Building Optical Constructions with Minimum Complexity

Hadas Kogan, Isaac Keslassy
Department of Electrical Engineering
Technion - Israel Institute of Technology
Haifa 32000, Israel
{rhadas@tx,isaac@ee}.technion.ac.il

***Abstract*—It is often claimed that future systems will necessarily be all-optical, because electronic devices are not fast enough to keep up with the increase in fiber capacity. However, two objections are commonly raised: first, optical systems need many basic optical components, which are typically very expensive; and second, optical systems need many switch reconfigurations, which are typically very slow.**

In this paper, we examine whether these two costs can be fundamentally bounded. First, we develop the equivalence between coding theory and optical system design by introducing the concept of super switches. Then, we show how the minimal expected number of switch reconfigurations is almost equal to the state space entropy of the optical system. Finally, we point out the trade-off between the two types of costs.

I. Introduction

Network systems such as routers, network processors or buffers are commonly implemented today using electronics. Consequently, their scaling abilities are limited by Moore's law and memory bandwidth growth, which have historically grown slower than optical link capacities [1]. As incoming fibers become faster, it becomes harder and harder for them to cope with the incoming traffic.

Because of this growth discrepancy, it is often claimed that future systems will necessarily be all-optical. However, two objections are commonly raised. First, optical systems need many basic optical components, which are typically very expensive. And second, optical systems need many switch reconfigurations, which are typically very slow, especially when compared to electronic switch reconfiguration times and packet arrival speeds. Is is unclear, though, whether these two types of costs are fundamentally present in all optical systems, and whether they can be lowered if we have some knowledge on the arrival traffic.n this paper, our goals will first be to model these two types of costs, then present fundamental lower bounds on these costs in general optical systems, and finally build optical constructions that reach these lower bounds.

The research in this area started when Shannon published his paper on the memory requirements of a telephone exchange [2]. Then, the discussion on the complexity of connecting networks was further extended in [3]–[5], which introduced links to information theory. Also related are works on time-slot-interchange complexity [3], minimum-complexity combined time-space switching [6]–[8] and minimum-complexity optical queues [9], [10]. Below, we will consider and explain many results of these papers under the angle of the two types of costs.

In this paper, we first consider the question of minimizing the expected number of switch reconfigurations. We find equivalences between optical constructions and coding theory. In fact, we find that the question of designing an optical system with as few switch reconfigurations as possible is very similar to the question of designing an optimal code. This is due to the fact that a 2×2 switch, which can be set either to "cross" or "bar" state, could be thought of as equivalent to a binary digit, which can be set either to zero or to one.For each state of the system, we define the switches that take part in the formation of that state as *active*, and the switches that are irrelevant to the formation of that state as *passive*. We further define the *theoretical complexity* C^* of the system as the minimum expected number of active switches, where the expectation is taken over the state space. This helps us provide close lower and upper bounds on the theoretical complexity of the system. In fact, if we denote the entropy of the state space by H, we prove that

$$H \leq C^* \leq H + 1.$$

Moreover, we present a general construction that achieves these bounds given some probability distribution. A construction with an expected number of switch

reconfigurations that is equal in growth to the theoretical complexity is said to be *theoretically optimal*.We then discuss the question of minimizing the number of 2×2 switches in the system. We show that if the number of possible states is K and the time it takes to perform one state is T, the practical complexity C is lower bounded as follows:

$$C \geq \lceil \frac{\log K}{T} \rceil.$$

A construction with a practical complexity that grows like this lower bound is said to be *practically optimal*. We show that there is a certain tradeoff between designing a system that is practically optimal, and a system that is theoretically optimal. This tradeoff appears usually when the states have a highly non-uniform distribution.

It is important to note that this paper mostly presents a fundamental approach to the complexity of optical systems, without putting a stress on practical implementations. By defining complexity and explaining how lower bounds on complexity can be obtained, it is laying the ground for more practical papers. In [11], we use these fundamental results to present constructions of optical buffers and optically-buffered routers that are practically optimal, i.e., have a number of basic components that grows like the practical complexity lower bound defined in this paper.

This paper is organized as follows. First, Section II presents definitions related to systems and complexity of systems. Then, Section III presents links between coding theory and the complexity of systems. The theoretical complexity of a construction is defined to be the theoretical minimum on the expected number of switch reconfigurations, where the expectation is calculated with respect to the states space. We find lower and upper bounds on the practical complexity of constructions, and define theoretically optimal constructions. Finally, Section IV, the practical complexity is defined to be the theoretical minimum on the number of 2×2 switches in a construction, and the tradeoff between the practical complexity and the theoretical complexity is presented.

II. Definitions

We will refer to a *system* as an ideal network element that has input links, output links and inner states. The outputs of the system are uniquely determined as a function of the inputs and the inner states of the system during the entire time of operation. Packet size is fixed, time is slotted and it takes one time-slot to transmit a packet. If packets have variable sizes, they are segmented into fixed size blocks during arrival and reassembled at departure. Let's first review the definition of *external states* and *internal states*, as defined in [3].

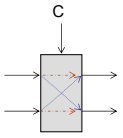

Fig. 1. A controlled 2×2 switch

Definition 1: A system has a set T of external states, where each external state is a distinguishable possible system output.

Definition 2: A system has a set S of internal states, where each internal state is a different setting of the system elements.

Consider the mapping $\sigma : S \rightarrow T$, linking each internal state to the resulting external state. By causality, to each external state corresponds some internal state, i.e., σ is surjective. Therefore, in order to reach T different outputs, at least an equal number of internal states is required: $|S| \geq |T|$.

The external states are not necessarily equiprobable. Using their probability distribution, we are now able to define the system entropy.

Definition 3: Assume that there exists a probability distribution P_T on the set of external states T, with $\sum_{t_i \in T} P_T(t_i) = 1$. The *system entropy* is given by the entropy of the external states:

$$H = -\sum_{t_i \in T} P_T(t_i) \log(P_T(t_i))$$

Example 1: Consider the case of an $N \times N$ switch, where all the permutations are equiprobable. There are $N!$ equiprobable external states. Therefore, the entropy of an $N \times N$ switch is given by:

$$H = -\sum_{i=1,..,N!} \frac{1}{N!} \log(\frac{1}{N!}) = \log(N!)$$

Definition 4: An optical construction *emulates* an ideal system if, under identical arrivals, identical packets depart from the optical construction within some bounded delay compared to the ideal system [12]. If the delay is fixed, we say that the construction *strongly emulates* the system. If there is no delay, we say that the construction *exactly emulates* the system.

he basic element in our optical constructions is an *optical* 2×2 *switch*.

