PSO and ACO Algorithms Applied to Optimization Resource Allocation to Support QoS Requirements in NGN

Dac-Nhuong Le
Faculty of Information Technology, Haiphong University, Vietnam

ABSTRACT

Next Generation Network (NGN) is the backbone of the overall network architecture based on IP network, supporting different access network technologies. This integrated wireless system will have to handle diverse types of traffics, such as data, voice, and multimedia, etc. NGN will provide advanced services, such as Quality of Service (QoS) guarantees, to users and their applications. In this paper, I have studied a pricing scheme for next generation multiservice networks and formulated the optimal resource allocation in a network/service node, given the QoS requirements of each service class that the network element serves. The non-linear pricing model responds well to changes of the characteristics in the input traffic, pricing parameters and QoS requirements. Furthermore, I proposed two new Particle Swarm Optimization and Ant Colony Optimization algorithms to solve it. Numerical results show that my proposed algorithms are easy and efficient to any number of service classes.

Keyword:
Resource Allocation
Quality of Service
Next Generation Network
Particle Swarm Optimization
Ant Colony Optimization

INTRODUCTION

In Next Generation Network (NGN), the backbone of the overall network architecture will be IP network, supporting different access network technologies such as wireless Local Area Network (WLAN), UMTS Terrestrial Radio Access Network (UTRAN), and WiMax. Moreover, this integrated wireless system will have to handle diverse types of traffics: data traffics (e.g. web browsing, e-mail, ftp), voice traffic (e.g. voIP), and multimedia traffics (e.g. video conferencing, online TV, online games), etc. NGN will provide advanced services, such as Quality of Service (QoS) guarantees, to users and their applications. As a result of these enhancements, it is expected that service providers will face an increasing number of users as well as a wide variety of applications. Under these demanding conditions, network service providers must carefully provision and allocate network resources (e.g. bandwidth, buffer size, CPU capacity) for their customers. Provisioning is the acquisition of large end-to-end network services (connections) over a long time scale. In contrast, allocation is the distribution of these provisioned services (via pricing) to individual users over a smaller time scale [1].

Determining the optimal amounts to provision and allocate remains a difficult problem under realistic conditions. Service providers must balance user needs in the short-term while provisioning connections for the long-term. Furthermore, this must be done in a scalable fashion to meet the growing demand for network services, while also being adaptable to future network technologies. In this paper, I used a Fractional Brownian Motion traffic model, because of its ability to adequately capture characteristics of real network traces, such as self-similarity and the presence of heavy tailed marginal distributions [2].

In [3], Xu Peng et al proposed a measurement-based resource allocation scheme based on a linear pricing model and average queue delay guarantees. This scheme has the disadvantage of not being scalable to
large number of service classes. Moreover, average queue delay is not always an appropriate QoS constraint. The authors in [4] perform maximization over a utility function provided from the network users and resources are shared based on the solution of that optimization problem. In [5], the authors studied the problem of resource allocation with dynamic pricing in which the network administrator controls the price of the resources that users demand based on the demand the prices are dynamically changed over different time periods so as to maximize the revenue of the administrator. Measurement-based resource allocation has also been studied in different contexts in [6][7].

In this paper, I propose two new algorithms based on Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) to optimizing resource allocation to support QoS requirements in NGN. My objective functions are determined by the provider’s profit based on pheromone matrix of ants satisfies capacity constraints to find good approximate solutions. Numerical results show that my proposed algorithms are easy and efficient to any number of service classes.

The rest of this paper is organized as follows. Section 2 presents the problem formulation. Section 3 and 4 present my new algorithms for resource allocation to support QoS requirements in NGN based on PSO and ACO algorithms. Section 5 presents our simulation and analysis results, and finally, section 6 concludes the paper.

2. PROBLEM FORMULATION

The employed modeling framework was introduced in [3].

Figure 1 presents a single network element, which may correspond to either a traditional network component, such as a switch, or a router, or a modern network “service center”, like IBM’s data power service oriented network appliances [8] or Cisco’s application-oriented network message routing systems [9]. It is assumed that the network element serves two categories of traffic classes; deterministic delay-bound classes and flexible delay-bound ones.

The proposed system is responsible for optimally allocating the excess resources to the remaining flexible delay bound classes. These classes enter the Measurement Based Optimal Resource Allocation (MBORA) system proposed in [10] and show in Figure 2.

