Proactive Caching for Producer Mobility Management in Named Data Networks

Hesham Farahat
ECE Department, Queen's University
Kingston, ON, Canada
h.farahat@queensu.ca

Hossam S. Hassanein
School of Computing, Queen's University
Kingston, ON, Canada
hossam@cs.queensu.ca

***Abstract*—Named Data Networks (NDNs) offer a promising paradigm for the future Internet to cope with the growing demand for data and the shifts in applications. One of the main challenges in NDNs is how to support a seamless operation during mobility. In this paper, we propose a proactive caching scheme (named *ProCacheMob*) to support Producer mobility that exploits location predictors and data requests patterns to cache data before handover occurs. In essence, *ProCacheMob* adopts the predicted future Interests, that will be sent to the mobile Producers, and caches their data contents ahead of handover. Thus, avoids Interest retransmission that increases the Consumer's delay and decreases the network efficiency during Producer's mobility. *ProCacheMob* is simulated in ndnSIM and evaluated against mainstream NDN mobility solutions. The simulation results show how the scheme is successful to avoid packets drops and decreases the delay experiencedby Consumers by 52% compared to other schemes.**

***Index Terms*—NDN, ICN, Proactive, Predictive, Heuristic optimization, Mobility, Caching**

I. INTRODUCTION

In the last decade the use of the Internet has changed due to the evolution of content, devices and applications. CISCO's Virtual Networking Index (VNI) [1] predicts that 164 Exabytes will travel the Internet permonth by 2019, nearly a triple increase over 2014 statistics. However, the Internet was built as host-based model which is not designed for the current use of content distribution. The current solution to maintain scalable operations is to patch the Internet with overlays such as Content Distribution Network (CDN) and Peer-to-peer (P2P), which are bound to fail under the projected future traffic [2].

Recent research efforts have been directed at designing a content-based, instead of hosts, paradigm for the new Internet, namely Information-centric Network (ICN). ICN is designed to target the growing traffic challenges with content-oriented application [3]. While considering the strength and weaknesses of the current design, several potential ICN architectures have been proposed such as Named Data Network (NDN) [2]. The main aim of NDN is to evolve the role of the IP architecture such that the packets can name content not hosts.

The number of mobile devices and amount of traffic generated by it is drastically increasing which is predicted to be 66% of IP traffic by 2019 [1]. Consequently, supporting mobility as a network primitive is a core feature of any future Internet design. The challenge of mobility in the Internet is in tracing hosts, whereas in NDN it is in tracing date. Moreover, supporting *seamless* operations, where users are able to move freely without impacting the quality of experience, is important in NDN. However, NDN's implicit assumption that mobility is supported intrinsically is impractical and a design of a seamless mobility management scheme is necessary to successful NDN.

Several efforts have targeted the Producer mobility challenge in ICN generally, and NDN specifically. Whereas most of the proposed schemes use reactive techniques to recover after a mobile event, in this paper, we propose a real-time proactive caching scheme to support seamless Producer mobility. The scheme, called *ProCacheMob*, harnesses location prediction techniques and users' access patterns to proactively store potential data on in-network caches in real-time. Thus, Interests generated during a mobility event can be satisfied without the need to reach the moving Producer. Finding a near-optimal placement that goals for no Interests drops and with bounded overhead has the potential to provide a proactive solution to the Producer mobility problem. Specifically, the main contributions of this work are:

1) We propose a real-time proactive caching scheme that uses position and user's request predictions to support seamless mobility in NDN.
2) We simulate the scheme in ndnSIM using the assessment framework in [4] to evaluate the effectiveness of the proposed scheme and comparing it to the benchmark designed in [5]. This is in addition to evaluating the performance against other mobility schemes.

The remainder of the paper is organized as follows. In Section II, an overview of the current NDN mobility management schemes is introduced. The proposed scheme is presented in Section III. . Our experiments are discussed in Section IV. Finally, we conclude our findings in Section V and present insights into future directions in NDN mobility support.

