Probability-Based Adaptive Forwarding Strategy in Named Data Networking

Haiyang Qian[*♯], Ravishankar Ravindran[*], Guo-Qiang Wang[*], Deep Medhi[†]
[*]Huawei Innovation Center, Santa Clara, CA USA [†]University of Missouri-Kansas City, Kansas City, MO, USA

基于概率的自适应转发；蚁群算法选择概率

将蚁群优化问题带到自适应转发

很好的实现负载均衡

Abstract—**Forwarding strategy is an important research problem in named data networking (NDN). A probability-based e forwarding (PAF) strategy is introduced in this paper. ony optimization is customized to a dynamic NDN ent to compute the selection probabilities. In addition, a statistical model in PAF to compute timeout for the nission mechanism at the NDN layer. PAF also provides rators a number of parameters to tune for different arios. Simulation results show PAF can achieve load is adaptive to network condition changes.**

能负载均衡，适应网络变化

I. Introduction

Jacobson *et. al.* proposed a Content-Centric Network (CCN) [4] architecture, also known as Named Data Networking (NDN) [8], to meet the changing needs from the host-oriented model to the content-oriented model. In an NDN router, there are three components: *Content Store* (CS), *Pending Interest Table* (PIT), and *Forwarding Information Base* (FIB). Upon receiving an *interest*, the CS of the router is checked for an exact match; if the content is available, the CS sends the requested data. Otherwise, it checks the PIT for an exact match to know whether another interest for the same content is still active; if so, the incoming face is added to the matching PIT entry. If no PIT entry is found, a new entry is created in the PIT and the interest packet is forwarded by consulting its FIB for the prefix.

重传和多播的冲突

单播可能导致重传，多播利用率低

Loop free data forwarding in NDN provides the flexibility of selecting forwarding face(s) from multiple faces, which is an inherent virtue of NDN. How to utilize this feature is left to the *strategy* layer [4]. Two naïve strategies can be used. One is to broadcast interest via all faces; the data returned from the fastest faces is accepted and the data returned from other faces is discarded. The fastest face is automatically used and the delay perceived by the NDN router is minimized. In addition, this strategy is resilient to network failure and adaptive to network condition changes. However, it introduces an inordinate amount of interests as well as multiple copies of data in the network. Multiple copies of data consume bandwidth, increase the network delay and are energy inefficient. The other strategy is to unicast interests to a random face where only one copy of data is transmitted in the network. However, the random face yields cannot guarantee best performance, e.g., minimal delay. The resilience and counter-measure for network condition change must be achieved by retransmission under the later strategy.

理想的策略是找到一个机会，即在概率意义上，将利益单播到延迟最小的face

Based on the above discussion, it is clear that the ideal strategy is to find an opportunity, that is, to unicast the interest to the face that has the least delay in a probabilistic sense. In this way, the load is automatically balanced so that performance is optimized. In addition, the strategy must be adaptive to network conditions. In this paper, *network condition* refers to all factors that have influence on the performance perceived by the routers; to name a few, network topology, network status, traffic pattern, traffic load, and content popularity. In addition to these factors, the content distribution in NDN is also prone to change due to caching and its management policies. Because files are usually chopped into small chunks in NDN, it is possible to collect statistics requiring a large population base. Therefore, we propose the *Probability-based Adaptive Forwarding* (PAF) strategy in NDN.

策略适应网络环境。网络状态指影响性能的因素。

PAF is inspired by ant colony optimization (ACO) [2], [3]; in particular, Caro and Dorigo introduced AntNet, which is an approach to the adaptive learning of routing tables in communication networks in [1]. Yi *et. al.* [7] pointed out that the interest state (new or retransmitted), control message (e.g., Interest NACK) and interface probing can be used to design an adaptive forwarding strategy. Shanbhag *et. al.* [6] proposed a service routing and service selection scheme in a service-centric network context on top of a content-centric network. By leveraging ACO, this scheme is implemented on top of CCN to manipulate FIB. Although we also use ACO, our work is targeted for the strategy layer in CCN/NDN, while providing a rigorous treatment.

可用来设计自适应转发

策略层

In PAF, ACO is used to compute selection probability. The faces are selected probabilistically so that the performance metric (*e.g.*, delay) is optimized, the load is balanced, and the network condition changes can always be detected. The timeout for retransmission is computed by our statistical model and it may be tuned to adjust the degree of explosive activities. It meets the dynamic property of NDN. We implemented our proposed strategy by modifying open source CCNx 5.0 and discuss PAF's performance. Although the cache is an important component of NDN, it is valid to consider the forwarding strategies without counting in cache capacities. The forwarding strategy problem exits no matter whether cache capacities are present or not.

有概率地选择这些face，优化性能指标(例如，延迟)，平衡负载，并检测到网络状况的变化

The rest of the paper is organized as follows. We present the PAF strategy in Section II. Experiments and results are presented in Section III, while Section IV concludes this paper and discusses future works.

II. Probability-Based Forwarding Strategy

At an NDN node, the next-hop is identified by a *face*. Functionally, we use interest and data to probe the performance (*e.g.*, delay) of faces. We use light-weight algorithms consuming low CPU time to update the selection probability for each face according to the feedback from probe interest/data.

兴趣包和数据包探测网络性能

耗时较短的轻量化算法，根据探测对象的兴趣包/数据包反馈更新每个face的选择概率

兴趣包根据转发概率发送

[♯] Now with China Mobile USA Research Center, Milpitas, CA, USA.

