The classical capacity of a quantum channel switching between two depolarizing channels

Ismail Akhalwaya, Nilanjana Datta and Tony Dorlas
Statistical Laboratory
Centre for Mathematical Sciences
Centre for Mathematical Sciences
University of Cambridge
Wilberforce Road, Cambridge CB3 0WB, U.K.

August 7, 2008

Abstract

We explore the classical product state capacity of quantum memory channels where the memory consists of a Markov chain switching between two single qubit channels, each of which is a depolarizing channel.

Contents

Chapter 1

Introduction

Quantum Mechanics introduces new resources and produces strange new effects in the field of information and communication. One subject of interest is the use of a quantum channel to transmit classical information. A lot is known about simple versions of this setup, for example the channel capacity for only product state inputs and memoryless channels. But there are still many more exciting and important extensions.

1.1 Motivation

It is the HSW theorem that gives the classical product state capacity of a memoryless noisy quantum channel. But real world communication channels are not memoryless. So one simple extension to the HSW-setup that may be considered is noise with memory.

One of the simplest examples of noise with memory is Markovian noise, where the probability of an error depends only on the most recent state of knowledge and is independent of any older history.

1.2 Model

1.2.1 Single Qubit Depolarizing Channel

To keep things simple, our noisy quantum channel with memory is made up of two single qubit depolarizing channels (Φ_1, Φ_2). These two channels represent two possible depolarizing errors which includes the possibility that one of them may be the no-error option (identity qubit channel).

The exact form of our two single qubit depolarizing channels is:

$$\Phi_i = \rho \mapsto (1 - p_i)\rho + \frac{p_i}{2}\mathbf{1}$$

In this form, the noise on this channel can be seen as either nothing or absolute mixing with the relevant probabilities.

Denoting $x_i = 1 - p_i/2$, we have

$$\Phi_i(\rho) = x_i\rho + (1 - x_i)(\mathbf{1} - \rho) \tag{1.1}$$

Thus, the noise on the channel can be seen as either doing nothing or 'flipping' (orthogonal complement) with the following different probabilities, x_i and $1 - x_i$. Note that $x_i \geq \frac{1}{2}$.

1.2.2 Markov Chain

Now N uses of the overall channel involves randomly switching between Φ_1 and Φ_2, N times. The chain of switches forms a Markov Chain.

A finite Markov Chain can be characterized by a transition matrix and a starting distribution γ. Each column of the transition matrix represents the probability profile of switching from the state numbered by that column to all other states. Thus each column must sum to one (any matrix with this property is called stochastic).

For simplicity we use a symmetric transition matrix, i.e. for example

$$Q = \frac{1}{3}\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

is used as the transition matrix when doing numerical simulations.

When calculating powers of the transition matrix its symmetric nature is very useful. We can diagonalize Q and raise just the eigenvalues on the diagonal.

$$\begin{aligned} Q &= \begin{pmatrix} 1-q & q \\ q & 1-q \end{pmatrix} \\ &= \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & 1-2q \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \end{aligned} \tag{1.2}$$

(The parameter q must of course satisfy $0 \leq q \leq 1$.) Now raising Q to the power n:

$$\begin{aligned} Q^n &= \begin{pmatrix} 1-q & q \\ q & 1-q \end{pmatrix}^n \\ &= \frac{1}{2^n}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & (1-2q)^n \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \end{aligned} \tag{1.3}$$

For $q = 1/3$, $1-2q = 1/3$. The unitary matrix $U = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, used in the diagonalization procedure features again later in helping simplify one of the steps. We also refer to the diagonal elements as $d_0 = 1$ and $d_1 = 1 - 2q$.

A Markov Chain has equilibrium probability distributions. For our two-state symmetric Markov chain the equilibrium distribution is always $\begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}$ for any q. It is of course an eigenvector of Q.

1.2.3 Definition of the channel

Now that we have introduced the single channels, (Φ_1,Φ_2), and the Markov chain, (Q,γ), we can formally build up our channel.

We choose to set the starting distribution to the equilibrium distribution $\gamma = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}$, to reflect the idea that we want to treat the channel selection symmetrically.

