Free semi-group ciphers

Alexander Towell
atowell@siue.edu

Abstract

We define the semantics of abstract data type for the *oblivious set*. We demonstrate a theoretical data structure, denoted the *Singular Hash Set*, that provides an optimal implementation of the oblivious set with respect to *entropy* and *space complexity*. Finally, we show how to use the *Bloom filter* and *Perfect Hash Filter* to implement oblivious sets and compare them to each other and the theoretically optimal *Singular Hash Set*.

Contents

1 Introduction

The *oblivious set*[**?**] is a fundamental data structure that may be used to construct other oblivious object types, like secure indices for Boolean Encrypted Search[**?**]

An *oblivious set* is an oblivious object type over $(2^{\mathcal{U}}, \in)$, which means that the oblivious set represents values in the set $2^{\mathcal{U}}$ over the binary *member-of* predicate $\in \colon \mathcal{U} \times 2^{\mathcal{U}} \mapsto \{0, 1\}$.

Definition 1.1 (n-fold Cartesian product)**.** *Let $\mathcal{X}_1, \ldots, \mathcal{X}_n$ denote n sets. The set*

$$\mathcal{X}_1 \times \cdots \times \mathcal{X}_n := \{(x_1, \ldots, x_n) \colon x_1 \in \mathcal{X}_1 \land \cdots \land x_n \in \mathcal{X}_n\} \tag{1}$$

is called the Cartesian product of sets $\mathcal{X}_1, \ldots, \mathcal{X}_n$.

A shorthand notation for the Cartesian product $\mathcal{X} \times \mathcal{X} \times \mathcal{X}$ is denoted by $\mathcal{X}^3$.

Suppose we have sets $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n$ and a surjective pairing function

$$\mathrm{f} \colon \mathbb{N}^2 \mapsto \mathbb{N} \,. \tag{2}$$

Recursively, the n-tuple encoding function $\mathrm{g}_n \colon \mathbb{N}^n \mapsto \mathbb{N}$ may be defined as

$$\mathrm{g}_2(x_1, x_2) \coloneqq \mathrm{f}(x_1, x_2) \tag{3}$$

$$\mathrm{g}_n(x_1, \ldots, x_n) \coloneqq \mathrm{f}(x_1, \mathrm{g}_{n-1}(x_2, \ldots, x_n)) \,. \tag{4}$$

Assuming we have a serializer $\mathrm{h}_j \colon X_j \mapsto \mathbb{N}$ for $j = 1, \ldots, n$, an encoder for tuples of tyiple $X_1 \times \cdots \times X_n$ is given by

$$\mathrm{encode}(X_1, \ldots, X_n) \coloneqq \mathrm{g}_n(\mathrm{h}_1(X_1), \ldots, \mathrm{h}_n(X_n)) \,. \tag{5}$$

Now, we may use any random approximate set over the natural numbers to represent any n-tuple relation.

A *regular function* over some abstract data type X behaves the same way if given any data structure that implements the behavior of X.

1. Approximate membership tests of specific elements may be performed with a false positive rate ε and a false negative rate ω. Note that there is no way to efficiently iterate over the elements.

2. The cardinality may be estimated to be within some range. The the degree of uncertainty can be made arbitrarily large entropy at the expense of its space complexity.

3. Set-theoretic operations like union, intersection, and complement generate oblivious sets that approximate the true operation as a function of the false positive and false negative rates and the degree of similarity between the exact sets under consideration.

In section 2, we precisely define the oblivious set.

In **??**, we derive the probablistic model of *oblivious sets*. In **??**, we derive the *entropy* of *oblivious sets*. In **??**, we derive estimators of properties of *oblivious sets*. In section 5, we provide a theoretically optimal implementation of the oblivious set.

2 Cipher sets

A set is given by the following definition.

Definition 2.1. *A set is an unordered collection of distinct elements from a universe of elements.*

A countable set is a *finite set* or a *countably infinite set*. A *finite set* has a finite number of elements. For example,

$$\mathcal{S} = \{1, 3, 5\}$$

is a finite set with three elements. A *countably infinite set* can be put in one-to-one correspondence with the set of natural numbers

$$\mathbb{N} = \{1, 2, 3, 4, 5, \ldots\} \,. \tag{6}$$

The cardinality of a set $\mathcal{S}$ is a measure of the number of elements in the set, denoted by

$$|\mathcal{S}| \,. \tag{7}$$

The cardinality of a *finite set* is a non-negative integer and counts the number of elements in the set, e.g.,

$$|\{1,3,5\}| = 3\,.$$

Informally, a cipher set of $\mathcal{S}$, denoted by $\breve{\mathcal{S}}$, provides a *confidential* in-place binary representation such that very little information about $\mathcal{S}$ is disclosed.

Note that an oblivious set where elements are from the universe $\mathcal{U}$ permits membership tests on elements in $\mathcal{U}$. These elements, and the universe $\mathcal{U}$, are *not* necessarily oblivious types. It is also possible to have an oblivious set over oblivious elements, where the elements have their own set of separate constraints, e.g., an oblivious type X in which only the less-than and equality predicates are possible. The oblivious set, of course, only allows *membership* tests to be performed on elements over $\breve{\mathcal{U}}$.

2.1 Notation

The binary set $\{0,1\}$ is denoted by $\mathcal{B}$. The set of all bit (binary) strings of length n is therefore $\mathcal{B}^n$. The cardinality of $\mathcal{B}^n$ is

$$|\mathcal{B}^n| = 2^n\,. \tag{8}$$

The set of all bit strings of length n or less is denoted by $\mathcal{B}^{\leq n}$, which has a cardinality $2^{n+1}-1$. The countably infinite set of all bit strings, $\mathcal{B}^{\leq\infty}$, is also denoted by $\mathcal{B}^*$.

The bit length of an object x is denoted by

$$\ell(x)\,, \tag{9}$$

e.g., the bit length of any $x \in \mathcal{B}^n$ is $\ell(x) = n$.

A (convenient) one-to-one correspondence between $\mathcal{B}$ and $\mathbb{N}$ is given by the following definition.

Definition 2.2. *Let the set of bit strings $\mathcal{B}^*$ and the set of natural numbers $\mathbb{N}$ have the bijection given by*

$$(b_1\, b_2\, \cdots\, b_m) \longleftrightarrow 2^m + \sum_{j=1}^{m} 2^{m-j} b_j\,. \tag{10}$$

We denote the mapping of a bit string (or natural number) x by x'.

An important observation of this mapping is that a natural number n maps to a bit string n' of length $\ell(n') = \lfloor \log_2 n \rfloor$.

