Optimal encoding on discrete lattice with translational invariant constrains using statistical algorithms.

Jarek Duda

Jagiellonian University, Reymonta 4, 30-059 Kraków, Poland,
email: dudaj@interia.pl

Abstract

In this paper it is shown how to almost optimally encode information in valuations of discrete lattice with some translational invariant constrains. The method is based on finding statistical description of such valuations and changing it into statistical algorithm: which allow to construct deterministically valuation with given statistics. Optimal statistic allows to generate valuations with uniform distribution - we get this way maximum information capacity. It will be shown that in this approach we practically can get as close to capacity of the model as we want (found numerically: lost 10^{-10}bit/node for Hard Square). There will be presented an alternative to Huffman coding too, which is more precise and practice with changing probability distributions.

1 Introduction

Consider all projections $\mathbb{Z}^2 \to \{0,1\}$ which fulfils constrains: there cannot be two neighboring "1" (each node has 4 neighbors). We will call *model* such triplet - space ($\mathbb{Z}^2$), alphabet ($\{0,1\}$) and some constrains.
We will call *nodes* - points of the space (usually lattice).
The model above is called Hard Square model(HS).
For a model, all projection as above fulfilling specified translative invariant constrains - will be called *elements* (of the model) - we can think about them as valuations of each node.
We will call *valuation* for a subset of the space - a restriction to this subset of some element - consistent with constrains.
When there are no constrains and we take a finite box in $\mathbb{Z}^2$ the number of valuations growths like 2^N, where N is the number of nodes of the subset(box) we are

restricting to - entropy per one point is

$$H_0 = \lg(2^N)/N = 1$$

in this case we can store 1bit/node.
For HS this number of valuations will grow slower, asymptotically as:

$$2^{H_{HS}N}$$

This time the (average) capacity is $H_{HS} \cong 0.5878911617753406$ bit/node.
It will be shown how to get as close as we want to this capacity.

Thanks of translative invariance, while taking the limit $N \to \infty$, we can speak about *statistical description* (p) - for each *shape* - a finite subset of the space, assign a probabilistic distribution of valuation (*patterns*) on this shape.
For example - in HS we will be interested in statistical descriptions for which for example: $p(11) = p(110) = ... = 0$, intuitively $p(01) = p(10)$.
Of course we have general equalities, like $p(0) + p(1) = 1$, $p(00) + p(01) = p(0)$.
It will occurs that for a given statistical description p, the number of valuations on a finite box with N nodes will asymptotically grow like:

$$2^{H_p N}$$

where H_p is the entropy for p - mean number of bits/node we can store with this statistical description.

For finite spaces, description which allows us to store most corresponds uniform distribution of valuations - we will generalize it to infinite space - approximating the space with finite sets or taking maximum of entropy from all statistical descriptions consistent with constrains ($H = \max_p H_p$).
We should get this way entropy of model and this maximum should be unique, but unfortunately I cannot prove it formally - it's equivalent to vanishing of long-range correlations.
Assuming its existence - we will call this description(p) - *optimal*.

For a given probabilistic description we can create *statistical algorithm* to encode data deterministically with capacity equal to the entropy of this description. Approximating the optimal description, we will be able to get as close as we want to the real capacity of the model.

Examples of usage:
We can think about for example hard disk, locally as valuing nodes (say - magnetizing round dots) of 2 dimensional lattice with 0 or 1 (without constrains).
Now let us change the lattice constant, as on fig. 1 - we have $\sqrt{2}^2 = 2$ times

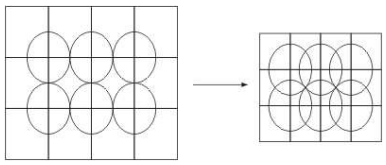

Figure 1: Rescaling the lattice without changing the size of magnetic dot.

more nodes, but we get some constrains - like in HS - so the capacity is now: $2 * 0.587 \cong 1.17$ - we get 17% of capacity.
We get it by more precise positioning of the head - it's technically easier to achieve than shrinking the dot. We will see that going further, we can increase the capacity potentially to infinity.

We can use statistical algorithm approach to generate random states (eg spin alignment) in statistical physics too. Usually we use Monte-Carlo methods - to generate "good" alignment we have to make many runs (the more, the better).
Now using for example many of this alignments, we can approximate its (local) statistical description with assumed "goodness" - we can generate so "good" alignments in one run using this statistical description.

In the **second section** we see how to analytically solve one dimensional model - find its optimal description and capacity - we will motivate that it should be Shannon entropy.
To find the optimum description we will just average over all elements.
In this case the statistical algorithm will be just Markov process - we will valuate node by node from left to right and the probability distribution for a node is found using only the valuation of the previous node.
Valuations of earlier nodes are now irrelevant, because they are separated by the previous one. This topological argument will be extended later.
In the **third section** it will be shown how to use statistical algorithm to deterministically encode data. To do this we will generalize binary system which from data without any redundances, generates sequence of uncorrelated 0 and 1 with probability $p(0) = p(1) = 1/2$. We need the same but with given different probabilities, which can change from node to node.
We could do it using Huffman coding, but presented algorithm is more precise and easier to use.
In the **fourth section** will be introduced formality for general models. It will be shown that for "reasonable" models - $X = \mathbb{Z}^n$, translative invariant constrains with finite range and which are "simple" - valuation of some nodes cannot enforce valuation of distant enough node - we can speak about its entropy, which is positive.
We will introduce here irreducibility of models too.
In the **fifth section** we will introduce methodology of statistical description.

Now we will straightforward define optimal description as the average over all elements. Unfortunately, in spite of giving many arguments, I couldn't prove existence of such average - we will assume it and show that it's equivalent vanishing of long-range correlation.
There will be introduced alternative definition of optimality (LOC) - in finite set of nodes, separated topologically from the rest of the space, all valuations are equally probable.
Then there will be discussed how to generate elements for given statistical description ('almost' statistical algorithm): fix some order for nodes, get probability distribution for a node using valuation of the previous ones.
In the **sixth section** we will analyze some algorithms as above, but this time there can be infinite number of previous nodes. We will assume that probability will be determined only by valuation of neighboring nodes.
We will analyze two simple types of algorithms:
- dividing the space into sets inside which constrains doesn't work and valuate them separately or
-take random order.
We will get near optimum and explain why we can't get to it in this way.
We will describe how to iteratively generate approximations of uniform distribution of elements on finite set. The longer we will iterate the process, the nearer uniform probability we get. We can use this generator to find numerically approximation of the optimal description.
In the **seventh section** it will be finally shown how to get as close as we want to the optimal algorithm. We will do it by narrowing space that it has only one infinite dimension and divide it into "straps", for which we can use one-dimensional analytical methods.
There will be shown numerical results, showing that we are really tending quickly to the optimum in this way.

2 One dimensional model

In this section we will look at the following model:

Definition 1. *One dimensional model* will be called a triplet: $(X, \mathcal{A}, M)$*:*
space $X = \mathbb{Z}$
alphabet $\mathcal{A}$ - finite set of symbols
constrains $M : \mathcal{A}^2 \to \{0, 1\}$

then *elements* of this model are $V(X)$*,* where for $A \subset X$*:*

$$V(A) := \{v : A \to \mathcal{A} | \forall_{i \in A \cap (A-1)} M(v_i, v_{i+1}) = 1\} \tag{1}$$

I will shortly justify, that any general one dimensional, translatively invariant model, can be easily reduced to above (with constrains range = 1):
Let $l+1$ be the largest range of constrains - for example $v_k = a \Rightarrow v_{k+l+1} = b$. Then we take as a new alphabet $\mathcal{A}^l$ - we group l consecutive symbols .
Now we can construct matrix as above:

$$M_{(v_i)_{i=1..l}(w_i)_{i=1..l}} = 1 \Leftrightarrow$$
$$\Leftrightarrow \quad (\forall_{i=2..l}\ v_i = w_{i-1} \text{ and sequence } v_1 v_2 ... v_l w_l \text{ is consistent with constrains})$$

So we can restrict to the model defined above ($l = 1$).

