Photo Agent: An Agent-Based P2P Sharing System

Jane Yung-jen Hsu, Jih-Yin Chen, Ting-Shuang Huang,
Chih-He Chiang, and Chun-Wei Hsieh

Department of Computer Science & Information Engineering,
Institute of Networking and Multimedia,
National Taiwan University, Taipei, Taiwan
yjh@csie.ntu.edu.tw
{b89014, b89009, b89015, b89027}@csie.ntu.edu.tw

Abstract. With the proliferation of content creation devices, sharing digital contents has become an increasingly common task in our daily lives. This research proposes the “Photo Agent” that helps users manage and share digital photos without explicit file manipulation and data communication. The agents share photos autonomously and pro-actively, so users can simply specify *which* pictures to share with *whom*, rather than *how* the pictures are actually distributed and searched. The prototype photo agent utilizes peer-to-peer networking to support efficient content sharing in a distributed environment.

1 Introduction

There has been a growing demand for easy ways to share the large amount of digitally captured contents with family and friends. InfoTrends Research Group forecasted worldwide digital camera sales to reach 53 million units sold in 2004, and to continue growing at a 15% annual rate to reach 82 million units sold in 2008[1]. While commercial or open source software packages, i.e. ACDS[2], Paint Shop Photo Album[3], and Gallery[4] etc., help people organize digital photos on their personal computers or the web, such tools often require too much direct manipulation, thereby rendering photo sharing a chore. Some important challenges for a better photo sharing system include:

- efficient storage of a large number of images captured over time;
- efficient retrieval of specific images from the collection;
- effective content sharing mechanism given differences in user device, network bandwidth and availability;
- guaranteed delivery of contents; and
- intelligent search of contents from shared sources.

The proposed *Photo Agent* employs a human-centric approach to managing and sharing digital photos over a peer-to-peer (P2P) network. Photo Agents share photos autonomously and pro-actively, so users simply specify *which* pictures to share with *whom*, rather than *how* the pictures are actually distributed

and searched. The intuitive agent-based interface enables users to be freed from file management and networking details.

In what follows, we start by presenting the overall system design architecture. We then define access rights and agent communication protocol. Sections 4/5 describe details of the sharing/searching algorithms respectively. The prototype system implementation is outlined in Section 6, followed by the conclusion.

2 System Architecture

The photo agents operate on a distributed P2P network without a central server. Each node, called a *peer*, is responsible to help relay contents from the other nodes. Contents are collected and re-distributed to other peers through end-system multicast[5]. The cooperation of peers helps reduce the load on the node that sends content to a large group of friends. An *agent-based peer*[6] is an autonomous node that perceives the current network environment and makes the best decision to facilitate photo transmission and redirection.

As is shown in Figure 1, each node consists of four major components: management agent, sharing agent, searching agent, and the photo database.

Fig. 1. Architecture of the photo agent

The *management agent* manages indexing, storage, and retrieval of digital images. It also delegates sharing and searching tasks from the user to the sharing and searching agents. The user interface provides functions for importing, sorting, and annotating photos. Duplicates photos are removed to optimize disk utilization. The *sharing agent* is in charge of distributing any set of photos to the target group as specified by the user. The cooperation of sharing agents on the network forms a sharing tree to ensure an efficient and reliable sharing process[7]. The *searching agent* helps a user to locate specific contents residing in the photo database of another peer. Depending on the searching algorithm and the access rights of the target content, multiple searching agents cooperate on the P2P network to return a group of useful results. Agents engage in active communication in carrying out the sharing and searching tasks.

3 Control and Communication

This section defines the access control and agent communication protocol in the proposed photo agent system.

In a P2P sharing system, it is necessary for users to protect their contents with access control[8]. Table 1 shows the definition of access rights in terms of four attributes[9]. The owner has complete access and can modify any attribute of his own photos. The value of *Share lifetime* decreases by one every time the photo is shared to another user. When the *Share lifetime* becomes 0, the *Search level* is set to *nobody* automatically. To preserve privacy, the *Search level* can only be adjusted lower except by the photo owner. Access right specifications should remain with a given photo, even when it is exported or shared.

Table 1. Attributes of access rights for a digital photo

Attribute	Attribute definition	Possible value
<i>Photo owner</i>	The user who imports the photo	The owner ID
<i>Export right</i>	The right to export this photo	Yes/No
<i>Share lifetime</i>	The number of nodes this photo can be shared	0/positive integer/ ∞
<i>Search level</i>	The level that a photo can be searched	all/friend/nobody

In sharing and searching photos, a photo agent needs to exchange status and preference information with the other agents. The basic message types include *share*, *request*, *info*, *report*, *search*, and *redirect*. The *info* message is used to communicate the capabilities and preferences of the given node. Figure 2 illustrates the sample communication transcripts of relaying reduced photos to be shared and search requests.