Definition 5: An *optical* 2×2 *switch* is a network element with 2 inputs, 2 outputs and a control input c, see Figure 1. If $c = 0$, the switch is in a "bar" state, and the inputs are passed forward to the output links. If $c = 1$, the switch is in a "cross" state, and the outputs are the inputs with interchanged positions.

Definition 6: A system is *with memory*, when the packets on the output links may have arrived during previous time-slots. A system is *memoryless*, when the packets on the output links arrived on the current time-slot.

Definition 7: A *Frames Switch* [6], [13] is a network element that has N input links and a frame size of B packets, where the output frame is a permutation both in space and time of the corresponding input frame.

Definition 8: A *Time Slot Interchange* (TSI) [3], [7] is a frames switch with one input link and one output link.

Definition 9: The *state duration* T_S of a system is the time it takes to form a single external state.

Example 2: The state duration of an unbuffered $N \times N$ switch is $T_S = 1$.

Example 3: The state duration of a TSI with buffer size B is $T_S = B$.

III. Theoretical complexity and relations to coding theory

A. Complexity of optical systems

There is an intuitive connection between the construction of optical systems and coding theory [2], [3]. In fact, a 2×2 switch could be thought of as equivalent to a binary digit. In the same way as the binary digit can be set to zero or to one, the 2×2 switch can be set to a "cross" or a "bar" state. For instance, Shannon has famously argued that the number of 2×2 switches needed to construct an $N \times N$ switch able to realize all possible $N!$ permutations is at least $C = \log(N!)$ [2], which is exactly the number of digits required to code $N!$ symbols with uniform distribution.

Our goal is to build constructions with minimal complexity. We will relate to two notions of complexity of optical constructions. The first notion is the expected number of connection reallocations necessary to set the system states. The second notion is the number of switches in the construction. We would like to find the connection between the probability distribution of the external states, and these two notions of complexity. It seems that if the external states are not equiprobable, we could use the prior knowledge on the probability of the states so as to design a switch that is more economic in some ways. For example, consider an output-queued switch with N links and queue size B. As shown in [11], at least $\Theta(N \log(NB))$ 2×2 switches are required to emulate it if all packet arrival patterns are allowed. But what if we know in advance that the packet arrival pattern is not uniform? For instance, if the packets on input link i are destined either to output link i or to output link $(i+1) \bmod N$? Intuitively, such a prior knowledge might help reduce the number of switch reallocations and the number of 2×2 switches.

B. Definition of theoretical complexity

In this section we will define a metric for the number of switch reconfigurations. We will denote it as the theoretical complexity, and present connections to coding theory.

We would first like to examine whether there is any connection between the entropy of the external states and the complexity of a network element, as would seem natural. The following example demonstrates that we should not measure the complexity of a network element only using its number of 2×2 switches, but also using some other metrics.

Example 4: Consider a system with N inputs: $(I_1, I_2, ..., I_N)$ and N outputs: $(O_1, O_2, ..., O_N)$, where N is even and $N \geq 4$. The system chooses between two equiprobable states: either the outputs are equal to the inputs, or the system interchanges the positions of every two consecutive inputs:

$$(O_1, O_2, ..., O_N) \in \quad \{(I_1, I_2, I_3, I_4 ..., I_{N-1}, I_N) \\ (I_2, I_1, I_4, I_3, ..., I_N, I_{N-1})\}$$

In other words, the probability distribution over the set of all possible $N!$ permutations is given by:

$$P_T = \{\frac{1}{2}, 0, ..., 0, \frac{1}{2}, 0, ..., 0\}$$

The entropy of a system with two equiprobable states is $H = 1$. However, it is not possible to construct such a system with less than $\frac{N}{2}$ 2×2 switches, because there are N changeable inputs. Therefore, this simple example shows that the number of 2×2 switches of a system is not necessarily equal, or even close, to its entropy.

Let's introduce a new type of construction to bridge this apparent gap between entropy and complexity. As illustrated in Figure 2, a construction emulating the previous example is a column of $\frac{N}{2}$ switches, which are all controlled by the same control input. If the common control input is set to the "bar" state, the inputs are passed forward to the output. If the control input is set to the "cross" state, the locations of each pair of inputs are interchanged. We will define such a set of switches, which are all controlled by a single control input, as a *super switch*.

Definition 10: A *super switch* is an ensemble of 2×2 switches, which are all controlled by the same control input (see Figure 2). The number of 2×2 switches

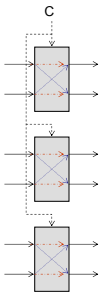

Fig. 2. A super switch

composing a super switch is called the *size* of the super switch.

The number of super switches in a construction equals the number of independent controls. These independent controls determine the internal states of the construction. Our goal will be to state a connection between the number of independent controls and the entropy of the network element.

In coding theory, the goal is to design a code such that the average length of codewords is minimized [14]. An optimal code is designed using the knowledge of the probabilities of source symbols.

It appears that the key in the connection between coding and switching lies in the number of independent controls introduced above: each digit in the codeword corresponds to one independent control. Similarly, each codeword corresponds to a given set of independent controls. However, the connection is not entirely straightforward: the number of digits in the codewords is not fixed, while the hardware in a construction is fixed. Therefore, we will define *active* and *passive* controls. For a specific external state, active controls are those that participate in forming that state, while the value of passive controls is irrelevant in forming the state. (Of course, controls that are active in forming one external state, might in turn be passive in forming another external state.)

Definition 11: A control input is called *active* for a specific external state if its value depends on the state. A control input is called *passive* if its value is predetermined independently of the state.

For instance, in the example above with the super switch (Figure 2), there is a single control input that is always active. If there were additional control inputs for switches that are never used, then these control inputs would be called passive.

Now we will define the *theoretical complexity* of an optical network element as the minimum expected number of active controls over all possible constructions emulating it. Our motivation will be to build optical constructions that reach that minimum.

