Design and Control of Next Generation Distribution Frames

Davide Cuda, Paolo Giaccone, Massimo Montalto
Dipartimento di Elettronica, Politecnico di Torino, Italy

***Abstract*—Today, the permutation of circuits in the Main Distribution Frames (MDF), which connect the subscriber lines to POTS and to DSLAMs, is still operated manually. However, new market regulations, allowing subscribers to change network operator frequently, and the new schemes to concentrate active ADSL users into few DSLAMs during off-peak hours, adopted by network operators to reduce the energy consumption in the access network (and the related operational costs), require more advanced, reliable and faster mechanisms than human operations. Indeed, Automated MDFs (AMDF) have recently become available on the market to provide cheap and almost real-time circuit switching.**

Even if grounded on more than 50 years of research activities on architectures for circuit switching, the considered scenario is quite peculiar and offers new interesting technical challenges, since classical multistage strictly non-blocking networks are too expensive for the number of required ports (sometimes very large, exceeding 100,000), and rearrangeable multistage networks can interrupt temporarily active circuits, affecting in a indefinite way the performance of ADSL lines. For these reasons, we propose the design of AMDFs based on recently proposed non-interruptive rearrangeable (NIR) networks and show how to optimize the routing control to minimize the setup time of a circuit. Finally, our findings are relevant both for the theory of multistage interconnection networks, and for those companies producing, engineering and operating large AMDFs.

***Index Terms*—Multistage switching architectures, non-interruptive networks.**

I. Introduction

In the access segment of a telecommunication network, as shown in Fig. 1, the Main Distribution Frame (MDF), placed at the Central Office (CO), provides many carriers with the possibility to reach the subscriber lines and connect them to DSLAMs and to POTS services. The MDF is basically a circuit switch, which has been manually operated so far because requests of setting up or tearing down connections were mostly occasional. However, the new telecommunication market regulations and current needs to reduce the energy consumption require now to move from slow human operations to almost real-time switching operations.

On the one hand, over the last 15-20 years, because of the new telecommunication market regulations in US and EU countries, competition has been rising tough in the context of the access network. The variety of ISPs coexisting in a CO and the consequent offer diversification have increased access network complexity. Migration of customers among carriers (churn) continues to grow at high rates due to market rivalry and networks evolution. According to [1] and [2], carriers experience on average 30-35% of yearly churn rate (i.e., percentage of customers changing their carrier for each year). As a result, operating the access network, heavily influenced by truck rolls, is today an expensive, slow and fault-prone process.

On the other hand, access networks represent the main culprit of the overall energy consumption of the Internet [3], [4]. Indeed, since all the ADSL subscribers are connected uniformly to the linecards of a DSLAM and to many DSLAMs, all these linecards/DSLAMs are always powered on, independently from the activity of the ADSL users. Nowadays some large European ISPs [5] are devising and evaluating smart policies to concentrate active users during off-peak hours into few linecards/DSLAMs and to power off all the other ADSL interfaces. Note that this process requires to aggregate all the inactive (or very low load) ADSL subscribers in few ports and implement some (in-band or out-band) wake-on-ADSL mechanisms, still proprietary and not included in the ANSI/ITU ADSL standards. Thanks to the large variability in the traffic load [6], powering off the linecards/DSLAM ensures a relevant power saving, when operated daily and based on the activity of each subscriber line.

A. From traditional to next-generation MDFs

From the switching point of view, the main requirements to design a traditional MDF have been the following: (i) non-blocking behavior, (ii) full connectivity between the input and output ports, (iii) large number of ports (1,000-10,000). These requirements have been met mainly by skillfully engineering the cabling system and the patch panels positions.

Current AMDFs support a very large number of input/output ports; 10,000-100,000 ports are already available in commercial products [7]–[9]. Two main approaches are being adopted for their design. The first approach consists of adopting a robot that performs the connections directly on the patch panel [10].

