CONTENT DISSEMINATION SCHEMES FOR MOBILE CLOUDS: MODELING, ANALYSIS, AND VERIFICATION

A Thesis
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Ankan Ghosh

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Electrical and Computer Engineering

November 2014

Fargo, North Dakota

North Dakota State University
Graduate School

Title

CONTENT DISSEMINATION SCHEMES FOR MOBILE CLOUDS:
MODELING, ANALYSIS, AND VERIFICATION

By

Ankan Ghosh

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Dr. Samee U. Khan
Chair

Dr. Scott C. Smith

Dr. Sudarshan Srinivasan

Dr. Ying Huang

Approved:

11.17.2014 Dr. Scott C. Smith

Date Department Chair

ABSTRACT

In this work, we capitalize on our previous work on the Hybrid Scheme for Message Replication (HSM) for opportunistic networks, to develop two content dissemination schemes for mobile clouds, namely, the Forecast and Relay (FAR) and Utility-Based Scheme (UBS). Simulation results with synthetic mobility models validate that the proposed schemes outperform existing routing strategies, such as the *PRoPHET*, *Epidemic*, *Random*, and *Wave*. We have exploited High-Level Petri Nets to model and analyze the communication processes encompassing HSM, FAR, and UBS. The UBS has been conceived to overcome a crucial design limitation of HSM and FAR, made evident through formal analysis. We have used the New Symbolic Model Verifier (NuSMV) to verify the three schemes against the identified limitation, by using optimization techniques. The verification results affirm the correctness and scalability of the models. The work corroborates that formal verification can be leveraged to design newer and efficient content dissemination schemes.

ACKNOWLEDGMENTS

I am grateful to acknowledge and thank all of those who have assisted me in my graduate program at the North Dakota State University. I would like to express my deepest gratitude and a bundle of thanks to my academic advisor Dr. Samee U. Khan. His guidance, support, and patience throughout my years as a graduate student are truly appreciated. Special thanks to my other graduate committee members, Dr. Scott C. Smith, Dr. Sudarshan Srinivasan, and Dr. Ying Huang. I would also like to express that I am extremely thankful to the Department of Electrical and Computer Engineering, for supporting me financially throughout the course of my degree.

Last but not the least, I would like to thank all of my colleagues at NDSU for their kind help and support during the course of my Master's program. I am grateful and deeply appreciate the support offered by Dr. Osman Khalid, Dr. Saif U. R. Malik, and Mohana A. L. Dubasi.

DEDICATION

I would like to dedicate this thesis to my parents, for their unconditional love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1. Overview and Motivation

The radical proliferation of the technologically enhanced and computationally enriched mobile devices, such as the smartphones, tablets, and PDAs, has substantially ameliorated the lives of the human beings. A wide range of applications adroitly supported by the smartphones, such as the social networking, pattern recognition, image processing, e-commerce, games, sensors, and context-aware applications, have played a key role in the amelioration. The conventional Mobile Cloud Computing (MCC) platforms further augment the computational capabilities and energy efficiency of the mobile devices, by offloading the complex and resource-hungry tasks to the infrastructure-based, resource-rich, remote cloud systems [1]. However, the context-aware applications necessitate the timely and cost-effective exchange of important information that might not be satiated by the remote cloud [2]. The factors that impede the cloud-based offloading services include: **(a)** high network traffic and bandwidth consumption [3], **(b)** latency in accessing the remote cloud [4], and **(c)** cloud connectivity [4], [5], and the requirement of the Internet [6].

The mobile cloud is an alternative MCC paradigm that has been conceived to transcend the limitations of the conventional infrastructure-based MCC frameworks [5], [7]. The mobile clouds efficiently capitalize on the diverse computational, processing, and storage capabilities of the mobile devices, by forming one or more local clouds or networks of devices [5]. Such frameworks bank on the opportunistic contacts among the mobile devices to exploit the computational resources, and are proactive, cost-effective, and dynamic in maneuvering the resources or adapting to the connectivity changes [5]. The processing time required to complete a task is reduced [5] and the requirement of the Internet is eliminated [6]. A mobile cloud can be ideated as an Opportunistic Mobile Network (OMN), or synonymously, a Delay-Tolerant Network (DTN) [7], [8]. Such networks can provide services, such as the: **(a)** opportunistic computing [7], **(b)** content delivery [8], [3], **(c)** opportunistic job sharing [5], **(d)** execution of computationally intensive tasks [9], **(e)** information dissemination [6], and **(f)** vehicular cloud computing [2].

An important aspect that becomes indispensable is the routing scheme (protocol) used for the content dissemination within the mobile cloud [3], [8]. The existing approaches to the content dissemination schemes for the mobile cloud networks have looked to draw advantage of the routing schemes for the OMNs or DTNs that already exist in the literature [7], [4], [6], [8], [9]. The research fraternity has dealt well with the multicast routing in the delay-tolerant Mobile Social Networks (MSNs) [3] and the context-aware unicast scenario in the DTNs [10]. However, the generic routing schemes for the mobile clouds have not received much attention. Moreover, a detailed, formally analyzed communication model for the mobile clouds that generalizes the complexities in the communication processes is still a requisite. Another key aspect that holds the potential to uplift the performance of any routing schemes is the formal verification. However, verification remains underutilized and is generally restrained to the use of ensuring the correctness of models.

In this work, we study the scenario of routing the content (messages) between the information producers (sources) and requesters (destinations), within a mobile cloud. The mobile nodes (smartphones, tablets, or PDAs) in the cloud form an OMN and rely on opportunistic contacts and mutual cooperation to disseminate and deliver the generated content. The content dissemination schemes (to be discussed) exploit the mobility patterns and the temporal contacts of the nodes to predict the future contact opportunities. The decisions to replicate or relay the messages are based on the aforesaid predictions. The predictions are based on the time-series data maintained by every node in the network, for every other node. The said predictions are the computations performed by the schemes, on the nodes, to achieve the task of content dissemination. In the said context, we have extended our work on the Hybrid Scheme for Message Replication (HSM) for the DTNs [11]. The *HSM* serves as a guiding base to model and analyze the communication process in the OMNs and to develop two efficient delay-tolerant routing schemes for mobile clouds. The schemes have been formally analyzed and verified. The work has been supported by the contributions of my colleagues Dr. Osman Khalid, Dr. Saif U. R. Malik, Mohana A. L. Dubasi, and my adviser Dr. Samee U. Khan.

1.2. Contributions

The first scheme that we present is the Forecast and Relay routing scheme (FAR), envisioned to lower the number of prediction-based computations than the *HSM*. The second routing scheme, namely the Utility-Based Scheme (UBS), has been conceived precisely to eliminate a crucial design limitation of the *HSM* and *FAR*. The limitation has been corroborated through formal analysis and verified using the New Symbolic Model Verifier (NuSMV). To model and formally analyze the three schemes, we have harnessed the flexibility and potency offered by the High-Level Petri Nets (HLPNs) and the Z specification language. The HLPN models furnish the: **(a)** comprehensive overview of the components and information flow, encompassing an opportunistic mobile cloud and **(b)** detailed mathematical analysis of the communication processes. The simulation results of the schemes are based on the synthetic mobility traces in the Opportunistic Network Environment (ONE) simulator [12]. The scalability of the schemes has been evaluated on the basis of performance metrics, such as the delivery ratio, latency, and overhead.

The verification involves the modeling of the highly unpredictable and dynamic communication processes in a comprehensive, yet optimized way, and is an exceedingly time-consuming process. The verification processes comprised of the: **(a)** translation of the HLPN models of the three schemes into the NuSMV models, written in the NuSMV language, **(b)** automated formal verification of the three models against an identified limitation as the specification in the Computational Tree Logic (CTL), through complete (not bounded) model checking, **(c)** testing of the models in the presence of up to a 100 nodes and 100 messages to verify the scalability and correctness, and **(d)** use of the optimization techniques offered by the NuSMV to verify the specifications in finite time. To the best of our knowledge, no prior work has been devoted to the formal modeling, analysis, and optimized verification of novel routing schemes for the mobile clouds. The work will provide the readers with a thorough understanding of all of the communication processes associated with the mobile cloud or OMN routing schemes that are generally overlooked. We affirm that the HLPNs can be used effectively to model the dynamic OMNs. Moreover, the work will corroborate the fact that the formal verification of complex routing schemes may not be merely limited to the verification of the correctness of the models. Contrarily, verification can also be capitalized upon to pave the way for newer routing models, verify their scalability, and to enhance the performances.

Before we move on to the next chapter, we briefly review the contributions of our work here. In this thesis, we model, formally analyze, and verify the correctness and scalability of the *HSM* [11], and develop two efficient novel content dissemination schemes (the *FAR* and *UBS*) for the mobile clouds that are also modeled, formally analyzed, and verified for the correctness and scalability. We identify a design limitation common to the *HSM* and *FAR* through formal analysis and verification and verify that the *UBS* (conceived to obliterate the design limitation) is devoid of the limitation and provides significant performance enhancements. To summarize, the contributions of our work are the following.

- We present a detailed model and formal analysis of the communication processes in the *HSM*.
- The *HSM* is evaluated against the existing OMN routing strategies, such as the *PRoPHET* [13], *Epidemic* [14], *Random* [15], and *Wave* [16]. The simulation has been performed with an increased node buffer size that is characteristic of a smart mobile cloud device.
- We have designed, modeled, formally analyzed, and evaluated the *FAR*, developed to lower the number of prediction-based computations than the *HSM*.
- We have identified and corroborated a crucial design limitation of the *HSM* and *FAR* through formal analysis and verified the same using the NuSMV.
- We have developed, modeled, formally analyzed, and evaluated the *UBS*, conceived specifically to obliterate the design limitation common to the *HSM* and *FAR*.
- We have verified that the *UBS* is devoid of the design limitation using the NuSMV and also provides significant performance enhancments.
- We have used model checking optimizations to verify the correctness and scalability of the models of the *HSM*, *FAR*, and *UBS*.

1.3. Thesis Outline

The organization of the thesis is summarized as follows. Chapter 2 reviews the related work. In Chapter 3, we describe the tools used in the work. The system model, assumptions, and descriptions of the presented content dissemination schemes, are discussed in Chapter 4. In Chapter 5, we present the modeling, analysis, and verification of the schemes. Chapter 6 examines and describes the results of the simulation and verification of the content dissemination schemes considered in this work. We conclude the thesis and review the future work in Chapter 7.

2. RELATED WORK

The existing content dissemination schemes for the mobile cloud networks, such as [7], [4], [6], [8], and [9], are primarily based on the existing routing schemes for the OMNs. The researchers have exploited the device clouds to address the issues, such as the: **(a)** multicast routing in the MSNs [3] and **(b)** unicast routing in the DTNs [10]. In general, the routing or disseminaton of content in the OMNs or DTNs is still an open research issue. The research fraternity tries to strike a balance between the flooding-based schemes [7], [15], [16], and the selective replication strategy [13]. The flooding-based schemes improve the message delivery and latency at the cost of the resource consumption. The selective replication strategies lower the resource consumption at the cost of decreased message delivery and increase in latency. The modeling and analysis of routing processes in the OMNs has remained restricted to the models of disparate aspects of the communication processes. Few of the instances are the: **(a)** Markov chain models of the message dissemination process [17], **(b)** stochastic models of the delivery delay and the task completion time [7], **(c)** Poisson model of the network and encounter process [18], **(d)** coalitional game model of the decision making process among nodes [18], **(e)** graph model of the mobility and mathematical forecasting model [10], and **(f)** Colored Petri Net (CPN) model of the anycast communication process [19]. The authors in [14] made use of the Queueing Petri Nets to model the communication-based aspects of the DTNs. However, the work was based on the existing routing schemes (similar to [7] and [17]) and presented a theoretical analysis.

Wireless network protocols such as the Bluetooth device discovery, and those encompassing the Wireless Sensor and Local Area Networks, have been verified extensively in the literature [20]. However, the verification of the OMN protocols has received minimal attention. In [19], the proposed CPN model for the opportunistic anycast communication process was abstractly verified without the use of a suitable model checking tool. None of the existing works were conceived for the design, formal analysis, and verification of novel content dissemination schemes for the mobile clouds. Evidently, this is the first amalgamation encompassing the following.

- Delay-tolerant routing in a mobile cloud.
- Detailed modeling and formal analysis of such routing solutions.
- Utilization of formal verification through complete model checking and optimizations to verify the correctness and scalability of such routing models, and to develop a novel and efficient routing scheme.