We then ask, given this setup, what is the probability of a certain sequence of switches $\underline{i} = (i_1, i_2, \ldots, i_n) \in \{1, 2\}^n$ occurring. We use this probability as the co-efficient of the tensor product of the corresponding sequence of single qubit channels. Finally we add up all such terms to form our final channel:

$$\Lambda_n : \rho_1 \otimes \ldots \otimes \rho_n \mapsto \sum_{i_1,\ldots,i_n} \gamma_{i_1} q_{i_1 i_2} \ldots q_{i_{n-1} i_n} \Phi_{i_1}(\rho_1) \otimes \ldots \otimes \Phi_{i_n}(\rho_n)$$

Chapter 2

Capacity

We are ultimately interested in calculating the product state classical capacity of our Markovian Switching Depolarizing Channel:

$$C^1_{\text{Classical}}(\Lambda_n) = \chi^*(\Lambda_n),$$

where χ^* is the maximization of χ over all possible input distributions and ensembles:

$$\chi(\{(p_i, \Lambda_n(\rho_i^{(n)}))\}) = \mathrm{S}(\sum_i p_i \Lambda_n(\rho_i^{(n)})) - \sum_i p_i \, \mathrm{S}(\Lambda_n(\rho_i^{(n)})).$$

S is the usual von Neuman entropy. This maximization need only be done over pure state inputs. *For now, we consider only the ensemble of pure state computational basis inputs with the uniform distribution.* Then the first term is just the entropy of the maximally mixed state, which is n (see **??**. Ensemble Output Average on page **??**). The second term is the focus of the numerical simulations.

2.1 Output Entropy

Now we look into the second term in more detail. The output entropy of a computational basis state turns out to be independent of the basis state itself, so take any one and call it $\rho^{(n)}$:

$$\rho^{(n)} = |l_1 \rangle\!\langle l_1| \otimes \ldots \otimes |l_n \rangle\!\langle l_n|,$$

where $\underline{l} = (l_1, l_2, \ldots, l_n) \in \{0,1\}^n$. (The computational basis is denoted $\{|0\rangle, |1\rangle\}$.)

Acting with Λ_n on this $\rho^{(n)}$ and using the form of the depolarizing channels (**??**) where the output is either unchanged or flipped, we get

$$\begin{aligned}\Lambda_n(\rho^{(n)}) = \sum_{i_1,\ldots,i_n} & \gamma_{i_1} q_{i_1 i_2} \ldots q_{i_{n-1} i_n} \\ & \times (x_{i_1}^{(1)} |l_1 \oplus 0 \rangle\!\langle l_1 \oplus 0| + x_{i_1}^{(2)} |l_1 \oplus 1 \rangle\!\langle l_1 \oplus 1|) \otimes \ldots \\ & \otimes (x_{i_n}^{(1)} |l_n \oplus 0 \rangle\!\langle l_n \oplus 0| + x_{i_n}^{(2)} |l_n \oplus 1 \rangle\!\langle l_n \oplus 1|) \quad (2.1)\end{aligned}$$

Here $\oplus$ denotes addition modulo 2, that is, $l_i \oplus 0 = l_i$ represents the unchanged single qubit choice, while $l_i \oplus 1 = 1 - l_i$ is the original choice flipped. Besides the above clear explanation $\oplus$ can be used as addition modulo 2, to give the correct answers.

2.1.1 Eigenvalues and eigenvectors

Now we can expand out the tensor product, using an indices $\underline{k} = (k_1, k_2, \ldots, k_n) \in \{0,1\}^n$ to indicate which term in the bracket is selected:

$$\Lambda_n(\rho^{(n)}) = \sum_{i_1,\ldots,i_n;k_1,\ldots,k_n} \gamma_{i_1} q_{i_1 i_2} \ldots q_{i_{n-1} i_n} x_{i_1}^{(k_1)} \ldots x_{i_n}^{(k_n)} \bigotimes_{j=1}^{n} |l_j \oplus k_j \rangle\!\langle l_j \oplus k_j| \quad (2.2)$$

where we write $x_i^{(0)} = x_i$ and $x_i^{(1)} = \overline{x_i} = 1 - x_i$.

