Achieving 100% Throughput in a Two-Stage Multicast Switch

Ruisheng Wang* Youjian Zhao† Ting Zhou‡
Department of Computer Science & Technology, Tsinghua University
Beijing, P.R.China, 100084
{wangrs06*, zhaoyj†}@csnet1.cs.tsinghua.edu.cn, zhout06‡@mails.tsinghua.edu.cn

Abstract—It is known that single-stage IQ based switch can provide 100% throughput guarantee for any admissible unicast traffic pattern. However, due to particular characteristics of multicast traffic, the performance of IQ switches in the case of general mutlicast traffic partterns is inferior to its performance in the case of unicast traffic, which motives us to find new architectures to provide better performance guarantee for multicast traffic. In this paper, we propose a Two-Stage Multicast Switch (TSMS) which is a serial combination of a Multicast To Unicast (MTU) switch to copy input cells from various sources simultaneously and a Combined Input and Output Queueing (CIOQ) switch to deliver copies of multicast cells to their final destinations. Based on MTU switch, we design a novel LFCNF-UMBA scheduling algorithm to determine how to copy multicast cells into unicast cells. By coordinately using Maximal Matching scheduling algorithm in CIOQ switch, we prove that speedup of $2-2/(N+1)$ is necessary and 2 is sufficient for a $M \times N$ TSMS to achieve 100% throughput under any admissible multicast traffic pattern, which is also verified by our simulation results.

I. INTRODUCTION

AS new Internet applications of these days, one to many applications, including video-on-demand and live media streaming, and many to many applications, including video conferencing and multiplayer games, take a large proportion of Internet traffic. Therefore, how to support multicast is becoming a key issue in the design of high-performance switches and routers.

A single-stage crossbar based architecture is one kind of promising switch architectures since crossbar is a kind of non-blocking fabric. Based on crossbar, switches can be divided into Input Queued (IQ) architecture and Output Queued (OQ) architecture according to the location of the buffer. OQ switch has minimum delay time and is easy to implement Qos-guaranteed scheduling algorithms. However, it needs both fabric and output buffers to run at N times of link rates. While IQ switch is more suitable for implementing high-speed routers since it does not require speedup. [1] has proven that IQ based switch with maximum weight matching algorithm can achieve 100% throughput under admissible i.i.d. unicast traffic pattern. With speedup of two, IQ combined with output queueing switch can be stable under any admissible unicast traffic pattern [2].

However, when it turns into multicast, things have changed. A lot of researches are based on single multicast queue [3], however, due to the similar HOL problem in unicast case, its performance of throughput is very poor. In order to avoid HOL problem, one method is to implement MultiCast-VOQ [4] similar to unicast VOQ. However, this method needs to maintain $2^N - 1$ multicast queues, which is too complex to be implemented in hardware. On the other hand, to find the optimal scheduling is NP hard [5]. Some researchers propose some low-complex multicast queue architectures which maintain k $(1 < k \ll N)$ multicast queues in an input port [6], [7], [8]. However, none of these architectures can guarantee throughput performance. The latest theoretical result shows that IQ and CIOQ architectures are inferior to OQ architectures in the case of general multicast traffic patterns, contrary to the case of unicast traffic, for which IQ and OQ switches were proved to be equivalent [4], which motivates us to find new switch architectures to provide better performance guarantee for admissible multicast traffic patterns.

Some early researchers propose a kind of two-stage multicast switch, which is a serial combination of a copy network and a point-to-point switch [9]. However, no proper scheduling algorithm has been proposed to guarantee 100% throughput.

In this paper, we propose a crossbar based Two-Stage Multicast Switch (TSMS) which contains a Multicast to Unicast (MTU) switch as the first stage and a Combined Input and Output Queueing (CIOQ) switch as the second stage. Based on this multicast architecture, we prove that speedup of $2 - 2/(N+1)$ is necessary and 2 is sufficient to guarantee 100% throughput under any admissible multicast traffic patterns.