Fig. 2. Agent communication

4 Photo Sharing

P2P file sharing systems, ranging from fully distributed models like Gnutella[10] to centralized models such as Napster, have gained tremendous popularity. Sharing personal experiences, in the form of images or videos, is a promising application area of P2P sharing technology. A good photo sharing system[11] needs *to distribute the contents efficiently and to ensure successful delivery of shared contents*. Instead of a central server that relays files to their destinations, content delivery is distributed in a P2P network. Each node is responsible for relaying photos to its peers. Figure 3(a) illustrates a sharing structure for distributing content from the *source node* A, to the *relay nodes* B and C, and then to all remaining *target nodes* D through H. The sharing agent at each node can autonomously decide the appropriate relay nodes from its sharing list[12]. To

Fig. 3. P2P Photo Sharing

reduce bandwidth requirement, reduced images of the shared photos are propagated through the sharing tree structure. The reduced photos act as the “tickets” for requesting the original photos from the owners.

The sharing agent on a relay node is in charge of relaying the photos to its assigned subset of target nodes. Each relay node is expected to report the successful (or failed) deliveries back to its parent node. Figure 3(b) illustrates the bottom-up reporting result if node D remains off-line. The Photo Agent system needs to ensure that each file is sent to its destination, even if the target peer is currently off-line. When node B gets no report from node D after a period of time, it reports back to the parent node A that D and H has not received the photo. The sharing agents on both nodes A and B will attempt to *catch* D and H when they get back online at a later time. Without a central server, the relay nodes have to record the unfinished sharing jobs. If a delivery is not possible when the nodes are never online at the same time, alternative delivery channels, e.g. email, may be used.

4.1 Construction of Sharing Tree

In general, the number of relay nodes R should increase with the node's bandwidth B and the number of target nodes T , i.e. $R = f(B, T)$. The function f needs to

- Bound the file transfer time of the source node to t , and
- Compact the height of the sharing tree to h .

Let n be the number of photos to be transferred and a be the average size of a typical photo image, we have $\frac{n \times a}{B} \times R \leq t$. As a result, the upper bound of R is

$$R \leq \frac{B \times t}{n \times a}. \quad (1)$$

The height of the sharing tree should be under h , that is, $1 + \lceil \log \lceil \frac{T}{R} \rceil \rceil \leq h$. The lower bound of R is found to be

$$R \geq \frac{T}{2^{h-1}}. \quad (2)$$

To select the best relay nodes, each target node is assigned a score s based on its current end-to-end connection quality with the source node, denoted as q , and its past recorded successes p . That is,

$$s = \alpha \cdot q + \beta \cdot p \quad (3)$$

where α and β specify the relative weights of bandwidth vs. past record.

Suppose that information about bandwidth and connection quality are aggregated in global lookup tables over time. Algorithm 1 presents the process of constructing the sharing tree. Let *sharingList* denote the list of target nodes, and *relayList* denote the list of relay nodes.

Algorithm 1. Sharing_with_global_info

Given global lookup tables Q , P and B , construct the sharing tree.

Require: *sharingList*: A list of nodes for tree construction

---

```

1: relayList  $\leftarrow$  source node
2: for  $i \in \textit{relayList}$  do
3:   for  $j \in \textit{sharingList}$  do
4:      $S(j) \leftarrow \alpha \cdot Q(i, j) + \beta \cdot P(j)$ 
5:   end for
6:    $R \leftarrow f(B(i), |\textit{sharingList}|)$ 
7:    $\textit{Child}(i) \leftarrow$  highest scoring  $R$  nodes from sharingList
8:    $\textit{relayList} \leftarrow \textit{relayList} \setminus \{i\} \cup \textit{Child}(i)$ 
9:    $\textit{sharingList} \leftarrow \textit{sharingList} \setminus \textit{Child}(i)$ 
10: end for
```

Algorithm 1 constructs the sharing tree by recursively selecting the R top-scoring nodes from *sharingList* to serve as the relay nodes. The construction

process terminates when *sharingList* is exhausted, and the entire sharing tree is created in one shot.

When the global information about bandwidth, connection quality, and past record is not available, or when the network properties change frequently, it is not a good idea to construct a static sharing tree as described above. Instead of aggregating information in global tables, each node owns a local table that contains the information of a limited number of nodes. Given such incomplete information, the sharing tree is constructed as a result of cooperation by all nodes involved. Each node probes the others to retrieve and update the necessary information. In contrast to Algorithm 1, which builds the complete sharing tree at the root node, each node contributes in Algorithm 2 by constructing a part of the sharing tree with *k-level lookahead*. Parameter *sharingTree* is the tree structure under construction, while parameter *sharingList* is a list of target nodes to be assigned to the *sharingTree*.

First, the current node decides the appropriate number of relay nodes and the levels of sharing tree it plans to build. Algorithm 2 then proceeds with constructing the sharing tree and all related nodes. Meanwhile, information in the local lookup table is updated through cooperation. Finally, the updated sharing tree and sharing list are sent to each relay node to continue the construction of the sharing tree.

Algorithm 2. Sharing_without_global_info

Construct the sharing tree with *k-level* lookup.

