DISTRIBUTED RATE ALLOCATION AND PERFORMANCE OPTIMIZATION FOR VIDEO COMMUNICATION OVER MESH NETWORKS

Bo Wang, Zhihai He

Dept. of Electrical and Computer Engineering
University of Missouri, Columbia, MO 65211
HeZhi@missouri.edu

Yu Sun

Department of Computer Science
University of Central Arkansas, Conway, AR
yusun@uca.edu

ABSTRACT

Video streaming imposes high rate requirement and stringent constraints on resource limited mesh networks. In this work, we develop a distributed asynchronous particle swarm optimization (DAPSO) algorithm for resource allocation and performance optimization scheme for video communication over large-scale mesh networks. Unlike many network resource allocation performance optimization algorithms in the literatures which are able to handle convex network utility functions, the proposed scheme is able to handle generic nonlinear network utility functions. We will use a specific rate allocation and quality optimization problem for an example to demonstrate the efficiency of the proposed scheme and compare its performance with other algorithms, such as distributed gradient search.

Keywords - Mesh networks, video streaming, distributed rate allocation, particle swarm optimization.

1. INTRODUCTION

In large-scale video mesh networks, a large number of sender devices transmit compressed video data, either storage or live video data, to a large number of receivers through multi-hop transmission. This type of video mesh networking technology is found in many important applications, such as web video communication, community video networking services, emergence response, battlefield communication, etc.

Recently, several distributed resource allocation and performance optimization have been proposed for networks [1, 2, 3, 4, 5, 6]. These algorithms are designed for network transmission of generic data often and assume network utility functions of relatively simple forms, such as additive and convex (or concave) functions. However, within the context of video communication over networks, the relationship between the video quality of service metric (or system performance metric) and resource utilization parameters are often nonlinear and complex. Therefore, there is a need to develop a distributed asynchronous optimization algorithm which is able to handle generic nonlinear network utility functions.

In this work, we develop a distributed asynchronous particle swarm optimization (DAPSO) algorithm for rate allocation and performance optimization scheme for video communication over large-scale mesh networks. Unlike many network resource allocation performance optimization algorithms in the literatures which are able to handle convex network utility functions, the proposed scheme is able to handle generic nonlinear network utility functions.

The rest of this paper is organized as follows. Section 2 presents the optimization problem. Section 3 describes DAPSO algorithm. Section 4 presents the experimental results of DAPSO on wireless video sensor networks. Section 5 gives the conclusions of this paper.

2. FORMULATION OF RATE ALLOCATION PROBLEMS

We model the mesh network as a graph with V network nodes $V = \{1, 2, \cdots V\}$ and L logical links $L = \{1, 2, \cdots L\}$. The mesh network is shared by a set of video transmission streams, denoted by $S = \{1, 2, \cdots, S\}$. In performance optimization of video mesh networks under resource constraints, all network nodes need to collaborate in their resource utilization behaviors so as to maximize the overall system performance under resource constraints. Let $X = \{x_1, x_2, \cdots, x_N\}$ be the set of resource parameters, such as encoding bit rate of a video stream.

Suppose we have N video streams that are sharing communication links and crossing over the network. Let $x_n \in X = \{x_1, x_2, \cdots, x_N\}$ be the bit rate of video stream n. Let

$$\mathcal{X}_L[l] = \{x_n \in X | \textit{video stream } n \textit{ uses link } l\}. \quad (1)$$

And let C_l be the link capacity. According the link capacity constraint, we have

$$\sum_{x_n \in \mathcal{X}_L[l]} x_n \le C_l, \quad 1 \le l \le L. \quad (2)$$

A commonly used metric for measuring the quality of a single video stream is the coding distortion. As suggested by the literature on rate-distortion (R-D) modeling for video coding, we use the following R-D model

$$D(x_n) = \sigma_n^2 \times 2^{-\lambda x_n}, \quad (3)$$

to describe the relationship between video coding distortion D and source rate x_n for video stream n. Here, σ_n^2 represents the picture variance, and λ is an encoder-related parameter. It should be noted that in this work we just use the R-D model in (3) as an example to demonstrate the proposed DAPSO algorithm. Certainly, this model can be

replaced by any other more accurate R-D model developed in the literature. As we can see, the optimization procedure of the proposed DAPSO algorithm does not depend on the specific expression of the R-D model.