The reminder of paper is organized as follows: Section II describes our TSMS from three aspects including fabrics, queues and schedulers. Section III analyzes the stability of our TSMS switch and presents the proof of 100% throughput guarantee under any admissible multicast traffic patterns. Section IV shows our simulation results about the delay performance of TSMS.

II. OUR MODEL

A. A General Multicast Switch

We study switches with M input ports and N output ports, where all input and output lines run at the same rate. The
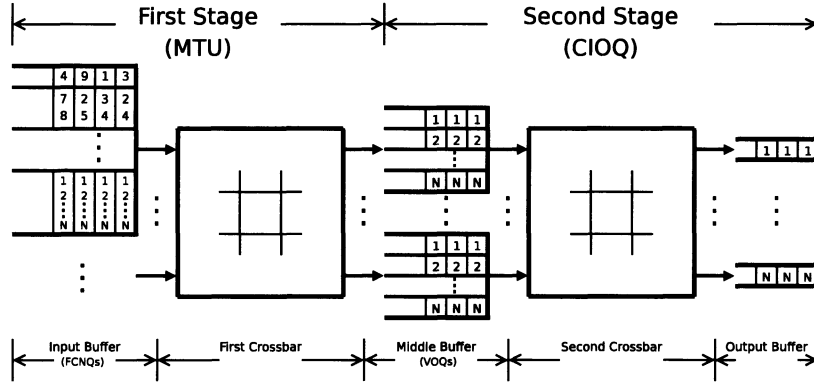
Figure 1. a two-stage multicast switch

switch is cell-based and operates in a slotted and synchronized fashion. A multicast cell is characterized by its fanout set, i.e. by the set of destinations. The *fanout cardinal number (FCN)* of a cell is the size of its fanout set.

There are two different service disciplines for multicast scheduling. One is non-fanout splitting in which all the copies of a cell must be sent in one time-slot. The other is fanout splitting in which multicast cells are permitted to be served partially in one time-slot. Although fanout splitting discipline is work conserving, a cell needs more than one time-slot to be scheduled, which causes an increase on the input load [10]. In order to avoid this kind of performance penalty, we only consider non-fanout splitting discipline in this paper.

A traffic pattern is admissible if neither input port nor output port is over-subscribed on average. We normalize input (output) loads to line rates: a load equal to 1 means a fully utilized input (output) line (one cell per time-slot). Let $\lambda_{i,j}$ be the arrival rate of the unicast flow from input port i to output port j. A unicast traffic pattern is admissible if and only if it satisfies

$$\lambda_i = \sum_{j=1}^{N} \lambda_{i,j} \leq 1, \lambda_j = \sum_{i=1}^{N} \lambda_{i,j} \leq 1, \lambda_{i,j} \geq 0 \qquad (1)$$

Let $\lambda_{i,F}$ be the arrival rate of the multicast flow from input port i to the output ports in F. A multicast traffic pattern is admissible if and only if it satisfies

$$\lambda_i = \sum_{F \subseteq U} \lambda_{i,F} \leq 1, \lambda_j = \sum_{i=1}^{N} \sum_{\substack{j \in F \\ F \subseteq U}} \lambda_{i,F} \leq 1, \lambda_{i,F} \geq 0 \qquad (2)$$

Let L denote the average load of all the Middle Buffers, which equals to the average number of copied cells received by the Middle Buffers in a time-slot. So, from (2), we have

$$L = \sum_{j=1}^{N} \lambda_j = \sum_{i=1}^{N} \sum_{F \subseteq U} |F| \lambda_{i,F} \leq N \qquad (3)$$

which means under admissible multicast traffic pattern, the load of Middle Buffers can not exceed N on average.