2.2 Oblivious object types

A *type* is a set and the elements of the set are called the *values* of the type. An *abstract data type* is a type and a set of operations on values of the type. For example, the *integer* abstract data type is defined by the set of integers and standard operations like addition and subtraction. A *data structure* is a particular way of organizing data and may *model* one or more abstract data types.

Suppose we have an abstract data type denoted by T with a set of operators $\mathcal{F}$ denoted the *computational basis* that are (at least partially) functions of T. We denote that an *object* x in computer memory models T by $T(x)$.

An *oblivious* object type[**?**] that models T is a related type denoted by $\breve{T}$ that provides guarantees about what can be learned about an object $\breve{x}$ by observing its representation, memory access patterns, or any other possible correlations.

Optimally, the only information that can be learned about $\breve{x}$ is given by the well-defined *behavior* of the the abstract data type T on a subset of its computational basis $\mathcal{F}$.[1] For instance, say an operator $\mathrm{g} \colon T \mapsto \mathcal{B}$ is not in the computational basis of $\breve{T}$, i.e., $\mathrm{g} \colon \breve{T} \mapsto \mathcal{B}$ is undefined. Suppose $\mathrm{g}(x) = 1$ for a particular object of T and half of the inputs map to 1 (and half map to 0). If the only information we have about x is given by $\breve{x}$, then $\mathrm{P}[\mathrm{g}(\breve{x}) = 1] = 0.5$, i.e., we can do no better than a random guess.

We consider random approximate cipher sets of the type $\boxed{2^{\breve{\mathcal{X}}}}^{\pm}$ where $\mathcal{X}$ is the underlying universal set of elements, i.e., cipher sets whose bit string representations are uncorrelated with its specific members over cipher elements, frequently denoted *trapdoors*. This is opposed to, say, $2^{\breve{\mathcal{X}}}$ in which we have a regular set over ciphers elements of type $\mathcal{X}$, or $\boxed{2^{\mathcal{X}}}$ cipher sets over elements of type $\mathcal{X}$.

It is generally easier to understand what we mean by this with respect to a *computational basis*, e.g., the *singular hash set* has a computational basis given by

$$\in \colon \boxed{2^{\mathcal{X}}} \times \breve{\mathcal{X}} \overset{+}{\underset{-}{\square\!\!\mapsto}} \mathcal{B}\,. \tag{11}$$

Other operations, like set-theoretic operations, may be composed using this computational basis. Such a composition may contain information about the objective sets, but an explicit composition of cipher sets is still generally considered to be a cipher set.

A more confidential variation of a cipher set is given by

$$\in \colon \boxed{2^{\mathcal{X}}} \times \breve{\mathcal{X}} \overset{+}{\underset{-}{\square\!\!\mapsto}} \breve{\mathcal{B}}\,, \tag{12}$$

i.e., its membership predicate returns a Boolean cipher. Since *representational equality* implies *equality*, there is always, at least implicitly, a predicate for the identity relation of the type

$$= \colon \boxed{2^{\mathcal{X}}} \times \boxed{2^{\mathcal{X}}} \mapsto \mathcal{B}\,. \tag{13}$$

However, equality does not necessarily imply representational equality, and so there is also an opportunity for a predicate of the type

$$= \colon \boxed{2^{\mathcal{X}}} \times \boxed{2^{\mathcal{X}}} \overset{+}{\underset{-}{\square\!\!\mapsto}} \breve{\mathcal{B}}\,, \tag{14}$$

but this may only be practical for relatively small cipher sets. If the cipher sets are random approximate sets over a *countably infinite* universe, then *equality* implies *representational equality*.

3 Cipher Boolean algebras

Cipher value types may come in many shapes. They may be parametric types, e.g., $\breve{T}$ is a cipher type of plain type T and they may provide differing levels of *confidentiality*, whether as a simple substitution cipher, homophonic cipher, or homomorphic encryption.

In what follows, we consider several ciphers sets, one in which is a Boolean algebra that provides for all set relations like *membership*, *identity*, and *subset*, and the other is a group over symmetric difference and intersection but only offers the *identity* relation (or only one its representations in a, say, homophonic encryption system). Both, however, model their relations as predicate functions which return *plaintext* Booleans rather than cipher variations, i.e., $\mathcal{B}$ as opposed to the more confidential $\breve{\mathcal{B}}$.

NOTE: If using something like homophonic encryption, then identity is no longer maintained. A cipher set $\breve{\mathcal{A}}$ sent in one round may be different is *incomparable* to a cipher set $\breve{\mathcal{A}}$ sent in another round.

[1]Frequently, some operations may reveal too much information about the values and so are excluded from the computational basis of the oblivious type.

NOTE 2: When, say, including bigrams (for whatever reason), if we want to be able to construct a cipher bigram from two cipher elements on the untrusted system, then in the cipher map or set on the untrusted system, all of these combinations must be included separately (unless rate-distorted, in which case errors). Since this may limit its value, we can not support these operations and just go

and we wish with these cipher sets, then identity is no longer provided for by the information in the sets alone. See *approximate maps* or *approximate sets*.

Say we have the Boolean algebra

$$A := \left(2^{\mathcal{B}*}, \cup, \cap, \overline{}, \varnothing, \mathcal{B}*\right) \tag{15}$$

where $\mathcal{B}^*$ is the free semigroup over $\mathcal{B}$, which is the closure over concatenation $+\!\!+ \colon \mathcal{B}* \times \mathcal{B}* \mapsto \mathcal{B}*$.

Since $\mathcal{B}*$ can encode any tuple of elements, and $2^{\mathcal{B}*}$ can encode any sum type of tuples, this Boolean algebra may be a reasonable model for hashing algebraic data types, while still permitting some operations on the resultant hash.

Consider the Boolean algebra

$$B := (\mathcal{B}^m, \vee, \wedge, \neg, 0^m, 1^m) \tag{16}$$

and suppose we wish to provide a homomorphism of type $A \mapsto B$ that, approximately, preserves relations while *obscuring* or *encrypting* the identities.

A *cryptographic hash function* is given by the following definition.

Definition 3.1. *The* hash function $\mathrm{h} \colon \mathcal{B}* \mapsto \mathcal{B}^m$ *maps (hashes) bit strings of arbitrary-length to bit strings of fixed-length m and approximates a* random oracle *over its codomain.*

Definition 3.2. *The homomorphism* $\mathrm{F} \colon A \mapsto B$ *is defined as*

$$\mathrm{F}(\beta) := \begin{cases} \mathrm{h}(x) & \beta \in \mathcal{B}* \\ \vee & \beta = \cup \\ \wedge & \beta = \cap \\ \neg & \beta = \overline{} \\ 0^m & \beta = \varnothing \\ 1^m & \beta = \mathcal{B}*\,, \end{cases} \tag{17}$$

where $\vee$, $\wedge$, *and* $\neg$ *are respectively bitwise* or, and, *and* negation.

