ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 3, No. 1, 2008, *pp*. 21-38

Stochastic Optimization of Distributed Sampling System Placement for Passive Measurement *

Chengchen Hu [1 +], Bin Liu [1], Zhen Liu [1], Shifang Gao [2] and Dapeng Oliver Wu [3]

[1] Dept. of Computer Science and Technology, Tsinghua University, Beijing, China
[2] Patent Examination Cooperation Center of SIPO, Beijing, China
[3] Dept. of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA

(*Received April 1 2007, accepted December 20 2007*)

Abstract. Flow-level traffic measurement is important for network traffic accounting, traffic engineering, and network security. However, flow-level measurement in high speed networks poses great challenges due to the requirements of high packet processing speed and large memory size (high time/space complexity). To reduce these demanding requirements, sampling is usually used and samplers are deployed in the network. But sampling incurs information loss. To address this issue, this paper studies the tradeoff between sampling loss and complexity in distributed sampling system. We formulate the distributed sampling problem as a constrained optimization problem; specifically, maximizing the measurement coverage (i.e., the percentage of sampled traffic among the total traffic) and minimizing the complexity/budget. Considering the stochastic nature of traffic flows, we further formulate the optimization problem under two stochastic criteria: Stochastic Expected Value Optimization criterion (which is concerned with average performance) and Stochastic Chance Constrained Optimization criterion (which is concerned with the distribution of performance measure). Then we propose a Hybrid Intelligent algorithm to decide the optimal deployment strategy for monitors' placement and the sampling rate at each monitor. Equipped with the proposed algorithm, we are able to address the optimal tradeoff between measurement coverage and deployment cost for networks with random traffic, which has not been studied before. The extensive simulations and experiments demonstrate the effectiveness of our models and algorithm: with careful deployment, monitoring over a small fraction of nodes in a high speed network is sufficient to maintain a high level of measurement coverage.

Keywords: Distributed Sampling System; Stochastic Expected Value Optimization; Stochastic Chance Constrained Optimization.

1. Introduction

Flow-level traffic measurement, which is important for network accounting, traffic engineering, network security, network diagnosis and more applications, can be classified into passive measurement and active measurement [1]. Passive measurement techniques (passively) monitor traffic and infer the network status by analyzing packets passing through the traffic monitors. As opposed to passive measurement, active measurement techniques directly probe network properties by generating the traffic needed to make the measurement (e.g., available bandwidth, packet loss ratio, delay). In this paper, we focus on the efficient approach to passive measurement.

With the continuous increasing of the line speed and the number of flows, per-flow passive measurement on a high speed network link faces a challenge on the demanding requirement for the packet processing speed and memory size within a monitor. In the literature, many novel sampling techniques were proposed for individual monitors to use limited resources for best-effort estimation of traffic statistics [2-6]. Sampling provides measurement scalability but incurs inherent loss of information meanwhile. Many detailed traffic

* This work is supported by 973 plan (2007CB310702), NSFC (60573121, 60625201), Cultivation Fund of the Key Scientific and Technical Innovation Project (705003) , Specialized Research Fund for the Doctoral Program of Higher Education of China (20040003048 and 20060003058) , Science and Technology Collaboration Research Fund (2006DFA11170), and Tsinghua Basic Research Foundation (JCpy2005054)

+ Corresponding author. Tel.: +86-10-62773441; fax: +86-10-62771138.
E-mail address: huc@ieee.org.

characteristics (i.e., the distribution of flow size) are difficult to be recovered from the sampled traffic [5, 7]. However, if we can develop a scalable sampling approach to collect traffic with tiny loss (say, less than 10% of the total packets), the detailed and accurate flow statistics information can be obtained in an easy way.

As a flow can be observed at any node on its routing path, we can place a number of monitors along the routing path so that they collaboratively monitor the same flow. The sampling rates on the collaborating monitors can be much lower than that on a single monitor, to achieve the same measurement coverage (monitored fraction of the flow) probabilistically. Following this idea, the *Distributed Sampling System (DSS)* is proposed [8-10], where several sampling monitors scatter over the network and work collaboratively. In this way, well deployed DSS has two nice features: 1) high scalability. The monitoring speed and monitored flow number can be reduced in each monitor by sampling, thereby reducing the packet processing rate and the memory size; 2) tiny loss. The sampling loss can be greatly amortized by configuring several monitors on the routing path of flows.

If the routing path has only a single hop, the benefit of DSS will be limited. Fortunately, such situation is rare. We do not consider the situation with single hop routing path and investigate the deployment strategy of DSS in this paper. The deployment strategy determines how to optimally deploy traffic monitors, which refers to the locations and the sampling rates of monitors. Intuitively, the more monitors (and/or the faster sampling rate), the larger fraction of traffic can be measured; however, placing a monitor incurs a certain amount of deployment cost, and configuring a faster monitor requires a higher operating cost. [8, 9] tried to strike the right tradeoff between the measurement coverage and the cost when considering the deployment strategy, but the main limitation in their work is that they did not take into account the randomness (fluctuation) of realistic traffic. Our experiments illustrate that, though the deployment strategy under the assumption of constant flow rate may obtain satisfied average measurement coverage, it can not guarantee the measurement coverage all the time. In fact, in our experiments with different flow rate distributions, the measurement coverage is less than the average measurement coverage in more than 40% of time. In some worse cases (flow rate is with large variability), the coverage can drop to 60% in some measurement intervals even when the object coverage is set to 80%.

In this paper, we treat flow rates as stochastic processes and introduce the concept of stochastic optimization [11] to formulize the DSS deployment problem. Specifically, the following problems are explored:

- How many monitors will be deployed?
- Where will these monitors be located?
- How fast will these monitors sample packets?
- How much traffic can be monitored by DSS?

The rest of the paper is organized as follows. The related work is presented in Section 2. In Section 3, we formulate the tradeoff in DSS deployment problem as a *Stochastic Expected Value Optimization (SEVO)* problem and a Stochastic *Chance Constrained Optimization (SCCO)* problem. Section 4 presents our *Hybrid Intelligent (HI)* algorithm to solve the problems. Experiments are conducted to evaluate the proposed algorithm in Section 5 and some discussions are given in Section 6. Finally we conclude the paper in Section 7.

2. Relate Work

2.1. Sampling

Recently many efforts have made to address the sampling problem in the context of traffic measurement and analysis. A pioneering work on statistical sampling of network traffic was published in [12], which used sampling mechanism in NSFNET backbone to estimate the distribution of packet size. Nowadays, the primary flow-level measurement tool by network operators is NetFlow [13]. It resorts to packet sampling, known as sampled NetFlow [3], to handle the volume and traffic diversity in high speed links. Since setting the sampling rate of NetFlow to reach a good balance between accuracy and resource consumption is not an easy job, adaptive sampling methods [4-6] have been proposed to adaptively tune the sampling rate according to the estimation accuracy [4], or traffic flow size [5], or memory consumption [6]. In [14], a different kind of packet sampling scheme has been proposed in order to better capture the statistics of large flows. Based on the aggregate flow properties (but not individual flow properties), the mean flow length was inferred in [7] and the distribution of flow length was estimated in [5, 15], from the sampled traffic.

These studies focused on sampling in an individual monitor. Since this sampling methodology suffers

heavy loss of traffic information especially in high speed network links, in general, it is difficult to estimate accurate and detailed characteristics of original traffic from sampled traffic, especially for individual flow properties.

2.2. Distributed Measurement

Prior distributed measurement systems, as known as IPMON [16], NLANR [17], NProbe [18] and etc., were designed to collect traffic on an on-demand basis. Most of the ISPs or network operators currently equip all the nodes with sampling tools (e.g. NetFlow) using the same sampling rate for limited measurement purposes (not scalable).

