Advance Access publication on 6 March 2013
doi:10.1093/comjnl/bxt022

A Node-Link-Based P2P Cache Deployment Algorithm in ISP Networks

HAIBIN ZHAI[1,3,**], ALBERT K. WONG[2], HAI JIANG[1], YI SUN[1], JUN LI[1], ZHONGCHENG LI[1]

[1]*Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China*
[2]*Hong Kong University of Science and Technology, Hong Kong, China*
[3]*Graduate University of Chinese Academy of Sciences, Beijing, China*
[*]*Corresponding author: zhaihaibin@ict.ac.cn*

Peer-to-peer (P2P) systems are imposing a heavy burden on internet services providers (ISPs). P2P caching is an effective way of easing this burden. We focus on the cache deployment problem as it has a significant impact on the effectiveness of caching. An ISP backbone network is usually abstracted to a graph comprising nodes representing core routers and links connecting adjacent core routers. While deploying P2P caches at nodes (NCD, node-based cache deployment) can reduce the amount of P2P traffic transmitted from access networks to the ISP backbone network, deploying P2P caches on links (LCD, link-based cache deployment) can directly reduce the amount of P2P traffic on the ISP backbone network. However, neither NCD nor LCD maximizes the performance of P2P caches. In this paper, we propose a node-link-based cache deployment method (NLCD), which optimally selects nodes or links as deployment locations during the cache deployment process. First, we propose an analysis model and define an optimal cache deployment problem for NLCD. Then, we prove that this problem is NP complete and develop a corresponding deployment algorithm. Experimental results show that the average link utilization of NLCD is 5–15% lower than that of LCD, and 7–30% lower than that of NCD.

Keywords: peer-to-peer network; peer-to-peer traffic cache; cache deployment algorithm

Received 30 July 2012; revised 7 February 2013
Handling editor: Jongsung Kim

1. INTRODUCTION

Peer-to-peer file-sharing systems such as BitTorrent, eDonkey and eMule have become popular in recent years. Reports by CacheLogic [1] and Ipoque [2] have indicated that P2P file-sharing traffic (called P2P traffic hereafter in this paper) accounted for 60–75% of the worldwide Internet traffic in 2005 and 43–70% in 2009, respectively. A report by China Telecom [3] has indicated that P2P traffic accounted for about 55% of the China traffic in 2010. P2P traffic is consuming a large amount of network resources and imposing a great burden on internet services providers (ISPs) worldwide.

Several approaches have been proposed to ease the burden imposed by P2P traffic. These include traffic limiting [4], traffic locality [5–7] and traffic caching [8–16]. Compared with the first two approaches, caching is completely transparent to the clients and does not require changes in the P2P software. The benefits of caching for P2P traffic have been studied in many papers. Karagiannis *et al.* in [14] identify that current P2P solutions are ISP unfriendly. They suggest that deploying caches to reduce the load on ISP networks. Gummadi *et al.* in [15] investigate the differences between web caching and P2P caching. They believe that caching has a stronger opportunity to benefit P2P systems than it has in the web. Hefeeda *et al.* in [9] propose two models of cooperative caching for P2P traffic and analyze the potential gain of cooperative caching. Besides the benefits of P2P caching, caching algorithms [8, 10, 16] and measurement studies of P2P systems [10, 15] have also been reported. Caching of P2P traffic may also raise legal issues. However, discussing these legal issues is beyond the scope of this paper.

Although the deployment strategy for P2P traffic caches can have a significant impact on their effectiveness, the problem has started to receive attention only in recent years [11–13]. Ye *et al.* in [11] study the link-based cache deployment (LCD) strategy, and Kamiyama *et al.* in [12, 13] study the node-based

cache deployment (NCD) strategy. These works have shown that either deploying caches at nodes or deploying caches on links can both effectively ease the ISPs' traffic burden. However, cache locations are limited to nodes in NCD and limited to links in LCD. Can we achieve a better cache performance through a node-link combination-based deployment method?

In this paper, we propose a node-link-based cache deployment method called NLCD where backbone nodes and links are both candidate P2P cache locations. NLCD is not a simple combination of NCD and LCD. First, we propose an analysis model for NLCD. This model is different from those of NCD and LCD, where cache hit ratios are dynamically determined by real cache replacement algorithms. In addition, the changing process of the network states is analyzed in the model. Based on this model, we develop an optimal cache deployment problem and prove that this problem is NP complete. Then we propose a heuristic greedy deployment algorithm for NLCD. ISP network topologies can be classified into two typical classes as described in [12, 13]: Hub and Spoke (H&S) network and Ladder network. Experimental results show that NLCD outperforms NCD and LCD by large margins for both these two typical categories of ISP networks.

The rest of this paper is organized as follows. Section 2 gives an overview of P2P cache and introduces related works. The analysis model and algorithm design for NLCD are described in detail in Section 3. The performance evaluations of NLCD are presented in Section 4. Section 5 concludes this paper.

2. BACKGROUND AND RELATED WORKS

2.1. Overview of P2P cache

2.1.1. *ISP network diagram with P2P caches*

The architecture of an ISP network with P2P caches can be abstracted to the diagram shown in Fig. 1.

Multiple access ISP networks connect to a transit ISP network by many gateway routers. The transit ISP then connects to the rest of the Internet. While access ISPs usually deploy P2P caches at the gateway routers to reduce the amount of P2P traffic sent to transit ISPs to reduce costs, transit ISPs usually focus on the traffic load problem that P2P traffic brings to their backbone networks. In this paper, we mainly focus on the cache deployment problem for transit ISPs. For LCD, caches are deployed at position *A* which is the backbone network link of the transit ISP. This method aims to directly reduce P2P traffic within the transit ISP network [11]. For NCD, caches are located at position *B* which is the transit link from an access network to a transit ISP [12, 13], or at location *C* which is near a transit ISP router connecting an access network [24–26]. Access networks can be seen as the lower layer networks compared with the transit ISP network. NCD actually aims to reduce the P2P traffic originating from access networks and sent to the transit ISP backbone network, thus can reduce the network traffic load on transit ISP network.

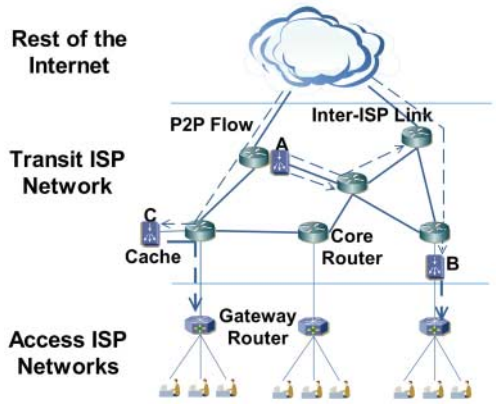

FIGURE 1. ISP Network Diagram with P2P Caches.

2.1.2. *Operation of P2P caches*

The basic operation of P2P caches is as follows. A P2P request passing through a cache is detected and interpreted. If the requested object or part of it is stored in the cache, it is served to the requesting client transparently. If a part of the requested object is not found in the cache, the P2P request is forwarded to its destination along the regular routing path without interference from the cache. The destination may be a remote server or a P2P client. Once the destination has decided to transfer the object, the cache inspects that object during the transfer. The cache will also update its stored objects for future requests according to a replacement policy such as least frequently used (LFU), least recently used (LRU) or least sent byte (LSB). In this way, a large fraction of P2P traffic is reduced as requests could be served by caches. The LSB policy considers object sizes as well as popularity, and removes objects with the smallest amount of sent data. It has been shown to achieve the highest byte hit ratio (BHR) [8, 10]. BHR is the total amount of traffic transferred from caches divided by the total amount of P2P traffic, and is usually used instead of the hit ratio to measure the cache performance of P2P caches. In this paper, we assume that ISPs adopt LSB as the cache replacement policy.