II. RELATED WORK

NDN is one of the promising architectures of the future ICN, which is designed to use receiver-driven communications. In this paper, we refer to the requester as Consumer and the Data owner as Producer. Every node in NDN has two structures to help forwarding Interests and Data. The first is Forwarding Information Base (FIB), which is used to forward

978-1-5090-4372-9/17/$31.00 ©2017 IEEE

Interests to Producers. The second structure is the Pending Interest Table (PIT), which tracks of unsatisfied Interests and forwards back the Data to Consumers. A third structure is used to support in-network caching in NDN, called Content Store (CS).

As mentioned earlier, the design on NDN does not include a scheme specifically to support mobility. It relies on Interests retransmission to recover from such events. However, this solution is not scalable in large networks with many moving Producers and it results in Interest retransmissions during the convergence time, time needed by the network to update the routing tables. Therefore, a Producer mobility scheme is required to be added to the current proposed design in NDN.

Multiple mobility management schemes have been proposed in the literature focusing on both Consumer and Producer mobility, te former is covered in [6]. There are two main approaches: Mobility Anchor and Location Resolution solutions. Detailed discussion and performance evaluation of the two approaches were presented in [4]. Mobility anchor schemes proposed in [7]–[10] are designed based on the MobileIP protocol [11] used in the current Internet. It uses special nodes in every network, called anchors, that forward the Interests to the Producer even if the latter is moving to another network (roaming). In this case, the Interests are not going to be dropped but it may take longer paths to reach the Producer since it has to pass through the anchor first. The approach in Location Resolution schemes [12]–[15] is similar to the one used in Domain Name Systems (DNSs), where the Consumer queries the location of the Producer before sending Interests. However, this approach requires early binding techniques similar to Data-Oriented Network Architecture (DONA) [16], which affects content naming used in NDN and requires extra overhead to maintain and query locations.

III. THE SCHEME *ProCacheMob*

The problem in Producer mobility is that during the convergence time, the network is not updated with the new location of the Producer. Hence, any Interest that is directed to this Producer's old Point of Attachment (PoA) will be dropped, unless it finds the Data in one of the caches along the path from the Consumer to the Producer. The main idea of *ProCacheMob* is to proactively place the Data that will be requested during the handover in the network before the event occurs. Two sets of information are required to find the best Data placement.

The first is the time of handover of a moving Producer which can be predicted using location estimators [17]–[19]. Such systems will predict the new position of the node at a specific time using current trajectories and history information. If the predicted location is out of the range of the current Producer's PoA, a handover event is predicted. The second type of information is to predict users request patterns during the convergence time. Request's predictors such as in [20]–[22] can be used to find the time, Data and Consumer of a future request. In this work, we assume such information to be available and correct (i.e., perfect knowledge of the future). However, we do sensitivity analysis in Section IV to test the effect of error in predictions.

Compiling the two information, we can find the set of potential Interests to be dropped after Producers mobility event. Given this set of requests, we can define the problem as finding the best Data placement in network caches such that no retransmissions of Interests are required by the Consumer and with bounded overhead on the network.

A. Algorithm details

The proposed scheme uses guided heuristics to find a solution to the predictive caching problem with low complexity which can be used in real-time. The set of Interests which are predicted to be dropped is the main input for the proposed algorithm. Using the given set, *ProCacheMob* finds the optimal placement for each Interest individually in order of request time. This is to give priority to Interests that will be dropped first if its Data is not cached. The optimal placement for one Interest is chosen to be the closest router to the Consumer that has enough space and does not violate the following constraints:

1) The solution should not violate the space requirements of the network. Namely, the amount of Data added to a router should not exceed the maximum capacity C_r^{max}.
2) Placing Data on routers that are not on the path from Consumer to Producer requires path update messages which is an overhead of the scheme. The number of path updates for each Data d is $\mathbb{P}^r_{u,u'} = (|P_{u\to r}| - |P_{u\to u'} \cap P_{u\to r}|)$ is the number of non-common edges between $P_{u\to u'}$ (path from the Consumer u to Producer u') and $P_{u\to r}$ (new path from the Consumer u to the chosen router r). Hence, to bound the overhead the summation of $\mathbb{P}^r_{u,u'}$ over all Data should be less than the overhead threshold ω.
3) Number of data replacements (in case the cache is full) should be less than α.
4) Total amount of traffic generated to support mobility should be less than a threshold ζ.
5) Each Data d has a content value (r.g. popularity) ϵ_d. The value K_r is the total content value of all the items in r. The reduction of this content value due to data replacement is bounded by a threshold β.