Recently, [8, 9] addressed the placement problem of monitors by locating the monitors' positions and configuring the monitors sampling rate. However, they only considered the static placement of monitors but neglected the varieties in flow rates of actual Internet traffic. These solutions are not optimal or effective when used in the realistic stochastic Internet. Our previous-work [19] was initial on determine the deployment of distributed passive measurement system considering the stochastic feature of flow rates.

2.3. Placement Problems

A large body of literatures studied the different deployment/placement problems. In [20], the authors concentrated on the problem of optimizing a scalable distributed SNMP-based polling system. The placement problem of object replicas was studied in [21] to meet QoS requirements of clients with the objective of minimizing the replication cost. The placement problem of mirror servers was investigated in [22]. The placement of Internet-wide Infrastructure used to collect distance information was discussed in [23]. In [8, 9], the authors studied the placement of passive monitors to maximize the measurement coverage, however, they suppose the flow rate to be a constant, which is not the case in real Internet.

The methodologies used in these earlier works to model different placement problems, can be recognized as the "deterministic optimization". They assigned certain values to parameters in optimization models, which can be solved by traditional graph theory or optimization methods. Such deterministic optimizations are not valid in practice for the DSS placement problem since realistic traffic flows tend to be random as we have mentioned above.

3. Problem Formulation

We address the fluctuation of flow rates on two time scales. 1) On the large time-scale, it has been known for a long time that the Internet traffic in particular exhibits strong daily and weekly patterns [24], and the behavior has not changed over the years [25]; 2) On the small time-scale, the flow rates vary over time for the reasons of user behaviors, temporary link failures, anomalies and etc.

The traffic in daytime is about ten times more than the one at night, and the traffic in workdays is also about ten times more than the one at the weekends. The time-of-day effect in link loads variability on large time-scale is of very large magnitude but very slow changing. To handle the fluctuation on the large time-scale, we can easily define three different deployment strategies for daytime in workdays, nights in workdays and weekends respectively. To achieve each deployment strategy, we should look into the flow rate variety on the small time-scale. Compared with that on the large time-scale, the feature of the varieties on small time-scale is of small magnitude but fast changing. Thus we consider the flow rates as stochastic processes and construct two categories of stochastic optimization model to deal with the fluctuation on the small time scale in this section.

Table I summarizes the main notations used in the paper. The network can be modeled as a graph $G(V,E)$, where V is the set of nodes and $E \subseteq V \times V$ is the set of links. A flow is defined as an *Origin-Destination (OD)* flow, in which the traffic consists of IP-level flows that enter the network at a given ingress POP and exit an another egress POP. Let us assume that: 1) traffic of each flow is fluid; 2) the traffic of flow *j* is a rate process $f_j(t)$, i.e., the instantaneous rate at time *t* is a random variable $f_j(t)$; 3) $f_j(t)$ is a stationary and ergodic process[1]; 4) The flows are assumed to be independent. Since we are interested in performance of the network monitors in the steady state, we can remove the time index *t* and simply replace $f_j(t)$ by f_j.

[1] For a stationary process, the time index *t* can be removed since its expectation is the same for all *t*; and for an ergodic process, the time average of the stochastic process is equal to its expectation.

TABLE I. NOTATIONS DESCRIPTION

NOTATIONS	DESCRIPTION
S	SET OF ALL FLOWS
R_J	MONITORED FRACTION OF FLOW J IN DSS
R_{IJ}	MONITORED FRACTION OF FLOW J IN NODE I
Y_{IJ}	Y_{IJ}=1 IF NODE I IS ON THE ROUTING PATH OF FLOW J, ELSE Y_{IJ}=0
C_I	COST OF PLACING A MONITOR IN NODE I, $i \in V$
Q_I	SAMPLING RATE IN THE NODE I, $i \in V$
$\vec{\mathbf{q}}$	DECISION VECTOR OF Q_I, $\vec{\mathbf{q}} = \{q_i\}, i \in V$
X_I	X_I=1 IF A MONITOR IS PLACED IN NODE I, ELSE X_I=0
$\vec{\mathbf{x}}$	DECISION VECTOR OF X_I, $\vec{\mathbf{x}} = \{x_i\}, i \in V$
F_J	TRAFFIC RATE OF FLOW J (STOCHASTIC PARAMETER)
$\mathbf{f}$	STOCHASTIC VECTOR OF F_J, $\vec{\mathbf{f}} = \{f_j\}, j \in S$
T	PREDEFINED COVERAGE THRESHOLD
L	MAXIMUM ACCEPTABLE MONITOR SAMPLING RATE
B	TOTAL BUDGET
N	NUMBER OF FLOWS IN THE NETWORK

We consider the node-based monitor that is implemented inside a node, so every link connected to the monitor will be monitored. The monitor i samples traffic at a rate of q_i Mb/s for all the ingress links, which should be lower than a maximum acceptable value L. The sampling rates can be configured independently in different monitors. Note that, our definition of sampling rate is different from the general ones in the form of some ratio (say, sample one packet from N packets). The general one is not so realistic or is not so efficient, since the monitors still have to work at the line rate to count the number of packets and then to decide whether to sample the packet. By the general definition, it will reduce the required memory, but will not lower the operation rate indeed. On the contrary, in this paper, monitors works at a lower rate q_i than the link capacity and use a small memory.

The cost of deploying a monitor should consist of two parts. One is a fixed component such as an expense on maintenance. The other is the hardware and software cost, which is typically relevant to a non-decreasing convex function[2] of its sampling rate. More specifically, we define the cost as

$$c_i = D(q_i) = C + \alpha(q_i / L)^m, \ m > 1. \tag{1}$$

where C is the unchanging part of the cost and α , m are constant parameters. The total cost should be the sum of the deployment cost of all the monitors, i.e.,

$$\sum_{i \in V} c_i x_i \ . \tag{2}$$

Note that, the monitor i samples packets at a rate of q_i Mb/s over all the flows pass through the same link, not over each flow. If monitor i samples over each flow with same sampling rate, a table needs to be maintained for every flow, and it is a heavy overhead in a backbone link, which all the related sampling techniques tried to eliminate. We use S_{ij} to denote the set of flows sharing the same ingress link in node i with flow j, and $S_{ij} \subseteq S$. Flow j can be monitored in node i only if node i is on the routing path of flow j (y_{ij}=1) and a monitor is placed in node i (x_i=1). It is straightforward that if the sampling rate of node i is faster than the aggregate rates of flows sharing the same link, r_{ij} should be one. Otherwise, r_{ij} should be the ratio of sampling rate to the aggregate flow rates in the same link. Accordingly, the monitored fraction of flow j in node i can be achieved by (3).

$$r_{ij} = \begin{cases} \dfrac{q_i x_i y_{ij}}{\sum_{k \in S_{ij}} f_j}, & \text{if } \dfrac{q_i x_i y_{ij}}{\sum_{k \in S_{ij}} f_j} \le 1 \ \text{ for } i \in V, j \in S, \\ 1, & \text{if } \dfrac{q_i x_i y_{ij}}{\sum_{k \in S_{ij}} f_j} > 1 \ \text{ for } i \in V, j \in S. \end{cases} \tag{3}$$

We assume the sampling of each flow j in different monitors to be independent. Consequently, the monitored fraction of flow j (R_j) can be represented by

[2] Define function $f : I \to R$ is a convex function on I, if $f(\lambda x_1 + (1-\lambda)x_2) \le \lambda f(x_1) + (1-\lambda) f(x_2)$, $\forall x_1, x_2 \in I, \lambda \in [0,1]$.

$$R_j = 1 - \prod_{i \in V} (1 - r_{ij}), j \in S \,. \tag{4}$$

The measurement coverage can be defined as the mean of monitored fraction of all flows. More complex definition form of measurement coverage can be applied, e.g., using weighted average of monitored fraction of all flows. The definition will not affect the solution a bit, and we choose a simple form as illustrated in (5). Note that, we also use *M* to represent $M(\vec{\mathbf{x}}, \vec{\mathbf{q}}, \vec{\mathbf{f}})$ for short in this paper.

$$M(\vec{\mathbf{x}}, \vec{\mathbf{q}}, \vec{\mathbf{f}}) = \frac{1}{n}[\sum_j R_j] \,. \tag{5}$$

Since the flow rates fluctuate randomly in small time-scale, we construct the *Stochastic Expected Value Optimization (SEVO) criterion* and the *Stochastic Chance Constrained Optimization (SCCO) criterion* to deal with the stochastic feature of Internet traffic.