This homomorphism is *one-way* for two independent reasons. First, F is not injective, i.e., multiple values in $\mathcal{B}*$ (countably infinite many, since $\mathcal{B}*$ is countably infinite) must map to at least one value in $\mathcal{B}^m$. Second, the hash function h is a cryptographic hash function that is resistant to preimage attacks, i.e., given a y it is *hard* to find a x such that $\mathrm{h}(x) = y$ but given an x it is *easy* to find a y such that $\mathrm{h}(x) = y$.

Thus, given a set $\{x, y, z\} = \{x\} \cup \{y\} \cup \{z\}$, F maps this to $\mathrm{h}(x) \vee \mathrm{h}(y) \vee \mathrm{h}(z)$.

These model *positive approximate sets* with a false positive rate given by the following theorem.

Theorem 3.1. *The false positive rate is a function of m and n and is given by*

$$\varepsilon(m, n) = \alpha^m(n) \tag{18}$$

where

$$\alpha(n) := \left(1 - 2^{-(n+1)}\right)\,. \tag{19}$$

Proof. Proof here. See paper sheets that are star-numbered. □

The *bit-rate*, the bits per element, is given by the following corollary.

Corollary 3.1.1.

$$\mathrm{b}(n, \varepsilon) = \frac{\log_2 \varepsilon}{n\alpha(n)} \tag{20}$$

Proof. Proof here. See paper sheets that are star-numbered. □

The *partial derivative* of b with respect to ε is given by

$$\frac{\partial \mathrm{b}}{\partial \varepsilon} = -\frac{\mathrm{b}}{\mathrm{B}(\varepsilon)}, \tag{21}$$

where $\mathrm{B}(\varepsilon) \coloneqq -\varepsilon \log_2 \varepsilon$ is the *binary entropy function*. This partial is less than zero everywhere but finds its minimum at $\frac{1}{e}$.

The *absolute efficiency* of this homomorphism, when implemented as a data structure for a false positive rate ε is given by

$$\mathtt{E}(n) = -n \log_2 \alpha(n)\,, \tag{22}$$

which is exponential with respect to n.

As $n \to \infty$, $\mathtt{E}(n)$ goes to 0. It is only practical for small values of n, where n is the number of elements of the set.

The *identity* relation has false positives given by the following.

Theorem 3.2. *False positives on the* equality *predicate occurs with probability*

$$\varepsilon(m) = ?\,, \tag{23}$$

where m is the bit length of the hash function in the Boolean cipher algebra.

Proof. Two sets are said to be identical if they map to the same bit string, and thus, as before, *false positives* are possible,

$$\mathrm{P}[\mathrm{G}\mathcal{X} = \mathrm{G}\mathcal{Y} \,\big|\, \mathcal{X} \neq \mathcal{Y}] \tag{a}$$

which occurs with probability ?... □

TODO: figured out membership fpr, now figure out subset fpr, identity fpr also.

4 Cipher Boolean ring

Say we have the Boolean ring

$$A \coloneqq \left(2^{\mathcal{B}*}, \triangle, \cap, \overline{}, \varnothing\right) \tag{24}$$

where $\mathcal{B}^*$ is the free semigroup over $\mathcal{B}$, which is the closure over concatenation $+\!\!+ \colon \mathcal{B}* \times \mathcal{B}* \mapsto \mathcal{B}*$.

Consider the Boolean ring

$$B \coloneqq (\mathcal{B}^m, \oplus, \wedge, \mathrm{id}, 0^m) \tag{25}$$

and suppose we wish to provide a homomorphism of type $A \mapsto B$ that *only* preserves identities.

Definition 4.1. *The homomorphism* $\mathrm{G}\colon A \mapsto C$ *is defined as*

$$\mathrm{G}(\beta) := \begin{cases} \mathrm{h}(x) & \beta \in \mathcal{B}* \\ \oplus & \beta = \triangle \\ \wedge & \beta = \cap \\ \mathrm{id} & \beta = \overline{} \\ 0^m & \beta = \varnothing \end{cases} \tag{26}$$

where $\oplus$, $\wedge$, *and* id *are respectively bitwise* exclusive-or, *bitwise* and, *and* identity.

This homomorphism is *one-way* for the same reason as before. The identity relation

$$(\mathrm{G}\,\mathcal{A})(\mathrm{F}\,\triangle)(\mathrm{G}\,\mathcal{B}) = \mathrm{G}(\mathcal{A}\triangle\mathcal{B}) \tag{27}$$

is guaranteeed by the homomorphism.[2]

There is an *isomorphism* between G and F since $\mathcal{A} \cup \mathcal{B} := (\mathcal{A}\triangle\mathcal{B}) \triangle (\mathcal{A} \cap \mathcal{B})$. If the trusted system applies only the *group*

$$D := (\mathcal{B}^m, \oplus, \mathrm{id}, 0^m) \tag{28}$$

the number of 1's and the cardinality of the set has no relation.

Given a set $\{x, y, z\} = \{x\} \cup \{y\} \cup \{z\}$, G maps this to $\mathrm{h}(x) \oplus \mathrm{h}(y) \oplus \mathrm{h}(z)$.

Since the hash function h is a random oracle, by the property of the exclusive-or operation each set in A maps to a *random* bit string in $\mathcal{B}*$. That is to say, it is indistinguishable from random noise. However, it only supports *identity* relations.

Theorem 4.1. *False positives on the* equality *predicate occurs with probability*

$$\varepsilon(m) = 2^{-m}\,, \tag{29}$$

where m is the bit length of the hash function in the Boolean cipher ring.

Proof. Two sets are said to be identical if they map to the same bit string, and thus, as before, *false positives* are possible,

$$\mathrm{P}[\mathrm{G}\,\mathcal{X} = \mathrm{G}\,\mathcal{Y} \,\big|\, \mathcal{X} \neq \mathcal{Y}] \tag{a}$$

which occurs with probability 2^{-m} since each set maps to a random bit string of size m. □

5 The *Singular Hash Set*

We would like to have properties of both homomorphisms F and G but with a much smaller space complexity.

In what follows, we provide a theoretical implementation of the *cipher set* that obtains optimality in the following ways:

1. The space complexity obtains the theoretical lower-bound of a random approximate positive set with an expected false positive rate ε.