B. Two-Stage Multicast Switch

Distinguished from admissible unicast traffic patterns, admissible multicast traffic patterns have two particular characteristics. One is related to spatial characteristic, which means in one time-slot, the arrival cell at single input port is enough to saturate the output ports, (i.e. the cell is a broadcast cell destined for all the output ports). While in unicast case, in order to saturate the switch, all of the input ports are needed to achieve their maximum load. This character is called *spatial unbalance* which implies input load may be concentrated to some input ports. The other is related to temporal characteristic, which means in some time-slots, copies of the arrival multicast cells may exceed the output capacity of the switch (for a $M \times N$ switch, the output capacity is N). While in unicast case, the input traffic in any time-slot can not exceed the output capacity. This character is called *temporal unbalance* which implies output load may be concentrated in some time-slots.

Our *Two-Stage Multicast Switch (TSMS)* is a serial combination of a *Multicast to Unicast (MTU)* switch and an ordinary *Combined Input and Output Queueing (CIOQ)* switch (shown in Figure 1). MTU switch includes Input Buffers used to address the problem of temporal unbalance and a crossbar used to solve the problem of spatial unbalance. In the following parts, we describe TSMS from three aspects: fabrics, queues and schedulers.

1) Fabrics: TSMS contains two crossbar fabrics. The first crossbar is used to copy a multicast cell at Input Buffer to multiple unicast cells at Middle Buffers. Due to the intrinsic multicast capacity of crossbar, the replication can be done in one time-slot. The second crossbar is used to deliver unicast cells from middle buffers to output buffers according to their destinations.

2) Queues: Based on two crossbar fabrics, TSMS contains three stages of buffers: Input Buffers, Middle Buffers and Output Buffers. Input buffers are located before the first crossbar, which are used to store arrival cells temporarily. The buffers are organized as Fanout Cardinal Number Queueing (FCNQ), which means the Buffer is divided into N queues according to the FCN of a multicast cell. For example, as is

shown in Figure 1, the cell with fanout set $\{2,4\}$ is inserted into the second FCNQ because its FCN is two.

Middle Buffers are located between two crossbar fabrics to receive cells from the first crossbar and send cells to the second crossbar. Middle Buffers are organized as VOQs which buffer unicast cells according to their destination output ports.

Output Buffers located after the second crossbar store the cells which can not be sent to outside link immediately. There are two reasons for the existence of Output Buffers. On the one hand, the output buffer is served as re-order buffer to solve the mis-sequence problem caused by multiple paths. On the other hand, the Output Buffer is used to handle the temporary overload caused by speedup.

As is shown in Figure 1, MTU switch which consists of Input Buffers and the first crossbar is used to transform a multicast traffic pattern into a unicast traffic pattern. CIOQ switch which consists of Middle Buffers, the second crossbar and Output Buffers is functioned as an ordinary point-to-point switch.

3) Scheduler: Since the two stages of TSMS have different functions, they use LFCNF-UMBA scheduling algorithm and Maximal Matching scheduling algorithm respectively. The LFCNF-UMBA scheduler implemented on MTU switch is constructed from two parts: one is Largest Fanout Cardinal Number First (LFCNF) policy used to determine which cells in Input Buffer should be copied and a Uniform Middle Buffer Allocation (UMBA) policy to determine which Middle Buffers the cell should be copied to.

Largest Fanout Cardinal Number First policy
Let I_{idle} be the set of idle inputs, and $|O_{idle}|$ be the number of idle outputs.
1: **for** $i = 1$ to N **do**
2:    **for** $in = 1$ to N **do**
3:       **if** $in \in I_{idle}$ and the $(N - i + 1)$th queue at input port in is not empty and $|O_{idle}| \geq (N - i + 1)$ **then**
4:          The cell at the head of $(N - i + 1)$th queue at input port in is selected to be scheduled
5:          $I_{idle} \leftarrow I_{idle} \setminus \{in\}$
6:          $|O_{idle}| \leftarrow |O_{idle}| - k$
7:       **end if**
8:    **end for**
9: **end for**

Figure 2. LFCNF Policy

The scheduler implemented on MTU is responsible for resolving two conflicts. One is input conflict which means that in a time-slot more than one cell at an input port are copied. The other is Middle Buffer overload which means in a time-slot the total number of copied cells exceeds the number of Middle Buffers (i.e. the output capacity of MTU switch).