Let's think how much information we can store in such sequence of length k?
Denote $N_k = \#V(\overline{k})$ ($\overline{k} := \{0, 1, ..., k-1\}$) the number of such sequences of length k. We can store here information by choosing one of them.
To choose one of 2^n elements we need n bits. So in sequence of length k we can store about $\lg(N_k)$ bits.
The mean number of bits we can store in one node is: $H := \lim_{k\to\infty} H^k$

$$H^k := \frac{lg(\#V(\overline{k}))}{k} \tag{2}$$

Denote vector: $c_a^k := \#\{v \in V(\overline{k}) : v_{k-1} = a\} \qquad c^k := (c_a^k)_{a\in\mathcal{A}}$
Sequence of length $k+1$ we get from sequences of length k:

$$c^{k+1} = c^k \cdot M \tag{3}$$

Lets remind:

Theorem 2 (Frobenius - Perron)**.** *Irreducible matrix with real, nonnegative coefficients have dominant real, nonnegative eigenvalue. Corresponding eigenvector has real, positive coefficients.*

In our case reducibility would means that starting from proper subset of alphabet we have stay in it - we could decompose it into irreducible cases.
We can say:

$$H = lg\lambda \tag{4}$$

where λ is the dominant eigenvalue of M.

Let's look at corresponding eigenvector: $C^T M = \lambda C^T$
Normalize it: $\sum_{a\in\mathcal{A}} C_a = 1$.
We could think that it is probability distribution of symbols in elements...

It's not true: it's a distribution of symbols on the end of unbounded in one direction sequence.

We can use this probability distribution to initiate algorithm we will find.

Basing on this derivation we will find the probability distribution inside such sequences - unbounded in both directions. Now we have to expand in both of them: For some $(v_i)_{i=0..m-1}$, we define a matrix c_v^k:

$$\#(c_v^k)_{a,b} := \{w \in V(\overline{m+2k}) : w_k w_{k+1}...w_{k+m-1} = v,\ w_0 = a,\ w_{2k+m-1} = b\} \quad (5)$$

We will call v *a pattern*, its domain $(\overline{m})$ - its *shape* and the extreme nodes of domain - its *boundary*: $\grave{v} := v_0,\ \acute{v} := v_{m-1}$

Now $c_v^{k+1} = M \cdot c_v^k \cdot M$

For the dominant eigenvalue we have left/right eigenvectors:

$$C^T M = \lambda C^T \qquad MD = \lambda D \quad (6)$$

This time we take normalization $C^T D = 1$.
Because it has dominant eigenvalue - the rest of eigenvectors are orthogonal - for large k we can assume: $M^k \cong \lambda^k D C^T$:

$$c_v^k \cong \lambda^{2k} D C^T c_v^0 D C^T = \lambda^{2k} D C^T e_{\grave{v}} e_{\acute{v}}^T D C^T = \lambda^{2k} (D C^T) C_{\grave{v}} D_{\acute{v}} \quad (7)$$

because $(c_v^0)_{a,b} = 1 \Leftrightarrow a = \grave{v},\ b = \acute{v}$.
Let's look at the probability distribution of admissible patterns the given shape $(\overline{m})$ inside such infinite sequences.

$$p_v = \lim_{k\to\infty} \frac{\sum_{a,b\in\mathcal{A}} (c_v^k)_{a,b}}{\sum_{w\in V(\overline{m})} \sum_{a,b\in\mathcal{A}} (c_w^k)_{a,b}} = \frac{C_{\grave{v}} D_{\acute{v}}}{\sum_{w\in V(\overline{m})} C_{\grave{w}} D_{\acute{w}}} \quad (8)$$

Notice that we can forget about the normalization assumption in this equation.

We get

1. Patterns on the same shape and boundary values are equally probable. In other words - after fixing values on boundary of some set, probability distribution of valuations inside is uniform. As we will see later - it will generalize to higher dimensions.

2. For $m = 1$ we get the probability distribution of symbols:
$$p_a = \frac{C_a D_a}{\sum_{b\in\mathcal{A}} C_b D_b} = \frac{C_a D_a}{C^T D} \quad (9)$$

3. For $m = 2$ we get
$$p_{ab} = \frac{C_a M_{ab} D_b}{\sum_{a',b'\in\mathcal{A}} C_{a'} M_{a'b'} D_{b'}} = \frac{C_a D_b M_{ab}}{C^T M D} = \frac{C_a D_b M_{ab}}{\lambda C D} \quad (10)$$

Basing on points above, we can propose statistical algorithm, which allow us to generate elements with statistic just found:

Having fixed values of nodes left to given one, we want to determine its probability distribution.
From 1. cames that essential is only value of previous node - say a. Now using 2. and 3. we should valuate actual node to b with probability:

$$S_{ab} = \frac{p_{ab}}{p_a} = \frac{M_{ab}}{\lambda}\frac{D_b}{D_a} \tag{11}$$

This Markov process ($P(a \to b) = S_{ab}$) should be our probability algorithm.

We will show its optimality by showing that we get this way the same amount of information per node as available - model's entropy H. This means that restricting to elements generated in this way, we asymptotically won't loose capacity.

Firstly look at the problem: we have sequences of 0 and 1 in which the probability of 1 is fixed to some $p \in (0,1)$. Let's count how much information corresponds asymptotically to one digit. Denote $\tilde{p} := 1 - p$

$$\begin{aligned}\binom{n}{pn} &= \frac{n!}{(pn)!(\tilde{p}n)!} \cong (2\pi)^{-1/2}\frac{n^{n+1/2}e^n}{(pn)^{pn+1/2}(\tilde{p}n)^{\tilde{p}n+1/2}e^n} = \\ &= (2\pi np\tilde{p})^{-1/2}p^{-pn}\tilde{p}^{-\tilde{p}n} = (2\pi np\tilde{p})^{-1/2}2^{-n(p\lg p+\tilde{p}\lg\tilde{p})}\end{aligned}$$

$$H_p = \lim_{n\to\infty}\frac{\lg\binom{n}{pn}}{n} = -p\lg p - \tilde{p}\lg\tilde{p} =: h(p) \tag{12}$$

where we used Stirling's formula: $\lim_{n\to\infty}\frac{n!}{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n} = 1$

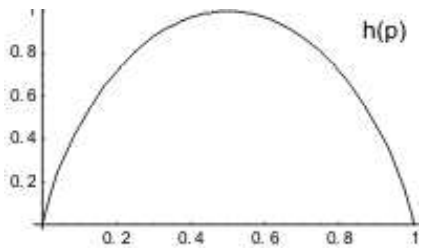

Figure 2: Entropy for fixed p.

That means that when we choose from all sequences, which number grow like 2^n, ones with given asymptotical probability of 1 (p), we get $2^{nh(p)}$ sequences. If we restrict to sequences with uncorrelated bits with $p = 1/2$ we can get every sequence with the same probability - the uniform distribution of sequences. While storing information in this distribution we get the maximum capacity: 1bit/digit.

Inserting some correlations, favoring some symbols, or part of the space would create redundancy.

Now when we have larger alphabet $\mathcal{A}$ and a probability distribution $p : \sum_{a\in\mathcal{A}} p_a = 1$, the mean number of information per symbol is (by induction)

$$H_p = -\sum_{a\in\mathcal{A}} p_a \lg p_a \tag{13}$$

it's the Shannon's entropy.
It is average capacity - simply: in a symbol with given probability distribution p we store H_p can bits.

In the case of found above Markov process:
$S = (S_{ab})_{a,b\in\mathcal{A}}$:

1. $\forall_{a,b\in\mathcal{A}}\ M_{ab} \geq S_{ab} \geq 0$
2. $\forall_{a\in\mathcal{A}} \sum_{b\in\mathcal{A}} S_{ab} = 1$
3. $pS = p,\ \sum_{a\in\mathcal{A}} p_a = 1$

The average amount of information per symbol is:

$$\begin{aligned}
H &= -\sum_a p_a \sum_b S_{ab} \lg S_{ab} = \\
&= -\sum_a \frac{C_a D_a}{CD} \sum_b \frac{M_{ab}}{\lambda}\frac{D_b}{D_a} \lg \frac{1}{\lambda}\frac{D_b}{D_a} = \frac{-1}{\lambda CD}\sum_{ab} C_a M_{ab} D_b \lg \frac{1}{\lambda}\frac{D_b}{D_a} = \\
&= \frac{CMD\lg\lambda}{\lambda CD} + \frac{1}{\lambda CD}\sum_{ab}(C_a(\lg D_a)M_{ab}D_b - C_a M_{ab} D_b(\lg D_b)) = \\
&= \lg\lambda + \frac{1}{\lambda CD}\sum_{ab}(C_a(\lg D_a)\lambda D_b - C_a\lambda D_b(\lg D_b)) = \lg\lambda
\end{aligned}$$

So encoding information this way gives us the maximum capacity.

We can use found method to analyze some **example**:

Definition 3. *k-model:*
$X = \mathbb{Z}\quad A = \{0,1\}\quad$ constrain*: after* 1 *follows k zeros.*

Because constrains have range $k+1$, we should group k elements into one of new symbols "0","1", ..."k":

- "0" : there were almost k zeros before (there can be 1 now)

- "i" : $k-i+1$ positions before was 1 (in i positions there can be 1)

In states different than "0", we have to put 0.
So the whole algorithm (Markov process) is the probability of putting 1 in state "0" - denote it q.

