A COP for Cooperation in a P2P Streaming Protocol

María Elisa Bertinat, Darío Padula,
Franco Robledo Amoza and Pablo Romero
Laboratorio de Probabilidad y Estadística,
Facultad de Ingeniería, Universidad de la República
Julio Herrera y Reissig 565,
11300, Montevideo, Uruguay,
Email: {bertinat,dpadula,frobledo,promero}@fing.edu.uy

Daniel De Vera and Pablo Rodríguez-Bocca
Departamento de Investigación Operativa,
Facultad de Ingeniería,
Universidad de la República
Julio Herrera y Reissig 565,
11300, Montevideo, Uruguay,
Email: {ddevera, prbocca}@fing.edu.uy

Gerardo Rubino
INRIA,
Campus de Beaulieu,
35042 Rennes, France,
Email: rubino@irisa.fr

***Abstract*—This paper introduces a Combinatory Optimization Problem (COP) which captures the performance in cooperation of a P2P Streaming Network, considered at the buffer level. A new family of strategies of cooperation is defined, which represents the order of query of pieces in a communication between peers. We define a measure of quality for each strategy, based on the most shocking video streaming parameters, namely the continuity and the start-up latency. Our aim is o find quasi-optimal strategies inside the proposed family, by means of on a model based on an Asymmetric Traveling Salesman Problem (ATSP). Finally, we solve this ATSP using an ACO (Ant Colony Optimization)-based algorithm. The results show substantial improvement with this approach, once compared with the ones obtained by previous defined strategies of cooperation.**

Keywords: P2P; Performance; COP; ATSP; ACO.

I. Introduction

Internet-based multimedia systems have many different architectures, depending on their sizes and on the popularity of their contents. The majority of them have a traditional CDN (Content Delivery Network) structure [1], [2], where a set of datacenters absorbs all the load, that is, concentrates the task of distributing the content to the customers. This is, for instance, the case of msnTV, YouTube, Jumptv, etc., all working with video content.

Another popular alternative consists in using the often idle capacity of the clients to share the video distribution with the servers through the present mature Peer to Peer (P2P) systems [3], [4], [5]. These are virtual networks developed at the application level over the Internet infrastructure. The nodes in the network, called peers, offer their resources to the other nodes, basically because they all share common interests. As a consequence, as the number of customers increases, the same happens with the global resources of the network. For this reason, P2P networks are said to *scale* well.

Nowadays P2P networks play an important role because of their popularity and their impact in Internet traffic. Some commercial P2P networks for live video distribution are available, all of them with proprietary source-codes and protocols. The most successful are PPlive [6], [7], SopCast [8], PPstream [9], TVAnts [10] and TVUnetwork [11]. This paper is part of the GoalBit project [12], [13], the first open-source P2P streaming network.

On one hand, the dynamism and freedom of peers in a P2P network are attractive and powerful tools for the users. On the other hand, they impose many challenges on the architecture design and on the protocols for sharing information [14], [15]. Live video P2P networks have harder constraints to satisfy, because the video has to arrive at each peer before its playback time and moreover many nodes only remain connected a few minutes [16]. Surveys about P2P networks for live video distribution [17], [18], [19] show that the continuity in the playback and the delay of the video are the most important factors in the quality of experience perceived by end users. See also [20], [21] for related details. A high cooperation between peers, and specially an efficient piece selection strategy, are indispensable in order to obtain high playback continuity and low latency in a P2P streaming network [22].

In this paper, we present a new piece selection strategy that has better results than previously considered proposals. For its design, we define a Combinatory Optimization Problem (COP), translated into a suitably Asymmetric Traveling Salesman Problem (ATSP), and solved metaheuristically following an Ant Colony Optimization (ACO) approach. The paper is organized as follows. Section II describes a simple model of piece selection strategies proposed in [23]. In Section III some popular piece selection strategies are presented. Section IV describes a new family of piece selection strategies and some of their properties. A quality measure for each member of this family is defined, which allows to propose a COP in order to look for optimal strategies. In Section V we translate the proposed COP into an ATSP model. The latter helps us to find an initial seed in order to initialize the pheromones of an ACO-based solution. Finally, in Section VI, we give a comparison between numerical results obtained with the proposed strategies with respect to previous ones, and the main conclusions of this work.

9781-4244-3941-6/09/$25.00

II. Simple model for sharing (from [23])

Let us consider a P2P network composed of M identical peers, each one with a buffer of size N, and a single server which has the entire video content. Time is slotted and in each slot, the server chooses one peer at random and uniformly to send the present video piece. Peers interchange the pieces using some cooperation strategy. They are responsible to assemble these pieces so as to obtain the desired video. At each time slot, every peer displays the oldest piece in its buffer (i.e., the piece at position N). A distortion is perceived by the user if she could not obtain this piece in time. See Figure 1 for a graphical description.

