Placement and Read Algorithms for High Throughput in Coded Network Switches

Rami Cohen and Yuval Cassuto
Department of Electrical Engineering
Technion - Israel Institute of Technology
Technion City, Haifa 3200003, Israel
Email: rc@campus.technion.ac.il, ycassuto@ee.technion.ac.il

***Abstract*—Coded switches write incoming packets with redundancy to increase the flexibility to read them later without contention. An important question pertaining to coded switches is what policy to follow when placing the coded packets in the switch memory. We study this question by proposing two such placement policies: cyclic placement and (block-) design placement. We show that these policies offer many advantages in switching throughput, algorithmic efficiency, and analysis amenability.**

I. Introduction

While mostly hidden from the public eye, network switches (and routers) are the workhorses carrying the immense growth of internet and cloud services. Scaling the switches to meet data demands is becoming more challenging with each product generation. To support the surging switching rates, the switches are designed in massively parallel architectures, with the switching fabric now comprising numerous memory units operated at their maximal bandwidth. With these parallel multi-unit memories come serious memory contention issues, whereby multiple requested packets need to be read at the same time instant from the same bandwidth-limited memory unit (MU).

An effective way to reduce memory contention is by introducing coding within the switch. Arriving packets are written to the switch memory with some prescribed redundancy, which later adds flexibility to avoid contention between multiple packets requested for read simultaneously. Each arriving packet is divided into k chunks, to which $n-k$ additional chunks are added by an encoder. The n chunks are then written to memory, one chunk per MU all written in parallel. Upon packet request, a subset of these n coded chunks are read to reconstruct the pure uncoded packet for outbound transmission. Coded switches work the tradeoff of a graceful increase in writing load, in return for potentially much higher read throughput.

Different approaches have been proposed for switch coding. In [1] and then in [2] coding is done under a strong model guaranteeing simultaneous reconstruction of worst-case packet requests. An alternative approach has been suggested in [3], wherein standard MDS codes are used with the objective of improving the average read throughput through efficient read algorithms for coded packets. This paper is a continuation of that latter approach in the direction of stronger and more practical switch coding schemes, which use existing codes in a clever way. Whereas the main focus of [3] is on maximizing read throughput for completely unstructured layout of packet chunks in MUs, here we add to the scope clever ways to place packet chunks in MUs at the write path. In particular, we develop two new practical packet placement policies, and provide their efficient optimal read algorithms. The first scheme, *cyclic placement*, exemplifies a case where the MDS property can be lifted without losing the k out of n property. The second, *(block-) design placement*, is based upon the use of combinatorial designs for choosing MU sets for the chunks, which admits an extremely efficient optimal read algorithm. For both placement schemes we derive closed-form analytical expressions giving lower and/or upper bounds on their average throughputs. The outcome from this study is that the block-design placement policy emerges as most attractive, thanks to both high throughput performance and low complexity of the read algorithm.

The paper is structured as follows. In Section II, we provide the problem formulation and conditions for the existence of a full-throughput solution. We then develop two practical placement policies and their optimal read algorithms in Section III. Finally, the paper is concluded in Section IV.

II. Problem Formulation and Analysis Tools

A. Problem formulation

A multi-MU memory system of a network switch (or router) writes packets upon their arrival, and reads them later for their outbound transmission. Out of the (typically numerous) packets currently stored in the switch memory, a set of L packets is requested each time instant, with the expectation that the memory system will read all of them in that time instant with no contention on MUs. In a coded switch, each packet consists of k chunks, which are MDS-encoded into n chunks. These encoded chunks are stored in n distinct MUs out of N available ones ($1 \leq k \leq n \leq N$), where overlapping is allowed (i.e., two or more packets may share one or more MUs). The MU index set is $\{0, 1, ..., N-1\}$, where for each packet an MU set $\mathcal{S}$ contains the n indices of the MUs storing the chunks of the packet. The L packets requested for read define a collection of L sets $\mathcal{S}_1, \ldots, \mathcal{S}_L$, specifying which MUs store the chunks of each packet. Our objective is to read as many packets as possible in a single time instant, under the constraint that each MU can read only one chunk per time instant due to limited memory bandwidth. Recall that k chunks out of the n encoded packet chunks are sufficient for recovering a packet. Hence given $\{\mathcal{S}_i\}_{i=1}^{L}$ we wish to assign MUs to packets such that at least k MUs are available to a packet, and the number of read packets is maximal. This problem is termed as the $[n,k]$-*maximal throughput problem* (nkMTP) [3].