Denote p_i - probability of state i.

Make one step ($p := p_0$): $\begin{cases} p_k = pq \\ p_i = p_{i+1} \quad \text{dla } i \in \{1, ..., k-1\} \\ p = p_1 + p\tilde{q} \end{cases}$

So $p_1 = p_2 = ... = p_k = pq$

$p = 1 - p_1 - p_2 - ... - p_k = 1 - kpq \qquad p = \frac{1}{1+kq}$

So the entropy of the model is the maximum over all algorithms:

$$H = \max_{q\in[0,1]} H_q = \max_{q\in[0,1]} \frac{h(q)}{1+kq}. \tag{14}$$

In the introduction we've seen an example of usage of this methodology to storage more data on two dimensional plane.
We've just found analytic expression for the one dimensional case - we have constant length of "magnetic dot" - say d, but we don't assume that potential positions cannot intersect (if we assume that - we can store 1bit/length d). But we assume only that two 1 (magnetized dot) cannot intersect.
Let say that we can position the dot with precision $\frac{d}{k+1}$.
That means exactly that after 1 there have to be k zeros - analyzed model.
We can now easily count that using k times more precise positioning, we get:

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13
benefit(%)	0	39	65	86	103	117	129	141	151	160	168	176	183	190

more capacity of information. For larger k we get fig. 3

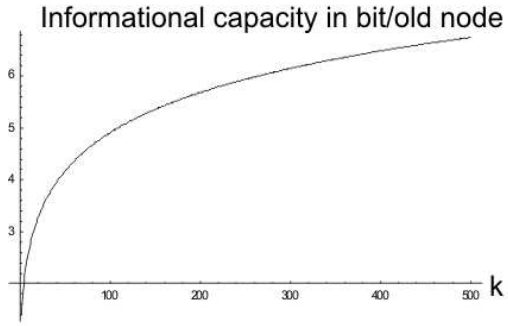

Figure 3: Informational capacity for rescaled k - models

It goes slowly to infinity with $k \to \infty$ - we can potentially increase the capacity of information any time by more precise positioning.

3 Encoding

We will now show how to use found Markov process (or generally - statistical algorithm) to deterministically encode some information without redundancy - a sequence of uncorrelated bits with $p(1) = 1/2$.
Using the data, we have to generate succeeding symbols with given probability distribution.
We can assume that we have $\{0, 1\}$ alphabet and digit 1 is chosen with probability $q \in (0, 1)$. In the general situation we create binary tree from symbols (like in Huffman) and use presented algorithm for each binary choice.

We will base on usual binary system - we will get it for $q = 1/2$.
Assume firstly that we have stored some information in natural number x. We now need:

$$x \quad \underset{\text{decoding}}{\overset{\text{encoding}}{\longrightarrow}} \quad (x', d) \tag{15}$$

We wants to encode bijectively some large number x into a digit d with fixed probability $(p(1) = q)$ and x' where the rest of information is stored.

Look at $\#\{k \in \mathbb{N} : k < x\} = k$. We will divide this set into two subsets: numbers from the first one will be translated to 1 and from the second - 0 and they should be distributed uniformly with given probability.
So there should be around qx in the first one.
We can take for example $x_1 := \lceil xq \rceil$ - the number of elements in the first subset $(d = 1)$
then $x_0 := x - \lceil xq \rceil$
So x is in first subset $\Leftrightarrow$ there is a jump of $\lceil xq \rceil$ after it:

$$d := \lceil (x+1)q \rceil - \lceil xq \rceil \tag{16}$$

直观感觉：
假设xq是一个自然数，则x处应该输入d=1
假设xq是2.1，(x+1)q是2.9，则x处应该d=0
假设xq是2.9，(x+1) q是3.1，则x处是d=1
最终，得出这个等价条件。

We've chosen d and came the subset corresponding do this digit - x' will be the position of x in that subset.

$$x' = x_d$$

for example for $q = 0.3$

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
x_0		0	1		2	3		4	5	6		7	8		9	10		11	12
x_1	0			1			2				3			4			5		

Asymmetric Binary System(非对称二进制系统，取0和1的概率分布非常不均匀)
https://en.wikipedia.org/wiki/Asymmetric_numeral_systems

这个公式可以转换为
ceil(a)-ceil(b)=ceil(a-ceil(b))
只要假设a=A+alpha，b=B+beta，其中alpha和beta介于[0,1)。
然后分四种情况来证明即可。

Decoding: we have d and x' and want to find x.
Denote $r := \lceil xq \rceil - xq \in [0,1)$
$d := \lceil (x+1)q \rceil - \lceil xq \rceil = \lceil (x+1)q - \lceil xq \rceil \rceil = \lceil (x+1)q - r - xq \rceil = \lceil q - r \rceil$

$$d = 1 \Leftrightarrow r < q$$

直观感觉：d=1时，说明x这个位置有一个d=1的输入，根据数组中概率的要求，必须x'=ceil(xq)
代入上面的表达式，得到x'=xq+r

When $d = 1$: $x' = \lceil xq \rceil = xq + r$
$x = \frac{x'-r}{q} = \left\lfloor \frac{x'}{q} \right\rfloor$ because it's natural number and $0 \le r < q$.

When $d = 0$ - $q \le r < 1$ so $\tilde{q} \ge 1 - r > 0$
$x' = x - \lceil xq \rceil = x - xq - r = x\tilde{q} - r$

直观感觉：仔细观察上面那个表格，到目前为止共计x个状态，除去ceil(xq)个的状态后，得到的就应该是d=0的状态数。
可以这样想：
因为处于当前这个状态x时，输入的符号是=0，因此ceil(xq)应该是d=1的输入数目与1的和。再考虑到d=0的输入中，状态号是从0开始计数的，它需要排除，最终可得x'=x-ceil(xq)

$$x = \frac{x'+r}{\tilde{q}} = \frac{x'+1}{\tilde{q}} - \frac{1-r}{\tilde{q}} = \left\lceil \frac{x'+1}{\tilde{q}} \right\rceil - 1$$

表达式的最后一项(1-r)/(1-q)是一个大于1的数

Finally **decoding**:

$d=0$	$d=1$
$x = \left\lceil \frac{x'+1}{\tilde{q}} \right\rceil - 1$	$x = \left\lfloor \frac{x'}{q} \right\rfloor$

For $q = 1/2$ it's usual binary system (with switched digits).

Digression: for large x it contains about $\lg(x)$ bits of information.
That means that $x \to x'$ with probability q takes $\lg(q)$ bits ($\lg(x/q) = \lg(x) - \lg(q)$),
with probability $\tilde{q}$, $lg(\tilde{q})$ bits.
So average $(-q\lg(q) - \tilde{q}\lg(\tilde{q}))$ bits - it's the same amount of information we put in d.

We know now to encode large (but finite) numbers (x).
We can use it to encode data stream - potentially infinite sequence of bits.
We will use a buffer with large $x \in I := [2^n, 2^{n+1} - 1]$ for fixed $n \in \mathbb{N}$ and ensure that we stay in that range.

In this case **stream coding** looks:

1. $(t, d) =$ coding (x) $\quad$ {so $t < x$}
2. Use d $\quad$ {for example to generate symbol}
3. $k = n - \lfloor \lg t \rfloor$ $\quad$ {how many bits we have to get from input}
4. $x = t \cdot 2^k + 2^{k-1}b_0 + 2^{k-2}b_1 + ... + b_k$ $\quad$ {input bits, chronologically}

stream decoding(d):

1. While $(t = \text{decoding"}(x, d)) \geq 2^{n+1}$:
   put to output the youngest bit of x, $x = \lfloor x/2 \rfloor$
2. $x = t$

It's easy to check that they are inverses of each other and that x stays in I.
Initialization: we start coding with getting first n bits from input and finish decreasing x to 0 using "old" coding (ignore step 4).
Decoding: start with "old" decoding from $x = 0$ as long as $x < 2^{n+1}$ and finish - put x (n bits) to output.

Given algorithm has practically (amortized) constant cost of a step - its time and memory cost are very low. It has similar results as Huffman coding, but there we can only approximate given q with powers of 2, and here we have simple exact and more universal algorithm.
The problem is (like in Huffman) that one error practically destroys whole further information. It's because we have no redundancy - but we can artificially add some to protect ourselves.

4 Information capacity of general models

In this section we will give formalism and methodology for general models. There is proven existence of entropy in for large class of models too.