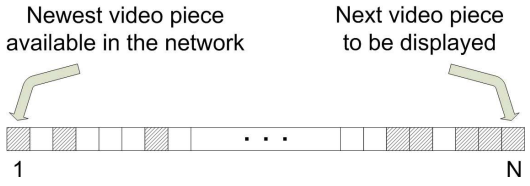

Fig. 1. Buffer model at each peer. Position 1 represents the newest video piece in the network, and N represents the next piece to be displayed. Observe that all peers are synchronized in the buffer consumption.

Pieces are also obtained from other peers. In general, and following some specific strategy, a peer A will look at a given position in its buffer. If it is empty, the peer will ask some other peer B for the missing piece. If the initial buffer position is already filled, the requesting peer goes to some other position and repeats the same iteration until it finds an empty one. If B does not have the corresponding piece, then A can ask for other piece it needs (looking again for an empty position in its buffer). This scheme leads either to a success, when A finally gets a piece from B, or to a failure, when there was no empty position in A's buffer that could be filled by a piece coming from B. The *extension* of the query is the number of buffer positions that A needs to examine in order to get a new piece. It is assumed that the whole query lasts no more than a time slot.

Let us call p_i the probability that a peer has the correct piece in the ith position of its buffer (by symmetry, assuming all peers use the same strategy, p_i is independent of the peer, and assuming stationarity, it does not depend on time neither). Consider that a particular peer k selected peer h to download a piece. Using a specific piece selection strategy, suppose that at some time t, the piece corresponding to k's buffer position i is missing, so, desirable for peer k, and owned by peer h. The probability of this event is denoted by s_i (for the reasons mentioned above, it only depends on i). The function mapping $i \in \{1, \ldots, N\}$ into $s_i \in [0, 1]$ is called the *selection* function of the strategy. Defining the index of the initial piece of the server as 1, it is shown in [23] that:

$$p_1 = 1/M, \tag{1}$$

$$p_{i+1} = p_i + (1 - p_i)p_i s_i, \quad i = 1, \ldots, N-1. \tag{2}$$

This shows that, as a function of i, p_i is monotone in the playback direction ($p_i \leq p_{i+1}$).

The continuity of playback is measured by $C = p_N$, i.e., the probability of having the next piece to be reproduced. The start up latency of the video streaming is measured by

$$L = \sum_{i=1}^{N} p_i,$$

which represent the start-up time (measured in slots) that is necessary to reach the stationary state (typically configured as the buffering time).

This paper is focused on proposing a new piece selection strategy with high continuity C, and at the same time, low latency L.

III. Classical strategies and a mixture

This section presents two classical piece selection strategies, named Rarest First and Greedy, and a third one defined as a mixture of them. Rarest First enjoys a prestigious place nowadays, because of its origins with the popular BitTorrent system [24]. With a Rarest First strategy the peer chooses to download the pieces that are locally rarest, in order to guarantee a high piece diversity. In a streaming context, it basically consists in selecting a missing piece that the contacted peer has and looks initially far away from the playback (for rare pieces, because of monotonicity of p), that is, starting from the beginning of the buffer. This leads to the following expression for s_i:

$$s_i = (1 - \frac{1}{M}) \prod_{j=1}^{i-1} \Big(p_j + (1 - p_j)^2 \Big). \tag{3}$$

In words, the piece at position i is selected under the following conditions: the server did not send the piece to the requesting peer, the requesting peer does not have the piece and the contacted peer does, and at positions 1 to $i-1$ either the requesting peers had already the corresponding piece or neither of the two peers have it.

The Greedy strategy is similar to the previous one, but looks first for the pieces nearest to the playback. This leads to the following expression for the probability s_i:

$$s_i = (1 - \frac{1}{M}) \prod_{j=i+1}^{N-1} \Big(p_j + (1 - p_j)^2 \Big). \tag{4}$$

Figure 2 shows graphically the Rarest First and Greedy strategies.

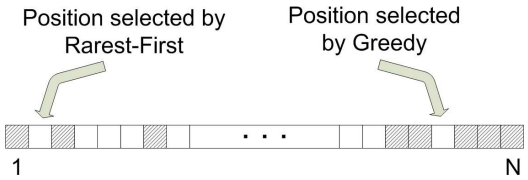

Fig. 2. Rarest First and Greedy buffer strategies

It is well known that Greedy achieves low latency, but it is not as scalable as Rarest First [25], [23]. A mixture of these

two strategies is possible, by simply dividing the buffer in two parts, from positions 1 to some index value m, $1 \leq m \leq N$, and from $m+1$ to N, and by and applying the Rarest First strategy in the first section and Greedy in the second one. This combination offers a more reduced latency than Rarest First and still possesses good continuity [23].