3. PRELIMINARIES

3.1. High-Level Petri Nets (HLPNs)

The Petri Nets are modeling tools, used for the graphical and mathematical modeling of various systems that can be characteristically concurrent, asynchronous, distributed, parallel, non-deterministic, or stochastic [21]. In this work, a variant of the classical Petri Nets, namely the HLPNs [21], have been used to model the delay-tolerant routing schemes for the mobile clouds. Relevant details on the Petri Nets have been presented in [22].

Definition 1 (HLPN) [21]. A HLPN can be defined as a 7-tuple $N = (P, T, F, \varphi, R, L, M_0)$, where the variable:

1. P represents a finite set comprising of the places.
2. T is a finite transition set, such that $P \cap T = \emptyset$.
3. F represents a flow relation (set of arcs), such that $F \subseteq (P \times T) \cup (T \cup P)$.
4. φ denotes a mapping function, used to map P to the data types, such that $\varphi: P \rightarrow Type$.
5. R represents the rules that are used for mapping T to the predicate logic formulae, such that $R: T \rightarrow Formula$.
6. L represents the labels, and is used to map F to the labels, such that $L: F \rightarrow Label$.
7. M_0 is the initial marking, such that $M: P \rightarrow Tokens$.

The variables P, T, and F, furnish the information about the structure of the HLPN and the variables φ, R, and L, contribute to the static semantics, signifying that the information present in the system is unvarying.

In a HLPN, the places may house tokens of one or more different data types. An example of a HLPN is shown in Fig. 1. The places shown in the figure can be considered to be mapped with various data types, such as: $\varphi(P_A) = \mathbb{P}(\text{Int})$, $\varphi(P_{BE}) = \mathbb{R}(\text{Float})$, $\varphi(P_C) = \mathbb{R}(\text{Double})$, and $\varphi(P_D) = \text{Char}$. To enable or fire a transition, the pre-condition to that transition must hold. The firing of a transition depends on the variables from the incoming arcs. As an example, in Fig. 1, the variables a and b from the places P_A and P_{BE}, respectively, will be responsible for the firing of the transition t_2. The post-condition is the result of a fired transition and utilizes the outgoing variables, such as c (for t_2). An example of a rule for the transition t_2 would be: $R(t_2) = (a = 1) \wedge (b = 2.5) \wedge (c \coloneqq 3.15)$. Simply put, firing an initial transition (t_1) enables the system. The transitions utilize the data flowing through the incoming arcs to perform computations and the outgoing arcs are used to carry the results to the corresponding places.

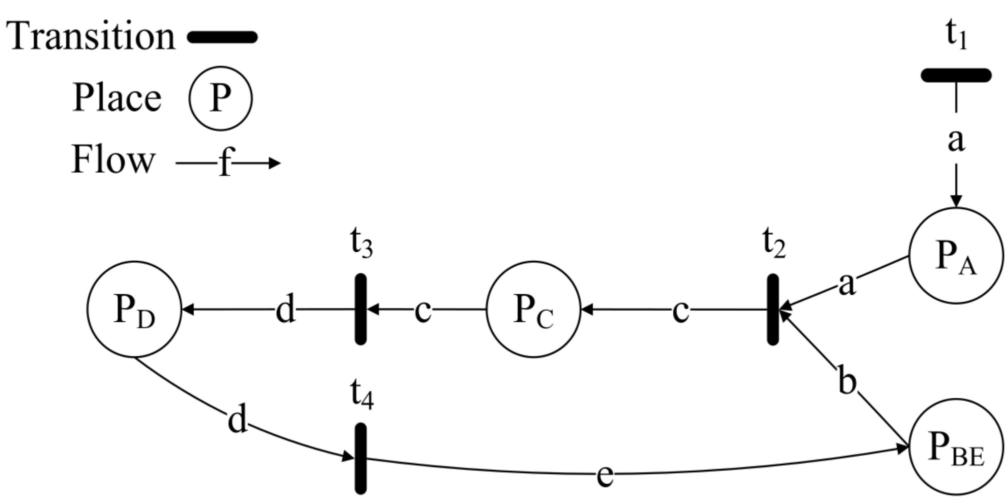


Fig. 1. A High-Level Petri Net.

3.2. NuSMV

The NuSMV [23] is the first model checking software tool that is based on the Binary Decision Diagrams (BDDs). The NuSMV is a reimplemented extension of the Symbolic Model Verifier, conceived for the verification of the Finite State Machines (FSMs) [23]. The NuSMV is an open model checking architecture that finds use in the verification of large industrial systems, as a core for custom verification tools, research applications, and as a testbed for verification techniques [23]. The NuSMV allows the description of complex systems to be decomposed into reinstantiable modules; thereby, facilitating a modular and hierarchical operation [23]. Each of the modules represents a FSM, and collectively, all of the modules form the FSM of the model comprising of a set of states. The FSM under verification is represented by a BDD [24].

The underlying BDD package in the NuSMV (providing the BDD functionalities) is called the CUDD [25]. The NuSMV facilitates the Satisfiability or SAT-based Bounded Model Checking (BMC) using solvers, such as the ZChaff and Minisat [25]. In this work, we have used the BDD-based symbolic model checking algorithms [25] that are complete and furnish a correct answer (*true* or *false*) in finite time. In comparison, the SAT-based BMC inspects until a certain depth of exploration and returns a *false* or an *unknown*. The complete, BDD-based algorithms provide the standard fixpoint computation for the CTL specifications that we have used. Relevant details on the NuSMV functionality can be found in [25].

4. SYSTEM MODEL, ASSUMPTIONS, AND CONTENT DISSEMINATION SCHEMES

4.1. System Model and Assumptions

The system is modeled as a typical OMN (topologically and temporally varying), comprising of a set of mobile nodes, such as the PDAs, smartphones, handheld devices, mobile sensors, and the fixed Access Points (APs). The computationally capable and storage-friendly devices or nodes make opportunistic contacts with each other and form a mobile cloud. The nodes rely on the opportunistic contacts for the exchange of the network state information (through in-band control signaling), and the content (message) dissemination, through mutual cooperation (a service). The content (e.g., emergency, healthcare, weather, local context, and sensor data) can be generated by any node, which can be an information producer or source, for any node acting as an information requester or destination. Fig. 2 exemplifies our mobile cloud model, where *S* and *D* represent a pair of Source and Destination nodes.

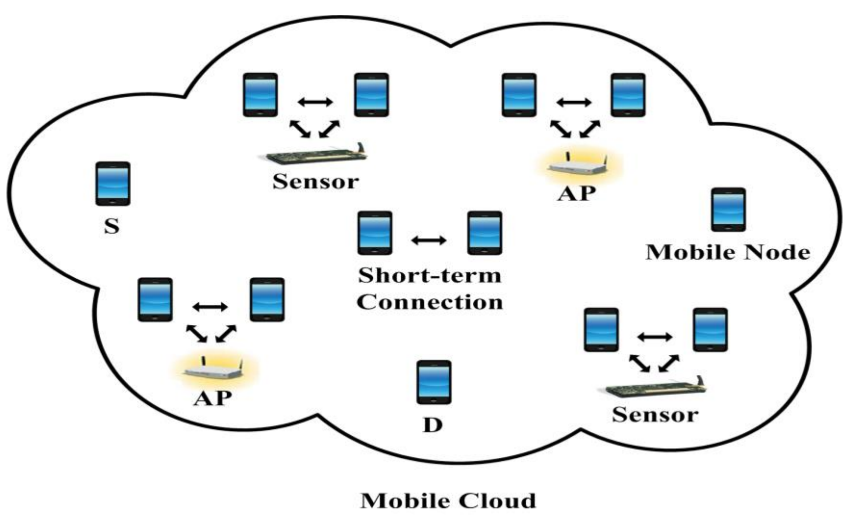


Fig. 2. Opportunistic communication in a mobile cloud.

All of the nodes in the mobile cloud have a unique network identifier or the Node ID (N_ID). Any of the nodes can be a source, relayer, or destination, with respect to the messages in the mobile cloud. The attributes of a message include a particular identifier (Message ID or M_ID), source, destination, life-time or Time-To-Live (TTL), and size. The three lists that a node maintains relative to the messages are the: **(a)** Message List (ML), containing the physical messages generated or to be relayed by the node, in the message buffers, **(b)** Received List (RL), comprising of the M_IDs of the messages that were destined for the node, and **(c)** Acknowledgment List (AL), consisting of the M_IDs of the messages that the node has successfully delivered to the corresponding destinations.

The transfer of the messages between any two nodes in the mobile cloud requires the two nodes to be in each other's transmission range. A message can be transferred directly to the requester, on making an opportunistic contact with the producer. In the absence of an end-to-end path from the producer to the requester, the replicas of the content can be opportunistically relayed by using replication through the intermediate information bearers known as the relayers, towards the requester. The decision to relay or not is based on the underlying content dissemination scheme. The scheme decides whether or not to retain a message after a node relays the replica to a relayer. We assume that the nodes follow repetitive and predictable mobility patterns. The fact that the humans follow predictable and repetitive schedules of meeting (temporally and spatially) conforming to power law distribution has been affirmed by the researchers [26], [27]. However, few of the nodes follow a random mobility pattern as well. To cooperate in the message dissemination, the resourceful nodes allocate a limited portion of their buffers for opportunistic data. In the following text, we briefly describe the three content dissemination schemes considered in this work. We review the *HSM* [11] to build the context for the modeling, analysis, and verification of the scheme.

4.2. Hybrid Scheme for Message Replication (HSM)

On making an opportunistic contact with a relay node (*j*), a node (*i*) carrying a message (content) that is not destined for *j*, decides whether or not to replicate the message on *j*. The *HSM* tackles the decision-making process in the cloud through conditional replication, by computing the utility values of *i* and *j* for the message to be replicated. The utilities are: **(a)** the probability that the message will be delivered to the destination before life-time expiry and **(b)** the probability that the contact duration between a node (*i* or *j*) and the destination will be greater than the time required to transfer the message. If both of the probabilities of *j* are greater than that of *i*, the message will be relayed to *j* and deleted from the buffer of *i*. The process is known as conditional deletion. In the case of *j* exhibiting a higher value of the point **(b)**, as compared to point **(a)**, the message will be replicated on *j* and retained in the buffer of *i*. The *HSM* performs the aforementioned prediction-based computations to conditionally replicate the message between two nodes. The parameters considered in the *HSM* for the conditional replication are the: **(a)** Contact Duration (CD) and **(b)** Inter-Contact Time (ICT).

The CD and ICT values between any two nodes *i* and *j* are denoted by C_i^j and I_i^j, respectively. Each of the nodes in the mobile cloud maintains a 2-tuple, bounded time-series data (of size ω) that are the CD and ICT values for every encounter, represented as: $< C_i^j[\tau], I_i^j[\tau] >$, at time instant τ. The parameter ω denotes the index of the last entry in the time-series. Let T_w^k denote the time since the creation of a message m^k destined for *d*, and T_L^k be the TTL of the message m^k where *k* represents the *kth* message. We call the probability that a node *i* in the mobile cloud will contact *d* before the expiry of the message's TTL as the utility $U_{i,d}^k$ of the node for the current message m^k. This utility value is based on the ICT and is given as [11]:

$$U_{i,d}^{k} = P\left[Z_i^d(\tau) < T_L^k - T_w^k\right]. \tag{1}$$

In the above equation, $Z_i^d(\tau)$ denotes the mean ICT between the nodes *i* and *d*, at time τ, that is obtained from the bounded time-series data stored by the nodes. Few of the nodes in the mobile cloud follow a partially scheduled mobility pattern. Such patterns allow us to use the exponential smoothing to forecast the value of $Z_i^d(\tau)$, given as:

$$Z_i^d(\tau) = (1-\alpha)^{\tau} \cdot Z_i^d[0] + \sum_{k=0}^{\tau-1} \alpha \cdot (1-\alpha)^{k} \cdot I_i^d[\tau - k - 1]. \tag{2}$$

In the above equation, the parameter $0 \leq \alpha \leq 1$ denotes the time-series smoothing constant, $I_i^d[\tau]$ represents the ICT of the node *i* with the node *d* at time instant τ, $Z_i^d[0]$ is the base value of the recursion, and $Z_i^d(\tau)$ denotes the forecasted ICT of the node *i* with the node *d*. The nodes in the mobile cloud allocate a limited memory for the opportunistic data and cannot store the information about all of the past meetings. The sliding time window $[1, \omega]$ limits the maximum number of entries that a node may store. To ensure freshness in the information, the entries in the range $[1, \omega]$ are assigned progressively decreasing weights that allows the recent entries to contribute more to the overall forecasting. The base case value of the recursion $Z_i^d[0]$ is given as:

$$Z_i^d[0] = \frac{1}{\omega} \cdot \sum_{j=0}^{\omega} I_i^d[j]. \tag{3}$$

Equation (3) represents the average of ω entries of the ICT between *i* and d. If T_t^k denotes the time required to transfer a message m^k during an opportunistic contact, then the message will be successfully transferred, if and only if the CD between the two nodes is greater than T_t^k. The utility value denoted as $V_{i,d}^k = P[T_t^k < C_i^d(\tau)]$ represents the probability that the message will be successfully transferred between the nodes *i* and *d* within the mean duration $C_i^d(\tau)$. To compute the CD-based $V_{i,d}^k$, the estimated value of duration between *i* and *d* can be found by replacing I_i^d with C_i^d in (2) and (3).