Definition 4. *Model* will be called a triple $(X, \mathcal{A}, M)$
space $X \subset \mathbb{Z}^m$
alphabet $\mathcal{A}$ -finite set of symbols
constrains N(by forbidden valuations):
$N \subset \{((p_1, p_2, ..., p_k), (s_1, s_2, ...s_k)) : k \in \mathbb{N} \quad \forall_i \; p_i \in X \; s_i \in \mathcal{A}\}$

them *elements of model* are $V(X)$, where for $A \subset X$ its *valuations* are:

$$V(A) := \{v : A \to \mathcal{A} | \forall_{(p,s)\in N} \; \exists_i \; v_{p_i} \neq s_i\} \qquad V := \bigcup_{A\in X:\ \#A<\infty} V(A)$$

Digression: it is a very general definition. Instead of defining by forbidden states we can do it by allowed ones (for example - take large enough subset of X and take all allowed valuations).
We can write for example tiling problems in that formalism (denote each title with separate symbol and put their shapes in "allowed constrains"). We know that there can happen very different situations - even enforcing unperiodic tiling.
To control our model we will have to limit this class.

The main example we will use is the Hard Square model.

Definition 5. *Hard-square model (HS):*

$$(X=\mathbb{Z}^2, \mathcal{A}=\{0,1\}, N=\{(((i,j),(k,l)),(1,1)) : i,j,k,l \in \mathbb{Z}, |i-k|+|j-l|=1\})$$

In each node of two dimensional lattice we put 0 or 1 such that there are no neighboring ones.
It is one of the simplest models which aren't solved analytically, In [5] we can find exact formula for entropy of similar Hard Hexagon Model (we add upper left and lower right to neighborhood), but this methodology creates some unsolvable singularities for HS.
In [6] 43 digits of this entropy is found - I will use it to evaluate algorithms. Unfortunately this methodology cannot be used to find statistics.

Define:

Definition 6.
Neighborhood of $x \in X$*:* $N_x := \{y \in X : \exists_{((z^1,...,z^k),(a_1,...,a_k))\in N}\ \exists_{i,j}\ z^i = x,\ z^j = y\}$
Range of constrains: $L := \max\{|y_i - x_i| : x \in X,\ y \in N_x,\ i \in \mathbb{N}\}$
Interior of $A \subset X$ *:* $A^- := \{x \in A : N_x \subset A\}$
Edge of $A \subset X$ *:* $A^o := A \backslash A^-$
Thickening of $A \subset X$*:* $A^+ := \bigcup_{x\in A} N_x$
We will call set $A \subset X$ *connected,* if

$$\forall_{x,y\in A}\exists_{n\in\mathbb{N},z^0,...,z^n\in X} \quad z^0 = x,\ z^n = y,\ \forall_i \sum_j |z_j^i - z_j^{i+1}| = 1.$$

Of course $A^{-+} \subset A \subset A^{+-}$.

For translative invariant models, we need to know only the neighborhood around any point, eg for HS: $N_x = x + \{(1,0),(0,1),(-1,0),(0,-1)\}$.

Digression: neighborhood is always symmetric set ($x \in N_y \Leftrightarrow y \in N_x$), $x \in N_x$, so

$$\rho(x,y) = \min_n \exists_{(x^0=x,x^1,...,x^n=y)}\ \forall_i x^{i+1} \in N_{x_i}$$

is natural metric for a given model.

HS model has many symmetries: generated by translations, axial symmetry and rotations by $\pi/2$.

Definition 7. A bijection $S : X \hookrightarrow X$ will be called *symmetry* if

$$((x^1,...,x^k),(a_1,...,a_k)) \in N \Leftrightarrow ((S(x^1),...,S(x^k)),(a_1,...,a_k)) \in N \qquad (17)$$

For the rest of this paper we assume, that:

- $X = \mathbb{Z}^m$
- has finite range of constrains
- is translational invariant - every translation is symmetry($t_x(y) := y + x$)
- is simple(#):

Definition 8. We call model *simple(#),* if $\#V(X) > 1$
and there exists $N > n \geq 0$:

$$\forall_{\text{connected } A \subset X} \ \forall_{v \in V((A^N)^o)} \ \forall_{u \in V(A^n)} \ \exists_{w \in V((A^N)^- \setminus A^n)} \quad u \cup w \cup v \in V(A^N)$$

where $A^0 := A,\ A^{i+1} := (A^i)^+$.

In another words: for sets "nice" enough (of the form A^n), exists thickening ($N - n$ times) that for any valuation of its boundary, every valuation of the A^n is still allowed.

For example models with neutral symbol - which isn't in any constrain (we can always use it - 0 in HS) are simple(#): $n = 0,\ N = 2$ - we fill $A^+ \setminus A$ with this symbol.

To the end of this paper we use notation:

$$N_u^A \equiv N(A, u) := \#\{w \in V(A) : w|_{D(u)} = u\}, \quad N(A) \equiv N(A, \{\}). \tag{18}$$

Lemma 9. *Denote* k-block*:* $\beta_k := \{0, 1, ..., k-1\}^m$*,* now:

$$\exists_{n' \geq \max(L,1)} \forall_{v \in V(\beta^+_{n'} \setminus \beta_{n'})} N(\beta^+_{n'}, v) > 1.$$

Simply speaking: for every valuation of neighborhood of large enough block we can valuate it in more than one way.
Proof: Take k: $\#V(\beta^k) > 1$ ($\#V(X) > 1$).
β_k^{N-1} can be placed in some n' - block ($n' > L$). □

Theorem 10. *For models as above, there exists entropy and is positive:*
There exists increasing sequence of sets (A_i)*:* $A \subset X,\ \#A < \infty,\ \bigcup_i A_i = X$*, that exists limit:*

$$0 < H \equiv \lim_{i \to \infty} \frac{lg(N(A_i))}{\#A_i} \leq \lg \#\mathcal{A}$$

Prove:
Take n' like in the lemma. We will operate on *blocks*: $B \equiv \beta_{n'}$ placed in nodes of lattice $n'\mathbb{Z}^m$.
Because $n' \geq L$, valuations of a block can be restricted only by valuations of adjoined blocks: $B + n'(\{-1, 0, 1\}^m \setminus \{0\})$.

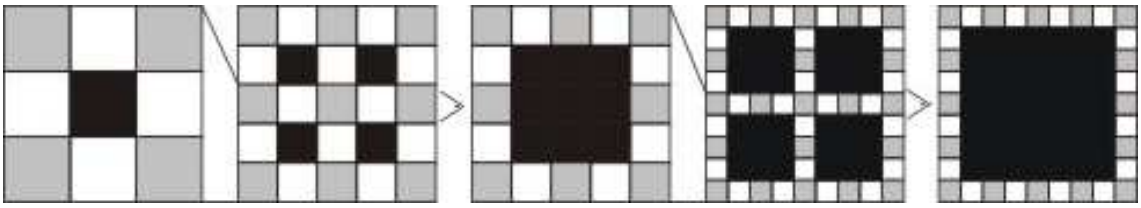

Figure 4: Block division for $A_1,\ C'_2,\ A_2,\ C'_3,\ A_3$. Black - "essence". The rest (filler) we valuate such that all blocks denoted with the same color has the same pattern.

First of all we will find the filling pattern - periodic valuation of the space.

Take the lattice $Y = 2n'\mathbb{Z}^m$ and numerate anyhow $\{0,1\}^m = \{x^i\}_{i=1,...,2^m}$.
Now every of blocks from $Y + B + x^1$ can be valuated independent from the other ($n' \geq$ range of constrains), choose any $w_1 \in V(B)$.
Now using lemma 9, after valuating that blocks we can find some $w_2 \in V(B)$ for $Y + B + x^2$ not colliding with all w_1. And so on we get the universal periodic filling:

$$V(X) \ni w : \forall_{y\in Y, x\in B, i\in 1,...,2^m} \quad w(y + x + x^i) := w_i(x).$$

Now we can go the main construction.
As the sequence we are looking for take

$$A_i := B + n'\{0, ..., 2^i\}^m, \qquad H_i := \frac{\lg(N(A_i))}{\#A_i}$$

Notice that $H_i \leq \#\mathcal{A}$ - we would get that entropy without constrains.
To prove that H_i has a limit, we will construct increasing sequence H'_i, that $\forall_i\ H'_i \leq H_i \leq \#\mathcal{A}$ and show that $\lim_{i\to\infty} H'_i - H_i = 0$.
Monotone and bounded sequence has the limit, so H_i has the same.