IV. A NEW FAMILY OF STRATEGIES

Let us consider a permutation of the first $N-1$ buffer positions:

$$\pi : \{1, \ldots, N-1\} \to \{1, \ldots, N-1\},$$
$$\forall i \neq j,\ \pi(i) \neq \pi(j).$$

With each permutation π we can associate the strategy defined as follows. Let us call A the requesting peer and B the requested one. In the first step, A examines its buffer at position $\pi(1)$. If the chunk is missing and B has it, the download is performed. Otherwise (the chunk is not missing, or it is, but B does not have it), A goes to position $\pi(2)$ and the process repeats (until a success or a failure, the two only possible final results).

This scheme defines $(N-1)!$ different strategies. We call them *permutation-based* strategies. For the strategy associated with permutation π, the corresponding selection probability expression is:

$$s_{\pi(i)} = (1 - \frac{1}{M}) \prod_{j=1}^{i-1} \Big(p_{\pi(j)} + (1 - p_{\pi(j)}) \Big)^2. \tag{5}$$

Observe that the identity permutation and the inverse one define the Rarest First and Greedy strategies respectively. The following permutation π defines the mixture of them associated with index m:

$$\pi(i) = i, \quad i = 1, \ldots, m; \tag{6}$$
$$\pi(i) = N - (i - m), \quad i = m+1, \ldots, N-1. \tag{7}$$

This family of strategies enjoys at least three useful properties, which guide us to define a measure of quality, and an algorithm for finding an efficient one.

Lemma IV.1. *$s_{\pi(i)}$ is a strictly monotone decreasing sequence.*

Proof: By definition, all permutation strategies satisfy the following recursion:

$$s_{\pi(1)} = 1 - \frac{1}{M} \tag{8}$$
$$s_{\pi(i+1)} = s_{\pi(i)} \big[p_{\pi(i)} + (1 - p_{\pi(i)})^2 \big] \tag{9}$$
$$i = 1, \ldots, N-2.$$

It is easy to verify by induction that, for each position i, we have $p_i \in \Big[\frac{1}{M}, 1 \Big)$. Then, the following relation holds:

$$s_{\pi(i+1)} - s_{\pi(i)} = s_{\pi(i)} (p_{\pi(i)} (p_{\pi(i)} - 1)) < 0, \tag{10}$$
$$i = 1, \ldots, N-2,$$

showing the monotonicity of $s_{\pi(i)}$. ∎

The $N-2$ equations defined by (9), together with the $N-1$ equations given in (2), allow to obtain a nonlinear determined system of size $2N-3$, in which the unknown parameters are $\{p_i\}_{i=2,\ldots,N} \cup \{s_i\}_{i=1,\ldots,N-1} - \{s_{\pi(1)}\}$. The system can easily be solved in $O(N)$ time. As a consequence, it is to possible to evaluate the performance (continuity and latency) for any particular permutation.

The following lemma will be needed in next section.

Lemma IV.2. *Consider a sequence of reals $(x_1, \ldots, x_N)$ where $x_i \neq x_j$ if $i \neq j$. Then, there exists a permutation π with the following property: in the associated strategy, the corresponding selection function satisfies $s_i > s_j$ whenever $x_i > x_j$.*

Proof: Given any sequence $(x_1, \ldots, x_N)$ such that $x_i \neq x_j$ if $i \neq j$, there is a permutation $\pi = (i_1, \ldots, i_N)$ such that

$$x_{i_1} > x_{i_2} > \cdots > x_{i_{N-1}}.$$

If s is the selection function of the strategy associated with π, then by Lemma IV.1 we have

$$s_{\pi(1)} > s_{\pi(2)} > \cdots > s_{\pi(N-1)}.$$

Therefore

$$s_{i_1} > s_{i_2} > \cdots > s_{i_{N-1}}.$$

∎

We propose now a definition of the quality of any permutation-based strategy. Let X_π be the random variable that counts the number of queries for obtaining a piece, using the strategy associated with permutation π (what we called the extension if the strategy).

Definition IV.3. *The quality Q_π of the strategy associated with permutation π is its expected number of queries: $Q_\pi = E(X_\pi)$.*

It is desirable that this expected value is as large as possible, given that a large number of queries means that peers have many filled buffers (the needed time is not a problem because it is assumed that in the worst case, in the longest one, it is shorter than a time slot). The goal of the COP proposed next is to maximize $E(X_\pi)$ among all possible permutation-based strategies, subject to the constraints given in (2) and (9).