3.1. Stochastic Expected Value Optimization criterion

Since $\vec{\mathbf{f}}$ is a stochastic vector of flow rate over time in practice, the measurement coverage $M(\vec{\mathbf{x}}, \vec{\mathbf{q}}, \vec{\mathbf{f}})$ is a stochastic variable. We can not simply predict what the measurement coverage will be under the decision vectors $\vec{\mathbf{x}}$ and $\vec{\mathbf{q}}$. A natural idea is to use the average measurement coverage to rank different decisions, shown as follows,

$$E[M] = \frac{1}{n} E[\sum_j R_j] \,. \tag{6}$$

where *E* denotes the expected value operator.

As mentioned in Section I, maximizing the measurement coverage and minimizing the deployment cost are two conflicting objectives. Thereby, we define two measurement problems to study the tradeoff according to the requirements from different angles. From the viewpoint of customers, the measurement coverage is what they concern about most. We introduce the *Coverage Constraint Problem (CCP)*, aiming at minimizing the cost with certain measurement coverage constraints. For ISPs' considerations, the budget is always an important concern. Thus, we should determine the deployment strategy by maximizing the measurement coverage with a budget constraint, which refers to *Budget Constraint Problem (BCP)*.

We can utilize the following SEVO criterion to formulize *Coverage Constraint Problem (CCP)*. It is related to the decision of the optimal value of $\vec{\mathbf{x}}$ and $\vec{\mathbf{q}}$, such that the budget achieves the minimal value under the expected measurement coverage constraint.

SEVO criterion for CCP

$$\text{Min } \sum_{i \in V} c_i x_i$$

Subject to:

$$E[M] \geq T \,. \tag{7}$$

Similarly, the *Budget Constraint Problem (BCP)* can be formulized as follows, whose objective is to maximize the coverage with the constraint of the total budget.

SEVO criterion for BCP

$$\text{Max } E[M]$$

Subject to:

$$\sum_{i \in V} c_i x_i \leq B \,. \tag{8}$$

3.2. Stochastic Chance Constrained Optimization Model

Building SEVO criterion is a simple but efficient way to deal with the stochastic nature of flow rates. However, we are not always concerned with the expected value. In fact, sometimes we have to consider the reliability, referring to as the probability that some favorable events (for example, maximizing the coverage or minimizing the cost) will occur. In SEVO, a given optimal solution is achieved regardless of whether it can be performed in practice and may be only possible to perform with small probability. Here is an example. Give two DSS deployment decisions. One can measure 100% of the traffic at 90% of the time but miss all the traffic during other 10% of the time. The other one can cover 85% of the traffic during the whole

measurement time. From SEVO criterion, it is no doubt to select the first decision; however, the second one should be preferred in many applications (e.g., accounting applications and security applications.) Due to the inability of SEVO in such cases, we introduce the SCCO criterion.

As opposed to the SEVO criterion, SCCO criterion gives probabilistic bound by adding stochastic constraints to measurement coverage and holds it at a confidence level of β.

Compared with SEVO criterion for CCP, we simply add a stochastic constraint to SCCO criterion for CCP. The probability of $\{M \geq T\}$ should be at least β, where T and β are predefined measurement coverage threshold and confidence level.

SCCO criterion for CCP

$$\text{Min } \sum_{i \in V} c_i x_i$$

Subject to:

$$\Pr\{M \geq T\} \geq \beta . \tag{9}$$

To build the SCCO criterion for BCP, we alternatively introduce optimistic values as defined in the following.

DEFINITION: Let ξ be a stochastic variable and $\beta \in (0,1]$. Then $\xi_{\sup}(\beta) = \sup\{r \mid \Pr\{\xi \geq r\} \geq \beta\}$ is called the β-optimistic value of ξ.

We construct a SCCO criterion for BCP to maximize the measurement coverage as shown below.

SCCO criterion for BCP

$$\text{Max } M_o$$

Subject to:

$$M_o = \sup\{r \mid \Pr\{M \geq r\} \geq \beta\} . \tag{10}$$

$$\sum_{i \in V} c_i x_i \leq B . \tag{11}$$

M_o is the β-optimistic value of M and β is the predetermined confidence level in (10). It represents in (10) the optimistic value of the measurement coverage: the maximal value of r that can meet the stochastic constraint $\Pr\{M \geq r\} \geq \beta$. The model is to find an optimal solution of $\bar{\mathbf{x}}$ and $\bar{\mathbf{q}}$ that maximizes M_o under the budget constraint.

SCCO criterion for BCP is equivalent to the following Maximax model.

Maximax model equivalent to SCCO for BCP

$$\max_{\bar{\mathbf{x}}, \bar{\mathbf{q}}} \max_{M_o} M_o$$

Subject to:

$$\Pr\{M \geq M_o\} \geq \beta \tag{12}$$

$$\sum_{i \in V} c_i x_i \leq B \tag{13}$$

3.3. Which Model to be Adopted

In this Section, we have built two categories of stochastic optimization models. Which one should be adopted? It depends on the applications and ISPs/users.

SCCO criterion gives a probability bound or guarantee but the stronger constraints in it than in SEVO criterion, requires more cost. SEVO criterion purchases the average measurement coverage and is sufficient to the applications where only the average performance is required (eg. drawing the traffic matrix). Adopting SEVO criterion can also save the deployment cost. If the applications are of sensitive (eg. security application) or the ISPs have enough budgets, the SCCO criterion is preferred. Otherwise, the SEVO criterion is sufficient.

Note that, the results achieved by these two kinds of models can be affected by the variance of flow rates.

If the flow rates are of little variability, the differences between the results obtained by these two models are small; otherwise, the differences are large. The experiments in Section 5 demonstrate the fact clearly.

4. Hybrid Intelligent Algorithm

The models proposed in section III are different from the classical optimization problems because of the existence of uncertain stochastic functions ((6), (9) and (10)). If the uncertain functions are determined, there would be no essential difference between stochastic optimization models and classical optimization models. Therefore, the crux lies in how to determine the uncertain functions. For this reason, we propose a *Hybrid Intelligent (HI)* algorithm to tackle the problem, which is composed of two parts — an uncertain function approximation part and an optimization method part. The former one is utilized to determine the uncertain functions and the latter one is employed to find the optimization solution after the uncertain functions are approximated.

4.1. Uncertain Function Approximation

Approximation method for expected value function

From the definition of stochastic expected value, there exist multiple integrals need to be calculated. In the traditional method of multiple integral by iteration, It depends on the number of dimensions that the number of sampling points are required to obtain a given degree of accuracy [11]. We use the following approximation method to approximate the expected value function, whose accuracy is independent of the dimensions.

