A Reinforcement Learning Approach to Adaptive Forwarding in Named Data Networking

Olumide Akinwande and Erol Gelenbe

Department of Electrical & Electronic Engineering
Imperial College London, UK

Abstract. This paper addresses Information Centric Networks, and considers in-network caching for Named Data Networking (NDN) architectures. We depart from forwarding algorithms which primarily use links that have been selected by the routing protocol for probing and forwarding, and propose an adaptive forwarding strategy using reinforcement learning with the random neural network (NDNFS-RLRNN), to leverage the routing information and actively seek possible deliveries outside these paths in a controlled way. Our simulations show that NDNFS-RLRNN achieves more efficient delivery performance than a strategy that strictly follows the routing layer or a strategy that retrieves contents from the nearest caches by flooding requests.

1 Introduction

Information-Centric Networking (ICN) treats content as fundamental by naming and addressing data objects at the network level, which is a significant departure from the Internet's host-centic architecture. This enables ICN to naturally support features like in-network caching and many-to-many communication which can be essential in meeting the current predominant use of the Internet [1,2]. This paper focuses on *Named Data Networking* (NDN) [3], one of the prominent ICN approaches. In NDN, where requests are sent in *Interest packets*, while *Data packets* carry the requested contents; both packets carry the name or identification of the desired data object. An Interest is uniquely identified by its name and nonce values, where the nonce is randomly generated by the requesting application. A NDN node maintains three main data structures for implementing the forwarding plane: a *content store* (CS) which acts as a cache for Data packets, a *pending interest table* (PIT) which keeps track of unanswered requests, and a *forwarding information base* (FIB) which maps reachable name prefixes to outgoing interfaces. The FIB is updated by a name-based routing protocol.

NDN differs significantly from most of the other ICN proposals because it incorporates an intelligent forwarding plane through its *strategy module*. The strategy module is a program or algorithm, defined for each prefix in the FIB, for making Interest forwarding decisions. It is expected to leverage on both the routing information in the FIB and the observed packet delivery measurements in making adaptive decisions. An important benefit of this layer is that it can relax the stringent convergence and correctness demands on the routing layer [4]. A NDN node matches an Interest, using the content name, in the CS, PIT and FIB, in that order. A match in the FIB results in the strategy being called upon to decide the Interest's next hop. Data packets are sent along the

reverse paths used by their corresponding Interest packets, following the states in the PITs. For scalability and practicality reasons, the name-based routing protocol in NDN can only announce and monitor paths leading to the actual content sources and designated repositories, thus leaving the responsibility of exploiting the in-network caching capability, which is important for delivering the design goals of NDN, entirely to the forwarding strategy. However, most of the proposals for the strategy layer, including the algorithm currently adopted by the NDN project, follow a "monitor-the-routing-paths" approach, whereby their forwarding and probing actions are restricted to the links suggested by the routing layer [5,6,7,8]. This approach limits the strategy from exploiting local caches closer but outside the routing paths. Our goal is to follow a more dynamic self-aware [9] approach for the strategy layer of the NDN architecture which actively seeks faster content delivery offered by the local content stores, while retaining the guarantees offered by the routing layer. Towards these objectives, we present a design for the strategy module of the NDN that uses the Random Neural Network (RNN) [10] based adaptive learning algorithm, inspired by the Cognitive Packet Network (CPN) [11,12].

The rest of this paper report is organised as follows. Section 1.1 reviews the related literature, focusing on the NDN strategy. We introduce our proposed strategy, the *NDN forwarding strategy using reinforcement learning with the RNN* (NDNFS-RLRNN) in Section 2. Simulation results are presented in Section 2.1, and conclusions and future work are presented in Section 3.

