Bee-Inspired Data Collection Methods for P2P Streaming Systems

Tomoki Yoshihisa
Osaka University
5-1 Mihogaoka, Ibaraki
Osaka, Japan 567-0047
yoshihisa@osaka-u.ac.jp

Tadashi Nakano
University of California, Irvine
3241 Donald Bren Hall
Irvine, CA 92697-3425
tnakano@ics.uci.edu

Shun N. Watanabe
University of California, Irvine
3241 Donald Bren Hall
Irvine, CA 92697-3425
shunw@ics.uci.edu

Tatsuya Suda
University of California, Irvine
3241 Donald Bren Hall
Irvine, CA 92697-3425
suda@ics.uci.edu

ABSTRACT

Recently, P2P (Peer-to-Peer) streaming systems have attracted great attention. In P2P streaming systems, streaming data such as video and audio are divided into a number of small pieces for efficient data distribution. This approach allows peers to collect the pieces from each other while playing the streaming data. Our investigation into ecological systems suggests that the piece collection in P2P streaming systems is similar to the bee's nectar collection and that their nectar collecting behavior can be used as a model in designing piece collection methods for P2P streaming systems. In this paper, we propose data collection methods for P2P streaming systems inspired by bee's nectar collecting behavior. There are several types of bees in the bee ecology, which exhibit different behavior, and accordingly, we propose three different methods inspired by three typical bee types; those are honey, bumble, and carpenter bees. Suitable environments for the three methods may differ and are investigated in this paper.

Keywords

Biological inspiration, Peer-to-peer streaming, Bittorrent

1. INTRODUCTION

Recently, P2P (Peer-to-Peer) streaming systems have attracted great attention. In general P2P streaming systems, streaming data (e.g., movie data) are divided into several parts and peers collect the divided parts from other peers. Divided parts are called pieces in P2P streaming systems. An advantage of this method is that a large data size streaming data is divided into small size pieces, and it is therefore easily transmitted and distributed over a number of peers.

However, pieces have to be carefully transmitted and distributed in such a way that peers can continuously play the movie data without experiencing an interruption during the play. If a peer does not have a piece collected by the time to play it, the peer has to stop playing the movie data; i.e., an interruption occurs. This means, for example, that a peer needs to collect the piece to be played at 10 minutes, within 10 minutes after the peer starts playing the movie data. Otherwise, the peer experiences an interruption.

In order to reduce the number of interruptions for P2P streaming systems, several methods and algorithms have been proposed[2, 4, 7]. The existing approaches are variants of the rarest first algorithm[1], in which pieces are distributed from the rarest one. The existing approaches may not work well in P2P streaming systems for the following two reasons. (1) In P2P streaming systems, peers tend to play streaming data from the beginning to the end sequentially and therefore former pieces are more valuable. However, the rarest first algorithm does not consider the "value" of pieces and instead distributes from the rarest piece. This is because the rarest first algorithm is originally designed for file sharing systems, in which time insensitive data is exchanged.

(2) P2P streaming systems are often deployed in a highly dynamic environment where peers join and leave the system frequently. In such an environment, it is important to distribute pieces as rapidly as possible. However, the rarest first algorithm does not consider the time needed to distribute pieces and only distributes pieces according to rarity(Note that the time needed to download rare pieces tends to be larger than that of popular pieces).

In this paper, we propose an alternative approach, considering the value of pieces and time needed to distribute pieces. In our approach, peers collect pieces from former ones. Although the speed of distributing rare pieces over peers is slower than the existing approaches, our proposed approach can reduce the interruption time effectively since peers collect the necessary data rapidly.

Our investigation into the bee ecology suggests that our proposed approach is very similar to the bee's foraging behavior (i.e., nectar collecting behavior). Since bees have evolved to efficiently collect nectar, their foraging behavior is potentiality applicable to piece collection in P2P streaming systems. In this paper, we propose data collection methods for P2P streaming systems inspired by bee's foraging behavior (i.e., nectar collecting behavior). In the ecological world, there are several different types of bees which exhibit different foraging behavior. Accordingly, in this paper, we propose three data collection methods inspired by three different bee types (i.e., honey, bumble, and carpenter bees.) Suitable environments for these methods may differ and are investigated in this paper.