4.3. Forecast and Relay Scheme (FAR)

The *FAR* is a modified version of the *HSM* that aims at lowering the number of probabilistic computations performed by a node during an opportunistic contact. The *FAR* considers only the CD between any two nodes as an indicator of the meeting quality for performing the prediction-based computation. When two nodes (*i* and *j*) make an opportunistic contact at a time instant *t*, the nodes record the meeting quality, denoted by $C_{ij}(t)$, which is quantified by the CD between the nodes. Each of the nodes in the cloud stores the meeting qualities for other nodes in the network in the form of time-series entries. The higher the value of meeting quality between two nodes, the greater is the probability of a successful message transfer. When a source node *s* generates a message *m* for a destination *d*, and cannot establish a direct contact with *d*, the node decides whether or not to replicate *m* on an intermediate relay node *r* (conditional replication), based on the following.

$$F_{sd}(t) = \phi \cdot C_{sd}(t-1) + (1-\phi) \cdot F_{sd}(t-1). \tag{4}$$

In (4), the parameter $0 \leq \phi \leq 1$ denotes the time-series smoothing constant, $C_{sd}(t)$ represents the meeting quality of *s* with *d* until the time *t*, $F_{sd}(t)$ is the current forecast of the meeting quality between *s* and *d*. In (4), *s* will replicate *m* on *r*, if and only if *r* has a better forecasted meeting quality with *d*. The condition may also be expressed as: $F_{rd}(t) > F_{sd}(t)$. A limit is set to the maximum number of time-series entries stored by a node (denoted as ω) using a sliding time window $[1, \omega]$. When a new entry is added, the oldest entry is automatically deleted. Information freshness and accuracy are ensured by assigning progressively decreasing weights to the older entries and by prioritizing the recent ones. In (4), if we substitute the value of $F_{sd}(t-1) = [\phi \cdot C_{sd}(t-2) + (1-\phi) \cdot F_{sd}(t-2)]$, we get:

$$F_{sd}(t) = \phi \cdot C_{sd}(t-1) + (1-\phi) \cdot [\phi \cdot C_{sd}(t-2) + (1-\phi) \cdot F_{sd}(t-2)]. \qquad (5)$$

By resubstituting the value of $F_{sd}(t-2)$ in (5) we obtain (6) and solving recursively, we finally obtain (7). In (7), each of the entries for the meeting quality $C_{sd}(t)$ has been assigned a certain weight such that as an entry becomes older, it contributes lesser to the overall forecasted value. The base case value of the recursion $F_{sd}(0)$ is given by (8). Equation (8) indicates the average of the meeting qualities of *s* and *d* within the interval $[1, \omega]$.

$$\begin{aligned} F_{sd}(t) = {} & \phi \cdot C_{sd}(t-1) + \phi \cdot (1-\phi) \cdot C_{sd}(t-2) + \phi \cdot (1-\phi)^2 \cdot C_{sd}(t-3) + \cdots \\ & + \phi \cdot (1-\phi)^{t-1} \cdot C_{sd}(0) + (1-\phi)^t \cdot F_{sd}(0). \end{aligned} \qquad (6)$$

$$F_{sd}(t) = (1-\phi)^t \cdot F_{sd}(0) + \sum_{k=0}^{t-1} \phi \cdot (1-\phi)^k \cdot C_{sd}(t-k-1). \qquad (7)$$

$$F_{sd}(0) = \frac{1}{\omega} \cdot \sum_{i=1}^{\omega} C_{sd}(i). \tag{8}$$

4.4. Utility-Based Scheme (UBS)

The *UBS* for the mobile clouds has been conceived and designed, specifically to obliterate a characteristic design limitation common to the *HSM* and *FAR*. The detailed formal analysis of the *HSM* and *FAR* in Chapter 5 corroborate the apparent design limitation that has been formally verified. Precisely, in the *HSM* and *FAR*, a node accepts the same message that it has previously relayed in the network and eventually deleted after one or more replications. A message may be deleted by a node, if the node finds the message to be in the RL or AL of a connected node, during an opportunistic contact. The deletion can also occur due to the conditional deletion in the *HSM* or the deletion due to the lack of buffer space in both of the schemes. Such relaying increases message replicas that cause considerable resource and energy consumptions during the replication process. If a message has been relayed on one or more occasions based on efficient prediction techniques and subsequently deleted, the exchange of newer messages need to be prioritized in a delay-tolerant scenario. The aforementioned approach improves the overall message delivery and overhead.

To overcome the design limitation, the *UBS* implements an additional list named as the Passed Message List (PML), maintained by the nodes in the mobile cloud. When a node replicates a message for the first time, the corresponding M_ID is recorded in the PML. During an opportunistic contact, the M_IDs of the incoming messages are checked against the contents of the PML, and the M_IDs that already exist are disregarded, even if they were deleted from the actual ML. The approach has been formally verified.

The *UBS* is a modified, flooding-based extension of the *HSM*. The modifications are the novelties in the *UBS*, envisioned only to improve the message delivery rate and latency. The nodes maintain the CD- and ICT-based information with the recent entries being prioritized, while the sliding time window sets a limit to the maximum number of entries, as discussed earlier in Section 4.2. If a node *i* is carrying a message destined for *d* and cannot establish a direct contact with *d*, then the node *i* decides whether or not to replicate the message on an encountered relayer, based on the mean CD and ICT values $C_i^d(\tau)$ and $Z_i^d(\tau)$, respectively. We modify and denote the aforementioned mean CD and ICT values here as $C_{i,d}^k(\tau)$ and $Z_{i,d}^k(\tau)$, respectively. The parameter *k* represents the *kth* message m^k and τ denotes the current time instant. With the knowledge of the CD and ICT values, we now compute the aggregate utility $W_{i,d}^k(\tau)$ for m^k, given as:

$$W_{i,d}^k(\tau) = \frac{C_{i,d}^k(\tau)}{Z_{i,d}^k(\tau)}. \tag{9}$$

The utility $W_{i,d}^k(\tau)$ in the above equation is a measure of how good a candidate a node *i* is, in terms of successfully delivering m^k to *d* before the life-time expiry. The higher the forecasted CD value $C_{i,d}^k(\tau)$ and the lower the forecasted ICT value $Z_{i,d}^k(\tau)$ between nodes *i* and *d*, the better are the chances of m^k being delivered to *d* by *i*. During an opportunistic contact between *i* and a relayer *r*, the *UBS* computes the difference of $W_{i,d}^k(\tau)$ and $W_{r,d}^k(\tau)$ for a message m^k and subsequently, for all of the messages in the ML of *i*. If M is the set of messages in the ML of *i*, the notation $\left(W_{i,d}^k(\tau) - W_{r,d}^k(\tau)\right)$, $\forall\, m^k \in M$ depicts the aforesaid computation. With the obtained differences of the aggregate utilities of *i* and *r* for each of the messages, the ML of *i* is reordered in an ascending order, such that the message for which the difference value is the least is moved to the top of the list. Therefore, the messages are reordered according to progressively decreasing probabilities of being delivered by *r*. Subsequently, each of the messages that are not in the PML of *r* is relayed. Evidently, the *UBS* performs flooding by prioritizing the messages and does not implement the conditional replication or deletion, as performed by the *HSM*. The number of prediction-based computations performed on a node remains the same, as we still use the CD- and ICT-based data. The benefit of eliminating a design limitation outclasses the cost of flooding, as indicated in Chapter 6.

5. MODELING, ANALYSIS, AND VERIFICATION

5.1. Overview of Communication Processes

In this chapter, we present the communication models of the *HSM*, *FAR*, and *UBS*, and a discussion on the verification of the three schemes. When two nodes come in the communication range of each other, the node that initiates the connection (Connection Initiator or CI) incepts the communication with the Connected Node (CN). The nodes exchange the information about the contents of their MLs, RLs, and ALs, to figure out the messages that can be exchanged. Firstly, the CI transfers the messages that are destined for the CN. The messages destined for the CI are then transferred by the CN. Subsequently, based on the prediction-based computations, the CI transfers the messages that can be relayed by the CN to the destinations of the messages. The CN then transfers the messages that can be relayed by the CI. However, these simple processes entail the consideration of several factors for the modeling and verification, as explained below.

5.2. Modeling and Analysis of *HSM*

The HLPN model of the *HSM* is illustrated in Fig. 3. As previously mentioned, a HLPN is a 7-tuple of the form $N = (P, T, F, \varphi, R, L, M_0)$. The nine places depicted in Fig. 3 constitute the set P. The names of the places and the corresponding mappings (φ) to the tokens or data of various data types, are shown in Table 1. The data types are described in Table 2. The acronyms considered in the tables in this chapter are: Acknowledgment (ACK), Communicating Nodes (Com-N), Destination (D), and with respect to (w.r.t). The set of transitions can be denoted as: $T = \{Ready, C\_F, C\_S\_D, C\_CD, C\_ICT, Rel\}$. The set of arcs F (flow relation) and the corresponding labels (L) are shown in Fig. 3. The initial marking (M_0) is simply the tokens of different data types placed at P, as shown in Table 1.

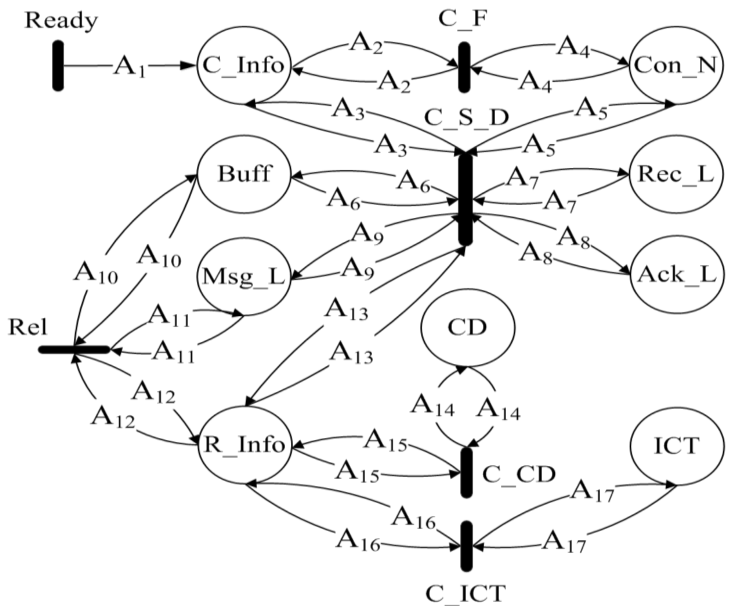


Fig. 3. HLPN model of the *HSM*.

Table 1. Places and Mappings of *HSM*.

Places	Mappings	Descriptions
φ (C_Info)	ℙ ($N1 \times M_{SIZE} \times M_D \times M_{ID} \times M_{TTL\_F} \times M_{ACK\_F}$)	Contains N_ID of CI, messages in its ML represented by key attributes, ACK flag.
φ (Con_N)	ℙ (Node_L)	Contains the N_IDs of CNs.
φ (Buff)	ℙ (Avlbl_B)	Contains available space, in message buffers of Com-N.
φ (Rec_L)	ℙ (M_{ID_D})	Holds the RLs of Com-N.
φ (Ack_L)	ℙ (M_{ID_A})	Holds the ALs of Com-N.
φ (Msg_L)	ℙ (M_{ID_R})	Holds the MLs of Com-N.
φ (R_Info)	ℙ ($N1 \times N2 \times M_D \times U_{CD\_N1\_D} \times U_{CD\_N2\_D} \times U_{ICT\_N1\_D} \times U_{ICT\_N2\_D} \times M_{ID} \times M_{TTL\_F} \times M_{SIZE} \times M_{ACK\_F}$)	Contains the N_IDs of Com-N, N_IDs of Ds of messages to be relayed by CI, utility values of Com-N based on CD and ICT, messages to be relayed in the ML of CI represented by key attributes, and ACK flag.
φ (CD)	ℙ ($N1 \times N2 \times M_D \times U_{CD\_N1\_D} \times U_{CD\_N2\_D}$)	Records the N_IDs of Com-N, N_IDs of Ds of messages to be relayed by CI, and CD-based utilities of Com-N.
φ (ICT)	ℙ ($N1 \times N2 \times M_D \times U_{ICT\_N1\_D} \times U_{ICT\_N2\_D}$)	Records the N_IDs of Com-N, N_IDs of Ds of messages to be relayed by CI, ICT-based utilities of Com-N.

