Modeling the Always Best Packet Switching Mechanism

Stefano Ferretti, Vittorio Ghini, Moreno Marzolla, Fabio Panzieri
Department of Computer Science, University of Bologna
Mura A. Zamboni 7, I-40127 Bologna, Italy
{sferrett, ghini, marzolla, panzieri}@cs.unibo.it

***Abstract*—In this work we investigate the performance of the Always Best Packet Switching (ABPS) operation approach. ABPS is a cross-layer wireless communication scheme that enables mobile terminals to exploit its multiple Network Interface Cards (NICs) concurrently. The ABPS software module installed on the mobile terminal determines dynamically the most appropriate NIC to use as single point of access to the Internet, and can switch to another NIC dynamically and transparently to the applications. For the assessment, we propose a performance model based on Markov reward models. Through it, we estimate the availability, reliability and throughput provided by ABPS, a standard SIP-based approach and conventional communication protocol employing a single NIC. Results show the performance improvements introduced by ABPS.**

I. Introduction

Current mobile communication terminals such as smartphones, tablets, laptops have several Network Interface Cards (NICs) installed, which allow the user to employ the most suitable one when he/she needs to connect to the Internet, make a VoIP call, retrieve (maybe via streaming) some audio/video flow, etc. Two main questions usually arise in this context: i) which NIC is the "best" one to choose at a given time, and ii) what to do when the NIC in use is no longer the best one (or even worse, when the connection related to that NIC is no longer available and some handoff occurs) [1]–[3]?

The Always Best Packet Switching (ABPS) operation mode solves these problems [4]. It is able to determine the most appropriate NIC to use at any instant. The system employs a distributed software architecture, whose main module is a software component, termed "Client Proxy", running in the Mobile Node (MN). Briefly, ABPS works as follows. A monitor runs in background on the MN; it is in charge of identifying novel possible connections and to configure NICs in background. Hence, the Client Proxy determines the best NIC to use, based on those currently active. Actually, all the available NICs can be exploited concurrently, and it is possible to switch from a network to another on a per packet basis, if the performance of that preferred NIC degrades or the connection fails. This is done transparently to the application, which is not aware of these dynamic reconfigurations. The selection of a NIC may depend on different criteria, e.g. the "best available" network may well be the one providing lowest communication latency, or that providing communications at the least expensive rate.

This work has been partially funded by the Italian Ministry of Education, University and Research (MIUR) through project STEM-Net (Prot. 2009S7RLXY_003 - Year 2009).

A prototype implementation of our ABPS mechanism is available to support multimedia services based on the SIP, RTP, RTCP protocols. The software architecture is described in detail in [4]. While actual experiments with the prototype have already shown the effectiveness of the approach in some specific configurations, we want to explore a wider range of configuration settings which are difficult (if not impossible) to set up experimentally. To this aim, we present a family of performance models of multi-NIC communication mechanisms based on Continuous Time Markov Chains (CTMCs) with rewards. The models are quite simple and can be solved efficiently; therefore, they are useful to perform "what-if" analysis over large spaces of configuration parameters. The proposed models can be easily generalized to other multi-NIC communication mechanisms. Furthermore, the models are extensible: by defining suitable reward structures they can be used to estimate a wide range of performance parameters. In this paper we focus on the following metrics: i) availability, i.e. the fraction of time during which the MN is able to communicate with its Correspondent Node (CN); ii) reliability, i.e. the probability that an uninterrupted service exceeds a certain time interval; iii) throughput i.e. the average rate of successful data transmission. We compare ABPS with a classic Session Initiation Protocol (SIP)-based multi-NIC communication approach, and a simple approach exploiting a single NIC. Results show that the ABPS mechanism exhibits consistently better performance.

This paper is organized as follows. Section II describes the ABPS mechanism. Section III briefly introduces the basic concepts of Markov chain models. In Section IV we describe the Markov models for a "simple" communication approach using a single NIC, as well as for the SIP-based and ABPS mechanisms. The models are evaluated in Section V. Finally, concluding remarks are given in Section VI.