Example 1: Suppose that the L = 3 requested packets are stored in the MU sets $\mathcal{S}_1 = \{1,2,3,4\}, \mathcal{S}_2 = \{1,2,5,6\}$ and $\mathcal{S}_3 = \{3,4,5,6\}$. If $k = 2$, then an optimal solution is to read packet 1 from MUs 1 and 2, packet 2 from MUs 5 and 6 and packet 3 from MUs 3 and 4, resulting in three read packets.

Without any assumption on the structure of the sets $\{\mathcal{S}_i\}_{i=1}^{L}$, nkMTP is NP-hard for $k \leq n \leq 3$ [3]. Our method around this hardness, which we develop in this paper, is to introduce structure to the sets, thereby simplifying both the read algorithm and the resulting read-throughput analysis. This structure will be introduced through two new packet-placement policies in the write path. Let us denote by L^* the maximal number of packets that can be read out of the L requested packets. We concentrate on the existence of an $L^* = L$ (full throughput) solution, which is desired in practice to avoid delaying or reordering the read packets before fulfilling the request.

B. Full-throughput conditions

We start with deriving a sufficient condition on the existence of an $L^* = L$ solution. An nkMTP instance can be represented as a generalized graph matching problem on a bipratite graph with the packets and MUs being the disjoint vertex sets [3]. In this representation, packet vertices need to be *matched* to disjoint sets of k MU vertices. According to the extended Hall's theorem [4], the L packet vertices can be matched (i.e., an $L^* = L$ solution exists) if and only if

$$\left| \bigcup_{j \in \mathcal{L}} \mathcal{S}_j \right| \geq k \left| \mathcal{L} \right| \tag{1}$$

for every subset $\mathcal{L} \subseteq \{1, 2, ..., L\}$. We will refer to (1) as the *Hall's condition*. In words, this condition requires at least $k|\mathcal{L}|$ distinct MUs in each $\mathcal{L}$ sub-family of the L MU sets. Let us extend the set notation to represent intersections of MU sets, that is, $\mathcal{S}_{\mathcal{I}} \triangleq \bigcap_{j \in \mathcal{I}} \mathcal{S}_j$, for $\mathcal{I} \subseteq \{1, 2, ..., L\}$.

Theorem 1: Let $\mathcal{S}_1, ..., \mathcal{S}_L$ ($L \geq 2$) be the MU sets of an nkMTP instance. Then an $L^* = L$ solution exists if

$$\forall i, j : i \neq j, \ |\mathcal{S}_i \cap \mathcal{S}_j| \leq \frac{2(n-k)}{L-1} \triangleq t_{\max}. \tag{2}$$

Proof Denote by $\Phi_{s,\mathcal{L}}$ the sum of cardinalities of intersections of s distinct sets taken from the MU sets indexed by a certain set $\mathcal{L}$

$$\Phi_{s,\mathcal{L}} = \sum_{\mathcal{I} \subseteq \mathcal{L}, |\mathcal{I}| = s} \left| \mathcal{S}_{\mathcal{I}} \right|. \tag{3}$$

As an example, if $\mathcal{L}$ is the set $\{1, 2, 3\}$, then $\Phi_{2,\mathcal{L}}$ is $|\mathcal{S}_1 \cap \mathcal{S}_2| + |\mathcal{S}_1 \cap \mathcal{S}_3| + |\mathcal{S}_2 \cap \mathcal{S}_3|$. Using the inclusion-exclusion principle, Hall's condition is equivalent to the requirement

$$n|\mathcal{L}| - \Phi_{2,\mathcal{L}} + \sum_{s=3}^{|\mathcal{L}|} (-1)^{s-1} \Phi_{s,\mathcal{L}} \geq k|\mathcal{L}| \tag{4}$$

for every $\mathcal{L} \subseteq \{1, 2, ..., L\}$, $|\mathcal{L}| \geq 2$ (note that for $|\mathcal{L}| = 1$ we get the requirement $n \geq k$ that always holds). The sum $\sum_{s=3}^{|\mathcal{L}|} (-1)^{s-1} \Phi_{s,\mathcal{L}}$ is non-negative, as it compensates for over-subtraction of pairwise intersection cardinalities in the inclusion-exclusion process. Therefore, (4) holds if the inequality

$$\Phi_{2,\mathcal{L}} \leq |\mathcal{L}| (n-k) \tag{5}$$

holds for every $\mathcal{L}$. We can bound $\Phi_{2,\mathcal{L}}$ by bounding the pairwise intersection cardinalities:

$$\Phi_{2,\mathcal{L}} = \sum_{i \neq j \subseteq \mathcal{L}} |\mathcal{S}_i \cap \mathcal{S}_j| \leq \binom{|\mathcal{L}|}{2} \max_{i \neq j \subseteq \mathcal{L}} |\mathcal{S}_i \cap \mathcal{S}_j|. \tag{6}$$