2.2. Related works

There are many studies on problems such as cache placement and cache resource allocation for web cache [17–20] and content delivery networks (CDNs) [21–23]. However, the reduction in P2P traffic on ISPs' links is usually not a major concern for web cache. The user request destination for web content is usually fixed, for example a known web server. But the request destinations of P2P content may be randomly dispersed peers around the whole network. Therefore, most web cache placement models cannot be used for P2P cache. In addition, there are several newly designed specialized replacement algorithms for P2P cache such as the LSB policy [8], and these

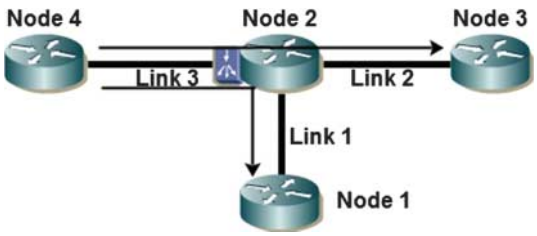

FIGURE 2. Cache deployment diagram with LCD.

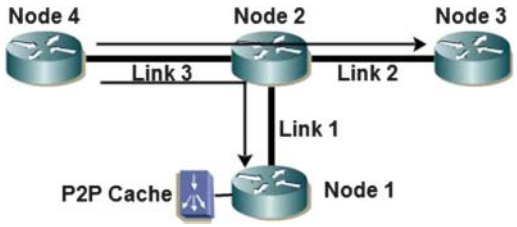

FIGURE 3. Cache deployment diagram with NCD.

replacement algorithms should be considered when designing cache deployment algorithms. In summary, the web proxy cache deployment algorithms cannot be directly used in a P2P setting.

The P2P cache deployment problem has started to receive attention only in recent years [11–13]. Ye *et al.* in [11] study the LCD strategy and propose a novel cache allocation method to maximize the benefits to transit ISPs. In their method, caches are located on links in the transit ISP backbone network. A greedy algorithm and a branch and bound algorithm are developed to provide a guideline to the deployment of caching devices. However, a simplifying assumption that caches have fixed cache hit ratios is made in this paper, thus the availability of this algorithm is decreased. To illustrate the LCD strategy, we give an example as shown in Fig. 2. We simply assume that a cache device can reduce 50% P2P traffic on a link. The P2P traffic from Node 4 to Node 1 is 400 Mb/s, while the P2P traffic from Node 4 to Node 3 is 600 Mb/s. After deploying a cache on Link 3, the P2P traffic on Link 3 changes from 1 Gb/s to 500 Mb/s, while the P2P traffic on Link 1 and Link 2 are not changed. The overall P2P traffic reduction in the network is 500 Mb/s.

Kamiyama *et al.* in [12, 13] investigate the *NCD* strategy in which caches can be regarded as being located within the nodes in the network, and propose a method to determine the optimal cache capacities and locations. In their scheme, cache locations are limited to nodes with transit links from access networks to transit ISP backbone networks, and the focus is to reduce the amount of P2P traffic transmitted from lower access networks to the transit ISP backbone. They propose a deployment algorithm based on dynamic programming and show that a strictly optimal solution can be found in polynomial time. To illustrate the NCD strategy, we give an example as shown in Fig. 3. We also assume that a cache device can reduce 50% P2P traffic. The P2P traffic from Node 4 to Node 1 is 400 Mb/s, while the P2P traffic from Node 4 to Node 3 is 600 Mb/s. After deploying a cache on node 1, the P2P traffic reduction on Link 1 and Link 3 are both 200 Mb/s, while the P2P traffic on Link 2 is not changed. The overall decreased P2P traffic in the network is 400 Mb/s.

Through experiments, we find that for four different network topologies such as Cable and Wireless (CW), Qwest, CAIS and NI, the average link utilization of LCD are 5, 2, 25 and 15% lower than that of NCD, respectively. However, the time complexity of LCD is higher than that of NCD, which will be described in detail in Section 4. Actually, the fundamental

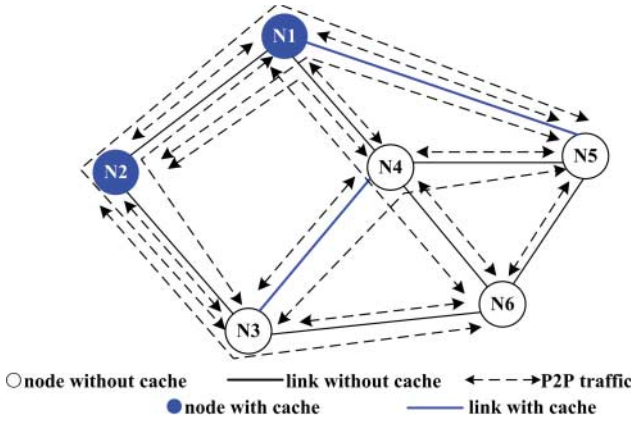

FIGURE 4. Model of P2P caching in transit ISP network.

difference between NCD and LCD is not different deployment locations, but different work processes. In NCD, the backbone network traffic load is eased by reducing the total amount of traffic transmitted from access networks. While in LCD, the load is eased by reducing the ISP traffic transmission hops and making traffic flow distribution on the backbone network more reasonable. In NLCD, both these two work processes are combined together to provide a higher performance.

3. ANALYSIS MODEL AND ALGORITHM DESIGN

3.1. Basic model

The cache deployment problem for transit ISPs is to optimally deploy a known set of caches to reduce the amount of P2P traffic within their backbone networks. We consider the case where caches are of identical sizes and leave the case of different sizes to future works. The transit ISP's network in Fig. 1 can be abstracted to a graph $G = \langle V, E \rangle$ as shown in Fig. 4.

$V = \{V_1, V_2 \ldots V_n\}$ are the nodes and $E = \{e_1, e_2 \ldots e_m\}$ are the edges. Each node corresponds to an access network. Each link corresponds to the aggregated connectivity between two adjacent routers. Each bidirectional dashed arrow in Fig. 4 refers to two reciprocal P2P flows with origin-destination nodes reversed.

Let the two sides of a link e_g be v_1 and v_2, which are the two routers connected by e_g. As shown in Fig. 5a, when a cache is

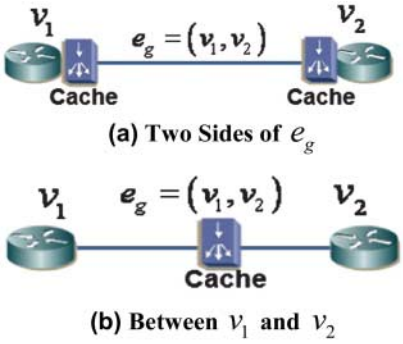

FIGURE 5. Candidate cache locations on link e_g.

deployed for e_g, it can be deployed at the v_1-side, the v_2-side or both sides. Although the cache deployed for LCD is very much a part of the core router, it works differently from those in NCD—it monitors and serves P2P traffic in both directions on e_g only. Deploying the cache in the middle of the link, as also shown in Fig. 5b, would mean breaking the original link and there is no reason to consider such an option.

Therefore, there are two candidate locations to deploy cache for each link which are the two ends of the link. For example, the candidate locations for e_g are v_1 and v_2. All these candidate locations on links and the node set V comprise the candidate location set U which contains $|V| + 2|E|$ elements. Let $F(S)$ denote the achieved ISP cost by deploying caches at a location set S, which is a subset that contains k locations chosen from U. ISPs can define the cost function according to their own requirements and preference. In this paper, we consider two classical definitions of the ISP cost function as follows:

$$F(S) = \frac{1}{r}\sum_{i=1}^{r}\frac{Y_i - \Delta Y_i}{C_i} \tag{1}$$

$$F(S) = \sum_{i=1}^{r}(Y_i - \Delta Y_i) \tag{2}$$

where C_i is capacity of link i, Y_i P2P traffic load on link i, ΔY_i P2P traffic reduction on link i after caches are deployed over subset S and$r$ is the total number of locations in S.