This step is significant because if we swap the order of the sums, that is we look at a given $\underline{k}$ and sum over $\underline{i}$, we have written $\Lambda_n(\rho^{(n)})$ as

$$\Lambda_n(\rho^{(n)}) = \sum_{k_1,\ldots,k_n} \lambda_n(\underline{k}) \bigotimes_{j=1}^{n} |l_j \oplus k_j \succ\!\prec l_j \oplus k_j|,$$

which is a sum of orthogonal projections with coefficients

$$\lambda_n(\underline{k}) = \sum_{i_1,\ldots,i_n} \gamma_{i_1} q_{i_1 i_2} \ldots q_{i_{n-1} i_n} x_{i_1}^{(k_1)} \ldots x_{i_n}^{(k_n)} \tag{2.3}$$

which are therefore its eigenvalues.

Of course, since $\Lambda_n(\rho^{(n)})$ is a density matrix, if we sum over all $\underline{k}$ (error sequences: see later) we get unity,

$$\sum_{\underline{k}} \lambda_n(\underline{k}) = 1. \tag{2.4}$$

This is easily checked using (**??**) and the fact that we have a Markov chain.

Note that the eigenvalue spectrum is the same regardless of the input computational basis (specified by $\underline{l}$). But, of course, the output eigenvector corresponding to $\underline{k}$ does depend on $\underline{l}$. The eigenvector, as we see above, is the original $\underline{l}$ tensor product modified by $\underline{k}$. To define this neatly let us introduce a function which takes a sequence and produces the corresponding computational basis density operator:

$$\Psi(\underline{s}) = \bigotimes_{j=1}^{n} |s_j \succ\!\prec s_j| \tag{2.5}$$

Here, n, is the length of the sequence $\underline{s}$.

Then in this notation, the output eigenvector, call it $e_{\underline{l}}(\underline{k})$, can be written:

$$e_{\underline{l}}(\underline{k}) = \Psi(\underline{l} \oplus \underline{k}) \tag{2.6}$$

Note, the $\oplus$ represents pair-wise addition modulo 2.

To give meaning to the $\underline{k}$ sequence, note that it represents the choices of whether or not there was a 'flip', regardless of which channel was selected (which is why we sum over $\underline{i}$ in the λ_n expression). Thus the number of 1's in $\underline{k}$ is the Hamming distance between the input computational state and the particular output eigenvector ($e_{\underline{l}}(\underline{k})$).

In the above introduced notation, the action of the channel can be represented as:

$$\Lambda_n(\Psi(\underline{l})) = \sum_{\underline{k}} \lambda_n(\underline{k}) e_{\underline{l}}(\underline{k}) \tag{2.7}$$

2.1.2 Average output ensemble

From (**??**) it is clear that for each $\underline{k}$ sequence we get a unique eigenvector (a computational basis state). Combined with (**??**), we see that for each $\underline{l}$ input state we get the same eigenvalue spectrum but attached to different eigenvectors, a mere reshuffling of all computational basis states. That is, each basis state will get associated once and only once in turn with each eigenvalue in the spectrum as we vary $\underline{l}$ or $\underline{k}$. This observation gives us the earlier mentioned result that if we look at the entropy of the average output ensemble we get the maximally mixed state. That is, each computational basis state (specified by $\underline{l'}$), will get a single $\lambda_n(\underline{k})$ contribution when the input state, $\underline{l}$ and the $\underline{k}$ modification satisfies $\underline{l'} = \underline{l} \oplus \underline{k}$. That is we work backwards by looking for an input state that is the right error vector away from the output. Using (**??**), but now across the uniform input ensemble, we get that the average output state is the maximally mixed state.