We will make some construction to ensure the monotonicity:

$$C_i := B + n'\{x \in \{0, ..., 2^i\}^m : \exists_i x_i \in \{0, 2^i\}\}$$

external blocks of A_i which will be filled with w.
We will need intermediate step:

$$C'_{i+1} := B + n'\{x \in \{0, ..., 2^i\}^m : \exists_i x_i \in \{0, 2^i, 2^{i+1}\}\}$$

$$H'_i = \frac{\lg(N(A_i, w|_{C_i}))}{\#A_i} \qquad H''_{i+1} = \frac{\lg(N(A_{i+1}, w|_{C'_{i+1}}))}{\#A_{i+1}} \tag{19}$$

Of course $H''_i \leq H'_i \leq H_i$.
Now the "essence" of $A_{i+1}\backslash C'_{i+1}$ is made exactly of 2^m "essences" from the previous step, with the same valuation of surrounding blocks.

So $N(A_{i+1}, w|_{C'_{i+1}}) = 2^m N(A_i, w|_{C_i})$, $\#A_{i+1} = \left(\frac{2^{i+1}+1}{2^i+1}\right)^m \#A_i < 2^m \#A_i$
We get: $H'_i < H''_{i+1}$,

$$H'_i < H''_{i+1} \leq H'_{i+1} < ... \leq \#\mathcal{A}.$$

There left only to show, that $\lim_{i\to\infty} H_i - H'_i = 0$

Look at $D_i := (A_i \backslash C_i)\overbrace{--...-}^{N-1}\overbrace{++...+}^{n}$.
$(A^{-+} \subset A)$, so from (#):
we can freely valuate D_i,independently to valuation on C_i.
Because $\beta_k^- \supset \beta_{k-2L} + (l, l, ..., l)$, so $\beta_{2^i-2NL} + (NL, ..., NL) \subset D_i$,
for large i:

$$\#D_i \sim (2^i)^m \qquad \#(A_i \backslash D_i) \sim (2^i)^{m-1}$$

So $\lim_{i\to\infty} \frac{\#(A_i\backslash D_i)}{\#D_i} = 0$.

We have $N(D_i) \leq N(A_i, w|_{C_i} \leq N(A_i)$
But $\frac{N(A_i)}{N(D_i)} \leq (\#\mathcal{A})^{\#(A_i\backslash D_i)}$ - equality would be without constrains.
So $\frac{\lg N(A_i) - \lg N(D_i)}{\#A_i} \leq \frac{\#(A_i\backslash D_i)\lg(\#\mathcal{A})}{\#A_i} \to 0$.

From three sequences: $\lim_{i\to\infty} H_i - H'_i = 0$ $\square$.

So: for large class of models we can speak about its entropy.

For the rest of this work we add assumption of *irreductibility*:

Definition 11. We call translational invariant model *irreducible,* if:

$$\bigcup_i A^i = X$$

Where $A^0 := \{0\}$, $A^{i+1} := (A^i)^+$.

If a model isn't irreducible (is reducible): $Y := \bigcup_i A_i \neq X$
Because $Y = -Y$, $Y = Y + Y$, so $y \in Y \Rightarrow y\mathbb{Z} \subset Y$ - Y is periodic lattice - there exists linearly independent $y^1, .., y^{m'} \in X$, such that:

$$Y = \sum_{i=1,...,m'} \mathbb{Z} y^i$$

Now if we make a transformation: $y^i \to (\delta_{ij})_{j=1,...,m'}$ we get the corresponding m'-dimensional irreducible model.

$x \sim y \Leftrightarrow x - y \in Y$ is equivalence relation, so $X = \bigcup_i x^i + Y$ (disjoint sum for some (x_i)) can be threaded as identical, independent lattices.
Now for example average entropy is the same as for reduced model.

5 Statistical approach

We will want now, similarly as in one-dimensional case, to find *optimal statistical description*, by averaging over all valuations.

Definition 12. *Statistical description* is function:

$$p : \{(A, f) : A \subset X,\ \#A < \infty,\ f : A \to \mathcal{A}\} \to [0, 1]$$

such that

$$\forall_{A\subset X,\ \#A<\infty} \forall_{x\in X\setminus A} \forall_{f:A\to\mathcal{A}} \sum_{a\in\mathcal{A}} p_{f\cup\{(x,a)\}} = p_f \tag{20}$$

$$p_{\{\}} = 1$$

where $p_f \equiv p(D(f), f)$, $\quad D(f)$ - domain of f.

It gives for each shape A, the probability distribution of valuations on this shape. Because of translation invariance, it doesn't depend on its position - for example $p(01)$ denotes the probability that when choosing a node, it and its right neighbor are valued correspondingly to 01.
The conditions (20) ensure normalization to 1. (eg p(01)+p(00)=p(0)).

Now we would like to take an average over elements, but we can count only valuations on finite sets - we have to choose some sequence of finite sets tending to whole space.

Definition 13.
Normal sequence of sets we call series of finite sets $(A_i)_{i\in\mathbb{N}}$ such that:

$$A_0 \subset A_1 \subset A_2 \subset ...$$

$$\bigcup_{i\in\mathbb{N}} A_i = X.$$

For $f, v \in V,\ A \subset X\ :\ D(v) \cap D(f) = \emptyset,\ D(f) \subset A,\ D(v) \subset A$
we call (A, v) *approximation of optimal description:*

$$p_f^{A,v} := \frac{N(A, v \cup f)}{N(A, v)}. \tag{21}$$

Denote: $p_f^A := p_f^{A,\emptyset}$.

Now for some $B \supset A,\ v \in V(B^o),\ D_f \subset A^-,\ f \in V(A)$:

$$p_f^{B,v} = \frac{N(B, v \cup f)}{N(B, v)} = \frac{\sum_{u \in V(A^o \setminus D_v)} N(B \setminus A^-, u \cup v) N(A, u \cup f)}{N(B, v)} =$$
$$= \sum_{u \in V(A^o \setminus D_v)} p_f^{A,u} \frac{N(B \setminus A^-, u \cup v) N(A, u)}{N(B, v)} = \sum_{u \in V(A^o \setminus D_v)} p_f^{A,u} p_u^{B,v} \quad (22)$$

We've divided the sum over all valuations into sum within and outside topologically separating set A^o.
We want to find optimal description p_f^o as the limit of succeeding approximations $p_f^{A,v}$ for sets from a normal sequence.
We've just shown that we get approximation for some set, as weighted average $\left(\sum_{u \in V(A^o \setminus D_v)} p_u^{B,v} = 1\right)$ of approximations from the previous one for the same pattern f.
So going to the next set don't give worse approximation:

$$A \subset B \Rightarrow \hat{p}_f^A \geq \hat{p}_f^B \geq \check{p}_f^B \geq \check{p}_f^A$$

where

$$\check{p}_f^A := \min_{v \in V(A^o)} p_f^{A,v} \qquad \hat{p}_f^A := \max_{v \in V(A^o)} p_f^{A,v} \qquad d_f^A := \hat{p}_f^A - \check{p}_f^A$$

We've explained that d_f^A isn't growing in normal sequence. If we would prove that for some normal series (A_i), $d_f^{A_i}$ is decreasing to 0, than taking any other normal sequence (B_i), because $\forall_i \exists_j\ A_i \subset B_j$, $\lim_{i\to\infty} p_f^{A_i} = \lim_{i\to\infty} p_f^{B_i}$.
So this limit would be the only reasonable optimal description.

Unfortunately I cannot prove formally its existence. I have to assume it:

Assumption 14 (*). *$d_f^{A_i} \to 0$ for some normal sequence (A_i) and any pattern f.*

We can now define:

Definition 15. *Optimal description,* to $p_f^o := \lim_{i\to\infty} p_f^{A_i}$
where (A_i) - any normal sequence.

Let's show that optimal description preserves symmetries
S - a symmetry of model
$S' : V(X) \to V(S(X)) = V(X) : S'(v)(x) = v \circ S^{-1}$ - bijection on $V(X)$
Now for a normal sequence (A_i) and some pattern f, take (using freedom of choice) normal sequence $(B_i) : B_i = S(A_i)$ and pattern $S'(f)$:

$$p_f^o = \lim_{i\to\infty} \frac{\#\{w \in V(A_i) : w|_{D(f)} = f\}}{\#\{w \in V(A_i)\}} =$$
$$= \lim_{i\to\infty} \frac{\#\{S'(w) \in V(B_i) : S'(w)|_{D(S'(f)) = S(D(f))} = S'(f)\}}{\#\{S'(w) \in V(B_i)\}} = p_{S'(f)}^o$$

Observation 16. *For any symmetry of model S and patter f*

$$p^o_f = p^o_{S'(f)}.$$

We will now discuss *local optimality condition* (LOC).
While having finite space, we have finite (N) number of elements - we can store there the largest number of information ($\lg(N)$), if they have uniform probabilistic distribution - we don't favor any.
We have analog to this condition in infinite space: when we valuate boundary of some finite set, all available valuations of its interior are equally probable - its LOC.