If there are more than one feasible routers with the same number of hops to the Consumer, the one with minimal path updates is chosen. This greedy approach does not guarantee an optimal solution, but it provides a near-optimal one. This is because the decision considers the delay then path updates minimization while satisfying all the constraints.

The *ProCacheMob* consists of the following four main stages as summarized in Algorithms 1-2 :

1) Data Preparation: The function *getNetworkMetrics* is used to collect statistics from the network before taking decisions. In particular, for every pair of Interests and routers, the number of path update messages needed to reach the router and the number of hops to the Consumer are calculated.

Algorithm 1 Pseudocode of *ProCacheMob*

```
Input: $network$: The current state of the network
Input: $D$: Set of requests to be optimized
Input: $R$: Set of Routers in the network
  Initialize $Sol.DA[r] = 0 \quad \forall r \in R$
  Initialize $Sol.DR[r] = 0 \quad \forall r \in R$
  GETNETWORKMETRICS(network,$D$,$R$)
  for all $d \in D$ do
      $R' \leftarrow$ SORTROUTERS$(d, R)$
      for all $r \in R'$ do
          $Sol' \leftarrow Sol$
          $Sol'.DA[r] \leftarrow Sol'.DA[r] + 1$
          if $C_r + Sol.DA[r] - Sol.DR[r] = C_r^{max}$ then
              $Sol'.DR[r] \leftarrow Sol'.DR[r] + 1$
          end if
          if $Sol'$.ISFEASIBLE($d$,$r$) then
              $Sol \leftarrow Sol'$
              $Sol.SelectedRouter[d] \leftarrow r$
              Break
          end if
      end for
  end for
  APPLYSOLUTION(Sol)
```

2) Greedy Placement and Removal: The greedy algorithm iterates over the set of Interests that are predicted to be dropped. For each Data d needed to be cached, the algorithm firstly sorts the full list of caching routers in ascending order based on the number of hops to the Consumer. Each subset of caching routers with the same number of hops is further sorted by the number of update messages needed to direct Interest i of Data d to the routers. While the former sorting aims to minimize the Consumer's delay, the latter aims to minimize the network overhead. The algorithm sequentially selects the routers and tests the possibility of caching the Data d. In case of capacity constraint violation, Data replacement occurs.
3) Feasibility Check: The feasibility of the above Data placement and removal actions are then checked using Algorithm 2. In particular, the solution (i.e. selected router) that satisfies all constraints is considered as optimal and the Data d will be stored in the current router r. Otherwise, if any of the constraints is violated (i.e. Algorithm 2 returns false), the solution is said to be infeasible and the next router will be checked. In case of visiting all the routers without finding a feasible one, the Data d will not be cached and will be sent upon the user's request.
4) Solution Broadcasting: After selecting routers of all Data if possible, the last stage places the Data on the chosen routers and sends FIB update messages to off-path routers.

B. Complexity Analysis

ProCacheMob finds a near-optimal solution in real-time since the complexity of the algorithms is polynomial. Specifically, the algorithm loops on the set of Data and sorts the list of routers per iteration. Assuming the sorting algorithm has a complexity $O(RLog(R))$, then the complexity of the algorithm is $O(DRLog(R))$.