978-3-901882-50-0 ©2013 IFIP

通过转发频率和选择概率生成探针兴趣包。根据探测数据包返回的信息，将最新的延迟发送给选择概率处理器(SPP)。

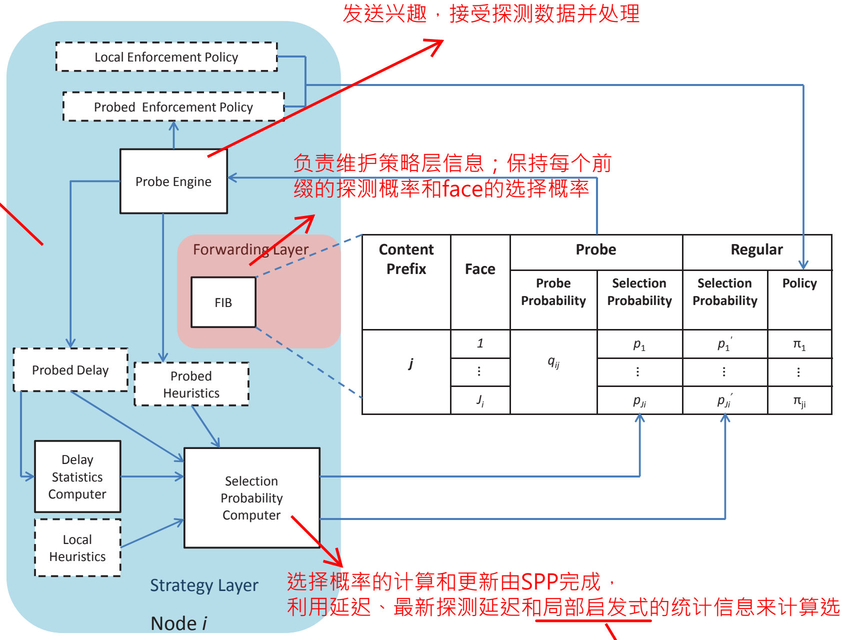
发送兴趣，接受探测数据并处理

负责管理转发策略

效率和开销折衷，有三种探测方式：1.探针由边缘节点注入，核心节点利用探针调优FIB前缀的兴趣转发

2.核心节点注入探针;这里，仅使用其探针更新其转发策略

3.核心节点利用边缘节点注入的探针调整转发策略



负责维护策略层信息；保持每个前缀的探测概率和face的选择概率

选择概率的计算和更新由SPP完成，利用延迟、最新探测延迟和局部启发式的统计信息来计算选择概率

对收到的指针用蚁群算法更新概率

Fig. 1. Functional modules in the forwarding plane

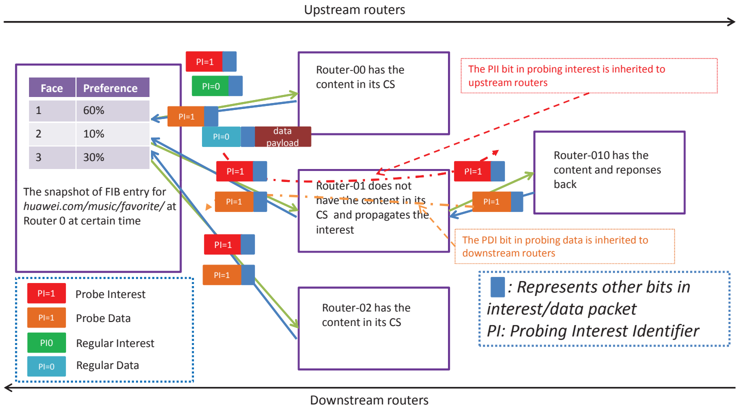

Upstream routers
Face
Preference
1
60%
2
10%
3
30%
The snapshot of FIB entry for huawei.com/music/favorite/ at Router 0 at certain time
PI=1
PI=0
data payload
Router-00 has the content in its CS
The PII bit in probing interest is inherited to upstream routers
Router-010 has the content and reponses back
Router-01 does not have the content in its CS and propagates the interest
The PDI bit in probing data is inherited to downstream routers
Router-02 has the content in its CS
PI=1 Probe Interest
PI=1 Probe Data
PI0 Regular Interest
PI=0 Regular Data
: Represents other bits in interest/data packet
PI: Probing Interest Identifier
Downstream routers

Fig. 2. Illustration of Interest Propagating and Data Receiving

Interests are sent probabilistically according to the selection probability.

Fig. 1 presents the functional modules in the forwarding plane of a node. There are two sub-layers in the forwarding plane: one is the strategy layer that manages the forwarding policy and the other one is FIB that maintains the information obtained from the strategy layer. The FIB is modified to maintain the probing probability for each prefix and selection probability per face per prefix. Probe Engine (PrE) is responsible for sending probe-interests, receiving probe-data, and post-processing them. Based on the desired tradeoff between efficiency and overhead, there are three ways in which the probes can be used. 1) Probes can be injected by the edge nodes while core nodes leverage these probes for tuning interest forwarding for the prefixes in its FIB. 2) In addition to the edge nodes, probes can also be injected by core nodes; here each node only uses its probe to update its forwarding policies. 3) A modification to the previous one is where the core nodes also leverage the probes injected by the edge node to tune its local forwarding policies.

PrE uses forwarding frequency and selection probability for probe-interest to generate probe-interests. PrE processes the information returned back from the probe-data and sends the latest delay to the Selection Probability Processor (SPP). In the

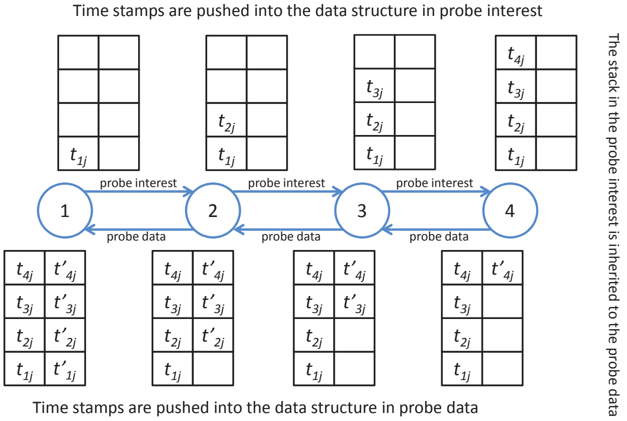

Time stamps are pushed into the data structure in probe interest
The stack in the probe interest is inherited to the probe data
probe interest
probe interest
probe interest
1
2
3
4
probe data
probe data
probe data
t1j
t2j
t3j
t4j
t'1j
t'2j
t'3j
t'4j
Time stamps are pushed into the data structure in probe data

Fig. 3. A probe-interest is initiated from node 1 and is propagated through node 1, 2, 3 and find the content in node 4

mean time, PrE informs the SPP of the probed enforcement policy. The computation and update of selection probability is performed by the SPP, which uses statistics of the delay, latest probed delay (adjusted by probed heuristics) and local heuristics to compute selection probability. More details will be given in subsections.