Known form the *Strong Low of Large Number (SLLN)*, we have $\frac{1}{N}\sum_{k=1}^{N} I(\vec{\mathbf{f}}_k) \to E[I(\vec{\mathbf{f}}_k)]$ with probability 1, as $n \to \infty$. Therefore, if we generate N random vectors $\vec{\mathbf{f}}_k (k = 1,2,\cdots,N)$ according to the flow rate distribution function, then the expected value is estimated by $\frac{1}{N}\sum_{k=1}^{N} f(\vec{\mathbf{f}}_k)$ provided that N is sufficiently large. The detailed approximation process of the expected value is shown as follows.

Step 1: initialization. Set $E = 0$, $k = 1$;
Step 2: generate $\vec{\mathbf{f}}_k$ randomly according to the flow rate distribution function;
Step 3: $E = E + M(\vec{\mathbf{f}}_k)$;
Step 4: $k = k + 1$;
Step 5: return to step 2 if $k \le N$;
Step 6: $E = E / N$.

Approximation method for $\Pr\{M(\vec{\mathbf{f}}_k) \ge T\}$

To approximate the uncertain function $\Pr\{M(\vec{\mathbf{f}}_k) \ge T\}$ in stochastic constraint (9), we first produce N random vectors $\vec{\mathbf{f}}_k$, $k = 1,2,\cdots,N$ and then use N' to count the occasions when inequality $M(\vec{\mathbf{f}}_k) \ge T$, $k = 1,2,\cdots,N$ is satisfied.

We first define an indicative function,

$$I(\vec{\mathbf{f}}_k) = \begin{cases} 1, & \text{if } M(\vec{\mathbf{f}}_k) \ge T, \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

Thus, we have

$$E[I(\vec{\mathbf{f}}_k)] = \Pr\{M(\vec{\mathbf{f}}_k) \ge T\} . \tag{15}$$

$$N' = \sum_{k=1}^{N} I(\vec{\mathbf{f}}_k) . \tag{16}$$

Since it follows SLLN, the following equation will hold true if N is sufficiently large.

$$\sum_{k=1}^{N} I(\vec{\mathbf{f}}_k) / N = E[I(\vec{\mathbf{f}}_k)] . \tag{17}$$

Substituting (15) and (16) into (17), we obtain

$$\Pr\{M(\vec{\mathbf{f}}_k) \geq T\} = \frac{N^{'}}{N}. \tag{18}$$

The approximation algorithm can be summarized as follows.

Step 1: initialization. Set $N' = 0$, $k = 1$;
Step 2: generate $\vec{\mathbf{f}}_k$ randomly according to the flow rate distribution function;
Step 3: if $M(\vec{\mathbf{f}}_k) \geq T$, $N' = N' + 1$;
Step 4: $k = k + 1$;
Step 5: return to step 2 if $k \leq N$;
Step 6: $N^{'} / N$.

Approximation method for β-optimistic value

The last uncertain function existing in our models is the optimistic value function. In order to approximate the optimistic value, we should find the maximal value M_o such that $\Pr\{M \geq M_o\} \geq \beta$. It is obvious that, $M_o \uparrow \rightarrow \Pr\{M \geq M_o\} \downarrow$. Consequently, the maximal value of M_o is achieved at the equality case $\Pr\{M \geq M_o\} = \beta$. In other words, the problem is changed to find an M_o and let $\Pr\{M \geq M_o\} = \beta$.

First, we define an indicative function as

$$I(\vec{\mathbf{f}}_k) = \begin{cases} 1, & if\ M \geq M_o, \\ 0, & otherwise. \end{cases} \tag{19}$$

And define N' as the number of the occasions that inequality $M(\vec{\mathbf{f}}_k) \geq Mo$, $k = 1,2,\cdots,N$ holds true. Again, from the SLLN, we have $\frac{1}{N}\sum_{k=1}^{N} I(\vec{\mathbf{f}}_k) = E[I(\vec{\mathbf{f}}_k)]$ if N is sufficiently large.

Note that $N' = \sum_{k=1}^{N} I(\vec{\mathbf{f}}_k)$ and $\Pr\{M \geq M_o\} = \beta$, thus,

TABLE II. Details results obtained by SEVO criterion

Scenario	Mean	Max	Min	Less than mean
One	0.8134	0.8516	0.7261	0.4682
One	0.9144	0.9341	0.8337	0.4308
Two	0.8004	0.8016	0.7917	0.4673
Two	0.9084	0.9108	0.9059	0.4539
Four	0.7987	0.8017	0.7962	0.4664
Four	0.9037	0.9056	0.9019	0.4220

TABLE III. Number of monitors in different topologies

No. of nodes	No. of monitors	Measurement Coverage
14	6	0.8
14	8	0.9
30	9	0.8
30	14	0.9
40	10	0.8
40	14	0.9
50	12	0.8
50	15	0.9

$$N' / N = E[I(\vec{\mathbf{f}}_k)] = \beta, \tag{20}$$

provided that N is sufficiently large. As N' should be an integer, $N' \approx \lfloor \beta N \rfloor$[3]. In such case, among N randomly

[3] $\lfloor \beta N \rfloor$ is the floor function of βN, which rounds βN to the nearest integer towards minus infinity.

generated vectors $\vec{\mathbf{f}}_k (k = 1,2,\cdots,N)$, the inequality $M(\vec{\mathbf{f}}_k) \geq M_o$ should hold true for N' times. In other words, the value of M_o can be taken as the $N'th$ largest element in the sequence $\{M(\vec{\mathbf{f}}_1), M(\vec{\mathbf{f}}_2), \cdots, M(\vec{\mathbf{f}}_N)\}$.

Following is the approximation algorithm for β-optimistic value.

Step 1: initialization. Set $N' = \lfloor \beta N \rfloor$, $k = 1$;

Step 2: generate $\vec{\mathbf{f}}_k$ randomly according to the distribution function;

Step 3: calculate and store the value of $M(\vec{\mathbf{f}}_k)$;

Step 4: $k = k + 1$;

Step 5: return to step 2 if $k \leq N$;

Step 6: find the $N'th$ largest element in the sequence of $\{M(\vec{\mathbf{f}}_1), M(\vec{\mathbf{f}}_2), \cdots, M(\vec{\mathbf{f}}_N)\}$.

These three algorithms are sufficient to approximate the uncertain functions; however, approximation methods are required whenever the uncertain functions are invoked. It makes the process time consuming. In order to optimize the approximation process, we introduce the *Artificial Neural Networks (ANN)* [26] to speed up the approximation. ANN has the ability to approximate the uncertain functions and to run at high speed of operation after being trained. Our motivation to introduce ANN is to train it once, and use it many times. In addition, another advantage of the utilizing of ANN is that it has the ability to compensate for the error of training data obtained from the approximations which are not very precise.

For each uncertain function, we first employ the corresponding approximation method to produce sufficient groups (3500 groups in our experiments) of training data to train an ANN, then we can use trained ANN to approximate the uncertain function.

4.2. Optimization Method

After the uncertain functions are determined, the aforementioned problems can be shown to be NP-hard by a straightforward reduction from the budgeted maximum coverage problem, which is defined in [27]. Thus, we should utilize some heuristic algorithm to solve the problem.