The cost described in (1) is the average link utilization function, and the cost described in (2) is the total amount of P2P traffic after cache deployment in the network. The part $((Y_i - \Delta Y_i)/C_i)$ is the link utilization of link i after caches are deployed over S. The cache deployment problem in NLCD is to choose the subset S such that $F(S)$ is minimized under a given P2P traffic pattern, which can be formulated as follows:

$$\begin{aligned} \min_{S} \quad & F(S) \\ \text{s.t.} \quad & S \in \mathrm{U} \\ & |S| = k \end{aligned} \tag{3}$$

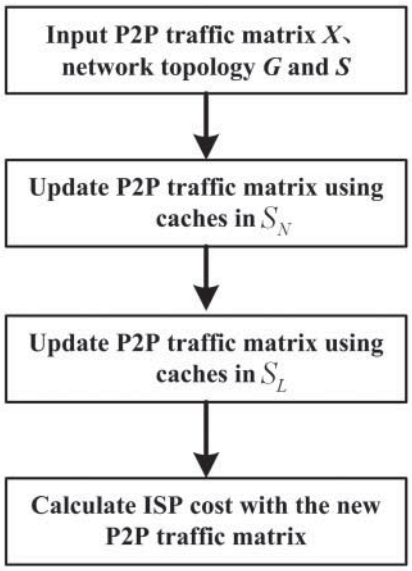

FIGURE 6. Calculation process diagram of ISP cost.

3.2. ISP cost calculation

In NLCD, S can be divided into two main parts: node location subset S_{N} and link location subset S_{L}. The ISP cost calculation process is shown in Fig. 6.

3.2.1. Update P2P traffic matrix using caches in S_{N}

Two steps are needed to update P2P traffic matrix using caches in S_{N} : the first is cache object selection and the second is P2P traffic matrix update.

(1) *Cache object selection*

Each cache must choose objects to store based on the replacement policy adopted. Let X_{ij} denote the initial volume of P2P traffic originating from node V_i and ending at node V_j, and let X_{ij}^q denote the part in X_{ij} that is due to the qth object. We use T_g^q to represent the sent bytes of an object with popularity rank q at V_g. It is calculated as

$$T_g^q = \sum_{\{h \neq g, h \in V\}} X_{hg}^q \tag{4}$$

As mentioned earlier, we assume that caches have the same capacity and each cache stores objects using the largest sent bytes (LSB) policy under the cache capacity constraint. This sent bytes value can be calculated by (4). We denote the set of cached objects in the cache at v_g as B_g.

(2) *P2P traffic matrix update*

For any node $V_g \in S_N$, X_{hg}^q becomes zero if the cache at V_g stores object q. Because all requests for object q will be

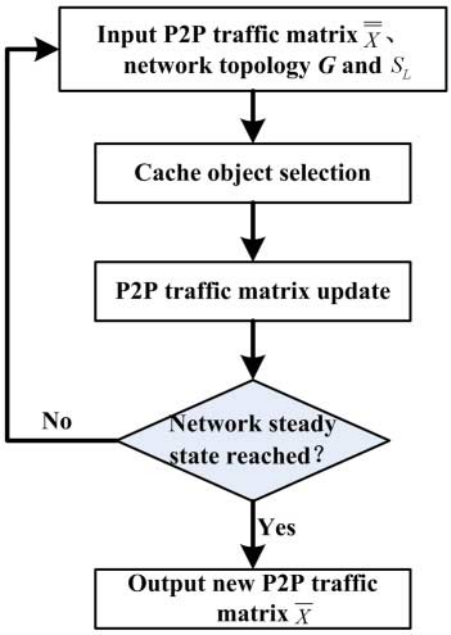

FIGURE 7. Traffic matrix update process using link caches.

served by cache and will not be transmitted across the backbone network. Then X_{hg} can be updated as follows:

$$X_{hg} = X_{hg} - \sum_{\{q \in B_g\}} X_{hg}^{q} \tag{5}$$

Other P2P traffic flows in the P2P traffic matrix are updated in the same way. Let the new P2P traffic matrix updated by the caches in S_N be $\overline{\overline{X}}$.

3.2.2. Update P2P traffic matrix using caches in S_L

Compared with the updating process with S_N, the traffic matrix update using caches in S_L is more complex. Because cached objects in the caches on links are dynamically changing during the deployment process. The P2P traffic matrix update ends after the network steady state is reached. The detailed process is shown in Fig. 7.

(1) *Cache object selection*

We use $\overline{\overline{T}}_g^q$ to represent the total number of bytes served for the qth object in the cache if it is deployed on link e_g. It is calculated as:

$$\overline{\overline{T}}_g^q = \sum_{\{i,j|e_g \in Path_{ij}\}} \overline{\overline{T}}_{ij}^q \tag{6}$$

As mentioned earlier, we assume that caches have the same capacity and each cache stores objects using the LSB policy under the cache capacity constraint. This sent byte values can be calculated by (6). The calculation process is the same as that of a node cache. Note that the number of caches deployed on e_g can be either *one* or *two*. We denote the set of cached objects in the caches on as $\overline{\overline{B}}_g$.

(2) *P2P traffic matrix update*

Each traffic flow $\overline{\overline{X}}_{ij}$ whose routing path Path_{ij} includes link $e_g = (v_1, v_2)$ can be updated as follows. Suppose $\overline{\overline{X}}_{ij}$ passes through v_1 first and then through v_2 according to Path_{ij}.

If the cache is deployed at v_1-*side*, for any $m \in \overline{\overline{B}}_g$ we have

$$\begin{aligned} \overline{\overline{X}}_{v_1 j}^{m} &= \overline{\overline{X}}_{v_1 j}^{m} + \overline{\overline{X}}_{ij}^{m} \\ \overline{\overline{X}}_{ij}^{m} &= 0 \end{aligned} \tag{7}$$

If the cache is deployed at v_2-*side* or *both sides*, for any $m \in \overline{\overline{B}}_g$ we have

$$\begin{aligned} \overline{\overline{X}}_{v_2 j}^{m} &= \overline{\overline{X}}_{v_2 j}^{m} + \overline{\overline{X}}_{ij}^{m} \\ \overline{\overline{X}}_{ij}^{m} &= 0 \end{aligned} \tag{8}$$

Other P2P traffic flows in the P2P traffic matrix are updated in the same way.

(3) *Network Steady-State Analysis*

As caches deployed on links are affected by each other, the P2P traffic matrix and the cached objects may keep on changing. We give the following definition:

DEFINITION 1. *The network steady state is the state where the P2P traffic matrix X and the cached objects in all caches are stable.*

Cached objects of caches at nodes are determined only by the traffic originating from the local access network of this node, and will not be affected by other caches. Thus the network steady state can be reached immediately after caches are deployed at nodes. In contrast, cached objects of caches on links are dynamically changing during the deployment process as caches on links may affect each other. We will show the convergence speed of our algorithm in Section 4.2.

Let $\overline{X}_{ij}$ denote the traffic from V_i to V_j, and $\overline{B}_g$ denote the cached objects of the cache on e_g when the network reaches the steady state. Based on $\overline{X}_{ij}$ and $\overline{B}_g$, the ISP cost can be calculated in the following section.