II. The ABPS Mechanism

The idea at the basis of the ABPS mechanism is simple: during application data transmission (and reception) the MN can use all the available NICs it has on board simultaneously, differentiating the choice of the actual NIC to use from datagram to datagram. This is carried out transparently to both the application running in the MN and producing (consuming)

Algorithm 1 ABPS Client Proxy

```
1:                                                    {NICs' management}
2: upon available connection for NIC
3: SETUP(NIC)
4:
5: upon successful setup for NIC
6: availableNICs.ADD(NIC)
7: activeNIC ← SELECTPREFERRED(availableNICs)
8:
9: upon disconnection for NIC
10: availableNICs.REMOVE(NIC)
11: activeNIC ← SELECTPREFERRED(availableNICs)
12:                                                   {Communication}
13: upon new data to transmit
14: SEND(data, activeNIC)
15: SETTIMEOUT(data)
16:
17: upon data received from ABPS server proxy
18: PUSHORDEREDBUFFER(data,bufferToApp)
19: SENDORDEREDDATA(bufferToApp)
20:
21: upon ACK for data received
22: REMOVETIMEOUT(data)
23:
24: upon timeout ACK for data
25: NIC ← availableNICs.SELECTALTERNATIVENIC()
26: SEND(data, NIC)
27: SETTIMEOUT(data)
```

Algorithm 2 ABPS Client Server

```
1: upon new data from ABPS client proxy
2: activeIP ← RETRIVEIPSENDER(data)
3: CN ← RETRIVEIPDESTINATION(data)
4: RELAY(data, CN)
5:
6: upon new data from CN
7: RELAY(data, activeIP)
```

the data being transmitted (received) and its peer at the CN; thus, applications can continue to employ classic end-to-end transport (and application)-layer communication protocols (UDP, TCP) regardless of ABPS.

In order to let the data flow through different NICs, and deliver the related contents as a single flow to the application, additional software components are needed, in charge of managing data coming from, and going out through, different NICs. Specifically, the ABPS requires the following two additional software components: i) the "ABPS client proxy", and ii) the "ABPS server proxy", described below (see [4] for a complete explanation).

A. ABPS Client Proxy

The ABPS Client Proxy runs in the MN. It is in charge of maintaining the seamless multi-path communication channel between an application running in its home MN and the corresponding peer at the CN. Algorithm 1 shows the principal activities carried out by this software component. There are two types of managed events: those related to the management of some NIC (corresponding to some state change in the model that we will present in the next Section), and those related to the communication of the MN with its CN.

As to the NICs' management, the Client Proxy handles the following three events (lines 1–10): i) when an access point is detected for a given NIC, the Client Proxy starts a setup procedure to activate a connection and exploit that NIC; ii) upon successful connection setup, the Client Proxy includes that NIC in a list of NICs available to transmit data; iii) when a disconnection is detected through a NIC, that NIC is removed from the list of active NICs.

As to the communication, the Client Proxy intercepts and manages data coming from/to the application interacting with the Server Proxy (lines 12–27). In essence, when there are data to transmit, it sends those data to the Server Proxy via the preferred NIC. Typically, data may have to be fragmented in a sequence of datagrams, in order to be transmitted. As multiple networks can be used during the communication, it is possible that the transmitted datagrams be received out of order. To this end, a sequence numbering scheme is used that enables the receiver Server Proxy to order the received datagrams and discard possible duplicates. Finally, the Client Proxy exploits ACKs to identify if some datagram is to be re-transmitted. When a timeout for the reception of an ACK occurs, the Client Proxy tries to retransmit that datagram through an alternative NIC among those active at that time.

B. ABPS Server Proxy

The ABPS Server Proxy runs on a remote host, separate from that of the Client Proxy. The Server Proxy operates basically as a relay between the MN and its CN. In practice, it is convenient to place the ABPS Server Proxy on a fixed node located outside any firewall and NAT system. Then, the ABPS Client and Server Proxies collaborate and adopt policies for load balancing and recovery, in order to maximize throughput and minimize loss rate and economic costs.

As summarized in Algorithm 2, the ABPS Server Proxy collects all the datagrams coming from the MN, via different NICs and sends them to the CN (which is unaware of the presence of the ABPS Server Proxy).