Finally, combining (6) and (5), the inequality (5) holds when

$$\max_{i \neq j \subseteq \mathcal{L}} |\mathcal{S}_i \cap \mathcal{S}_j| \leq \frac{2(n-k)}{|\mathcal{L}| - 1}. \tag{7}$$

We now observe that the condition of the theorem (2) implies (7) because $|\mathcal{L}| \leq L$ for every $\mathcal{L}$. ∎

In addition to the sufficient condition of Theorem 1, we also have a necessary condition for the existence of a full-throughput solution. Since each packet requires at least k MUs, the following *coverage condition* must hold

$$\left| \bigcup_{i=1}^{L} \mathcal{S}_i \right| \geq kL, \tag{8}$$

which amounts to requiring (1) only for $\mathcal{L} = \{1, 2, ..., L\}$. We will use the conditions above to obtain bounds on the probability of a full-throughput solution when random instances are considered. As the pairwise condition (2) is sufficient, its probability (when an instance is drawn at random) serves as a lower bound. On the other hand, the coverage condition (8) is necessary, and its probability serves as an upper bound.

III. Placement Policies and Optimal Read Algorithms

In this section, two placement policies that posses certain structural properties are introduced. In practice, packets are stored in MU sets in a random-like manner for improved write rates and better MU load balancing. Thus, we consider ensembles of random instances characterized by k, n, N, L and the placement policy in use, where an instance is obtained by an independent and uniform placement of L packets. We then provide probabilistic analysis of the full-throughput performance for each placement policy. Apart from probabilistic analysis, we provide efficient optimal read algorithms for both placement policies.

A. Cyclic placement

In the first placement policy we propose, termed as *cyclic* placement, the MU sets are restricted to contain n *cyclic consecutive* MU indices. As an example, if $n = 3$ and $N = 4$, the possible MU sets are $\{0, 1, 2\}, \{1, 2, 3\}, \{2, 3, 0\}$ and $\{3, 0, 1\}$. This policy is an extension of the consecutive placement policy [3], in which wrap-around is not allowed (i.e., the last two sets in the example above are forbidden). The advantage of the cyclic placement is that it distributes the load evenly among MUs, while the consecutive placement puts higher load on the lower/upper MU indices. We will also see that with cyclic placement the structure of the optimal read solution lifts the need of using MDS codes, where simple (e.g., binary) cyclic erasure codes are sufficient.

Let us think of the MUs as N discrete points on a circle, where an MU set is an *arc* covering n cyclic consecutive points. An example for such a circle-arc representation is provided in Fig. 1. To find a lower bound on the probability of a full-throughput solution in an instance drawn from a cyclic ensemble, we calculate the probability $p_{\text{pair}}^{\text{cyc}}$ that the pairwise intersection cardinalities of the MU sets are at most $t_{\max}$ (see Theorem 1). We assume that $t_{\max}$ is an integer, otherwise we take its integral part.

Theorem 2: Consider an instance drawn at random from a cyclic nkMTP ensemble. The probability that the maximum

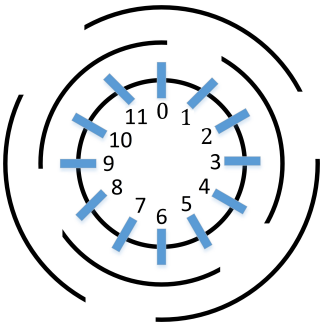

Fig. 1: Cyclic nkMTP instance in a circle-arc representation. The marks on the inner circle represent $N = 12$ MUs, where the $L = 6$ outer arcs represent packets stored each in $n = 4$ cyclic consecutive MUs.

pairwise intersection cardinality is at most $t_{\max}$ is

$$p_{\text{pair}}^{\text{cyc}} = N^{1-L} \prod_{i=1}^{L-1} \left(N - L\left(n - t_{\max}\right) + i\right). \tag{9}$$