Algorithm 2 Data structure $Solution$ with attributes and feasibility test function

```
class $Solution$
+ $PathUpdates$                  ▷ Total path updates
+ $ProducerHops$                 ▷ Total number of hops to send all Data
+ $DA[R]$                        ▷ Array to hold number of added Data per router
+ $DR[R]$                        ▷ Array to hold number of removed Data per router
+ $VDA[R]$                       ▷ Array to hold value of added Data per router
+ $VDR[R]$                       ▷ Array to hold value of removed Data per router
+ $SelectedRouter[D]$            ▷ Array to hold all selected routers

  function ISFEASIBLE($d$,$r$)
      $cons \leftarrow$ GETCONSUMER($d$)
      $prod \leftarrow$ GETPRODUCER($d$)
      $P_1 \leftarrow$ GETPATH($network$,$prod$,$r$)
      $P_2 \leftarrow$ GETPATH($network$,$cons$,$r$)
      if $ProducerHops + |P_1| > \zeta$ then
          return False
      end if
      if $PathUpdates + PathUpdates[d][r] > \omega$ then
          return False
      end if
      if $C_r + DA[r] - DR[r] > C_r^{max}$ then
          return False
      end if
      if $DA[r] - DR[r] > 0$ then
          return False
      end if
      if $DR[r] > \alpha$ then
          return False
      end if
      if $(VDR[r] - VDA[r])/K_r > \beta$ then
          return False
      end if
      $ProducerHops \leftarrow ProducerHops + |P_1|$
      $PathUpdates \leftarrow PathUpdates + PathUpdates[d][r]$
      return True
  end function
end class
```

IV. RESULTS AND DISCUSSION

A. Simulation Model

We use the assessment framework proposed in [4]. The benchmarking tool is designed to evaluate both Producer and Consumer mobility management schemes in several scenarios using multiple performance metrics. The tool creates hierarchical topologies, requests patterns and mobility scenarios generated by Simulation of Urban MObility (SUMO). This information builds the scenario that is fed to the Network Simulator which is based on ndnSIM [23].

B. Experiment Setup

The parameters of the experiments are summarized in Table I. There are 50 Consumers and 50 Producers moving in 7x7 street grid plan. Consumers request Data with a rate of 50 to 80 Interests per second. The content requested is based on a popularity distribution Zipf with parameter $s = 0.2$. The topology used is hierarchal with 40 core routers distributed in 5 domains. Every core router has a cache size of 1000 items and uses Least Recently Used (LRU) replacement policy.

TABLE I
SIMULATION PARAMETERS

	Parameter	**Value**
General	Simulation Duration	1000s
	Transit Period	80s
	Map size	$1400m \times 1400m$
	Number of Blocks	7×7
	Number of Users	100
	Producers	50
	Consumers	50
Application	Interest Rate	50-80 I/s
	Zipf's s	0.2
	Content per Producer	$1000 \times 1KB$
Topology	APs	49
	AP Range	200m
	Number of Routers	40
	Core router's links	10Mbps
	Access router's links	5Mbps
	Propagation delay	10ms
Mobility	Model	Manhattan
	Handover delay	0.5s
	Speed	70 km/h
NDN	Forwarding Scheme	BestRoute
	Cache replacement	LRU
	Cache Size	1000 objects
ProCacheMob	Prediction Window	2s
	α	10%
	β	10%
	ω	$1.2 \times D$

For every experiment one factor is varied to test the impact of it on the performance of the scheme. *ProCacheMob* is controlled by a time window which is set to 2 seconds) and three thresholds to bound the overhead. After tuning the parameters, the thresholds are set to the following:

1) ζ: controls how far the Data will travel from the Producer side. The value is calculated using the current state of the network and it is based on the average number of hops needed to reach the Producer from the Consumer. Hence, the Data will not be cached farther than the Producer.
2) α: the maximum number of replacements per router is set to 10% of the cache size.
3) β: the total content value of a router should not decrease more than 10% of its total value.

We execute each experiment for 20 runs with different random seeds, then the averages of all metrics were calculated. The 95% confidence interval has a maximum deviation of 5% from the reported average values.