3.2.3. ISP cost calculation with new P2P traffic matrix

When the network steady state is reached, the ISP cost $F(S)$can be calculated. Let $\text{Path}_{ij} = \{e_{i1}, \ldots, e_g, \ldots, e_{ik}\}$. If the cache is deployed at v_1-*side*, $Path_{ij}$ is split into the two parts as follow:

$$\begin{aligned} \text{Path}_{ij}^{\text{front}} &= \{e_{i1}, \ldots e_{g-1}\} \\ \text{Path}_{ij}^{\text{end}} &= \{e_g \ldots, e_{ik}\} \end{aligned} \tag{9}$$

If the cache is deployed at v_2-*side* or *both sides*, Path_{ij} is split into:

$$\begin{aligned} \text{Path}_{ij}^{\text{front}} &= \{e_{i1}, \ldots e_g\} \\ \text{Path}_{ij}^{\text{end}} &= \{e_{g+1} \ldots, e_{ik}\} \end{aligned} \tag{10}$$

$\overline{X}_{ij}$, when passing through e_g, will be affected if caches are deployed on some locations in $\text{Path}_{ij}^{\text{front}}$, but will not be affected if caches are deployed in $\text{Path}_{ij}^{\text{end}}$. Let B_{ij}^{front} be the set of distinct objects stored by caches deployed in $\text{Path}_{ij}^{\text{front}}$. We have

$$B_{ij}^{\text{front}} = \bigcup_{g \in \{\text{Path}_{ij}^{\text{front}} \cap S\}} \overline{B}_g \tag{11}$$

The amount of P2P traffic reduction on link e_g after caches are deployed over S can be calculated as follows:

$$\Delta Y = \sum_{i,j \in V} \sum_{m \in B_{ij}^{\text{front}}} \overline{X}_{ij}^{m} \tag{12}$$

The amount of P2P traffic reduction on other links can be calculated in the same way.Then the ISP cost can be computed by substituting (12) into (1) or (2).

3.3. Problem complexity analysis

LEMMA 1. *The cache deployment problem in NLCD is NP complete.*

Proof. First, we must show that the NLCD cache deployment problem is in NP. The certificate is simply the assignment of S. It is easy to count the number of medians and to calculate the total cost given the medians. Then we show that the problem is also NP-hard using a reduction from the *k-median* problem. That is, we will show that:

k-median $\leq_{\text{p}}$ The cache deployment problem in NLCD.

The classic *k-median* problem can be described as follows:

Input: The set of facilities F, the set of clients C and the distance metric; also an integer k which is the maximum number of facilities that can be opened.
Output: A set $D \subseteq F$ such that $|F| \leq k$, and $\sum_{i \in C} \text{dist}(i, D)$ is minimized.

The reduction process is as follows:

Consider a fully connected network. Suppose that there is only *one* file in the network, and the capacity of each cache is also *one*. Assume utilization function (1) is used. The path between any pair of nodes, according to shortest path, is a distinct single link. As there is only *one* file in the network, at most *one* cache needs to be deployed for each link. Therefore, the candidate location set U equals the link set E here. We treat the links in E as both '*facilities*' and '*clients*'. Let link i be a client with two ends (i_1, i_2), then $X_{i_1 i_2} = X_{i_2 i_1}$. The traffic reduction on link i if a cache is deployed on it will thus be $X_{i_1 i_2}$ according to (9). Suppose that S is the location subset chosen from U to deploy caches, and $T = \{i, i_1, i_2\}$. Then the distance between any client i and its nearest facility in S is calculated as:

$$\text{dist}(i, S) = \begin{cases} \dfrac{2X_{i_1 i_2}}{C_i} & \text{if } |T \cap S| \geq 2 \\ \dfrac{X_{i_1 i_2}}{C_i} & \text{if } |T \cap S| = 1 \\ 0 & \text{if } |T \cap S| < 1 \end{cases} \tag{13}$$

The cache deployment problem in NLCD can then be reduced to a *k-median* problem as follows:

Input: The set of facilities U, the set of client E, the distance metric as described in (13), and an integer k which is the maximum number of facilities that can be opened.
Output: A set $S \subseteq U$ such that $|S| \leq k$, and $\sum_{i \in E} \text{dist}(i, S)$ is minimized.

Since the *k-median* problem is NP complete, the cache deployment problem in NLCD is NP complete. □

3.4. Deployment algorithm design

In this section, we present a heuristic greedy algorithm for NLCD. We choose the greedy algorithm as it has been shown to be very effective for cache deployment problems in previous works [11, 18]. It has even been shown that the performance of the greedy algorithm is close to optimal [18]. The basic idea of the greedy algorithm is as follows. To choose k locations among the potential locations, we choose one location at a time. In the first iteration, we compute the achieved cost associated with each location of the potential locations and pick the location that yields the smallest cost. In the second iteration, we look for a second location which, in conjunction with the location already picked, yields the smallest cost. The algorithm iterates until k nodes are chosen.

The greedy algorithm is shown in Table 1, where one location is chosen from U and put into set S in each iteration until the total number of caches is reached. During each iteration, we calculate the achieved cost in steady state for each union of S with one candidate location in U (lines 3–9). The location with the smallest cost is extracted from U and put into S (lines 10–11). The output of this algorithm is the deployment set S and its cost $F_{\min}$. According to Section 3, it is easy to derive that the time complexity of the ISP cost calculation in line 4 is $O(N^2)$, where N is the total amount of nodes in the network. Then the time complexity of the deployment algorithm is $O(N^2 k|U|)$, where $|U|$ is the number of candidate locations. $|U|$ equals $N + 2|E|$.

For small scale networks, other algorithms such as the branch and bound algorithm [14] may be used to find a better solution. First, calculate a solution using the greedy algorithm. Then, consider the greedy solution as an initial lower

TABLE 1. A heuristic greedy deployment algorithm.

Notations
U: Set of candidate locations
S: Set of locations chosen to deploy caches
k: Number of locations chosen to deploy caches

Initialization:
1. $S \leftarrow \emptyset$;
2. $F_{min} \leftarrow \infty$;

Deployment Algorithm:
1. for $i = 1$ to k
2. $U_{tmp} \leftarrow U$;
3. for each location L in U_{tmp}
/*calculate the cost when deploying caches in $S \cup L$*/
/* the calculation process is shown in Section 3*/
4. $F \leftarrow calculateCost(S \cup L)$;
/*F_{min} stores the smallest cost in this iteration */
/*L_{min} stores the optimal location selected in this iteration */
5. if $F < F_{min}$ then
6. $F_{min} \leftarrow F$;
7. $L_{min} \leftarrow L$;
8. end if
9. $U_{tmp} \leftarrow U_{tmp} \backslash L$;
10. end for
/* put the optimal selected location L_{min} into set S */
11. $S \leftarrow S \cup L_{min}$;
/* remove L_{min} from U */
12. $U \leftarrow U \backslash L_{min}$;
13. end for

bound, and looking for a better solution using the branch and bound algorithm. However, the branch and bound algorithm is unsuitable for large-scale networks as the running time is too long.

4. PERFORMANCE EVALUATIONS

4.1. Simulation setup

The ISP network topology is classified into two typical classes as described in [12, 13]. If the relative maximum node degree is greater than or equal to 0.4, or if the average node degree is greater than or equal to 3, the network is classified as Hub and Spoke (H&S). Otherwise, the network is classified as Ladder. Data concerning backbone networks of commercial ISPs is publicly available at the CAIDA web site [28]. For evaluation, we choose two H&S networks—CW and Qwest, and two Ladder networks—Netrail Incorporated (NI) and CAIS Internet (CAIS). The chosen network topologies are shown in Fig. 8. We want to confirm that our deployment algorithm works well for both types of topologies. We assume that the routing path in an underlay network is the minimum hop route. The routing path can easily be found using Floyd's algorithm [29].