III. MARKOV CHAINS

A stochastic process $\{X(t), t \geq 0\}$ defined over the discrete state space $\{1, \ldots, N\}$ is a CTMC if, for any sequence $t_0 < t_1 < \ldots < t_n < t_{n+1}$ of times, the probability that the system is in state $X(t_{n+1})$ at time t_{n+1} only depends on the previous state $X(t_n)$ at time t_n [5]:

$$P\left(X(t_{n+1}) = x_{n+1} | X(t_n) = x_n, \ldots, X(t_0) = x_0\right) = P\left(X(t_{n+1}) = x_{n+1} | X(t_n) = x_n\right)$$

A CTMC can be fully defined in term of an *infinitesimal generator matrix* $\mathbf{Q} = [Q_{i,j}]$: for each $i \neq j$, $Q_{i,j}$ is the transition rate from state i to state j. The elements $Q_{i,i}$ are defined such that $\sum_{j=1}^{N} Q_{i,j} = 0$.

We denote with $\boldsymbol{\pi}(t) = (\pi_1(t), \pi_2(t), \ldots, \pi_N(t))$ the *state occupancy probability vector* at time t, $\pi_i(t)$ being the probability that the system is in state i at time $t \geq 0$.

Given the infinitesimal generator matrix $\mathbf{Q}$ and the initial state occupancy probabilities $\boldsymbol{\pi}(0) = (\pi_1(0), \pi_2(0), \ldots, \pi_N(0))$, the occupancy probabilities $\boldsymbol{\pi}(t)$ at time t can be computed as the solution of the differential equation $\frac{\mathrm{d}\,\boldsymbol{\pi}(t)}{\mathrm{d}\,t} = \boldsymbol{\pi}(t)\mathbf{Q}$ given initial condition $\boldsymbol{\pi}(0)$, which can be expressed in closed form as:

$$\boldsymbol{\pi}(t) = \boldsymbol{\pi}(0)\exp(\mathbf{Q}t) \tag{1}$$

where $\exp(\mathbf{Q}t)$ is the matrix exponential of $\mathbf{Q}t$. Under certain conditions [5], there exists a *stationary state occupancy probability* $\boldsymbol{\pi} = \lim_{t\to+\infty}\boldsymbol{\pi}(t)$, which is independent from $\boldsymbol{\pi}(0)$. The stationary state occupancy probability vector $\boldsymbol{\pi}$ can be computed as the solution of the following linear system:

$$\boldsymbol{\pi}\mathbf{Q} = \mathbf{0}, \quad \boldsymbol{\pi}\mathbf{1}^T = 1 \tag{2}$$

Of particular interest for this paper are *Markov reward models*. We associate to each state i of the state space $\{1, \ldots, N\}$ a reward r_i; $\mathbf{r} = (r_1, \ldots, r_N)$ is the vector of per-state rewards. Rewards have the following meaning: for each period of duration $\mathrm{d}\,t$ spent in state i, the total accumulated reward is increased by $r_i\,\mathrm{d}\,t$. Given initial state occupancy probabilities $\boldsymbol{\pi}(0)$, the mean total reward $L(t)$ accumulated in $[0, t]$ is

$$L(t) = \int_0^t \boldsymbol{\pi}(t)\mathbf{r}^T dt \tag{3}$$

If there are no absorbing states (a state is absorbing if it has no outgoing transitions, meaning that once an absorbing state is reached the system will stay there indefinitely), we can define the mean stationary reward rate as

$$L = \boldsymbol{\pi}\mathbf{r}^T \tag{4}$$

If there are absorbing states, L is unbounded and hence not meaningful. A different quantity, the *mean total reward until absorption* $L(\infty)$, can be computed as follows. Let τ the set of transient (i.e., non-absorbing) states, and $\mathbf{Q}_\tau$ the restriction of $\mathbf{Q}$ to the transient states only. Taking the restriction $\boldsymbol{\pi}_\tau(0)$ of the initial probability vector, we have [5]:

$$L(\infty) = -\boldsymbol{\pi}_\tau(0)\mathbf{Q}_\tau^{-1}\mathbf{r}^T \tag{5}$$

IV. Modeling Network Interfaces

In this section, we model the ABPS mechanism and compare it with the following two different mechanisms; namely, i) one based on a classic communication scheme employing a single network (termed "simple" in the rest of this paper), and ii) another SIP-based scheme that allows the mobile terminal to utilize different network interfaces. We model the above three mechanisms as CTMCs and measure their performance in terms of availability, and power consumption. Our models go beyond existing results (e.g., [6]) as we consider the high-level behavior of the whole MN rather than the low level details of the network interfaces.