Putting all this together we have the following proof that

$$\mathrm{S}(\sum_i p_i \Lambda_n(\rho_i^{(n)})) = n,$$

given that $p_i = \frac{1}{2^n}$ and $\rho_i^{(n)} = \Psi(\underline{\mathrm{Binary}(i)})$:

$$
\begin{aligned}
\mathrm{S}\left(\sum_i p_i \Lambda_n(\rho_i^{(n)})\right) &= \mathrm{S}\left(\sum_{\underline{l}} \frac{1}{2^n}\Lambda_n(\Psi(\underline{l}))\right) \\
&= \mathrm{S}\left(\sum_{\underline{l}} \frac{1}{2^n}\sum_{\underline{k}} \lambda_n(\underline{k}) e_{\underline{l}}(\underline{k})\right) \\
&= \mathrm{S}\left(\frac{1}{2^n}\sum_{\underline{l}}\sum_{\underline{k}} \lambda_n(\underline{k})\Psi(\underline{l}\oplus\underline{k})\right) \\
&= \mathrm{S}\left(\frac{1}{2^n}\sum_{\underline{k}}\sum_{\underline{l}} \lambda_n(\underline{k})\Psi(\underline{l}\oplus\underline{k})\right) \\
&= \mathrm{S}\left(\frac{1}{2^n}\sum_{\underline{k}}\left[\lambda_n(\underline{k})\sum_{\underline{l}}\Psi(\underline{l}\oplus\underline{k})\right]\right) \\
&= \mathrm{S}\left(\frac{1}{2^n}\left[\sum_{\underline{k}}\lambda_n(\underline{k})\right]\left[\sum_{\underline{l'}}\Psi(\underline{l'})\right]\right) \\
&= \mathrm{S}\left(\frac{1}{2^n}\sum_{\underline{l'}}\Psi(\underline{l'})\right) \\
&= n
\end{aligned}
\tag{2.8}
$$

The last line is because the entropy of the maximally mixed state is $\log_2$ of the dimension, giving $\log_2(2^n) = n$.

2.1.3 Diagonalization

The next step is to better understand the eigenvalues and its sum over $\underline{i}$.

If we only look at the product of the transition probabilities and the initial probability distribution we see that we have a simple Markov chain. The diagonalization tools mentioned in the Markov Chain section (see **??**. Markov Chain on page **??**) would be useful and in fact if we look at the sum we would get unity. However, for each path through possible transitions we still have

to take into account the contribution from the $x_{i_m}^{(k_m)}$ factors. These factors represent the probability of a k_m modification being done to the m^{th} qubit by the i_m^{th} single qubit channel. We have already seen that taken all together (including summing over $\underline{k}$) we do still get unity (see **??**. Sum Eigenvalues on page **??**).

We need to rewrite this sum over $\underline{i}$, to better understand $\lambda_n(\underline{k})$'s dependence on $\underline{k}$. We use the Markov diagonalization tools, but now to try and reduce the $x_{i_m}^{(k_m)}$ factors dependence on a specific path. After all, many paths may have the same contribution and we are summing over all paths.

To this end recall that $q_{rs} = \sum_j u_{rj} d_j u_{js} \quad r, s, j \in \{0, 1\}$. Substitute this form into (**??**):

$$\lambda_n(\underline{k}) = \sum_{i_1,\ldots,i_n;j_1,\ldots,j_{n-1}} \gamma_{i_1} u_{i_1 j_1} d_{j_1} u_{j_1 i_2} \ldots u_{i_{n-1} j_{n-1}} d_{j_{n-1}} u_{j_{n-1} i_n} x_{i_1}^{(k_1)} \ldots x_{i_n}^{(k_n)} \tag{2.9}$$

Now we group neighbouring $u_{j_m i_m}$ and $u_{i_m j_{m+1}}$ and also $x_{i_m}^{(k_m)}$ and note that these factors contain the only i_m dependence. So we evaluate this sum for $i_m \in \{0, 1\}$ and get:

$$\sum_{i_m} u_{j_m i_m} u_{i_m j_{m+1}} x_{i_m}^{(k_m)} = \frac{1}{2}(x_1^{(k_m)} + (-1)^{j_m + j_{m+1}} x_2^{(k_m)})$$

(Note that, for a simple Markov Chain, there would be no $x_{i_m}^{(k_m)}$ factor and the remainder of the expression would evaluate to 1 if $j_m = j_{m+1}$ and to 0 otherwise. Showing that in the basis of the eigenvectors of the transition matrix, repeated application of the transition matrix is simply the repeated multiplication of the relevant eigenvalue.)