The dynamics of the network such as peer join or peer churn is not significant parameters for the cache deployment algorithm. The P2P traffic of each node, the total number of objects and the object popularity distribution are more important. In accordance to [30], we assume that the traffic generated by users in access network i is proportional to the ratio of the population covered by network i versus the total population. Then the P2P traffic flow X_{ij} from node V_i to node V_j is proportional to $P_i * P_j$, the product of the two population ratios. We set the number of objects to $|M| = 1000$. It is reported that P2P traffic obeys the Mandelbrot–Zipf distribution. This distribution was first observed by Saleh and Hefeeda in [10], and confirmed by Carlinet *et al.* in [27]. Under this distribution, the relative popularity $p(i)$ of object i is given by $p(i) = a/(i+q)^b$, where a is the normalized constant, i.e. $a = 1/\sum_{i=1}^{M} p(i)$, b the parameter determining the skewedness of the distribution and q is the parameter determining the decrease ratio of access frequencies of popular content compared with those of the Zipf distribution. We set C_i, the capacity of each link, so that the initial link utilization of P2P traffic will be 50–70%. It was reported that the object size distribution in BitTorrent network can be summarized roughly as follows: 8% of 10 Mbyte, 28% of 100 Mbyte, 40% of 300 Mbyte, 11% of 700 Mbyte and 13% of 1.4 Gbyte [12]. We randomly set the size of each object according to this size distribution, and set the size of each cache to 50 Gbytes. In this section, three cache deployment algorithms are analyzed and compared:

NCD: the node-based cache deployment algorithm proposed in [12].
LCD: an improved version of the link-based cache deployment algorithm proposed in [11], which removes the simplifying assumption that caches have fixed cache hit ratios.
NLCD: the algorithm proposed in this paper.

4.2. Evaluation of NLCD

4.2.1. *A sample optimal solution of NLCD*

First, we give a sample optimal solution of NLCD. We fix the ISP cost function as the average link utilization and the network topology as that of CW. When *eight* caches are deployed, the optimal cache location selected in each iteration is shown in Table 2. In the first iteration, the optimal cache location is the San Bernardino side of link (San Bernardino, Dallas). Then the type of the optimal cache location in the second iteration becomes node Los Angeles. This proves that whether deploying caches at nodes or on links is better is dynamically changing during the deployment process. Therefore, the performance of P2P caches cannot be maximized whether limiting cache locations to just nodes or links. In the third to eight iteration, the following positions are chosen in sequence: node Chicago, the Houston side of link (Dallas, Houston), the Greensboro side of link (Austell, Greensboro), the Chicago side of link

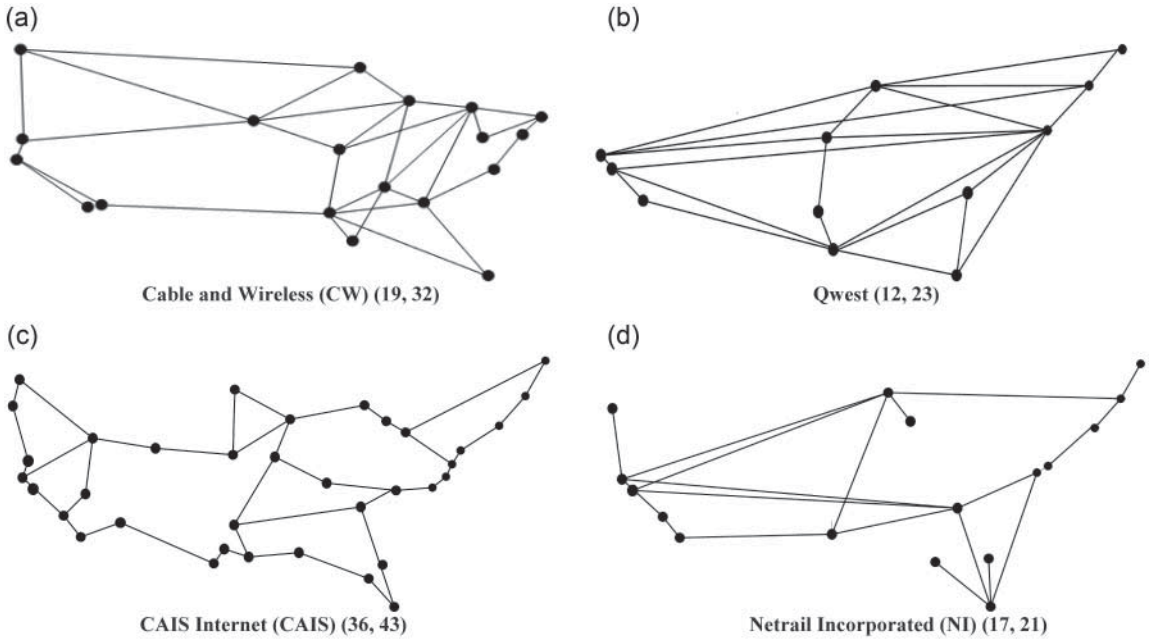

FIGURE 8. Network topologies used for evaluation. In each topology, value of (x, y) indicates (no. of nodes, no. of links). (a) Cable and wireless (CW) (19, 32), (b) Qwest (12, 23), (c) CAIS Internet (CAIS) (36, 43) and (d) Netrail Incorporated (NI) (17, 21).

(Chicago, Independence), the Seattle side of link (Seattle, San Jose) and the Los Angeles side of link (San Jose, Los Angeles). As described in Section 2, LCD outperforms NCD. Therefore, an intuitive conclusion can be achieved that the number of caches deployed on links should be larger than the number of caches deployed on nodes in NLCD. The result in Table 2 confirms this conclusion: when *eight* caches are deployed, *six* caches are deployed on links while only *two* caches are deployed on nodes.

The diagram of the sample optimal solution is shown in Fig. 9. We find that when deploying a cache on a link, it is usually better to deploy it at the end with a larger population city. However, this is not always the truth. For the link between Dallas and Houston, the cache is deployed at the Dallas side although the population of Houston is larger than that of Dallas. Because there are many factors that determine the optimal cache deployment locations such as the P2P traffic matrix, the cached objects of each cache, and the routing path.

TABLE 2. Optimal Cache Location Selected in Each Iteration.

Number of the deployed caches	Optimal cache location selected in this iteration	Type of the optimal cache location
0	Link: (San Bernardino, Dallas) Side: San Bernardino	Link
1	Los Angeles	Node
2	Chicago	Node
3	Link: (Dallas, Houston) Side: Houston	Link
4	Link: (Austell, Greensboro) Side: Greensboro	Link
5	Link: (Chicago, Independence) Side: Chicago	Link
6	Link: (Seattle, San Jose) Side: Seattle	Link
7	Link: (San Jose, Los Angeles) Side: Los Angeles	Link

4.2.2. *Execution efficiency of NLCD*

Each iteration in the greedy algorithm in Table 1 involves an ISP cost calculation step as described in Fig. 6, which is itself an iterative process for reaching network steady state (in cached objects distribution). Therefore, we want to investigate the convergence speed towards network steady state. In Fig. 10, we show the resulting average link utilizations for different number of deployed caches using 1 to 200 iterations in the process in Fig. 6. We observed that the average link utilization decreases rapidly initially, but changes only slightly after about 20 iterations. This means that near-steady state can be reached with a small number of steps and the ISP cost can be computed quickly. This process is easy to understand. At first, empty caches are deployed in the network. Then a large number of objects are starting to be cached and fill the caches in a short time interval. Many requests can now be served by caches and the average link utilization decreases rapidly. After that, the

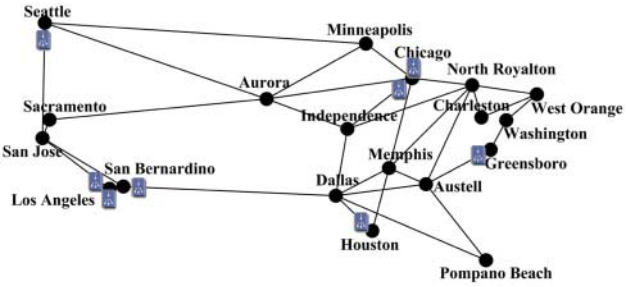

FIGURE 9. A sample solution for CW with eight caches deployed.