In the models below, we mainly focus on the local states of a MN, without going into the details of the creation and the management of a connection. In fact, this would entail to consider issues concerned with the communication with other distributed entities, and issues arising at the data link, transport and session layers. All the steps related to the configuration of a NIC, the connection of the MN to an access point, and the configuration of the NIC's IP address, are common to all the considered models. Hence, we will model all the activities related to the setup of a communication with a single state. Similarly, the time interval during which the MN is properly configured to communicate with a CN is modeled with a single state, without discriminating if a real communication (e.g. a VoIP call with the CN) is active or not. In fact, this does not influence the network management at the MN.

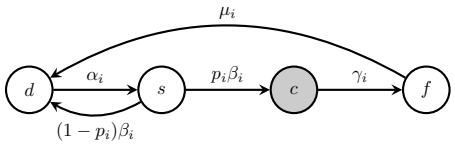

Fig. 1. Markov model of a single NIC

A. Single interface

During a traditional connection-oriented communication, which employs a single network interface among those available (say, NIC i), the behavior of a MN can be modeled using the four-state chain shown in Figure 1. At the beginning, the mobile terminal is disconnected (state d); at this state, the MN is scanning the selected network, looking for some access point to connect with. (This state includes also the time spent trying to connect unsuccessfully to unavailable access points.)

Once an access point is found, the MN sets up a connection with it; the setup is successful with probability p_i, and fails with probability $(1 - p_i)$. After a successful setup, the NIC is configured and can be identified by an IP address (either dynamically or statically assigned). This setup phase corresponds to state s in Figure 1. This state includes also session layer setup operations, e.g. registration to some lobby server, such as the SIP registration server.

Then, the mobile terminal is connected to the service and it may communicate with other nodes (state c). If the connection is lost, for any reason, the mobile terminal passes to a state f. In this state, the communication has failed and no datagrams can be delivered to the receivers, but the MN is not yet aware of such failure and continues to send data until it detects, eventually, that the connection was lost and enters in state d.

The transition from a state to another is modeled through transition rates (namely α_i, β_i, γ_i, μ_i) whose magnitude is inversely proportional to the average time a node passes in the departing state, as in [5]. These transitions depend on the particular NIC i in use, since different network technologies usually offer different capabilities to connect to a given access point, depending on different characteristics such as its availability and signal strength while the MN is moving. In this CTMC, the MN can communicate only when it is in state c (the shaded states in Figure 1).

If we enumerate the states as $d = 1$, $s = 2$, $c = 3$ and

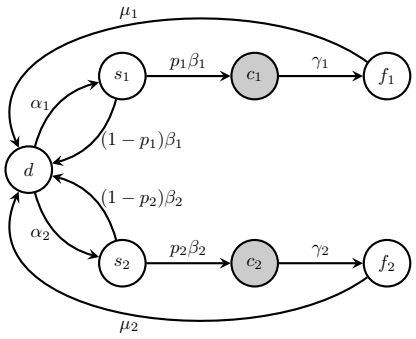

Fig. 2. SIP-based communication with two NICs

$f = 4$, the infinitesimal generator matrix $\mathbf{Q}$ for the chain in Figure 1 is defined as:

$$\mathbf{Q} = \begin{pmatrix} -\alpha_i & \alpha_i & 0 & 0 \\ (1-p_i)\beta_i & -\beta_i & p_i\beta_i & 0 \\ 0 & 0 & -\gamma_i & \gamma_i \\ \mu_i & 0 & 0 & -\mu_i \end{pmatrix} \quad (6)$$

B. Multiple Interfaces: SIP

The Session Initiation Protocol (SIP) is a signaling protocol widely used for controlling communication sessions [7]. When dealing with multiple wireless NICs, it is quite usual to employ a SIP-based approach [1], [2], [4], [8]–[10]. When modeling a SIP-based approach, we assume that the MN has the possibility of accessing different networks through different NICs, among those it has on board. Once the MN selects a network, it sets up the connection through a specific NIC and then can communicate through that NIC until that connection is released or a failure occurs.