Genetic Algorithm (GA) is an effective meta-heuristic optimization algorithm inspired by natural selection and population genetics [28, 29]. It is selected as the basic optimization method of our HI algorithm due to 1) It is efficient to find global optimal solution for a wide range of objective functions. Most of the classical optimization methods, e.g. direct methods, gradient methods and Hessian methods, are closely dependent upon the objective functions and may stop searching at a local optimal solution. 2)Some heuristic neighborhood search methods, like Tabu search, Simulated Annealing, are efficient in finding global optimal solutions. However, in DSS placement problems, it is hard to code the decision variables (two kinds of numbers: integer and real) and to define an efficient search neighbor. 3) The convergence of canonical GA (we adopt canonical GA in this paper) has been well proved by convergence-schema theorem [30] and Markov Chain analysis [31] (A comprehensive survey can be found in [29]). 4) We can use one general algorithm to solve all the models formulized above instead of developing one for each.

Instead of a single sample from the solution space, GA maintains a population of vectors. At very beginning, the initialization operation generates a number of solution candidates (chromosomes) randomly in the solution space. In each iteration, GA uses evaluation and selection operation to find the fittest chromosomes. These selected out chromosomes are utilized to generate new search chromosomes for the next iteration (generation) by crossover and mutation operations. This is repeated until the algorithm converges or we run out of the predefined iteration times. The solution produced by GA is the most beneficial chromosome in the last iteration. Fig. 1 lists the flowchart of GA.

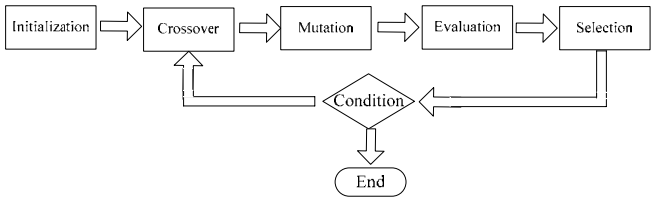

Fig. 1. Genetic Algorithm

In order to utilize GA, we should first specifically code the chromosomes and design the operations for the problem.

A 2*l*-dimension vector $V=(x_1,x_2,\cdots,x_l,q_1,q_2,\cdots,q_l)$ is used as a chromosome to represent a candidate decision vector, where *l* is the number of nodes in the topology. Note that, $q_i=0$ if $x_i=0$.

Let us describe the crossover operation on chromosome pair (V_1, V_2) ,which is denoted as

$$V_1=(x_1^1,x_2^1,\cdots,x_l^1,q_1^1,q_2^1,\cdots,q_l^1),\ V_2=(x_1^2,x_2^2,\cdots,x_l^2,q_1^2,q_2^2,\cdots,q_l^2).$$

One pair of crossover position (n_1, n_2) is randomly generated such that $1\le n_1<n_2\le l$. The genes between n_1 and n_2, $l+n_1$ and $l+n_2$, are swapped, thus the following two children are produced.

$$V_1^{'}=(x_1^1,\cdots,x_{n_1-1}^1,x_{n_1}^2,\cdots,x_{n_2}^2,x_{n_2+1}^1,\cdots,x_l^1,q_1^1,\cdots q_{n_1-1}^1,q_{n_1}^2,\cdots,q_{n_2}^2,q_{n_2+1}^1\cdots,q_l^1),$$
$$V_2^{'}=(x_1^2,\cdots,x_{n_1-1}^2,x_{n_1}^1,\cdots,x_{n_2}^1,x_{n_2+1}^2,\cdots,x_l^2,q_1^2,\cdots q_{n_1-1}^2,q_{n_1}^1,\cdots,q_{n_2}^1,q_{n_2+1}^2\cdots,q_l^2)$$

In order to mutate a chromosome, one pair of positions (m_1, m_2) is randomly generated such that $1\le m_1<m_2\le l$. A new chromosome is formed by randomly regenerating $\{x_{m_1},x_{m_1+1},\cdots,x_{m_2}\}$ and $\{q_{m_1},q_{m_1+1},\cdots,q_{m_2}\}$:

$$V_1^{'}=(x_1,\cdots,x_{m_1-1},x_{m_1}^{'},\cdots,x_{m_2}^{'},x_{m_2+1},\cdots,x_n,q_1,\cdots q_{m_1-1},q_{m_1}^{'},\cdots,q_{m_2}^{'},q_{m_2+1}\cdots,q_n).$$

The evaluation operation uses a rank-based evaluation function to evaluate the chromosomes. Suppose the population size is *pop_size*. We sort the chromosomes according to their objectives and use V_i to denote the *i*th fittest chromosome; saying V_1 is the fittest one and V_{pop_size} is the worst one. Thus, further giving a parameter $a\in(0,1)$, fitness can be assigned by the following function,

$$fitness(V_i)=a(1-a)^{i-1},i=1,2,\cdots,pop\_size$$

The selection operation is a fitness-proportional selection according to the fitness and the roulette-wheel method is adopted as its implementation.

4.3. Hybrid Intelligent Algorithm

The HI algorithm is a combination of the aforementioned two parts as shown in Fig. 2.

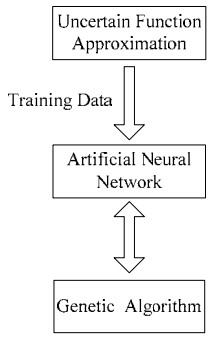

Fig. 2. Hybrid Intelligent algorithm

We employ approximation method to generate sufficient groups of training data for ANN weights training. After being trained, ANN can be used to approximate the uncertain function. Then GA is activated to find an optimal solution. In each iteration, the evaluating operation of GA utilizes ANN to get objective values of the chromosomes or to check the constraints. The detailed procedure is summarized as follows.

Hybrid Intelligent Algorithm

Step 1: Generate training data (both input and output) for the uncertain function by approximation method;

Step 2: Train an ANN to approximate the uncertain function according to the generated training data;

Step 3: Initialize a population of randomly constructed chromosomes $(\vec{\mathbf{x}},\vec{\mathbf{q}})$ and check their feasibility (by trained ANN);

Step 4: Update the chromosomes $(\vec{\mathbf{x}},\vec{\mathbf{q}})$ by crossover and mutation operations. Check the feasibility of the offspring (by trained ANN);

Step 5: Calculate the objective values for all the chromosomes (by trained ANN) and the fitness of each chromosome;

Step 6: Select the chromosomes for next generation;
Step 7: Return to step 4 until pre-configured generations are produced.

HI algorithm is a general algorithm for all the proposed models. Only small modifications are required when it is applied to different models. 1) Different models have different uncertain functions, thus they should adopt corresponding approximation method discussed in IV.A. 2) With CCP models, ANN is used to check the feasibility of chromosomes and in BCP models, ANN is employed to evaluate the chromosomes.

5. Experiments and Evaluation

5.1. Traffic and Topology Settings

A 14-node real topology [32] is considered in our experiment, as redrawn in Fig. 3. It has been shown in [33] that while the majority of SD pairs donate only a small number of flows, the bulk of flows are generated by a tiny fraction of SD pairs. It is also indicated in [34] that there exists large flows, media flows and small flows within a single link. Based on these observations, we employ a mixture of three distributions to imitate the features of real traffic flows. Without loss of generality, we generate four traffic scenarios.

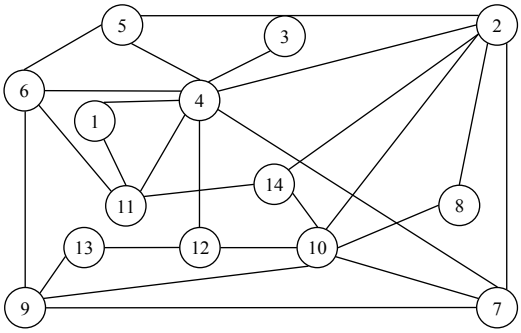

Fig. 3. A real-life network topology

Scenario one: For each flow, its rate is generated by the first exponential distribution ($\lambda = 250$) with probability 0.05, by the second exponential distribution ($\lambda = 20$) with probability 0.15, and the third exponential distribution ($\lambda = 1$) with probability 0.8.