C. Schemes and Evaluation Metrics

We evaluate *ProCacheMob* against the optimal solution proposed in [5] as a benchmark in addition to three mobility management approaches. Specifically, NDN with no scheme (Pure-NDN), Mobility Anchor (*MA*) represented by [7] and Location Resolution Scheme (*LRS*) represented by [12]. Additionally, We consider the following three main metrics that reflect the satisfaction of the Consumer: 1) Consumer Delay: Calculated as the time difference between the first attempt of sending the Interest and successfully receiving the Data at the Consumer. 2) Delivery Ratio: is the proportion of successful Data packets received by the Consumer to the total number of Interests sent. This metric is a measure of how successful is the scheme in avoiding both Interest and Data drops. 3) Overhead: Calculated as the percentage of total number of control packets generated by the scheme to the total number of Interests.

D. Producer Mobility

The first scenario evaluates all mobility management schemes with different number of mobile Producers in the network. The results in Fig. 1 and Fig. 2 demonstrate the ability of the introduced scheme to achieve very close performance to the benchmark used by proactively caching the future content to be requested. Therefore, avoids retransmission of Interests to the new location and attained near maximum delivery ratio irrespective of the mobile Producers percentage. Moreover, the Data placement near Consumer's location resulted in the minimal delay, which is also stable over the percentage of mobility events as shown in Fig. 1. This near-optimal performance requires an extra overhead, shown in Fig. 3, for delivering the mobile Producer's content to the selected cache and sending path update packets in the network.

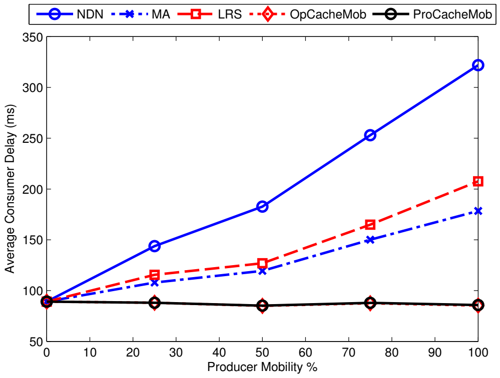

Fig. 1. Average Consumer delay with different mobility % for all schemes

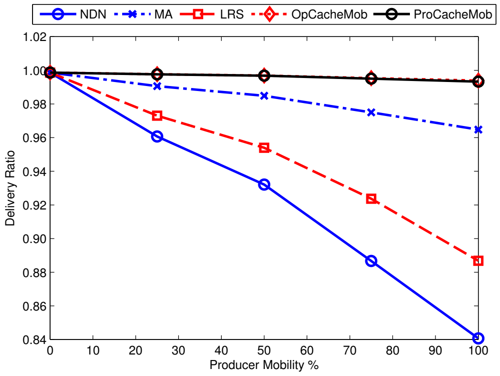

Fig. 2. Average delivery ratio with different mobility % for all schemes

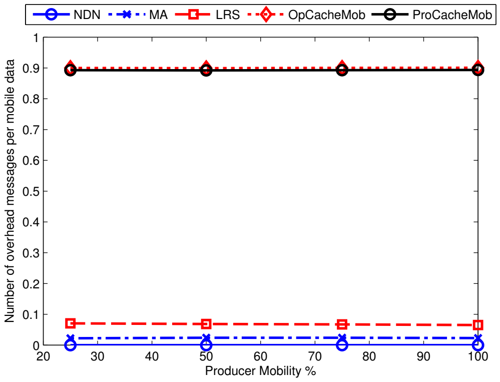

Fig. 3. Scheme overhead with different mobility percentages for all schemes

As such, the results revealed a significant improvement of

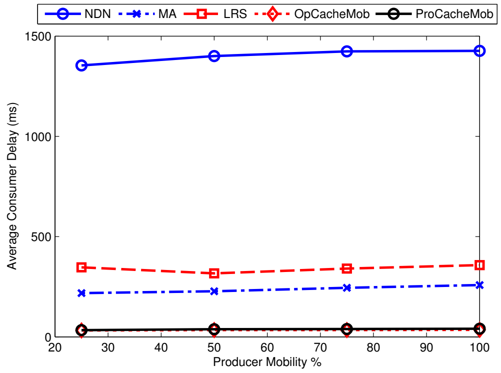

Fig. 4. Average Consumer delay for Interests issues during handover with different mobility percentages for all schemes

