,[10] first by binning molecules by their A values, then applying the bounds within each bit in the order: bit, then $|a - b|$, and finally XOR; starting from the bin associated with the query, and moving in alternation toward smaller or higher values of A.

7. CONCLUSION

In summary, a new approach has been developed for fingerprint-based chemical searches, which relies on storing together with each fingerprint a short signature of typical length $n = 128$ containing a lossy summary of the full fingerprint, obtained by applying the logical XOR operator to the full fingerprint modulo n. This short header can be used to derive tight bounds on the similarity measure, Tanimoto or other, between the query molecule and each molecule in the database. In turn, the bounds can be used to prune the database and restrict the search to only a fraction of the molecules. Theoretical probabilistic models have been introduced that allow one to understand the statistical behavior of the XOR operation and precisely predict the

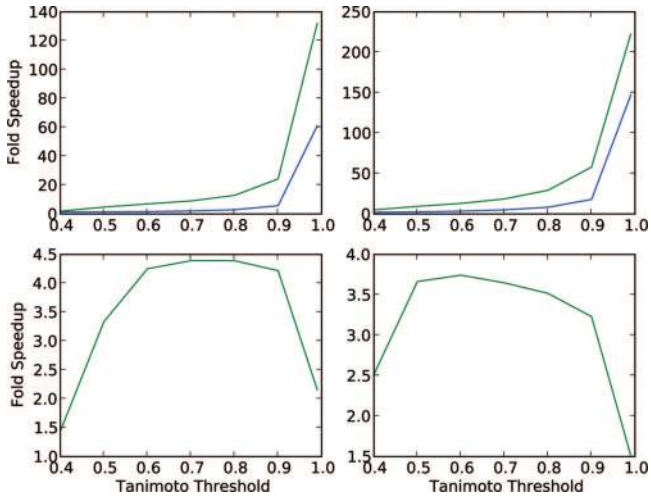

Figure 10. Fold speedups associated with the experiments of Figure 9. Blue curves correspond to pruning using the bit bound approach. Green curves correspond to pruning using the bit bound approach first, followed by the XOR approach. For the upper-row figures, the unit on the *y* axis corresponds to a full search of the database with no pruning. For the lower-row figures, the unit on the *y* axis corresponds to the bit bound approach. Left column results obtained using 100 queries randomly selected from the ChemDB database. Right column results obtained using 100 queries randomly selected from the logs of ChemDB queries received over the Jan−Feb 2008 period.

statistical behavior of the resulting headers, bounds, and amount of pruning. The XOR approach, and its theoretical models, have been tested in simulations and shown to be very effective in further speeding up chemical searches, with 2-fold or greater speedups over current implementations, at a small storage cost. The XOR approach is not limited to chemical data but ought to be applicable to other areas where long binary vectors are used in combination with similarity measures based on intersections and unions.

ACKNOWLEDGMENT

This work was supported by NIH Biomedical Informatics Training grant (LM-07443-01), NSF MRI grant (EIA-0321390), and NSF grant 0513376 to P.B. We would like also to acknowledge the OpenBabel project and OpenEye Scientific Software for their free software academic licenses.

APPENDIX

A. EXCHANGEABILITY WITHOUT INDEPENDENCE

A1. Approximate Model with Indistinguishable Balls. Below, we give another nonexact way of approximating the quantities of interest assuming A fixed and considering the balls as indistinguishable. To compute probabilities under this exchangeable fixed-size model, recall that there are $\binom{n+A-1}{A}$ ways of placing A indistinguishable balls (or bits) into n boxes. The number of ways of putting A balls into n boxes, so that box 1 is empty, is $\binom{n+A-2}{A}$. The number of ways of putting A balls into n boxes so that box 1 contains exactly one ball is $\binom{n+A-1}{A-1}$. More generally, the number of ways of putting A balls into n boxes, so that box 1 contains exactly k balls ($k \leq A$) is

$$\binom{A+n-k-2}{A-k} \tag{42}$$

Thus, for $k \leq A$

$$P(Y_1 = k) = \frac{\binom{A+n-k-2}{A-k}}{\binom{n+A-1}{A}} \tag{43}$$

Likewise,

$$P(Y_1 = k, Y_2 = l) = \frac{\binom{A+n-k-l-3}{A-k-l}}{\binom{n+A-1}{A}} \tag{44}$$

From here, we can proceed as described in the general case to derive exact expressions, or use the lower-order terms to derive approximations when the number of balls is relatively small compared to the number of boxes (Figure 11).