Table 2. Data Types used in the model of *HSM*.

Types	Descriptions
N1	Integer type, for the N_ID of the CI.
M_{SIZE}	Float type, for the size of a message.
M_D	Integer type, for the N_ID of the D of a message.
M_{ID}	Integer type, for the M_ID of a message.
M_{TTL_F}	Boolean type TTL flag that is TRUE throughout the life-time of a message, is FALSE on life-time expiry.
M_{ACK_F}	Boolean type ACK flag that sets to TRUE on the successful ACK of a message, is FALSE otherwise.
Node_L	Integer array type, for the N_IDs of the CNs.
Avlbl_B	Float type, for available space, in a node's message buffer.
M_{ID_D}	Integer array type, for the M_IDs, in a node's RL.
M_{ID_A}	Integer array type, for the M_IDs, in a node's AL.
M_{ID_R}	Integer array type, for the M_IDs of messages, in a node's ML.
N2	Integer type, for the N_ID of the CN.
$U_{CD\_N1\_D}$	Float type, for the utility value of the CI w.r.t the D of a message, based on the forecasted CD.
$U_{CD\_N2\_D}$	Float type, for the utility value of the CN w.r.t the D of a message, based on the forecasted CD.
$U_{ICT\_N1\_D}$	Float type, for the utility value of the CI w.r.t the D of a message, based on the forecasted ICT.
$U_{ICT\_N2\_D}$	Float type, for the utility value of the CN w.r.t the D of a message, based on the forecasted ICT.

A brief overview of the communication processes has been presented in Section 5.1. We now define the pre- and post-conditions to be mapped to T, termed as formulae or rules (R) for the processes. The firing of the initial transition $Ready$, acting as an inlet for new tokens or data, enables the communication processes in the model. Our model does not require a specific condition to operate or create new data. Therefore, the rule for the transition $Ready$ is stated as: $\boldsymbol{R}\ (Ready) = \exists t \in T \mid \bullet t = \emptyset$. The transition C_F is mapped to the following formula.

$$\boldsymbol{R}\ (C\_F)\ =\ \forall\, a_2 \in A_2, \forall\, a_4 \in A_4 \mid \qquad (10)$$

$$a_4\ :=\ Nil.$$

Equation (10) indicates that C_F is fired only when the CI (placed at C_Info) does not find a CN. Therefore, no communication takes place. The next transition C_S_D is fired when the CI successfully finds one or more nodes (the CNs placed in Con_N) to be in the transmission range. The CI initiates the communication with the node that is the first to be available in the transmission range (CN). C_Info contains the attributes of the messages (size, destination, M_ID, and TTL) in the ML of the CI. The next processes are exchange of network information and messages that are destined for the communicating nodes (11).

$$
\begin{gathered}
\boldsymbol{R}\,(C\_S\_D) \;=\; \forall\, a_3 \in A_3, \forall\, a_5 \in A_5, \\
\forall\, a_6 \in A_6, \forall\, a_7 \in A_7, \forall\, a_8 \in A_8, \\
\forall\, a_{91} \in A_9 \mid \\
a_5 \;\neq\; Nil \;\wedge\; conn(a_3[1], a_5) \;=\; TRUE \;\wedge \\
a_3[5] \;=\; TRUE \;\wedge\; a_{91} \;:=\; M\_L(a_3[1]) \;\wedge \\
a_6 \;:=\; Avlbl\_Buff(a_3[1]) \;\wedge \\
a_7 \;:=\; Rcvd\_L(a_5) \;\wedge\; a_8 \;:=\; Ackn\_L(a_5) \;\wedge \\
(a_{91} \;=\; a_7 \;\vee\; a_{91} \;=\; a_8) \;\wedge \\
A_9' \;:=\; A_9 \setminus \{(a_{91})\} \;\wedge \\
a_6 \;:=\; a_6 \;+\; Size(a_{91}) \;\wedge \\
A_6' \;:=\; A_6 \;\cup\; \{(a_6)\} \;\wedge \\
\forall\, a_{61} \in A_6, \forall\, a_{71} \in A_7, \forall\, a_{81} \in A_8, \forall\, a_{92} \in A_9, \\
\forall\, a_{13} \in A_{13} \mid \\
a_{61} \;:=\; Avlbl\_Buff(a_3[1]) \;\wedge \\
a_{71} \;:=\; Rcvd\_L(a_5) \;\wedge\; a_{81} \;:=\; Ackn\_L(a_3[1]) \;\wedge
\end{gathered} \tag{11}
$$

$$a_{92} := M\_L(a_3[1]) \wedge a_3[3] = a_5 \wedge$$

$$a_3[5] = TRUE \wedge$$

$$a_{71} := a_{71} \cup \{(a_3[4])\} \wedge a_3[6] := TRUE \wedge$$

$$a_{81} := a_{81} \cup \{(a_3[4])\} \wedge$$

$$a_{92} := a_{92} \setminus \{(a_3[4])\} \wedge a_{61} := a_{61} + (a_3[2]) \wedge$$

$$A'_6 := A_6 \cup \{(a_{61})\} \wedge A'_7 := A_7 \cup \{(a_{71})\} \wedge$$

$$A'_8 := A_8 \cup \{(a_{81})\} \wedge A'_9 := A_9 \cup \{(a_{92})\} \wedge$$

$$a_3[6] := FALSE \wedge a_{13}[1] := a_3[1] \wedge$$

$$a_{13}[2] := a_5 \wedge a_{13}[3] := a_3[3] \wedge$$

$$a_{13}[8] := a_3[4] \wedge a_{13}[9] := a_3[5] \wedge$$

$$a_{13}[10] := a_3[2] \wedge a_{13}[11] := a_3[6] \wedge$$

$$A'_{13} := A_{13} \cup \{(a_{13}[1], a_{13}[2], a_{13}[3], a_{13}[4], a_{13}[5],$$

$$a_{13}[6], a_{13}[7], a_{13}[8], a_{13}[9], a_{13}[10], a_{13}[11])\}.$$

In the above formula, the connectivity between the CI and CN is checked through the function $conn$ and information is exchanged for a successful connection. The functions M_L, $Rcvd\_L$, and $Ackn\_L$, provide access to the MLs, RLs, and ALs of the communicating nodes that are placed at Msg_L, Rec_L, and Ack_L, respectively. The function $Avlbl\_Buff$ furnishes the available space in the buffers of the communicating nodes from $Buff$. In (11), the CI deletes those M_IDs from the ML whose life-times did not expire and that are in the: **(a)** RL of the CN and **(b)** AL of the CN. The function $Size$ returns the total size of the messages deleted through **(a)** and **(b)**. The buffer space is updated accordingly. Finally, the CI sends those messages from its ML that are destined for the CN. The CN adds the M_IDs of the incoming messages to its RL.

On receiving the ACKs, the CI adds the M_IDs of the sent messages to its AL, removes them from the ML, and updates the buffer space accordingly. Subsequently, the CN will execute the same process (11), on the CI. The exchange of the messages that are destined for the two nodes is complete. The N_IDs of the two nodes and the attributes of the remaining messages in the ML of the CI are passed to R_Info to commence the replication, in (11). The replications require the computation of the CD- and ICT-based utilities. Therefore, C_CD and C_ICT are fired.

$$\begin{gathered}
\boldsymbol{R}\ (C_CD)\ =\ \forall\, a_{14} \in A_{14}, \forall\, a_{15} \in A_{15}\ | \\
a_{14}[1]\ :=\ a_{15}[1]\ \wedge\ a_{14}[2]\ :=\ a_{15}[2]\ \wedge \\
a_{14}[3]\ :=\ a_{15}[3]\ \wedge \\
a_{14}[4]\ :=\ Comp_CD(a_{14}[1], a_{14}[3])\ \wedge \\
a_{14}[5]\ :=\ Comp_CD(a_{14}[2], a_{14}[3])\ \wedge \\
a_{15}[4]\ :=\ Comp_CD(a_{15}[1], a_{15}[3])\ \wedge \\
a_{15}[5]\ :=\ Comp_CD(a_{15}[2], a_{15}[3])\ \wedge \\
A'_{15}\ :=\ A_{15}\ \cup\ \{(a_{15}[1], a_{15}[2], a_{15}[3], a_{15}[4], a_{15}[5], \\
a_{15}[6], a_{15}[7], a_{15}[8], a_{15}[9], a_{15}[10], a_{15}[11])\}\ \wedge \\
A'_{14}\ :=\ A_{14}\ \cup\ \{(a_{14}[1], a_{14}[2], a_{14}[3], a_{14}[4], a_{14}[5])\}.
\end{gathered} \tag{12}$$

$$\begin{gathered}
\boldsymbol{R}\ (C_ICT)\ =\ \forall\, a_{16} \in A_{16}, \forall\, a_{17} \in A_{17}\ | \\
a_{17}[1]\ :=\ a_{16}[1]\ \wedge\ a_{17}[2]\ :=\ a_{16}[2]\ \wedge \\
a_{17}[3]\ :=\ a_{16}[3]\ \wedge \\
a_{17}[4]\ :=\ Comp_ICT(a_{17}[1], a_{17}[3])\ \wedge \\
a_{17}[5]\ :=\ Comp_ICT(a_{17}[2], a_{17}[3])\ \wedge
\end{gathered} \tag{13}$$

$$a_{16}[6] := Comp\_ICT(a_{16}[1], a_{16}[3]) \wedge$$

$$a_{16}[7] := Comp\_ICT(a_{16}[2], a_{16}[3]) \wedge$$

$$A'_{16} := A_{16} \cup \{(a_{16}[1], a_{16}[2], a_{16}[3], a_{16}[4], a_{16}[5],$$

$$a_{16}[6], a_{16}[7], a_{16}[8], a_{16}[9], a_{16}[10], a_{16}[11])\} \wedge$$

$$A'_{17} := A_{17} \cup \{(a_{17}[1], a_{17}[2], a_{17}[3], a_{17}[4], a_{17}[5])\}.$$

In (12) and (13), the information on the nodes and messages is extracted to CD and ICT from R_Info, and the functions $Comp\_CD$ and $Comp\_ICT$ are used to compute the CD- and ICT-based utilities, as discussed in Chapter 4. The computed utilities of the communicating nodes with the destinations of the remaining messages in the ML of the CI are recorded in CD and ICT. The messages are processed one by one and the utilities are made available at R_Info. Rel is fired to initiate the replication.

The replication process is instantiated in (14), aided by the information on the communicating nodes, messages, and the utilities, made available at R_Info through (11), (12), and (13). The MLs and buffers of the communicating nodes are accessed from Msg_L and $Buff$, through the functions M_L and $Avlbl\_Buff$, respectively. The function $Buff\_Size$ is used to retrieve the buffer sizes of the nodes.