1.1 RelatedWork

In [13] it is shown that coupling caching and forwarding is essential in ICN to significantly benefit from ubiquitous caching. An ideal policy *ideal Nearest Replica Routing* (iNRR), forwards requests to the nearest possible replica with the help of an "oracle" that keeps track of a network's caching state, before proposing practical implementations that regularly explore a given neighbourhood in the network through scoped flooding of requests. In the NDN context, Jacobson et al. [5] proposed forwarding an Interest along all the interfaces suggested by the routing layer excluding the interface the Interest arrives on, also called the *multicast strategy*. The *best route strategy* [6] forwards interests using the available upstream with the lowest routing costs. The current NDN strategy [7,14,3] combines interface ranking and colour classification in order to decide where to forward Interests. Interfaces suggested by the routing protocol are first classified using a colour scheme according to how well they are known to return Data, then the interfaces are ranked in each class using some metric, usually the SRTT. The forwarding logic is to use the highest ranked available interface in the best possible classification. Interest NACKs were introduced in NDN to address the inefficiencies that result from the dangling states in the PIT caused by unsatisfied Interests. When it cannot forward nor satisfy an Interest, an NDN router responds with an Interest NACK; the Interest NACK also carries a code describing the reasons. Therefore, Interest NACKs can help the network to quickly and informedly detect faults and, when possible, try other forwarding options. To eliminate undetected loops in NDN, the *Strategy for Interest Forwarding and Aggregation with Hop-Counts* (SIFAH) was proposed in [8]. In

SIFAH, a node accepts to forward an Interest packet only if there exists, based on distance information to the content source, a neighbour node that moves the packet closer to the content. The distance information used in SIFAH is the number of hops to the content repository and it is provided to the forwarding plane by the routing protocol. The multipath forwarding strategies [15,16,17] dynamically assign to each interface of an NDN router, a forwarding weight or probability per name prefix, which determines the proportion of request traffic sent on each interface, so as to achieve good load balancing and manage network congestion. *INFORM* as presented in [18] and inspired by the Q-routing [19]. INFORM alternates between exploration and exploitation phases when making forwarding decisions. In the initial exploration phase when no learning has occurred, a received Interest is sent using the best interface according to the information in the FIB and a copy of the Interest is also sent on a randomly selected interface. The same actions are repeated in subsequent exploration phases except that the best interface is the one learnt by the Q-learning algorithm. Only the best interface computed after an exploration phase is used during the subsequent exploitation phase. The authors do not address the possible inefficiencies that could be introduced by the dangling states in the PIT and the handling of Interest NACKs.

Our approach uses the CPN routing protocol [20] which employs reinforcement learning using the RNN [21] to establish and maintain connections between network nodes. The algorithm used has been shown to possess fast convergence properties during an initial learning phase and good sensitivity to environmental changes [22]. Other work on CPN can be found in [23,24,25,26].

2 NDN forwarding using reinforcement learning with the RNN

In this section, we present NDNFS-RLRNN where we use this reinforcement learning with RNN algorithm to realize the strategy module per prefix in the NDN architecture. The learning algorithm is expected to be supported by a name-based routing protocol and the online measurements from the state recorded in the PIT. In NDNFS-RLRNN, an RNN is created for each prefix announced into the FIB by the routing protocol. The RNN has as many neurons as the number of outgoing interfaces at the NDN router. An RNN in the initial state only knows of the routing preferences for its prefix and is yet to be updated by packet delivery measurements. In this state, NDNFS-RLRNN's forwarding decisions will be to use the best interface according to the routing layer.

On the arrival of a Data packet, a reward value is computed for the arrival interface and used to update the RNN. The goal of our distributed reinforcement learning is to minimise the delay for retrieving contents, so we estimate reward using the RTT values. Let $T^j{}_I$ be the time an NDN router forwards an Interest along an interface j and $T^j{}_D$ be the arrival time of the Data packet satisfying the Interest which also arrives on the same interface, we can estimate the reward, R^j as the inverse of the RTT using

$$R^j = (T^j{}_D - T^j{}_I)^{-1} \tag{1}$$

In applying the reinforcement learning algorithm, it is important to consider in this application that the links suggested by the routing protocol offer better assurances on content delivery. Therefore, only positive reinforcements are applied to the neurons

corresponding to these interfaces, ensuring that alternative interfaces will be used only when they are known to improve delivery. Also, whenever an entry in the PIT expires without any response or the RNN receives no updates for T_r time units, the learning algorithm reinitializes, thereby falling back to the routing layer preferences.