Proof Consider a circle-arc representation of the cyclic nkMTP instances. Assume clockwise order, and that each packet arc does not precede the first packet arc. Each placed packet prevents the placement of the start of any other packet in its first $n-t_{\max}$ MUs. In a legal placement (i.e., when the pairwise intersection cardinality is at most $t_{\max}$), there are $N-L(n-t_{\max})$ MUs that do not belong to the first $n-t_{\max}$ MUs of any packet. Thus, the number of legal placements (given the order constraint above) is equivalently the number of ways to partition $N - L(n - t_{\max})$ MUs to L sets of cyclic consecutive MUs. Thinking of the latter MU sets as *gaps*, they can be distributed in $\binom{N-L(n-t_{\max})+L-1}{L-1}$ ways, which is the number of L non-negative integers (gap lengths) whose sum is $N - L(n - t_{\max})$ [5]. Each legal placement is obtained (uniquely) as a combination of 1) the starting MU for the first drawn packet, 2) a gap configuration, and 3) a permutation of the other $L - 1$ packets. Hence to get the total number of legal placements we multiply the number of gap configurations by N (the number of possible starting points for the first packet) and by $(L - 1)!$ (the number of permutations of $L-1$ packets). After normalizing by the total number of (legal and illegal) placements N^L, we obtain (9). ∎

Equipped with the circle-arc representation, the probability that a random cyclic instance satisfies the coverage condition (8) is the probability that at least kL points of the circle are covered once L arcs of n cyclic consecutive points are placed on the circle at random. In [6, Th. 1], the full probability distribution of the number of vacant points in this case is derived. From this distribution we find the probability that at least kL MUs are covered, which we denote by $p_{\text{cover}}^{\text{cyc}}$. Note that when an uncoded (i.e., $n = k$) cyclic placement is used, $p_{\text{cover}}^{\text{cyc}}$ becomes exact, as the coverage condition becomes both necessary and sufficient. To that end, we have both lower ($p_{\text{pair}}^{\text{cyc}}$) and upper ($p_{\text{cover}}^{\text{cyc}}$) bounds on the full-throughput probability for the cyclic placement.

As the packets are placed in n cyclic consecutive MUs, every intersection between two MU sets consists of cyclic consecutive MUs. Thus, there exists a solution of a cyclic nkMTP instance where the k MUs assigned to each read packet are cyclic consecutive. Based on this observation, we propose a read algorithm that finds a solution with this property. Let us assume a circle-arc representation. Define an order of the packets with respect to packet j_0, such that the packets are sorted according to their arcs' starting points relatively to packet j_0's starting point in clockwise order.

Example 2: Consider the cyclic instance in Fig. 1, where the order is with respect to the topmost packet arc ($\{11, 0, 1, 2\}$). The ordered packets are $\{11, 0, 1, 2\}$, $\{1, 2, 3, 4\}$, $\{3, 4, 5, 6\}$, $\{5, 6, 7, 8\}$, $\{7, 8, 9, 10\}$ and $\{9, 10, 11, 0\}$.

In the algorithm we begin with two empty sets Λ and Ω, which will eventually contain the read packets and their assigned MUs, respectively. We also initialize the sets Λ_j and Ω_j (for $j = 1, 2, ..., L$) as empty sets. The following algorithm solves optimally a cyclic nkMTP instance.

Algorithm 1: (Cyclic placement, optimal read algorithm) For $j = 1, 2, ..., L$, do:

1) Consider the set of packets $\left\{\tilde{\mathcal{S}}_i\right\}_{i=1}^{L}$ sorted with respect to packet j. Set $i := 1$.
2) If $|\tilde{\mathcal{S}}_i| \geq k$, add i to Λ_j, and add the first k MUs in $\tilde{\mathcal{S}}_i$ to Ω_j. Remove the added MUs from all other packets.
3) Set $i := i + 1$. If $i \leq L$, go to Step 2. Otherwise, go to Step 4.
4) If $|\Lambda_j| > |\Lambda|$, set $\Lambda := \Lambda_j$, $\Omega := \Omega_j$.

Theorem 3: The set of packets Λ and their corresponding MUs in Ω found by Algorithm 1 are an optimal solution to a cyclic nkMTP instance.