A2. Approximate Model with Distinguishable Balls. Yet another non-exact way of approximating the quantities of interest assuming A fixed can be obtained by considering the ball as distinguishable. In this case, there are n^A ways of putting the A distinguishable balls into the N boxes, provided $M \geq A$. Under these assumptions, the number of ways of putting A balls into n boxes, so that box 1 is empty, is $(n-1)^A$. The number of ways of putting A balls into n boxes so

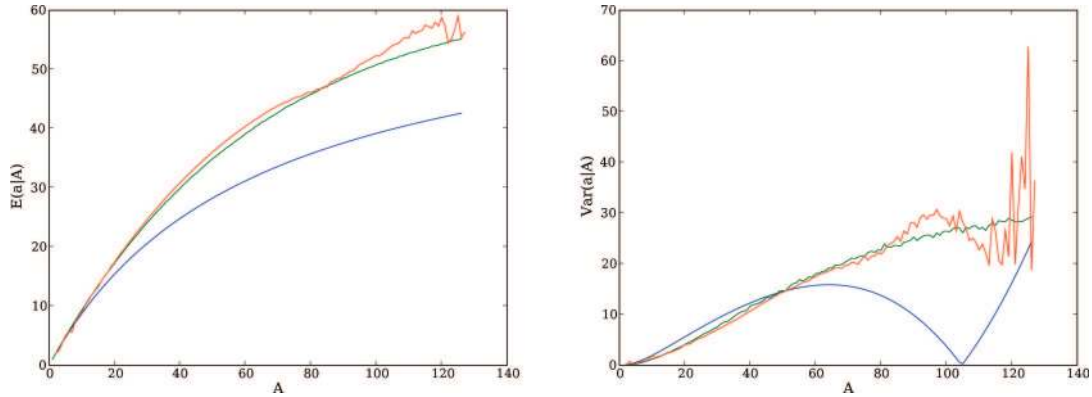

Figure 11. (Left) Expected value of the number a of 1-bits in the XOR-folded fingerprints given the number A of 1-bits in the unfolded fingerprint. The red curve corresponds to the actual average values in ChemDB. The blue curve is the theoretical value provided by the Approximate model with indistinguishable balls (eqs 42–44). The green curve is obtained by Monte Carlo simulation with 100 000 trials of dropping a fixed number A of balls into $N = 128$ boxes, corresponding to a uniform distribution over all configurations. The green curve corresponds to the Approximate model with distinguishable balls (eqs 45–47). (Right) Same figure, but for the variance.

that box 1 contains exactly one ball is $A(n-1)^{A-1}$. More generally, the number of ways of putting A balls into n boxes, so that box 1 contains exactly k balls ($k \leq A$), is

$$\binom{A}{k}(n-1)^{A-k} \tag{45}$$

Thus, for $k \leq A$

$$P(Y_1 = k) = \frac{\binom{A}{k}(n-1)^{A-k}}{n^A} \tag{46}$$

Likewise

$$P(Y_1 = k, Y_2 = l) = \frac{\binom{A}{k}\binom{A-k}{l}(n-2)^{A-k-l}}{n^A} \tag{47}$$

A3.. Extensions to Count-Based Fingerprints. In count-based fingerprints, a molecule **A** is represented by a vector $\vec{A} = (A_i)$ of length N, where the A_i are integers. Most often, these integers count the number of occurrences of a particular feature, substructure, or graph in **A**. For count-based fingerprints, the MinMax similarity measure defined by $S(\vec{A}, \vec{B}) = \sum_i \min(A_i, B_i)/\sum_i \max(A_i, B_i)$ generalizes the Tanimoto measure.

The bounds and pruning ideas presented in ref 10 and in this paper can be extended to count-based fingerprints. While a complete study of optimal signature for count-based fingerprints is beyond the scope of this paper, here we show how the simple bit bound approach can be extended to count-based fingerprints. To derive bounds on S, we can first store with each molecule **A** a header containing the number $A = \sum_i A_i$. Alternatively, or in addition, we can store the number $\bar{A}$ of nonzero components in $\vec{A}$ and their maximum value $\max_i A_i$. In addition (see below), one could also store the minimum of the A_i values which are not equal to 0, denoted by $\min_i A_i(\neq 0)$. We can then derive a number of simple bounds such as

$$\sum_i \min(A_i, B_i) \leq \min(A, B) \leq \frac{A+B}{2} \tag{48}$$

$$\sum_i \min(A_i, B_i) \leq \min(\bar{A}, \bar{B}) \min(\max_i A_i, \max_i B_i) \tag{49}$$

[This is because $\min(A_i, B_i) \neq 0$ only for nonzero components that are common to $\vec{A}$ and $\vec{B}$ and there are at most $\min(\bar{A},\bar{B})$ such components. Furthermore, for any component i, $\min(A_i, B_i) \leq \min(\max_i A_i, \max_i B_i)$.]