Last, the following result provides a means for the evaluation of Q_π.

Proposition IV.4. *The expected number of queries needed to obtain a piece with the strategy associated with permutation π is:*

$$E(X_\pi) = \frac{M}{M-1} \sum_{i=1}^{N-1} i \big(p_{\pi(i)+1} - p_{\pi(i)} \big). \tag{11}$$

Proof: Let α_i be the probability of having a successful query in step i. Then:

$$\begin{aligned} E(X_\pi) &= \sum_{i=1}^{N-1} i\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j) \\ &= \sum_{i=1}^{N-1} ip_{\pi(i)}(1-p_{\pi(i)}) \prod_{j=1}^{i-1}\left(p_{\pi(j)} + (1-p_{\pi(j)})^2\right) \\ &= \frac{1}{1-\frac{1}{M}} \sum_{i=1}^{N-1} ip_{\pi(i)}(1-p_{\pi(i)})s_{\pi(i)} \\ &= \frac{M}{M-1} \sum_{i=1}^{N-1} i\left(p_{\pi(i)+1} - p_{\pi(i)}\right). \end{aligned}$$

■

V. Translation of the Problem

A. Previous Search Experience

An exhaustive study of all permutation strategies can be done only for limited buffer sizes, because of the prohibitive computational effort. A concept of pseudodistance between permutations will be useful for treating this problem via heuristics.

Definition V.1. *The pseudodistance between two permutations is the minimum number of swaps needed to transform one permutation into the other.*

Figure 3 presents the pseudo-code of Algorithm 1, which accepts a desired probability of occupation, and constructs an initial permutation which is expected to obtain a probability of occupation p' similar to that of the input. Finally, a Local Search is applied, updating in each step the permutation with the best neighbor permutation.

Algorithm 1 was applied to many different inputs p for testing purposes. The first input sequence that was tested was exponential:

$$p_\epsilon(i) = M^{-\frac{N-i}{N-1}}, \quad i = 1, \ldots, N.$$

In this case, s_{ideal} (see Algorithm 1) is monotonous increasing, and the algorithm falls in the Greedy strategy. The second input was a segmented line shown in Figure 4 for the case $M = 1000$, $N = 40$. The correspondent output can be seen in Figure 5.

It can be seen that a peak in the permutation strategy generates an abrupt change in the objective (a change in slopes). The place of the peak allows to choose the position of the change in the behavior of the objective sequence.

The following tests are based on the generation of a peak. A direct peak in the middle of the buffer results in higher continuities than Rarest First but also in higher startup latencies.

Algorithm 1 PermutationFinderAlgorithm

Input: Probability of occupation: $p_i, \quad i = 1, \ldots, N$
Number of iterations: n
Output: Permutation π

{Construction of an initial permutation}
1: Obtain the ideal strategy sequence $s_{ideal}(i)$ that achieves the latter occupation probabilities p, through:

$$s_{ideal}(i) = \frac{p_{i+1} - p_i}{(1-p_i)p_i}, \; i = 1, \ldots, N-1.$$

2: Find the permutation π that better approximates s_{ideal} such as it was stated in Lemma IV.2.
3: Solve the nonlinear system defined by equations (2) and (9), in order to find the correspondent selection function of the permutation π, s_π, and the new probabilities p'.
4: Compute the quality of the strategy asociated with π, defined by:

$$Q_\pi = E(X_\pi) = \frac{M}{M-1} \sum_{i=1}^{N-1} i(p'_{\pi(i)+1} - p'_{\pi(i)}).$$

{Local Search phase}
5: **for each** $j \in \{1, \ldots, n\}$ **do**
Look for the best permutation π_{new} at distance 1 from π, with quality Q_{new}.
if $Q_{new} > Q$ **then**
update: $Q \leftarrow Q_{new}$ and $\pi \leftarrow \pi_{new}$
else {this permutation is better than any neighbor}
break and **finish**
end if
end for

Fig. 3. Permutation Finder Algorithm. It receives a probability of occupation p and a number of iterations n, and returns a permutation π.