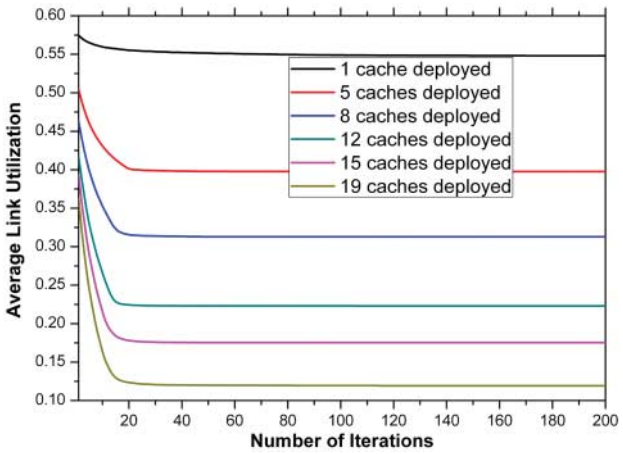

FIGURE 10. ISP cost with different number of iterations.

change in cached objects becomes smaller as there are only some object replacement occurs. Correspondingly, the decrease in average link utilization is slower. When the network steady state is reached, the cached objects in all caches are stable and the average link utilization is stable too.

4.2.3. *Evaluation of different ISP cost functions*

We fix the network topology as that of CW and investigate both forms of ISP cost functions as described in (1) and (2). The ISP cost results of (1) and (2) with different number of caches are shown Figs 11 and 12, respectively. The number of caches increases from 1 to $\min(|V|, 2|E|)$ in both Figs 11 and 12. For the network CW, we have $\min(|V|, 2|E|) = |V|$ and $|V| = 19$. As shown in Figs 11 and 12, both the average link utilization and the total amount of P2P traffic in the network decrease as the number of deployed caches increases. In addition, NLCD outperforms both LCD and NCD under both ISP cost functions. For example, the average link utilization of NLCD is 8% lower than that of LCD, and 13% lower than that of NCD. We have also tried some other forms of cost functions and found that NLCD outperforms LCD and NCD no matter what ISP cost functions are used.

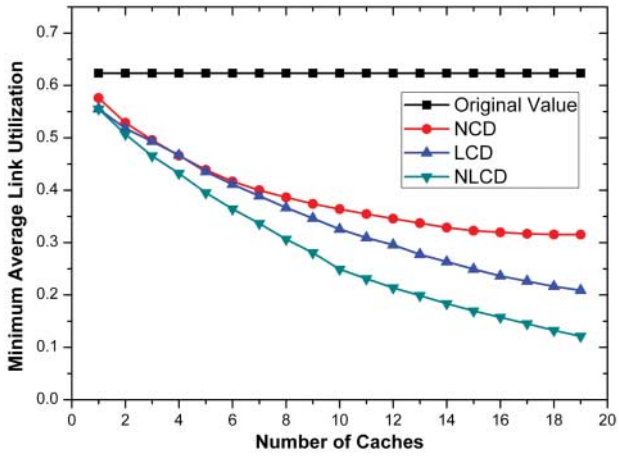

FIGURE 11. Min average link utilization for CW.

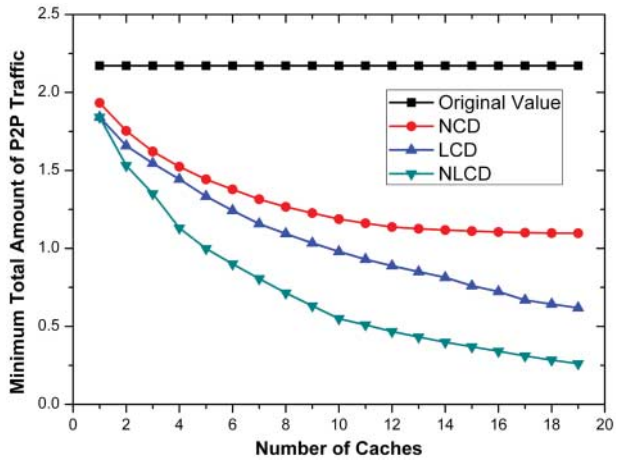

FIGURE 12. Min total amount of P2P traffic for CW.

4.2.4. *On the effects of the network traffic load*

Using the topology of CW, the optimum P2P traffic load on links with different number of caches is shown in Fig. 13. We can see that the P2P traffic load has been dramatically reduced compared with the original load. For example, the maximum link traffic load decreases by 84% when 19 caches are deployed. As the number of caches increases, the traffic load on links decreases too. In addition, the traffic load difference on different links is also reduced, and the distribution of traffic load on the network is more reasonable. Thus NLCD is very effective in decreasing the ISP network traffic load.

4.3. Comparison with other algorithms

4.3.1. *Performance comparison*

We first fix the ISP cost function as the average link utilization (equation (1)), and compare NLCD, LCD and NCD using different network topologies. The results of minimum average

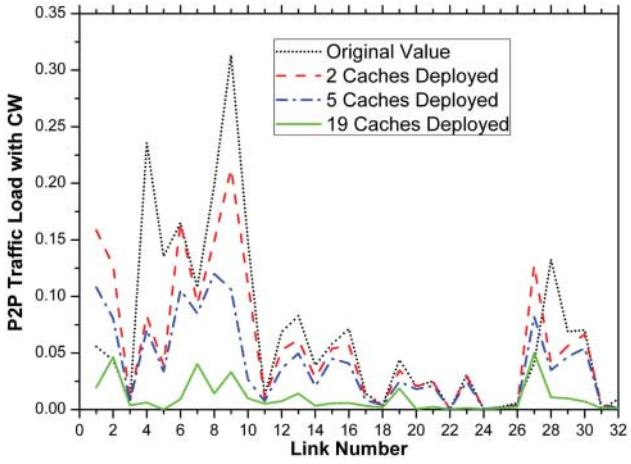

FIGURE 13. P2P traffic load on links for CW.

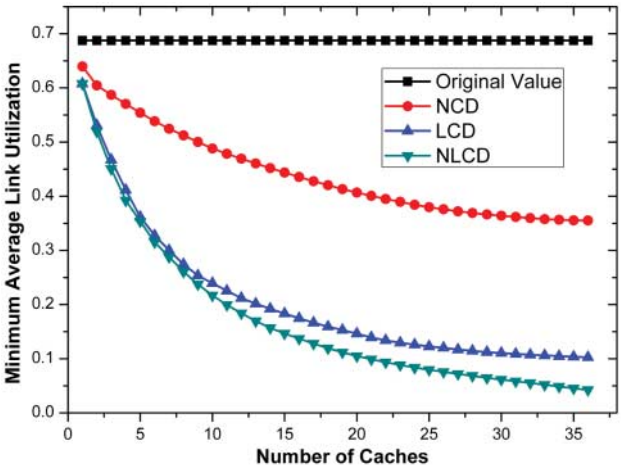

FIGURE 15. Min average link utilization for CAIS.

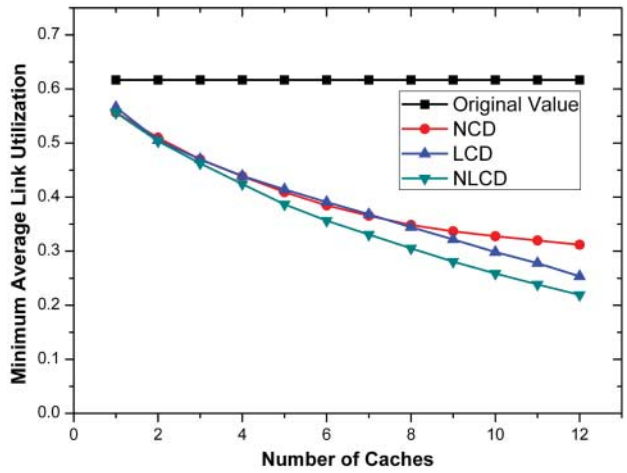

FIGURE 14. Min average link utilization for Qwest.