A node in the network is denoted by $i, i \in \mathcal{I} = \{1, 2, \cdots, I\}$. Content prefix in node i is denoted by $j, j \in \mathcal{J}_i = \{1, 2, \cdots, J_i\}$. A face for prefix j at node i is denoted by $k, k \in \mathcal{K}_{ij} = \{1, 2, \cdots, K_{ij}\}$. For prefix j at node i, $\mathcal{K}_{ij}$ is the set of faces that can be forwarded to.

A. Probe Engine

There are two modes to initiate probe-interests. PrE initiates probe-interests independently of the regular interest. The purpose of sending the probe-interest packets is not to fetch the content but to find an efficient path to the content prefix. This is called the *Independent Mode* (IM). The other option is to sample the regular interest and use the sampled interest to probe paths; this is called the *Dependent Mode* (DM). We discuss the IM first. The prefixes that match the incoming interest often are probed more frequently. The probing probability for a nameprefix is proportional to the frequency of the nameprefix being consulted, since probing probability directly reflects the frequency of the FIB lookup. Note that this frequency of consulting FIB may not (in most cases, it is not since a popular content is easier to be stored in CS) be directly related to the popular content, but related to the popularity of the content prefix. A probe frequency is added to FIB, as shown in Fig. 1. In the implementation, we keep a counter for each prefix and use an appropriate weighted moving average to adjust the counter. We use the exponentially weighted moving average (EWMA) method so that the weight for the previous counter decreases exponentially. Let Δt be the time interval between two continuous forwarding requests. In this paper, when the interval is not deterministic, Δt is used to represent the time interval between two events. Let $c_{ij}(t)$ be the most recent previous forward counter; then the next forward counter, $c_{ij}(t + \Delta t)$, is updated by

$$c_{ij}(t + \Delta t) = \alpha \times c_{ij}(t) + (1 - \alpha), \qquad (1)$$

where α is the decreasing weight of the previous counter. Whenever a probing action is initiated, the prefix probe probability is calculated by normalizing the latest counter (denoted by $c_{ij'}$) over all prefixes in real-time

$$q_{ij} = c_{ij} / \sum_{j' \in \mathcal{J}_i} c_{ij'} \qquad (2)$$

The frequency of initiating probe-interest can be tuned based on network stability (if the network condition changes a lot). For a stable network, the frequency should be low while for an unstable network, the frequency should be high. The probe interest can be probabilistically unicasted via a face according to the selection probability for the probe, or probabilistically multicasted via multiple faces according to the selection probability for the probe, or even broadcast via all faces deterministically. In the order of unicast, multicast, and broadcast, the traffic overhead increases while improving the sensitivity (adaptively) to network conditions. The broadcast mode can be used to explore new paths.

In the DM, the probe activity is in line with the prefix forwarding frequency and incurs no overhead. Thus, the probe probability is *not* used in the DM. In the IM, a bit is used in the interest/data to identify itself as a probe. This bit is named Probe Identifier (PI). If the PI is set, the interest/data is probe-interest/data. If it is not, the interest/data is a regular interest/data. The sequence of checking CS, PIT, and FIB for probes follows the same sequence of processing regular interest/data. There are two differences on handling probe and regular interests. One is that probe and regular interest for the same prefix have their own PIT entry; they also could be logically separated for efficient implementation, using the PI marking. Hence, the PIT can now be seen to have different tables for regular and probe-interests. The PIT for probe and regular interests is denoted by *PIT-P* and *PIT-R*, respectively. For the probe-interest, the data returned does not contain the real data payload to save bandwidth. This data type is called *probe-data*. When a node receives a probe-data, its CS is not updated. Only the time at which the prob data is received is recorded by PrE. In DM, the sampled interest/data are identified by the PI bit as well. The sampled interest/data is treated the same as the regular interest/data except the sending and receiving time are recorded by PrE.

If a probe-interest in either IM or DM is propagated to an upstream node and the upstream node does not have the content, the value of the PI is inherited and further propagated to upstream nodes. Fig. 2 depicts an example of propagating probe/regular interests and sending probe/regular data in IM with the unicast option (it acts similarly in DM) when the interest of `huawei.com/music/favorite` is expressed via router-0. To probe the performance of available faces, the PE of router 0 sets the PI bit in the interest and probabilistically selects one face according to the selection probability for the probe. Once an upstream router receives the probe-interest, it goes through the standard procedure of interest processing, and if the CS has the content, the router responds with the probe-data. Otherwise, the router checks the PIT. If the entry exactly matches with an entry in the PIT-P, the requesting face is added to the requesting face list of the entry in the PIT-P. If this entry is not found, a new entry is created and FIB is consulted. If there is a longest match entry in the FIB, the probe-interest is further propagated according to the forwarding strategy, and the PI is inherited. Probe-data can also be utilized to carry other information that can be collected from the data source node or any other nodes *en route*.