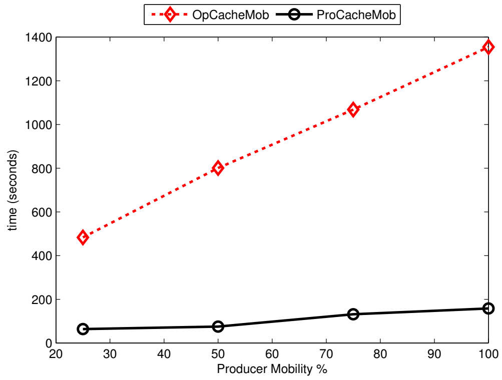

Fig. 5. Execution time for proactive schemes

the total delay which decreased by more than 50%, in the case of the pure-NDN, when only 50% of the Producers change their location Fig. 1. This is attributed to the simplicity of NDN that depends on Interest retransmission to recover from Producer Mobility and thus suffers from a poor delivery ratio shown in Fig. 2 especially at a higher mobility percentage.

The existing non-predictive *MA* and *LRS* attain acceptable delays at low Producer's mobility (i.e., 25%), redirecting the Interests to the new locations is suboptimal at more dynamic scenarios. Thus, when 75% of Producers are mobile, the Consumers suffered from a 90% and 65% increase in the delay compared to the predictive schemes, using *LRS* and *MA*, respectively as shown in Fig. 1. This delay is also associated with an increased number of Interest retransmissions when the original Interests become outdated for the Consumers (i.e., after the time out). Thus, a drop in the delivery ratio is also experienced by the network under these non-predictive techniques as shown in Fig. 2. Figure 4 depicts more detailed results by showing the average Consumer delay for the Interests that were sent during handover only.

Moreover, *ProCacheMob* achieves such performance with lower complexity as explained before in Section III. Fig. 5 shows the total time taken to run both optimization techniques in different mobility scenarios. As depicted, it is 7x faster than the benchmark when 100% of Producers are mobile.

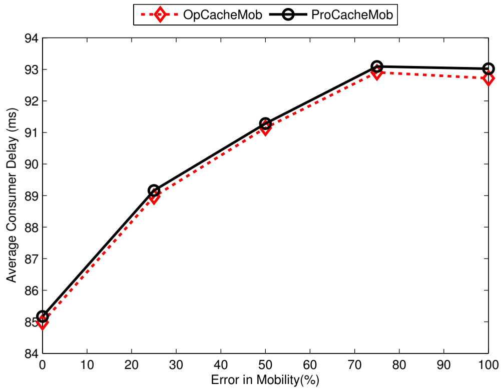

Fig. 6. Impact of errors in mobility - average delay

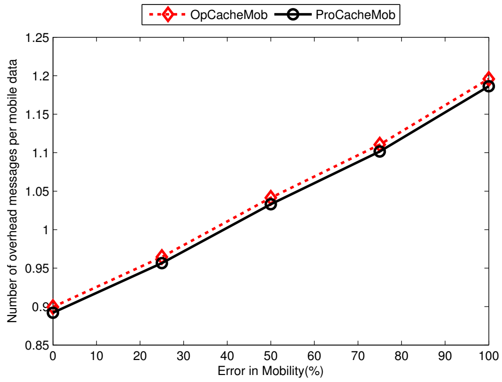

Fig. 7. Impact of errors in mobility - scheme overhead

E. Sensitivity Analysis

The above performance of the proposed *ProCacheMob*was evaluated under perfect knowledge of both Producers' mobility and Consumers' requests. We therefore perform sensitivity analysis to evaluate the effect of prediction errors on the aforementioned gains.