2. The entropy obtains the upper bound for the given expected space complexity.

[2]Observe that if $\mathcal{A}$ and $\mathcal{B}$ are disjoint, $\mathcal{A}\triangle\mathcal{B} = \mathcal{A} \cup \mathcal{B}$.

Definition 5.1. *The* data type *for the* cryptographic *singular hash set is defined as* $\mathsf{SHS}_{\mathrm{F}\mathcal{X}} := \mathcal{B}^k \times \mathcal{B}^*$ *with a value constructor* $\mathtt{shs}_{\mathrm{F}\mathcal{X}} \colon 2^{\mathcal{X}} \times \{\, 2^{-k} \in \mathbb{Q} \mid k \in \mathbb{N} \,\} \mapsto \mathsf{SHS}_{\mathrm{F}\mathcal{X}}$ *defined as*

$$\mathtt{shs}_{\mathrm{F}\mathcal{X}}(\mathcal{A}, \varepsilon) := \langle q, n' \rangle \tag{30}$$

where

$$\begin{aligned}
\mathcal{A} &= \{x_{(1)}, \ldots, x_{(m)}\}\,, \\
k &:= -\log_2 \varepsilon\,, \\
\mathrm{h}_k(b, n) &:= \mathtt{tr}\Big(\mathrm{h}\big(b + n'\big), k\Big)\,, \\
\phi(w, n) &:= \mathrm{h}_k\left(\mathrm{E}(w), n\right)\,, \\
\Omega_l &:= \{\, \phi(x, l) \in \mathcal{B}^k \mid w \in \mathrm{F}(\mathcal{A}) \,\}\,, \\
n &:= \min\{\, j \in \mathbb{N} \mid \Omega_j \in 2^{\mathcal{B}^k} \wedge |\Omega_j| = 1 \,\}\,, \\
q &:= \phi\big(\mathrm{F}\, x_{(1)}, n\big)\,.
\end{aligned} \tag{31}$$

where $\mathrm{F} \colon \mathcal{X} \mapsto \mathcal{W}$ *is a homomorphism and* $\mathrm{E} \colon \mathcal{W} \mapsto \mathcal{B}^*$ *is a prefix-free coder.*[3]

If F is the *identify* function $\mathrm{id} \colon \mathcal{X} \mapsto \mathcal{X}$ then the singular hash set models a random approximate positive set over $\mathcal{X}$, denoted by $\left(2^{\mathcal{X}}\right)^+$. However, typically, the singular hash set is intended to be used for as a *cipher set*, a parametric type $\boxed{\breve{\mathcal{X}}}^+$ where $\mathrm{F} \colon \mathcal{X} \mapsto \breve{\mathcal{X}}$, e.g., $x \overset{\mathrm{F}}{\mapsto} \breve{x}$ where $\breve{x}$ may be, say, an *equivalence class* $\{\breve{x}_1, \ldots, \breve{x}_k\}$ in which each of the elements map back to x under $\mathrm{F}^{-1} \colon \breve{\mathcal{X}} \mapsto \mathcal{X}$.

Theorem 5.1. *The data type* SHS *with value constructor* $\mathtt{shs} \colon \mathcal{P}(\mathcal{B}^*) \times \{\, 2^{-k} \in \mathbb{Q} \mid k \in \mathbb{N} \,\} \mapsto \mathsf{SHS}$ *over the computational basis* $\in \colon \mathsf{SHS} \times \mathcal{B}^* \mapsto \mathcal{B}$ *is a random approximate positive set of* $\breve{\mathcal{A}}$ *with a* false positive rate ε *over* $\breve{\mathcal{X}} \setminus \breve{\mathcal{A}}$*. That is,* a priori*,*

$$\mathtt{shs}_{\mathcal{X}}(\mathcal{A}, \varepsilon) \sim \breve{\mathcal{A}}^+_{\varepsilon} \tag{32}$$

for any $\mathcal{A} \in 2^{\mathcal{X}}$ *and* $\varepsilon \in \{\, 2^{-k} \in \mathbb{Q} \mid k \in \mathbb{N} \,\}$*.*

Proof. In order for the value $\mathtt{shs}_{\mathcal{X}}(\mathcal{A}, \varepsilon)$ to be a random approximate set with a distribution given by $\breve{\mathcal{A}}^+_{\varepsilon}$, it must satisfy two conditions as specified by **??**:

1. $\breve{\mathcal{A}}$ is a subset of $\breve{\mathcal{A}}^+$. This condition guarantees that no *false negatives* may occur.

2. An element in $\breve{\mathcal{X}}$ that is not a member of $\breve{\mathcal{A}}$ is a member of $\breve{\mathcal{A}}^+$ with a probability ε, denoted the false positive rate, i.e.,

$$\mathrm{P}\Big[x \in \breve{\mathcal{A}}^{\pm} \,\Big|\, x \notin \breve{\mathcal{A}}\Big] = \varepsilon \tag{a}$$

   for any $x \in \mathcal{X}$.

To prove the first condition, note that **??** tests any element x for membership in $\mathcal{S}^+$ by computing the hash of x concatenated with the bit string b_n and returning **true** if the hash is h_k where definition 5.1 finds bit strings b_n and h_k such that each element of $\mathcal{S}$ concatenated with b_n hashes to h_k.

To prove the second condition, suppose we have a set $\mathcal{S} = \{x_1, \ldots, x_m\}$ and each element in $\mathcal{S}$ hashes to $y = \mathrm{h}(x_1)$. By **??**, $\mathrm{h} \colon \mathcal{B}^* \mapsto \mathcal{B}^k$ approximates a random oracle and thus uniformly distributes over its domain of 2^k possibilities. Since y is a particular element in $\mathcal{B}^k$, the probability that an element not in $\mathcal{S}$ hashes to y is 2^{-k}. □

[3]If E is not prefix-free, then an approximation error independent of parameter ε is introduced.

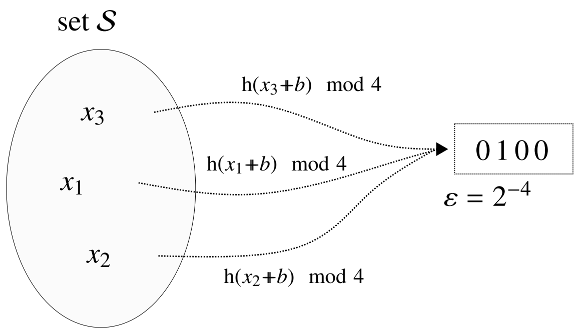


Figure 1: *Singular Hash Set* over a *countably infinite* universe

In particular, we consider a data type that supports both the membership and identity relations while obtaining the information-theoretic expected lower-bound for space with respect to the false positive rate. Since it obtains the information theoretic lower-bound, the bit strings generated naturally obtain *maximum entropy*, i.e., any set $\mathcal{A} \in 2^{\mathcal{X}}$ a priori maps to random bit string in its representation. The only information it leaks is related to its expected bits per element, i.e., it is possible to estimate the cardinality of the set with high certainty.[4]

The singular hash set obtains the theoretically optimal lower bound for a random approximate set (and by definition therefore obtains maximum entropy) while providing for set-membership.