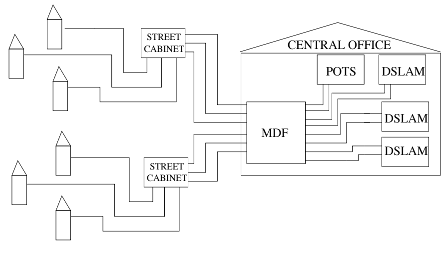

Fig. 1. Basic scheme of the current access network

The second approach adopts a multistage circuit switching network and different technologies to implement each basic switching module. Commercial AMDFs employ electro-mechanical relays [7] or MEMS (Micro Electro-Mechanical Systems) [11], [12] as crosspoints (i.e., basic switching devices), whereas new promising technologies appear to be NEMS (Nano Electro Mechanical Systems) [13] and TEMS (Transparent Embedded Magnetic Switch) [14]. However, independently form the technology adopted, the switching architectures are based on classical rearrangeable Clos networks [15], since they allow good scalability in terms of cost, measured as the number of crosspoints (or basic modules), when the size of the switch increases.

Due to the larger number and higher frequency of reconfigurations, next-generation AMDFs will support a switching activity that must be transparent for pre-existing circuits, i.e. new requests must "not-interrupt" any other circuit that is already set up. In particular, the *not-interruptive* (NIR) property is crucial for ADSL subscriber lines, since active ADSL lines are not robust to interruptions in the electrical continuity of the circuit. Indeed, in the case of an interruption, the ADSL connection enters a retrain phase to adapt to the dynamic channel conditions [16]; unfortunately, this process takes between 1 and 10 seconds, which is not acceptable for frequent (daily) reconfigurations. Thus, the NIR property is a make-or-break requirement, and actual AMDF designs do not satisfy it when based on classical *rearrangeable non-blocking* (RNB) networks, in which pre-existing circuits can be temporary interrupted when adding a new circuit.Alternative architectures, based on *strictly non-blocking* (SNB) networks, satisfy the NIR requirement by construction, but their cost is almost twice the one of RNB networks and are used only in small size switches. Recently, the seminal work in [17] has shown how to design NIR networks with a cost very close to classical RNB networks, but such design is not optimized to cope with the particular scenarios considered in our work. Furthermore, some cheap electro-mechanical technologies adopted to implement each switching module (e.g. in the case of relays) require a non-negligible amount of time (seconds or even minutes) to change their internal state and reconfigure. In this case, for large networks, reconfiguration delays can grow to minutes/hours and become unacceptable for practical deployment.

In this paper we show how to optimize jointly the design and the routing control of multistage switching networks, expressively designed for AMDFs. We have considered only NIR architectures, without considering RNB networks, which cannot cope with frequent reconfigurations, and SNB networks, whose cost is too high, being almost twice than NIR networks.

The main contributions of our paper are (i) to propose the corresponding routing algorithm to connect new circuits and (ii) to compare numerically different design architectures. All these results may strongly help to reduce costs of future AMDFs.

The paper is organized as follows. In Sec. II we discuss the design of NIR networks and in Sec. III we propose the corresponding routing algorithms. Finally, in Sec. IV we show some numerical example of design.

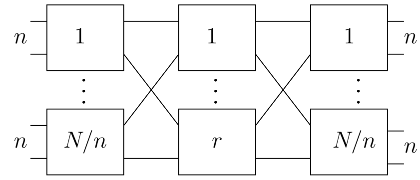

Fig. 2. Three stage Clos network

TABLE I
SYNOPTIC VIEW OF CANDIDATE NETWORKS-(N,n) FOR ADMF

Network archit.	Routing algorithm	C_X	NIR
SNB	trivial	$4nN - 2N + 2N^2/n - N^2/n^2$	yes
RNB	Paull	$2nN + N^2/n$	no
NIR1	Conf-NIR1	$2nN + 2N + N^2/n + N^2/n^2$	yes
NIR2	Conf-NIR2	$2nN + N^2/n + 4N + 2N^2/n^2$	yes

II. MULTI-STAGE SWITCHING NETWORKS

We consider the design of a $N \times N$ circuit switch, designed to be fully-connected and non-blocking, hence it is always possible to connect any idle input to any idle output. We evaluate the cost of the switch in terms of crosspoints (C_X). A basic module of size $n \times m$ built with a crossbar has cost $C_X = nm$. When a new circuit between an input to an output must be established, the switching architecture can be [15] either *strictly non-blocking* (SNB), i.e. when the path to connect the circuit is established without rerouting pre-existing active paths of former circuits, or *rearrangeable non-blocking* (RNB), i.e. when the path may require to rearrange pre-existing paths.