Definition 12: Assume that there exists a probability distribution P_T on the set of external states T, and that l_{t_i} is the number of active controls necessary to form a state $t_i \in T$ in a given construction. The *theoretical complexity* C^* of a network element is the minimal expected number of active controls, where the minimum is taken over all possible emulating constructions:

$$C^* = \min \sum_{t_i \in T} P_T(t_i) l_{t_i}$$

For instance, consider the example above. There is a single control that is always active, therefore the expected number of active controls in the above construction is 1. This is an upper bound on the minimum expected number, thus $C^* \leq 1$. (In fact, we will see below that $C^* = 1$.)

We will now find lower and upper bounds on the theoretical complexity of a network element. First, we will only discuss memoryless network elements. Then, we will extend our discussion and also consider network elements with memory.

C. A lower bound on the theoretical complexity

The derivation of the lower bound on the theoretical complexity will parallel the derivation of the lower bound on the expected length of codewords in coding theory [14].

A known result related to coding is Kraft's inequality. It states that if the length of the codewords for code L are $l_1, l_2, ..., l_m$, then it holds that $\sum_i 2^{-l_i} \leq 1$. Now, we can state an equivalent theorem also for switches. We will use this theorem in finding a lower bound on the theoretical complexity. (For the sake of presentation, all proofs in this paper are presented in the Appendix.)

Lemma 1: (*Kraft's inequality*) Assume that the number of active controls for an external state $t_i \in T$ is given by l_{t_i}. It holds that

$$\sum_i 2^{-l_{t_i}} \leq 1.$$

By minimizing the expected number of active controls under Kraft's inequality, we get a lower bound on the theoretical complexity:

Theorem 1: Assume that the number of active controls for an external state $t_i \in T$ is given by l_{t_i} and that its probability is given by p_{t_i}. Then the theoretical complexity of the network element is lower bounded by its entropy:

$$C^* \geq H.$$

Example 5: Consider a system with 4 input links I_1, I_2, I_3, I_4. The outputs may be one of three possible

permutations of the inputs, with the following probability distribution:

$$P((O_1,O_2,O_3,O_4)=(I_1,I_2,I_3,I_4))=\frac{1}{2}$$
$$P((O_1,O_2,O_3,O_4)=(I_2,I_1,I_4,I_3))=\frac{1}{4}$$
$$P((O_1,O_2,O_3,O_4)=(I_3,I_4,I_1,I_2))=\frac{1}{4}$$

All the other permutations have a probability that is equal to zero.
The entropy of the system is provided by

$$H=\frac{1}{2}\log\frac{1}{2}+2*\frac{1}{4}\log\frac{1}{4}=\frac{3}{2}$$

Therefore, by Theorem 1, a lower bound on the theoretical complexity is given by: $C^* \geq \frac{3}{2}$.
To achieve the lower bound, we will build another construction that uses the prior knowledge of state probabilities, as demonstrated in Figure 3. Assume that if the control input is 0, the switch is in a "bar" state, and if the control input is 1, the switch is in a "cross" state. The packets arrive to a super switch of size 4. The super switch is controlled by C_1. If $C_1 = 0$, the inputs pass forward to the outputs. In this case, the value of C_2 is irrelevant, and we say that C_2 is passive. If $C_1 = 1$, the inputs are passed to the lower super switch which is controlled by C_2. C_2 chooses between the two permutations with lower probability. A calculation of the expected number of active controls L shows that for this construction:

$$L=\sum_{t_i\in T}P_T(t_i)l_{t_i}=\frac{1}{2}*1+\frac{1}{4}*2+\frac{1}{4}*2=\frac{3}{2}.$$

Therefore the expected number of active controls in the construction is equal to the entropy of the system. We can conclude that in that case, the theoretical complexity is equal to its entropy lower bound:

$$C^*=L=H=\frac{3}{2}.$$

The previous example presented a construction with a theoretical complexity that achieves the entropy lower bound. Note that the way that construction was built is similar to the construction of an optimal code, such as a Huffman code [14]. The idea is that the most probable states are formed by a small number of active controls, and the least probable states are formed by a larger number of active controls.

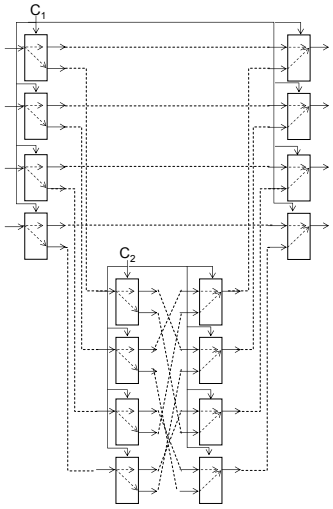

Fig. 3. An example of a construction that achieves the theoretical complexity

D. An upper bound on the theoretical complexity

In this section, we will find an upper bound on the theoretical complexity of memoryless network elements. The upper bound will be found by building a constructing that uses the Huffman coding. We will build a construction emulating a general system that has N inputs, M outputs and K states. It can perform a set of mappings, from a subset of inputs to the outputs. The only limitation on the operation of the system is that arriving packets cannot be duplicated, i.e., an input link cannot be connected to two output links. We will build a general emulation of such a system and show that an upper bound on the theoretical complexity is given by $C^* \leq H + 1$.

Definition 13: A *generalized space switch* is a network element with N inputs, M outputs and K states. There is a probability distribution over the states that is given by $\{\pi_i\}, 1 \leq i \leq K$.

We will build a construction emulating the generalized space switch. In order to do that, we will first define and implement two network elements: a distributer (also known as de-multiplexer), and a multiplexer.

Definition 14: A $1 \rightarrow K$ *distributer* is a network element with 1 input link, K output links and control links. The distributer places the arriving packet on one of the output links, according to the state of the controls.

Definition 15: A $K \rightarrow 1$ *multiplexer* is a network element with K input links, 1 output links and control links. According to the state of the controls, the packet on one of the input links is chosen and sent to the departure link.