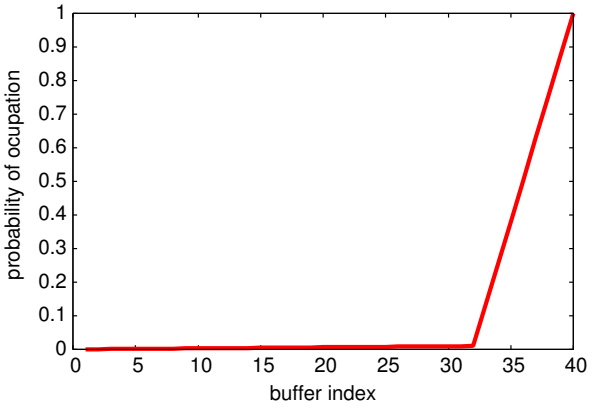

Fig. 4. Segmented objective

Definition V.2. *The subfamily of permutations which present a peak is defined as follows:*

$$\pi(i) = N - i, \qquad i = 1, \ldots, I \tag{12}$$

$$\pi(I + j) = j, \qquad j = 1, \ldots, J \tag{13}$$

$$\pi(I + J + k) = \left\lfloor \frac{N + J - I}{2} \right\rfloor + \left\lceil \frac{k}{2} \right\rceil (-1)^{k+1}, \tag{14}$$
$$k = 1, \ldots, N - I - J - 1.$$

The peak is reduced in magnitude because of the selection of the firsts $I + J$ index permutations and the monotonicity of $s_{\pi(i)}$. This produces lower latencies than in previous cases. Moreover, the decision of the firsts queries in the extremes

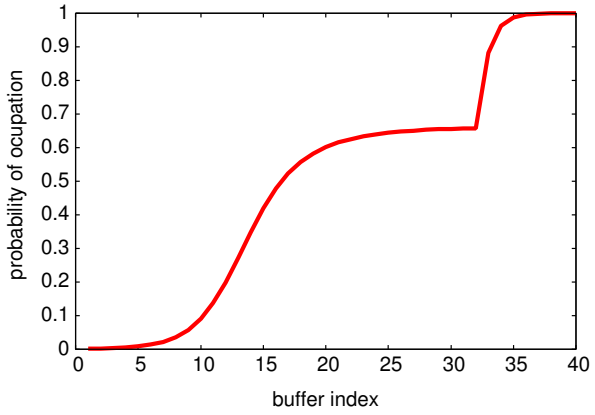

Fig. 5. Objective obtained in the segmented case

of the buffer helps (while not abruptly) in incrementing the continuity of playback.

B. Translation of the Problem

We translated the COP problem into an instance of the well-known TSP model as sketched in Figure 6. An N-clique allows to get a bijection between a cycle and a given permutation. Observe that the TSP must be asymmetric, otherwise Greedy and Rarest First define the same cycle, while being naturally different in the original problem. In order to define a distance to the edges, we use an Ant exploration learning mechanism. We enumerate the nodes from 1 to N, and put an ant at the auxiliary starting node N. The first ant chooses the edges randomly and uniformly, without repeating nodes. After a cycle, the visited edges are weighted according to the obtained quality of this cycle, as expressed in Line 1 of Algorithm 2 and detailed in the function "InitializeEdges" (see Figure 7).

Ants find their best path through pheromones. The initial pheromone weights are taken by considering the subfamily of permutations, as it is detailed in the Function *InitializePheromone* in Figure 8.

C. ACO Implementation

Once a weighted graph is obtained the ATSP must be solved. Given that the exploring nature of the problem is similar to those of ants, reduced computational effort and simplicity of implementation, we decided to choose an ACO-based metaheuristic, mixed with a Local Search.

At each step, the ants take their decisions by means of the following probabilities [26] (see Function ApplyACO detailed in Figure 9):

$$p_{i,x} = \frac{Pheromone(i,x)^{\alpha}\ Distances(i,x)^{-\beta}}{\sum_{possible\ y} Pheromone(i,y)^{\alpha}\ Distances(i,y)^{-\beta}}$$

After each ant makes its cycle, the pheromone is updated based on $E(X_\pi)$, with priority to latency, given that the subfamily has an important margin to improve latency, but continuity:

$$Pheromones(edge(j)) = \frac{10(N-j)Q}{Q_{max}\sum_{i=1}^{N} p_i}$$

Algorithm 2 ACOBasedSearchforπ
Input:
Subfamily of permutations: $SubFamily$
Number of ants: $ants$
Number of iterations: $iterations$
Maximum local iterations: $maxiter$
Output:
Permutation π

{Construction of the network}
1: $Distances = InitializeEdges(ants)$
2: $Pheromones = InitializePheromone(SubFamily)$

{Resolution of the resulting ATSP}
3: $\pi = ApplyACO(Distances, Pheromones, iterations)$
4: Solve the nonlinear system defined by equations (2) and (9), in order to find the probabilities p_i associated with π.
5: Calculate the quality of the strategy associated with π, defined by:

$$Q_\pi = E(X_\pi) = \frac{M}{M-1}\sum_{i=1}^{N-1} i(p_{\pi(i)+1} - p_{\pi(i)}).$$