We consider a symmetric three-stage Clos network [15], depicted in Fig. 2, built with N/n modules of size $n \times r$ at the I stage, r modules of size $(N/n) \times (N/n)$ at the II stage and N/n modules of size $r \times n$ at the III stage. Depending on r, the Clos network shows different properties. The condition $r = 2n - 1$ guarantees SNB, with a trivial routing algorithm. Moreover, the condition $r = n$ guarantees RNB behavior, when the *Paull algorithm* (described in [15]) is adopted. The cost of RNB and SNB are reported in Table I. Asymptotically, for $N \to \infty$, it is well known that the cost of SNB networks is almost twice the cost of RNB: $C_X(\text{SNB}) \approx 2C_X(\text{RNB})$.

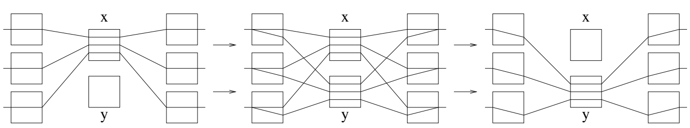

Fig. 3. Example of operations to rearrange the circuits through x without any interruption by exploiting duplicated paths through y

Fig. 4. Example of output and input divertability for a 4×4 basic module

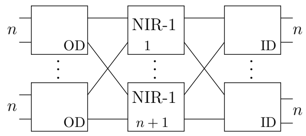

Fig. 5. NIR1 network obtained with recursive construction.

A. NIR rearrangeable Clos networks

Quite recently, [17] has defined the class of *non-interruptive rearrangeable* (NIR) networks, that are rearrangeable but, whenever adding a new circuit, pre-existing circuits do not experience any electrical interruption. The basic idea is to create duplicate paths through additional (with respect to RNB Clos networks) II-stage modules, following the sequence of operations shown in Fig. 3. Interestingly, the condition $r = n + 2$ is enough to build NIR Clos networks; thus, only two additional modules in the II stage (independently from the switch size) are sufficient to observe an end-to-end behavior similar to SNB. The NIR property is achieved thanks to duplicated paths through the two additional II-stage modules, in order to "backup" all the paths that must be rearranged and avoid interruptions. We will denote as *NIR2* such class of NIR networks. The corresponding cost, computed in Table I, grows asymptotically as an RNB: $C_X(\text{NIR2-}(N, n)) \approx C_X(\text{RNB-}(N, n))$; the notation "network-$(N, n)$" refers to a $N \times N$ network with (N/n) modules at the I and III stage. In addition, [17] has shown that it is possible to further reduce the number of II-stage modules by setting $r = n + 1$, i.e. with just one additional middle stage module, and adopting a more complex routing algorithm. This architecture will be later referred as *NIR1* and Table I shows that $C_X(\text{NIR1-}(N, n)) \approx C_X(\text{RNB-}(N, n))$.

To support the NIR construction, it is necessary to create double paths between the I-stage modules and the III-stage-modules. Such modules must support the property of *output* and *input divertability*, respectively. A switching network is said to be output-divertable (input-divertable) if it is always possible to non-interruptively append a new path from the input (output) of a given existing path to some unused output (input) [17], as shown in Fig. 4. Note that a single crossbar switch of size $n \times m$, provides, thanks to the nm crosspoints, both input and output divertability.

B. Recursive construction of large switches

To design networks with a very large number of ports, as for the AMDF considered in the CO scenario, it is possible to adopt a recursive construction, through which each II-stage module is implemented through another multistage network. This allows a modular and scalable design.

