Random extents and random closure systems

Bernhard Ganter

Institut für Algebra
Technische Universität Dresden

Abstract. We discuss how to randomly generate extents of a given formal context. Our basic method involves counting the generating sets of an extent, and we show how this can be done using the Möbius function. We then show how to generate closure systems on seven elements uniformly at random.

1 Introduction

Let RANDOM(0,1] denote an operator that generates a random number between 0 and 1 with equal probability. From such a (memoryless) random number generator an operator RANDOM_SUBSET(S) can be derived that produces, upon each invocation, a random subset of a given finite set S, such that all subsets are equally likely (see, e.g., [6]).

Building on this we derive in this article an operator that randomly selects a closed set from a given closure system[1] on a finite set.

Note that this is a trivial task for moderately sized systems of which you can label the closed sets by numbers $1, \ldots, n$. For such you could simply randomly pick a number between 1 and n and select the closed set labeled by this number. Since the size of a closure system is at most exponential in the size of its carrier, this trivial algorithm clearly requires polynomial time. However, a potentially exponential list of closed sets must be pre-computed and stored.

For example we aim at generating *closure systems* at random[2]. But there are many closure systems, even for small carrier sets. On seven elements the number was recently computed by Colomb, Irlande, and Raynaud [3] to be 14 087 648 235 707 352 472. Maintaining a list of this size is not an inviting idea, and thus the trivial approach is not very realistic.

Our motivation comes from recent experimental computer investigations by D. Borchmann that yielded surprising results. Borchmann raised the question if these were artefacts caused by the non-uniform choice of the random input data.

Have a look at Figure 1. It shows five diagrams, each with 27 rows and 13 columns, corresponding to the possible number of meet reducible and irreducible closed sets in a closure system on a five element set (the trivial system with zero irreducibles is omitted). A system with r reducibles and i irreducibles corresponds

[1] That is, from an intersection-closed familiy of sets. Such families are also called *Moore families*.
[2] The family of all closure systems on a fixed set is itself a closure system.

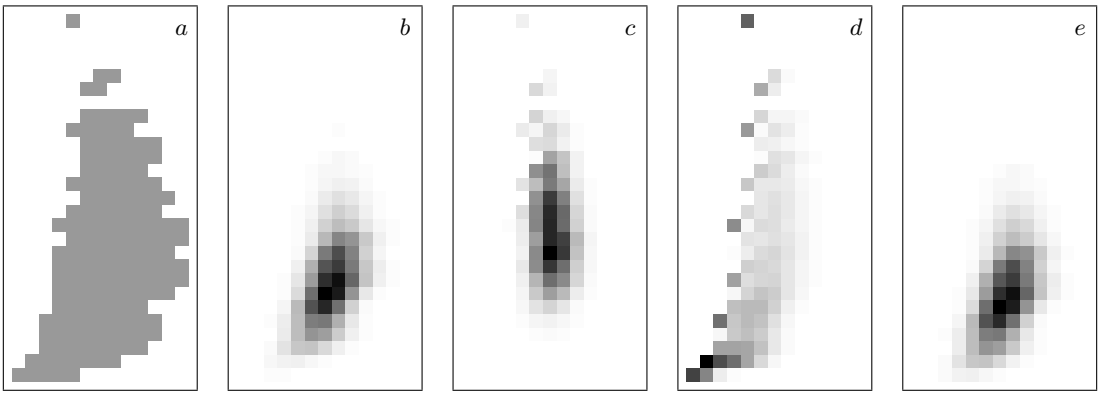

Fig. 1. Closure systems on five elements by their number of meet-irreducible (horizontal) and reducible closed sets (vertical). Tile a shows the possible values, tile b the true relative frequencies, tile c and d come from "random contexts" and tile e from picking systems uniformly at random.

to the cell in the r-th row from the bottom and the i-th column from the left. The first diagram depicts which combinations of r and i are possible, while the other four display relative frequencies (the darker, the higher). The second diagram shows the true frequencies, counted over all 1 385 552 closure systems on five elements. The other three show frequencies of randomly chosen closure systems (1 000 000 samples each). For the diagram in the middle, the systems were made by putting "random crosses" in a 5×13–context. The fourth diagram was obtained by putting "random crosses" with random density in a formal context with a random number of columns. The fifth diagram shows the distribution of a sample picked with uniform distribution.