{Local Search phase}
6: **for each** $j \in \{1, \ldots, maxiter\}$ **do**
Look for the best permutation π_{new} at distance 1 from π, with quality Q_{new}.
if $Q_{new} > Q$ **then**
update: $Q \leftarrow Q_{new}$ and $\pi \leftarrow \pi_{new}$
else {this permutation is better than any neighbor}
break and **finish**
end if
end for

Fig. 6. ACO-Based Permutation finder. It consists of a Construction of the network which captures the original problem, a Resolution of the resulting ATSP model and a final Local Search phase.

The ACO output's cycle is built considering the most visited edge in each step, without repeating nodes. This output cycle enters the Local Search Phase, in which the permutation is changed at each step by its best permutation neighbor. The process finishes when we cannot improve the quality. It is worth observing that in each iteration of the local search we need to measure $\binom{N-1}{2} = \frac{(N-1)(N-2)}{2}$ quality values, versus a total of $(N-1)!$ in a linear search between all permutations. The former is computationally tractable, and the algorithm provides a good trade-off between quality and time consumed.

The adaptation of ACO's parameters is based on [27], and then tuned in accordance with our particular problem. Our final implementation used Algorithm 2 with $\alpha = 0.4, \beta = 1.5, \rho = 0.5$ and 100 ants. Figure 10 shows the result of a comparison between previous strategies and the output of Algorithm 2 for the case $N = 30$ and $M = 100$. Table I shows that the obtained permutation achieves an excellent continuity and at the same time a latency comparable to the one reached by Greedy.

Function InitializeEdges
Input:
Number of ants: $ants$
Buffer size: N
Output:
Distance between nodes: $Distances$

{Initialization}
1: $Distances = ones(N)$
2: $Q_{max} = Q_{RarestFirst}$

{Cycle of each ant}
3: **for each** $i \in \{1, \ldots, ants\}$ **do**
4: **for each** $j \in \{1, \ldots, N-1\}$ **do**
5: Draw Step j of the cycle weighting probabilities according with Distances of possible nodes.
6: **end for**
7: Solve the nonlinear system defined by equations (2) and (9), in order to find its corresponding probability of occupation p_i.
8: Calculate the quality of the strategy asociated with π, obtained by the ant:

$$Q_\pi = E(X_\pi) = \frac{M}{M-1} \sum_{i=1}^{N-1} i(p_{\pi(i)+1} - p_{\pi(i)}).$$

if $E(X_\pi) > Q_{max}$ **then**
update: $Q_{max} = E(X_\pi)$
end if

{Update the distance between used edges}
9: **for each** $j \in \{1, \ldots, N-1\}$ **do**
$Distances(edge(j)) = \frac{10(N-j)E(X_\pi)}{Q_{max}}$
end for
end for

Fig. 7. Initialization of distance between nodes of the network. Each ant makes a stochastic cycle considering probabilities of each walk according to distances.

Function InitializePheromone
Input:
Subfamily of permutations: $SubFamily(I, J)$
Buffer size: N
Output:
Pheromones for each edge: $Pheromone$

{Initialization}
1: $Pheromone = ones(N)$
2: $Q_{max} = Q_{RarestFirst}$

{Cycle of each ant}
3: **for each** $i \in \{1, \ldots, I\}$ **do**
4: **for each** $j \in \{1, \ldots, J\}$ **do**

{Each ant makes the deterministic cycle}
5: $\pi = SubFamily(i, j)$
6: Solve the nonlinear system defined by equations (2) and (9).
7: Calculate the quality of the strategy asociated with π:
$E(X_\pi) = \frac{M}{M-1} \sum_{i=1}^{N-1} i(p_{\pi(i)+1} - p_{\pi(i)})$

if $E(X_\pi) > Q_{max}$ **then**
update: $Q_{max} = E(X_\pi)$
end if

{Update the Pheromones of used edges}
8: **for each** $k \in \{1, \ldots, N-1\}$ **do**
$Pheromones(edge(k)) = \frac{10(N-k)E(X_\pi)}{Q_{max}}$
end for
end for
end for

Fig. 8. Pheromones of the ACO Application are initialized according to the experience obtained by the subfamily of permutations

Function ApplyACO(Distances, Pheromones, iterations)
Input:
Cost of each edge: $Distances$
Pheromones of edges: $Pheromones$
Number of ants: $iterations$
Output:
Best cycle: π