Figure 2 shows such a model where two networks are considered; the model can be easily extended to include an arbitrary number of networks (if there are K interfaces, the model will have $1 + 3K$ states). From the state d, the MN has multiple choices, one for each NIC (hence, two in the example depicted in Figure 2). Once a selection is made, the operation mode is the same as that described the previous Subsection. Depending on the networks involved, the transition rates may be different. Again, the states in the shaded nodes in Figure 2 represent those states where a communication is available. Note that the complexity of a SIP-based approach, with respect to a simple communication based on a single network, is hidden behind the message exchange between the MN and the SIP server, during the setup phase (state si).

C. Multiple Interfaces: ABPS

The ABPS mechanism can be modeled through a CTMC as depicted in Figure 3. For the sake of simplicity, in the model we assume that a MN embodies two active NICs. Needless to say, as in the case of the SIP model, the ABPS model can be easily extended to deal with MNs incorporating an arbitrary number of NICs (if there are K NICs, the ABPS model will have 4^K states). In Figure 3, states are identified by two letters; the first letter indicates the state of the first NIC, while the second indicates the state of the second NIC. The two NICs behave independently of each other; thus, the set of possible states is given by the Cartesian product of the states that each single NIC can assume. The system starts from state dd, where both NICs are disconnected; hence, the MN has no active connections, nor any network is in a set up phase. Once an access point is found for a given NIC (e.g., NIC 1), the MN sets up a connection (entering state sd) and eventually moves to the state where that NIC is connected (cd, in our example). In general, owing to the above mentioned independent behavior of the NICs, it is possible that, as a NIC goes through a state transition, the other NIC (namely, NIC 2 in Figure 3) changes its state as well, regardless of the state of the first one. As in the previous models, there is the possibility that a connection fails. In this case, the state of the considered NIC passes from state c to a state f, before going back to d. It is worth pointing out that the d, s, c states for a NIC correspond to events handled by the Client Proxy during the NIC's management. As already mentioned, state f is not handled by the Client Proxy as it corresponds to the time period during which the MN is unaware of a disconnection.

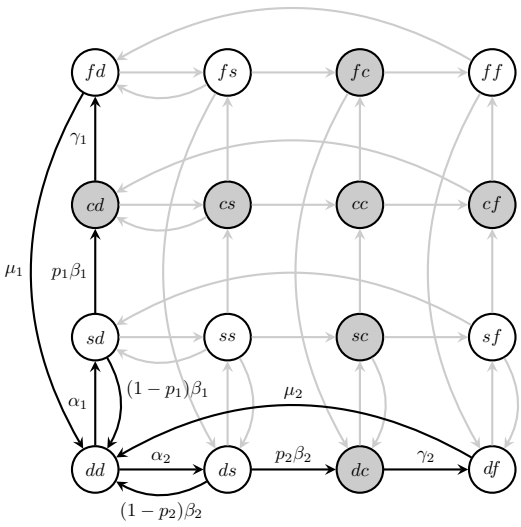

Fig. 3. ABPS communication approach, 2 NICs employed.

The shaded nodes in Figure 3 show that the MN is able to continue its communication when at least one of the available NICs is in the connected state c. The Markov model of ABPS assumes that the probability that both NICs change state at the same time is negligible [5].

V. Analysis

The Markov models described in the previous section can be used to compute many useful performance measures. In this paper we will focus on availability, reliability and throughput. We consider the scenario in which the MN has two NICs (a UMTS and Wi-Fi card, respectively). The ABPS approach employs both networks on a per-packet basis, provided that the required connections be active. The SIP approach configures

TABLE I
PARAMETER VALUES

	α	β	γ	μ	p
UMTS	$1/6.024$	$1/1.5$	$1/500 - 1/2500$	1	0.99
Wi-Fi	$1/7.5$	$1/1.5$	$1/5 - 1/50$	1	0.9

one of the available connections, while the "simple" approach uses the UMTS connections only. To analyze the CTMCs, we have employed the queueing toolbox for GNU Octave [11].