Now, we will build a construction of a generalized space switch that uses distributers and multiplexers. The

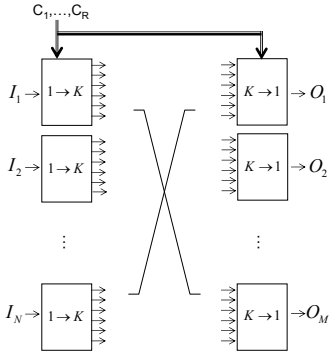

Fig. 4. A general construction with minimum expected number of active controls

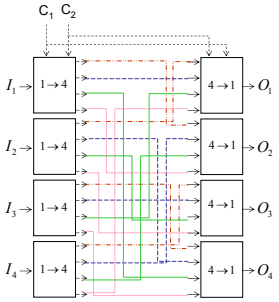

Fig. 5. An example of a construction of a generalized space shift

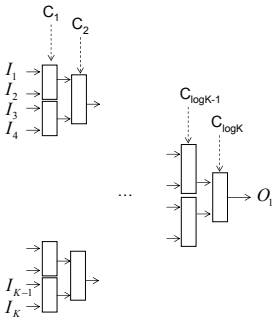

Fig. 6. A theoretically optimal construction of a $K \rightarrow 1$ multiplexer

construction is presented in Figure 4. It is built as follows: each input is connected to a $1 \rightarrow K$ distributer and each output is a departure link of a $K \rightarrow 1$ multiplexer. There is some interconnection pattern that connects the outputs of the $1 \rightarrow K$ distributers to the inputs of the $K \rightarrow 1$ multiplexers. That pattern is set in advance according to the desired external state space, it includes only fiber lines without any additional 2×2 switches. The same control links are connected to the multiplexers and to the distributers. The control links are set according to the desired state.

Theorem 2: The construction presented in Figure 4 is an emulation of a generalized space switch, with N input links, M output links and with K states.

We will present an example of such a construction.

Example 6: Consider a system with 4 inputs and 4 outputs. The outputs are a permutation of the inputs, and there are 4 possible equiprobable permutations:

$$\begin{aligned}(O_1, O_2, O_3, O_4) \quad \in \quad & \{(I_2, I_1, I_4, I_3) \\ & (I_1, I_4, I_3, I_2) \\ & (I_3, I_4, I_2, I_1) \\ & (I_4, I_1, I_2, I_3)\}\end{aligned}$$

A construction emulating that system is given in Figure 5. The inputs arrive to four parallel $1 \rightarrow 4$ distributers. According to the chosen permutation j, each input is placed on the j^{th} output link of the corresponding $1 \rightarrow 4$ distributer. The packets are then sent to four parallel $4 \rightarrow 1$ multiplexers, where the links are preset according to the set of permutations. Finally, the multiplexers are set according to the chosen permutation, and the outputs exactly follow the desired permutation.

Now, we want to minimize the expected number of active controls. Since the control inputs only affect the multiplexers and distributers, we should focus on building constructions of multiplexers and distributers with a minimum expected number of active controls. We will first present constructions of these elements for the case where the number of inputs/outputs is a power of 2 and all states are equiprobable. Then, we will show how to build a construction for the general case.

- *A $K \rightarrow 1$ multiplexer* has K different states, as the number of input links that might be chosen is K. Therefore, the entropy of a $K \rightarrow 1$ multiplexer is $H = \log K$. The construction of a $K \rightarrow 1$ multiplexer is presented in Figure 6. The control inputs are the inverse of the binary representation of the output link.
- *A $1 \rightarrow K$ distributer* has K different states, as the number of output links on which the arriving packet can depart is K. Therefore, the entropy of a $1 \rightarrow K$ distributer is $H = \log K$. A construction of a $1 \rightarrow K$ distributer is a mirror of the construction presented in Figure 6. The control inputs are equal to the binary representation of the output link.

Now, we will discuss the general case, where the states are non equiprobable and the number of inputs/outputs is not necessarily a power of 2. We will first bring an example, demonstrating the way to construct a $1 \rightarrow K$ distributer in the general case.

Example 7: Consider a system with one input and six outputs, where the probabilities that the input is routed to the i^{th} output are given by: $\pi = (\frac{1}{4}, \frac{1}{4}, \frac{1}{5}, \frac{1}{10}, \frac{1}{10}, \frac{1}{10})$

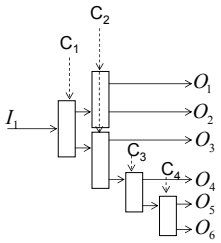

Fig. 7. A construction of a $1 \rightarrow 6$ distributer using the Huffman coding

With this example, the entropy is given by:

$$H = -\sum_i \pi_i \log \pi_i = 2.461$$

Using the Huffman procedure to generate a code that matches these probabilities, we get

$$Code = \{00, 01, 10, 110, 1110, 1111\}$$

A construction of a $1 \rightarrow 6$ distributer that matches that code is demonstrated in Figure 7.

By calculating the expected number of active controls we get:

$$L = \sum_i \pi_i Code_i = 2.5$$

The Huffman code in that case does not achieve the lower bound, but it is very close. Note that a construction of a $6 \rightarrow 1$ multiplexer with the same probabilities will be a mirror of the construction presented in Figure 7.

Recall that the Huffman code is optimal [14], in the sense that it minimizes the expected length of codewords. We claim that it is always possible to build a construction of a multiplexer or a distributer using the Huffman procedure, and that the *expected* number of active controls is minimized. Note that the number of controls is not necessarily minimized. For example, in the previous example, 4 controls are needed. If we use a construction of a $6 \rightarrow 1$ multiplexer with equiprobable states as presented in Figure 6, we will need only 3 controls. But since all the controls are active all the time that way, the expected number of active controls will be higher.

Lemma 2: It is possible to build a $K \rightarrow 1$ multiplexer or a $1 \rightarrow K$ distributer given some probability distribution by using the Huffman procedure. The resulting construction has a minimal expected number of active controls.

From coding theory, it is known that the expected length of codewords L of a code constructed by the Huffman procedure is upper bounded by $L \leq H + 1$. Now, we can state an equivalent upper bound here as well.

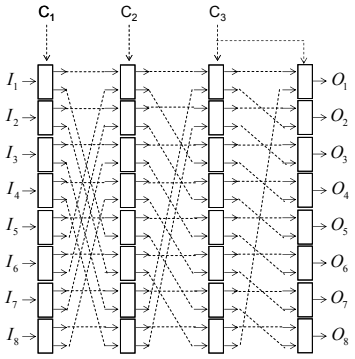

Fig. 8. A theoretically optimal construction of a shifter of 8 inputs

Theorem 3: The theoretical complexity is upper bounded as follows:

$$C^* \leq H + 1$$

Now that we have close lower and upper bounds, we get a good measure of the optimality of a construction by using the expected number of active controls.