Observation 17. *Optimality of statistical description p is equivalent to LOC(Local Optimality Condition):*
for any $B \subset X,\ \#B < \infty,\ A \subset B^-,\ f \in V(B)$:

$$p(f) = \frac{p(f|_{B\setminus A})}{N(B, f|_{B\setminus A})}$$

Proof:

1. Take $p = p^o,\ (A_i)$ - normal sequence, $A,\ B,\ f$ like above

   $$N(A_i, f|_{B\setminus A}) = \#\{u \in V(A) : u \cup f|_{B\setminus A} \in V(B)\} N(A_i, f) = N(B, f|_{B\setminus A}) N(A_i, f)$$

   $$p^o_f = \lim_{i\to\infty} \frac{N(A_i, f)}{N(A_i)} = \lim_{i\to\infty} \frac{N(A_i, f|_{B\setminus A})}{N(B, f|_{B\setminus A}) N(A_i)} = \frac{p(f|_{B\setminus A})}{N(B, f|_{B\setminus A})}$$

2. assume now LOC for $p:\ B \subset X:\ \#B < \infty,\ D(f) \subset A = B^-$

   $$p(f) = \sum_{v \in V(B^o)} p(f \cup v) = \sum_{v \in V(B^o)} N(B, v \cup f) \frac{p(v)}{N(B, v)} = \sum_{v \in V(B^o)} p(v) p^{B,v}_f \quad (23)$$

   we get the first equality using normalization (20), the second from LOC (we can fill $A\setminus D(f)$ in $N(B, v \cup f)$ ways)
   Taking as B succeeding elements of some normal sequence, we get thesis. □

Remarks

1. for pattern $f,\ A \subset X : D(f) \subset A^-$ we have(23):

   $$p^o_f = \sum_{v \in V(A^o)} p^o_v p^{A,v}_f$$

   Now take for example sequence $A_0 := 0,\ A_{i+1} := A_i^+$, form (*): $p^{A_i,v}_f \to p^o_f$ - dependence of probability distribution of distant nodes decrease:
   Assumption (*) is equivalent vanishing of long distance correlations.

2. LOC is equivalent *pLOC - point local optimality condition*:

$$\forall_{B:N_0^o\subset B\subset X\setminus\{0\},\ \#B<\infty}\forall_{v\in V(B)}\forall_{a,b\in\mathcal{A}}\ p_{v\cup\{(0,a)\}}=p_{v\cup\{(0,b)\}}$$

where $N_0^o=N_0\setminus\{0\},\ N_0$ - neighborhood of 0.

It gives smaller set of conditions for optimality - we have only to check LOC for $\#A=1$.
We will use it later to generate approximations of the uniform distribution over elements.
It says for example for HS that for any finite pattern:

$$p\begin{pmatrix} x & 0 & x\\ 0 & 0 & 0\\ x & 0 & x\end{pmatrix}=p\begin{pmatrix} x & 0 & x\\ 0 & 1 & 0\\ x & 0 & x\end{pmatrix}$$

Where "x" denotes valuations somewhere outside N_0.

formally:

$$\forall_{A\subset X\setminus N_0:\ \#A<\infty}\forall_{v\in V(A)}\ p_{f_0^*\cup v}=p_{f_1^*\cup v}$$

where $f^*:=0|_{N_0\setminus\{0\}}\ ,\ f_a^*:=f^*\cup\{(0,a)\}$
Other valuations of N_0^o enforce 0 in the middle.

Proof: by induction for $\#A$: for $\#A=1$ - pLOC
assume we've proved LOD for $\#A=k-1$
Take some $A,B,v:\#A=k,\ B\supset A^+,\ v\in V(B\setminus A)$
We want to show that for any $f,g\in V(A)$

$$p_{v\cup f}=p_{v\cup g}$$

If there exists $x\in A$ such, that $f(x)=g(x)$, than we move x to B and use induction assumption.
If not, we make intermediate step to $f'=f\setminus\{(x,f(x)\}\cup\{(x,g(x)\}$ for some $x\in A$. $\square$

3. take sequence $A_0:=\{0\},\ A_{i+1}:=A_i^+$
than:

$$\lim_{i\to\infty}\frac{\lg(N(A_i))}{-\sum_{v\in V(A_i)}p_v^o\lg p_v^o}=1$$

Proof: set $i\in\mathbb{Z}$
Take any valuation of A_{i+1}^o , we have uniform distribution of available valuations on A_i. Using (#) (n i N), we can freely valuate A_{i-N+n}, so we have more valuations than $N(A_{i-N+n})$:

$$-\sum_{v\in V(A_i)}p_v^o\lg p_v^o\geq\lg N(A_{i-N+n})$$

Now we repeat discussion from the end of proof of theorem 10 ($\lim_{i\to\infty}\frac{\#A_{i-N+n}}{\#A_i}=1$) and get the thesis. $\square$

So we've justified that **optimal description has the same mean capacity as the model.**

In statistical description we have some excessive information because of the normality conditions.
We can get rid of them in eg two ways:

1. We can limit to patterns without fixed symbol (a). Eg for HS - $s : A \to p_{0|_A}$
   Proof: We want to get probability of some f with $k+1$ appearances of a, eg: $f(x)=a$, now:

$$p_f = p_{f\setminus\{(x,a)\}} - \sum_{b\in\mathcal{A}\setminus\{a\}} p_{f\setminus\{(x,a)\}\cup\{(x,b)\}} \ .$$

2. *Sequential description :*
   Take any numeration of nodes of space: $\{x^i\}_{i=0,1,\dots\infty}=X$
   and fix some $a\in\mathcal{A}$

$$\forall_{v=(v_0,v_1,\dots,v_{k-1})\in\mathcal{A}^k}\ \forall_{b\in\mathcal{A}\setminus\{a\}} \quad q_b(v) := \frac{p_{f_v\cup\{(x_k,b)\}}}{p_{f_v}}$$

   where $D(f_v) := \{x_i\}_{i=0,\dots,k-1},\ f_v(x^i) := v_i$

   We are taking successively x^i and using valuation of previous nodes we get its probability distribution of valuations .

   Proof: We want to find some p_f,
   $\exists_k\ \{x_0,...,x_k\}\supset D(f),\ A=\{x_0,...,x_k\}$

$$\forall_{u\in V(A)}\ p_u = \prod_{i=0}^{k} q_{u(x_i)}((u(x_0),...,u(x_{i-1})) \qquad \left(q_a(w)=1-\sum_{b\in\mathcal{A}\setminus\{a\}} q_b(w)\right)$$

$$p_f = \sum_{u\in V(A\setminus D(f)):u\cup f\in V(A)} p_{f\cup u} \ .$$

   In the next section we will forget about assumption that there are only finite number of previous nodes.

So we are able to generate elements with given statistical description - visit successively x^i and generate its valuation with appropriate distribution.

Digression: assume now that we have some element f generated with uniform distribution (eg generated using optimal algorithm).
So for a given shape A, value of $f|_A$ should be given by optimal description. Than assuming vanishing of long-range correlation - we should get optimal description by "averaging" $f|_{A+x}$ over $x \in nX$ where n is some large number. The same would be for any translations of nX, so:
we can get optimal description from any random element: by taking "average" of $f|_{A+x}$ over all points of the space: $x \in X$.

6 Statistical algorithms

Let's say we have a statistical description (the best would be optimal). Now we want to construct an element using this statistic. If it would be optimal - we get this way uniform distribution of elements - we can store the same amount of information as model's capacity.

We can use sequential approach like in previous section, but it would favor some points (eg first). We would like to use transitional invariance of the space - we cannot assume that there were only finite number of previous points.
We are still assuming that long range correlations vanishes, so as an approximation of optimal algorithm, we can assume that probability distribution for a given point depends only on valuations of neighboring ones.

Definition 18. *Statistical algorithm* call a pair $(<, q)$*:*

- " $<$ " $\subset X^2$ - linear order X
- $q_a : \{(x, v) : v \in V(x_<)\} \to [0, 1] \qquad (v \cup \{(x, a)\} \notin V \Rightarrow q_a(x, v) = 0)$

such that $\sum_{a \in \mathcal{A}} q_a((x, v)) = 1$
where $x_< := \{y \in X : y < x\}$.

In practice q will depend only on neighboring nodes.
Algorithm must be consistent with model - some q are enforced to 0.