Figure 2 shows the Measurement Based Optimal Resource Allocation System.
The MBORA system consists of a measurement module, an optimization module and a resource orchestrator module. The statistics of the arrival traffic are measured by the measurement module. It is assumed that the traffic can be accurately approximated by a Fractional Brownian motion model, which can account for the burstiness and long-range dependence observed in real traffic traces. Such a model can be fully described by the following parameters: the Hurst parameter $H$, the mean arrival rate $\bar{\alpha}$ and the variance $\sigma$ of the marginal distribution.

An algorithm for on-line measurement of these parameters is discussed in [11]. The optimization module receives the traffic characteristics of each class and calculates the optimal allocation of resources by solving the optimization problem using PSO and ACO algorithms discussed in Section 3 and 4. It should be noted that the optimization problem is solved only when there is a significant change in traffic characteristics. The optimal solution is fed to the resource orchestrator which dynamically updates the allocation of resources for each traffic class and forwards the packets (or, more generally, the messages, for example XML) toward their destination.

I start by introducing the pricing model, whose solution yields the optimal allocation of resources to the network service node. Suppose that the node can provide $N$ different types of services. The proportions of these services to be allocated are denoted by $s_1, s_2, \ldots, s_N$. According to [12], the profit of a provider is the difference between the revenue $r(s)$ that is obtained for providing these services and the cost $c(s)$ that incurs from producing them.

The aim of this provider is to maximize the profit function subject to the feasibility constraints is defined by:

$$f(s) = \max \{r(s) - c(s)\} = \max \left\{ \sum_{i=1}^{N} (r_i(s_i) - c_i(s_i)) \right\}$$

Subject to the feasibility constraints:

$$\begin{cases} s_i \geq 0, & \forall i = 1..N \\ \sum_{i=1}^{N} s_i \leq 1 \end{cases}$$

The revenue is given by a linear function, while the cost by a nonlinear one. Specifically,

$$r_i(s_i) = p_i s_i, \quad \forall i = 1..N$$

And the cost function is given by:

$$c_i(s_i) = b_i D_i(s_i) e^{\beta_i d_i}, \quad \forall i = 1..N$$

where,

- The coefficient $p_i$ corresponds to the price that the provider charges for the $i^{th}$ service.
- $b_i$ is the amount the provider has to reimburse the users whenever the service level agreement (SLA) [13][14][15][16] are not satisfied. A higher priority class $u$ requires better service than a lower one $v$ and thus it is charged accordingly (i.e., $p_u > p_v$ and $b_u > b_v$).
- $\beta_i$ is the parameter controls the steepness of the cost function.
- $D_i(s_i)$ denotes the value of the performance metric experienced by users of service $i$.
- $d_i$ is the target level under the SLA.

I adopted a linear from the revenue function so as to represent the bandwidth profit (i.e., product of price times bandwidth allocated) that provider would receive. In addition, linearity offers concavity and simplicity which are required characteristics for my optimization problem formulation. On the other hand, my cost function has a nonlinear form shown in Figure 3.
PSO & ACO Algorithms Applied to Optimization Resource Allocation to Support QoS Requirements in NGN (Dac-Nhuong Le)

The exponential shape allows a more severe penalization of the provider (i.e., cost penalty), when services experience larger queue delays than those agreed under SLA. Hence, if \( D_i(s_j) > d_i \), the users are not receiving adequate resources from the provider, which would incur a cost, until the situation is rectified. Figure 3 show the steep increase in the cost observed beyond the desired by the users SLA value of threshold \( d_i \). In this case, \( \beta_i = 10 \) if

\[
\text{cost} = \beta_i (\text{delay})^2
\]

The exponential shape allows a more severe penalization of the provider (i.e., cost penalty), when services experience larger queue delays than those agreed under SLA. Hence, if \( D_i(s_j) > d_i \), the users are not receiving adequate resources from the provider, which would incur a cost, until the situation is rectified. Figure 3 show the steep increase in the cost observed beyond the desired by the users SLA value of threshold \( d_i \). In this case, \( \beta_i = 10 \) if

\[
\text{cost} = \beta_i (\text{delay})^2
\]

The exponential shape allows a more severe penalization of the provider (i.e., cost penalty), when services experience larger queue delays than those agreed under SLA. Hence, if \( D_i(s_j) > d_i \), the users are not receiving adequate resources from the provider, which would incur a cost, until the situation is rectified. Figure 3 show the steep increase in the cost observed beyond the desired by the users SLA value of threshold \( d_i \). In this case, \( \beta_i = 10 \) if

\[
\text{cost} = \beta_i (\text{delay})^2
\]