E. Theoretically optimal constructions

We saw that it is always possible to build a construction by using the Huffman procedure, and that this construction is theoretically optimal. However, most of the time, this naive construction will not be practical due to the very high number of 2×2 switches necessary. Therefore, we will loosen the strict theoretical condition by only requiring that the expected number of active controls would grow like the theoretical complexity. Constructions satisfying this condition will be called *theoretically optimal*.

Definition 16: A construction is called *theoretically optimal* if its expected number of active controls is equal in growth to the theoretical complexity: $L = \Theta(C^*)$.

Let's show two examples of theoretically optimal constructions. The first example is a shifter, in which the inputs are shifted in space; and the second example is a switch with non equiprobable permutations.

Definition 17: A *shifter* is a system with N input links and N output links. The outputs of the shifter are equal to the inputs shifted by $0 \leq k \leq N-1$, i.e.,

$$(O_1, ..., O_N) = (I_{(k \bmod N)+1}, ..., I_{((k+N-1) \bmod N)+1}).$$

Example 8: We will build a theoretically optimal emulation of a shifter operating on N inputs with equal shifting probabilities, where N is a power of 2. A shifter has N different states, because k can receive N different values. Therefore, the entropy of a shifter is $H = \log N$. A shifter construction can be built as follows. The construction has $\log N$ stages, each stage

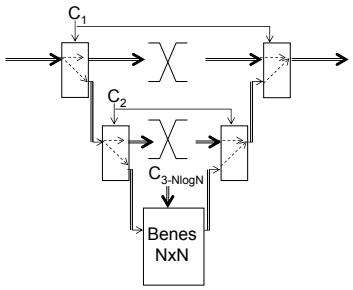

Fig. 9. A construction performing non-equiprobable permutations

is controlled by a single control input. The first stage can shift the inputs by $\frac{N}{2}$, the second stage can shift the inputs by $\frac{N}{4}$, and so on. We present in Figure 8 an example of a shifter, operating on 8 inputs. Since the number of independent controls is equal to the entropy of the shifter, the construction is theoretically optimal.

Example 9: Consider an $N \times N$ switch, where two permutations have a very high probability and the rest of the permutations have a very low probability, i.e., the probabilities of the permutations are given by:

$$\pi = (\frac{1-\epsilon}{2}, \frac{1-\epsilon}{2}, \frac{\epsilon}{N!-2}, ..., \frac{\epsilon}{N!-2}),$$

where $0 < \epsilon \ll 1$. The entropy of this system is approximately given by:

$$H \approx 1 + \epsilon \log(N!)$$

Now, consider the construction in Figure 9. For simplicity of drawing, each switch represents a super switch of length N. This construction has three main stages. The first stage chooses between performing the first dominant permutation or continuing to the other options. The second stage chooses between performing the second dominant permutation or continuing to the other options. The third stage is a full Benes network [15], [16] that performs the permutations with the lower probability. Note that although the number of switches in this construction is almost similar to the number of switches in a regular Benes network, the number of switch reconfigurations is much lower in this construction. This is due to the fact that a Benes network is optimized to the case where all permutations are equiprobable, while here, the state probabilities are far from being equal. Therefore, the proposed construction using the a-priori knowledge of the state probabilities is more suitable.
We will now show that this construction is theoretically optimal. In order to do so, we will calculate the expected number of active controls with this construction:

$$\begin{aligned} L &= \sum_i \pi_i l_{t_i} \\ &= \frac{1-\epsilon}{2} * 1 + \frac{1-\epsilon}{2} * 2 + \epsilon * (N \log N + 2) \\ &\approx H + \frac{1}{2} \end{aligned}$$

Therefore this construction is theoretically optimal.

We have shown bounds on the theoretical complexity, defined theoretical optimality and shown examples for memoryless systems. Now, we will extend our discussion also to systems with memory. We will show that all the results derived so far are relevant also to systems with memory.

F. Systems with memory

We will use a space-time transformation [3], [6], [7], to show that the results for memoryless systems are valid also for systems with memory. The time-space transformation is constructed by using fiber delay lines that align the packets arriving on different times to the same time and vice versa. First, we will have to define what is the expected number of active controls for systems with memory.

Definition 18: Assume that the external states form a set A, the state duration is T_S and there is a probability distribution associated with the set $P_a, a \in A$. The *theoretical complexity* C^* of a network element is the minimal expected number of active controls over all possible constructions. The number of active controls per state is the sum of active controls over the entire state duration.

$$C^* = \min P_a \sum_{t \in T_S} l_a(t),$$

where $l_a(t)$ is the number of active controls *on time-slot* t for state a.

Note that a control is passive if its value is predetermined only as a function of time, independently of the state. The next example demonstrates how to calculate the number of active controls for systems with memory.

Example 10: Consider the emulation of TSI presented in [7]. The construction includes $2 \log B - 1$ consecutive switches. The control of each switch is predetermined to the "cross" state during half of the times, independently of the state. During the other half, the value of the each control is set independently according to the desired state. We say that it is passive when it is predetermined to "cross", and that it is active when its value is set according to the state. A total of B packets pass through each switch, therefore there are $\frac{B}{2}$ active controls for

each switch. The total number of active controls in that case is:

$$L = \frac{B(2\log B - 1)}{2} = B\log B - \frac{B}{2}$$

In order to define theoretical optimality also for the case of systems with memory, we will check if the lower and upper bounds on the theoretical complexity are still valid.

The lower bound derived in Theorem 1 is still valid, as the arguments we used in proving it were related only to the number of states and their probabilities. We will show now that the upper bound derived in Theorem 3 is still valid. This is done using a space-time transformation [3], [6], [7]. The idea is that we can use a construction performing operations in space, in order to perform operations in time. Since the upper bound in Theorem 3 was proved by building a specific construction, the same construction is relevant also for systems with memory.

Theorem 4: Assume that the theoretical complexity is defined as in Definition 18. By using a time-space mapping, we get the following bounds on the theoretical complexity:

$$H \leq C^* \leq H + 1$$

Definition 16 of theoretically optimal constructions is valid also for systems with memory. Let's present an example of a theoretically optimal system with memory.