Next, we discuss how the raw round-trip time (RTT) is obtained for the node initiating probe-interests and also for nodes in the path from initiator to the data source. How time stamps are recorded during interest propagation and data return is exemplified in Fig. 3. Let τ_{ij} be the time stamp of the probe-interests while τ'_{ij} be the time stamp of the probe-data received at node i for prefix j. As shown in the upper part of Fig. 3, a probe-interest is initiated by node 1 and the time stamp of initiating the probe-interest, τ_{1j} is recorded in the data structure maintained in the probe-interest. At nodes 2, 3, 4, the probe-interest is propagated and the time stamps of receiving the probe-interest are recorded in the same data structure. When the probe-interest reaches a node that has the content (i.e., node 4 in Fig. 3), it is transformed into probe-data and the data structure in the probe-interest is inherited by the probe-data. At the lower part of Fig. 3, the probe-data follows exactly the same path as the probe-interest due to the inherent design of NDN. The time stamp of initiating the probe-data, τ'_{4j}, is recorded by the same data structure at node 4. If the processing time from receiving probe-interest to initiating probe-data is negligible, we have $\tau_{4j} = \tau'_{4j}$. The time stamps of receiving probe-data are recorded in the data structure at nodes 3, 2, 1 along with the path probe-data returning. The probed delay for prefix j at source node s, which is equivalent to the round trip time from source node s to destination node d, is given by

$$d_{sj} = \mathrm{RTT}_{sd} = \tau'_{sj} - \tau_{sj}. \tag{3}$$

In addition, because each node autonomously selects the face (according to the selection probability), the delay for the same prefix j at intermediate node i *en route* to the destination node d is given by

$$d_{ij} = \mathrm{RTT}_{id} = \tau'_{ij} - \tau_{ij}, i \in \mathcal{P}_{sd}, \tag{4}$$

where $\mathcal{P}_{sd}$ denotes the set of all nodes in the path from s to d inclusively. For node d, it is the processing time from receiving interest to initiate data. This delay for intermediate nodes to the destination can be collected to update the selection probability since these nodes can leverage the probe originated by node i. Note that due to the location diversity of the content, the dynamics of NDN, and probabilistic selection, these nodes may not be in the path towards content for subsequent regular interests from node s.

The above calculated delay is based on the assumption that the content corresponding to the probe-interest is not cached in any node in the path towards the content. That is a *host-oriented measurement*. However, this assumption does *not* hold if the data is cached during the time between sending the probe-interest and receiving the response. This is true for both IM and DM. If the data is cached in any of the downstream nodes other than the one responding to the probe-interest, the following modification is suggested. Let s_{ij} denote if content j is stored in node i. The information recorded in the probe-data becomes a 2-tuple: $\langle \tau'_{ij}, s_{ij} \rangle$. We have $s_{id} \equiv 1$ since the destination node is known to have the data at the moment when the probe happens. The probed delay for content j is adjusted as

$$d_{ij} = \mathrm{RTT}_{ii'} = \mathrm{RTT}_{id} - \mathrm{RTT}_{i'd} = (\tau'_{id} - \tau_{id}) - (\tau'_{i'd} - \tau_{i'd}), \tag{5}$$

where $i' = \mathrm{argmin}_{i'' \in \mathcal{P}_{sd}} \{s_{i''j} = 1\}$ represents that the node has the requested data and is closest to source node s. Note that the closest node that has content may change before the next request happens due to the dynamics of CS in NDN context. Providing such information in real-time will incur excessive exchanges, which is not desirable.

B. Selection Probability

Our algorithm is inspired by Ant Colony Optimization (ACO) [3] and is based on the algorithms used to compute the *Pheromone Table* in [1]. It is customized to align in the NDN context. For completeness, we summarize the major part of this algorithm. Each entry in the FIB maintains the mean and variance of the delay, denoted by μ_{ij} and σ^2_{ij}, respectively. Upon receiving the probe-data, the FIB updates μ_{ij} and σ^2_{ij} for prefix j. The previous and latest information is differentiated by using a moving window. We adopt the same technique, EMAW, as we used to compute the frequency of consulting FIB. This is reported as an effective model in [1] and used in estimating retransmission timeouts in TCP [5]. Let $\mu_{ij}(t)$ be the mean at time t for prefix j at router i and $\mu_{ij}(t + \Delta t)$ be the updated mean at time $t + \Delta t$. Let the delay probed at time $t + \Delta t$ be $d_{ij}(t + \Delta t)$. The updated mean of the delay is given by

$$\mu_{ij}(t + \Delta t) = \eta \times d_{ij}(t + \Delta t) + (1 - \eta) \times \mu_{ij}(t), \tag{6}$$

where η can be proportional to the elapsed time since the last update; this is because the more time elapsed, the less we should trust old information; alternately, a constant in $[0, 1]$ can be used. Similarly, the update for variance is given by

$$\sigma^2_{ij}(t+\Delta t) = \eta \times (d_{ij}(t+\Delta t) - \mu_{ij}(t+\Delta t))^2 + (1-\eta)\sigma_{ij}(t)^2. \tag{7}$$

We choose a constant value for η in the NDN context since the updating frequency is high enough to ignore the factor of elapsed time. Note that $d_{ij}(t + \Delta t)$ is the measurement of RTT for the face which is selected to send the probe at $t + \Delta t$. We cannot assert that this face is good or bad only by the fact that it is selected since the RTTs for other faces are not available to compare. (The broadcast option in the Independent Mode is an exception.) However, we can assert how good this face is by comparing $d_{ij}(t + \Delta t)$ with the statistical information of *all* recorded RTT. AntNet [1] calls this parameter as goodness, denoted by r. The goodness is a function of the most recently probed delay and statistics of delay, more specifically, the mean and the variance of the delay. To be able to apply reinforcement to the face from which the probe-data returns, the face should be enhanced to be inversely proportional to the most recently probed delay. That is, less delay causes more reinforcement. On the other hand, the goodness should be awarded more under stable conditions since stable conditions mean the path quality is unlikely to change. Namely, the reinforcement should be proportional to statistical dispersion in a certain window size, which is no larger than the effective window size. We use the goodness function from [1], reported to be the best by testing different linear, quadratic and hyperbolic combinations of the most recent delay and statistics of the delay. It may be possible that this function is problem dependent. However, finding the suitable reinforcement function is out of the scope of this paper.