We use the language of Formal Concept Analysis [4] and, in particular, that every closure system is the set $\mathrm{Ext}(G, M, I)$ of all extents of some formal context (G, M, I). We construct an operator RANDOM_EXTENT(G, M, I) which selects, upon each invocation, randomly an extent of (G, M, I), with equal probability for all extents.

The closure operator for the extents will be denoted by

$$X \mapsto X''.$$

If $X = Y''$, then Y is called a *generating set* of the extent X (not necessarily a *minimal* one). The number of generating sets of an extent E shall be denoted by $egen(E)$. We extend this definition to arbitrary subsets so that

$$egen(Y) := |\{X \subseteq G \mid X'' = Y''\}|$$

gives the number of generating sets of the extent generated by Y. Of course then, $egen(Y) = egen(Y'')$. Therefore if E is an extent then obviously

$$\sum_{\{Y|Y''=E\}} \frac{1}{egen(Y)} = 1.$$

Computing the function *egen*() is a nontrivial task. We shall discuss this below.

Our method could theoretically be applied to many instances, such as generating random partitions, random subgroups, etc. However, its runtime performance is very bad. For most such situations algorithms are known that are much more efficient than what we suggest. Indeed, we do not believe that our method will be very useful in practice. Our contribution is meant as a challenge to come up with a more efficient approach.

We are grateful to the referees for several useful hints. We were unaware of the paper by Boley, Gärtner, and Grosskreutz [2], which addresses the same problem, but with a different and more general approach. It may well be that their algorithm yields better results even for generating random closure systems. We have also learnt that the problem of generating random extents is known to belong to a (difficult) complexity class: it is equivalent to the $\#\mathrm{RH}\Pi_1$-hard problem of counting formal concepts (again, see [2] and the references given there). We already knew (because our colleagues of the stochastics group told us so and recommended the book by Asmussen and Glynn [1] as a standard reference) that our approach is an instance of the so-called *acceptance-rejection method*.

2 Random Extent

Our innocent looking algorithm for generating a random extent of a given formal context (G, M, I) goes like this:

Algorithm 1 RANDOM_EXTENT: Generating a random extent

Input: A formal context (G, M, I).
Output: A random extent of (G, M, I)
  repeat
    $S :=$ RANDOM_SUBSET(G)
  until RANDOM$(0,1] \leq \frac{1}{egen(S)}$
  return S''.

What the algorithm does essentially is to pick a random subset and output its closure with probability one over the number of generating sets[3]. It is quite elementary to prove that it does what it is supposed to do:

Proposition 1 *The algorithm* RANDOM_EXTENT *generates extents of* (G, M, I) *with equal probability.*

The proposition is an instance of the following lemma from elementary stochastics, for which we provide a proof. To obtain the proposition from the lemma, let

[3] One of the referees pointed out that a much simpler algorithm with the same number of expected iterations is obtained by replacing the "until" statement by "until S is closed". We see however no straightforward way to a recursive version of this algorithm.

A be the set of all subsets of G, let B be the set of all extents, and let f be the map that associates a subset to the extent it generates.

Lemma 1 *Let $f : A \to B$ be a surjective (i.e., onto) map between finite sets A and B and let* RANDOM(A) *be an operator that picks elements from A with equal probability. Then Algorithm 2 outputs elements of B with equal probability.*

Algorithm 2 RANDOM IMAGE: Random image of a mapping

Input: An onto map $f : A \to B$ and an operator RANDOM(A)
Output: A random element of B

```
repeat
    a := RANDOM(A)
    r := RANDOM(0,1]
    b := f(a);
until r ≤ 1/|f^{-1}(b)|
return b.
```