One commonly used measure to describe the overall video quality of multiple video streams is the aggregated video distortion, i.e.,

$$U(x_1, x_2, \cdots, x_N) = \sum_{n=1}^{N} D(x_n) = \sum_{n=1}^{N} \sigma_n^2 \times 2^{-\lambda x_n}. \quad (4)$$

As noted by a number of researchers, when characterizing the overall quality over multiple video streams, besides minimizing the aggregated video distortion, we also need to minimize the quality variation between different video streams. From a user's perspective, minimizing the quality variation is also a very important part of maintaining the fairness among users and different video services. Now, the rate allocation and performance optimization can be formulated as

$$\begin{aligned} \min \quad & U(x_1, \cdots, x_N) = \sum_{n=1}^{N} [w_1 D(x_n) + w_2 |D(x_n) - \bar{D}|], \\ s.t. \quad & \sum_{x_n \in \mathcal{X}_L[l]} x_n \le C_l, \quad 1 \le l \le L. \end{aligned} \quad (5)$$

Here $\bar{D} = \frac{1}{N} \sum_{n=1}^{N} D(x_n)$. From this rate allocation example we can see that the network utility function in video communication over networks $U(x_1, x_2, \cdots, x_N)$ is often a nonlinear non-convex (or non-concave) function. It should be noted for many other resource parameters, such as transmission power, channel coding rate, etc, their network utility functions could also be non-convex.

For large-scale networks, the performance optimization problems in (5) is often a high-dimensional nonlinear non-convex constrained optimization problem. Existing methods developed for convex optimization and existing algorithms for distributed rate allocation, flow control, and resource allocation, such as distributed gradient search, cannot be applied. In this work, based on a swarm intelligence principle, we develop a distributed asynchronous particle swarm optimization (DAPSO) algorithm to solve the constrained nonlinear rate allocation problem in (5).

3. DISTRIBUTED AND ASYNCHRONOUS PARTICLE SWARM OPTIMIZATION

3.1. Introduction to Particle Swarm Optimization

Particle swarm optimization (PSO), developed by Kennedy and Eberhart in 1995 [7], is a promising population based new optimization technique which models the set of potential problem solutions as a swarm of particles moving about in a virtual search space. Some attractive features of PSO include the ease of implementation, and no gradient information is required. PSO can be used to solve a wide range of different optimization problems, including most of the problems that can be solved using Generic Algorithms.

Research results demonstrate that PSO outperforms other nonlinear optimization techniques, such as genetic algorithm and simulated annealing [7]. The high-level idea of PSO can be summarized as follows. To find the minimum of an objective function $f(\mathbf{x})$ ($\mathbf{x}$ is a vector) within a solution space P, the PSO algorithm starts with a set of candidate solutions (called particles), $\{x_m | 1 \le m \le M\}$ distributed in P. A typical value of M is between 20 and 50 [7]. During the optimization process, each particle x_m moves within the solution space in search for the minimum of $f(\mathbf{x})$, and the corresponding movement path is denoted by $x_m(t)$, where t represents time. At each time step, the movement of particle x_m is given by

$$\mathbf{x_m(t+1)} = \mathbf{x_m(t)} + \mathbf{v}, \quad (6)$$

where

$$\mathbf{v} = \mathbf{w} \cdot \mathbf{v} + \mathbf{c_1 \Theta_1 [x_p^m - x_m(t)]} + \mathbf{c_2 \Theta_2 [x_g - x_m(t)]}. \quad (7)$$

Here, w, c_1 and c_2 are weighting factors, Θ_1 and Θ_2 are two random numbers. $\mathbf{x_p^m}$ is the best solution that the particle itself has found so far, and $\mathbf{x_g}$ is the best solution that all particles have found so far. Each particle, when determining its next move, always balances the behaviors of its own and the group [7].