The paper organized as follows. Section 2 introduces related work. Section 3 explains bee-inspired P2P streaming systems. Our proposed methods are explained in Section 4 and evaluated in Section 5. Finally, we conclude the paper in Section 6.

2. RELATED WORK

In general P2P systems, a data item (e.g., a movie file) is divided into a number of small pieces for data distribution, and peers collect the pieces from each other. For the piece collection, one of the most famous P2P systems, BitTorrent[1], uses the rarest first algorithm. In the algorithm, peers are divided to form some groups called swarms and each peer collects pieces from the rarest one in its swarm. This is to ensure that all pieces are distributed widely and evenly over the peers and are always available to peers for piece collection. However, the algorithm is not directly applicable to streaming data systems, in which peers are collecting and playing the data pieces simultaneously. If a peer does not have a piece by the time to play it, an interruption occurs. From this view point, it is better to collect former pieces rather than the rarest piece in P2P streaming systems.

To reduce the number of interruptions, Shah et al. adopt the sliding window technique to the rarest first algorithm[4]. In the algorithm, peers collect pieces from the rarest one within a window. A window contains n pieces starting from the most former piece that a peer does not have, where n is called the window size. Since the algorithm allows peers to collect from former (and somehow rare) pieces, the number of interruptions during play is reduced compared with the original rarest first algorithm.

In this paper, we propose an alternative solution to P2P streaming systems inspired by bee foraging behavior. In bee foraging, bees choose flowers to maximize the energy gain from the nectar, relative to the cost of obtaining the nectar (e.g., distance to the flowers). In choosing flowers, bees consider the quality of nectar from various factors (e.g., stimuli of color and smell). This is analogous to P2P streaming systems, where peers need to collect pieces in a resource efficient manner (e.g., in terms of bandwidths), and in addition, peers need to consider the quality of pieces, (i.e., former pieces are more valuable) for streaming data.

Biological inspiration has been successfully applied in computer networks (see [5] for a survey of biologically inspired networking). For example, bee foraging behaviors have so far been used to design ad hoc networks (called BeeAdHoc [6]) and sensor networks (called BeeSensor[3]). In this paper, we apply inspiration from bee foraging behavior to develop data collection algorithms for P2P streaming systems. In the next section, we explain bee-inspired P2P streaming systems.

3. BEE-INSPIRED P2P STREAMING SYSTEMS

3.1 Bee's Nectar Collecting Behavior

There are many flowers in a flower garden and a flower has nectar. Bees fly to a nearby flower garden and collect nectar. Nectar is carried little by little by bees. Flowers have their best season. Nectar quality of flowers in their best season is high and bees prefer best season flowers. There are different types of bees that show different foraging behaviors. Some types of bees collect nectar as rapidly as possible regardless of the quality of the nectar, and some other types of bees collect high-quality nectar. Some types of bees form bee groups, and some other types do not. Here, we introduce three typical types of bees that are honey, bumble, and carpenter bees[8].

3.1.1 Honey Bees

Honey bees form bee groups. They prosper in hazardous environments where hostile bee groups coexist or there are only a few flower gardens. Therefore, they collect nectar as rapidly as possible.

3.1.2 Bumble Bees

Bumble bees form bee groups. They prosper in calm environments where few hostile bee groups coexist or there are many flower gardens. Therefore, they consider the quality of nectar prior to the time and speed of collecting nectar. They collect nectar from best season flowers.

3.1.3 Carpenter Bees

Carpenter bees do not form bee groups. They collect nectar alone. When they find a flower, they collect the nectar. Therefore, they collect nectar in their nearest flower.

3.2 P2P Streaming Systems and Bee Ecology

Collecting data pieces for P2P streaming systems is similar to bee foraging behavior. First, streaming data is similar to nectar. In P2P streaming systems, data is divided into pieces. Peers need all pieces to play the whole data. Also, former pieces are prior for peers because they play the data from the beginning to the end sequentially. Nectar in a flower is similarly divided into a little amount so that a bee can carry it. Also, bees prefer nectar in best season flowers. This corresponds to former pieces. Next, peers are similar to flower gardens. Peers have pieces. Flower gardens similarly have flowers including nectar. Lastly, communication between peers is similar to behaviors of bee groups. Peers communicate with each other to receive pieces. The speed to receive pieces is restricted by their bandwidths. Bee groups similarly communicate to collect nectar. The speed to collect nectar is restricted by the number of bees in a bee group.