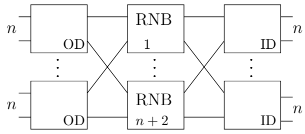

Fig. 6. NIR2 network obtained with recursive construction

Fig. 5 an 6 show the basic recursive rule to design NIR1 and NIR2 networks [17], respectively. "ID" stands for input-divertable network, and "OD" stands for output-divertable network. In NIR1 networks, the II-stage modules must be NIR1 to guarantee NIR behavior. On the contrary, in NIR2 networks, the II-stage modules are simple RNB networks. Note that the routing algoritms must be carefully chosen to ensure NIR behavior, as discussed in Sec. III.

III. Routing in NIR1 and NIR2 networks

We propose two algorithms, *Conf-NIR1* and *Conf-NIR2*, to route the circuits in NIR1 and NIR2 networks, respectively. They both (i) minimize the reconfiguration time, by exploiting parallel operations, (ii) avoid interruptions for pre-existing paths, by exploiting duplicated paths. These algorithms are based on the algorithms proposed in [17] and work in incremental way, by adding one circuit at the time.

We will use the Paull matrix [15] to describe the operations of the algorithm. Let Ω_i be the set of modules of the i-stage and $|\Omega_i|$ be the corresponding number of modules. The Paull matrix describes in a compact way the active paths in a three stages Clos network. It is a matrix $M = [m_{ij}]$ of size $|\Omega_1| \times |\Omega_3|$, in which each row corresponds to a I-stage module and each column corresponds to a III-stage module; each element $m_{ij} \subseteq \Omega_2$ is the set all the II-stage modules used to connect $i \in \Omega_1$ to $j \in \Omega_3$. For example, if the II-stage module $a \in m_{ij}$, then it exists a circuit whose path connects I-stage module i to III-stage module j through a, i.e. the ith input of a is connected to its jth output.

A. Circuit routing in a NIR1 network

Pseudocode of the operations to add a circuit between a free input of I-stage module m_1 and a free output of III-stage module m_3 is provided in Fig. 7. Note that notation $x \rightarrow y$, being x and y modules in subsequent stages, refers to the presence of an active path between the two modules; notation $x \nrightarrow y$ means the opposite. When adding a new circuit, the algorithm checks if it is needed to rearrange, exactly as in the Paull algorithm. If it is not necessary, the circuit is added by creating a path along I-stage m_1, II-stage a and III-stage m_3; no interruption occurs for pre-existing circuits. Otherwise, it is necessary to rearrange the paths that are currently passing through *exactly* 2 specific II-stage modules, namely a and b, independently from N [15].

Fig. 8 shows an example of the steps in the algorithm, described through the temporal evolution of the Paull matrix, in the case a new circuit must be added between the I-stage module and III-stage module corresponding to the position

*** **Conf-NIR1** ***

function **add-new-connection**($m_1 \in \Omega_1$, $m_3 \in \Omega_3$)
let $c \in \Omega_2$ be the backup module
if exists ($a \in \Omega_2$ s.t. $m_1 \nrightarrow a \nrightarrow m_3$) // no need for rearrangement
  connect $m_1 \rightarrow a \rightarrow m_3$
else it exists ($a, b \in \Omega_2$ s.t. $m_1 \nrightarrow a$ & $m_1 \rightarrow b$ & $a \rightarrow m_3$ & $b \nrightarrow m_3$) // need to rearrange
  P =find-path(a, b, m_1, m_3) $[s_0]$
  duplicate($a \in P \Rightarrow c$) $[s_1]$
  attach(c) $[s_2]$
  detach($a \in P$) $[s_3]$
  duplicate($b \in P \Rightarrow a$), connect $m_1 \rightarrow a \rightarrow m_3$ $[s_4]$
  attach(a) $[s_5]$
  detach($b \in P$) $[s_6]$
  duplicate($c \in P \Rightarrow b$) $[s_7]$
  attach(b) $[s_8]$
  detach(c) $[s_9]$
end if

Fig. 7. Pseudocode to add a circuit from I-stage m_1 to III-stage m_3 in a NIR1 switch