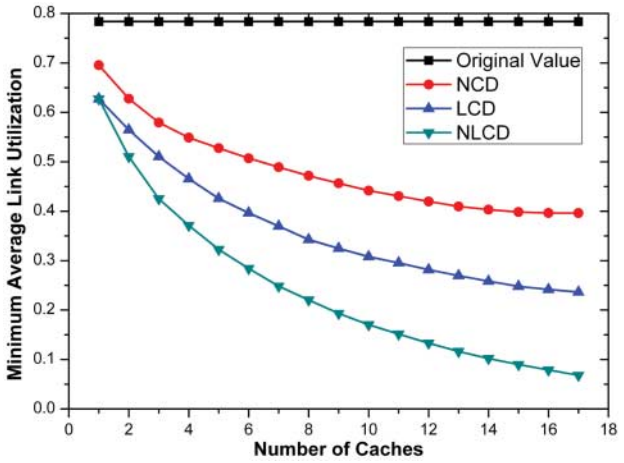

FIGURE 16. Min average link utilization for NI.

link utilization for two H&S networks, CW and Qwest, are shown in Figs 11 and 14, respectively. The results of minimum average link utilization for two Ladder networks, CAIS and NI, are shown in Figs 15 and 16, respectively. The number of caches increases from 1 to $\min(|V|, 2|E|)$ in all these figures. For example, we have $\min(|V|, 2|E|) = |V|$ and $|V| = 36$ for CAIS as shown in Fig. 15. We can see that NLCD always outperforms both LCD and NCD. The same conclusion can be reached when using other H&S or Ladder network topologies. In addition, the average link utilization difference between different algorithms remains stable as the number of caches increases. For four networks CW, Qwest, CAIS and NI, the average link utilization of NLCD is 8, 5, 5 and 15% lower than that of LCD, respectively, and 13, 7, 30 and 30% lower than that of NCD, respectively. We then fix the ISP cost function as the total amount of P2P traffic (equation (2)) and make a comparison between NLCD, LCD and NCD. The results of minimum total amount of P2P traffic for two H&S networks, CW and Qwest, are shown in Figs 12 and 17, respectively. The results of minimum total amount of P2P traffic for two Ladder networks, CAIS and NI, are shown in Figs 18 and 19, respectively. The same conclusion can be achieved: NLCD always outperforms both LCD and NCD. The reason behind this result is as follows. In NCD, caches are deployed at nodes with the aim of reducing the amount of P2P traffic transmitted from lower access networks to the ISP backbone network. Hence, the effectiveness of caches is dispersed to all the links of the network. In contrast, the first step of LCD is to find the links with the heaviest traffic load. Then caches are deployed on these links. In this way, caches can maximize their effects, and thus can reduce more P2P traffic load for ISPs. In NLCD, deployment locations are optimally selected according to the dynamic changing of P2P traffic matrix during the cache deployment process. Therefore, NLCD can outperform both LCD and NCD.

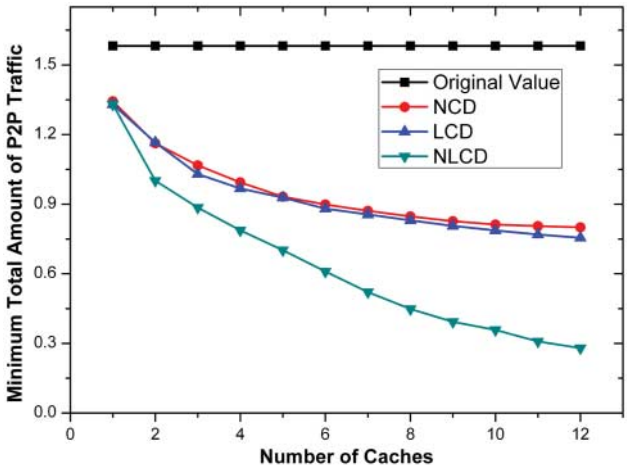

FIGURE 17. Min total amount of P2P traffic for Qwest.

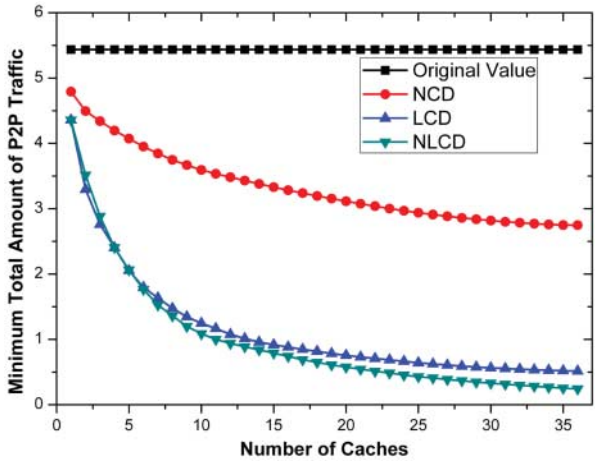

FIGURE 18. Min total amount of P2P traffic for CAIS.

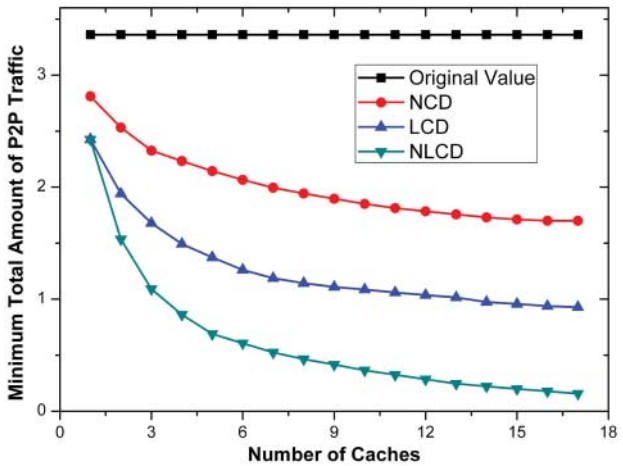

FIGURE 19. Min total amount of P2P traffic for NI.

4.3.2. Algorithm time complexity comparison

Compared with NCD, NLCD has higher performance but also higher time complexity. The time complexity of NCD is $O(Nk^2)$, which has been analyzed in detail in [12]. In NCD, the problem of optimizing N variables $C_1, C_2, \ldots, C_N$ is studied where C_i is the allocated cache capacity of node i. The problem can be transformed to a recursive formula. By solving this recursive formula with N iterations, the optimum cache capacities can be obtained. In each iteration, the maximum number of patterns the cache capacity can take is k. Therefore, the required time to derive the optimum capacities in NCD is $O(Nk^2)$. The time complexity of NLCD is $O(N^3k + 2N^2k|E|)$ which has been analyzed in Section 3.4. While the strictly optimal solution for the node-based cache deployment problem can be obtained using NCD, only an approximate optimal solution for the node-link-based cache deployment problem can be obtained using NLCD as it is a heuristic greedy algorithm.

Compared with LCD, NLCD also has higher performance and higher time complexity. The time complexity of LCD is $O(N^2k|E|)$. As described in Section 3.4, the time complexity of the deployment algorithm in Table 1 is $O(N^2k|U|)$ where $|U|$ is the number of candidate locations. $|U|$ equals $|E|$ for LCD, thus the time complexity of LCD is $O(N^2k|E|)$. We can easily see that the time complexity of LCD and NLCD are of the same order of magnitude.