As described above, we believe that piece collection for P2P streaming systems is similar to bee ecology. Table 1 summarizes the correspondence between P2P streaming systems and bee ecology.

Since bees collect nectar rapidly so that they can collect more nectar, their collecting behavior is also effective for P2P streaming systems.

4. PROPOSED METHODS

Table 1: Correspondence between P2P Streaming Systems and Bee Ecology

P2P Streaming Systems	Bee Ecology
Peer	Flower Garden
Piece	Flower (including nectar)
Former Piece	Best Season Flower
Communication	Bee Behaviors

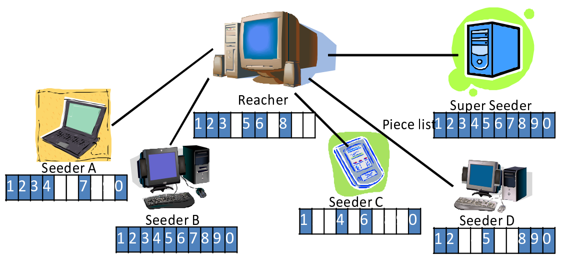

Figure 1: System Environment

In this paper, we propose data collection methods for P2P streaming systems inspired by bee's foraging behavior.

4.1 Assumed System Environment

Our assumed system environment is based on an actual P2P system, BitTorrent[1]. Its data collection method, rarest first, is not suitable for P2P streaming systems. In BitTorrent systems, peers requesting pieces are called reachers, peers providing pieces are called seeders, and peers who have all pieces are called super seeders. Figure 1 illustrates an example system environment, where the streaming data is divided into 10 pieces and the reacher is collecting the pieces from the super seeder and seeders A-D. We briefly explain the system below.

- Peers know all other peers' IP addresses.
- Peers collect pieces after requesting playing the data.
- There is only one type of streaming data in the system.
- Peers leave the system when they finish playing the data.
- Data is a streaming data. Shorter interruption time is preferable.
- There is one super seeder that has all pieces.

We explain the appropriateness of the above environment. In a general P2P streaming system, a special node called tracker exists to manage the status of participating peers in the system, including the information about which peers are connected to and disconnected from the system. It is, therefore, possible to get a list of existing peers and their IP addresses from the tracker. Also, a super seeder exists, making all pieces available in the system.

It is an on-demand streaming system, and peers do not receive pieces beforehand. If peers collect pieces before the request, peers need unlimited storage space to store a large number of streaming data that may exist in the system.

```
When the peer P finishes receiving a piece:
output: target peer Q
for i ∈ seeders do //for all seeders
    b=GetBandwidth(P,i)//get bandwidth between P and i.
    if maxb < b then
        maxb=b
        Q=i
    end if
end for
```

Figure 2: Algorithm for Honey Bee Method

Also, there are many peers, and peers do not distribute multiple streaming data simultaneously. That is, a peer contributes distribution of one streaming data. To receive multiple streaming data, peers receive its pieces from other group of peers. This does not mean that our assumed system cannot distribute multiple streaming data. Peers cannot communicate with many peers simultaneously. However, in practice, a peer achieves multiple streams by alternating among the groups of peers.

4.2 Assumed Scenario

The following illustrates how a peer joins and leaves the system described in Section 4.1. Assume that a peer P requests playing a 60-minute movie file. P gets the tracker's address from a web page and it connects to the P2P streaming system. The tracker adds P's address to the existing peer list. P gets the existing peer list from the tracker and it sends a request of receiving a piece to existing peer. P uses one of our proposed methods or an existing method to determine from which peer to request a piece. Q denotes the peer that P requests a piece. P gets the piece list for Q. The piece list describes the pieces that Q has. After receiving the piece, P requests the next piece to other peers. When P has finished the reception of the first piece of the streaming data, it starts playing the data. Peers repeat such process until the final piece is collected. While P plays the data, other peers can connect to P and receive the pieces that P has. When P finishes playing the data, it leaves the P2P system. Pieces that P has then become inaccessible.