We practically cannot just "start" algorithm with infinite number of previous nodes - we usually need some initialization - it will be discussed on the end of next section.
While generating element, we will have some mean entropy per choice (node) - we will count it on examples.
By *optimality of algorithm* we will understand: how distant is the capacity

given by algorithm to the real capacity (model's entropy). For many dimensional models we usually won't be able to construct optimal algorithm, only approximate it.

We analyze now two simple algorithms for Hard Square model.

Algorithm I: Filling over independent sets

Divide the space into separate subsets Y_i, inside which constrains doesn't work (each valuation is allowed).
For example: generally $Y = L\mathbb{Z}^m,\ I = \{1, ..., L^m\},\ \{x^i\}_{i\in I} = \{0, ..., L-1\}^m$
(L -range of constrains)
Now $Y_i := Y + x^i$

For HS we can take $Y_i = \{(i, j) : \mathrm{mod}(i + j, 2) = i\}$ - nodes with even/odd sum of coordinates.

Algorithm: fill Y_0 with the same probability distribution (eg: with probability q put 1). Now the same with Y_1 (with probability q') - but this time only some of nodes can be value to 1. This time valuations don't influence any nodes - we can store here as much information as possible - take $q' = 1/2$.
So our algorithm is described by one number: q.

Let's count entropy: $H_q = \frac{1}{2}h(q) + \frac{1}{2}(1-q)^4$
Because in the half of nodes we get entropy $h(q)$ and in the rest, if it's possible (all neighbors have zeros - probability $(1-q^4)$ 1bit/node.

So the best capacity we can achieve this way is $\max_q H_q \cong H - 0.0217$
So we lose about ΔH =0.0217 bit/node: 4%.

Why we couldn't get optimum?
We can take q to fulfill relation:

$$(p_* :=)p\begin{pmatrix} & 0 & \\ 0 & 0 & 0 \\ & 0 & \end{pmatrix} = p\begin{pmatrix} & 0 & \\ 0 & 1 & 0 \\ & 0 & \end{pmatrix},\ \text{but then eg } p\begin{pmatrix} 1 & 0 & \\ 0 & 0 & 0 \\ & 0 & \end{pmatrix} > p\begin{pmatrix} 1 & 0 & \\ 0 & 1 & 0 \\ & 0 & \end{pmatrix}$$

Explanation:
If the middle node is in Y_1 - we have equality in both cases($q' = 1/2$).
We get strong inequality if the middle is in Y_0.
Probability that we value it to 1 is q (probability on the right side).
If we value it to 0 we have to additionally value its neighbor to 0, which is more probable when we fix 1 in the corner.

Remind that we have countable number of equalities to fulfil (pLOC), so usually finite numbers of parameters won't be enough - we rather cannot get practical optimal algorithm, only its approximation.

Element generator

To study next algorithm, we would need to generate valuations on finite set(A) with probability distribution close to uniform.
We could do it using approximation of statistical algorithm, but in presented here method, we don't need it to get to uniform distribution as close as we want and it doesn't favor any statistics.
The disadvantage to the statistical algorithm is that this time we have to visit every node many times(using this method to coding is highly unpractice).
We can use this generator for example to find optimal statistics, but it occurs that the tendency is slow .

It is based on pLOC - we will equalize iteratively this probabilities, by allowing to some fluctuations.
For HS after any initialization from $V(A)$ (eg using some statistical algorithm) repeat many times:
choose randomly a point in A, if it's 4 neighbors has values 0 then change its value ($0 \leftrightarrow 1$)

Let's explain why we will tend to unciform distribution.
We are making some Markov process on $V(A)$. We can go from $f \in V(A)$ to $g \in V(A)$ in one step if they different in one node. Then the probability of this transition is $\frac{1}{\#A}$. The same for the opposite direction.
So the stochastic matrix describing this process is symmetric - bistochastic - $(1, 1, ..., 1)$ is the dominant eigenvector corresponding to eigenvalue 1 (matrix is irreducible).
So iterating this process we tend to the uniform distribution.

Generally we see that any bistochastic process would be appropriate.

Practically I've used about 2-5 $\#A$ iteration to get first valuation and than to get next uncorrelated about $\#A$ iterations.

We can now analyze "intuitively optimal" algorithm - with random order.

Algorithm II: Random seed

For every node take a random real number from [0,1] with uniform distribution:

$$t : X \to [0, 1]$$

It defines our order: $x < y \equiv t(x) < t(y)$

So in the following step of algorithm we will take random, unvisited yet node:
- if it has a neighbor valuated to 1 - we valuate it to 0
- else - we valuate it to 0 or 1 with given probability.

We could chose this probability as a constant, but we will be more sophisticated. When we are in a point with some t, that means that statistically t of nodes were already visited. So we can take some function:
charging profile - $q : [0, 1] \to [0, 1]$ - if we are in node t and it's consistent with constrains, with probability $q(t)$ value it to 1.

To count entropy, we need to find for a given q second function:
$a : [0, 1] \to [0, 1]$ - probability that a node with given t we can still valuate to 1.

Assume that a, q - continuous.
Now entropy will be:

$$H_q = \int_0^1 a(t)h(q(t))dt \tag{24}$$

Notice that for optimality we would need:

- $a(0) = 1$ - we have no 1 yet
- $q(1) = \frac{1}{2}$ - this 1 are not blocking anything
- $a(1) = 2p_*^o$ - they are neighbored by 4 zeros
- $q(0) = p_*^o$ - this elements will be 1 for sure ($a = 1$)

Unfortunately finding a from q seems very difficult, so finding optimal q seems even worse.

We can find it numerically, using Monte-Carlo type method:
On some finite set (say: a square), generate different random orders and valuations with uniform distribution (using found generator) and take the averages to find q and a.

On fig.5 are shown results ($A = \{1, ..., 300\}^2$, 4000 measures).
This graphs are very blurred. It's because we've simplified: q parameter should depends on the order of neighboring nodes.

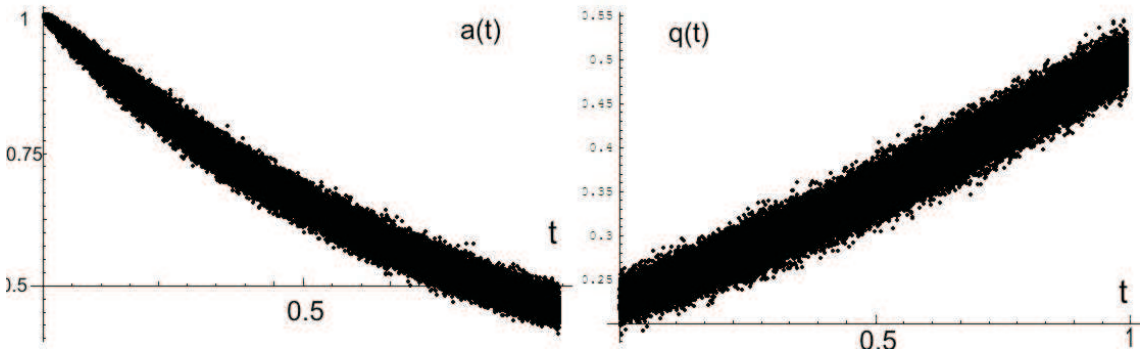

Figure 5: Numerically found q i a.

This algorithm looks translative invariant, but only choosing the order is so.

After fitting 4th order polynomials and integrating (24), I've got entropy larger then model's. It's a lesson that found a doesn't corresponds to q now.
We can do it exactly by generating valuations using found q: I've got about $\Delta H \cong 0.01 - 0.02$bit/node.

7 The way to optimum

In this section will be shown practical method which (if (*) is true) can gives us description as near to optimal as needed. We can use it to encode information with practically real model's capacity.
We will show numerical results for HS - very good tendency to optimum.

The idea is to approximate model to be able to use one-dimensional solution:

1. Approximation of model - all dimensions but one (we will call it *essential*) we shrink to finite width (n),
2. New alphabet - all consistent valuations of cross-section orthogonal to essential direction of width: range of constrains - 1 ($L - 1$),
3. Transfer matrix: introduce (transfer) matrix of available succeeding (in essential direction) new symbols,
4. Solution to one dimensional model - as in section 2,
5. Find algorithm - using found description. It will work to fill lines succeedingly and threat all node the same way,

It's good to make one more step:

6. Algorithm evaluation - use it to reconstruct the real statistical description and calculate entropy.

We will go through this steps for HS:

1. Fix (1,0) as essential direction on $\mathbb{Z}^2$. Fix width - n.

$$Y = \mathbb{Z}(1,0) + (1,1)\overline{n},$$

where $\overline{n} := \{0, ..., n-1\}$.

We have to chose some boundary conditions. We can do it eg in 2 ways:

(a) *cyclic*: $\forall_i v(i+n, n) := v(i, 0)$,

(b) *zero*: $\forall_i v(i, -1) = v(i, n) := 0$.