Proof As we saw, there exists an L^* solution in which the k MUs assigned to a read packet are cyclic consecutive. We further show that without loss of generality, there is at least one packet j_0 in the solution that is assigned its first k MUs. If there is no such packet, we can shift the solution counter-clockwise until this condition is met. Given that j_0 is a packet assigned its first k MUs, we prove that the algorithm finds the optimal solution in iteration $j = j_0$. To prove this, note that once the packets are sorted with respect to packet j, we have a consecutive nkMTP instance. This instance can be solved greedily as in steps 2-3 of Algorithm 1, as proved in [3, Theorem 5]. Finally, maximizing the size of the packet set Λ_j over all indices j is guaranteed to give the optimal solution, because at least one packet j is qualified as a j_0 that is in the optimal solution with its first k MUs. ∎

Algorithm 1 requires simple sorting and comparison operations rendering this algorithm efficient. The k MUs assigned to a read packet in Algorithm 1 are cyclic consecutive. Thus, the $n - k$ MUs not assigned to the packet are cyclic consecutive as well, and they can be viewed as a cyclic *burst* of $n-k$ *erasures*. An example is shown in Fig. 2. Such bursts can be corrected using an $[n, k]$ binary cyclic code, as cyclic codes are capable of recovering from any cyclic burst erasure of length up to $n - k$ [7]. The use of binary cyclic codes simplifies the coding process considerably. The reason is that non-trivial MDS codes require non-binary field arithmetic and impose certain restrictions on the code parameters, which can mostly be lifted once cyclic binary codes are used.

B. Design placement

In this sub-section, we propose a new placement policy, based on *combinatorial block designs*, which turns out to be the most promising. Inspired by the sufficient condition of Theorem 1, we choose for the packets MU sets with overlap at most $t_{\max}$. We do so by using the so called t-designs [5] with carefully chosen parameters. A t-(N, n, λ)

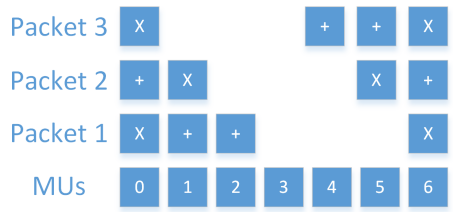

Fig. 2: A solution of a cyclic nkMTP instance ($k = 2, n = 4$). '+' denotes an MU assigned to the packet where 'X' denotes an MU not assigned to the packet (erasure).

design consists of n-element subsets (*blocks*) taken from a set of N elements, such that every t elements taken from the set appear in exactly λ subsets. Constructing designs for a certain set of parameters is largely an open problem, and one usually refers to a list of known block designs, such as in [8]. Our interest lies in block designs with $t = t_{\max} + 1$ and $\lambda = 1$, such that the pairwise intersection cardinality is at most $t_{\max}$. When $\lambda = 1$, t-designs are known as *Steiner systems*, and they contain (when exist) $b = \binom{N}{t}/\binom{n}{t}$ blocks [5]. We use the notation t-(N, n) for Steiner systems (where $\lambda = 1$ is implied).

Example 3: Consider the set $\mathcal{S} = \{1, 2, 3, 4, 5, 6, 7\}$. The blocks $\mathcal{S}_1 = \{1, 2, 3\}, \mathcal{S}_2 = \{1, 4, 5\}, \mathcal{S}_3 = \{1, 6, 7\}, \mathcal{S}_4 = \{2, 4, 6\}, \mathcal{S}_5 = \{2, 5, 7\}, \mathcal{S}_6 = \{3, 4, 7\}$ and $\mathcal{S}_7 = \{3, 5, 6\}$ form a 2-$(7, 3)$ Steiner system. There are $\binom{7}{2}/\binom{3}{2} = 7$ blocks in this design, known as *Fano plane* [5].

The placement policy we propose here, termed *design* placement, is based on constraining the MU sets to be the design blocks. It is sufficient that the L MU sets are different design blocks, and the request is full-throughput by the design properties and the sufficient pairwise condition (2). Thus, the probability that a random design nkMTP instance contains a full-throughput solution is lower bounded by the probability that the L MU sets are distinct. To find this probability, we use the balls-and-bins model [9]. In this model, there are L balls and b bins (recall that b is the number of blocks in the design), where the balls are placed independently and uniformly at random in the bins. The probability of L non-empty bins serves as a lower bound on $\Pr(L^* = L)$ for a design nkMTP ensemble, and is given as

$$p_{\text{pair}}^{\text{des}} = \binom{b}{L} \frac{1}{b^L} \sum_{j=0}^{L} (-1)^j \binom{L}{j} (L-j)^L. \qquad (10)$$

We note that if $k > n/2$, each block can serve only one packet, and $p_{\text{pair}}^{\text{des}}$ is the probability of a full-throughput solution rather than a lower bound.