4.3 Methods

In the following, we propose three bee-inspired data collection methods for P2P streaming systems. In each method, a peer first selects the peer to receive a piece from, and then receives the most former piece from the selected peer.

4.3.1 Honey Bee Method

Honey bees collect nectar as rapidly as possible. Therefore, in the honey bee method, a peer selects the peer that provides the fastest download speed. Download speeds are calculated by communicating peers for getting the piece list when the peer starts receiving the next piece. This algorithm is shown in Figure 2.

4.3.2 Bumble Bee Method

Bumble bees collect nectar from best season flowers. Therefore, in the bumble bee method, each peer selects the peer that has the most former piece that the requesting peer does not have. The peer can recognize the most former piece by getting piece lists from all other peers. The algorithm is shown in Figure 3.

```
When the peer P finishes receiving a piece:
output: target peer Q
for i ∈ seeders do //for all seeders
    p=GetFirstPiece(P,i)
            //get the first piece that i has but P does not have.
    if p < minp then
        minp=p
        Q=i
    end if
end for
```

Figure 3: Algorithm for Bumble Bee Method

```
When the peer P finishes receiving a piece:
output: target peer Q
tp=GetRequestTime(p)
                //get the time that p requests the first piece
for i ∈ seeders do //for all seeders
    ti=GetRequestTime(i)
    if maxt < ti and ti < tp then
        maxt=ti
        Q=i
    end if
end for
```

Figure 4: Algorithm for Carpenter Bee Method

4.3.3 *Carpenter Bee Method*

Carpenter bees collect nectar in their nearest flower. That is, in the carpenter bee method, each peer selects the peer that requests playing the data before the requesting peer. This algorithm is shown in Figure 4.

5. EVALUATION

In this section, we show some evaluations for our proposed methods. We evaluated our proposed methods by computer simulation. In the simulation, we used parameters shown in Table 2. The request arrival rate means the average interval for peers to request playing the streaming data. For example, peers request playing the streaming data every 30 sec. on average according to Poisson distribution when the request arrival rate is 30 sec. Minimum and maximum out/in bandwidths are parameters for the peer bandwidth. We set a peer's bandwidths value between the minimum and the maximum according to Uniform distribution. To make the simulation more practical, we set the minimum bandwidth. We assume that peers disconnect from the network immediately after they finish playing the data.

We compare our proposed method with previously proposed other methods for P2P streaming systems. One is the rarest first algorithm explained in Section 2. This is a basic algorithm for P2P systems. The other is the sliding window algorithm. This is also explained in Section 2 and is more suitable for P2P streaming systems than the rarest first algorithm.

5.1 Interruption Time

The main purpose of our proposed method is reducing the interruption time for peers while they are playing the streaming data. In this subsection, we show the summation of interruption time for each peer. The result is shown in Figure 5. The vertical axis is the sum of the interruption time and the horizontal axis is the request arrival time for each peer. For example, a plot for the time 0 means the sum of the interruption time for the peer that request playing the data at time 0 in the simulation time. The figure does not show the rarest first algorithm since the sum of the interruption time under the algorithm is very large and exceeds the plot's range. "sequential" means a method that peers select the peer that requests playing the data just before the requesting peer.

Table 2: Simulation Parameters

Parameters	Values
Requst Arrival Rate	Poisson Distribution 60 sec.
Bit Rate	512 Kbps
Duration	30.0 minutes
Data Size of a Piece	256 Kbytes
Simulation Time	6 hours
Minimum Out Bandwidth	400 Kbps
Maximum Out Bandwidth	800 Kbps
Min In Bandwidth	1 Mbps
Maximum In Bandwidth	1.5 Mbps