The outer two u factors can be put into a similar form as the inner $n-2$ pairs of factors:

$$\sum_{i_1} u_{i_1 j_1} x_{i_1}^{(k_1)} = \frac{1}{\sqrt{2}}(x_1^{(k_1)} + (-1)^{j_1} x_2^{(k_1)})$$

and

$$\sum_{i_n} u_{j_{n-1}i_n} x_{i_n}^{(k_n)} = \frac{1}{\sqrt{2}}(x_1^{(k_n)} + (-1)^{j_{n-1}} x_2^{(k_n)})$$

Thus putting it all together and using the fact that $\gamma_i = \frac{1}{2}$ is the stationary state,

$$\lambda_n(\underline{k}) = \sum_{j_1,\ldots,j_{n-1}} \frac{1}{2^n} d_{j_1} \ldots d_{j_{n-1}} (x_1^{(k_1)} + (-1)^{j_1} x_2^{(k_1)}) \\ \times \ldots (x_1^{(k_m)} + (-1)^{j_m + j_{m+1}} x_2^{(k_m)}) \ldots (x_1^{(k_n)} + (-1)^{j_{n-1}} x_2^{(k_n)}) \quad (2.10)$$

We now observe that there is a single term in the $\underline{j}$ sum which makes the largest contribution by far; this is the $j_m = 0 \quad \forall \quad m$. This is because for that assignment all the $d_{j_m} = d_0$'s are equal to 1. For all other $\underline{j}$ terms there is at least one factor d_1 so that the contribution is multiplied by 1/3. In addition, some of the exponents $j_m + j_{m+1}$ are odd which also makes the corresponding term relatively smaller.

To properly see this latter point, let us look at four possibilities for the pair $(k_m, (-1)^{j_m + j_{m+1}})$ and its effect on the general term. Firstly, $(0, 1)$:

$$(x_1^{(1)} + x_2^{(1)}) = 1 - p_1/2 + 1 - p_2/2 = 2 - (p_1 + p_2)/2$$

Secondly, $(0, -1)$:

$$(x_1^{(1)} - x_2^{(1)}) = 1 - p_1/2 - 1 + p_2/2 = (p_2 - p_1)/2$$

Thirdly, $(1, 1)$:

$$(x_1^{(2)} + x_2^{(2)}) = p_1/2 + p_2/2 = (p_1 + p_2)/2$$

Fourthly, $(1, -1)$:

$$(x_1^{(2)} - x_2^{(2)}) = p_1/2 - p_2/2 = (p_1 - p_2)/2$$

Here it can be seen that the largest term is the first option and the second largest is the third option. Both occur when $j_m + j_{m+1}$ is even, corroborating the earlier point.

Now the largest term occurs when $k_m = 0$, that is, there is no error on the m^{th} qubit. So $\lambda_n(\underline{k})$'s primary dependence is on the Hamming weight of the $\underline{k}$ sequence. That is for low Hamming weight, corresponding to few errors, $\lambda_n(\underline{k})$ is relatively high.

This last point coupled with the first observation lend themselves to a rough analytic expression for the λ_n in terms of the number of 1's (call it h) in the $\underline{k}$ sequence:

$$\begin{aligned} \lambda_n(h) &= \frac{1}{2^n}\left[2-(p_1+p_2)/2\right]^{n-h}\left[(p_1+p_2)/2\right]^h \\ &= \frac{1}{2^n}\left[\frac{4-p_1-p_2}{2}\right]^n\left[\frac{p_1+p_2}{4-p_1-p_2}\right]^h \end{aligned} \tag{2.11}$$

2.2 Transfer matrices

The expression we want to compute is:

$$\bar{S} = -\frac{1}{n}\mathbb{E}\left[\lambda_n \log \lambda_n\right],$$

where I now write n instead of n, and where

$$\begin{aligned} \lambda_n \;=\; & \frac{1}{2^n}\sum_{j_1,\ldots,j_{n-1}=0,1} d_{j_1}\ldots d_{j_{n-1}}(x_1(k_1)+(-1)^{j_1}x_2(k_1)) \\ & \times(x_1(k_2)+(-1)^{j_1+j_2}x_2(k_2))\ldots(x_1(k_{n-1})+(-1)^{j_{n-2}+j_{n-1}}x_2(k_{n-1})) \\ & \times(x_1(k_n)+(-1)^{j_{n-1}}x_2(k_n)). \end{aligned}$$