Proof It is obvious that the algorithm produces elements of B. In order that a given element b is produced in one iteration of the loop, the element a must belong to $f^{-1}(b)$ and, independently, $r \leq \frac{1}{|f^{-1}(b)|}$. The probability that this happens is

$$\frac{|f^{-1}(b)|}{|A|} \cdot \frac{1}{|f^{-1}(b)|} = \frac{1}{|A|},$$

independently of b. The probability that some element is selected after one step thus is $\frac{|B|}{|A|}$. The probability that the element b is produced after k steps is

$$\left(1 - \frac{|B|}{|A|}\right)^{k-1} \cdot \frac{1}{|A|}.$$

The probability that b is produced is

$$\sum_{k=1}^{\infty} \left(1 - \frac{|B|}{|A|}\right)^{k-1} \cdot \frac{1}{|A|} = \frac{|A|}{|B|} \cdot \frac{1}{|A|} = \frac{1}{|B|},$$

as claimed. □

The expected number of iterations until success is

$$\frac{|A|}{|B|} = \frac{\#\text{subsets}}{\#\text{extents}}.$$

The algorithm may therefore need quite some time. For example, would this algorithm be applied to the standard context of closure systems to generate a random

closure system on a 6-element set, it requires, on average, 121 402 088 iterations of the loop, since that context has $2^6 - 1$ objects and 75 973 751 474 extents ([5]). For closure systems on a seven-element set the average number of loop iterations for obtaining a single random closure system would be 12 077 330 482 260 320 447. As already mentioned we shall develop a better method for this case below. Before we do so, we study the problem of computing the value of $egen(A)$.

3 Counting generating sets and hitting sets

The algorithm in the previous section uses the number $egen(A)$ of a given extent A, and that by itself is not easy. Of course, each such generating set must be a subset of A. On the other hand, a subset $S \subseteq A$ is a generating set of A iff it is not contained in a lower neighbor of A. It is worthwhile to consider the formal context

$$(A, \mathcal{N}, \in),$$

where $\mathcal{N}$ is the family of lower neighbor extents of A. For this context, the elements of $\mathcal{N}$ are precisely the maximal extents below A, and thus the generating sets of A are the same as before. Counting generating sets thereby has been reduced to counting generating sets of the unit element in a co-atomistic lattice.

Every subset of A is generating set of exactly one extent of $(A, \mathcal{N}, \in)$. The total number of generating sets thus is $2^{|A|}$. Indeed, for every extent B we obtain

$$\sum_{E \leq B} egen(E) = 2^{|B|},$$

where E runs over extents. By Möbius inversion we obtain

$$egen(A) = \sum_{E \leq A} \mu(E, A) \cdot 2^{|E|},$$

where μ is the Möbius function of the lattice $\underline{\mathfrak{B}}(A, \mathcal{N}, \in)$.

The evaluation of this formula poses no algorithmic difficulties. Using the standard NEXT_INTENT algorithm ([4]) to generate the extents in descending order, and using, for every constructed extent E, the same algorithm again for producing all extents F between E and A, suffices to compute the Möbius function by the well known recursion

$$\mu(E, A) = - \sum_{E < F \leq A} \mu(F, A).$$

Note that this also counts the hitting sets of any finite hypergraph. A **hitting set** of a family $\mathcal{H} \subseteq \mathfrak{P}(A)$ of subsets of A is a set $T \subseteq A$ that has nonempty intersection with each $H \in \mathcal{H}$ (such sets are also called **(weak) transversal**s).

Obviously, T is a hitting set iff T is not a subset of a complement of some $H \in \mathcal{H}$, that is, iff T is a generating set of the extent A in the formal context

$$(A, \mathcal{H}^c, \in),$$

where

$$\mathcal{H}^c := \{A \setminus H \mid H \in \mathcal{H}\}.$$

The algorithm given above therefore applies. However, using the Möbius function for counting generating sets is costly. Fortunately, it is unnecessary in many cases, as we shall explain in the next section.