We now describe the reinforcement function following [1] for our problem. Let $\mu^*_{ij}(t+\Delta t)$ be the best mean delay for prefix j at router i over sliding window with size w_s at time $t+\Delta t$. It is also the lower bound of estimated delay. Because the number of effective observations is approximately $5/\eta$, the sliding window size w_s is given by

$$w_s = 5 \times \Phi/\eta, \text{ where } 0 < \Phi \leq 1. \tag{8}$$

Φ is used to tune the window size of the historical observation. The upper bound of the estimated delay is given by

$$\mu_{ij}(t+\Delta t) + (1/\sqrt{(1-\gamma)}) \times (\sigma_{ij}(t+\Delta t)/\sqrt{w_s}), \tag{9}$$

where $\gamma \in [0,1)$ represents the confidence level. The higher the γ is, the tighter is the estimated upper bound. This results from Chebyshev's inequality, which gives a loose bound without needing to know the specific distribution. It meets our need, since there is no way to know the distribution of the delay. Even if we can predict by some technique, such as machine learning, its computational complexity makes this impractical. The difference between the upper bound and lower bound is the estimated maximum dispersion, denoted by $\theta_{ij}(t+\Delta t)$. The reinforcement, r, is defined [1] as

$$\begin{aligned} r_{ijk}(t+\Delta t) &= \beta \times \frac{\mu^*_{ij}(t+\Delta t)}{d_{ijk}(t+\Delta t)} \\ &+ (1-\beta) \times \frac{\theta_{ij}(t+\Delta t)}{\theta_{ij}(t+\Delta t) + (d_{ijk}(t+\Delta t) - \mu^*_{ij}(t+\Delta t))}, \end{aligned} \tag{10}$$

where $\beta \in [0,1]$ weights two terms. β can be tuned for different network conditions. The first term reflects the first factor (*i.e.*, how good is the probe delay compared with the best delay in the shorter term window). As can be seen, the first term is proportional to the best average delay and is inversely proportional to the new probed delay. The second term evaluates the stability of the probed delay in the shorter term window. For clarity, the second term is transformed to $1/(1+(d_{ijk}(t+\Delta t) - \mu^*_{ij}(t+\Delta t))/\theta_{ij}(t+\Delta t))$. It is clear that the smaller of the ratio of latest observed dispersion $(d_k - \mu^*)$ for probed delay and estimated maximum dispersion (θ) is, the larger this term becomes. Since estimated maximum dispersion reflects the network stability in a window of size w_s and the latest observed dispersion reflects the distance of the probed delay and best delay, the second term evaluates the reinforcement with respect to the ratio of network stability and the goodness of new probed delay. At router i, the selection probability of face k of prefix j at time t is denoted by $p_{ijk}(t)$.

The selection probability for prefix j at node i of all faces at time 0 is initialized as

$$p_{ijk}(0) = 1/K_{ij}, \tag{11}$$

where K_{ij} is the number of faces for prefix j at node i. Every time a probe-data for prefix j is returned from any face k, the selection probability of those faces in the prefix is incremented. If a probe-data comes back after Δt, the selection probability for face k is updated by

$$p_{ijk}(t+\Delta t) = p_{ijk}(t) + r_{ijk}(t+\Delta t)(1-p_{ijk}(t)). \tag{12}$$

The selection probability for all faces other than k is given by

$$p_{ij\bar{k}}(t+\Delta t) = p_{ij\bar{k}}(t) - r_{ijk}(t+\Delta t) \times p_{ij\bar{k}}(t), \forall \bar{k} \in \mathcal{K}_{ij} \setminus \{k\} \tag{13}$$

To prevent stagnation on a certain face, a threshold of the highest probability is set, denoted by ϵ. In other words, it should always be able to find an alternative path.

C. Enforcement Policy

The enforcement policy is either imposed by the forwarding strategy or by nodes along the path. For example, the forwarding policy may forward the interest to n faces simultaneously to improve resilience and delay. The probe-interest and probe-data may be used to carry the enforcement policy of the nodes on the path. A new field is added into the data structure carried by the probe-interest and the probe-data to indicate the availability of the node. Any node that is indicated as unavailable on the path will set the selection probability for the face as 0. Let $\mathcal{P}_{ijk}$ be the set of nodes along the path of request content j from node i via face k. Let $a_{ijki'}$ be the Boolean availability notation of node i' belonging to $\mathcal{P}_{ijk}$. That is,

$$\prod_{i' \in \mathcal{P}_{ijki'}} a_{ijki'} = 0 \Rightarrow q_{ijk} = 0. \tag{14}$$

Thus, if any node along the path is unavailable, the selection probability is forced to 0. Note that $\mathcal{P}_{ijk}, a_{ijki'}$, and q_{ijk} are dependent on time. The time domain notation is omitted for the sake of brevity. Any local or non-local enforcement policy may be easily added to the disclosed forwarding strategy.