For an updated RNN, the forwarding decision of NDNFS-RLRNN is, excluding the arrival interface, with a high probability p only the interface corresponding to the most excited neuron is used or with probability $(1-p)$ a probing decision is made. Before explaining the probing process, we first introduce the idea of marking an Interest packet for probing. This means an Interest can be identified at the NDN routers as either a "normal" Interest or a probe Interest. The probing process involves sending two packets: the original Interest which is sent along the best known interface and a copy which is sent as a probe Interest on a randomly selected interface. We impose that an NDN router only forwards a probe Interest along the best known interface. The motivation for this probing process is to control exploration in order to reduce the possibility of unnecessarily using longer paths to retrieve contents and to manage overhead.

In NDNFS-RLRNN, we have also adopted the use of NACKS as in the current NDN. Clearly, the reward computation is explicit when the Interest is satisfied; but, this is not the case when a NACK is returned or no response is received before the Interest times out. In such cases, the reward can be estimated as

$$R^j = [c(T - T^j{}_I)]^{-1} \tag{2}$$

where T is the current time and c is a constant large enough such that a negative reinforcement is applied.

2.1 Performance Evaluation

In this section, we present initial evaluations of the performance of NDNFS-RLRNN through extensive simulations using the *ndnSim* [27], an NS-3 based simulator which already exists for NDN. For the simulations, we consider an NDN-based network connecting users to content servers. We use the real topology *Elibackbone* shown in Figure 1 according to the dataset in [28]. Each node receives external requests for contents at

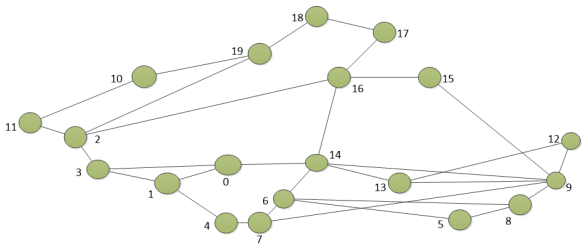

Fig. 1. *Elibackbone* topology. The network consists of 20 nodes and 30 links.

an average of 5 request packets per second with a Poisson distribution. We model the

requests according to the Mandelbrot-Zipf (MZipf) distribution. The plateau parameter for the distribution is fixed at $p = 5.0$ while the skew parameter, α at each node is randomly chosen in the range [0.6, 1.2]. The cache capacity is fixed for each node at 10 data units and we investigate performances under different catalogue sizes. The simulations begin with empty caches at the nodes. We consider a single access router for the network through which requests are served from the content servers. Furthermore, we compare NDNFS-RLRNN with the *Adaptive SRTT-based Forwarding Strategy* (ASF)[4] and a Nearest Replica Routing (NRR) strategy [13]. The ASF strategy uses the upstream with the lowest measured SRTT and probes alternative interfaces suggested in the FIB at intervals. The length of a probing interval is chosen to be 3 seconds and we install all possible routes in the FIB such that no loop exists in the forwarding. We have considered a small probing interval for ASF in comparison with a probing probability of 0.3 for NDNFS-RLRNN. For the NRR strategy, we implement a multicast algorithm that sends each request on all the interfaces of a node except the arrival interface, which guarantees that the requested data objects are retrieved from the closest caches. The strategies are implemented per content in the catalogue. Also, since the cache states in the network will influence performance, the strategies are compared under three different cache policies from the literature:

- Leave copy everywhere combined with LRU replacement policy (*LCE*).
- In the "Betweenness centrality policy and LRU (*Betw*)" [29], only the routers with the highest betweenness centrality values along the delivery path, cache the content.
- In *ProbCache and LRU* routers cache content based on some probability. *ProbCache* [30], computed by weighing the caching capacity of the delivery path with the relative position of the router along the path.

Finally, we measure athe *cache hit rate* and the *network load per request*. The cache hit rate is the proportion of the requests arriving into the network which are satisfied from the local caches. The network load or overhead is the total number of hops traversed by the network packets, which includes Interests, Data and NACK packets, per request sent into the network.