Let us take a concrete example. For the case $L = 3$ and $n = k + 1$ we can take the 2-$(k^2 + k + 1, k + 1)$ Steiner system [5], which is suitable for a system with $N = k^2 + k + 1$ MUs. The sufficient pairwise condition (2) requires for these parameters $t_{\max} = n - k = 1$, which is guaranteed by the $t = 2$ parameter of the design. To have a full-throughput solution we need that the $L = 3$ blocks drawn from the $k^2 + k + 1$ blocks of the design will be all distinct. In Fig. 3, we plot $p_{\text{pair}}^{\text{des}}$, the design nkMTP lower bound on $\Pr(L^* = L)$, in comparison to the cyclic nkMTP lower and upper bounds on $\Pr(L^* = L)$ ($p_{\text{pair}}^{\text{cyc}}$ and $p_{\text{cover}}^{\text{cyc}}$, respectively). For comparison we plot the upper bound on the full-throughput probability when no structure is imposed

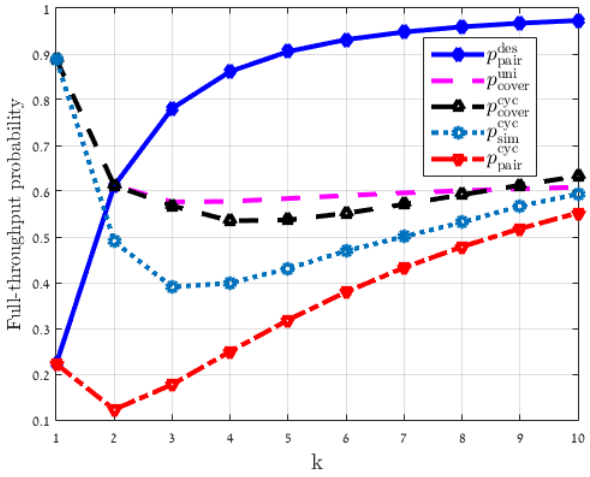

Fig. 3: A comparison of full-throughput performance bounds ($n = k + 1, N = k^2 + k + 1$).

on the MU sets, denoted $p_{\text{over}}^{\text{uni}}$, as derived in [3]. The exact probability for an uncoded cyclic placement is found using the coverage condition (8), and in this case it coincides with $p_{\text{cover}}^{\text{cyc}}$. We also plot $p_{\text{sim}}^{\text{cyc}}$, the only graph in Fig. 3 obtained using simulations, which is the empirical $\Pr(L^* = L)$ in the cyclic case. The results clearly demonstrate that the design policy exhibits significantly superior performance.

We now turn to show how a design nkMTP instance can be solved efficiently. We assume here that the L packets are stored in L distinct MU sets (blocks), otherwise a subset of packets stored in distinct blocks is considered. It turns out, and proven next, that an extremely efficient optimal read algorithm exists owing to the fact that the sufficient pairwise condition is strong enough that an optimal solution does not need to use MUs contained in sets of *more than two* packets. Denote by $\dot{\mathcal{S}}_i$ the MUs indexed in $\mathcal{S}_i$ that are not shared by any other packet, and by $\dot{\mathcal{S}}_{ij}$ the MUs indexed in both $\mathcal{S}_i$ and $\mathcal{S}_j$ but not in any other MU set. Clearly, the sets $\dot{\mathcal{S}}_i$ and $\dot{\mathcal{S}}_{ij}$ are disjoint. In addition, denote by x_{ij}^i (resp. x_{ij}^j) the number of MUs assigned in the optimal solution to packet i (resp. j) out of the set $\dot{\mathcal{S}}_{ij}$ (where two disjoint subsets of MUs are taken from $\dot{\mathcal{S}}_{ij}$). We have to find x_{ij}^i and x_{ij}^j using the following system of inequalities

$$\begin{aligned} &|\dot{\mathcal{S}}_i| + \sum_{j \neq i} x_{ij}^i \geq k, \ \text{for } i = 1, 2, ..., L, \qquad (11) \\ &\text{s.t. } \ x_{ij}^i + x_{ij}^j \leq |\dot{\mathcal{S}}_{ij}|, \ \text{for } 1 \leq i \neq j \leq L. \end{aligned}$$