We can think about cyclic conditions as additional constrain, so we get smaller entropy then original.
Zero conditions - space is split into straps - we have all constrains instead those between straps - lines ni and $ni + 1$ ($i \in \mathbb{Z}$) - we reduce number of constrains - increase entropy.
So choosing proper boundary conditions we can bound entropy from below or above.

In this paper we are not interested in counting entropy, but probability distribution - we want that alphabet to be as small as possible. So the best would be cyclic conditions - thanks of symmetry, we will be able to identify many states.

2. We are interested in valuations of $(1,1)\overline{n}$ - we don't have constrain inside - new alphabet has 2^n symbols: $\mathcal{A}' := (1,1)\overline{n} \to \{0,1\}$
but we can identify $v, w \in \mathcal{A}'$, that:

(a) cyclic translation: $\exists_k \forall_i\ v(i) = w(i + k \bmod n)$

(b) symmetry: $\forall_i\ v(i) = w(n-1-i)$

(c) unimportant zero: $v(0) = v(1) = v(2) = 1 \wedge w(1) = 0 \wedge \forall_{i \neq 1} v_i = w_i$

We have first two conditions from cycle symmetry.
Third says that two 1 blocks neighbor of node between them - its value is unimportant.
This condition is the advantage of taking diagonal - the number of symbols behave like 1.5^n and for vertical straps: $\varphi^n \cong 1.618^n$.

Using this identifications for example for $n = 21$ we get 3442 symbols instead of 2^{21} - 609 times less.

3. $M_{vw} = 1 \Leftrightarrow u \in V$
where $D(u) := \{(0,0),(1,0)\} + (1,1)\overline{n},\ u(i,i) := v(i),\ u(i+1,i) := w(i)$.

4. For M find eigenvectors (eg power method) and using methods from second section - we get probability distribution of two succeeding straps.

5. **Algorithm III: straps** of precision (a,b)
Order: $(i,j) < (k,l) \Leftrightarrow (i-j < k-l) \vee (i-j = k-l \wedge i < k)$
$q : \{0,1\}^a \times \{0,1\}^b \to [0,1]$
$q(u,v) =$ if there are no neighboring 1 and
u - valuation of previous a nodes,
v - valuation of b nodes which will influence to following nodes,
then with this (q) probability valuate this node (?) to 1.

We get q using two-straps probability distribution - look fig. 5.

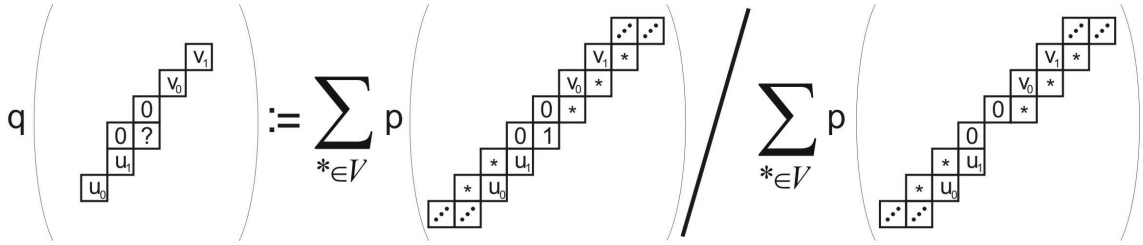

Figure 6: Finding q for $a = b = 2$.

Here are some algorithms as an example, found this way ($n = 24$):

(a) $a = b = 0$ $\quad q = 0.3602994$

(b) $a = b = 1$ $\quad q = \begin{pmatrix} 0.3365899 & 0.2946553 \\ 0.4406678 & 0.3999231 \end{pmatrix}$

(c) $a = b = 2$ $\quad q = \begin{pmatrix} 0.3350090 & 0.2918920 & 0.3265314 & 0.2910261 \\ 0.3507870 & 0.3075943 & 0.3423356 & 0.3067152 \\ 0.4419816 & 0.4001011 & 0.4342912 & 0.3992940 \\ 0.4421921 & 0.4004149 & 0.4345211 & 0.3996084 \end{pmatrix}$

Where the number of line denotes u (00?,10?,01?,11?), column - v analogically.

6. Algorithm evaluation.
We need to find statistical description that is really archived using given algorithm. It should be "similar" to used to find the algorithm(we will use it as the

starting point), but while constructing algorithm, we've lost some information. The description we are looking for, should be fixed point of iteration below (fig.7):

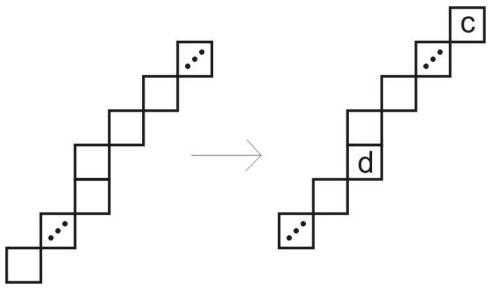

Figure 7: Iteration transforming algorithm into description

Where probability distribution for d is given by statistical algorithm.
There is a problem with c - we cannot find it straightforward.
We can although find its distribution, assuming that we've already found good approximation of probability distribution of strap valuations.

Algorithm:

(a) as starting description take used to find algorithm

(b) while in following iteration we get description more distant than some fixed boundary:

   i. using actual description find probability distribution for $(1,1)\overline{k}$

   ii. using algorithm and found distribution make some iterations from the fig.7

While having statistical description we can count capacity we get this way. Then we can compare it to the result from [6] - 43 digits of model's entropy.

Here are results: $-\log_{10}(H - H_q)$ for different a, b:

$a \setminus b$	0	1	2	3	4	5
0	2.06	2.113	2.1153	2.1153	2.1153	2.1153
1	3.32	3.82	3.99	4.001	4.002	4.003
2	3.50	4.37	5.19	5.44	5.466	5.436
3	3.50	4.42	5.55	6.43	6.74	6.78
4	3.50	4.42	5.58	6.71	7.60	7.96
5	3.50	4.42	5.58	6.74	7.83	8.72

Initialization

To use found algorithm in practice, we still need to initiate it. We could valuate the first strap anyhow and in a few straps we would tend to assumed statistical description - we loose only some information on the boundary.

But assume we want use whole space optimally.
On the first look - to valuate the first strap we should use probability distribution for one strap. But on one side of this strap there will be not constrains - we should be able to store here a bit more of information.

So for the first strap, we should use the statistical description for straps following strap filled with zeros (or suitable boundary conditions): $p(v) = S_{0v}$.
Second: Straps (first(v),second(u)) should have probability distribution: $p(v,u) = S_{0v}S_{vu}$ - we can find statistical algorithm for second line from this distribution.
And so on. Of course we are tending to original algorithm this way.

8 Conclusions

- We are now able to encode data sequences with given statistics. We could do it using Huffman's coding, but using it we usually couldn't get exact statistic and has difficulties with changing statistics.
- For one dimensional codings, we can analytically find optimal statistical description - we can encode with full capacity.
- We have criteria to ensure that given model has average informational capacity.
- For models in which long range correlations vanishes, we can speak about optimal statistical description, which gives us uniform distribution over elements.
- If we don't need exact optimality, we can use some simple algorithms, like filling over independent sets or over random sequence.
- We can generate approximations of uniform distribution on finite set.
- We can find algorithm as close to optimal as needed - there are results for Hard Square model in this paper.

References

[1] N. Calkin, H.Wilf, *The number of independent sets in a grid graph*, SIAM J. Discrete Math 11 (1998) 1,54-60

[2] Zs. Nagy, K. Zeger, *Bit Stuffing Algorithms and Analysis for Run Length Constraned Channels in Two and Three Dimensions*, IEEE Transactions of Information Theory 50(2004) 1, 3146-3169

[3] A. Kato, K. Zeger, *On the capacity of two-dimensional run-length constrained channels*, IEEE Transactions of Information Theory 45(1999) 5,1527-1540

[4] A. Burstein, *Independent sets in certain classes of (almost) regular graphs*, arXiv:math.CO/0310379

[5] R. J. Baxter, *Exactly Solved Models in Statistical Mechanics*, Academic, London (1982)

[6] R. J. Baxter, *Planar lattice gases with nearest-neighbour exclusion*, arXiv:cond-mat/9811264

[7] R. J. Baxter, *Variational Approximations for Square Lattice Models in Statistical Mechanics*, J. Stat. Phys. 19 (1978) 461 – 478

[8] W. Guo, H. W. J. Blöte, *Finite-size analyzis of the hard-square lattice gas*, Physical Review E 66, 046140 (2002)