Results We present simulation results in Figure 2 where a plot on each graph represents the average value from five randomised simulation runs each lasting 400 secs; the error bars, which are included where readability is not affected, represent the standard deviations. The figures report the cache hit rate as a function of the catalogue size for three different on-path caching policies.

We observe that NDNFS-RLRNN delivers a better hit rate performance than both ASF and NRR under the LCE cache policy. It provides an average improvement of about 37% compared with ASF, and an improvement between $10-13\%$ in comparison with NRR for the considered catalogue sizes. The cache pollution caused by the flooding of requests in NRR combined with the characteristic high eviction rate of the LCE policy offsets the gains of content retrieval from the closest cache. Improved results are obtained generally as the cache policy tries to cache packets at the most "central" nodes along the delivery paths. Under the *Betw* policy, compared with ASF, NDNFS-RLRNN

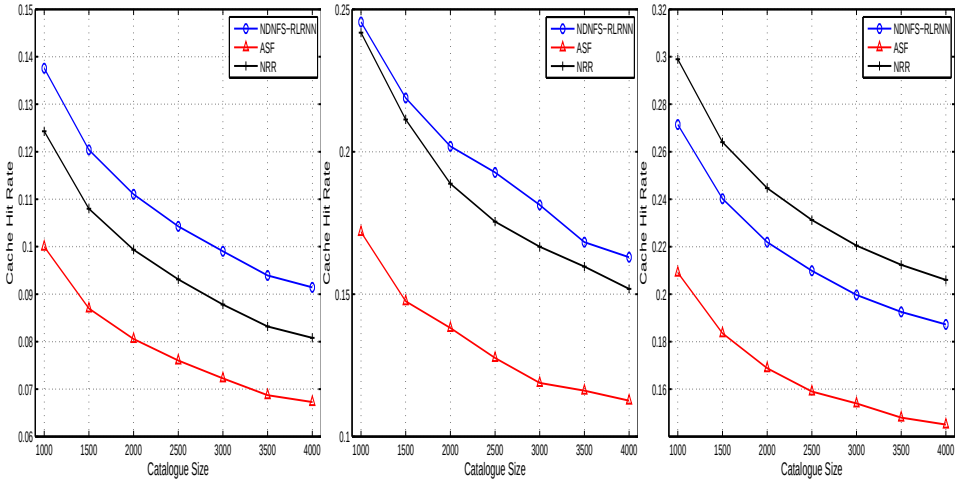

Fig. 2. Cache hit rate performance comparison of NDNFS-RLRNN, ASF and NRR under the (i) LCE and LRU cache policy (Left), (ii) *Betw* and LRU cache policy (Centre), and (iii) *ProbCache* and LRU cache policy, with different catalogue sizes. The cache size of each node is fixed at 10 data objects. Content popularity is modelled according to the Mandelbrot-Zipf (MZipf) distribution with settings $p = 5.0$ and $\alpha \in [0.6, 1.2]$. The results reported are the average of 5 randomized simulation runs.

provides an improvement between $40-53\%$ in terms of the cache hit rate While, compared with NRR, we see an improvement between $1-10\%$. The *ProbCache* policy produces the best hit results for the strategies as observed in because it uses the cache space more efficiently than the other policies. The reduced eviction rate suits the NRR as it produces the best performance in terms of the cache hit rate. NDNFS-RLRNN still provides an average performance improvement of about 30% over ASF for the considered catalogue sizes. These results suggest that when caching is effective, NDNFS-RLRNN can take better advantage compared with ASF which restricts its forwarding decisions to the delivery paths according to the FIB. Finally, Figs. 3, report sthe overhead per request as a function of the catalogue size for the different on-path caching policies. Generally, we have observed NDNFS-RLRNN generates about one-fifth of the overhead produced in NRR under all the scenarios. Also, the results show that NDNFS-RLRNN achieves better hit rates compared with ASF under all the cache policies with less overhead.