I now label the states $(k_1,\ldots,k_2)$ by the first element $k_1 = 0,1$ and the lengths of the subsequent rows of 0's and 1's. Thus, if $k_1 = 0$, n_1 is the length of the first set of 0's, i.e. $k_1,\ldots,k_{n_1} = 0$ and $k_{n_1+1} = 1$, etc. Assuming that $k_1 = 0$ for the moment, we have

$$\lambda_n = \frac{1}{2^n}\sum_{j'_1,\ldots,j'_{m-1}=0,1}(T^{n_1})_{0,j'_1}\left(\overline{T}^{n_2}\right)_{j'_1,j'_2}(T^{n_3})_{j'_2,j'_3}\ldots, \tag{2.12}$$

where

$$(T^n)_{j,l} = \sum_{j_1,\ldots,j_{n-1}} d_j^{1/2} d_{j_1} \ldots d_{j_{n-1}} d_l^{1/2} (x_1 + (-1)^{j+j_1} x_2) \times (x_1 + (-1)^{j_1+j_2} x_2) \ldots (x_1 + (-1)^{j_{n-1}+l} x_2),$$

and similarly for $\overline{T}$. This is indeed the n-th power of a transfer matrix:

$$T_{j,l} = d_j^{1/2} (x_1 + (-1)^{j+l} x_2) d_l^{1/2}$$

or

$$T = \begin{pmatrix} x_1 + x_2 & \frac{1}{\sqrt{3}}(x_1 - x_2) \\ \frac{1}{\sqrt{3}}(x_1 - x_2) & \frac{1}{3}(x_1 + x_2) \end{pmatrix}.$$

The corresponding eigenvalues are

$$\lambda_\pm = \frac{2}{3}(x_1 + x_2) \pm \frac{1}{3}\sqrt{\Delta},$$

where

$$\Delta = (x_1 + x_2)^2 + 3(x_1 - x_2)^2.$$

It is now easily shown that we can write

$$T^n = \frac{1}{2\sqrt{\Delta}} \begin{pmatrix} \sqrt{\Delta}(\lambda_+^n + \lambda_-^n) + (x_1 + x_2)(\lambda_+^n - \lambda_-^n) & \sqrt{3}(x_1 - x_2)(\lambda_+^n - \lambda_-^n) \\ \sqrt{3}(x_1 - x_2)(\lambda_+^n - \lambda_-^n) & \sqrt{\Delta}(\lambda_+^n + \lambda_-^n) - (x_1 + x_2)(\lambda_+^n - \lambda_-^n) \end{pmatrix}. \quad (2.13)$$

So far so good. But now comes a BIG leap of faith! The distribution of the lengths of 0's and 1's in $\underline{k}$ is exponential, i.e. $n_r = n$ with probability 2^{-n}. It therefore seems that the factor $(T^n)_{0,0}$ should occur approximately $2^{-n}\frac{n}{8}$ times, etc. This would yield the answer:

$$-\frac{1}{n}\mathbb{E}\left[\lambda_n \log \lambda_n\right] \approx -\frac{1}{8}\sum_{n=1}^{\infty} 2^{-n} \left[(T^n)_{0,0} \log(T^n)_{0,0} + 2(T^n)_{0,1} \log(T^n)_{0,1} + (T^n)_{1,1} \log(T^n)_{1,1} + (\overline{T}^n)_{0,0} \log(\overline{T}^n)_{0,0} + 2(\overline{T}^n)_{0,1} \log(\overline{T}^n)_{0,1} + (\overline{T}^n)_{1,1} \log(\overline{T}^n)_{1,1}\right] \quad (2.14)$$