Scenario two: For each flow, its rate is generated by the first uniform distribution (225, 275) with probability 0.05, by the second uniform distribution (18, 22) with probability 0.15, and the third uniform distribution (0.9, 1.1) with probability 0.8.

Scenario three: For each flow, its rate is a constant of 250 with probability 0.05, is a constant of 20 with probability 0.15, and is a constant of 1 with probability 0.8.

Scenario four: For each flow, its rate is generated by the first Gaussian distribution (250, 30) with probability 0.05, by the second Gaussian distribution (20, 3) with probability 0.15, and the third Gaussian distribution (1, 0.3) with probability 0.8. (A constraint that the flow rate should be positive is given.)

5.2. Results

The cost function for calculation is defined specifically as $c_i = D(q_i) = 1 + \frac{1}{2}(q_i / 500)^2$ and the confidence level in the SCCO criterion is set to be 0.9. The produced population size and generation of GA are configured as 200 and 3000 respectively in the experiments. The probability of mutation operation and crossover operation in GA are set to be 0.2 and 0.5 respectively.

Due to the introduction of uncertain parameters of flow rates, it is impossible to compare our calculation results with the exact optimal solutions. Instead, we set up simulations to assess the constructed model and the proposed HI algorithm. We first use the stochastic optimization model and HI algorithm to calculate the deployment strategy in the aforementioned topology. Then a simulation environment is further created for evaluation, which is carried out under same topology and traffic scenarios by configuring the monitors with the calculated deployment strategy (locations and sampling rates).

Fig. 4 – Fig. 7 show the results obtained for CCP by the SEVO criterion and the SCCO criterion respectively.

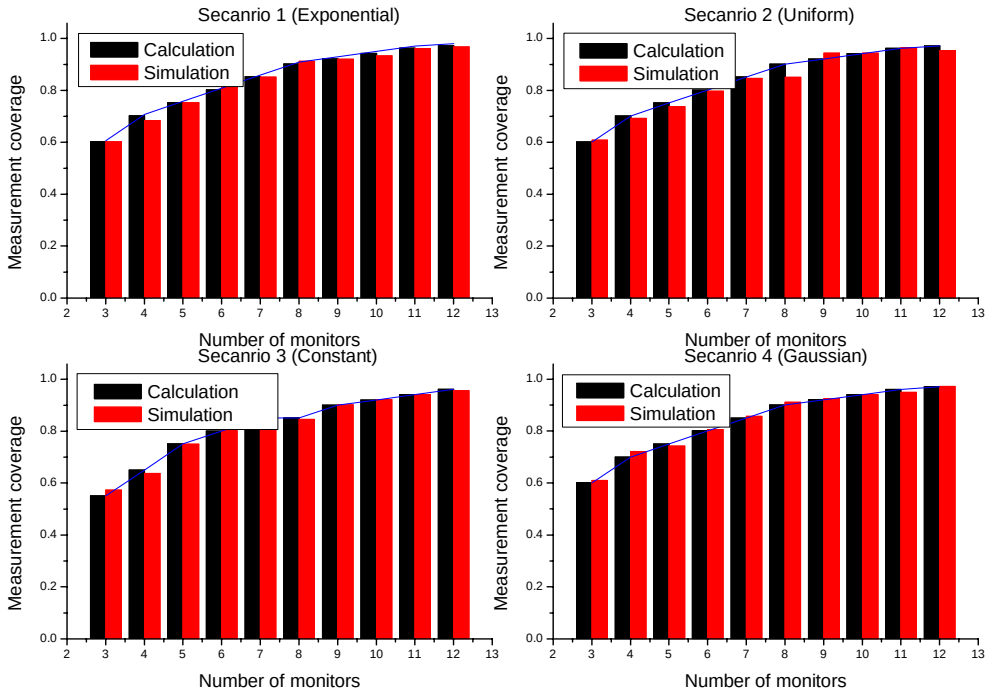

Fig. 4. SEVO criterion for CCP, Number of monitors vs. Coverage

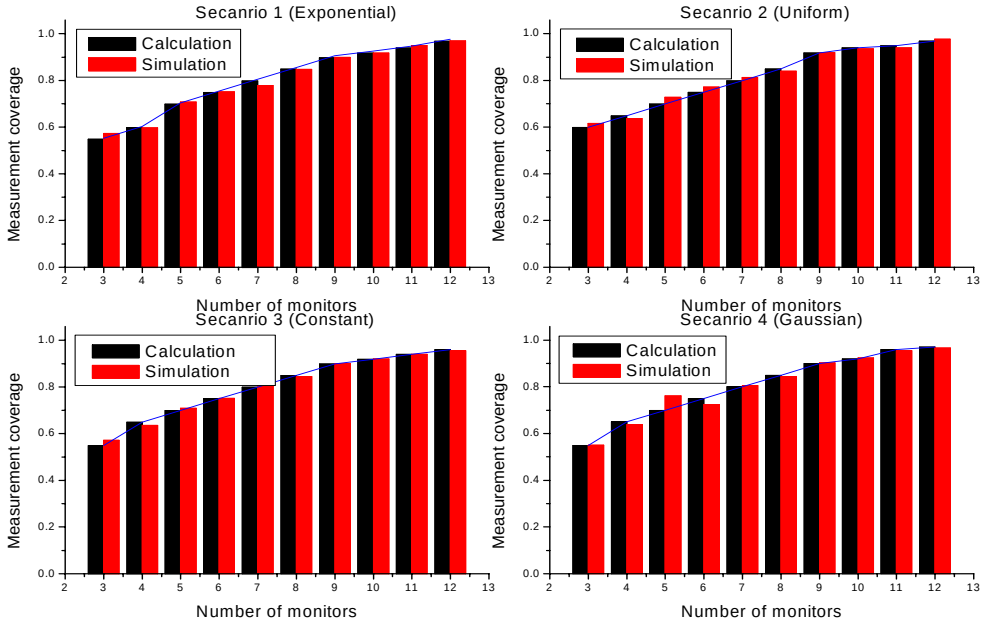

Fig. 5. SCCO criterion for CCP, Number of monitors vs. Coverage

Fig. 4 shows the relationships between the number of monitors and the measurement coverage under SEVO criterion for CCP. It is demonstrated that the calculation results (black bars) match the simulation results (red bars) quit well. We observe from the figure that about 6 monitors can monitor 80% of the traffic and 8 monitors can monitor 90% of the traffic. Note that, the measurement coverage curves (the blue lines in the Fig. 4) are approximately concave functions[4] of the number of monitors. The measurement coverage increases slowly by adding new monitors when the coverage is more than 0.9 in each traffic scenario.

Fig. 5 depicts the number of monitors vs. the measurement coverage under SCCO criterion for CCP model, in which the results are similar with the ones in Fig. 4. The calculation results meet the simulation results well and the measurement coverage curves have the similar trends of diminishing measurement

[4]Define $f: I \to R$ is a concave function on I, if $f(\lambda x_1 + (1-\lambda)x_2) \geq \lambda f(x_1) + (1-\lambda) f(x_2)$, $\forall x_1, x_2 \in I, \lambda \in [0,1]$.

coverage gains.

Fig. 6 and Fig. 7 plot the relationship between the measurement coverage constraints and deployment cost under SEVO and SCCO criterion for CCP respectively. In these two figures, the deployment cost increases sharply when the coverage is larger than 0.9 and the curves of four traffic scenarios are all convex curves. It is learned from the two models for CCP that, the marginal increase in fraction of deployment cost increases as the coverage constraint increases. By comparing the Fig. 6 and Fig. 7, we observe that SCCO criterion cost a little more under the same coverage constraint.