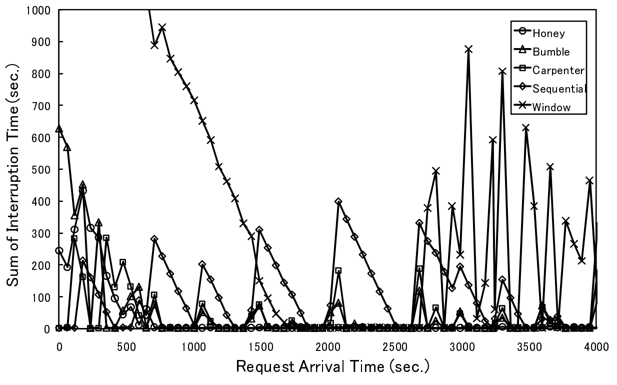

Figure 5: Interruption Time

From this figure, we can see that the honey bee method gives shorter interruption time than other methods almost all over the time. Even when the value under other methods suddenly increases at times 709, 1065, 1491, and so on, the value under the honey method does not increase. The reason of the increase is disconnections of peers. When a peer A disconnects from the network, the reception of the piece from A interrupts. Therefore, peers that receive a piece from A have to start the reception again and the interruption time increases. However, in the honey bee method, since peers select a peer that has the fastest download speeds, peers can finish the reception before the disconnection. In the early stage of the simulation, the bumble and the carpenter bee methods gives better performance than the honey bee method. This is because the honey bee method is not effective when the number of peers is small.

For instance, the sum of the interruption time under honey method for the peer that requests the data at time 3000 is 3.4 sec. This is the minimum in all methods. In this case when the peer starts playing the data immediately after requesting the data, the peer has to wait 3.4 sec. while playing the data. If the peer waits 3.4 sec before starting playing the data, it has no interruption time although it has to wait after requesting the data.

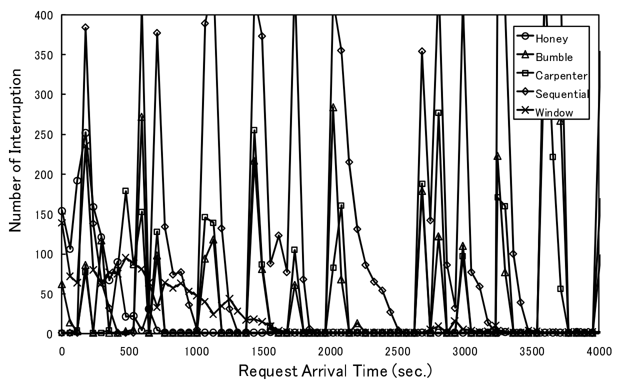

Figure 6: Number of Interruption

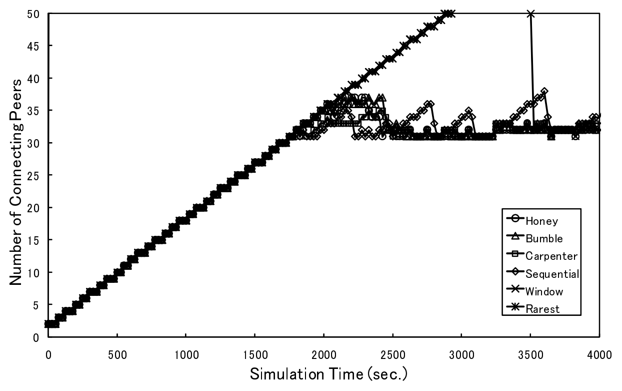

Figure 7: Number of Connecting Peers

Peers prefer smaller number of interruptions. We show the number of interruption in Figure 6.

The vertical axis is the number of the interruption and the horizontal axis is the request arrival time, same as the previous figure. Also regarding the number of the interruption, the honey bee method gives better performance. Since the sequential method does not consider any factors such as download speeds and the piece number, the interruption occurs frequently.

5.2 The Number of Connecting Peers

To show the situation for P2P streaming systems, we show the number of connecting peers. We can confirm that how many peers can contribute the distribution of pieces from the result. The result is shown in Figure 7. The vertical axis is the number of connecting peers and the horizontal axis is the simulation time.

Although the value under the rarest fast method continues to increase, that under other methods converges. That is, some peers disconnect from the networks when they finish playing the data. The number of connecting peers under other methods looks almost similar.