The exponential shape allows a more severe penalization of the provider (i.e., cost penalty), when services experience larger queue delays than those agreed under SLA. Hence, if \( D_i(s_j) > d_i \), the users are not receiving adequate resources from the provider, which would incur a cost, until the situation is rectified. Figure 3 show the steep increase in the cost observed beyond the desired by the users SLA value of threshold \( d_i \). In this case, \( \beta_i = 10 \) if

\[
\text{cost} = \beta_i (\text{delay})^2
\]

The exponential shape allows a more severe penalization of the provider (i.e., cost penalty), when services experience larger queue delays than those agreed under SLA. Hence, if \( D_i(s_j) > d_i \), the users are not receiving adequate resources from the provider, which would incur a cost, until the situation is rectified. Figure 3 show the steep increase in the cost observed beyond the desired by the users SLA value of threshold \( d_i \). In this case, \( \beta_i = 10 \) if

\[
\text{cost} = \beta_i (\text{delay})^2
\]

The exponential shape allows a more severe penalization of the provider (i.e., cost penalty), when services experience larger queue delays than those agreed under SLA. Hence, if \( D_i(s_j) > d_i \), the users are not receiving adequate resources from the provider, which would incur a cost, until the situation is rectified. Figure 3 show the steep increase in the cost observed beyond the desired by the users SLA value of threshold \( d_i \). In this case, \( \beta_i = 10 \) if

\[
\text{cost} = \beta_i (\text{delay})^2
\]
For simplicity, in my cost function I refer to \( D_{\text{max}}(s_i) \) as \( D_i(s_i) \). I have a SLA violation if at given traffic conditions \( \alpha_i \), \( \sigma_i \) and \( H_i \), the stochastic delay bound, \( D_i(s_i) \), for the agreed QoS, \( \varepsilon_i \) is greater than the desired delay bound \( d_i \). A stricter QoS implies a small value of \( \varepsilon_i \) that generates a larger \( D_i(s_i) \).

Hence, the SLA is more likely to be violated for a given delay threshold \( d_i \) and, therefore, the provider is motivated to allocate more resources to that service class. Putting the revenue and cost components together, the provider’s profit problem becomes:

\[
f = \max_s \left\{ \sum_{i=1}^{N} p_i \times s_i \times C - \sum_{i=1}^{N} d_i \times D_i(s_i) \times e^{\beta(D_i(s_i)-d_i)} \right\}
\]

The cost function to the feasibility constraints previous described, plus the constraints:

\[
s_i > \alpha_i, \ \forall i = 1..N
\]

It should always stand true due to the fact that whenever \( s_i \leq \alpha_i \), I have

\[
Pr\{Q(t) > q_{\text{max}}\} = 1
\]

this implies that we are in an unstable case and the queue would never be able to accommodate the incoming traffic.

3. PSO ALGORITHM FOR THE OPTIMAL ALLOCATION OF RESOURCES

3.1 Particle Swarm Optimization

Particle swarm optimization (PSO) is a stochastic optimization technique developed by Dr. Eberhart and Dr. Kennedy [19][20], inspired by social behaviour of bird flocking or fish schooling. It shares many similarities with other evolutionary computation techniques such as genetic algorithms (GA). The algorithm is initialized with a population of random solutions and searches for optima by updating generations. However, unlike the GA, the PSO algorithm has no evolution operators such as the crossover and the mutation operator.

In the PSO algorithm, the potential solutions, called particles, fly through the problem space by following the current optimum particle. By observing bird flocking or fish schooling, we found that their searching progress has three important properties. First, each particle tries to move away from its neighbours if they are too close. Second, each particle steers towards the average heading of its neighbours. And the third, each particle tries to go towards the average position of its neighbours. Kennedy and Eberhart generalized these properties to be an optimization technique as below.

Consider the optimization problem \( P \). First, we randomly initiate a set of feasible solutions; each of single solution is a “bird” in search space and called “particle”. All of particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles. The particles fly through the problem space by following the current optimum particles. The better solutions are found by updating particle’s position. In iterations, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called \( p\text{best} \). Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called \( g\text{best} \).

When a particle takes part of the population as its topological neighbours, the best value is a local best and is called \( l\text{best} \).