The spirit of LFCNF policy is to give cells with larger FCN a higher scheduling priority. As is shown in Figure 2, it has N iterations and just considers one FCNQ at each Input Buffer in one iteration. In the ith iteration, the scheduler examines all the FCNQs with FCN of $(N - i + 1)$ at Input Buffers, and selects the cells with FCN of $(N - i + 1)$ as many as possible if they do not make two kinds of conflicts mentioned above.

Uniform Middle Buffer Allocation Policy
Let Allocation Iterator (AI) to indicate the last Middle Buffer to which the latest cell is copied.
When a cell with FCN of k requests Middle Buffers,
1: Copy the cell to the following Middle Buffers: $AI\%N + 1$, $(AI + 1)\%N + 1, ..., (AI + k - 1)\%N + 1$
2: $AI \leftarrow (AI + k - 1)\%N + 1$

Figure 3. UMBA Policy

UMBA policy is described in Figure 3. When a cell with FCN of k request Middle Buffers, UMBA policy allocates k consecutive Middle Buffers which are after the Middle Buffer to which AI points (Here we define that the 1st Middle Buffer is succeeded to the Nth Middle Buffer). The purpose of UMBA policy is to distribute copies of multicast cells to Middle Buffer uniformly. LFCNF policy and UMBA policy run in concert to function as a complete scheduler.
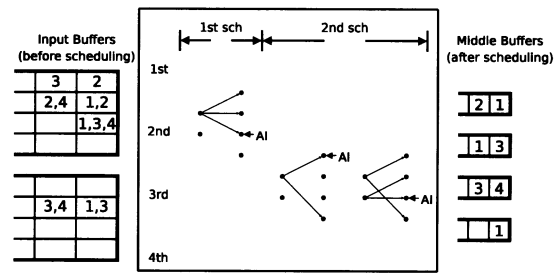

Figure 4. an example of LFCNF-UMBA scheduler

An example of LFCNF-UMBA scheduling on a 2×4 MTU switch is shown in Figure 4. Before LFCNF-UMBA scheduling, the state of Input Buffers is shown in the left of Figure 4 and all the Middle Buffers are empty. There are two consecutive scheduling cycles and no cells arrived during these cycles. The initial value of AI is 0. Each scheduling contains 4 iterations.

- *(1st sch)* According to LFCNF, the 3rd FCNQ at Input Buffer 1 is selected during the 2nd iteration, and according to UMBA, the cell $\{1, 3, 4\}$ is routed to the 1st, 2nd and 3rd Middle Buffer. Then AI is updated to 3.
- *(2nd sch)* According to LFCNF, both the 2th FCNQ at Input Buffer 1 and the 2th FCNQ at Input Buffer 2 are selected during the 3rd iteration. UMBA allocates the 4th, 1st Middle Buffer to cell $\{1, 2\}$ first $(AI \leftarrow 1)$, and then allocates the 2nd, 3rd Middle Buffer to cell $\{1, 3\}$ $(AI \leftarrow 3)$.

After two scheduling cycles, the state of Middle Buffer is shown in the right of Figure 4 (for simplicity, the Middle Buffer is not presented in a VOQ form.).

In CIOQ switch, any Maximal Matching scheduler, such as iSLIP, PIM, can be implemented to guarantee a high throughput.

The whole multicast scheduling process is like this: when a multicast cell arrives at an input port, it is first inserted into the corresponding FCNQ in Input Buffer. And then under LFCNF-

UMBA scheduling, it is copied to Middle Buffer through the first crossbar. After that, CIOQ switch with Maximal Matching scheduler forwards the cell from Middle Buffer to Output Buffer according to its destination. Finally, the cell is delivered to the outside link.