Require: *sharingTree, sharingList*

- 1: Decide r , the number of relay nodes from the current node.
  - 2: Decide k , the levels of sharing to be constructed from the current node.
  - 3: **for** $i = 1$ to k **do**
  - 4:   **for** $y \in$ next i -level relay nodes **do**
  - 5:   Select the best r nodes as the relay nodes.
  - 6:   Construct the next level of sharing trees from node y .
  - 7:   **end for**
  - 8: **end for**
  - 9: Update *sharingTree* and *sharingList*.
  - 10: Send the updated lists to the relay nodes at the next level.
-

The sharing agents distribute reduced images of the pictures to be shared using the sharing tree. The target user is then notified about the newly shared pictures. Requests for the original photos are only made when the user decides to accept the selected content.

5 Photo Searching

An alternative mode of sharing is to allow users to *search* for specific photos from the collections maintained by their peers. Given a search request, the *searching*

agent broadcasts the request to its peers[12]. Each request message should include the following fields.

RootID: ID of the agent making the original request.

Deadline: The maximum time for request propagation.

Query: Specification of the target images.

Relay: Whether the request is a relay search.

To eliminate redundant search requests, path information is recorded so that a request will not be sent to the same node twice. When a match is found, the corresponding reduced image is sent back to the requesting agent along with the original query and agent ID. The user can then select from the matching photos to request originals from their owners. Figure 3 illustrates the searching algorithm. *Line 2* queries the local collection using `local_search`. *Lines 3-7*

Algorithm 3. Searching

Searching Algorithm.

Require: *RootID*, *Deadline*, and *Query*

```

1: if Deadline > 0 then
2:   result  $\leftarrow$  local_search(Query)
3:   for  $z \in \textit{result}$  do
4:     if access_right( $z$ ) then
5:       send_photo(RootID,  $z$ )
6:     end if
7:   end for
8:   Forward the request with a reduced deadline to everyone on the friend's list.
9: end if
```

---

check the access rights for legal distribution. *Line 8* forwards the request to all peers for more matching results.

6 Implementation and Concluding Remarks

The Photo Agent is designed to provide a simple interface that helps users manage photos conveniently, share photos efficiently, and protect photos safely. A prototype system has been developed to support content sharing in a server-less decentralized peer-to-peer network. Figure 4 shows the browser interface implemented in PHP with MySQL for data management. P2P sharing/searching is built on top of JXTA[13]. Experiments of the prototype photo agents were carried out using desktop and laptop PCs running Windows XP. Initial results show that agent-based P2P sharing is a light-weight and efficient approach to content management and distribution on the Internet.

As the population of digital camera owners grows rapidly, photo agents will become an indispensable tool for ad hoc photo sharing. Further experiments and analysis are necessary to improve the sharing and searching algorithms to

(a) Sharing selected photos

(b) Notification

Fig. 4. Photo Agent Interface

ensure fast and reliable content delivery. The proposed agent-based P2P sharing mechanism can be readily extended to other kinds of digital content in the near future.

This research is sponsored in part by the ROC National Science Council grant NSC-93-2213-E-002-087.

References

1. InfoTrends Research Group: Worldwide consumer digital camera forecast (2003)
2. ACD Systems: ACDSee 6.0 (2004) <http://www.acdsystems.com/>.
3. Jasc Software: Paint shop photo album 4 (2004) <http://www.jasc.com/products/photoalbum/>.
4. SourceForge.net: Gallery 1.4.2 (2004)
5. Chu, Y.H., Rao, S.G., Zhang, H.: A case for end system multicast. In: ACM Sigmetrics. (2000) 1–12
6. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Second edn. Prentice Hall (2002)
7. Nagi, K., Elghandour, I., Konig-Ries, B.: Mobile agents for locating documents in Ad Hoc networks. In: Proceedings of the Workshop on Agents and Peer-to-Peer Computing at AAMAS 2003. (2003)
8. Liu, Q., Safavi-Naini, R., Sheppard, N.P.: Digital rights management for content distribution. In: Proceedings of the Australasian information Security Workshop Conference on ACSW Frontiers 2003. (2003) 49–58
9. Feigenbaum, J., Freedman, M., Sander, T., Shostack, A.: Privacy engineering for digital rights management systems. In: Proceedings of the 2001 ACM Workshop on Security and Privacy in Digital Rights Management. (2001) 76–105
10. Clip2: The Gnutella Protocol Specification v0.4. (2001)
11. Aberer, K., Cudre-Mauroux, P., Datta, A., Hauswirth, M.: PIX-Grid: A platform for P2P photo exchange. In: Proceedings of Ubiquitous Mobile Information and Collaboration Systems (UMICS 2003), collocated with CAiSE'03. (2003)
12. Turcan, E., Shahmehri, N., Graham, R.L.: Intelligent software delivery using P2P. In: IEEE Proceedings of Second International Conference on Peer-to-Peer Computing. (2002) 136–143
13. Sun Microsystems: Project JXTA v2.0: *Java^{TM}* Programmer's Guide. (2003)