D. Multiple Metrics

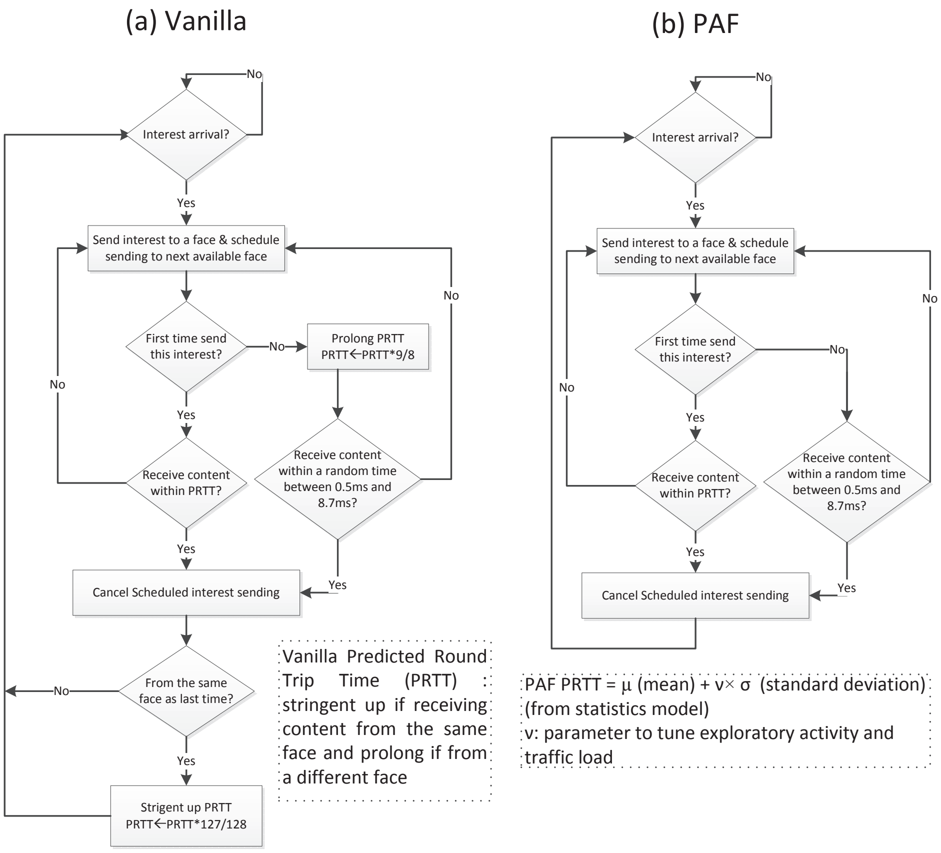
So far the delay is used as a single metric to compute selection probability. The single metric may be extended to include multiple metrics by using a composite function of weighted probed metrics. Assuming that the set of metrics to consider is $\mathcal{M}$, let d_{ijkm} be the collected value for metric m, and let d_{ijk} be the composite value for all metrics at node i requesting content j via face k and ρ_m be the weight given to metric m. Thus, we have

$$d_{ijk} = \sum_{m \in \mathcal{M}} \rho_m \times d_{ijkm}, \text{ where } \sum_{m \in \mathcal{M}} \rho_m = 1. \tag{15}$$

The local statistics may also be obtained with respect to the composite metric. The rest of the computation and updates for selection probability follow the same procedure as described for the single metric. To avoid modifying the formula to measure the goodness of the path, any other metric should be consistent with the delay metric with respect to its relationship with the goodness of the path. Namely, the metrics should be inversely proportional to the goodness of the path. This is the case with most metrics, such as link utilization and service load, which conform to this property. If the metric does not, an appropriate transformation may be made, *e.g.*, taking the reciprocal.

E. Retry Mechanism

Because each prefix may have more than one faces, the NDN strategy can build a retransmission mechanism at the NDN layer to: 1) improve resiliency in the case of link/node failures in the network, 2) use an alternate better face in the case that the current path (face) in use is overloaded, where overloading is inferred based on the delay. Retry mechanism


(a) Vanilla
(b) PAF
Interest arrival?
No
Yes
Send interest to a face & schedule sending to next available face
First time send this interest?
Prolong PRTT PRTT←PRTT*9/8
Receive content within PRTT?
Receive content within a random time between 0.5ms and 8.7ms?
Cancel Scheduled interest sending
From the same face as last time?
Strigent up PRTT PRTT←PRTT*127/128
Interest arrival?
Send interest to a face & schedule sending to next available face
First time send this interest?
Receive content within PRTT?
Receive content within a random time between 0.5ms and 8.7ms?
Cancel Scheduled interest sending

Vanilla Predicted Round Trip Time (PRTT) : stringent up if receiving content from the same face and prolong if from a different face

PAF PRTT = μ (mean) + ν× σ (standard deviation) (from statistics model)
ν: parameter to tune exploratory activity and traffic load

Fig. 4. Retry mechanism

is applied when propagating interest is part of the forwarding strategy. We summarize the retry mechanism implemented in CCNx 5.0 and our modifications.

Fig. 4(a) and Fig. 4(b) show retry mechanisms in vanilla and PAF, respectively. We first explain the heuristic implemented in CCNx 5.0. The first and second places in the face list of each nameprefix in FIB are ordered by historic information. As shown in Fig. 4(a), upon receiving an interest to propagate, this is sent via the first face on the list and at the same time it schedules the sending time via the second face (if there is one) after time interval of predicted round trip time (PRTT). If this interest is *not* sent the first time, it means the RTT is under-estimated. Thus, PRTT is prolonged by $1/8$. If the content is received within a time interval, the scheduled sending time should be canceled. If the face returning content is the same face that successfully returned content at the most recent previous time for the same prefix in the FIB, this means that the PRTT is over-estimated. Hence, PRTT is tightened by 1/128. If the content is not received within a time interval, the scheduled sending time is triggered and a retransmission time, via the next available face (if there is one) is scheduled. The time intervals mentioned above are different for two cases. If the interest is send the first time, the PRTT is used as the time interval. On the other hand, if the interest is a retrial, a random number between 0.5ms and 8.7ms is used as the time interval. This is a simple heuristic based on the assumption that once a retrial is triggered, the PRTT makes less sense and a random time should be used instead. The crucial point of this mechanism is estimating RTT. The degree of increasing and decreasing PRTT is different. Note that the vanilla strategy solely uses this retry mechanism to select changes in path and combat network conditions.