III. STABILITY

In this section, we first analyze the stability of MTU switch, and then prove that speedup of two can make the whole TSMS stable under any admissible multicast traffic patterns.

Theorem 1 (necessity): a $M \times N$ $(M > 1)$ MTU switch with non-fanout splitting scheduling needs a speedup of at least $2 - 2/(N + 1)$ to be stable under any admissible multicast traffic pattern.

Proof: We first describe a specific two-port admissible multicast traffic pattern, and then we prove that no MTU scheduler can make MTU switch stable with speedup of less than $2-2/(N + 1)$ under this specific multicast traffic pattern.

Two Ports (TP) traffic pattern:
Only two input ports are active.

- The first input port receives N unicast cells whose FCN is 1 every $N + 1$ time-slots. The probabilities that cells are destined for each output port are the same.
- The second input port receives N broadcast cells whose FCN is N every $N + 1$ time-slots.

Figure 5. Two-Ports Traffic Pattern

The two-port admissible multicast traffic pattern is described in Figure 5.

TP traffic pattern is admissible. Because each of two input loads is $N/(N + 1)$, which is less than 1, and each output load is $(N + N \cdot N)/(N + 1)$, which equals to 1, i.e. neither input load or output load is over-subscribed.

Because of input conflict, no more than one unicast/broadcast cell can be scheduled in a scheduling cycle. And due to Middle Buffer overload, a unicast cell and a broadcast cell can not be scheduled together in a scheduling cycle. Therefore, only one cell can be scheduled in one scheduling cycle. Consequently, in every $N + 1$ time-slot, MTU needs $2N$ scheduling cycles to schedule these $2N$ cells, i.e. requires speedup of $2N/(N + 1)$. ∎

Theorem 2: [sufficiency] A $M \times N$ MTU switch with speedup of two is stable under any admissible multicast traffic pattern by using LFCNF policy.

Proof: The proof is presented in Appendix A. ∎
The output traffic pattern of MTU switch is not admissible since Middle Buffer may receive two cells in a time-slot. So we use UMBA policy to guarantee that the load of Middle Buffer will not exceed one on average under the admissible multicast traffic pattern.

Lemma 1: The output traffic pattern of MTU switch with UMBA policy is admissible unicast traffic if the input traffic is an admissible multicast traffic pattern.

Proof: Let $\lambda'_{i,j}$ be the unicast flow from Middle Buffer i to Output Buffer j and $\lambda_{i,F}$ be the multicast flow from input port i to the output ports in F. Suppose input traffic $\lambda_{i,F}$ is

admissible and traffic $\lambda'_{i,j}$ is the output traffic of MTU switch. According to (2), we know that

$$\lambda'_j = \sum_{i=1}^{N} \lambda'_{i,j} = \sum_{i=1}^{N} \sum_{\substack{j \in F \\ F \subseteq U}} \lambda_{i,F} \le 1$$

and due to uniformization of UMBA, we have

$$\lambda'_i = \sum_{j=1}^{N} \lambda'_{i,j} = (\sum_{i=1}^{N} \sum_{F \subseteq U} |F|\lambda_{i,F})/N \le 1$$

Therefore, $\lambda'_{i,j}$ is admissible unicast traffic. ∎

Theorem 3: TSMS with speedup of two can be stable under any admissible multicast traffic pattern.

Proof: The proof has two steps.

1) Under any admissible multicast traffic pattern, MTU with speedup of two can be stable (Theorem 2), and export admissible unicast traffic pattern(Lemma 1).

2) According to [2], CIOQ switch with speedup of two can be stable under any admissible unicast traffic pattern by using any Maximal Matching scheduling.

Therefore, the whole TSMS with speedup of two can be stable under any admissible multicast traffic pattern. ∎

IV. SIMULATION

In this section, we simulate a 16×16 TSMS to evaluate its performance of throughput.