Example 11: Consider the entropy of a TSI, where all the permutations in time are equiprobable. It is equal to the entropy of a space switch:

$$H = \log(N!).$$

The theoretical complexity is therefore bounded as follows:

$$\log(N!) \leq C^* \leq \log(N!) + 1.$$

The expected number of active controls in the emulation of TSI presented in [7] is:

$$L = \frac{B(2\log B - 1)}{2} = B\log B - \frac{B}{2} = \Theta(C^*)$$

Therefore, the construction is theoretically optimal.

To conclude, in this section we have presented equivalences between coding theory and the construction of switching networks. We have defined the theoretical complexity of a system as the minimal expected number of active controls. We have found lower and upper bounds on the theoretical complexity, and shown that it is almost equal to the entropy of external states of the system. We have defined theoretical optimality, and shown examples of theoretically optimal constructions.

Until now, the measure of optimality we were interested in was the expected number of active controls, which is related to the number of switch reallocations. Now, we will consider the question of minimizing the number of 2×2 switches in the construction.

IV. Practical complexity versus theoretical complexity

A. Definition of practical complexity

Until now, we considered only the number of different states and their probabilities in calculating the theoretical complexity. The structure of states themselves had no significance. For example, the theoretical complexity of an unbuffered switch that performs $N!$ permutations is identical to the theoretical complexity of a TSI that performs $N!$ permutations in time. In this section, we want to find bounds on the actual number of 2×2 switches required to build a network element. It is clear that now we have to take into account also what the states are, and not only the number of states. For example, much less 2×2 switches are required to emulate a TSI performing $N!$ time permutations than to emulate an $N \times N$ switch performing $N!$ space permutations. Intuitively, this is because an $N \times N$ switch has to perform many actions simultaneously in order to set the state, and the actions the TSI has to perform are spread over the time frame.n this section, we will define the practical complexity of a network element as the number of 2×2 switches required to emulate a system, and illustrate the tradeoff between practical and theoretical complexity.

Definition 19: The *practical complexity* C of a construction is the number of 2×2 switches in the construction.

In order to find the connection between the number of states and the practical complexity, we will consider the state duration. We will assume that all the states have equal state duration. In order to find the connection between the number of states and the practical complexity we will use arguments similar to those mentioned in [5]. Consider the operation of C switches during T time-slots. If each switch is a single switch, i.e., not a part of a super switch, and is always active, the maximal number of internal states formed is 2^{CT}. Since the number of internal states is lower bounded by the number of external states, we get the following theorem:

Theorem 5: Assume that the number of different states in the construction is K. It holds that number of 2×2 switches C is lower bounded as follows:

$$C \geq \lceil \frac{\log K}{T} \rceil,$$

where T is the state duration.

Note that if the states are equiprobable, the theoretical complexity is $C^* = \log K$, and we get $C \geq \frac{C^*}{T}$. Now we can define a practically optimal construction to be a construction with a number of 2×2 switches that grows like the introduced lower bound.

Definition 20: Denote by C the practical complexity of a system with K states, and with state duration T. We say that a construction is *practically optimal* if $C = \Theta(\frac{\log K}{T})$, i.e., there is some constant a such that

$$\lceil \frac{\log K}{T} \rceil \leq C \leq a \frac{\log K}{T}$$

To illustrate this result, in [11], we present constructions of optical buffers and optically-buffered routers that are practically optimal. We first use the definitions above to provide a lower bound on the number of 2×2 switches required, and then present practical constructions with a number of 2×2 switches that grows like the introduced lower bound.

Definition 21: A construction that is both practically optimal and theoretically optimal will be called *optimal*.

Note that not every practically optimal construction is also a theoretically optimal construction, and not every theoretically optimal construction is also practically optimal. In fact, there often exists a tradeoff between theoretical and practical optimality. Let's present examples demonstrating this tradeoff.

Example 12: In Example 4, the construction emulating the system is a super switch of length $\frac{N}{2}$. When the control $c = 0$, the inputs are simply passed forward to the output. When the control $c = 1$, the positions of every two consequent packets are interchanged. The theoretical complexity of this system is $C^* = 1$ since there are only two states. Therefore, the construction is *theoretically optimal*, because $C^* = H$. However, it is not *practically optimal*, since the practical complexity is $C = \frac{N}{2} \gg 1$.

Example 13: The theoretical and practical complexity of an $N \times N$ switch with equiprobable permutations are given by $C^* = \log(N!) = \Theta(N \log N)$. Further, the number of switches (and of independent controls) of a Benes network is also $\Theta(N \log N)$ [15], [16]. Therefore, a Benes network is both *practically optimal* and *theoretically optimal*. Hence, it is called *optimal*.

Example 14: Consider the case of an $N \times N$ switch with two dominant permutations as presented in Example 9. On the one hand, emulating this system with a Benes network is practically optimal, but not necessarily theoretically optimal (because if the two dominant permutations are those from Example 4, it can easily be shown that a Benes network needs at least $N/2$ super switches). On the other hand, the construction presented in Figure 9, while theoretically optimal, would require more 2×2 switches than the Benes network.

The previous example illustrates the tradeoff between theoretical and practical complexity. Assuming that the reallocation of connections is an operation that requires time, we do not want to "waste" many reallocations on states that are very frequent. So for this example, a Benes network is highly wasteful and not theoretically optimal, even though it is practically optimal. On the other hand, note that a theoretically optimal construction for this case will require slightly more switches – although it is both practically optimal and theoretically optimal, and therefore optimal.

Example 15: The theoretical complexity of a TSI with buffer size B is given by $\log B!$, and its state duration is B. Therefore, its practical complexity is lower bounded as follows:

$$C \geq \frac{\log(B!)}{B} = \Theta(\log B)$$

In [7] a construction of a TSI with $2 \log B - 1$ 2×2 switches is presented. This construction is both practically optimal and theoretically optimal, and therefore optimal.

Example 16: The theoretical complexity of a frames switch with N input links and buffer size B is given by $\log((NB)!)$, and the state duration is B. Therefore, the practical complexity is lower bounded as follows:

$$C \geq \frac{\log((NB)!)}{B} = \Theta(N \log(NB))$$

In [6] a construction of a frames switch with $O(N \log(NB))$ 2×2 switches is presented. This construction is both practically optimal and theoretically optimal, and therefore optimal.