3.2. Overview of DAPSO

Our major idea in distributed and asynchronous PSO (DAPSO) can be summarized as follows. We propose to decompose the global optimization problem into a set of local optimization modules, each of which is associated with a communication link. For each link, which corresponds to a local optimization module, we introduce a set of local resource parameters, $X_l = \{x_{nl}\}$, where x_{nl} represents the bit rate of video stream n at local optimization module l. We define the l-th local optimization module to be

$$\begin{aligned} \min \quad & U_l(X_l) = \sum_{n \in \mathcal{X}_L[l]} [w_1 D(x_{nl}) + w_2 |D(x_{nl}) - \bar{D}^*|], \\ s.t. \quad & \sum_{n \in \mathcal{X}_L[l]} x_{nl} \le C_l, 1 \le l \le L. \end{aligned} \quad (8)$$

Here $w1$ and $w2$ are weight parameters for overall video distortion and distortion difference, $\bar{D}^* = \frac{1}{N^*} \sum_{n \in S(l)} D(x_{nl})$, and N^* is the total number of the video streams path link l. Note that, for a video stream, its bit rate on each link along the transmission path should be the same. In other words,

$$x_{nl} = x_{nk}, \quad \forall\, l, k \in \mathcal{L}_S[n], \quad (9)$$

where $\mathcal{L}_S[n]$ is the set of links used by video stream n. We can see that the original global optimization problem has been decomposed into L local optimization modules in (9) plus a set of resource constraints in (9). In the proposed DAPSO algorithm, each local constrained optimization in (9) is solved using the PSO algorithm discussed in Section 3.1. Each local PSO module has a group of local particles moving around the solution space defined by the local constraints for the local optimum. Neighboring PSO modules share status information about their "group-best", denoted by X_l^g, using this external information to guide the moves of its internal particles.

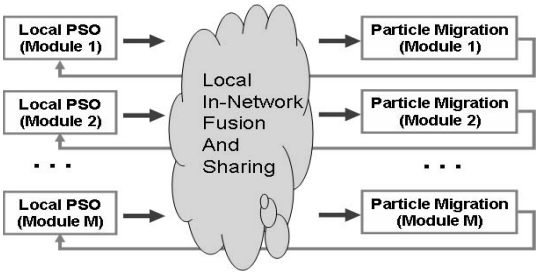

Figure 1: Distributed and Asynchronous PSO Algorithm.

The high level idea for DAPSO algorithm is illustrated in Fig. 1. One of major challenges in network resource allocation is to deal with bottleneck links where a large number of video streams share a communication link with a very limited resource capacity, as illustrated in Fig. 2. To address this bottleneck issue, we propose a method called *particle migration*, to incorporate the bottleneck information sharing into in-network fusion. Our basic idea is to introduce a resource budget window for each resource parameter in each local PSO modules. For example, the resource parameter x_{nl} is the bit rate of video stream n in the l-th PSO module. We impose the following resource budget window constraint on x_{nl}

$$C_{nl}^{-} \leq x_{nl} \leq C_{nl}^{+} \tag{10}$$

and incorporate this constraint into the local PSO module formulated in (9). During in-network fusion, in addition to using the *group-best* particles to modify their *group-best* particles, each PSO module also fuse and modify their resource budget window information. One simple fusion rule is as follows: At one check time during the operation procedure for one local PSO module, if this module gets a new *group-best* particle, it will generate a new resource budget window, and at the same time, it will also check if the current local group-best particle is inside the new *group-best* particle which it just get. If the local group-best particle is inside the new *group-best* particle, this means this source rate maybe can still reach more resource, then we need to increase its resource budget window size step by step, otherwise, the resource budget window size will be decreased step by step.

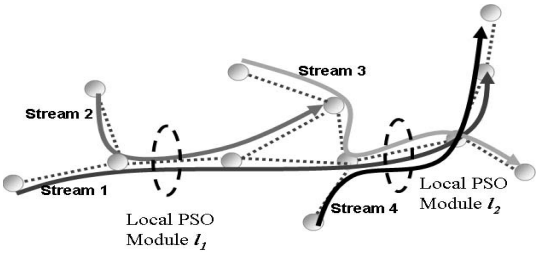

Figure 2: An example of critical link in a multihop mesh network.