{Initialization}
2: $Q_{max} = \frac{E(X_{RarestFirst})}{LatencyRarestFirst}$

{Each ant makes a stochastic cycle}
1: **for each** $i \in \{1, \ldots, ants\}$ **do**
2: **for each** $j \in \{1, \ldots, N-1\}$ **do**
3: Draw the next node x of ant i according with:

$$p_{i,x} = \frac{Pheromone(i,x)^\alpha \ Distances(1,x)^{-\beta}}{\sum_{Possible \quad y} Pheromone(i,y)^\alpha \ Distances(1,y)^{-\beta}}$$

end for
5: Be π the cycle of ant i. Solve equations (2) and (9) in order to find its corresponding probability of occupation, named p_i.
6: Calculate the quality of the strategy asociated with π, done by ant i, with priority to latency:

$$Q = \frac{NE(X_\pi)}{\sum_{i=1}^{N} p_i}$$

7: **if** $Q > Q_{max}$ **then**
update: $Q_{max} = Q$
end if

{Update the Pheromones of used edges}
8: **for each** $j \in \{1, \ldots, N-1\}$ **do**
$Pheromones(edge(j)) = \frac{10(N-j)Q}{Q_{max} \sum_{i=1}^{N} p_i}$
end for
end for

{Find output cycle}
9: **for each** $j \in \{1, \ldots, N-1\}$ **do**
$cycle(edge(j)) = MostVisitedEdge(node)$
end for

Fig. 9. Pheromones of the ACO Application are initialized according to the experience of nice results obtained by the subfamily of permutations

TABLE I
PERFORMANCE OF DIFFERENT STRATEGIES.

Strategies	Continuity	Latency
Rarest First	0.9571	21.0011
Greedy	0.9020	4.1094
Mixture	0.9953	11.1253
Algorithm 2	0.9998	7.9821

VI. CONCLUSIONS AND FUTURE WORK

This paper defines a new family of piece selection strategies based on permutations of the linear order of piece requests. An outstanding property of this family is that the ascendant-descendant behavior of every injective sequence can be approximated by a strategy of the family. This allows to define a first algorithm which provides a subfamily of strategies with nice performances. The quality of each permutation in this subfamily plays the role of an initial pheromone for a second ACO-based algorithm.

The weights of the ATSP model is based in biased walks

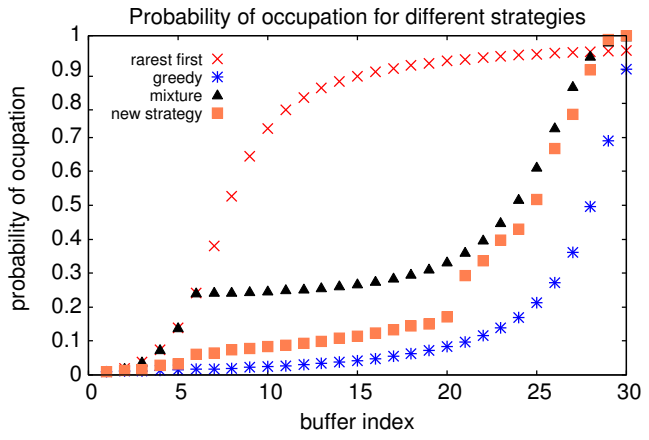

Fig. 10. Comparison between different strategies

of ants. The final results show that the initial COP problem was correctly captured via the ATSP model, and that the Ant Colony Optimization allowed to find much better results than the ones obtained via classical strategies. Moreover, the results were better than the obtained by the members of the subfamily, which served as a seed in the second algorithm.

Our future work is focused on adapting the mathematical model in order to capture the natural dynamism and heterogeneity present in P2P Streaming Networks. This should be directly applied to video streaming networks, and particulary to the Goalbit platform.

ACKNOWLEDGMENTS

This work was partially supported by project "Sistema eficiente de distribución de video y TV en tiempo real" of the national Uruguayan telephony operator ANTEL, the French DGE project "P2Pim@ges", and the French-Uruguayan ECOS project "Réseaux sans-fil de type mesh et applications multimédia P2P: outils pur la guarantie de la qualité d'expérience".

REFERENCES

[1] C. D. Cranor, M. Green, C. Kalmanek, D. Shur, S. Sibal, J. E. V. der Merwe, and C. J. Sreenan, "Enhanced streaming services in a content distribution network," *IEEE Internet Computing*, vol. 5, no. 4, pp. 66–75, 2001.

[2] S. Wee, W. Tan, J. Apostolopoulos, and S. Roy, "System design and architecture of a mobile streaming media content delivery network (msdcdn)," Streaming Media Systems Group, HP-Labs, Tech. Rep., 2003.