To conclude, we have presented a lower bound on the practical complexity, i.e., the minimal required number of 2×2 switches. We have defined a construction as practically optimal if the number of 2×2 switches in the construction achieves the lower bound. We have discussed the tradeoff that exists, in cases of non-equiprobable states, between the practical complexity and the theoretical complexity. A construction that is both practically optimal and theoretically optimal is simply called *optimal*.

V. Conclusion

In this paper we were interested in two different cost measures: number of switches and expected number of switch reconfigurations. First, we presented links between switching theory and coding theory. We found that the design of a network element with a minimized expected number of switch reconfigurations is equivalent

to the construction of optimal codes. Then, we presented a general construction of a switch that achieves cost lower bounds given some state probability distribution. Finally, we argued that such a construction is not always practical, and discussed the question of minimizing the number of 2×2 switches in the system.

As noted above, this paper lays a fundamental ground, by providing lower bounds on the complexity of practical implementations. In [11], we use these fundamental results to present constructions of optical buffers and optically-buffered routers that are practically optimal, i.e., have a number of basic components that grows like the practical complexity lower bound defined in this paper.

ACKNOWLEDGMENTS

The authors would like to acknowledge the support of the ATS-WD Career Development Chair.

REFERENCES

[1] N. McKeown, *Internet Routers: Past, Present and Future,* BCS Ada Lovelace Lecture, London, June 2006.
[2] C. E. Shannon, *Memory requirements in a telephone exchange*, Bell Syst. Tech. J., vol. 29, pp. 343–349, 1950.
[3] M. J. Marcus, *New approaches to the analysis of connecting and sorting networks*, Technical Report, 1972.
[4] N. Pippenger, *The complexity theory of switching setworks*, Technical Report, 1973.
[5] A. Ephremides, B. Hajek, *Information theory and commumication networks: an unconsummated union*, IEEE Transactions on Information Theory, vol. 44, no. 6, Oct. 1998.
[6] D. K. Hunter and D. G. Smith, *New architectures for optical TDM switching*, IEEE Journal on Lightwave Technology, vol. 11, pp. 459-511, March 1993.
[7] F. Jordan, D. Lee, K.Y. Lee, and S.V. Ramanan, *Serial array time slot interchangers and optical implementations*, IEEE Transactions on Computers, vol. 43, no. 11, pp. 1309-1318, 1994.
[8] R. Kannan, D. Lee, K.Y. Lee and H.F. Jordan, *Optical TDM sorting networks for high-speed switching*, IEEE Transactions on Communications, vol. 45, no. 6, pp. 723-736, June 1997.
[9] C. S. Chang, D. S. Lee and C. K. Tu, *Recursive construction of FIFO optical multiplexers with switched delay lines*, IEEE Transactions on Information Theory, vol. 50, pp. 3221-3233, 2004.
[10] C. S. Chang, Y. T. Chen, and D. S. Lee, *Construction of optical FIFO queues and non-overtaking delay lines*, Technical Report, 2005.
[11] H. Kogan, I. Keslassy, *Optimal-complexity optical router*, IEEE Infocom '07, Anchorage, AK, 2007.
[12] S. Chuang, A. Goel, N. McKeown, B. Prabhakar, *Matching output queueing with a combined input/output-queued switch*, IEEE J.Sel. Areas in Communications, Vol. 17, no. 6, pp. 1030-1039, June 1999.
[13] A. Waksman, *A permutation network*, J. Assoc. for Comput. Mach., vol. 15, no. 1, pp. 159 163, Jan. 1968.
[14] T. M. Cover, J. A. Thomas, *Elements of information theory*, 1st Edition. New York: Wiley-Interscience, 1991.
[15] V. E. Benes., *Mathematical Theory of Connecting Networks and Telephone Traffic*, New York: Academic Press, 1965.
[16] J. Hui, *Switching and Traffic Theory for Integrated Broadband Networks,* Boston: Kluwer Academic Publishers, 1990.

APPENDIX

A. Proof of Lemma 1

The proof is based on the proof of Kraft's inequality provided in [14]. First, let's show that the controls of the construction form a prefix code. (A code is called a prefix code if no codeword is the prefix of another codeword.) Let's assume that this is not the case. Then there would exist at least one case in which packets either form a state (i.e., the controls form a codeword), or they continue being routed through other switches (i.e., the controls form a prefix for another codeword). Since these other switches are not influenced by the controls, they are passive. But by definition, the above case cannot happen, because passive switches cannot lead to two different states. Therefore, it is not possible that the controls do not form a prefix code.

Now, assume that the number of control inputs is L, and the number of active controls for an external state $t_i \in T$ is given by l_{t_i}. It holds that:

$$L \geq \max_i l_{t_i}$$

Let S be the set of internal states, and T the set of external states. We already saw that $|T| \leq |S|$. Further, since the number of active controls is upper-bounded by L, and each of these controls does not have more than two possible settings, we get $|S| \leq 2^L$. Therefore, the total number of external states $|T|$ is upper bounded as follows:

$$|T| \leq |S| \leq 2^L.$$

For each state defined by l_{t_i} active controls, there exists a group of "descendant internal states" that is formed by setting the passive controls in all the possible combinations. Each state has no more than $2^{L-l_{t_i}}$ descendant states, due to the fact that there are $2^{L-l_{t_i}}$ different combinations of passive controls. The descendant internal states of different states are disjoint, because the controls form a prefix code. Since the maximal number of internal states is 2^L, we get the following inequality:

$$\sum_i 2^{L-l_{t_i}} \leq 2^L,$$

or:

$$\sum_i 2^{-l_{t_i}} \leq 1.$$

B. Proof of Theorem 1

The proof is based on [14]. Assume that the number of control inputs is L, and the number of active controls for an external state $t_i \in T$ is given by l_{t_i}. The expected number of active controls is given by

$$L = \sum_i p_i l_{t_i}.$$

We would like to minimize the expected number of active controls under the condition of Lemma 1:

$$\sum_i 2^{-l_{t_i}} \leq 1.$$

Obviously, since the theoretical complexity C^* is the result of the same minimization, but over a set that is included in the above (by Lemma 1), the result of this minimization will be a lower bound on C^*.