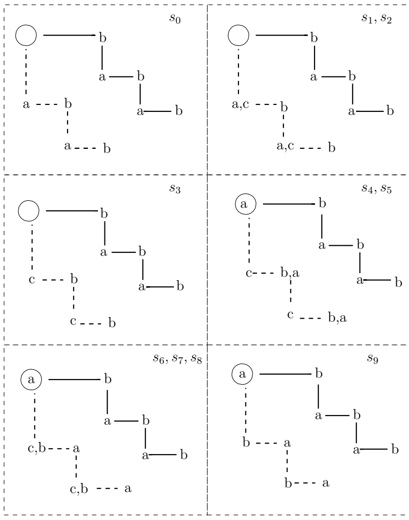

Fig. 8. Example of evolution of the Paull matrix in Conf-NIR1 when rearranging the paths.

highlighted by the circle. During step s_0, the ab-path is computed, for completeness Fig. 8 reports both the ab-path P (dotted) and the ba-path. As reminder, the "ab-path" is defined in the classical Paull algorithm and it is basically an alternating sequence of active paths passing through modules a and b; the details can be found in [15].

The *key idea* of our algorithm is to invert the position of a and b along the ab-path (i.e., to rearrange the corresponding circuits), by exploiting a "backup" module c; at the end, a is used to satisfy the new circuit and c is set free to be available later. In details, during s_1, all the paths passing through a and belonging to P are replicated into c; this requires the divertability of the I-stage and III-stage modules, since the paths must be duplicated to prevent interruption. After its configuration, in s_2 module c is connected to the I stage and to the III stage, and then all the paths through a belonging to P are removed, by configuring properly m_1 and m_3. During s_4, the paths crossing b and present in P are duplicated through a; by construction, this operation is always permitted. Concurrently, the new circuit between m_1 and m_3 is made with the path through a, now available. After, module a is connected to the network. In s_6, the paths through b in P are removed and then b is used to duplicate the paths through c. Finally, in s_8 module b is again connected to the network and, during the last step c is isolated to make it available for the future.

We now evaluate the configuration time of the algorithm. Let δ be the maximum time needed to configure each single module; we assume that many paths can be established/removed in parallel in the same module, i.e., connecting $x_1 \rightarrow y_1$ at the same time of $x_2 \rightarrow y_2$ takes at most δ. Notice that each step $s_1 - s_9$ in Fig. 7 lasts no more than δ and, since all of them must be operated *sequentially* to prevent interruptions, we can claim:

Property 1: The overall configuration time for a NIR1-(N, n) network is $T_R \leq 9\delta$ for each new circuit.

In the case of recursive construction, notice that the *connect* operation in s_4 requires to apply *connect* on a smaller NIR1 network. This increases the configuration time by 9δ at each level of recursion. When $N = n^k$, it holds:

Property 2: The overall configuration time for a NIR1-(n^k, n) network, built with $k-1$ recursive levels of Clos construction, is

$$T_R \leq 9\delta(k-1) = 9\delta\left(\frac{\log N}{\log n} - 1\right)$$

for each new circuit.
As a conclusion, the configuration time grows logarithmically with the size of the switch.

B. Circuit routing in a NIR2 network

Fig. 9 shows the pseudocode to route a new circuit in a NIR2 network and Fig. 10 shows an example. The *key idea* of Conf-NIR2 is simply to use two backup modules c and d in the II stage to duplicate the pre-existing paths across a and b that must be reconfigured. Now a and b can be detached, reconfigured in isolation and then reconnected again to the network. At the end, c and d are set free. Coming back to the pseudocode, s_0, s_4 are exactly the same as in the classical Paull algorithm [15]. Note that paths are duplicated exploiting the divertability of the I and III stages during s_1 and s_4. Since each step, except for s_0, lasts at most δ:

Property 3: The overall configuration time for a NIR2-(N, n) network is $T_R \leq 6\delta$, for each new circuit.