The DAPSO algorithm can be stated as follows:
Step 1: Initializes local PSO module on every link l.
Step 2: After several iteration, doing in-network fusion and *particle migration*, sharing *group-best* particles from all local PSO modules during the in-network fusion.
Step 3: Generates "resource budget window" as new source rate capacity constraints for every source in each local PSO module to handle these kinds of inter-dependent resource constraints.
Step 4: When one local link gets the new capacity constraints, such as a new resource budget window and a new *group-best* particle, mapping all particles into this "resource budget window", re-initializes PSO on this local link.
Step 5: In each local PSO module, during local PSO optimization process, adapts resource budget window size for every source pathes this link.
Step 6: Go through Step 2 to Step 5 until all the local PSO modules are converged and balanced

4. EXPERIMENTAL RESULTS

We use wireless video sensor network (WVSN) as an example to demonstrate the DAPSO algorithm. We test the proposed distributed and asynchronous optimization scheme using DAPSO with different WVSN topologies. In the following experiment, a sensor network has 6 sources and 15 links was selected.

The other parameters used in DAPSO are set as follows: link capacity $c_l = 100$, particle size = 20, PSO update iteration = 10, initial window size = $0.3 \times c_l$, decrease window step size = 0.7, and increase window step size = 2. The parameters used in WVSN are set as follows: $\lambda = 0.023$, $g(P_i) = 10$, and $\sigma_1^2 = 100$, $\sigma_2^2 = \sigma_6^2 = 200$, $\sigma_3^2 = \sigma_5^2 = 400$, $\sigma_4^2 = 800$. And the weight parameters are set as follows: $w1 = 1, w2 = 0.1$.

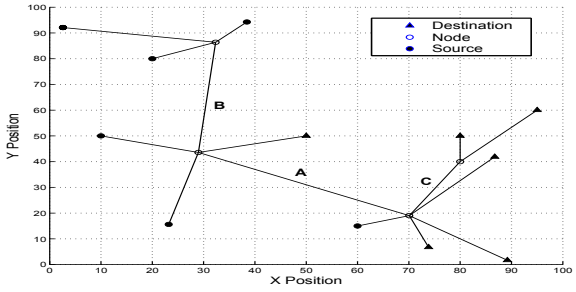

Figure 3: A WVSN topology with 6 sources and 15 links.

Fig. 3 depicts an example network topology which has 6 sources and 15 links. For each source, it pathes several links and has several different utility function value. We use the average, minimum and maximum utility function value of each source for DAPSO algorithm. Fig. 4 shows that the overall DAPSO utility function solution quickly reaches to the convergence optimum solution after several PSO update iterations.

Fig. 5, 6, and 7 also show the movement path of each source during the search process in several critical links same as the previous experiment. Here in each particle cluster, we only trace the best particle's moving path. And the critical links' capacity usage percentage for link A, B

and C are near 100%, 84% and 65% when the whole network is balanced. From these experimental results, we can see that the proposed scheme works very efficiently. Our experimental results with other settings of WSN with different source number and link number yield similar results. They are omitted here due to page limitation.

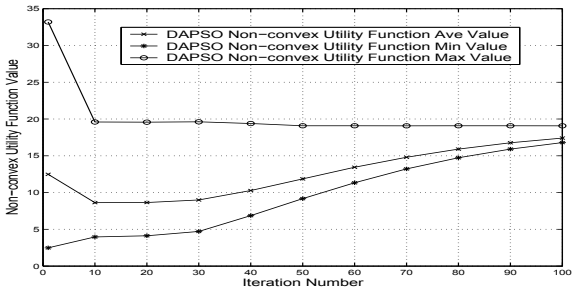

Figure 4: The performance function value decrease as the particles update their positions.