However, this is incorrect because there is correlation between the subsequent matrix elements. We need to count. Divide the j_r into groups of length $m_1, m_2, \ldots, m_p$. Then $T_{0,0}$ and $(\overline{T})_{0,0}$ occur approximately $\frac{1}{2}(m_1 - 1 + m_3 - 1 + \ldots$ times, $T_{1,1}$ and $(\overline{T})_{1,1}$ occur about $\frac{1}{2}(m_2 - 1 + m_4 - 1 + \ldots)$ times, and $T_{0,1}$, $T_{1,0}$, $(\overline{T})_{1,0}$ and $(\overline{T})_{0,1}$ occur approximately $p/4$ times each. Thus,

$$\begin{aligned}\lambda_n &\approx \sum_{k=1}^{m} \sum_{p=1}^{k\wedge(m-k)} \binom{k}{p} \binom{m-k}{p} (T_{0,0}(\overline{T})_{0,0})^{\frac{k-p}{2}} (T_{1,1}(\overline{T})_{1,1})^{\frac{m-k-p}{2}} \\ &\quad \times (T_{0,1}T_{1,0}(\overline{T})_{0,1}(\overline{T})_{1,0})^{\frac{p}{4}}. \end{aligned} \tag{2.15}$$

Here each T and $\overline{T}$ stands for one of the operators T^n and $\overline{T}^n$. Note that it does not matter in which position so we can simply count their average number. This is determined by the distribution of $k_1, \ldots, k_N$. The expression (4) is analogous to the Ising model and can be evaluated by the transfer matrix method again: $\lambda_n \approx \Lambda_{\max}^m$, where

$$\begin{aligned}\Lambda_{\max} &= \frac{1}{2}\Big[(T_{0,0}(\overline{T})_{0,0})^{1/2} + (T_{1,1}(\overline{T})_{1,1})^{1/2} \\ &\quad + \sqrt{\big((T_{0,0}(\overline{T})_{0,0})^{1/2} - (T_{1,1}(\overline{T})_{1,1})^{1/2}\big)^2 + 4(T_{0,1}T_{1,0}(\overline{T})_{0,1}(\overline{T})_{1,0})^{1/2}}\Big] \end{aligned} \tag{2.16}$$

We now have to replace each T and $\overline{T}$ by the appropriate occurrence of T^n resp. $\overline{T}^n$. Assuming that a sequence of n zeros in $k_1, \ldots, k_N$ occurs with frequency p_n and a sequence of n ones with frequency q_n we have

$$\sum_{n=1}^{\infty} n(p_n + q_n) = 1 \tag{2.17}$$

and we must replace $T_{0,0}$ by $\prod_{n=1}^{\infty}(T_{0,0}^n)^{p_n}$ and similarly $\overline{T}_{0,0}$ by $\prod_{n=1}^{\infty}(\overline{T}_{0,0}^n)^{q_n}$. Denoting the resulting expression by $\lambda_n(\{p_n, q_n\})$ the final answer is

$$\begin{aligned}&\frac{1}{n}\sum_{r_1=0}^{n}\sum_{s_1=0}^{n-r_1}\sum_{r_2=0}^{[(n-r_1-s_1)/2]}\sum_{s_2=0}^{[(n-r_1-s_1-2r_2)/2]} \ldots \lambda_n(\{\frac{r_1}{n}, \frac{r_2}{n}, \ldots, \frac{s_1}{n}, \frac{s_2}{n}, \ldots\}) \\ &\qquad \times \log \lambda_n(\{\frac{r_1}{n}, \frac{r_2}{n}, \ldots, \frac{s_1}{n}, \frac{s_2}{n}, \ldots\}) \\ &\approx \sup_{\{p_n, q_n\}} \left(\sum_{n=1}^{\infty}(p_n + q_n)\right) \lambda_n(\{p_n, q_n\}) \log \lambda_n(\{p_n, q_n\}). \end{aligned} \tag{2.18}$$

Note that this result is easily computed because the sum converges very rapidly. It is also quite efficiently checked numerically, using the exact formula (2), because of the exponential distribution of the lengths. The best way is probably to compute the values of $(T^n)_{j,l}$ for low values of n (say up to 20), and tabulate them first. Higher values of n are extremely unlikely and can be ignored. Then simply do a sum over m and a multiple sum over the individual $n_r \geq 1$ with conditions such that their sum is n and each is no greater than 20 (perhaps it is better to first take a smaller number, say 10). The sum over $\underline{j}$ can perhaps also be simplified by first counting the number of equal n_r's. All the same, this is a non-trivial exercise.