In (14), each of the remaining messages in the ML of the CI whose life-times did not expire and that are not in the ML of the CN are processed for replication. Only the messages that are smaller in size, as compared to the buffer size and the available buffer space of the CN, are processed. A message is conditionally replicated on the CN if and only if, the: **(a)** CD-based utility of the CN is higher than that of the CI or **(b)** if both of the utilities (CD- and ICT-based) of the CN are higher than that of the CI, as discussed in Chapter 4. The corresponding M_ID is added to the ML of the CN and after the buffer space is updated, an ACK is sent to the CI. For the point **(b)**, on receiving the ACK, the M_ID is conditionally deleted from the ML of the CI, and its buffer space is updated accordingly. If the CN cannot accommodate an incoming message due to the lack of space in the buffer, the function $Del\_Old\_Msgs\_Until\_Spc$ is used to delete the older messages to create space. The message is accommodated after the space is created in the buffer. Thereafter, the CN executes the process (14), on the CI.

$$
\begin{aligned}
\boldsymbol{R}\ (Rel)\ =\ & \forall\, a_{10} \in A_{10}, \forall\, a_{110} \in A_{11}, \\
& \forall\, a_{111} \in A_{11}, \forall\, a_{12} \in A_{12} \mid \\
& a_{110}\ :=\ M\_L(a_{12}[1])\ \wedge\ a_{111}\ :=\ M\_L(a_{12}[2])\ \wedge \\
& a_{110}\ \neq\ a_{111}\ \wedge\ a_{12}[9]\ =\ TRUE\ \wedge \\
& a_{12}[10]\ \leq\ Buff\_Size(a_{12}[2])\ \wedge \\
& a_{12}[10]\ \leq\ Avlbl\_Buff(a_{12}[2])\ \wedge \\
& a_{12}[5]\ >\ a_{12}[4]\ \wedge\ a_{10}\ :=\ Avlbl\_Buff(a_{12}[2])\ \wedge \\
& a_{10}\ :=\ a_{10} - a_{12}[10]\ \wedge \\
& a_{111}\ :=\ a_{111}\ \cup\ \{(a_{12}[8])\}\ \wedge\ a_{12}[11] := TRUE\ \wedge \\
& A'_{11}\ :=\ A_{11}\ \cup\ \{(a_{111})\}\ \wedge\ A'_{10}\ :=\ A_{10}\ \cup\ \{(a_{10})\}\ \wedge \\
& a_{12}[11] := FALSE\ \wedge \\
& \forall\, a_{101} \in A_{10}, \forall\, a_{102} \in A_{10}, \forall\, a_{112} \in A_{11}, \forall\, a_{113} \in A_{11} \mid \\
& a_{112}\ :=\ M\_L(a_{12}[1])\ \wedge\ a_{113}\ :=\ M\_L(a_{12}[2])\ \wedge \\
& a_{112}\ \neq\ a_{113}\ \wedge\ a_{12}[9]\ =\ TRUE\ \wedge \\
& a_{12}[10]\ \leq\ Buff\_Size(a_{12}[2])\ \wedge \\
& a_{12}[10]\ \leq\ Avlbl\_Buff(a_{12}[2])\ \wedge \\
& a_{12}[5]\ >\ a_{12}[4]\ \wedge\ a_{12}[7]\ >\ a_{12}[6]\ \wedge
\end{aligned}
\qquad (14)
$$

$$a_{101} := Avlbl\_Buff(a_{12}[1]) \wedge$$

$$a_{102} := Avlbl\_Buff(a_{12}[2]) \wedge$$

$$a_{102} := a_{102} - (a_{12}[10]) \wedge$$

$$a_{113} := a_{113} \cup \{(a_{12}[8])\} \wedge a_{12}[11] := TRUE \wedge$$

$$a_{101} := a_{101} + (a_{12}[10]) \wedge$$

$$a_{112} := a_{112} \setminus \{(a_{12}[8])\} \wedge$$

$$A'_{11} := A_{11} \cup \{(a_{112})\} \wedge A'_{11} := A_{11} \cup \{(a_{113})\} \wedge$$

$$A'_{10} := A_{10} \cup \{(a_{101})\} \wedge A'_{10} := A_{10} \cup \{(a_{102})\} \wedge$$

$$a_{12}[11] := FALSE \wedge$$

$$\forall a_{103} \in A_{10}, \forall a_{104} \in A_{10}, \forall a_{114} \in A_{11}, \forall a_{115} \in A_{11} \mid$$

$$a_{114} := M\_L(a_{12}[1]) \wedge a_{115} := M\_L(a_{12}[2]) \wedge$$

$$a_{114} \neq a_{115} \wedge a_{12}[9] = TRUE \wedge$$

$$a_{12}[10] \leq Buff\_Size(a_{12}[2]) \wedge$$

$$a_{12}[10] > Avlbl\_Buff(a_{12}[2]) \wedge$$

$$\{(a_{12}[5] > a_{12}[4]) \vee$$

$$(a_{12}[5] > a_{12}[4] \wedge a_{12}[7] > a_{12}[6])\} \wedge$$

$$a_{103} := Avlbl\_Buff(a_{12}[1]) \wedge$$

$$a_{115} := Del\_Old\_Msgs\_Until\_Spc(a_{115}) \wedge$$

$$a_{104} := Avlbl\_Buff(a_{12}[2]) \wedge$$

$$A'_{11} := A_{11} \cup \{(a_{115})\}.$$

After the communication between the CI and CN is complete, the CI looks for the availability of the remaining CNs in Con_N. The same procedures are instantiated with each one of the CNs. All of the abovementioned processes will be repeated for each of the CIs in the cloud (through the presented formulae). Therefore, the model captures the behavior of all of the nodes in the cloud. Evidently, the model does not check if an incoming message to a node had been previously replicated by the node.

5.3. Modeling and Analysis of *FAR*

The HLPN model of the *FAR* is depicted in Fig. 4. Evidently, the model resembles the model of the *HSM*. The only notable difference is that the ICT-based place (ICT) and transition (C_ICT) are not considered in the model of the *FAR*. The reason is that the replications in the *FAR* are solely based on the CD, as discussed in Chapter 4. The places and the corresponding mappings to the data types are shown in Table 3 and Table 4, respectively. Table 3 and Table 4 entail only the mappings and data types that have been modified from the contents of Table 1 and Table 2. The modification is simply the exclusion of the ICT-based data and the inclusion of the CD-based meeting quality values (against the CD-based utility), as compared to the model of the *HSM*. Apart from the acronyms declared previously, the Meeting Quality (MQ) is the only acronym introduced in Table 3 and Table 4. The contents and mappings for the rest of the places, alongside the flow and labels, remain the same as that of the *HSM*. The set of transitions is given as: $T = \{Ready, C\_F, C\_S\_D, C\_CD, Rel\}$.

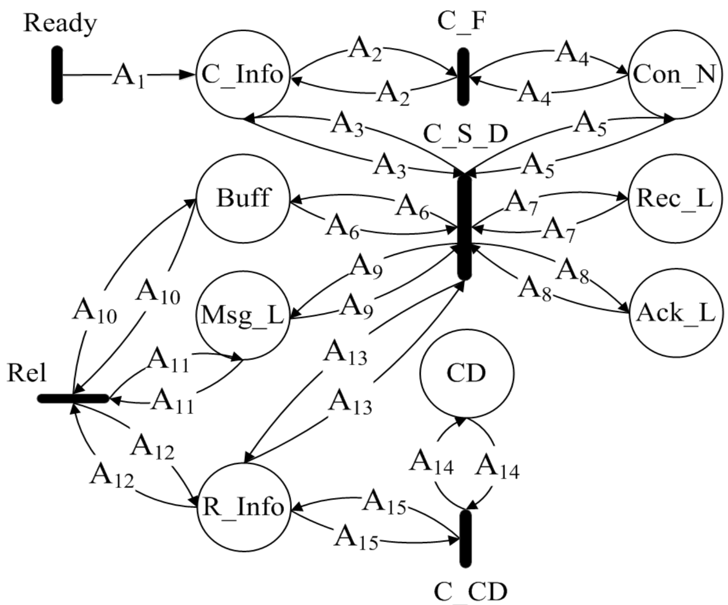


Fig. 4. HLPN model of the *FAR*.

Table 3. Places and Mappings of *FAR*.

Places	Mappings	Descriptions
φ (R_Info)	ℙ ($N1 \times N2 \times M_D \times F_{CD\_N1\_D} \times F_{CD\_N2\_D} \times M_{ID} \times M_{TTL\_F} \times M_{SIZE} \times M_{ACK\_F}$)	Same as that of Table 1, except for the inclusion of the CD-based current MQ values of Com-N instead of CD- and ICT-based utilities.
φ (CD)	ℙ ($N1 \times N2 \times M_D \times F_{CD\_N1\_D} \times F_{CD\_N2\_D}$)	Same as Table 1, only CD-based utilities are replaced by current MQs of Com-N.

Table 4. Data Types used in the model of *FAR*.

Types	Descriptions
$F_{CD\_N1\_D}$	Float type, for the forecasted MQ value of the CI w.r.t the D of a message, based on the CD.
$F_{CD\_N2\_D}$	Float type, for the forecasted MQ value of the CN w.r.t the D of a message, based on the CD.

Apart from Rel, the functionality of the pre- and post-conditions (formulae) for the rest of the transitions remain the same as discussed in Section 5.2, and have not been shown. The functions used in the model remain the same as the ones used for the *HSM*. The only difference is that the transition C_CD employs the function FMQ_CD to compute the forecasts of the meeting qualities of the communicating nodes. The function $Comp\_CD$ had been used for the *HSM* to compute the CD-based utility. Therefore, apart from the replication process, the behavior of the model is exactly the same as that of the *HSM* that allows us to move on to the replication process Rel (15). The information on the communicating nodes, messages, and forecasted meeting qualities, are passed to R_Info, as discussed in Section 5.2. Subsequently, *Rel* is fired.

$$
\begin{gathered}
\boldsymbol{R}\ (Rel)\ =\ \forall\, a_{10} \in A_{10}, \forall\, a_{110} \in A_{11}, \qquad (15)\\
\forall\, a_{111} \in A_{11}, \forall\, a_{12} \in A_{12} \mid \\
a_{110}\ :=\ M\_L(a_{12}[1])\ \wedge\ a_{111}\ :=\ M\_L(a_{12}[2])\ \wedge \\
a_{110}\ \neq\ a_{111}\ \wedge\ a_{12}[7]\ =\ TRUE\ \wedge \\
a_{12}[8]\ \leq\ Buff\_Size(a_{12}[2])\ \wedge \\
a_{12}[8]\ \leq\ Avlbl\_Buff(a_{12}[2])\ \wedge \\
a_{12}[5]\ >\ a_{12}[4]\ \wedge\ a_{10}\ :=\ Avlbl\_Buff(a_{12}[2])\ \wedge \\
a_{10}\ :=\ a_{10} - a_{12}[8]\ \wedge \\
a_{111}\ :=\ a_{111}\ \cup\ \{(a_{12}[6])\}\ \wedge\ a_{12}[9] := TRUE\ \wedge \\
A'_{11}\ :=\ A_{11}\ \cup\ \{(a_{111})\}\ \wedge\ A'_{10}\ :=\ A_{10}\ \cup\ \{(a_{10})\}\ \wedge \\
a_{12}[9] := FALSE\ \wedge \\
\forall\, a_{100} \in A_{10}, \forall\, a_{112} \in A_{11}, \forall\, a_{113} \in A_{11} \mid \\
a_{112}\ :=\ M\_L(a_{12}[1])\ \wedge\ a_{113}\ :=\ M\_L(a_{12}[2])\ \wedge \\
a_{112}\ \neq\ a_{113}\ \wedge\ a_{12}[7]\ =\ TRUE\ \wedge \\
a_{12}[8]\ \leq\ Buff\_Size(a_{12}[2])\ \wedge \\
a_{12}[8]\ >\ Avlbl\_Buff(a_{12}[2])\ \wedge\ a_{12}[5]\ >\ a_{12}[4]\ \wedge
\end{gathered}
$$

$$a_{113} := Del\_Old\_Msgs\_Until\_Spc(a_{113}) \wedge$$

$$a_{100} := Avlbl\_Buff(a_{12}[2]) \wedge$$

$$A'_{11} := A_{11} \cup \{(a_{113})\}.$$

In the above equation, the only differences as compared to the replication process of the *HSM* in (14) are: **(a)** a message is replicated by the CI on the CN, if the CN bears a better forecasted meeting quality (CD-based) with the destination of the message, instead of the CD- and ICT-based utilities in (14) and **(b)** there is no conditional deletion. The rest of the process is the same as (14). Again, the model does not check if an incoming message to a node had been previously replicated by the node.

5.4. Modeling and Analysis of *UBS*

The HLPN model of the *UBS* (an extension of the *HSM*) is exhibited in Fig. 5. The inclusions of the place PML containing the PMLs of the communicating nodes and a sorted list of aggregate utilities of the communicating nodes are the modifications made to the model of the *HSM*. We use the mean CD and ICT values from the *HSM* computed in Chapter 4, instead of the CD- and ICT-based utilities. The modifications improve the design limitation discussed in Chapter 4 and facilitate the flooding-based routing. The design limitation in the *HSM* and *FAR* is that a node accepts the same message that it has previously relayed in the network and eventually deleted after one or more replications.