The singular hash set is a data type that implements the *oblivious* random positive approximate set abstract data type. The implementation consists of a product data structure (tuple) $\mathcal{B}^k \times \mathcal{B}^*$, an algorithm that *generates* the data structure, and an algorithm that implements the *member-of* function by appropriately *querying* the data structure.

The *membership* relation is defined by the predicate $\in \colon \breve{\mathcal{X}} \times \mathsf{SHS}_{\mathcal{X}} \mapsto \mathcal{B}$, is defined as

$$\langle h, b \rangle \in \breve{x} \coloneqq q' \mod N \tag{33}$$

where

$$\begin{aligned} k &\coloneqq \lceil \log_2 N \rceil \,, \\ r &\coloneqq \mathrm{h}(x' \mathbin{+\!\!\!+} b) \,, \\ q &\coloneqq \mathtt{tr}(r, k) \,. \end{aligned} \tag{34}$$

Definition 5.2.

$$\in \colon \mathcal{X} \times \mathsf{SHS}_{\mathcal{X}} \mapsto \mathcal{B} \tag{35}$$

5.1 Time and space complexity

The probability that every element of $\mathcal{S}$ collides for a particular bit string in **??** is given by the following theorem.

Theorem 5.2. *The number of time steps the algorithm for* $\mathtt{shs}(m, \varepsilon)$ *is* geometrically *distributed*

$$\mathrm{Q} \sim \mathrm{GEO}(\varepsilon^{m-1}) \tag{36}$$

with an expected *number of steps given by*

$$\varepsilon^{-(m-1)} \,. \tag{37}$$

[4]Of course, the objective set may be poisoned with nonsense values before generating the set such that, for instance, every set is expected to have the same bit length.

Proof. Suppose we have a set of bit strings $\mathcal{A} = \{b_1, \ldots, b_m\}$ and b_1 hashes to $y = \mathrm{h}_k(b_1, n)$ where $\mathrm{h}_k \colon \mathcal{B}^* \mapsto \mathcal{B}^k$ is a random hash function that uniformly distributes over its domain of 2^k possibilities. Since y is a particular element in $\mathcal{B}^k$, the probability that b_j for $j = 2, \ldots, m$ hashes to y is given by

$$\frac{1}{2^k} = \varepsilon \,. \tag{a}$$

Since $\mathrm{h}_{\mathcal{S}}$ is a random hash function, the hashes of $b_1, \ldots, b_m$ are independent. Thus, the joint probability that $b_2, \ldots, b_m$ hash to y is given by the product of their marginal probabilities

$$\varepsilon^{m-1} \,. \tag{b}$$

□

Theorem 5.3. *The expected bit length of the Singular Hash Set obtains the information-theoretic lower-bound given by*

$$-\log_2 \varepsilon \text{ bits/element}\,, \tag{38}$$

where ε is the false positive rate.

Proof. Suppose $|\mathcal{A}| = m$. By definition 5.1, $\mathtt{shs}_{\mathrm{F}\,\mathcal{X}}(\mathcal{A}, \varepsilon)$ is a value constructor that returns a pair of elements, $\langle n', q \rangle$ which has a bit length $\ell(n') + \ell(q)$. The bit length of $\ell(q)$ is just $-\log_2 \varepsilon$. The bit length of n' depends on the minimum value of n found.

Computationally, we search in the order of increasing n, from 0 to ∞. The first case when a perfect collision occurs is a geometric distribution $\mathrm{Q} \sim \mathrm{GEO}\big(p = \varepsilon^{m-1}\big)$. By definition 2.2, the n-th trial uniquely maps to a bit string of length $\lfloor \log_2 n \rfloor$. Thus, the bit string has a random length given approximately by $\mathrm{N} = \log_2 \mathrm{Q}$ bits. We approximate the logarithm with a second-order Taylor series around the *expected* value of Q as given by

$$\mathrm{N} \approx \log_2 \mathrm{E}[\mathrm{Q}] - \frac{\big(\mathrm{Q} - \mathrm{E}[\mathrm{Q}]\big)^2}{2\,\mathrm{E}[\mathrm{Q}]^2} \log_2 e \text{ bits}\,. \tag{a}$$

We are interested in the *expected* value of N,

$$\mathrm{E}[\mathrm{N}] \approx \log_2 \mathrm{E}[\mathrm{Q}] - \frac{\mathrm{V}[\mathrm{Q}]}{2\,\mathrm{E}[\mathrm{Q}]^2} \log_2 e \text{ bits}\,. \tag{b}$$

The expectation and variance of Q is known to be $1/p$ and $(1-p)/p^2$ respectively, and thus we may rewrite eq. (b) as

$$\mathrm{E}[\mathrm{N}] \approx= -\log_2 p - \frac{1-p}{2} \log_2 e \text{ bits}\,. \tag{c}$$

Substituting ε^{m-1} for p in eq. (c) results in

$$\mathrm{E}[\mathrm{N}] \approx -(m-1)\log_2 \varepsilon - \frac{1-\varepsilon^{m-1}}{2} \log_2 e \text{ bits}\,. \tag{d}$$

In total, the bit length of $\langle n', q \rangle$ is thus

$$\ell = -m \log_2 \varepsilon + \frac{1-\varepsilon^{m-1}}{2} \log_2 e \text{ bits} \tag{e}$$

which asymptotically converges to $-\log_2 \varepsilon$ bits per element as $m \to \infty$. □

By **??**, **??** has an expected time complexity that grows exponentially as m grows The algorithm is intended to illustrate theoretical properties, not necessarily be used in practice.

5.2 Probability distributions

The bit representation of a singular hash set of $\mathcal{A}$ is uncorrelated with the specific elements of but is highly correlated with the its *cardinality* $|\mathcal{A}|$.

The probability that $\mathrm{B} = b$ is given by

$$\mathrm{f}_{\mathrm{B}|\mathrm{N}}(b \mid n) = 2^{-n} \mathbb{1}_{\mathcal{B}^n}(b)\,. \tag{39}$$

Proof. Sketch proof. □

Given a false positive rate ε, the probability that $\mathrm{H} = h$ is given by

$$\mathrm{f}_{\mathrm{H}}(h \mid \varepsilon) = \varepsilon \, \mathbb{1}_{\mathcal{B}^k}(h)\,, \tag{40}$$

where $k = -\log_2 \varepsilon$.