5.3 The Number of Played Peers

The number of peers that finish playing the streaming data is also the factor to evaluate our proposed methods. Therefore, we simulate the number of played peers. The

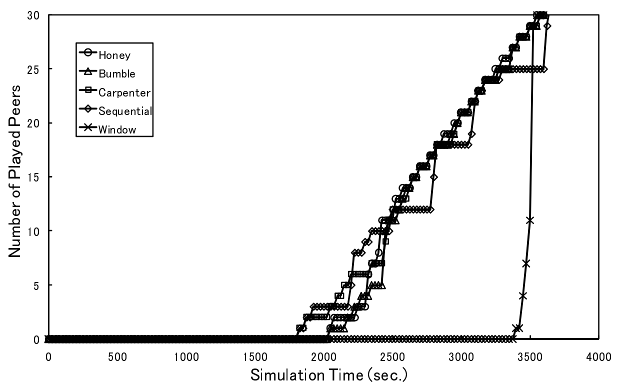

Figure 8: Number of Played Peers

result is shown in Figure 8. The vertical axis is the number of played peers and the horizontal axis is the simulation time.

All methods except for the window method suddenly increase at the time approximately 1800. This is because the duration of the data is 1800 sec. Although the sequential and the carpenter bee methods give larger number of played peers in the early stage, the honey bee method gets better after that. The reason is the same as the case of interruption time.

6. CONCLUSION

In this paper, we proposed data collection methods for P2P streaming systems inspired by bee's foraging behavior. We proposed three methods inspired by typical bees; namely, honey, bumble, and carpenter bees. Our simulation results showed that the bumble and carpenter bee methods give better performance in the early stage of the data piece distribution. However, after the number of connecting peers increases enough, the honey bee method gives better performance than them. Future work includes a more extensive simulation evaluation using different criteria such as load balancing as well as mathematical analysis into the characteristics of the proposed data collection methods.

7. ACKNOWLEDGMENTS

This research was supported in part by the Telecommunications Advancement Foundation, Japan , in part by Grant-in-Aid for Scientific Research (A) numbered 20240007 from the Japanese Ministry of Education, Culture, Sport, Science and Technology of Japan, and in part by Research and Development Program of 'Ubiquitous Service Platform' (2008), The Ministry of Internal Affairs and Communications, Japan.

8. REFERENCES

[1] BitTorrent. http://www.bittorrent.com/.

[2] Dana, C., Li, D., Harrison, D., and Chuah, C.-N. BASS: BitTorrent Assisted Streaming System for Video-on-Demand. In Proc. of IEEE Workshop on Multimedia Signal Processing, pp. 1-4, 2005.

[3] Saleem, M. and Farooq, M. BeeSensor: A Bee-Inspired Power Aware Routing Protocol for

Wireless Sensor Networks . In Springer Lecture Notes in Computer Science, Vol. 4448, pp. 81-90, 2007.

[4] Shah, P. and Paris, J.-F. Peer-to-Peer Multimedia Streaming Using BitTorrent. In Proc. of International Performance of Computers and Communication Conference (IPCCC'07), pp. 340-347, 2007.

[5] Suda, T., Nakano, T., and Fujii, K. Applications of Biological Concepts to Designs of Computer Networks and Network Applications. The Handbook of Computer Networks, John Wiley & Sons Inc, 2007.

[6] Wedde, H. F., Farooq, M., Pannenbaecker, T., Vogel, B., Mueller, C., Meth, J., and Jeruschkat, R. BeeAdHoc: An Energy Efficient Routing Algorithm for Mobile Ad Hoc Networks Inspired by Bee Behavior. In Proc. of Genetic and Evolutionary Computation, pp. 153-160, 2005.

[7] Vlavianos, A., Iliofotou, M., and Faloutsos, M. BiToS: Enhancing BitTorrent for Supporting Streaming Applications. In Proc. of IEEE International Conference on Computer Communications (INFOCOM'06), pp. 1-6, 2006.

[8] Wright, R., Mulder, P., and Reed, H. Honey Bees, Bumble Bees, Carpenter Bees, and Sweat Bees. In Oklahoma Cooperative Extension Fact Sheets, EPP-7317, 2007.