After finding the two best values, the particle updates its velocity and positions with following equation (12) (which use global best \( g\text{best} \)) or (13) (which use local best \( l\text{best} \)) and (14).

\[
v[i] = v[i] + c_1 \times \text{rand}() \times (p\text{best}[i] - \text{present}[i]) + c_2 \times \text{rand}() \times (g\text{best}[i] - \text{present}[i])
\]

\[
v[i] = v[i] + c_1 \times \text{rand}() \times (l\text{best}[i] - \text{present}[i]) + c_2 \times \text{rand}() \times (g\text{best}[i] - \text{present}[i])
\]
In those above equation, \( \text{rand()} \) is a random number between 0 and 1; \( c_1 \) and \( c_2 \) are cognitive parameter and social parameter respectively.

3.2 Solving the optimal allocation of resources based on PSO

In this section, we present application of PSO technique for the optimal allocation of resources problem. My algorithm is described as follows. I consider that configurations in the evolution algorithm are sets of \( N \) different types of services. The Figure 4 presents process of my algorithm to solving the optimal allocation of resources based on PSO.
4. ACO ALGORITHM FOR THE OPTIMAL ALLOCATION OF RESOURCES

4.1 Ant Colony Optimization

The ACO algorithm is originated from ant behaviour in the food searching. When an ant travels through paths, from nest food location, it drops pheromone. According to the pheromone concentration the other ants choose appropriate path. The paths with the greatest pheromone concentration are the shortest ways to the food. The optimization algorithm can be developed from such ant behaviour.

The first ACO algorithm was the Ant System [21], and after then, other implementations of the algorithm have been developed [22][23].

4.2 Solving the optimal allocation of resources based on ACO

In this section, we present application of ACO technique for the optimal allocation of resources problem. My new algorithm is described as follows. I consider that configurations in the evolution algorithm are sets of \( N \) different types of services. The encoding of the ant configuration is by means of real array of length \( N \), say ant \( \{ s_1, s_2, ..., s_N \} \) where \( s_i \) is the proportions of these services to be allocated and \( s_i \) is generated as uniformly distributed random number within the interval \([0, 1]\).

I use fully random initialization in order to initialize the ant population.

In my case the pheromone matrix is generated with matrix elements that represent a location for ant movement, and in the same time it is possible receiver location. We use real encoding to express an element of matrix \( A_{N \times N} \) (where \( N \) is the number of services): \( a_{ij} \) is the profit distance of two providers given by:

\[
a_{ij} = [r_i(s_i) - c_i(s_i)] - [r_j(s_j) - c_i(s_j)]
\]

(15)

Each ant can move to any location according to the transition probability defined by:

\[
p_{ij}^k = \frac{\tau_{ij}^{c_1} \eta_{ij}^{c_2}}{\sum_{l \in N_i^{+}} \tau_{il}^{c_1} \eta_{il}^{c_2}}
\]

(16)

where,

- \( \tau_{ij} \) is the pheromone content of the path from service \( s_i \) to service \( s_j \),
- \( N_i^{+} \) is the neighborhood includes only locations that have not been visited by ant \( k \) when it is at service \( s_i \),
- \( \eta_{ij} \) is the desirability of service \( s_j \), and it depends of optimization goal so it can be my cost function.
- The influence of the pheromone concentration to the probability value is presented by the constant \( c_1 \), while constant \( c_2 \) do the same for the desirability. These constants are determined empirically and our values are \( c_1 = 1, c_2 = 10 \).
The Figure 5 presents process of my algorithm to solving the optimal allocation of resources based on ACO.

![Ant Colony Optimization algorithm’s flow chart](image)

The ants deposit pheromone on the locations they visited according to the relation.

$$\tau_{ij}^{new} = \tau_{ij}^{prev} + \Delta \tau_{ij}$$

(17)

where, $\Delta \tau_{ij}^k$ is the amount of pheromone that ant $k$ exudes to the service $s_j$ when it is going from service $s_i$ to service $s_j$.

This additional amount of pheromone is defined by:

$$\Delta \tau_{ij}^k = \frac{1}{f(s)}$$

(18)

In which, $f(s)$ is maximize the profit function given by

$$f(s) = \max \sum_{i=1}^{N} (r_i(s_i) - c_i(s_i))$$

(19)
The cost function for the ant \( k \) is the provider’s profit given by:

\[
f_k = \max_s \left\{ \sum_{i=1}^{N} p_i s_i C - \sum_{i=1}^{N} d_i D_i(s_i) e^{\beta(D_i(s_i)-d_i)} \right\}
\]  

(20)

The stop condition I used in this paper is defined as the maximum number of interaction \( N_{\text{max}} \) (\( N_{\text{max}} \) is also a designed parameter).

5. EXPERIMENTS AND RESULTS

In my experiments, I have already defined parameters for the PSO and ACO algorithm shown in Table 1, and Table 2.