Theorem 5.4. *The random bit length* N *has a probability mass function given by*

$$\mathrm{f}_{\mathrm{N}}(n \mid m, \varepsilon) = \left(\frac{1}{q} - q\right) q^{2^n} \mathbb{1}_{\mathbb{N}}(n)\,, \tag{41}$$

where $q = 1 - \varepsilon^{m-1}$, m is the cardinality of the objective set, and ε is the false positive rate.

Proof. The probability that a bit string of length n occurs is given by the probability that Q realizes some value in the range $\{2^n, \ldots, 2^{n+1} - 1\}$, which is given by

$$\begin{aligned} \mathrm{P}\Big[2^n - 1 < \mathrm{Q} \leq 2^{n+1} - 1\Big] &= \mathrm{F}_{\mathrm{Q}}\Big(2^{n+1} - 1\Big) - \mathrm{F}_{\mathrm{Q}}(2^n) && \text{(a)} \\ &= \Big(1 - q^{2^{n+1}-1}\Big) - \Big(1 - q^{2^n - 1}\Big) && \text{(b)} \\ &= q^{2^n - 1} - q^{2^{n+1}-1}\,, && \text{(c)} \end{aligned}$$

which is equivalent to the result. □

Given a false positive rate $\mathcal{E} = \varepsilon$ and an objective set of cardinality m, the joint distribution of B, H, and N is given by

$$\mathrm{f}_{\mathrm{B,H,N}}(b, h, n \mid m, \varepsilon) = 2^{-n} \varepsilon \, \mathrm{f}_{\mathrm{N}}(n \mid m, \varepsilon) \, \mathbb{1}_{\mathcal{B}^n \times \mathcal{B}^k \times \mathbb{N}}(b, h, n) \tag{42}$$

where $k = -\log_2 \varepsilon$.

In the singular hash set, the false positive rate and bit length N are *observable*. Therefore, a primary statistic of interest is the joint distribution of B and H given $\mathrm{N} = n$ and $\mathcal{E} = \varepsilon$, which is given by

$$\mathrm{f}_{\mathrm{B,H|N}}(b, h \mid n, \varepsilon) = 2^{-n} \varepsilon \, \mathbb{1}_{\mathcal{B}^n \times \mathcal{B}^k}(b, h) \tag{43}$$

where $k = -\log_2 \varepsilon$.

The marginal distribution of B is given by marginalizing over the joint distribution with respect to B, which yields the result

$$\mathrm{f}_{\mathrm{B}}(b \mid \varepsilon, m) = 2^{-\ell(b)} \, \mathrm{f}_{\mathrm{N}}(\ell(b) \,\big|\, m, \varepsilon)\,. \tag{44}$$

A few key features of B is that it realizes the empty string with probability ε^{m-1} and as n goes to infinity, the probability that $\ell(\mathrm{B}) = n$ goes to 0.

NOTE: the singular hash set does not expose correlations by bit-wise operations, which may be a good thing, unlike in the bitwise models discussed previously.

The entropy of N is given by the following theorem.

Theorem 5.5. *?*

Proof.

$$\mathcal{H}(\mathrm{N}) = -\,\mathrm{E}\Big[\log_2 \mathrm{f_N}(\mathrm{N})\Big]\,. \tag{a}$$

$$\mathcal{H}(\mathrm{N}) = -\sum_{n=0}^{\infty} \log_2 \mathrm{f_N}(n)\, \mathrm{f_N}(n) \tag{b}$$

$$= -\sum_{n=0}^{\infty} \log_2\!\Big(q^{2^n-1}\left(1 - q^{2^n}\right)\Big) \mathrm{f_N}(n)\,. \tag{c}$$

$$\mathcal{H}(\mathrm{N}) = -\,\log_2 q \sum_{n=0}^{\infty} (2^n - 1)\, \mathrm{f_N}(n) - \sum_{n=0}^{\infty} \log_2\left(1 - q^{2^n}\right) \mathrm{f_N}(n)\,. \tag{d}$$

$$\mathcal{H}(\mathrm{N}) = -\,\log_2 q \left[\sum_{n=0}^{\infty} (2^n\, \mathrm{f_N}(n)\right] - \sum_{n=0}^{\infty} \log_2\left(1 - q^{2^n}\right) \mathrm{f_N}(n)\,. \tag{e}$$

□

Theorem 5.6. *The random bit string that codes the singular hash set is a* maximum entropy *coder for the* approximate set *abstract data type.*

Proof. The result immediately follows from the fact that the bit string is *incompressible* and thus obtains maximum entropy. The bit string $h_k \in \mathcal{B}^k$ is, a priori, a random bit string uniformly distributed over $\mathcal{B}^k$ by the property of the cryptographic hash function. The bit string $b_n \in \mathcal{B}*$ realizes a length n with probability $\mathrm{f_N}(n)$.[5] Given $\mathrm{N} = n$, the bit string $b_n \in \mathcal{B}^n$ is, a priori, a random bit string uniformly distributed over $\mathcal{B}^n$. □

There are two predictable regularities in the bit string representation of a singular hash set.

1. The *randon bit length* N has relatively low entropy and is expected to obtain the information-theoretic lower-bound.

2. The false positive rate has no uncertainty, i.e., zero entropy.

By construction, we are able to estimate that an element is a member or non-member of $\mathcal{A}$ by determining if it is a member or non-member of $\breve{\mathcal{A}}^+$, e.g., positive predictive value and other binary classification measures.

By construction, the singular hash set obtain the information-theoretic lower-bound, which may be used to estimte the cardinality of the objective sets being modeled by a singular hash set. For instance, given a set $\mathcal{A}$, the random bit length N of bit string b_n has a probability mass concentrated around the theoretical lower-bound. Therefore, a *method-of-moments* estimator of the cardinality of the $\mathcal{A}$ is given by

$$|\hat{\mathcal{A}}| = -\frac{\ell(\mathcal{A}^+)}{\log_2 \varepsilon}\,, \tag{45}$$

were ℓ is the bit length function and ε is the *false positive rate*.

[5]As such, we can further compress b_n using a geometric coder that assigns shorter bit strings to more probable lengths, but we require an *in-place* bit string and so stick with the original code.

5.3 Relaxing optimality

By **??**, the time complexity is *exponential* with respect to the cardinality m of the objective set being modeled with a cipher set. We can trade space complexity for time complexity to design more practical algorithms.