In the case of recursive construction, it is crucial to note that in s_2, s_3, s_5, s_6 only I-stage and III-stage modules are reconfigured. Instead, s_1 and s_4 require a recursive call to the routing algorithm, but all the required operations occur in parallel. Indeed, in s_1 the reconfiguration of *all internal*

*** **Conf-NIR2** ***

function **add-new-connection**($m_1 \in \Omega_1$, $m_3 \in \Omega_3$)
let $c, d \in \Omega_2$ be the backup modules
if exists ($a \in \Omega_2$ s.t. $m_1 \nrightarrow a \nrightarrow m_3$) // no need for rearrangement
  connect $m_1 \rightarrow a \rightarrow m_3$
else it exists ($a, b \in \Omega_2$ s.t. $m_1 \nrightarrow a$ & $m_1 \rightarrow b$ & $a \rightarrow m_3$ & $b \nrightarrow m_3$) // need to rearrange
  P =find-path(a, b, m_1, m_3) $[s_0]$
  duplicate($a \Rightarrow c$), duplicate($b \Rightarrow d$) $[s_1]$
  attach(c), attach(d) $[s_2]$
  detach(a), detach(b) $[s_3]$
  duplicate($b \in P \Rightarrow a$), duplicate($a \in P \Rightarrow b$) $[s_4]$
  connect $m_1 \rightarrow a \rightarrow m_3$ $[s_4]$
  attach(a), attach(b) $[s_5]$
  detach(c), detach(d) $[s_6]$
end if

Fig. 9. Pseudocode to add a circuit from I-stage m_1 to III-stage m_3 in a NIR2 switch

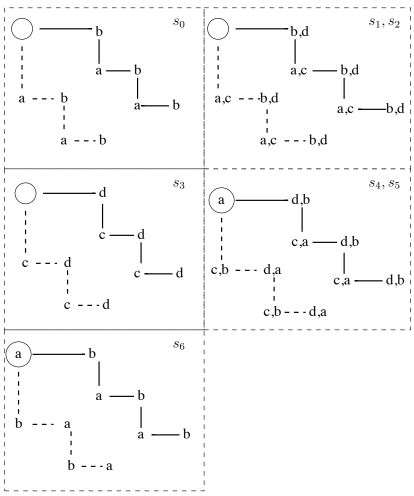

Fig. 10. Example of evolution of the Paull matrix in Conf-NIR2 when rearranging the paths.

modules of c and d can occur in parallel, since c and d are isolated from the network. Similarly, for a and b during s_4. Hence, the reconfiguration time of the II-stage modules is "paid" only once and we can claim:

Property 4: The overall configuration time for a NIR2-(n^k, n) network is $T_R \leq 6\delta$, for each new circuit.
As a conclusion, the configuration time is independent from the size of the switch and from the levels of recursion to construct the network. This property shows that NIR2 networks are superior to NIR1 in terms of reconfiguration time. This is a crucial issue that must be considered when δ is not negligible.

IV. Examples of designs for NIR networks

We assume that the overall switch is built through a single basic physical module (denoted as PHY-M), since it is realistic to assume that a real switch is built with the same building blocks, fully engineered and optimized. We assume that such PHY-M is able to support: $k \times k$ modules with output/input divertability, $k \times (k+1)$ and $(k+1) \times k$ modules with output/input divertability (needed for NIR1 networks), and $k \times (k+2)$ and $(k+2) \times k$ modules with output/input divertability (needed for NIR2 networks). Finally, we evaluate the design using PHY-Ms with k between 16 and 128, which are realistic values for the new technologies adopted in our scenario [11]. We can define two topologies:

- logical topology, describing the interconnection network in terms of the logical modules (denoted as LOG-M), obtained by the (recursive) construction discussed in Sec. II.
- physical topology, mapping each LOG-M to one or more PHY-Ms, according to standard packing techniques, optimized to minimize the number of needed PHY-Ms. For example, a PHY-M can implement many (smaller) LOG-Ms by reserving different group of ports to each LOG-M. As second example, a larger LOG-M can be implemented by interconnecting many PHY-M according to a Clos or crossbar topology.

In the following results, the mapping between logical topology and physical topology has been optimized to minimize the total number of PHY-Ms, hence the final cost.

We present a numerical analysis of the design of NIR1 and NIR2 networks. In particular, we investigate how different design methodologies impact on the final cost and we identify the most convenient conditions to exploit NIR2 rather than NIR1 architectures. We consider $N \times N$ networks, with $N \in [k^2, 2 \cdot 10^5]$, with k being the size of PHY-M.