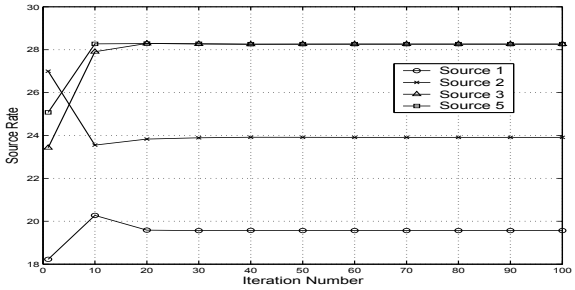

Figure 5: The traces of the particles moving in critical link A.

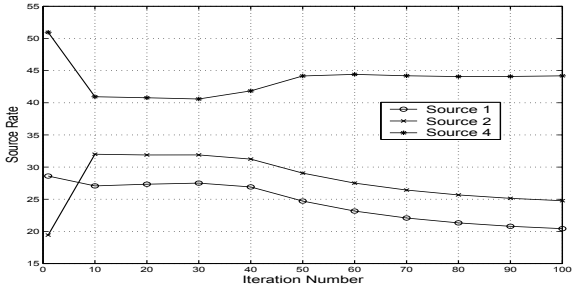

Figure 6: The traces of the particles moving in critical link B.

The weight parameters here will control the optimization results, when we change the parameters to: $w1 = 1$ and $w2 = 1$. The critical links' capacity usage percentage for link A, B and C are near 68%, 52% and 36% which drop quickly when the whole network is balanced.

5. CONCLUSION

In this paper, based on the properties of WVSN and the swarm intelligence principle, we have developed an evolutionary optimization scheme to solve the communication network distributed and asynchronous optimization problem. The basic operation involves utility function maximization optimization in each link and transmits its best local results to the network. The outstanding contribution of this paper is there are no convex or concave requirement for the utility function, which are required in other traditional distributed optimization algorithms. Simulation results show that our evolutionary optimization scheme is very efficient for different network topology. In our future work, we plan to extend our work to solve the optimization problem when the network circumstance is not stable.

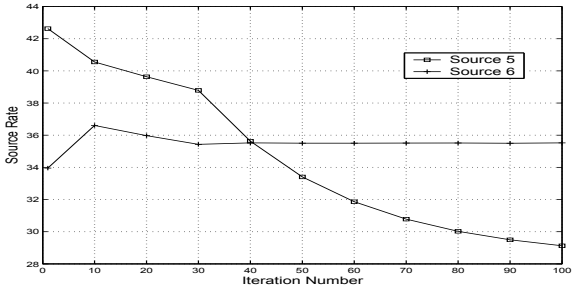

Figure 7: The traces of the particles moving in critical link C.

6. REFERENCES

[1] M. Rabbat, and R. Nowak,"Distributed optimization in sensor networks," *Information Processing in Sensor Networks*, pp.20-27, Apr. 2004.

[2] S.H. Son, M. Chiang, S.R. Kulkarni, and S.C. Schwartz,"Clustering in Distributed Incremental Estimation in Wireless Sensor Networks," *www.princeton.edu/ ∼ chiangm/cluster.pdf*

[3] F.P. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: Shadow prices, proportional fairness and stability," *Journal of the Oper. Res. Soc.*, vol. 49, no. 3, pp. 237-252, Mar. 1998.

[4] J.W. Lee, M. Chiang, and R.A. Calderbank, "Price-based distributed algorithm for optimal rate-reliability tradeoff in network utility maximization," *IEEE Journal of Sel. Areas Commun.*, vol. 24, no. 5, pp. 962-976, May 2006.

[5] S.H. Low, and D.E. Lapsley, "Optimization flow control-I: Basic algorithm and convergence," *IEEE/ACM Trans. Netw.*, vol. 7, no. 6, pp. 861-874, Dec. 1999.

[6] X. Zhu, J.P. Singh, and B. Girod, "Joint routing and rate allocation for multiple video streams in ad-hoc wireless networks," *Journal of Zhejiang University, Science A*, vol. 7, no. 5, pp. 727-736, May 2006.

[7] R. Eberhart, Y. Shi, and J. Kennedy, *Swarm Intelligence*, Morgan Kaufmann, 2001.