A. Traffic scenarios

We consider two kinds of traffic pattern, in both of which cells are generated according to an *i.i.d.* Bernoulli process and are uniformly distributed over all output ports.

- *(Uniform Traffic)* Cells arrive at each input port with the same probability, and the average FCN is $N/2$.
- *(Two Port Traffic)* In a time-slot, a unicast cell (FCN=1) with probability of $\rho N/(N + 1)$ arrives at input port 1, while a broadcast cell (FCN=N) with probability of $\rho N/(N + 1)$ arrives at input port 2. Other input ports are always idle. (ρ is output load.)
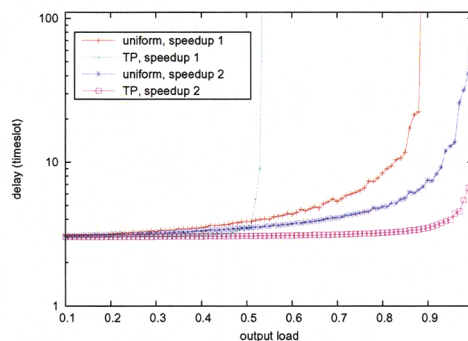
B. Simulation Results

Figure 6. the delay of a 16×16 TSMS with different speedup

We simulate a 16 × 16 with different speedup under two traffic patterns mentioned above.

From Figure 6, we know that with speedup of two, our TSMS can achieve 100% throughput under both Uniform Traffic pattern and Two Port Traffic pattern. With speedup of one, TSMS can achieve almost 90% throughput under uniform traffic pattern, however, it can only achieve less than 54% throughput under Two Port Traffic pattern.

REFERENCES

[1] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch," *Communications, IEEE Transactions on*, vol. 47, pp. 1260–1267, Aug. 1999.

[2] J. Dai and B. Prabhakar, "The throughput of data switches with and without speedup," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, 2000, pp. 556–564 vol.2.

[3] B. Prabhakar, N. McKeown, and R. Ahuja, "Multicast scheduling for input-queued switches," *Selected Areas in Communications, IEEE Journal on*, vol. 15, pp. 855–866, Jun. 1997.

[4] M. Marsan, A. Bianco, P. Giaccone, E. Leonardi, and F. Neri, "Multicast traffic in input-queued switches: optimal scheduling and maximum throughput," *Networking, IEEE/ACM Transactions on*, vol. 11, pp. 465–477, 2003.

[5] M. Andrews, S. Khanna, and K. Kumaran, "Integrated scheduling of unicast and multicast traffic in an input-queued switch," in *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, 1999, pp. 1144–1151 vol.3.

[6] S. Gupta and A. Aziz, "Multicast scheduling for switches with multiple input-queues," *High Performance Interconnects, 2002. Proceedings. 10th Symposium on*, pp. 28–33, 2002.

[7] A. Bianco, P. Giaccone, E. Leonardi, F. Neri, and C. Piglione, "On the number of input queues to efficiently support multicast traffic in input queued switches," in *High Performance Switching and Routing, 2003, HPSR. Workshop on*, 2003, pp. 111–116.

[8] A. Bianco, P. Giaccone, C. Piglione, and S. Sessa, "Practical algorithms for multicast support in input queued switches," *High Performance Switching and Routing, 2006 Workshop on*, pp. 6 pp.–, Jun. 2006.

[9] T. T. Lee, "Nonblocking copy networks for multicast packet switching," *Selected Areas in Communications, IEEE Journal on*, vol. 6, pp. 1455–1467, Dec. 1988.

[10] C. K. Kim, T. T. Lee, and M. Bellcore, "Performance of call splitting algorithms for multicast traffic," in *INFOCOM'90. Ninth Annual Joint Conference of the IEEE Computer and Communication Societies.'The Multiple Facets of Integration'. Proceedings., IEEE*, 1990, pp. 348–356.