Fig. 11 shows the cost of NIR2 networks, measured in terms of number of PHY-Ms (C_M), for different values of k and N. All the points present in the graph have not been interpolated and correspond to actual values of N. As expected, a larger k implies lower cost in terms of PHY-Ms. However, the difference between different curves is variable, due to the packing strategy to map the logical topology into the physical one; indeed, the PHY-Ms can have some unused crosspoints, and this level of "waste" is related to k and N. Hence, the optimal choice of k depends on the actual cost (in monetary terms) of each PHY-M and the specific values of N. Similar results have been observed for NIR1 networks, as shown in Fig. 12.

A comparative analysis between NIR1 and NIR2 networks is reported through Fig. 13, which shows the ratio between the corresponding costs in PHY-Ms. Also in this figure, the points have not been interpolated. Both architectures present similar costs, varying at most by 10%; this behavior is quite erratic and depends on the specific values of N and k. Interestingly, for $N \geq 20,000$, NIR2 networks show either lower costs or higher cost no more than 1%, with respect to NIR1. This means that all the benefits of NIR2 networks (lower configuration time) are achieved also for the network architecture with (almost) minimum cost.

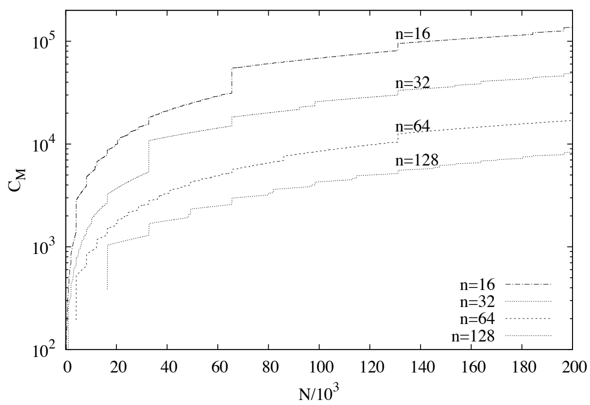

Fig. 11. Number of PHY-Ms needed for NIR2-(N, k) networks

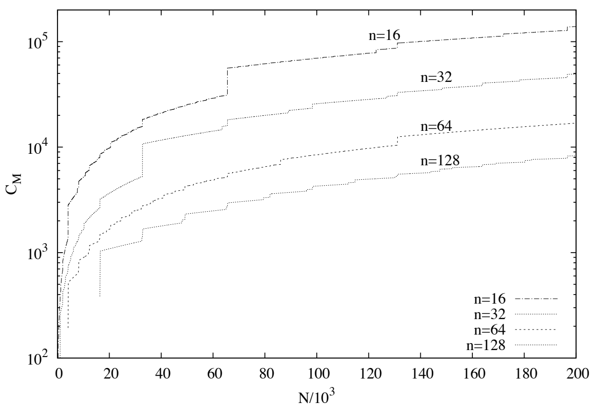

Fig. 12. Number of PHY-Ms needed for NIR1-(N, k) networks

V. Conclusions

We have addressed the design of non-interruptive rearrangeable (NIR) networks in the specific scenario of next-generation large AMDFs present in the access network. We have considered two multi-stage architectures, denoted as NIR1 and NIR2 networks. Regarding NIR1 networks, we have proposed a routing algorithm, denoted as Conf-NIR1, able to minimize the configuration time. Unfortunately, we have shown that the configuration time grows logarithmically with the size of the switch. On the contrary, we have shown that NIR2 networks, if controlled by our routing algorithm Conf-NIR2, show a constant configuration time independently from the switch size. Finally, comparing total costs of NIR1 and NIR2 networks, NIR2 are better than NIR1 in most of the scenarios. As a conclusion of our investigations, we believe that NIR2 networks will be the reference architectures for the next generation AMDFs, outperforming classical rearrangeable networks at almost no-extra cost.

References

[1] C. Borna, "Combating customer churn - product information," *The CBS Interactive Business Network*, Mar. 2010.