<table>
<thead>
<tr>
<th>Table 1. The PSO algorithm specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameters</td>
</tr>
<tr>
<td>Population size</td>
</tr>
<tr>
<td>Maximum number of interaction</td>
</tr>
<tr>
<td>Cognitive parameter</td>
</tr>
<tr>
<td>Social parameter</td>
</tr>
<tr>
<td>Update population according to</td>
</tr>
<tr>
<td>Number of neighbor</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2. The ACO algorithm specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameters</td>
</tr>
<tr>
<td>Ant Population size</td>
</tr>
<tr>
<td>Maximum number of interaction</td>
</tr>
<tr>
<td>Parameter</td>
</tr>
</tbody>
</table>

And, my goal is to investigate how the utility function \( f(\bar{x}) \), and the resource allocation vector \( \bar{x} \) respond to changes of various parameters, including the mean arrival rate \( \alpha_i \), the price \( p_i \), the delay threshold \( d_i \) and the Hurst parameter \( H_i \).

I start my analysis with a simple system of two service classes \((s_1, s_2)\). Hence, the corresponding cost function is given by:

\[
f(s_1, s_2) = p_1 s_1 C + p_2 s_2 C - b_1 D_1(s_1) e^{1/(D_1(s_1)-d_1)} - b_2 D_2(s_2) e^{1/(D_2(s_2)-d_2)}
\]  

(21)

where,

\[
D_i(s_i) = \left( s_i - \bar{\alpha}_i \right) \frac{H_i}{1-H_i} \times \frac{1}{s_i} \times \frac{1}{k_i} \times H_i \frac{1}{1-H_i} (1-H_i), i = 1, 2
\]  

(22)

The optimization problem is given by:

\[
f = \max_s \left\{ f(s_1, s_2) \right\}
\]  

(23)

Subject to

- \( s_1 + s_2 = 1 \)
- \( s_1 > \bar{\alpha}_1 \)
- \( s_2 > \bar{\alpha}_2 \)

The parameters of cost function used in my experience are shown in Table 3 below:
Table 3. Parameters of each service class \(i (i=1, 2)\)

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>(p_i (\text{price _unit / Mbps}))</td>
<td>(s_1)</td>
</tr>
<tr>
<td>(h_i (\text{price _unit / ms}))</td>
<td>0.1</td>
</tr>
<tr>
<td>(d_i (\text{delay _unit}))</td>
<td>0.01</td>
</tr>
<tr>
<td>(\text{QoS } (= \varepsilon))</td>
<td>(10^{-6})</td>
</tr>
<tr>
<td>(\sigma_i)</td>
<td>0.01</td>
</tr>
<tr>
<td>(H_i)</td>
<td>0.7</td>
</tr>
</tbody>
</table>

In which, the traffic parameters \(\sigma_i\) and \(\sigma_i\) are normalized to the capacity \(C\). I analysis fifteen problems with three cases:

- The first case, the arrival rate \(\overline{\alpha}_1, \overline{\alpha}_2\) varies while all other parameters are held fixed (see in Table 1). The optimal allocations (normalized to \(C\)) and cost function values are shown in Table 4.

Table 4. Changing the arrival rate \(\overline{\alpha}_1, \overline{\alpha}_2\)

<table>
<thead>
<tr>
<th>Problem</th>
<th>Parameters</th>
<th>PSO</th>
<th>ACO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(\overline{\alpha}_1)</td>
<td>(\overline{\alpha}_2)</td>
<td>(f (s_1, s_2))</td>
</tr>
<tr>
<td>#1</td>
<td>0.2</td>
<td>0.2</td>
<td>(0.5, 0.5)</td>
</tr>
<tr>
<td>#2</td>
<td>0.3</td>
<td>0.2</td>
<td>(0.5462, 0.4538)</td>
</tr>
<tr>
<td>#3</td>
<td>0.4</td>
<td>0.2</td>
<td>(0.5911, 0.4089)</td>
</tr>
<tr>
<td>#4</td>
<td>0.4</td>
<td>0.3</td>
<td>(0.5215, 0.4785)</td>
</tr>
<tr>
<td>#5</td>
<td>0.4</td>
<td>0.5</td>
<td>(0.4532, 0.5467)</td>
</tr>
</tbody>
</table>

- The second case, the delay threshold \(d_1, d_2\) varies while all other parameters are held fixed. The optimal allocations (normalized to \(C\)) and cost function values are shown in Table 5.