The places and their data type mappings that are modified from or added to the contents of Table 1 and Table 2, are shown in Table 5 and Table 6, respectively. The Aggregate Utility (AU) has been added to the list of acronyms declared in Section 5.2. The contents and mappings for the rest of the places, alongside the flow and labels, remain the same as that of the *HSM*. The set of transitions can be written as: $T = \{Ready, C\_F, C\_S\_D, C\_CD, C\_ICT, Rel\}$. Being an extension of the *HSM*, the basic communication processes and the transitions in the *UBS* remain the same. Apart from Rel, which is the replication process, the functionality of the formulae to the rest of the transitions are the same as that of the *HSM*.

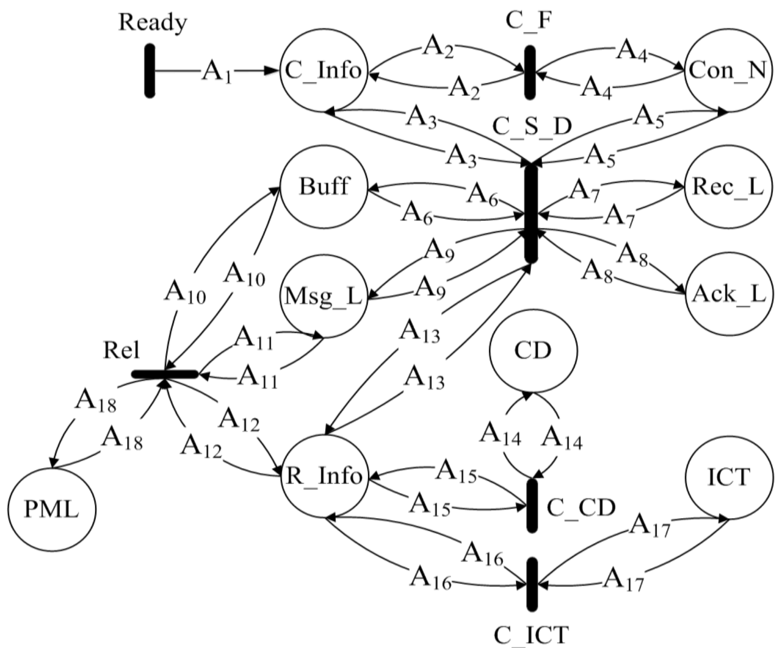


Fig. 5. HLPN model of the *UBS*.

Table 5. Places and Mappings of *UBS*.

Places	Mappings	Descriptions
φ (R_Info)	$\mathbb{P}$ $(N1\times N2\times M_{D}\times M_{CD\_N1\_D}\times M_{CD\_N2\_D}\times M_{ICT\_N1\_D}\times M_{ICT\_N2\_D}\times SL_{AU\_N1\_N2\_D}\times M_{ID}\times M_{TTL\_F}\times M_{SIZE}\times M_{ACK\_F})$	Same as Table 1, only CD- and ICT-based utilities of Com-N are replaced with mean CDs and ICTs and a sorted list of subtracted AUs of Com-N is introduced.
φ (CD)	$\mathbb{P}$ $(N1\times N2\times M_{D}\times M_{CD\_N1\_D}\times M_{CD\_N2\_D})$	Same as Table 1, mean CDs replace CD-based utilities.
φ (ICT)	$\mathbb{P}$ $(N1\times N2\times M_{D}\times M_{ICT\_N1\_D}\times M_{ICT\_N2\_D})$	Same as Table 1, mean ICTs replace ICT-based utilities.
φ (PML)	$\mathbb{P}$ (M_{ID_P})	Holds the PML of a node.

Table 6. Data Types used in the model of *UBS*.

Types	Descriptions
$M_{CD\_N1\_D}$	Float type, for mean CD of CI w.r.t a message's D.
$M_{CD\_N2\_D}$	Float type, for mean CD of CN w.r.t a message's D.
$M_{ICT\_N1\_D}$	Float type, for mean ICT of CI w.r.t a message's D.
$M_{ICT\_N2\_D}$	Float type, for mean ICT of CN w.r.t a message's D.
$SL_{AU\_N1\_N2\_D}$	Float array type, for sorted (in ascending order) list of values, obtained after subtracting AUs of CN w.r.t Ds of messages in CI's ML, from those of CI with the same Ds.
M_{ID_P}	Integer array type, for the M_IDs, in a node's PML.

In terms of the replication process Rel, the only difference from the *HSM* lies in the flooding-based replication and the incorporation of the PML. The functions used in the model remain the same as the ones used for the *HSM*. The functions $Comp\_CD$ and $Comp\_ICT$ are modified to compute the mean CD and ICT values, instead of the CD- and ICT-based utilities of the *HSM*. As discussed in Section 5.2, the information on the communicating nodes, messages, and the mean CDs and ICTs, are made available at R_Info and Rel is fired to initiate the replications, as indicated in (16). In (16), the subtracted aggregate utility values of the CI and CN for each of the messages in the ML of the CI, are the inputs to the function AU_SSA. The function sorts the values in an ascending order and stores them at R_Info.

The function PM_L provides access to the PMLs of the communicating nodes, placed at PML. The function $Update\_N1\_ML$ (reorders the ML of the CI according to the sorted aggregate utilities) is introduced in the model. In (16), the only differences as compared to the replication process of the *HSM* in (14) are: **(a)** messages are processed from the reordered ML of the CI and only the messages that are not in the ML and PML of the CN are considered for replication, **(b)** after relaying a message successfully, the CI adds the corresponding M_ID to its PML, if it was not already included, and **(c)** all of the messages referred to in **(a)** are replicated and there is no conditional replication or deletion. The rest of the process is the same as (14).

$$\boldsymbol{R}\,(Rel) \;=\; \forall\, a_{10} \in A_{10}, \forall\, a_{110} \in A_{11}, \tag{16}$$

$$\forall\, a_{111} \in A_{11}, \forall\, a_{180} \in A_{18}, \forall\, a_{181} \in A_{18},$$

$$\forall\, a_{12} \in A_{12} \mid$$

$$a_{110} \;:=\; M\_L(a_{12}[1]) \;\wedge\; a_{111} \;:=\; M\_L(a_{12}[2]) \;\wedge$$

$$a_{180} \;:=\; PM\_L(a_{12}[1]) \;\wedge\; a_{181} \;:=\; PM\_L(a_{12}[2]) \;\wedge$$

$$a_{12}[8] \;:=\; AU\_SSA(a_{12}[4]/a_{12}[6] - a_{12}[5]/a_{12}[7])$$

$$\wedge\; Update\_N1\_ML(a_{12}[8], a_{110}) \;\wedge$$

$$A'_{11} \;:=\; A_{11} \;\cup\; \{(a_{110})\} \;\wedge$$

$$A'_{12} \;:=$$

$$A_{12} \;\cup\; \{(a_{12}[1], a_{12}[2], a_{12}[3], a_{12}[4], a_{12}[5], a_{12}[6],$$

$$a_{12}[7], a_{12}[8], a_{12}[9], a_{12}[10], a_{12}[11], a_{12}[12])\} \;\wedge$$

$$a_{110} \;\neq\; a_{111} \;\wedge\; a_{110} \;\neq\; a_{181} \;\wedge$$

$$a_{12}[10] \;=\; TRUE \;\wedge$$

$$a_{12}[11] \;\leq\; Buff\_Size(a_{12}[2]) \;\wedge$$

$$a_{12}[11] \;\leq\; Avlbl\_Buff(a_{12}[2]) \;\wedge$$

$$a_{10} \;:=\; Avlbl\_Buff(a_{12}[2]) \;\wedge$$

$$a_{10} \;:=\; a_{10} - a_{12}[11] \;\wedge$$

$$a_{111} \;:=\; a_{111} \;\cup\; \{(a_{12}[9])\} \;\wedge\; a_{12}[12] := TRUE \;\wedge$$

$$\{(a_{180} \;\neq\; a_{12}[9] \;\wedge\; a_{180} \;:=\; a_{180} \;\cup\; \{(a_{12}[9])\} \;\wedge$$

$$A'_{18} \;:=\; A_{18} \;\cup\; \{(a_{180})\}) \;\vee$$

$$(a_{180} \;=\; a_{12}[9])\} \;\wedge$$

$$A'_{11} \;:=\; A_{11} \;\cup\; \{(a_{111})\} \;\wedge\; A'_{10} \;:=\; A_{10} \;\cup\; \{(a_{10})\} \;\wedge$$

$$a_{12}[12] := FALSE \;\wedge$$

$$\forall\, a_{100} \in A_{10}, \forall\, a_{112} \in A_{11}, \forall\, a_{113} \in A_{11},$$

$$\forall\, a_{182} \in A_{18}, \forall\, a_{183} \in A_{18} \mid$$

$$a_{112} \ := \ M\_L(a_{12}[1]) \ \wedge \ a_{113} \ := \ M\_L(a_{12}[2]) \ \wedge$$

$$a_{182} \ := \ PM\_L(a_{12}[1]) \ \wedge \ a_{183} \ := \ PM\_L(a_{12}[2]) \ \wedge$$

$$a_{12}[8] \ := \ AU\_SSA(a_{12}[4]/a_{12}[6] - a_{12}[5]/a_{12}[7])$$

$$\wedge \ Update\_N1\_ML(a_{12}[8], a_{112}) \ \wedge$$

$$A'_{11} \ := \ A_{11} \ \cup \ \{(a_{112})\} \ \wedge$$

$$A'_{12} \ :=$$

$$A_{12} \ \cup \ \{(a_{12}[1], a_{12}[2], a_{12}[3], a_{12}[4], a_{12}[5], a_{12}[6],$$

$$a_{12}[7], a_{12}[8], a_{12}[9], a_{12}[10], a_{12}[11], a_{12}[12])\} \ \wedge$$

$$a_{112} \ \neq \ a_{113} \ \wedge \ a_{112} \ \neq \ a_{183} \ \wedge$$

$$a_{12}[10] \ = \ TRUE \ \wedge$$

$$a_{12}[11] \ \leq \ Buff\_Size(a_{12}[2]) \ \wedge$$

$$a_{12}[11] \ > \ Avlbl\_Buff(a_{12}[2]) \ \wedge$$

$$a_{113} \ := \ Del\_Old\_Msgs\_Until\_Spc(a_{113}) \ \wedge$$

$$a_{100} \ := \ Avlbl\_Buff(a_{12}[2]) \ \wedge$$

$$A'_{11} \ := \ A_{11} \ \cup \ \{(a_{113})\}.$$

5.5. Verification of *HSM*, *FAR*, and *UBS*

Formal verification is a methodical procedure that incorporates mathematical reasoning for the development, specification, and verification of the correctness of systems [20]. Model checking is a verification technique, used to verify the properties of a system. The process encompasses an exhaustive search of all of the possible states that the system may enter during the execution [20]. The process comprises of the: **(a)** specification of the system properties, **(b)** system modeling, and **(c)** verification of the specifications, using tools, such as NuSMV.

Definition 2 (Model Checking) [21]. Formally stated, given a Kripke structure of the form $M = (S, I, R, L)$ and a temporal logic formula φ, the model checking problem is to find the set of states satisfying φ, given as: $\{s \in S \mid M, s \vDash \varphi\}$, where: S is a finite set of states, I is the set specifying the initial states, $R \subseteq S \times S$ is a transition relation used to specify the possible state-to-state transitions, and L is a labeling function for labeling the states with atomic propositions.

Definition 3 (NuSMV Model) [24]. A NuSMV model is a Kripke structure of the form $M = (S, I, R)$, where: each of the states of S can be labeled by a predicate $\bigwedge_{i=1}^{k}(v_i = d_i)$, the finite set $var(M) = \{v_1, \dots, v_k\}$ represents the set of state variables, with the set $\{d_1, \dots, d_k\}$ representing their interpreted values over the domain $\{D_1, \dots, D_k\}$, and R is the transition relation that updates the state variable interpretation.

The NuSMV language flexibly describes the transition relation of a finite Kripke structure [24] using the propositional calculus [23]. State variables with certain domains are used to depict the behavior of a NuSMV model. Each of the states in the model corresponds to an assignment of values to the state variables [24]. The NuSMV transforms the FSM of the system under verification to a BDD. A Boolean formula and its BDD are a compact depiction of the set of states that satisfy the formula. The transition relation for the Kripke structure may be represented by a Boolean formula and consequently the BDD, comprising of the current and next state variables [24]. A temporal logic, such as the CTL that has been used in this work for property specifications, is used to express the behavior of the Kripke structures. Relevant details on the Kripke structures and the CTL temporal operators may be found in [21].