3 Conclusions

This paper has proposed an adaptive forwarding strategy, NDNFS-RLRNN for the NDN architecture. The approach is dynamic and does not persist on the links put forward by the routing protocol, so that it may better recognize the role of the forwarding plane to take advantage of the in-network caching capability of NDN. Our experiments suggest that when in-network caching is effective, NDNFS-RLRNN achieves better delivery than a strategy that persists with existing static routing layer preferences.

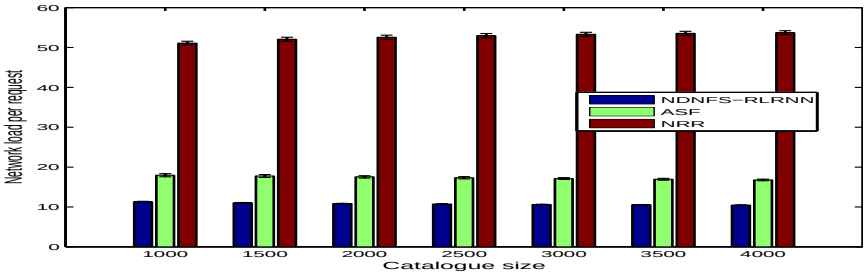

Fig. 3. Network load performance of NDNFS-RLRNN, ASF and NRR under the LCE and LRU cache policy and different catalogue sizes. The cache size of each node is fixed at 10 data objects. Content popularity is modelled according to the Mandelbrot-Zipf (MZipf) distribution with settings, $p = 5.0$ and $\alpha \in [0.6, 1.2]$. The results are averages of 5 randomized simulation runs.

References

1. Xylomenos, G., Ververidis, C.N., Siris, V.A., Fotiou, N., Tsilopoulos, C., Vasilakos, X., Katsaros, K.V., Polyzos, G.C.: A survey of information-centric networking research. IEEE Communications Surveys Tutorials **16**(2) (Second 2014) 1024–1049
2. Cisco, V.: Cisco visual networking index: Forecast and methodology 2016–2021.(2017) (2017)
3. Zhang, L., Afanasyev, A., Burke, J., Jacobson, V., claffy, k., Crowley, P., Papadopoulos, C., Wang, L., Zhang, B.: Named data networking. SIGCOMM Comput. Commun. Rev. **44**(3) (July 2014) 66–73
4. Lehman, V., Gawande, A., Zhang, B., Zhang, L., Aldecoa, R., Krioukov, D., Wang, L.: An experimental investigation of hyperbolic routing with a smart forwarding plane in ndn. In: 2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS). (June 2016) 1–10
5. Jacobson, V., Smetters, D.K., Thornton, J.D., Plass, M.F., Briggs, N.H., Braynard, R.L.: Networking named content. In: Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies. CoNEXT '09, New York, NY, USA, ACM (2009) 1–12
6. Afanasyev, A., Shi, J., Zhang, B., Zhang, L., Moiseenko, I., Yu, Y., Shang, W., Huang, Y., Abraham, J.P., DiBenedetto, S., et al.: Nfd developers guide. Dept. Comput. Sci., Univ. California, Los Angeles, Los Angeles, CA, USA, Tech. Rep. NDN-0021 (2014)
7. Yi, C., Afanasyev, A., Wang, L., Zhang, B., Zhang, L.: Adaptive forwarding in named data networking. SIGCOMM Comput. Commun. Rev. **42**(3) (June 2012) 62–67
8. Garcia-Luna-Aceves, J., Mirzazad-Barijough, M.: Enabling correct interest forwarding and retransmissions in a content centric network. In: Proceedings of the Eleventh ACM/IEEE Symposium on Architectures for Networking and Communications Systems. ANCS '15, Washington, DC, USA, IEEE Computer Society (2015) 135–146
9. Gelenbe, E.: Self-aware networks: The cognitive packet network and its performance. In: Self-Aware Computing Systems. Springer (2017) 659–668
10. Gelenbe, E., Fourneau, J.M.: Random neural networks with multiple classes of signals. Neural Computation **11**(4) (May 1999) 953–963
11. Gelenbe, E.: Steps toward self-aware networks. Commun. ACM **52**(7) (July 2009) 66–75