Let $\mathcal{A}$ be the objective set. A general approach is to use a *multi-level* hashing scheme such that each level generates smaller independent sub-problems.

We consider a two-level scheme. At the first level, we find a hash function $\mathrm{h}_k \colon \mathcal{X} \mapsto \{0, 1, \ldots, k-1\}$ that maps m/k elements in $\mathcal{A}$ to each element in the codomain.

Definition 5.3. *The cryptographic m/k-perfect hash function is a variation of the perfect hash function. over some specified set of m elements, maps $\frac{k}{m}$ elements per bucket, k in total,*

Definition 5.4. *The cryptographic α-perfect hash function is a variation of the perfect hash function. For each element in the codomain, proportion α of the domain maps to it. If $\alpha = \frac{1}{m}$, then it reduces to the* perfect hash function*. If $\alpha = 1$, then it reduces to a normal cryptographic hash function.*

The cryptographic m/k-perfect hash function has a space and time complexity given by the following theorem.

Theorem 5.7. *A cryptographic m/k-perfect hash function has an expected space complexity given by*

$$-\alpha \log_2 \alpha \tag{46}$$

and an expected time complexity given by

$$\sqrt{\frac{k}{\alpha}}\alpha^{\frac{k}{2}} \tag{47}$$

where $\alpha := \frac{m}{k}$.

Proof. Consider the family of hash functions of the type $\mathcal{X} \mapsto \{1, 2, \ldots, k\}$. When we restrict this family to some objective set $\mathcal{A}$ of cardinality m, we have hash functions of type $\mathcal{A} \mapsto \{1, 2, \ldots, k\}$. There are exactly k^m hash functions of this type.

We are interested only in those hash functions of this type that evenly k-partition the domain into the codomain, i.e., the preimage of each element in the codomain consists of $\alpha := m/k$ elements in the domain.

To count the number of possible functions, we choose α of the m elements in the domain for element 1 in the codomain. There are a total of $\binom{m}{\alpha}$ ways of making this selection. We remove these α elements from the domain. We repeat the procedure again. We choose α of the remaining $m - \alpha$ elements in the domain for element 2 in the codomain. There are a total of $\binom{m-\alpha}{\alpha}$ ways of making this selection. If there are a ways to make a first choice and b ways to make a second choice, by the product rule there are ab ways to make those two choices. Thus, continuing this pattern, there are

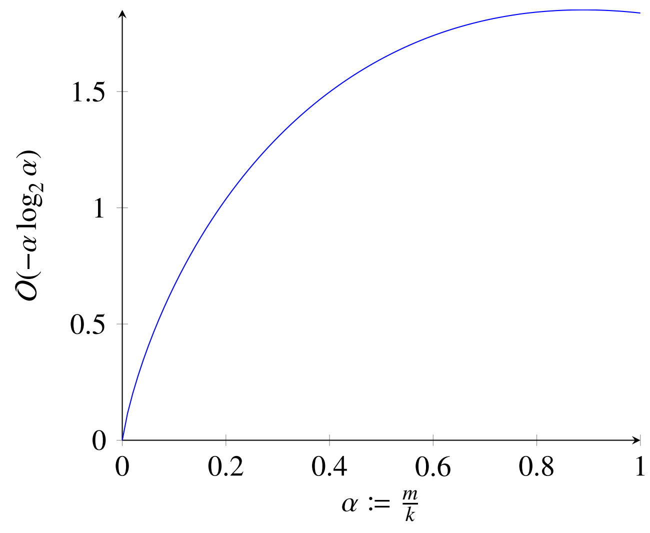
$$\prod_{j=1}^{k} \binom{m - j\alpha}{\alpha} \tag{a}$$

ways to make the selections for the k elements in the codomain, which simplies to

$$\frac{m!}{(\alpha!)^k}\,. \tag{b}$$

To generate perfect hash functions, we append some bit string to each input. By the property of cryptographic hash functions, each bit string serves as an index into the cryptographic hash function family

Figure 2: The expected bits per element of the cryptographic k-perfect hash function.



$\mathcal{A} \mapsto 0, 1, \ldots, k-1$. Thus, since there are k^m functions in this family in total, and only $\frac{m!}{(\alpha!)^k}$ m/k-perfect hash functions of this type, each time we index into the family, with probability

$$p \sim e\sqrt{\frac{k}{\alpha}}\alpha^{\frac{k}{2}} \tag{c}$$

will the hash function be a α-perfect hash function.

The probability of successfully finding an m/k-perfect hash function for each trial bit string is thus *geometrically* distributed with a probability of success p.

The time complexity is just the *expected* number of trials, which is $1/p$ for the geometric distribution, and the length of the successful bit string is just $\log_2 1/p$. □

We employ the m/k-perfect hash function to construct a more practical variation of the singular hash set.

We k-partition the m elements in a specified set using the m/k-perfect hash function, giving us m/k elements per bin, and then we apply the singular hash set to each of these independently.

This is a negative binomial distribution.

Suppose we have two cryptographic hash functions, $\mathrm{h}_1 \colon \mathcal{B}^* \mapsto \mathcal{B}^w$ and $\mathrm{h}_2 \colon \mathcal{B}^w \mapsto \mathcal{B}^k$ where, in general, $w < k$ but this is not necessarily the case. If $w >> |\mathcal{A}|$, then the size of the cipher set is just $-w \log_2 \varepsilon$ where $\varepsilon = 2^{-k}$ since each element of $\mathcal{A}$, with high probability, maps to a unique hash by h_1.

The first-level hash function $\mathrm{h}_k := \mathcal{X} \mapsto \{0, 1, \ldots, k-1\}$ is defined as

$$\mathrm{h}_k(x) := \langle h, n' \rangle \tag{48}$$

where

$$\begin{aligned} \phi(w, n) &:= \mathrm{h}(n' + x') \mod k\,, \\ \mathcal{Y}_l &:= \{\, \phi(x, l) \in \mathcal{B}^k \mid x \in \mathcal{A} \,\}\,, \\ n &:= \min\left\{\{\, j \in \mathbb{N} \mid \mathcal{Y}_j \in 2^{\mathcal{B}^k} \wedge |\mathcal{Y}_j| = 1 \,\}\right\}\,. \end{aligned} \tag{49}$$

The first application of the cryptographic hash function maps each element to a particular index. By the property of the cryptographic hash function, the elements are expected to be equally partitioned into the k indices. Next, at each index, we associate an independent bit string that serves the same purpose as the bit string b_n in the singular hash set. We then independently apply the singular hash set algorithm to each partition. Let the number of trials necessary for singular hash set of the j-th partition be denoted by T_j. The total expected number of trials is just the sum of the geometrically distributed random variables $\mathrm{T} = \mathrm{T}_1 + \mathrm{T}_2 + \cdots + \mathrm{T}_k$, which is a *negative binomial distribution*,

$$\mathrm{T} \sim \mathrm{NB}\left(k, \varepsilon^{\frac{m}{k}-1}\right) . \tag{50}$$

The extra degree of freedom we have with the number of partitions k means we can quantitatively trade space complexity for time complexity.