The above is an integer linear program (ILP) admitting a total unimodular constraint matrix, hence can be solved optimally and efficiently by a standard linear program (LP). However, it can be proved that a much simpler algorithm than LP can solve this optimally. For clarity and lack of space, we pretend in the sequel that the sizes of the sets $\dot{\mathcal{S}}_{ij}$ are all even. We later explain how we can adapt the assignment algorithm and correctness proof to the general case.

Proposition 4: Assume that $\left|\dot{\mathcal{S}}_{ij}\right|$ are all even. Then setting $x_{ij}^i = x_{ij}^j = \left|\dot{\mathcal{S}}_{ij}\right|/2$ results in a valid solution of (11).

Proof According to the inclusion-exclusion principle,

$$\left|\dot{\mathcal{S}}_i\right| = \sum_{\mathcal{J}\supseteq\{i\}} (-1)^{|\mathcal{J}|-1} |\mathcal{S}_{\mathcal{J}}|, \tag{12}$$
$$\left|\dot{\mathcal{S}}_{ij}\right| = \sum_{\mathcal{J}\supseteq\{i,j\}} (-1)^{|\mathcal{J}|-2} |\mathcal{S}_{\mathcal{J}}|.$$

$\left|\dot{\mathcal{S}}_{ij}\right|/2$ are integers, since $\left|\dot{\mathcal{S}}_{ij}\right|$ are all even. Let us examine a certain i. Substitute the right-hand sides of (12) in the first inequality of (11) as follows: $\left|\dot{\mathcal{S}}_i\right|$ is substituted by the sum expanding it in (12), and each x_{ij}^i is substituted by half the sum expanding $\left|\dot{\mathcal{S}}_{ij}\right|$. Each set $\mathcal{J}\supseteq\{i\}$ appears in the combined sums once due to $\left|\dot{\mathcal{S}}_i\right|$, and additional $|\mathcal{J}|-1$ times (with an opposite sign) due to the summation of $\left|\dot{\mathcal{S}}_{ij}\right|$ over $j\neq i$. The left-hand side of the first inequality in (11) then becomes

$$n - \frac{1}{2}\sum_{j\neq i}|\mathcal{S}_{ij}| + \frac{1}{2}\sum_{\mathcal{J}\supseteq\{i\},|\mathcal{J}|\geq 3}(-1)^{|\mathcal{J}|-1}(3-|\mathcal{J}|)\,|\mathcal{S}_{\mathcal{J}}| \tag{13}$$
$$= n - \frac{1}{2}\sum_{j\neq i}|\mathcal{S}_{ij}| + \frac{1}{2}\sum_{\mathcal{J}\supseteq\{i\},|\mathcal{J}|\geq 4}(-1)^{|\mathcal{J}|}(|\mathcal{J}|-3)\,|\mathcal{S}_{\mathcal{J}}|.$$

We claim that the right sum in (13) is non-negative. This sum counts the number of occurrences of MUs in intersection sets of 4 packets or more that include packet i, multiplied by the factor $(|\mathcal{J}|-3)/2$, and with alternating signs. Consider a certain MU shared by exactly $T\geq 4$ packets including packet i. This MU appears in $\binom{T-1}{|\mathcal{J}|-1}$ intersection sets of cardinality $4\leq|\mathcal{J}|\leq T$ (we subtract 1 as the packet index i is always contained in $\mathcal{J}$). The contribution of this MU to the count is

$$\frac{1}{2}\sum_{|\mathcal{J}|=4}^{T}(-1)^{|\mathcal{J}|}(|\mathcal{J}|-3)\binom{T-1}{|\mathcal{J}|-1} \tag{14}$$
$$= \frac{1}{2}\sum_{|\mathcal{J}|=0}^{2}(-1)^{|\mathcal{J}|}(|\mathcal{J}|-2)\binom{T-1}{|\mathcal{J}|} = (T-3)/2\geq 0,$$

where we used the binomial identities

$$\sum_{j=0}^{T}(-1)^j\binom{T}{j} = \sum_{j=0}^{T}j(-1)^j\binom{T}{j} = 0. \tag{15}$$