[3] P. Rodríguez-Bocca, "Redes de Contenido: Taxonomía y Modelos de evaluación y diseño de los mecanismos de descubrimiento de contenido." Master's thesis, Universidad de la República, Facultad de Ingeniería, Intituto de Computación. ISSN 0797-6410 INCO-RT-05-13, Montevideo, Uruguay, October 2005.

[4] S. Horovitz and D. Dolev, "LiteLoad: Content unaware routing for localizing P2P protocols," in *Proceeding of the IEEE International Symposium on Parallel and Distributed Processing (IPDPS'08)*, Miami, USA, April 2008, pp. 1–8.

[5] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu, "Peer-to-peer computing," HP Labs, Tech. Rep. HPL-2002-57, 2002. [Online]. Available: citeseer.ist.psu.edu/milojicic02peertopeer.html

[6] PPLive Home page, http://www.pplive.com, 2007.

[7] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross, "Insights into pplive: A measurement study of a large-scale p2p iptv system," in *In Proc. of IPTV Workshop, International World Wide Web Conference*, 2006.

[8] SopCast - Free P2P internet TV, http://www.sopcast.org, 2007.

[9] PPStream home page, http://www.ppstream.com/, 2007.

[10] TVAnts home page, http://cache.tvants.com/, 2007.

[11] TVUnetworks home page, http://tvunetworks.com/, 2007.

[12] GoalBit - The First Free and Open Source Peer-to-Peer Streaming Network, http://goalbit.sf.net/, 2008.

[13] M. E. Bertinat, D. D. Vera, D. Padula, F. Robledo, P. Rodríguez-Bocca, P. Romero, and G. Rubino, "Goalbit: The first free and open source peer-to-peer streaming network," in *To appear in Proceedings of the 5th international IFIP/ACM Latin American conference on Networking*. New York, USA: ACM, 2009.

[14] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Multimedia Computing and Networking*, 2002.

[15] D. Zeinalipour-Yatzi and T. Folias, "A quantitative analysis of the gnutella network traffic," Department of Computer Science, University of California, Riverside, Tech. Rep., 2002.

[16] K. Sripanidkulchai, B. Maggs, and H. Zhang, "An analysis of live streaming workloads on the internet," in *IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*. New York, NY, USA: ACM Press, 2004, pp. 41–54.

[17] A. Sentinelli, G. Marfia, M. Gerla, L. Kleinrock, and S. Tewari, "Will iptv ride the peer-to-peer stream?" *Communications Magazine, IEEE*, vol. 45, pp. 86–92, 2007.

[18] S. Tewari and L. Kleinrock, "Analytical model for bittorrent-based live video streaming," in *4th IEEE Consumer Communications and Networking Conference (CCNC'07)*, Las Vegas, NV, January 2007, pp. 976–980.

[19] S. Alstrup and T. Rauhe, "Introducing octoshape - a new technology for large-scale streaming over the internet," European Broadcasting Union (EBU), Tech. Rep., 2005.

[20] P. Rodríguez-Bocca, "Quality-centric design of Peer-to-Peer systems for live-video broadcasting," Ph.D. dissertation, INRIA/IRISA, Université de Rennes I, Rennes, France, april 2008.

[21] S. Mohamed and G. Rubino, "A Study of Real–time Packet Video Quality Using Random Neural Networks," *IEEE Transactions On Circuits and Systems for Video Technology*, vol. 12, no. 12, pp. 1071–1083, Dec. 2002.

[22] B. Cohen, "Incentives build robustness in bittorrent," *www.bramcohen.com*, vol. 1, pp. 1–5, May 2003.

[23] Y. Zhou, D. M. Chiu, and J. Lui, "A Simple Model for Analyzing P2P Streaming Protocols," in *Proceeding of the IEEE International Conference on Network Protocols (ICNP'07)*, Beijing, China, October 2007, pp. 226–235.

[24] A. Legout, G. Urvoy-Keller, and P. Michiardi, "Rarest first and choke algorithms are enough," in *IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*. New York, NY, USA: ACM, 2006, pp. 203–216.

[25] C.-J. Wu, C.-Y. Li, and J.-M. Ho, "Improving the download time of bittorrent-like systems," in *IEEE International Conference on Communications 2007 (ICC 2007)*, Glasgow, Scotland, June 2007, pp. 1125–1129.

[26] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[27] H. Duan, G. Ma, and S. Liu, "Experimental study of the adjustable parameters in basic ant colony optimization algorithm," *IEEE Congress on Evolutionary Computation*, vol. 1, no. 1, pp. 149–156, 2007.