PRTT in PAF is computed as

$$\text{PRTT} = \mu + \nu \times \sigma, \qquad (16)$$

where ν controls the timeout. An inordinately long timeout is bad to necessary retransmission. An inordinately short timeout misleads the system; it causes too many unnecessary retransmissions, burdens the network, and increases delays.

III. Experiments and Results

We modified CCNx 5.0 to implement the PAF strategy in the DM. The sending interest time from a face and receiving corresponding content time from the same face is recorded to compute and update the selection probability of that face. When propagating an interest, faces are selected according to the current probabilities in the case of more than one face. NS3-DCE is used as an interface between CCNx and NS3. The network is set up in NS3. The simulation is run on a Dell PowerEdge T310 with Xeon Processor 2.53GHz and 8GB memory.

Since our work is primarily focused on proposing a strategy layer design in an NDN context, we consider the simulation of a 3-node network to be sufficient to validate the idea and to examine load balance, delay, and adaptivity performance of PAF. Node 0 is connected with node 1 and 2 over TCP over IP. Thus CCN protocol is running on top of TCP. There are 100 files named `ccnx:/file/file1` to `ccnx:/file/file100` whose size is geometrically distributed with mean 100KB. Each file is chopped into multiple 1KB chunks. Node 1 and 2 both have the copies of the 100 files. In the FIB of node 0, the entry `ccnx:/file` has two faces, 1 and 2. Node 0 requests these 100 files randomly with different request rates. When requesting a file, the interests of chunks of that file are sent in order. The link transmission rate is set as 10Mbps for both links in NS3. We change link delay to represent *change of network conditions* in our experiments. Because both the file size and request process are random, we choose 5 different seeds for each case to compute the mean and a 95% confidence interval for performance metrics. For clarity, a simulation using one seed is referred to as an *instance*.

A. Scenario: Stable Delay

In the 3-node topology, the first scenario we studied is where the link delays of links 0-1 and 0-2 are kept constant at 4ms during the entire simulation. It is referred to as the 3-node *stable* scenario. The parameter settings are as follows: $\beta = 0.7, \eta = 0.5, \phi = 0.8, \gamma = 0.9999, \epsilon = 0.999, \nu = 2$. Fig. 5 shows the selection probability for face 1 and face 2 in an instance with a request rate of 5. Note that the selection probabilities for two faces are uniformly distributed. That is, PAF can equally choose two faces. Even when selection probability is stuck at high probability for face 1 and low probability for face 2 at around 7s to 9s and 17s to 20s, it can automatically jump out of stagnation.

We compare the performance of PAF and vanilla in three aspects for this scenario. Due to the explosive transmissions and the network condition changes, there are duplicated transmissions. The effective transmission ratio (ETR) is the ratio of effective transmissions over overall transmissions. The lower this ratio is, more duplications are transmitted. Fig. 6(a) shows that the ETR of PAF is lower than the vanilla approach. We tradeoff lower ETR with more active explosive activity by tuning the parameter ϵ. The ETR of PAF is stable with regard to the change request rate compared with that of vanilla. Fig. 6(b) compares the average delay (RTT) of effective transmission for PAF and vanilla. The average delay in PAF is greater than that in vanilla. And the gap is more pronounced when the request rate is greater than 7.5. There are two reasons. First, the effective transmission ratio of PAF is lower than that of vanilla as shown in Fig. 6(a). Duplicated transmissions burden extra load on the network, and thus, increase the network delay. Note that the delay increases exponentially with respect to the traffic load and the delay is more pronounced in a higher request rate. Secondly, our modifications on the strategy layer are not in line with inherent heuristics in CCNx 5.0 since we want to minimize the modification. There is no other easy way to improve this aspect but to comply with all heuristics in the source code. The balance degree is defined as the common logarithm of the ratio of the number of transmissions between two faces. Therefore, the closer to 1 this ratio is, the more balanced flow distributed between two faces the strategy results. Fig. 6(c) represents the balance degree of two strategies. PAF achieves better balance than vanilla at a low request rate (2.5 and 5) while there is no difference in the statistical sense when the request rate is higher than 5. This is because, in the lower request rate, the explosive activity is not enough to switch faces.

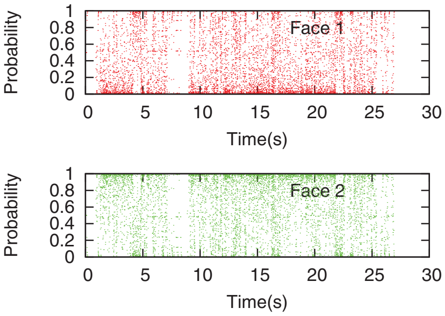


Fig. 5. The selection probability in 3-node stable scenario

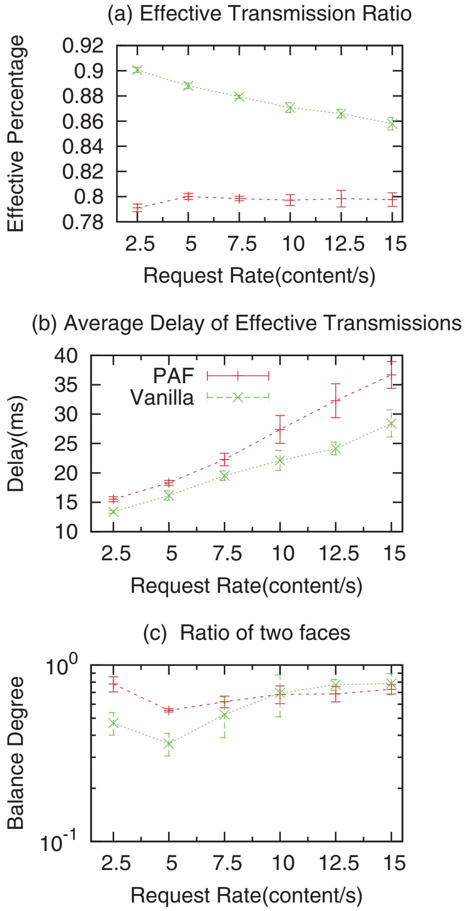


Fig. 6. Performance in 3-node stable scenario

B. Scenario: Dynamic Delay

To test the adaptivity of our approach, we also studied a network condition that changes with time. The link delay of 0-1 is set as 4ms at 0s, which changes to 2ms, 18ms, 6ms,