Table 5. Changing the delay thresholds \(d_1, d_2\)

<table>
<thead>
<tr>
<th>Problem</th>
<th>Parameters</th>
<th>PSO</th>
<th>ACO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(d_1)</td>
<td>(d_2)</td>
<td>(f (s_1, s_2))</td>
</tr>
<tr>
<td>#6</td>
<td>0.01</td>
<td>0.01</td>
<td>(0.5, 0.5)</td>
</tr>
<tr>
<td>#7</td>
<td>0.01</td>
<td>0.03</td>
<td>(0.5358, 0.4642)</td>
</tr>
<tr>
<td>#8</td>
<td>0.01</td>
<td>0.06</td>
<td>(0.4805, 0.5195)</td>
</tr>
<tr>
<td>#9</td>
<td>0.01</td>
<td>0.09</td>
<td>(0.4761, 0.5239)</td>
</tr>
<tr>
<td>#10</td>
<td>0.01</td>
<td>0.12</td>
<td>(0.4501, 0.5499)</td>
</tr>
</tbody>
</table>

- The final case, the coefficient \(p_1, p_2\) varies while all other parameters are held fixed. The optimal allocations (normalized to \(C\)) and cost function values are shown in Table 6.

Table 6. Changing the pricing factors \(p_1, p_2\)

<table>
<thead>
<tr>
<th>Problem</th>
<th>Parameters</th>
<th>PSO</th>
<th>ACO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(p_1)</td>
<td>(p_2)</td>
<td>(f (s_1, s_2))</td>
</tr>
<tr>
<td>#11</td>
<td>1</td>
<td>1</td>
<td>(0.5, 0.5)</td>
</tr>
<tr>
<td>#12</td>
<td>2</td>
<td>1</td>
<td>(0.6951, 0.3049)</td>
</tr>
<tr>
<td>#13</td>
<td>3</td>
<td>2</td>
<td>(0.6527, 0.3473)</td>
</tr>
<tr>
<td>#14</td>
<td>4</td>
<td>1</td>
<td>(0.7298, 0.2702)</td>
</tr>
</tbody>
</table>
Figure 6 show that when the arrival rate varies the optimal solution can be observed. In the equal arrival rates case the resource are equally shared (if a class bring more traffic load, then it is assigned a large portion of resources). Moreover, the system becomes more stressed when the overall profit of the provider decreases substantially.

Figure 7 show the sensitivity of my model with respect to the delay threshold $d_i$. The threshold increases, the profit of the provider also increases which is due to the fact that

$$\frac{\partial f}{\partial d_i} = b_i \beta_i D_i e^{\beta_i (0, -d_i)} > 0$$

(24)

And, it is also worth notice that, the class with stricter QoS requirements is allocated more resources.

Figure 6. Sensitivity to mean arrival rate

Figure 7. Sensitivity to mean delay thresholds

Figure 8 shows the sensitivity of my model with respect to the price coefficient. If equal prices we obtain equal allocation, while the allocation of resources exhibits a strong sensitivity to the price ratio $p_1/p_2$.

Finally is Figure 9 compare cost function between PSO and ACO algorithms.

Figure 8. Sensitivity to mean price coefficient

Figure 9. Comparision cost function between PSO and ACO algorithms

6. CONCLUSIONS

In this paper, I have studied a pricing scheme for next generation multiservice networks and formulated the optimal resource allocation in a network/service node, given the QoS requirements of each service class that the network element serves. The non-linear pricing model responds well to changes of the characteristics in the input traffic, pricing parameters and QoS requirements. I propose two new Particle Swarm Optimization and Ant Colony Optimization algorithms to solving it. Numerical results show that my proposed algorithms are easily and efficiently to any number of service classes. Additionally, it is sensitive to traffic changes and responds well to pricing parameters and QoS requirements. The optimal available resources allocations dynamically in a network of multiple service intermediaries and multiple types of resources will be my next research goals.

REFERENCES


PSO & ACO Algorithms Applied to Optimization Resource Allocation to Support QoS Requirements in NGN (Dac-Nhuong Le)


**BIOGRAPHY OF AUTHOR**

**Dac-Nhuong Le**

He received the BSc degree in computer science and the MSc degree in information technology from College of Technology, Vietnam National University, Vietnam, in 2005 and 2009, respectively. He is a lecturer at the Faculty of information technology in Haiphong University, Vietnam. He is currently a Ph.D student at Hanoi University of Science, Vietnam National University. His research interests include algorithm theory, computer network and networks security.