Theorem 5.8. *The value of k that is expected to yield a solution at t trials is given by*

$$k = \alpha\, \mathrm{E}[\mathrm{N}] , \tag{51}$$

where $\frac{1}{\alpha} = \log_2\left(\frac{\varepsilon t}{k+t}\right)$.

Proof. We take the approach of solving for some t number of trials and finding a lower-bound k that satisfies the equation,

$$\mathrm{E}[\mathrm{T}] = t . \tag{a}$$

Substituting the expectation of the negative binomial into the above yields the equation

$$\frac{k\varepsilon^{m/k-1}}{1 - \varepsilon^{m/k-1}} = t . \tag{b}$$

Solving for k yields the result. □

We know that, approximately, the bit length is given by

$$\mathrm{N} = \log_2 \mathrm{T} , \tag{52}$$

which may be solved as described in **??**.

We may also quantify the probability that, after t trials, some jointly chosen value of k, t, and ε fails to realize a solution. In fact, we can quantify the probability that r of the m elements fails, which yields a *rate-distortion* $\frac{r}{m}$.

If after t trials no solution is found, we may either continue with the search or stop short, resulting in not just an approximate set with *false positives* but also *false negatives*. We refer to this outcome as a *rate-distorted set*, where the rate distortion is a function of time. However, note that we could also make the rate-distortion a function of *space*, i.e., limit the maximum size of the bit string code and choose the code that yields the smallest false negative rate.

If we are interested in rate-distorted sets with false negative rates greater than 0, then an alternative approach is to consider the probability of failure for a particular k, ε, and m. By the survival function of the negative binomial, the probability that more than t trials (or $\log_2 t$ bits) is required for a particular parameterization k and $p = \varepsilon^{\frac{m}{k}-1}$ is given by

$$\mathrm{I}_p(t+1, k) , \tag{53}$$

where I_p is the *regularized incomplete beta function*.

Theorem 5.9. *A value of type* $\mathsf{SHS}_{\mathcal{X}}^{k}$ *constructed with* $\mathtt{shs}_{\mathcal{X}}(\mathcal{A}, \varepsilon \mid k = m)$ models *a cryptographic perfect hash function* $\mathrm{h}_{\mathcal{A}} \colon \mathcal{X} \mapsto \{0, 1, \ldots, -\log_2 \varepsilon\}$ *where* $\mathcal{A} \subseteq \mathcal{X}$ *and* $r = \frac{|A|}{-\log_2 \varepsilon}$.

5.4 Boolean algebra

The data structure does not trivially model a Boolean algebra with simple bit-wise operators on its representation, as is the case with F and G, but it may be *lifted* to a Boolean algebra by the fact that

$$\mathcal{A} \cup \mathcal{B} = \{ x \in \mathcal{X} \mid x \in \mathcal{A} \lor x \in \mathcal{B} \} \tag{54}$$

and other similar identities. Specifically, we define a *recursive* type for the Boolean algebra as

$$\begin{aligned} \mathrm{B}_{\breve{\mathcal{X}}} &:= \mathsf{0}_{\breve{\mathcal{X}}} + \mathsf{1}_{\breve{\mathcal{X}}} && (55)\\ &+ \mathrm{Just}\,\mathsf{SHS}_{\breve{\mathcal{X}}} && (56)\\ &+ \mathrm{Join}\ \mathrm{B}_{\breve{\mathcal{X}}}\ \mathrm{B}_{\breve{\mathcal{X}}} && (57)\\ &+ \mathrm{Meet}\ \mathrm{B}_{\breve{\mathcal{X}}}\ \mathrm{B}_{\breve{\mathcal{X}}} && (58)\\ &+ \mathrm{Not}\ \mathrm{B}_{\breve{\mathcal{X}}}\ . && (59)\end{aligned}$$

Then, for instance, set-union $\cup\colon\ \mathrm{B}_{\breve{\mathcal{X}}} \times \mathrm{B}_{\breve{\mathcal{X}}} \mapsto \mathrm{B}_{\breve{\mathcal{X}}}$ is defined as $a \cup b := \mathrm{Join}\, a\, b$.

Over the Boolean algebra $\mathrm{B}_{\breve{\mathcal{X}}}$, the membership predicate

$$\in\colon\ \mathrm{B}_{\breve{\mathcal{X}}} \times \mathrm{B}_{\breve{\mathcal{X}}} \mapsto \mathcal{B} \tag{60}$$

is defined as

$$\begin{aligned} x \in \mathsf{1}_{\breve{\mathcal{X}}} &:= 1\,, && (61)\\ x \in \mathsf{0}_{\breve{\mathcal{X}}} &:= 0\,, && (62)\\ x \in \mathrm{Just}\, a &:= x \in a\,, && (63)\\ x \in \mathrm{Join}\, a\, b &:= (x \in a) \lor (x \in b)\ , && (64)\\ x \in \mathrm{Meet}\, a\, b &:= (x \in a) \land (x \in b)\ , && (65)\\ x \in \mathrm{Not}\, a &:= \neg\,(x \in a)\ . && (66)\end{aligned}$$

The nature of the composition reveals something about the sets and decreases entropy.

With respect to the above, any Boolean algebra

$$A := \left(2^{\mathcal{X}}, \cup, \cap, \overline{}, \varnothing, \mathcal{X}\right). \tag{67}$$

may be mapped to the Boolean algebra

$$B := \left(\mathrm{B}_{\breve{\mathcal{X}}}, \cup, \cap, \overline{}, \mathsf{0}_{\breve{\mathcal{X}}}, \mathsf{1}_{\breve{\mathcal{X}}}\right). \tag{68}$$

Interestingly, the mapping from A to B is not a homomorphism since, for instance, $\mathtt{shs}_{\breve{\mathcal{X}}}(\mathcal{A}) \cup \mathtt{shs}_{\breve{\mathcal{X}}}(\mathcal{B})$ does not necessarily equal $\mathtt{shs}_{\breve{\mathcal{X}}}(\mathcal{A} \cup \mathcal{B})$ due to the random approximate set model. That is, the mapping only *probabilistically* preserves relations according to the *higher-order* random approximate set model. One might say that it is *asymptotically* homomorphic ($\varepsilon \to 0$).