Fig. 8 – Fig. 11 illustrate the results achieved for BCP by SEVO criterion and SCCO respectively.

Fig. 8 and Fig. 9 indicate the number of monitors vs. the measurement coverage under SEVO BCP model and SCCO BCP model respectively. The two figures show the similar results in three aspects. First, again, the calculation results approach the simulation result quit well. Second, a small number of monitors are sufficient to monitor the whole network. Third, it is shown that the decreasing gain of measurement coverage.

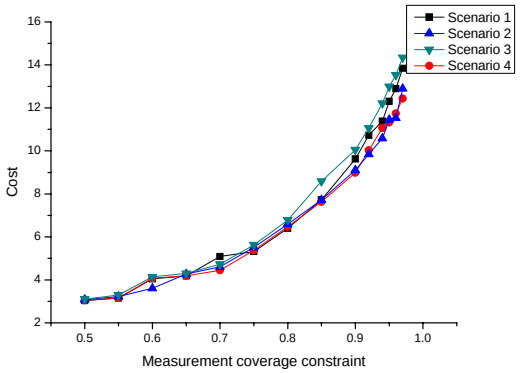

Fig. 6. SEVO criterion for CCP, Coverage vs. Deployment Cost

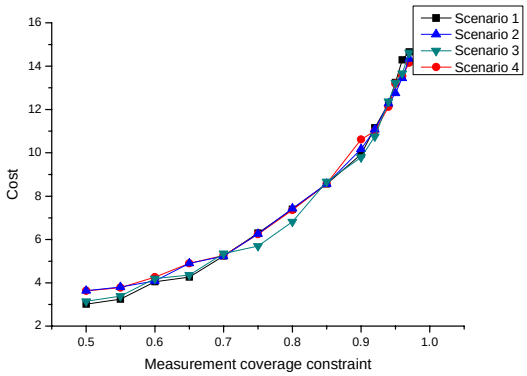

Fig. 7. SCCO criterion for CCP, Coverage vs. Deployment Cost

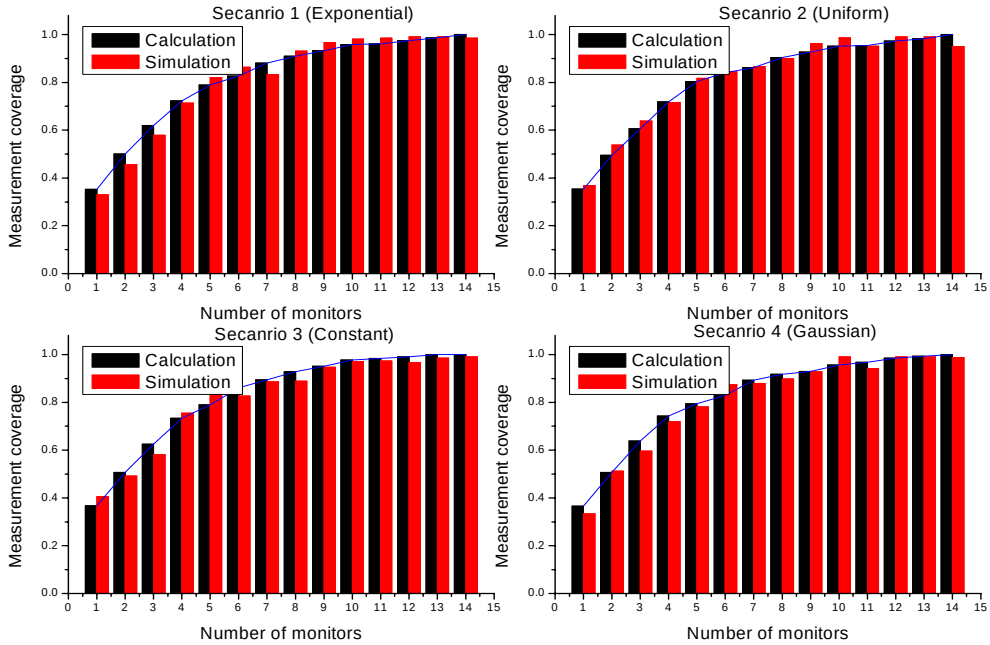

Fig. 8. SEVO criterion for BCP, Number of monitors vs. Coverage

Fig. 10 and Fig. 11 show the measurement coverage under different budget constraints in SEVO and SCCO criterion for BCP, respectively. Measurement coverage increases as the budget increases, but when the budget is more than 10, the increased budget has little effect in term of enlarging measurement coverage. It is indicated that, blindly increasing budget will lead to ineffectiveness in sense of incurring disproportion penalty in implementation cost vis-à-vis the gain in increased coverage. The comparison of these two figures demonstrates that less measurement coverage can be achieved in SCCO criterions than in SEVO criterions under the same budget constraint, and the reason is that the definitions of "measurement coverage" in two

models are different. The coverage measurement defined in SEVO criterions is the expected value, while in SCCO criterion, it should be a probability guarantee under a confidence level of β (90%).