Parameter settings: Table I show the parameter values we consider in the rest of this paper. The values have been selected as the inverse of estimates we measured experimentally. We considered a mobile user walking at an average speed of 6 Km/h in an urban area. We assumed that, during his/her walk, the user is able to find a Wi-Fi access point every 20–50 m, on average (this is true in our university district in Bologna–Italy). Therefore, a new connection is attempted on average every $20s$, which results in $\alpha_{\text{WiFi}} = 1/20$. For the parameters γ_{WiFi} and γ_{UMTS}, which control the duration of an active connection, we consider a range of possible values and study how the performance measures change. We keep the values of other parameters fixed (β_{WiFi}, β_{UMTS}, μ_{WiFi}, μ_{UMTS}). The reason is that the duration of an active connection (controlled by the rates γ) is more likely to change due to different mobility patterns or availability of access points, while the time needed to set up a connection, or detect that a connection has failed, are likely to remain unaffected.

Availability: The system availability is the long term fraction of actually available service, i.e., the fraction of time during which the MN is able to communicate with its CN. To compute the availability we assign a unitary reward rate to states where the MN is allowed to communicate, i.e., the shaded states in Figures 1–3. All other states are given reward zero. From (4), the steady state availability A of a given communication model is simply the sum of steady state probabilities π_k for "up" states k:

$$
\begin{aligned}
A_{\text{UMTS}} &= \pi_c \\
A_{\text{SIP}} &= \pi_{c_1} + \pi_{c_2} \\
A_{\text{ABPS}} &= \pi_{c,d} + \pi_{c,s} + \pi_{c,c} + \pi_{c,f} + \pi_{d,c} + \pi_{s,c} + \pi_{f,c}
\end{aligned}
$$

Figure 4 shows the availability of the three different systems described above. ABPS exhibits higher (better) availability than the other two communication models, which is somewhat expected. Interestingly, the SIP model provides worse results than the "simple" model employing a single UMTS connection. This can be explained by observing that, with the parameters from Table I, the Wi-Fi connection is more unreliable than UMTS: in fact the average duration of a Wi-Fi connection is $1/\gamma_{\text{WiFi}}$, which is significantly lower than $1/\gamma_{\text{UMTS}}$. Therefore, if the MN chooses a Wi-Fi network, then it will probably lose that connection shortly after; this will cause the system to open a new connection. Conversely, the "simple" approach always uses UMTS, which in our case is more reliable. In contrast, ABPS can switch from one network to the other causing no disconnections at the session/application layers.

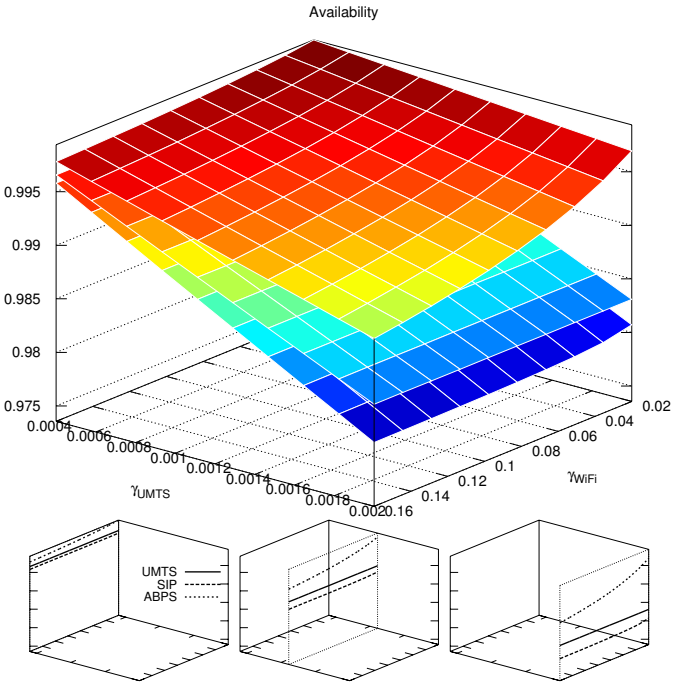

Fig. 4. Availability (higher is better)

A. Reliability

One of the most important metrics in the context of wireless multimedia communications, such as VoIP over mobile nodes, is the probability that uninterrupted service exceed a certain time interval. Specifically, we look for an estimation of how long a communication can last before a session disconnection occurs, which we denote with Mean Time To Failure (MTTF).