Using Lagrange multipliers, we should minimize the following term:

$$J = \sum_i p_i l_{t_i} + \lambda(\sum_i 2^{-l_{t_i}} - 1).$$

Differentiating with respect to l_i we obtain:

$$\frac{\partial J}{\partial l_i} = p_i - \lambda 2^{-l_i} \log 2.$$

Setting the derivative to zero and substituting in the constraint, we finally get an expression for l_i^*:

$$l_i^* = -\log p_i.$$

(Note that this number of control inputs is not necessarily an integer.) The resulting minimum expected number of control inputs is:

$$L^* = \sum_i p_i l_i^* = -\sum p_i \log p_i = H.$$

Therefore, the theoretical complexity C^* is lower bounded by the entropy of the network element: $C^* \geq H$.

C. Proof of Theorem 2

We will show that the construction presented in Figure 4 is an emulation of a generalized space switch, with N input links, M output links and with K states. Each state is a mapping of a subset of the input packets $P_1, ..., P_N$ to the outputs. Assume that the states are described by:

$$S_i = \{P_{j_{1,k}}, ..., P_{j_{M,k}}\}, k = 1, ..., K$$

We want to emulate that set of states by using the construction presented in Figure 4. At the output stage, when state S_k is chosen by setting the control inputs accordingly, packets $\{P_{j_{l,k}}, ..., P_{j_{M,k}}\}$ must be valid at the k^{th} inputs of the M multiplexers. At the input stage, when state S_k is chosen by the same control inputs, the arriving packets are sent to the k^{th} outputs of the distributers.

The routing of packets to the appropriate outputs is done by setting connecting wires between the outputs of the distributers and inputs of multiplexers. For example, the setting of the k^{th} state is done by setting a connection between the k^{th} output of each distributer to the appropriate k^{th} input of each multiplexer. If the packet $P_{j_{i,k}}$ must be valid at the k^{th} input of the i^{th} multiplexer, than a connection must be set between the k^{th} output of distributer $j_{i,k}$ and the k^{th} input of the i^{th} multiplexer. If the appropriate connections are set in advance, than the appropriate packets are valid at the inputs of the multiplexers and the corresponding states are set.

D. Proof of Lemma 2

It is proven in [14] that given some probability distribution, the Huffman coding is optimal. If C^* is the Huffman code and C is any other code, then $L(C^*) \leq L(C)$, where L is the expected length of codewords.

Here, we will show that it is possible to build a $K \rightarrow 1$ multiplexer or a $1 \rightarrow K$ distributer, given some probability distribution, by using the Huffman procedure. For simplicity, we will not not prove here that the expected number of active controls is minimized, as the proof with induction is similar to the proof for codewords.

The first stage in the construction of a $K \rightarrow 1$ multiplexer or a $1 \rightarrow K$ distributer is to construct a Huffman code for the given probability distribution. This is done by combining the two least likely states, where the combined state receives the probability of the sum, until we are left with only one combined state. Then, codewords are assigned to the states. The number of bits in a codeword is equal to the number of times it was combined with other states. That procedure forms a tree, where the leafs are the codewords.

An optical construction of a $1 \rightarrow K$ distributer by the Huffman coding is built by forming the tree, where the input is the root, the leafs are the outputs and the nodes are 2×2 switches. The number of active switches for a specific state is the codeword representing that state with the Huffman coding. The expected number of active switches is minimized because the expected code length is minimized with Huffman procedure.

An optical construction of a $K \rightarrow 1$ distributer is a mirror construction of the corresponding $1 \rightarrow K$ multiplexer with the same probability distribution.

E. Proof of Theorem 3

It is shown in [14] that with Shannon coding, the length of a codeword is

$$l_{t_i} = -\lceil \log p_i \rceil.$$

Since the expected code length with Huffman coding is lower that the expected length of any other code, it is

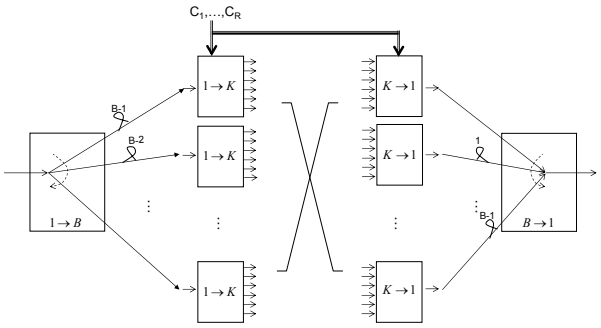

Fig. 10. A system with memory performing K states with non-equiprobable permutations

also lower than the expected code length with Shannon code, therefore:

$$C^* \leq -\sum_i p_i \lceil \log p_i \rceil,$$

and since $\lceil \log p_i \rceil \leq \log p_i + 1$

$$C^* \leq -\sum_i p_i \log p_i + \sum_i p_i = H + 1.$$

F. Proof of Theorem 4

We will use a time-space mapping, so as to build a system operating in time using a system operating in space. We will show that the extra switches necessary to implement the space-time mapping do not increase the theoretical complexity, as their switching pattern is predetermined as a function of the time and is independent of the chosen state.

Assume that we want to implement a system with memory, that receives a time frame of B slots. The outputs are one of K states, where each state defines a subset of the inputs, permuted according to some ordering.

A construction emulating this system is shown in Figure 10. The input link is connected to a $1 \rightarrow B$ distributer that operates in a round robin fashion. Output i of the distributer is delayed by $B - i$ time-slots. Therefore, the B packets of the time frame are aligned in time and reach together the central block. The central block is a memoryless system as presented in Figure 4, with K states emulated using the Huffman coding. The i^{th} output is delayed by $i-1$ time-slots, and the outputs are concatenated using a $B \rightarrow 1$ multiplexer. Therefore, the system performs the K desired states in time, by transforming the inputs to the space domain, performing the operation in space, and transforming back to the time domain.

Now, consider the theoretical complexity of this construction. The transformations to the space domain and back do not increase the number of active switches, as they depend only on time and not on the desired state. Therefore, the theoretical complexity of a system performing K states in time with some probability distribution on the states is identical to that of a system performing K states in time with the same probability distribution. And the upper bound $C^* \leq H + 1$ holds also for systems with memory.

G. Proof of Theorem 5

Consider the operation of C switches during T time-slots. There are $|S| = 2^{CT}$ internal states. Since there are K external states and the number of internal states is lower bounded by the number of external states, we get the following inequality:

$$2^{CT} \geq K.$$

Since C, the number of switches in the construction, must be an integer, we get the following inequality:

$$C \geq \lceil \frac{\log K}{T} \rceil.$$