1) Imperfect Mobility Prediction: Error in predicting the future location of a node may lead to incorrect PoA. To simulate this type of error, we add a uniformly distributed random number to anticipated Consumer's positions such that the PoA of nodes on the edge of an AP is altered to one of the neighboring PoA. Fig. 6 and Fig. 7 depict the delay and overhead of the proposed scheme compared to the optimal benchmark with an error in mobility prediction. The average Consumer delay of all the schemes is increased by 9% when all locations are erroneously. The extra delay is due to the scenarios when handover events are missed because of wrong prediction. Nevertheless, the gap between Mobility Anchor schemes and the predictive schemes while wrong predictions are made, is 90%. On the other hand, the overhead is increased by 33% because of the false predicted handovers.

2) Imperfect Consumer Request Prediction: The proactive schemes proposed use request predictors to identify the Data that should be prefetched before Producer's handover. If wrong requests are predicted, the delays of these requests are shortened whereas the mobile requests are not. In this experiment, a uniform random error is added to the predicted

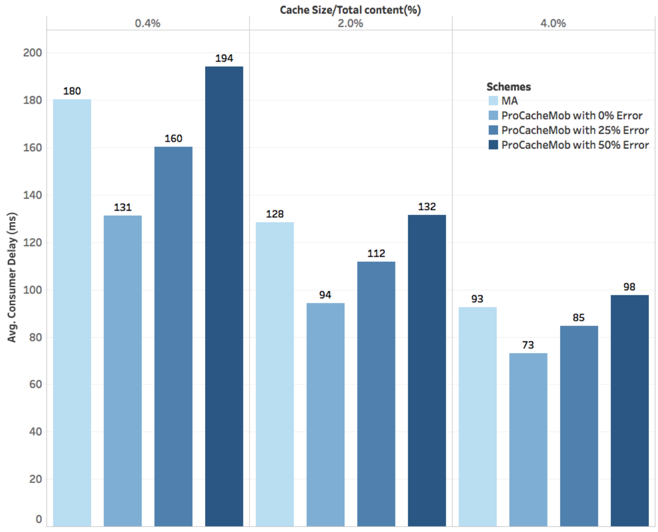

Fig. 8. Impact of errors in requests

Interests such that the name of the request is changed to another random name. Fig. 8 demonstrates the average delay at 50% mobile Consumers for the Mobility Anchor scheme and *ProCacheMob* with 0%, 25% and 50% of error in prediction. Furthermore, 3 different cache sizes are tested. The delay increased on average by 57% during 50% wrong predictions. In this particular case, the average delay is 3% longer than the delay of Mobility Anchor scheme.

V. CONCLUSIONS AND FUTURE WORK

Supporting seamless mobility in NDN is crucial for smooth network operation under both Consumer's and Producer's mobility. This work designed and implemented an proactive Producer mobility management scheme *ProCacheMob*that utilizes the available cache resources, and both location and Data predictions. Under perfect knowledge, the results demonstrated the ability of *ProCacheMob* to proactively cache the future content of the anticipated mobile Producers and avoid unnecessary delays. While generating extra network overhead, both the Consumer delay and delivery ratio remained almost constant, regardless of the number of mobility events. Moreover, sensitivity analysis proved the robustness of the introduced *ProCacheMob*to imperfect mobility predictions and suboptimal traffic forecasts. Thus, provides the network operator with accuracy requirements for the prediction techniques and the corresponding operational region of the predictive scheme.

Our future work will consider two main improvements to the current heuristic proposal: 1) Introduce distributed algorithm to add more scalability to the scheme. 2) Adopts stochastic optimization to extend the robustness of the predictive scheme at higher Data prediction errors.

REFERENCES

[1] "Cisco visual networking index: Forecast and methodology, 2014–2019," White Paper, Cisco, May 2015.

[2] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *SIGCOMM Computing Commun. Rev.*, vol. 44, no. 3, pp. 66–73, Jul 2014.

[3] G. M. Brito, P. B. Velloso, and I. M. Moraes, *Information Centric Networks: A New Paradigm for the Internet*. Wiley, 2013.

[4] H. Farahat and H. Hassanein, "On the design and evaluation of producer mobility management schemes in named data networks," in *ACM Int. Conf. Modeling, Anal. and Simulation Wireless and Mobile Systems(MSWIM)*, Cancun, Mexico, 2015, pp. 171–178.