To compute the MTTF of our three systems, we first make all "down" states absorbing, by removing all outgoing transitions. Specifically, in the model in Fig. 1, the state f becomes absorbing. In the SIP-based model of Fig. 2 states f_1 and f_2 become absorbing. Finally, in the ABPS model of Fig. 3 the absorbing states are all states in which one interface is in state f (*failed*) and the other one is not in state c (*connected*).

Since only states in which the MN can transmit are considered operational, we define reward vectors $\mathbf{r}$ which associates a reward 1 to "up" states (the shaded states in Fig. 1, 2 and 3) of the modified models. The system reliability R is the the total accumulated reward until absorption using Eq. (5).

This measure, obtained using the same parameter setting as before, is shown in Figure 5. This Figure shows that ABPS provides notably higher reliability than SIP and and the "simple" approach with a single UMTS connection.

B. Throughput

We compute the throughput using a reward model. Let T_1 and T_2 the the throughput of the two NICs. For each model, we associate a reward T_1 for states in which the first NIC is in the transmitting state (state c), and a reward T_2 for states

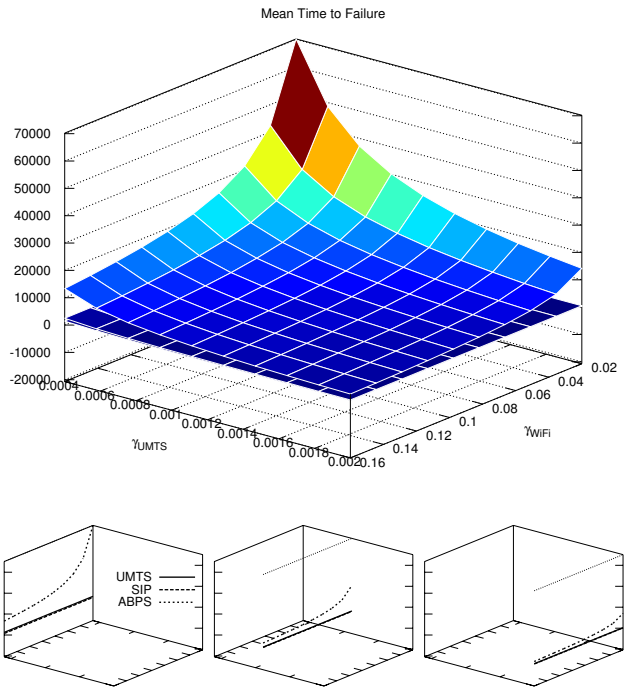

Fig. 5. Reliability: Mean Time to Failure (higher is better)

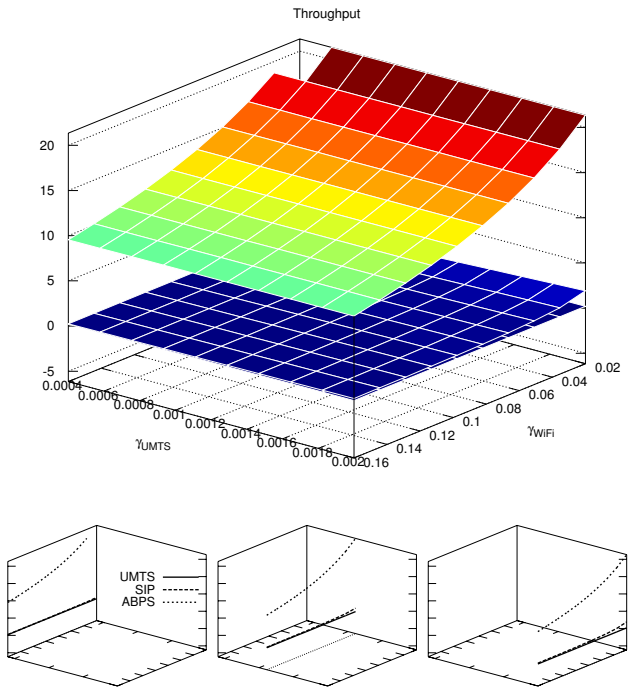

Fig. 6. Throughput (higher is better)