APPENDIX
PROOF OF THEOREM 2

Before proving the Theorem 2, we first define the stability of the system, which implies 100% throughput of the switch.

Definition 1 (stability): a $M \times N$ MTU switch is stable if any cell in its queues can be scheduled in a finite time.

Proof of Theorem 2: Suppose a cell a with fanout set F_a ($|F_a| = k$) arrives at input port i at time-slot t_{start}. Let I_i be the set of the cells at input port i, and M be the set of cells with FCN of no less than k at input port i, i.e. $M = \{c||F_c| \geq k, c \in I_i\}$. Due to speedup of two, there are two scheduling cycles in a time-slot. Let $L_t(L_s)$ be the average number of copied cells received by the Middle Buffers in a time-slot (scheduling cycle).

We suppose Cell a will not be scheduled in a finite time. Correspondly, we have $M \neq \emptyset$ since $a \in M$.

1) If $k \geq \lceil (N+1)/2 \rceil$

Under LFCNF policy, the cell with Largest FCN will be scheduled first. Since Cell a is not scheduled, there must be a cell with FCN of no less than k scheduled in each of two scheduling cycles ($L_s \geq k$). So, we have $L_t = 2L_s \geq 2k \geq N + 1 > N$.

2) If $k < \lceil (N+1)/2 \rceil$

In every time-slot, the following three cases may happen.

   a) $|M|$ increases by 1 (time-slot A)

     In this case, a cell c ($|F_c| \geq k$) arrives at input port i, no cell in M is scheduled. In each scheduling cycle, the number of copied cells is no less than $N - k + 1$, otherwise according to LFCNF policy, the cell with FCN of k at input port i will be scheduled together. Hence, we have $L_t = 2L_s \geq 2(N - k + 1)$.

   b) $|M|$ decreases by 1 or 2 (time-slot B)

     Since Cell a is not scheduled, according to LFCNF, there must be a cell with FCN of no less than k scheduled in each of two scheduling cycles ($L_s \geq k$). Therefore, we have $L_t = 2L_s \geq 2k$.

   c) $|M|$ stays the same (time-slot C)

     i) A cell c ($|F_c| \geq k$) arrives at input port i, and at the same time a cell in M is scheduled.
In one scheduling cycle, we have $L_{s1} \geq k$ since a cell in M is scheduled. In the other scheduling cycle, we have $L_{s2} \geq N - k + 1$ since no cell with FCN of k at input port i is scheduled. So, we have $L_t = L_{s1} + L_{s2} \geq N + 1$.

     ii) No cell c ($|F_c| \geq k$) arrives at input port i, and no cell in M is scheduled.
we have $L_s \geq N - k + 1$ since no cell with FCN of k at input port i is scheduled. Due to $k < \lceil (N+1)/2 \rceil$, we have $L_t = 2L_s \geq 2(N - k + 1) > N + 1$.

Let $M_{t_{start}}$ be the state of M at time-slot t_{start}. Suppose T time-slots has passed from time-slot t_{start}, which contains t_A time-slot A, t_B time-slot B and t_C time-slot C. Since $M \neq \emptyset$, we have $t_A + |M_{t_{start}}| > t_B$. From a) b) c) discussed above, we know that the average load of Middle Buffer is

$$L_t \geq \frac{t_A \cdot 2(N - k + 1) + t_B \cdot 2k + t_C \cdot (N + 1)}{t_A + t_B + t_C}$$
$$> (N + 1) - \frac{|M_{t_{start}}|(N + 1 - 2k)}{T}$$

So, we know $L_t > N$ when $T \geq |M_{t_{start}}|(N + 1 - 2k)$.

From 1) 2), we know if Cell a will not be scheduled in a finite time, the load of Middle Buffer will exceeds N, which is conflicted with the supposition that the multicast traffic pattern is admissible (From (3), we know that the Middle Buffer load of admissible multicast traffic pattern can not exceed N.)

■