$$\sum_i \max(A_i, B_i) \geq \max(A, B) \geq \frac{A+B}{2} \tag{50}$$

$$\sum_i \max(A_i, B_i) \geq \max(\bar{A}, \bar{B}) \max(\min_i A_i(\neq 0), \min_i B_i(\neq 0)) \geq \max(\bar{A}, \bar{B}) \tag{51}$$

using an argument similar to the argument given for eq 49. From these inequalities, one can easily get bounds on the MinMax similarity such as

$$S(\vec{A}, \vec{B}) = \frac{\sum_i \min(A_i, B_i)}{\sum_i \max(A_i, B_i)} \leq \frac{\min(A, B)}{\max(A, B)} \tag{52}$$

or

$$S(\vec{A}, \vec{B}) = \frac{\sum_i \min(A_i, B_i)}{\sum_i \max(A_i, B_i)} \leq \frac{\min(\bar{A}, \bar{B})\min(\max_i, A_i, \max_i, B_i)}{\max(\bar{A}, \bar{B})\max(\min_i A_i(\neq 0), \min_i B_i(\neq 0))} \tag{53}$$

$$\leq \frac{\min(\bar{A}, \bar{B})\min(\max_i, A_i, \max_i, B_i)}{\max(\bar{A}, \bar{B})} \tag{54}$$

These inequalities are the equivalent for count-based fingerprints of the bit bound for binary fingerprints. From these inequalities on the MinMax similarity, one can proceed with pruning the search space in the same way as for binary fingerprints.

REFERENCES AND NOTES

(1) Chen, J.; Swamidass, S. J.; Bruand, J.; Baldi, P. ChemDB: a public database of small molecules and related chemoinformatics resources. *Bioinformatics* **2005**, *21*, 4133–4139.

(2) Chen, J.; Linstead, E.; Swamidass, S. J.; Wang, D.; Baldi, P. ChemDB Update - Full Text Search and Virtual Chemical Space. *Bioinformatics* **2007**, *23*, 2348–2351.

(3) Fligner, M. A.; Verducci, J. S.; Blower, P. E. A Modification of the Jaccard/Tanimoto Similarity Index for Diverse Selection of Chemical Compounds Using Binary Strings. *Technometrics* **2002**, *44*, 110–119.

(4) Flower, D. R. On the Properties of Bit String-Based Measures of Chemical Similarity. *J. Chem. Inf. Comput. Sci.* **1998**, *38*, 379–386.

(5) James, C. A.; Weininger, D.; Delany, J. Daylight Theory Manual. http://www.daylight.com/dayhtml/doc/theory/ (accessed April 9, 2008).

(6) Xue, L.; Godden, J. F.; Stahura, F. L.; Bajorath, J. Profile scaling increases the similarity search performance of molecular fingerprints containing numerical descriptors and structural keys. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 1218–1225.

(7) Xue, L.; Stahura, F. L.; Bajorath, J. Similarity search profiling reveals effects of fingerprint scaling in virtual screening. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 2032–2039.

(8) Leach, A. R.; Gillet, V. J. *An Introduction to Chemoinformatics*; Kluwer Academic Publishers: Dordrecht, The Netherlands, 2003; Vol. 1, pp 53−75.

(9) Baldi, P.; Benz, R. W.; Hirschberg, D.; Swamidass, S. Lossless Compression of Chemical Fingerprints Using Integer Entropy Codes Improves Storage and Retrieval. *J. Chem. Inf. Model.* **2007**, *47*, 2098–2109.

(10) Swamidass, S.; Baldi, P. Bounds and Algorithms for Exact Searches of Chemical Fingerprints in Linear and Sub-Linear Time. *J. Chem. Inf. Model.* **2007**, *47*, 302–317.

(11) Bloom, B. H. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM* **1970**, *13*, 422–426.

(12) Holliday, J. D.; Hu, C. Y.; Willett, P. Grouping of coefficients for the calculation of inter-molecular similarity and dissimilarity using 2D fragment bit-strings. *Comb. Chem. High Throughput Screening* **2002**, *5*, 155–166.

(13) Tversky, A. Features of similarity. *Psychol. Rev.* **1977**, *84*, 327–352.

(14) Rouvray, D. H. Definition and role of similarity concepts in the chemical and physical sciences. *J. Chem. Inf. Comput. Sci.* **1992**, *32*, 580–586.

(15) Hert, J.; Willett, P.; Wilton, D. J.; Acklin, P.; Azzaoui, K.; Jacoby, E.; Schuffenhauer, A. Comparison of topological descriptors for similarity-based virtual screening using multiple bioactive reference structures. *Org. Biomol. Chem.* **2004**, *2*, 3256–3266.

(16) Bender, A.; Mussa, H.; Glen, R.; Reiling, S. Similarity Searching of Chemical Databases Using Atom Environment Descriptors (MOLPRINT 2D): Evaluation of Performance. *J. Chem. Inf. Model.* **2004**, *44*, 1708–1718.

(17) Hassan, M.; Brown, R. D.; Varma-O'Brien, S.; Rogers, D. Cheminformatics analysis and learning in a data pipelining environment. *Mol. Diversity* **2006**, *V10*, 283–299.

(18) Swamidass, S.; Baldi, P. A Mathematical Correction for Fingerprint Similarity Measures to Improve Chemical Retrieval. *J. Chem. Inf. Model.* **2007**, *47*, 952–964.

CI800076S