in which the second NIC is transmitting; all other states have reward set to zero. For ABPS we use a reward T_1 for states $(c_1, \cdot)$, a reward T_2 for states $(\cdot, c_2)$ and a reward $\max(T_1, T_2)$ for state (c_1, c_2), in which the client uses the fastest interface. We use $T_1 = 0.2\text{Mbps}$, which is the throughput of a UMTS connection, and $T_2 = 26\text{Mbps}$, which is the throughput of a WiFi connection. The throughput $Tput$ of the various connection mechanisms can then be computed as:

$$\begin{aligned} Tput_{\text{UMTS}} &= T_1 \pi_c \\ Tput_{\text{SIP}} &= T_1 \pi_{c_1} + T_2 \pi_{c_2} \\ Tput_{\text{ABPS}} &= T_1 \left(\pi_{c_1,d_2} + \pi_{c_1,s_2} + \pi_{c_1,f_2}\right) \\ &\quad + T_2 \left(\pi_{d_1,c_2} + \pi_{s_1,c_2} + \pi_{c_1,c_2} + \pi_{f_1,c_2}\right) \end{aligned}$$

Results are shown in Fig. 6: unsurprisingly, SIP and ABPS provides increasingly higher throughput, with ABPS offering the better throughput due to its ability of using both NICs at the same time.

VI. Conclusion

In this paper we proposed a family of analytical models for evaluating multi-NIC communication mechanisms. The models have been used to analyze availability, throughput and reliability of ABPS and compare them to results obtained for SIP-based and traditional single NIC systems. The model is based on CTMCs with rewards and can be efficiently solve to compute many useful performance measures. The results of this comparison show that ABPS outperforms the other two mechanisms in terms of availability in the scenarios we have considered. The results are providing useful insights on how to improve ABPS.

References

[1] P. Bellavista, A. Corradi, and L. Foschini, "Ims-compliant management of vertical handoffs for mobile multimedia session continuity," *Comm. Mag.*, vol. 48, pp. 114–121, April 2010.

[2] G. Camarillo and M.-A. Garca-Martn, *The 3G IP Multimedia Subsystem: Merging the Internet and the Cellular Worlds*, 3rd ed. Wiley Publishing, 2008.

[3] S. Ferretti and V. Ghini, "A web 2.0, location-based architecture for a seamless discovery of points of interests," in *Fifth Advanced Int. Conf. on Telecommunications, AICT 2009*, 2009, pp. 226–231.

[4] V. Ghini, S. Ferretti, and F. Panzieri, "The "always best packet switching" architecture for SIP-based mobile multimedia services," *Journal of Systems and Software*, vol. 84, no. 11, pp. 1827–1851, 2011.

[5] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, *Queueing networks and Markov chains - modeling and performance evaluation with computer science applications; 2nd Edition*. Wiley, 2006.

[6] M. K. Chibesakunda and P. S. Kritzinger, "A methodology for analyzing power consumption in wireless communication systems," Tech. Rep. CS03-28-00, Dec. 15 2003.

[7] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session initiation protocol," Internet Engineering Task Force, United States, 2002.

[8] A. Devlic, "SIP-based context distribution: does aggregation pay off?" *SIGCOMM Comput. Commun. Rev.*, vol. 40, pp. 35–46.

[9] M. Femminella, R. Francescangeli, F. Giacinti, E. Maccherani, A. Parisi, and G. Reali, "Design, implementation, and performance evaluation of an advanced SIP-based call control for voip services," in *Proceedings of the 2009 IEEE international conference on Communications*, ser. ICC'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 1465–1469.

[10] A. Peddemors, H. Zandbelt, and M. Bargh, "A mechanism for host mobility management supporting application awareness," in *Proc 2nd Int. Conf. on Mobile systems, applications, and services (MobiSys '04)*. New York, NY, USA: ACM, 2004, pp. 231–244.

[11] M. Marzolla, "The `qnetworks` toolbox: A software package for queueing networks analysis," in *Analytical and Stochastic Modeling Techniques and Applications, 17th International Conference, ASMTA 2010, Cardiff, UK, Proceedings*, ser. Lecture Notes in Computer Science, K. Al-Begain, D. Fiems, and W. J. Knottenbelt, Eds., vol. 6148. Springer, Jun.14–16 2010, pp. 102–116.