12. Birke, R., Cámara, J., Chen, L.Y., Esterle, L., Geihs, K., Gelenbe, E., Giese, H., Robertsson, A., Zhu, X.: Self-aware computing systems: Open challenges and future research directions. In: Self-Aware Computing Systems. Springer (2017) 709–722
13. Rossini, G., Rossi, D.: Coupling caching and forwarding: Benefits, analysis, and implementation. In: Proceedings of the 1st ACM Conference on Information-Centric Networking. ACM-ICN '14, New York, NY, USA, ACM (2014) 127–136
14. Yi, C., Afanasyev, A., Moiseenko, I., Wang, L., Zhang, B., Zhang, L.: A case for stateful forwarding plane. Comput. Commun. **36**(7) (April 2013) 779–791
15. Qian, H., Ravindran, R., Wang, G.Q., Medhi, D.: Probability-based adaptive forwarding strategy in named data networking. In: 2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013). (May 2013) 1094–1101
16. Nguyen, D., Fukushima, M., Sugiyama, K., Tagami, A.: Efficient multipath forwarding and congestion control without route-labeling in ccn. In: 2015 IEEE International Conference on Communication Workshop (ICCW). (June 2015) 1533–1538
17. Posch, D., Rainer, B., Hellwagner, H.: SAF: Stochastic adaptive forwarding in named data networking. IEEE/ACM Transactions on Networking **25**(2) (April 2017) 1089–1102
18. Chiocchetti, R., Perino, D., Carofiglio, G., Rossi, D., Rossini, G.: Inform: A dynamic interest forwarding mechanism for information centric networking. In: Proceedings of the 3rd ACM SIGCOMM Workshop on Information-centric Networking. ICN '13, New York, NY, USA, ACM (2013) 9–14
19. Boyan, J.A., Littman, M.L.: Packet routing in dynamically changing networks: A reinforcement learning approach. In: Proceedings of the 6th International Conference on Neural Information Processing Systems. NIPS'93, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (1993) 671–678
20. Gelenbe, S.E.: Cognitive packet network. US Patent (6804201) (2004)
21. Gelenbe, E.: Réseaux neuronaux alatoires stables. Comptes-rendus de l'Académie des Sciences. Série 2, Mécanique, Physique, Chimie, Sciences de l'Univers, Sciences de la Terre **310**(3) (1990) 177–180
22. Halici, U.: Reinforcement learning with internal expectation for the random neural network. European Journal of Operational Research **126**(2) (2000) 288 – 307
23. Gelenbe, E., Lent, R.: Power-aware ad hoc cognitive packet networks. Ad Hoc Networks **2**(3) (2004) 205–216
24. Sakellari, G.: The cognitive packet network: A survey. The Computer Journal **53**(3) (2010) 268
25. Gelenbe, E., Liu, P., Lainé, J.: Genetic algorithms for route discovery. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **36**(6) (2006) 1247–1254
26. Wang, L., Gelenbe, E.: Adaptive dispatching of tasks in the cloud. IEEE Trans. Cloud Computing **6**(1) (2018) 33–45
27. Mastorakis, S., Afanasyev, A., Moiseenko, I., Zhang, L.: ndnSIM 2: An updated NDN simulator for NS-3. Technical Report NDN-0028, Revision 2, NDN (November 2016)
28. Knight, S., Nguyen, H., Falkner, N., Bowden, R., Roughan, M.: The internet topology zoo. Selected Areas in Communications, IEEE Journal on **29**(9) (october 2011) 1765 –1775
29. Chai, W.K., He, D., Psaras, I., Pavlou, G.: Cache less for more in information-centric networks (extended version). Computer Communications **36**(7) (2013) 758 – 770
30. Psaras, I., Chai, W.K., Pavlou, G.: Probabilistic in-network caching for information-centric networks. In: Proceedings of the Second Edition of the ICN Workshop on Information-centric Networking. ICN '12, New York, NY, USA, ACM (2012) 55–60