For each of the HLPN models (M) presented in this work, we specify a property φ in CTL. The NuSMV verifies φ by finding all of the states that satisfy φ, that can be represented as: $\{s \in S \mid M, s \vDash \varphi\}$, according to Definition 2. The NuSMV performs the verification and furnishes the result as a *true* or a *false*. The property that we have verified for the HLPN models of the *HSM* and *FAR*, reflects their design limitation. The property is stated as: *a node accepts the same message that it has previously relayed in the network and eventually deleted from the ML after one or more replications*. The property that we have verified for the HLPN model of the *UBS* encompasses the addressing of the aforementioned design limitation. The property is stated as: *once a message is replicated by a node, the same message can never be accepted back by the node, even after being deleted from the ML*.

In the three schemes, a message may be deleted by a node, if the node finds the message to be in the RL or AL of a CN. While the deletion of a message in the schemes can also occur due to the lack of buffer space, *HSM* also supports conditional deletion. The model checking of the communication processes in an OMN necessitates a very high computation time. The optimizations used in this work to achieve the results in finite time are [25]: **(a)** dynamic variable reordering, **(b)** forcing the construction of a partial model comprising of only the variables that affect the specification by using the cone of influence, **(c)** disabling the computation of reachable states, and **(d)** disabling the generation of counterexamples.

6. RESULTS AND DISCUSSION

6.1. Simulation Results

The map-based mobility model of the ONE simulator has been exploited to recreate a large-scale OMN, comprising of places marked as homes, offices, shops, meeting points, and bus stops. The mobile nodes follow various routes on the map independently. The parameters considered for the simulations are: **(a)** world size 4500×3900m, **(b)** node range 50-100, **(c)** buffer size 500MB, **(d)** transmission range 20m, **(e)** message size 500KB-1MB, **(f)** message TTL 500min, and **(g)** time per simulation run 12h. The world size and buffer size are taken to be large enough to represent a typical OMN and the storage-friendly nodes, respectively. The parameter values $\alpha = 0.6$, $\omega = 50$, and $n = 10$, for the *HSM* and *UBS*, and $\phi = 0.6$ and $\omega = 50$ for the *FAR*, have been determined empirically under multiple test runs. The schemes exhibit the best performances for the aforementioned parameter values. The performance metrics considered in this work are the following.

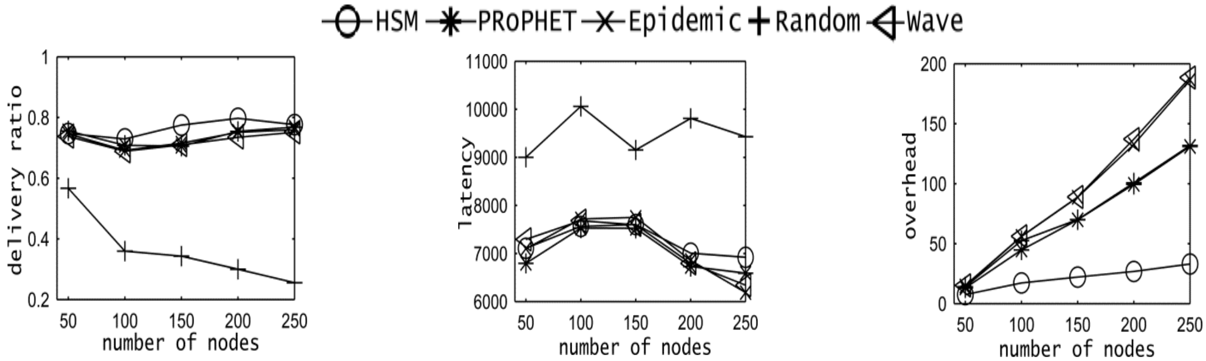
$$Message\ Delivery\ Ratio = \frac{1}{\mathrm{M}}\sum_{k=1}^{\mathrm{M}} R_k\ . \tag{17}$$

$$Latency\ average = \frac{1}{\mathcal{M}}\sum_{k=1}^{\mathcal{M}}(Receive\ Time_k - Creation\ Time_k). \tag{18}$$

$$Overhead = \frac{Total\ msgs\ relayed - Total\ msgs\ delivered}{Total\ msgs\ delivered}. \tag{19}$$

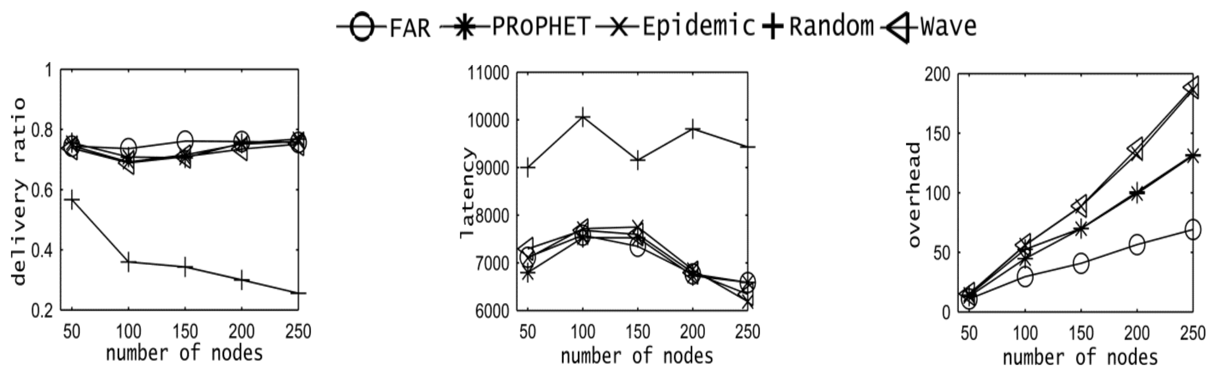
In (17), M is the total number of messages created, and $R_k = 1$, if the message m_k is delivered; otherwise, $R_k = 0$. In (18), $\mathcal{M}$ is the total number of messages received. Fig. 6–Fig. 8 represent the performance comparisons of the *HSM*, *FAR*, and *UBS*, respectively. The results of the *HSM* have been presented to reflect the performance with an increased buffer size of the storage-friendly nodes, as compared to our previous work [11]. The existing OMN routing schemes that have been compared to the three schemes are the *PRoPHET* [13], *Epidemic* [14], *Random* [15], and *Wave* [16]. The scalability performances of the *HSM* (Fig. 6(a)–Fig. 6(c)) and *FAR* (Fig. 7(a)–Fig. 7(c)) indicate that the schemes outperform the compared schemes, in terms of the delivery ratio and overhead. The *UBS* (Fig. 8(a)–Fig. 8(c)) outclasses the compared schemes, in terms of the delivery ratio, latency, and overhead.

The accomplishments of the *HSM*, *FAR*, and *UBS*, are attributed to the accuracy in forecasting the future contacts through the online analysis of the size-bound time-series data. The *PRoPHET* forecasts the future contacts, merely on the basis of the number of contacts. If the CD is not considered, a message can be replicated on a node that may not stay in contact with the destination for the required message transfer-time. The consideration of the ICT in the *HSM* and *UBS* ensures that a message is not replicated on a node that may not contact the destination, ahead of its life-time expiry.



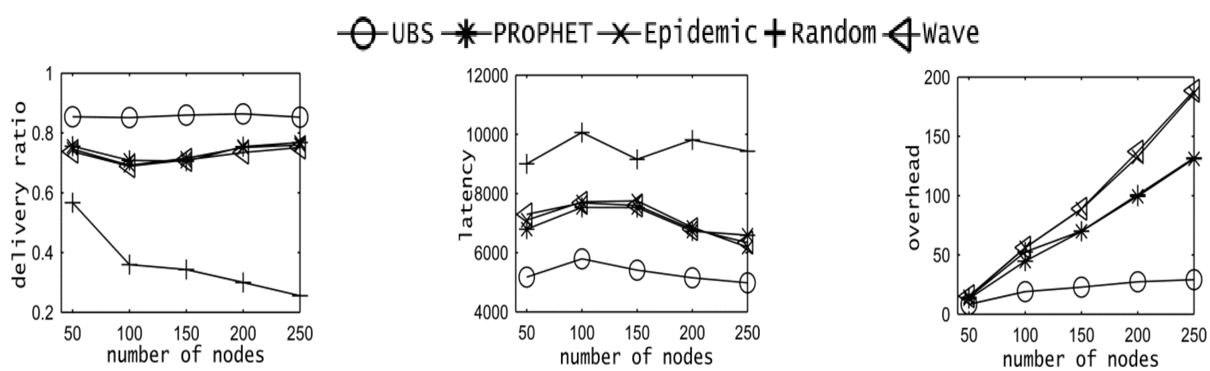
(a) (b) (c)

Fig. 6. Performance comparisons of the *HSM* in terms of the: (a) delivery ratio, (b) latency, and (c) overhead.



(a) (b) (c)

Fig. 7. Performance comparisons of the *FAR* in terms of the: (a) delivery ratio, (b) latency, and (c) overhead.



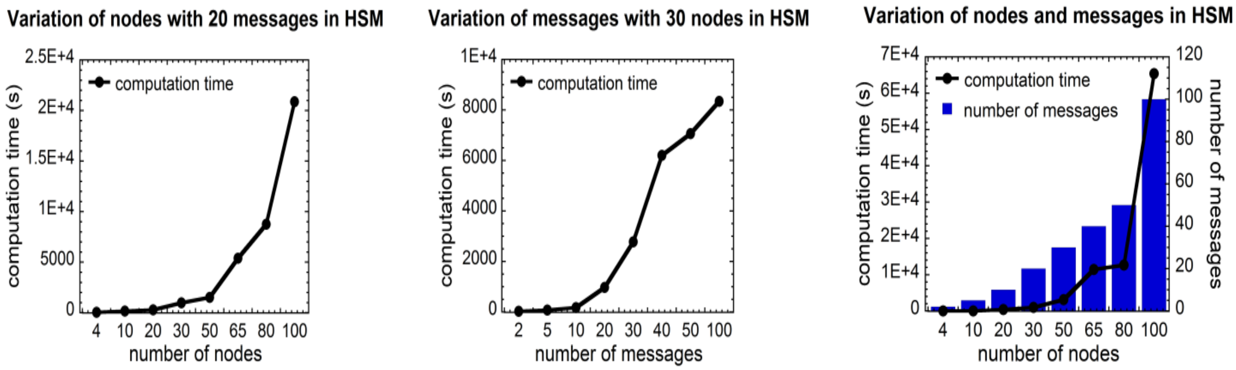
(a) (b) (c)

Fig. 8. Performance comparisons of the *UBS* in terms of the: (a) delivery ratio, (b) latency, and (c) overhead.

The augmented flooding in the *Epidemic* improves the message delivery at the cost of the overhead and message drop rate. In contrast, the *HSM* and *FAR* perform selective replication that assures a lower overhead and a higher delivery ratio. Sorting the aggregate utilities in the *UBS* ensures that each of the messages with a node contributes to the improvement in message delivery and latency. As all of the messages are transferred, the messages spend less time waiting in the buffers; thereby, lowering the latency. The *UBS* increases the message copies and does not perform conditional deletion; thereby, enhancing the delivery ratio. Disallowing the transfer of an already-replicated message in the *UBS* prevents message loops, lowers the message drop rate, and improves delivery ratio and overhead. The *Random* (relays a single message copy to any random neighbor) and *Wave* (utilizes tracking lists to curb flooding) implement selective and controlled replication. The schemes do not exploit the past meeting patterns of the nodes to perform replication. Therefore, in comparison, the *HSM*, *FAR*, and *UBS*, perform better.