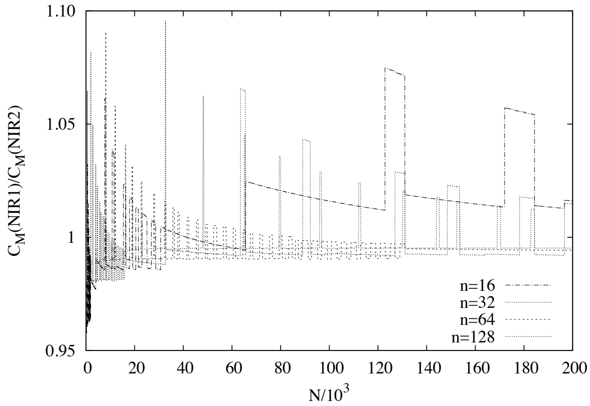

Fig. 13. Ratio between the cost of NIR1 and NIR2 networks

[2] J. Lu, "Predicting customer churn in the telecommunications industry - an application of survival analysis modeling using SAS," in *SAS Conference Proceedings: SAS User Group International 27*, Apr. 2002.

[3] C. Lange, D. Kosiankowski, C. Gerlach, J.-F. Westphal, and A. Gladisch, "Energy consumption of telecommunication networks," in *European Conference on Optical Communication, ECOC '09*, Sept. 2009.

[4] R. Tucker, R. Parthiban, J. Baliga, K. Hinton, R. Ayre, and W. Sorin, "Evolution of WDM optical IP networks: A cost and energy perspective," *Journal of Lightwave Technology*, vol. 27, no. 3, pp. 243–252, 2009.

[5] E. Goma, M. Canini, A. L. Toledo, N. Laoutaris, D. Kostic, P. Rodriguez, and R. Stanojevic, "Insomnia in the access or how to curb access network related energy consumption," in *ACM SIGCOMM'11*, 2011.

[6] A. Finamore, M. Mellia, M. Meo, M. Munafò, and D. Rossi, "Live traffic monitoring with tstat: Capabilities and experiences," in *8th International Conference on Wired/Wireless Internet Communication, WWIC 2010*, 2010, pp. 290–301.

[7] "Automating access to local loop and network facilities," Technical White Paper, TWP-0045 Rev 1.0.

[8] S. Panattoni, "Telecom Italia view and perspectives," C5 World Forum 2007 - Networks and Technologies. Automated Provisioning of New Services: Automated Distribution Frames, Gruppo Telecom Italia, Tech. Rep., 2007.

[9] A. Nyquist, "ADF in next generation access solutions," White Paper, Network Automation, Aug. 2009.

[10] F. Kaufhold, "Robotic distribution frame," Product Line Overview, issue 1.3, UTEL Labs.

[11] S. Braun, J. Oberhammer, and G. Stemme, "MEMS single-chip 5x5 and 20x20 double-switch arrays for telecommunication networks," in *IEEE 20th International Conference on Micro Electro Mechanical Systems*. IEEE 20th International Conference on Micro Electro Mechanical Systems, Jan. 2007, pp. 811–814.

[12] C. Ebeling, F. Reblewski, O. Lepape, and J. Barbier, "Crossbar device constructed with MEMS switches," U.S. Patent US2010/0108479, May 2010.

[13] D. Czaplewski, G. Patrizi, G. Kraus, J. Wendt, C. Nordquist, S. Wolfley, M. Baker, and M. de Boer, "A nanomechanical switch for integration with CMOS logic," *Journal of Micromechanics and Microengineering*, vol. 19, no. 8, p. 085003, July 2009.

[14] Micro magnetic latching switches for automated cross-connect systems. Telepath Networks. [Online]. Available: http://www.telepathnetworks.com/s.nl/sc.5/category.36/.f

[15] J. Hui, *Switching and traffic theory for integrated broadband networks*. Kluwer Acad. Publ., 1990.

[16] A. Johansson, "ADSL Lite - the broadband enabler for the mass market," *Ericsson Review*, no. 4, 1998.

[17] F. Hwang, W.-D. Lin, and V. Lioubimov, "On noninterruptive rearrangeable networks," *IEEE/ACM Transactions on Networking*, vol. 14, no. 5, pp. 1141–1149, Oct. 2006.