4 Splitting the loop

The algorithm poses no conditions on *how* the random set and the random number are generated, and in which order. A huge number of iterations is necessary when we first generate the extent and then perform a random experiment with very low success probability. A better strategy is to perform the random experiment in several steps and interrupt the loop at an early stage, if necessary. For given $0 \leq p < 1$ and $p \leq \alpha < 1$ we may replace the experiment

$$\textit{pick a random number } r = \textsc{Random}(0,1] \textit{ and test if } r \leq p$$

by

$$\textit{pick two random numbers } r = \textsc{Random}(0,1] \textit{ and } s = \textsc{Random}(0,1] \\ \textit{and test if } s \leq \alpha \textit{ and } r \leq \tfrac{p}{\alpha},$$

because they have the same success probability. This is the key to the following lemma.

Lemma 2 *Let $f : A \to B$ and $g : B \to C$ be surjective mappings between finite sets and let* RANDOM(B) *be an operator that picks elements from B with equal probability. Then Algorithm 3 outputs elements of C with equal probability.*

Proof Suppose that we start with a random choice $a \in A$ and apply Algorithm 3 to the mapping $g \circ f$. We would draw a random number $r := \textsc{Random}(0,1]$ and output $c := g(f(a))$ if

$$r \leq p := \frac{1}{|f^{-1}(g^{-1}(c))|}.$$

Equivalently we may split the drawing of r as described above. We first compute $b := f(a)$ and $\alpha := \frac{1}{f^{-1}(b)}$, draw a random number $s = \textsc{Random}(0,1]$ and continue only if

$$s \leq \alpha := \frac{1}{|f^{-1}(b)|}.$$

Algorithm 3 RANDOM IMAGE 2: Random image of a composed mapping

Input: Two onto maps $f : A \to B$ and $g : B \to C$ and an operator RANDOM(B) that randomly outputs elements of B

Output: A random element of C

repeat
  $b :=$ RANDOM(B)
  $r :=$ RANDOM(0,1]
  $c := g(b)$;
until $r \leq \frac{|f^{-1}(b)|}{|f^{-1}(g^{-1}(c))|}$
return c.

We then draw another random number r and output c if

$$r \leq \frac{p}{\alpha} = \frac{|f^{-1}(b)|}{|f^{-1}(g^{-1}(c))|}.$$

According to Lemma 1 the first part of this process generates the elements of B with equal probability. The first part therefore may be replaced by a random choice of elements of B, as stated in the lemma. □

Lemma 2 can be applied in several ways to the random extent problem. Typically, A is the set of all subsets of G and C is the set of all extents, while B is a selected family of generating sets. For example we may take some subset $G_0 \subseteq G$ and let B consists of all sets of the form $E \cup R$, where E is an extent of the formal context $\mathbb{K}_0 := (G_0, M, I \cap (G_0 \times M))$ and $R \cap G_0 = \emptyset$. This yields Algorithm 4.

Algorithm 4 RECURSIVE RANDOM EXTENT

Input: A formal context $\mathbb{K}$ and an operator RANDOM_EXTENT($\mathbb{K}_0$) producing random extents of a subcontext $\mathbb{K}_0 \leq \mathbb{K}$

Output: A random extent of $\mathbb{K}$.

repeat
  $E :=$ RANDOM_EXTENT($\mathbb{K}_0$)
  $A := E \cup$ RANDOM_SUBSET($G \setminus G_0$)
  $r :=$ RANDOM(0,1];
until $r \leq \frac{egen(E,\mathbb{K}_0)}{egen(A)}$
return A''.

5 Random Moore family

Our original motivation was generating closure systems (also called Moore families) at random. The number of generating sets of a closure system is easy to

determine without using the Möbius function. Every closure system has a unique *minimal* generating set, consisting of all meet-irreducible elements. The total number of generating sets therefore is 2 to the power s, where s is the number of *reducible* closed sets.