5. CONCLUSION

Caching is an effective means for easing the burden imposed by P2P traffic on ISPs. P2P caches can be transparently deployed without needing change of client P2P software. The cache deployment problem is significant as it can greatly affect the effectiveness of caching. This paper addresses this problem. In this paper, we first present an analysis model for ISP cost calculation for NLCD. We find that an approximate network steady state can be reached and the ISP cost determined quickly. Then, we proved that the cache deployment problem for NLCD is NP-complete and proposed a heuristic greedy deployment algorithm for this problem. Finally, we compare NLCD with both NCD and LCD, and conclude that NLCD outperforms both NCD and LCD by a large margin. For four networks CW, Qwest, CAIS and NI, the average link utilization of NLCD is 8, 5, 5 and 15% lower than that of LCD, and 13, 7, 30 and 30% lower than that of NCD, respectively.

FUNDING

This work was supported by the National Basic Research Program of China (2012CB315802); and the National Natural Science Foundation of China (61070188, 61100176, 61100177) and the China Postdoctoral Science Foundation (2011M500401); and Shanghai Science and Technology Committee (11dz1500700).

REFERENCES

[1] Mennecke, T. (2005) *P2P is changing.* http://www.slyck.com/story914_CacheLogic_Study_P2P_is_Changing (accessed September 16, 2005, July 25, 2012).

[2] Schulze, H. and Mochalski, K. (2009) *Internet study 2008/2009.* http://www.ipoque.com/sites/default/files/mediafiles/documents/internet-study-2008-2009.pdf (accessed April 29, 2009, July 25, 2012).

[3] Wei, L. (2010) *ChinaTelecom Report—development trends and challenges of telecommunications industry and telecommunications technology*. http://wenku.baidu.com/view/be139cc78bd63186bcebbcdc.html (accessed October 15, 2010, July 25, 2012).

[4] Blue, C. *PacketShaper: it's your network. Own it*. http://www.packeteer.com/products/packetshaper (accessed December 6, 2011, July 25, 2012).

[5] Bindal, R., Cao, P. and Chan, W. (2006) Improving Traffic Locality in BitTorrent via Biased Neighbor Selection. Proc. IEEE Int. Conf. Distributed Comput. Syst. (ICDCS2006), Lisboa, Portugal, July 4–7, pp. 1–9. IEEE Press, Piscataway, NJ.

[6] Xie, H., Yang, Y., Krishnamurthy, A. and Silberschatz, A. (2008) P4P: Provider Portal for Applications. *Proc. SIGCOMM 2008*, Seattle, WA, USA, August 17–22, pp. 353–362. ACM Press, New York, NY.

[7] Lin, M., Lui, J. and Chiu, D. (2010) An ISP-friendly file distribution protocol: analysis, design, and implementation. IEEE Trans. Parallel Distrib. Syst., **21**, 1317–1329.

[8] Wierzbicki, A., Leibowitz, N. and Ripeanu, M. (2004) Cache Replacement Policies Revisited: the Case of P2P Traffic. *Proc. CCGRID 2004*, Chicago, IL, USA, April 19–22, pp. 182–189. IEEE Press, Piscataway, NJ.

[9] Hefeeda, M. and Noorizadeh, B. (2010) On the benefits of cooperative proxy caching for peer-to-peer traffic. IEEE Trans. Parallel Distrib. Syst., **21**, 998–1010.

[10] Hefeeda, M. and Saleh, O. (2008) Traffic modeling and proportional partial caching for peer-to-peer systems. IEEE Trans. Netw., **16**, 1447–1460.

[11] Ye, M., Wu, J. and Xu, K. (2008) Caching the P2P Traffic in ISP Network. *Proc. ICC 2008*, Beijing, China, May 19–23, pp. 5876–5880. IEEE Press, Piscataway, NJ.

[12] Kamiyama, N., Kawahara, R. and Mori T. (2010) Optimally Designing Capacity and Location of Caches to Reduce P2P Traffic. *Proc. ICC 2010*, Cape Town, South Africa, May 23–27, pp. 1–6. IEEE Press, Piscataway, NJ.

[13] Kamiyama, N., Kawahara, R. and Mori T. (2011) Optimally designing caches to reduce P2P traffic. Comput. Commun., **34**, 883–897.

[14] Karagiannis, T., Rodriguez, P. and Papagiannaki, K. (2005) Should Internet Service Providers Fear Peer-assisted Content Distribution? *Proc. IMC 2005*, Berkeley, CA, USA, October 19–21, pp. 63–76. ACM Press, New York, NY.

[15] Gummadi, K., Dunn, R. and Saroiu S. (2003) Measurement, Modeling, and Analysis of a Peer-to-Peer File-Sharing Workload. *Proc. SOSP 2003*, the Sagamore, Lake George, New York, USA, October 19–22, pp. 314–329. ACM Press, New York, NY.

[16] Wei, Q., Qin, T. and Fujita, S. (2012) A two-level caching protocol for hierarchical peer-to-peer file sharing systems. J. Convergence, **2**, 11–16.

[17] Li, B., Golin, M., Ialiano, G. and Deng, X. (1999) On the Optimal Placement of Web Proxies in the Internet. *Proc. INFOCOM 1999*, New York, NY, USA, March 21–25, pp. 55–59. IEEE Press, Piscataway, NJ.

[18] Qiu, L., Padmanabhan, V. and Voelker, G. (2001) On the Placement of Web Server Replicas. *Proc. INFOCOM 2001*, Anchorage, USA, April, pp. 1587–1596. IEEE Press, Piscataway, NJ.

[19] Zhou, X., Ge, Y., Chen, X., Jing, Y. and Sun, W. (2012) A distributed cache based reliable service execution and recovery approach in MANETs. J. Convergence, **3**, 5–12.

[20] Aikebaier, A., Enokido, T., and Takizawa, M. (2011) Trustworthy group making algorithm in distributed systems. 10.1186/2192-1962-1-6. http://link.springer.com/article/10.1186%2F2192-1962-1-6#.

[21] Laoutaris, N., Zissimopoulos, V. and Stavrakakis, I. (2004) Storage Capacity Allocation Algorithms for Hierarchical Content Distribution. *Proc. IFIP Net-Con 2004*, Palma de Mallorca, Spain, November 2–5.

[22] Laoutaris, N., Zissimopoulos, V. and Stavrakakis, I. (2005) On the optimization of storage capacity allocation for content distribution. Comput. Netw., **47**, 409–428.

[23] Luo, H. and Shyu, M. (2011) Quality of service provision in mobile multimedia—a survey. 10.1186/2192-1962-1-5. http://link.springer.com/article/10.1186%2F2192-1962-1-5#.

[24] Hefeeda, M. (2012) *pCache overview*. http://nsl.cs.surrey.sfu.ca/wiki/index.php/pCacheOverview (accessed December 1, 2008, July 25, 2012).

[25] Oversi, L. (2012) *OverCache Delivery Platform*. http://oversi.com/en/products/overcache-msp/overcache-p2p.html (accessed October 4, 2010, July 25, 2012).

[26] PeerApp, L. *What Is Transparent Caching?*, http://www.peerapp.com/Products/transparentcaching.aspx (accessed August 22, 2008, July 25, 2012).

[27] Carlinet, Y., Debar, H., Gourhant, Y. and Mé, L. (2010) Caching P2P Traffic: What Are the Benefits for an ISP? *Proc. 9th Int. Conf. Netw.*, Menuires, France, April 11–16, pp. 376–383. IEEE Press, Piscataway, NJ.

[28] Claffy, K. and Monk, T. *Mapnet: macroscopic internet visualization and measurement*. http://www.caida.org/tools/visualization/mapnet/ (accessed July 15, 2009, July 25, 2012).

[29] Floyd, R. (1962) Algorithm 97: shortest path. Commun. ACM, **5**, 345.

[30] Cho, K., Fukuda, K., Esaki, H. and Kato A. (2006) The Impact and Implications of the Growth in Residential User-to-User Traffic. *Proc. SIGCOMM* 2006, Pisa, Italy, September 11–15, pp. 207–218. ACM Press, New York, NY.