This means that the right sum in (13) is indeed non-negative. Now, $|\mathcal{S}_{ij}|\leq t_{\max}$, so (13) is lower-bounded by

$$n - \frac{1}{2}\sum_{j\neq i}t_{\max} = n - \frac{1}{2}(L-1)\,t_{\max} = k, \tag{16}$$

meaning that at least k MUs are assigned to packet i. ∎

We now sketch how the simple half assignment of Proposition 4 can be adapted to the case when $\left|\dot{\mathcal{S}}_{ij}\right|$ are not all even. In this case, there exists an optimal assignment of either *floor/ceiling* values of $\left|\dot{\mathcal{S}}_{ij}\right|/2$ to x_{ij}^i and x_{ij}^j. To find this assignment, we construct an undirected graph whose vertices are the packet indices, and connect two vertices i and j by an edge if $\left|\dot{\mathcal{S}}_{ij}\right|$ is odd. We then remove from the graph vertices not connected by an edge to any other vertex. An *orientation* of an undirected graph is an assignment of a direction to each edge in the graph (leading to a directed graph). There always exists an orientation such that the number of edges entering and exiting a vertex differs by at most one, and it can be found in in time linear in the number of edges [10]. An example is shown in Fig. 4. Equipped with such an orientation of the undirected graph constructed above, an edge entering vertex j from vertex i is interpreted as contributing the floor value of $\left|\dot{\mathcal{S}}_{ij}\right|/2$ to x_{ij}^j. An edge in the opposite direction contributes the ceiling value of $\left|\dot{\mathcal{S}}_{ij}\right|/2$ to x_{ij}^i. In addition, $\dot{\mathcal{S}}_{ij}$ of even cardinality contribute $\left|\dot{\mathcal{S}}_{ij}\right|/2$ to both x_{ij}^i and x_{ij}^j. By extending Proposition 4, the assignment above can be proved to be optimal.

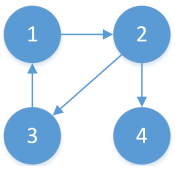

Fig. 4: An orientation example where the number of edges entering/existing a vertex differs by at most one.

IV. Conclusion

In this paper we studied two coded-switch placement policies and their respective read algorithms. The advantages of the cyclic placement is that cyclic binary codes can be used instead of MDS codes, and a maximal-throughput solution can be found with an efficient algorithm. The advantage of the design placement is that full-throughput solutions can be found instantly by an extremely simple assignment algorithm. Both policies are shown to be amenable for closed-form analysis, providing an important tool to predict the performance of coded switches. Important future directions are finding tractable algorithms for weaker sufficient conditions, considering other natural placement policies, and constructing non-MDS codes that give similarly good throughput performance in this framework.

Acknowledgement

This work was supported in part by the Israel Science Foundation, the Israel Ministry of Science and Technology, and by the Intel ICRI-CI center.

References

[1] Z. Wang, H. M. Kiah, and Y. Cassuto, "Optimal binary switch codes with small query size," *2015 IEEE International Symposium on Information Theory (ISIT)*, pp. 636–640, June 2015.
[2] Z. Wang, O. Shaked, Y. Cassuto, and J. Bruck, "Codes for network switches," *2013 IEEE International Symposium on Information Theory Proceedings (ISIT)*, pp. 1057–1061, July 2013.
[3] R. Cohen and Y. Cassuto, "Algorithms and throughput analysis for MDS-coded switches," *2015 IEEE International Symposium on Information Theory (ISIT)*, pp. 656–660, June 2015.
[4] M. Viderman, "LP decoding of codes with expansion parameter above 2/3," *Information Processing Letters*, vol. 113, no. 7, pp. 225 – 228, 2013.
[5] J. H. van Lint and R. M. Wilson, *A course in combinatorics*. Cambridge University Press, 2001.
[6] G. Barlevy and H. N. Nagaraja, "Properties of the vacancy statistic in the discrete circle covering problem," FRB of Chicago, Tech. Rep., 2015.
[7] W. Ryan and S. Lin, *Channel codes: Classical and modern*. Cambridge University Press, 2009.
[8] C. J. Colbourn and J. H. Dinitz, *Handbook of Combinatorial Designs, Second Edition (Discrete Mathematics and Its Applications)*. Chapman & Hall/CRC, 2006.
[9] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.
[10] A. Bondy and U. Murty, *Graph theory*. Springer, 2008.