The natural formal context for the closure systems on a set S is given by

$$(\mathfrak{P}(S), \mathrm{Imp}(S), \models),$$

where

$$\mathrm{Imp}(S) := \{A \to b \mid A \subseteq S,\ b \in S \setminus A\}$$

is the set of all implications with singleton conclusion over S and

$$X \models A \to b \quad :\Longleftrightarrow \quad A \not\subseteq X \text{ or } b \in X.$$

It is well known and easy to see that the extents of this formal context are precisely the closure systems on S. However, this context contains a reducible object, which is S. The standard context therefore is

$$(\mathfrak{P}(S) \setminus \{S\}, \mathrm{Imp}(S), \models).$$

The case we are interested in is $S := \{0, 1, \ldots, n-1\}$. The object set G of the context then consists of all proper subsets of S. We choose

$$G_0 := \{X \subseteq S \mid n-1 \notin X\}$$
$$G_1 := \{X \subset S \mid n-1 \in X\}.$$

The extents of $(G_1, \mathrm{Imp}(S), \models)$ are (when S is added) precisely those closure systems on S that contain $n-1$ in every closed set. These are in an obvious bijection to the closure systems on $S \setminus \{n-1\}$.

The context $(G_0, \mathrm{Imp}(S), \models)$ has precisely twice as many extents as there are closure systems on $\{0, \ldots, n-2\}$: Each closure system $\mathcal{C}$ on $\{0, \ldots, n-2\}$ is an extent, but also $\mathcal{C} \setminus \{S\}$ is.

Let $S := \{0, \ldots, n-1\}$. For applying Lemma 2 we let A be the set of all subsets of $\mathfrak{P}(S) \setminus \{S\}$, C be the family of all closure systems on S, let G_0 and G_1 be as above and B consist of those set families $\mathcal{F} \subseteq \mathfrak{P}(S) \setminus \{S\}$ for which $\mathcal{F} \cap G_0$ and $\mathcal{F} \cap G_1$ are extents of the respective subcontexts. Every closure system $\mathcal{C}$ on S is in B, but not conversely. The closure system generated by a family $\mathcal{F}$ from B may contain additional sets, obtained as intersections of a set in G_0 and a set in G_1. However, such new sets can only be contained in G_0, since a set containing the element $n-1$ cannot be obtained as an intersection involving one not containing $n-1$.

Algorithm 5 encodes the subsets of $S := \{0, \ldots, n-1\}$ in the natural manner as the integers from 0 to $2^n - 1$, using the bitwise AND-operation for intersecting

sets. A closure system on S is represented by an array $F[0\,..\,2^n-2]$ with

$$F[i] := \begin{cases} 0 & \text{if } i \text{ is not closed} \\ 1 & \text{if } i \text{ is closed and reducible} \\ 2 & \text{if } i \text{ is closed and irrreducible.} \end{cases}$$

The algorithm starts with two closure systems on $\{0,\ldots,n-2\}$ and concatenates them. In addition, a random decision is made if the set $\{0,\ldots,n-2\}$ is to be counted as a closed set. The result is not necessarily a closure system and needs to be made intersection closed. This is done in the second part of the algorithm (beginning with "SUCCESS:= **true**"). Whenever a new reducible element is encountered, the generation process is abandoned with probability 0.5 and is started over. Only if this never happens, a closure system on $\{0,..,n-1\}$ is achieved for output.

Algorithm 5 RANDOM_MOORE(n): Random Moore family.

Input: An operator RANDOM_MOORE($n-1$) generating random Moore families on $\{0,\ldots,n-2\}$.
Output: A random Moore family on $\{0,\ldots,n-1\}$.

repeat
  $F_0 :=$ RANDOM_MOORE$(n-1)$
  $F[0\,..\,2^{n-1}-2] := F_0[0\,..\,2^{n-1}-2]$
  $F[2^{n-1}-1] := \begin{cases} 0 & \text{if RANDOM}(0,1] < 0.5 \\ 1 & \text{else.} \end{cases}$
  $F_1 :=$ RANDOM_MOORE$(n-1)$
  $F[2^{n-1}\,..\,2^n-2] := F_1[0\,..\,2^{n-1}-2]$
  $i := 2^n-1$
  SUCCESS := **true**
  while SUCCESS **and** $(i > 2^{n-1})$ **do**
    if $F[i] = 2$ **then**
      $j := 2^{n-1}-1$
      while SUCCESS **and** $(j > 0)$ **do**
        MEET := i AND j
        if $(j \neq$ MEET$)$ **and** $(F[\text{MEET}] \neq 1)$ **then**
          SUCCESS := (RANDOM$(0,1] < 0.5$)
          $F[\text{MEET}] := 1$
        $j := j-1$
    $i := i-1$
until SUCCESS
return F.