[5] ——, "Optimal caching for producer mobility support in named data networks," in *IEEE Int. Conf. Commun. (ICC)*, May 2016, pp. 1–6.

[6] H. Farahat and H. S. Hassanein, "Supporting consumer mobility using proactive caching in named data networks," in *IEEE Global Commun. Conf. (GLOBECOM)*, Dec 2016, pp. 1–6.

[7] J. Lee, S. Cho, and D. Kim, "Device mobility management in content-centric networking," *IEEE Commun. Mag.*, vol. 50, no. 12, pp. 28–34, 2012.

[8] F. Hermans, E. Ngai, and P. Gunningberg, "Global source mobility in the content-centric networking architecture," in *Proc. ACM Workshop Emerging Name-Oriented Mobile Networking Design - Architecture, Algorithms, and Appl.*, Hilton Head, SC, USA, 2012, pp. 13–18.

[9] D. Han, M. Lee, K. Cho, T. Kwon, and Y. Choi, "Publisher mobility support in content centric networks," in *Int. Conf. Inform. Networking (ICOIN)*, Phuket, Thailand, Feb 2014, pp. 214–219.

[10] Y. Luo, J. Eymann, and A. Timm-Giel, "Mobility support for content centric networking," *Telecommunication Syst.*, pp. 1–18, 2014.

[11] E. C. Perkins, "IP mobility support for IPv4, revised," RFC 5944, 2010.

[12] D.-h. Kim, J.-h. Kim, Y.-s. Kim, H.-s. Yoon, and I. Yeom, "End-to-end mobility support in content centric networks," *Int. J. Commun. Syst.*, vol. 28, no. 6, pp. 1151–1167, 2015.

[13] Z. Zhu, A. Afanasyeva, and L. Zhang, "A new perspective on mobility support," NDN, Technical Report NDN-0013, 2013. [Online]. Available: http://named-data.net/publications/techreports/trmobility/

[14] A. Azgin, R. Ravindran, and G. Wang, "A scalable mobility-centric architecture for named data networking," *CoRR*, vol. abs/1406.7049, 2014. [Online]. Available: http://arxiv.org/abs/1406.7049

[15] Z. Zhou, X. Tan, H. Li, Z. Zhao, and D. Ma, "MobiNDN: A mobility support architecture for NDN," in *Chin. Control Conf. (CCC)*, Nanjing, China, Jul. 2014, pp. 5515–5520.

[16] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," *SIGCOMM Computing Commun. Rev.*, vol. 37, no. 4, pp. 181–192, 2007.

[17] N. Samaan and A. Karmouch, "A user centric mobility prediction approach based on spatial conceptual maps," in *IEEE Int. Conf. Commun. (ICC)*, vol. 2, Seoul, Korea, May 2005, pp. 1413–1417.

[18] K. Laasonen, "Route prediction from cellular data," in *Proc. 2nd Workshop Context Awareness for Proactive Syst.*, Helsinki, Finland, 2005, pp. 147–158.

[19] H. Abu-Ghazaleh and A. Alfa, "Application of mobility prediction in wireless networks using markov renewal theory," *IEEE Trans. Veh. Technol.*, vol. 59, no. 2, pp. 788–802, Feb. 2010.

[20] X. Chen and X. Zhang, "A popularity-based prediction model for web prefetching," *IEEE Computer*, vol. 36, no. 3, pp. 63–70, Mar 2003.

[21] J. G. Cleary and I. Witten, "Data compression using adaptive coding and partial string matching," *IEEE Trans. Commun.*, vol. 32, no. 4, pp. 396–402, Apr 1984.

[22] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan, "Web usage mining: Discovery and applications of usage patterns from web data," *SIGKDD Explorations Newslett.*, vol. 1, no. 2, pp. 12–23, Jan. 2000.

[23] A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM: NDN simulator for NS-3," NDN, Technical Report NDN-0005, Oct. 2012.