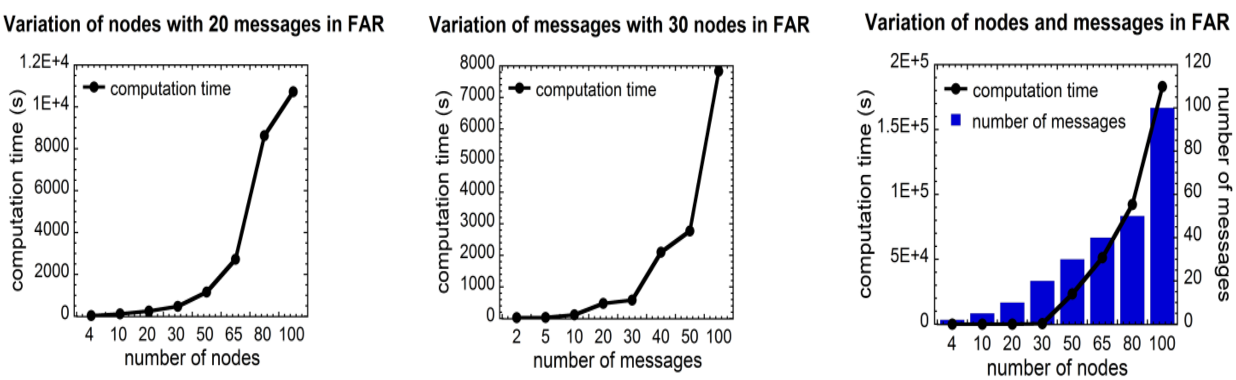
6.2. Verification Results

The verification has been done by translating the HLPN models of the *HSM*, *FAR*, and *UBS*, to the respective NuSMV models written in the NuSMV language. The properties discussed in Chapter 5 are specified in the CTL. To verify the communication processes and the scalability, we have chosen a communication path in each of the models. The lengths of the paths have been scaled up progressively by increasing the number of nodes and messages. The paths have been verified in the presence of up to 100 nodes and 100 messages. Being a highly time-consuming procedure, computation time is an important metric to consider for the verification of the schemes. Fig. 9 demonstrates the computation time (in seconds) required to verify the property of the *HSM*, by varying the number of: **(a)** nodes with 20 messages in Fig. 9(a), **(b)** messages with 30 nodes in Fig. 9(b), and **(c)** nodes and messages in Fig. 9(c). Fig. 10(a)–Fig. 10(c) and Fig. 11(a)–Fig. 11(c) illustrate the computation time for *FAR* and *UBS*, respectively.



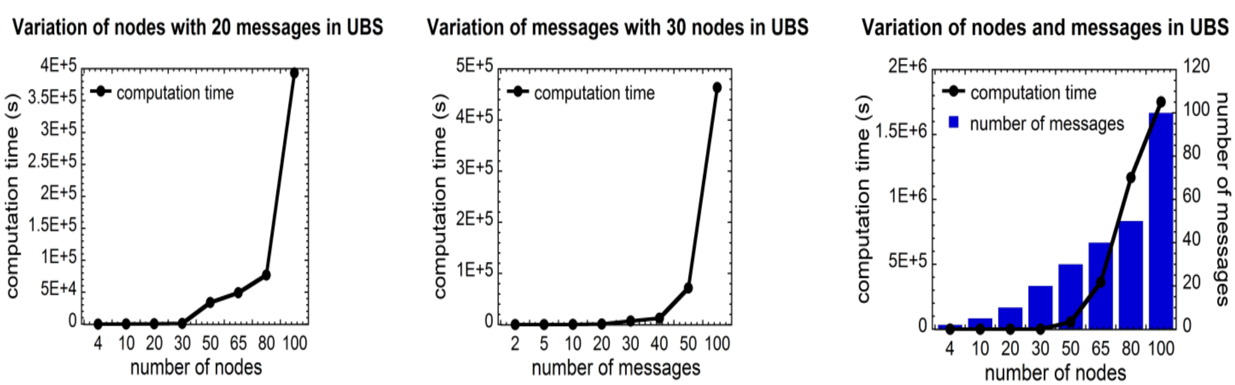
(a) (b) (c)

Fig. 9. Computation time of the *HSM* by varying the number of: (a) nodes, (b) messages, and (c) nodes and messages.



(a) (b) (c)

Fig. 10. Computation time of the *FAR* by varying the number of: (a) nodes, (b) messages, and (c) nodes and messages.



(a) (b) (c)

Fig. 11. Computation time of the *UBS* by varying the number of: (a) nodes, (b) messages, and (c) nodes and messages.

The three schemes exhibit similar trends of increasing computation times in the results, in all of the cases. The reason behind the aforementioned phenomenon is that the lengths of the communication paths are simultaneously scaled up as well. Moreover, the number of combinations encompassing the variables and parameters considered in the models proliferate with the scaling.

The computation of the ICT in the *HSM* mandates higher computation time (Fig. 9(a)–Fig. 9(b)) than the *FAR* (Fig. 10(a)–Fig. 10(b)). However, as the number of nodes and messages are increased simultaneously, the *FAR* exhibits a greater computation time (Fig. 10(c)) than the *HSM* (Fig. 9(c)). This is due to the fact that the deletion of a message from a node's buffer in the *HSM* can be comparatively quicker, due to the inherent conditional deletion. The *UBS* records the highest computation time (Fig. 11). The reasons are: **(a)** while the models of the *HSM* and *FAR* require a single execution path to corroborate the design limitation, the *UBS* requires each of the execution paths to be free of the limitation and **(b)** the implementation of the PML, aggregate utilities, and their sorting, further adds to the processing time.

7. CONCLUSION AND FUTURE WORK

7.1. Conclusion

In this work, we have capitalized on our previous work on the *HSM* to present two content dissemination schemes, the *FAR* and *UBS*, envisioned for the mobile clouds. The communication processes in the presented schemes have been formally analyzed in detail, aided by the HLPNs. The presented schemes exploit the mobility patterns and the temporal contacts of the nodes to predict the future contact opportunities. The results assert that the schemes (including the *HSM*) are ideal for the content dissemination in the dynamic and delay-tolerant mobile clouds. The schemes are shown to outperform the existing schemes. The *UBS*, conceived to obliterate a design limitation common to the *HSM* and *FAR*, distinctly outperforms the existing schemes. The *HSM* and *FAR* have been formally verified against the design limitation using complete model checking, while the *UBS* is shown to eliminate the limitation. To verify the specifications in finite time, model checking optimizations have been used. The verification results affirm the scalability and correctness of the models of the *HSM*, *FAR*, and *UBS*. The work corroborates that the: **(a)** HLPNs can be effectively exploited to depict the communication processes in the OMNs and **(b)** formal verification can be capitalized upon to design efficient routing solutions.

7.2. Future Work

As a part of our future endeavors, we intend to design, model, analyze, and verify the platforms for opportunistic computing, content sharing, job distribution, and information search, in the mobile clouds. We also aim at investigating modeling techniques that eliminate the requirement of model checking optimizations. As a part of our future endeavors, we intend to design, model, analyze, and verify the platforms for opportunistic computing, content sharing, job distribution, and information search, in the mobile clouds. We also aim at investigating modeling techniques that eliminate the requirement of model checking optimizations.

REFERENCES

[1] A. R. Khan, M. Othman, S. A. Madani, and S. U. Khan, "A Survey of Mobile Cloud Computing Application Models," *IEEE Communications Surveys & Tutorials,* Vol. 16, No. 1, 2014.

[2] M. Gerla, "Vehicular Cloud Computing," *11th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net '12),* 2012.

[3] Y. Wang, J. Wu, and W. Yang, "Cloud-Based Multicasting with Feedback in Mobile Social Networks," *IEEE Transactions on Wireless Communications,* Vol. 12, No. 12, Dec. 2013.

[4] C. Shi, M. H. Ammar, E. W. Zegura, and M. Naik, "Computing in cirrus clouds: the challenge of intermittent connectivity," *1st MCC workshop on Mobile cloud computing (MCC '12),* 2012.

[5] N. Fernando, S. W. Loke, and W. Rahayu, "Dynamic Mobile Cloud Computing: Ad Hoc and Opportunistic Job Sharing," *4th IEEE Int'l Conf. Utility and Cloud Computing (UCC '11),* Dec. 2011.

[6] S. Pal and T. Henderson, "MobOCloud: extending cloud computing with mobile opportunistic networks," *8th ACM MobiCom workshop on Challenged networks (CHANTS '13),* Sep. 2013.

[7] D. Zeng, S. Guo, I. Stojmenovic, and S. Yu, "Stochastic modeling and analysis of opportunistic computing in intermittent mobile cloud," *8th IEEE Conference on Industrial Electronics and Applications (ICIEA '13),* June 2013, pp. 1902-1907.

[8] R. Beraldi, K. Massri, and H. Alnuweiri, "Erasure-Coding Based Dissemination Protocol for Mobile Clouds," *8th Int'l Conf. P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC '13),* 2013.

[9] C. Shi, V. Lakafosis, M. H. Ammar, and E. W. Zegura, "Serendipity: enabling remote computing among intermittently connected mobile devices," *13th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '12),* 2012.

[10] M. Musolesi and C. Mascolo, "CAR: Context-Aware Adaptive Routing for Delay-Tolerant Mobile Networks," *IEEE Transactions on Mobile Computing,* Vol. 8, No. 2, Feb. 2009, pp. 246-260.

[11] O. Khalid, S. U. Khan, S. A. Madani, K. Hayat, L. Wang, D. Chan, and R. Ranjan, "Opportunistic Databank: A Context-aware on-the-fly Data Center for Mobile Networks," in *Handbook on Data Centers,* S. U. Khan and A. Y. Zomaya, eds., Springer-Verlag, New York, USA. (Forthcoming.)

[12] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE Simulator for DTN Protocol Evaluation," *2nd International Conference on Simulation Tools and Techniques (Simutools '09),* Mar. 2009, 10 pp.

[13] A. Lindgren, A. Doria, E. Davies, and S. Grasic, "Probabilistic Routing Protocol for Intermittently Connected Networks," http://tools.ietf.org/html/rfc6693, 2012, accessed July 2014.

[14] V. Mahendran, R. Gunasekaran, and C. Murthy, "Performance Modeling of Delay-Tolerant Network Routing via Queueing Petri Nets," *IEEE Transactions on Mobile Computing,* 2013.

[15] A. Balasubramanian, B. N. Levine, and A. Venkataramani, “Replication Routing in DTNs: A Resource Allocation Approach,” *IEEE/ACM Trans. Networking (TON),* Vol. 18, No. 2, Apr. 2010.

[16] J. Ott, A. Keränen, and E. Hyytiä, “BeachNet: Propagation-based Information Sharing in Mostly Static Networks,” *3rd Extreme Conference on Communication: The Amazon Expedition (ExtremeCom ’11),* Article 15, Sep. 2011, 6 pp.

[17] Y. Li, P. Hui, D. Jin, L. Su, and L. Zeng, “Performance Evaluation of Routing Schemes for Energy-Constrained Delay Tolerant Networks,” *IEEE Int’l Conf. Communications (ICC ’11),* June 2011.

[18] K. Akkarajitsakul, E. Hossain, and D. Niyato, “Cooperative Packet Delivery in Hybrid Wireless Mobile Networks: A Coalitional Game Approach,” *IEEE Transactions on Mobile Computing,* Vol. 12, No. 5, May 2013, pp. 840-854.

[19] Y. Wang, H. Wang, and C. Xia, “An Anycast Communication Model for Opportunistic Network,” *Int’l Conf. Computational and Information Sciences (ICCIS ’11),* Oct. 2011, pp. 867-870.

[20] M. Jung, K. Han, and J. Cho, “Advanced verification on WBAN and cloud computing for u-health environment,” *Multimedia Tools and Applications,* May 2014, pp. 1-18.

[21] S. U. R. Malik, S. U. Khan, and S. K. Srinivasan, “Modeling and Analysis of State-of-the-art VM-based Cloud Management Platforms,” *IEEE Trans. Cloud Computing,* Vol. 1, No. 1, 2013, pp. 1,1.

[22] *Lectures on Petri Nets I: Basic Models, Advances in Petri Nets,* Vol. 1491, W. Reisig and G. Rozenberg, eds., Springer, Berlin, 1998.

[23] *NuSMV Overview,*

http://nusmv.fbk.eu/NuSMV/index.html, accessed July 2014.

[24] P. Arcaini, A. Gargantini, and E. Riccobene, “Using mutation to assess fault detection capability of model review,” *Software Testing, Verification and Reliability,* 2014.

[25] *NuSMV 2.5 User Manual,*

http://nusmv.fbk.eu/NuSMV/userman/v25/nusmv.pdf,

accessed July 2014.

[26] I. Rhee, M. Shin, S. Hong, K. Lee, S. J. Kim, and S. Chong, “On the Levy-Walk Nature of Human Mobility,” *IEEE/ACM Transactions on Networking (TON),* Vol. 19, No. 3, June 2011, pp. 630-643.

[27] C. Song, Z. Qu, N. Blumm, and A. L. Barabási, “Limits of Predictability in Human Mobility,” *Science,* Vol. 327, No. 5968, 2010.