We have implemented Algorithm 5 for $n = 7$ and present first experimental results. Note that the number of random samples produced by this experiment is small compared to the number of closure systems: we have generated less than 0.000 000 000 000 4% of all closure systems on seven points.

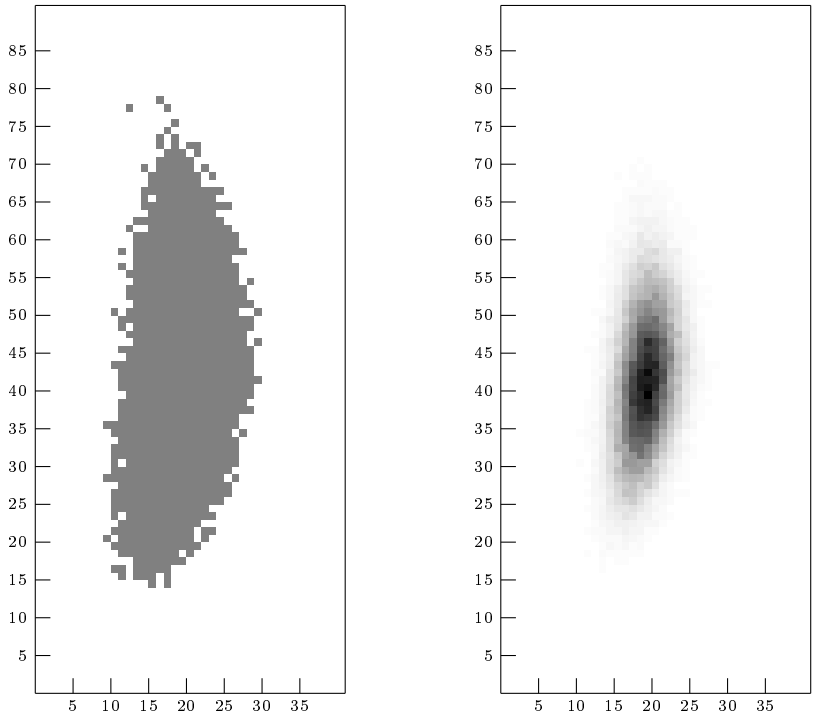

Fig. 2. A random sample of 50 000 closure systems on a seven element set, plotted according to their number of irreducible closed sets (horizontal) and reducible closed sets (vertical). The left image shows which sizes occurred at least once. The right image expresses higher frequencies by darker shadings.

The computation took one night on a 1.4 GHz PC. We did not even attempt to generate random closure systems on eight elements using Algorithm 5. We believe that a substantially better idea is needed for that case and beyond.

References

1. S. Asmussen and P. W. Glynn. *Stochastic Simulation*. Springer-Verlag, New York, 2007.
2. Mario Boley, Henrik Grosskreutz, and Thomas Gärtner. Formal concept sampling for counting and threshold–free local pattern mining. In *Proc. of the SIAM Int. Conf. on Data Mining (SDM 2010)*. SIAM, 2010.
3. Pierre Colomb, Alexis Irlande, and Olivier Raynaud. Counting of Moore families for $n = 7$. In *ICFCA'10*, pages 72–87, 2010.
4. Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis - mathematical foundations*. Springer Verlag, 1999.
5. Michel Habib and Lhouari Nourine. The number of Moore families on $n = 6$. *Discrete Mathematics*, pages 291–296, 2005.
6. Albert Nijenhuis and Herbert S. Wilf. *Combinatorial algorithms*. Academic Press, 1975.