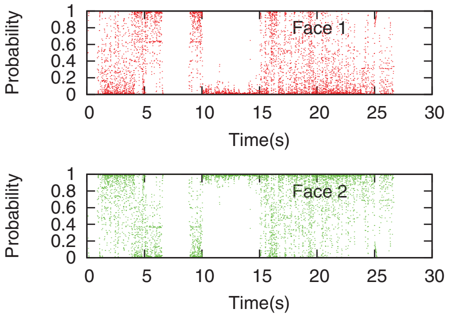


Fig. 7. The selection probability in 3-node dynamic scenario

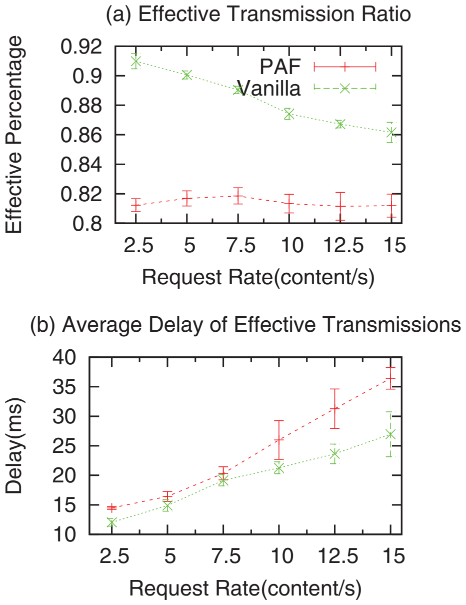


Fig. 8. Performance in 3-node dynamic scenario

8ms at 5s, 10s, 15s, 20s, respectively; however, the link delay of 0-2 is kept fixed at 6ms. We call this scenario the 3-node *dynamic* scenario. Otherwise, the parameter settings are the same as the stable scenario.

The selection probabilities of an instance with request of 5 in this scenario are depicted in Fig. 7. Note that the probability we discuss here is in the statistical sense. As we move from time 0s to 5s, the selection probability for face 2 is higher than face 1 at the beginning. The selection probability for face 2 decreases gradually while that for face 1 increases. At 4s, the selection probability for face 1 becomes higher than face 2. This shows that PAF is sensitive to the differences of face quality, even if the time difference is very small. From time 5s to 10s, the selection probability for face 1 is pushed close to 1 because the delay of link 0-1 becomes 2ms. From 10s to 15s, the selection probability is pushed close to 0 because the delay of link 0-1 is changed to 18ms at 10s. At 15s, the delay of link 0-1 is changed to 6ms. This is the equal cost path case. Thus, the selection probabilities are uniformly distributed from 15s to 20s. At 20s, the delay of link 0-1 is increased to 8ms. Thus, the selection probability of face 2 is slightly higher than face1. This shows that PAF is adaptive to the change in network conditions.

Fig. 8 compares the performance metrics for PAF and vanilla in the 3-node dynamic scenario. The performance in this scenario is consistent with one for the stable scenario. For the same reasons stated in the stable scenario, the average delay in PAF is higher than that in vanilla when the request rate is greater than 10. We also experimented with a larger topology. We observed similar results regarding the average delay.

IV. Conclusion and Future Works

In this paper, we propose the PAF strategy by customizing ACO in the NDN context. In this strategy, we probabilistically select face by its quality so that the load is automatically balanced in terms of minimizing delay while the change in the network condition can be detected. In addition, we use our statistical model to set the timeout for retransmission in the strategy layer. We validated PAF through simulation. It shows that PAF is adaptive to changes in network conditions and can be tuned to achieve good performance in different network conditions. In our simulation, all the files we requested have not been cached in requesting nodes. In the future, we plan to study the performance of PAF when files are cached in intermediate nodes.

Acknowledgement

We thank Asit Chakraborti of Huawei Innovation Center for help with understanding CCNx source code and for inspiring discussions.

References

[1] G. D. Caro and M. Dorigo, “AntNet: Distributed stigmergetic control for communications networks,” *Journal of Artificial Intelligence Research*, vol. 9, pp. 317–365, 1998.

[2] M. Dorigo, “Optimization, learning and natural algorithms,” *Ph. D. Thesis, Politecnico di Milano, Italy*, 1992.

[3] M. Dorigo and T. Stützle, *Ant Colony Optimization*. Scituate, MA, USA: Bradford Company, 2004.

[4] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, “Networking named content,” in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, ser. CoNEXT ’09. New York, NY, USA: ACM, 2009, pp. 1–12.

[5] P. Karn and C. Partridge, “Improving round-trip time estimates in reliable transport protocols,” in *Proceedings of the ACM workshop on Frontiers in computer communications technology*, ser. SIGCOMM ’87. New York, NY, USA: ACM, 1988, pp. 2–7.

[6] S. Shanbhag, N. Schwan, I. Rimac, and M. Varvello, “Soccer: services over content-centric routing,” in *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*, ser. ICN ’11. New York, NY, USA: ACM, 2011, pp. 62–67.

[7] C. Yi, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, “Adaptive forwarding in named data networking,” *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 3, pp. 62–67, June 2012.

[8] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, K. Claffy, D. Krioukov, D. Massey, C. Papadopoulos, T. Abdelzaher, L. Wang, P. Crowley, and E. Yeh, “Named Data Networking (NDN) Project, NDN-0001,” PARC, Tech. Rep., Oct. 2010. http://www.parc.com/content/attachments/named-data-networking.pdf