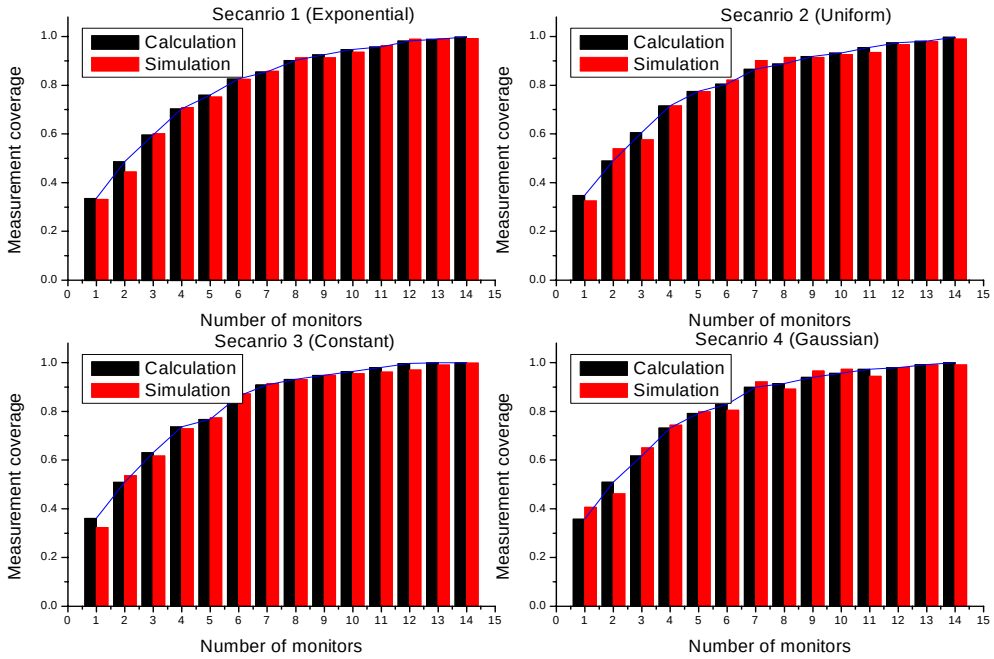

Fig. 9. SCCO criterion for BCP, Number of monitors vs. Coverage

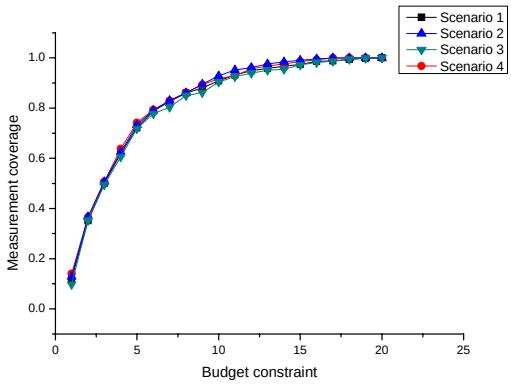

Fig. 10. SEVO criterion for BCP, Budget vs. Coverage

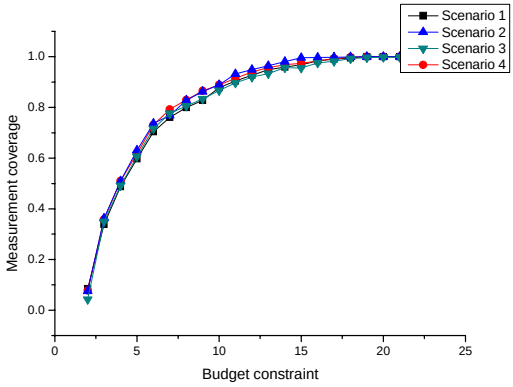

Fig. 11. SCCO criterion for BCP, Budget vs. Coverage

错误！未找到引用源。 shows the detailed results of SEVO criterion, including average measurement coverage, maximum measurement coverage, minimum measurement coverage and probability when the coverage is less than the average coverage. It indicates that the probability of measurement coverage being less than the average coverage is more than 40%. In scenario one, the difference between maximum and minimum coverage is quit large, while in scenario two and three, the variance of measurement coverage is relatively small. It is caused by the variance of flow rate itself. If the flow rate is of large variability, the SCCO criterion is preferred; otherwise, SEVO criterion can be an acceptable solution.

Using the Waxman model [35], we randomly generate larger topologies to investigate the effectiveness between the number of monitors and the topology scales. **错误！未找到引用源。** lists the results obtained by SCCO criterion for CCP under larger topologies. It depicts that deploying DSS is much more efficient in larger topologies, since the increase in the number of monitors is much less than the increase in the number of topology nodes.

We conclude the experiments in three main observations.

- It demonstrates the effectiveness of the DSS deployment strategy: a small number of monitors are sufficient to monitor the whole network and it is more efficient in large topologies.

- It is indicated that, blindly increasing budget will lead to ineffectiveness since the differential coefficient of cost is larger when the measurement coverage is increased.
- Because of the different definitions of measurement coverage in SEVO and SCCO criterion, more cost is required in SCCO criterions than in SEVO criterions to achieve the same measurement coverage regardless of CCP or BCP.
- The results achieved by SEVO criterion and SCCO criterion can be affected by the variance of flow rates. If the flow rates are of little variability, the differences between the results obtained by two models are small; otherwise, the differences are large. In other words, if the flow rates are of little variability, SEVO criterion is sufficient.

5.3. Comparison analysis

We does not directly list the comparisons with the results obtained by some deterministic optimization like [8, 9], but the performance of SEVO is a bound of their works. Since previous works used certain values for flow rates, their methods are at most as good as SEVO if they could accurately estimate the mean value of flow rates. That is also the reason why we only demonstrate the comparison between SEVO and SCCO. As we mentioned above, SEVO, as well as these previous methods, is not suitable for many cases. In addition, previous works failed to present an efficient estimation method for the mean flow rate. But in our HI algorithm, we could well derive the expected value and the β-optimistic value of flow rates even we have less or no advanced knowledge on the flow rate distribution (see Section 6.1).

6. Discussions

6.1. Self-configuration

As mentioned in Section 3, three deployment strategies are required to address the characteristics of flow rates in large time-scale. The monitor should have a self-contained clock/timer, thus it can synchronize the time (day or night, workday or weekend) and further be automatically reconfigured by selecting the corresponding deployment strategy. Since we take account of the flow fluctuation in small time-scale meanwhile to determine the deployment strategies, these deployment strategies can be pre-calculated in general. Therefore, the re-configuring process only needs a kind of lookup operation on the deployment strategy table.

In order to obtain the input parameters of HI algorithm, we should first have some prior knowledge of flow rates distributions in a specific network. But for a specific backbone network without firstly deploying a monitoring system to obtain the specific distribution, how to learn about flow rates distributions and further achieve the optimal deployment strategy?

In such cases, we can trigger the following iteration to tackle this difficulty.

Step 1: Calculate a rough deployment strategy by assuming the flow rate distributions;

Step 2: Configure the monitors with the deployment strategy and collect traffic;

Step 3: Use the collected information as the input of uncertain function approximation and get a new deployment strategy;

Step 4: Go to step 2.

Note that, in Step 3, it is no need for us to know the exact flow distribution to generate corresponding training data. Instead, we can directly input the sampled information to train the ANN and further achieve the deployment strategy. We perform experiments to test the convergence. Flow rates are first synthesized as aforementioned scenario one, two and four. Monitors are first located and configured according to the constant flow rates (scenario three), and then trigger the above algorithm. We observe that the self-configuration will converged in two or three iterations.

6.2. Routing Dynamics

In our models, we do not address the routing dynamics which may be caused by the following two reasons.

First, routing path may be changed due to topology changes. Fortunately, since the monitors are implemented in mid-nodes, like routers, the topology is relatively stable in some period. It is possible to detect the changes in network topology and then refresh the deployment. In fact, the topology of mid-nodes is usually changed determinately by ISPs, rather than randomly. Thus the deployment strategy can still be

pre-calculated.

Second, routing paths can also be changed by network failures or BGP updates. It is not hard to detect such routing changes since the routing messages will propagate to all the routers in most cases. The DSS or the network operators can discern whether routing path change is a permanent one or temporary one. The deployment strategy should be recalculated for permanent routing path change. The discern algorithm is out of the scope of this paper and [36] gave an insight into the monitoring of routing stability.

6.3. Computational Time

In general, the deployment strategies can be pre-computed since the randomness of traffic has been considered in two time-scales. However, some routing changes will result in the recalculation of deployment strategy. Therefore, we still provide an insight into the computational time in this section.

It takes less than 20 minutes to run the computation for one scenario using a PC with a P4 1.7GHZ CPU. In fact, the computational time can be greatly amortized in three aspects. First, GA is not the fastest algorithm, but the inherent parallelism makes it attractive. The individual solution candidate can be evaluated and mutated independently, thus the evaluation and mutation of each chromosome can be handled in parallel. Although the crossover and selection operations require some serial processing, the overhead of these operations can be made smaller by splitting the global population into subpopulations, which evolve relatively independently [37, 38]. Second, the approximation method of each training data is also independent and can be handled in parallel. As the parallel and distributed computing becomes more readily available, the computational time complexity can be greatly amortized in aforementioned two aspects. Third, training ANN using classical gradient-based method (e.g. back-propagation algorithm [26]) is a time consuming task when it is applied to a large topology. However, recent breakthroughs in ANN researches shorten the training time thousands of times. A new training algorithm called Extreme Learning Machine (ELM) [39, 40], which randomly chooses the input weights and hidden neuron biases for Single-hidden Layer Feedforward Neural Networks (SLFNs) and analytically determines the output weights of SLFNs. Both in theory and in experiments, it is showed that ELM provides the good generalization performance at a extremely fast learning speed. Such outcomes are easy to use (no parameters setting problem) and have been utilized by us to speed the training process in the HI algorithm (The source code of ELM is available in [41]).

7. Conclusion

This paper investigated the deployment strategy of DSS, especially, how to optimally locate the monitors and sample stochastic traffic flows so as to maximize the measurement coverage under the constrained budget, or to minimize the cost under the measurement coverage. Different from prior works, we considered the randomness of traffic rate and provide a two-level approach. In the large time-scale level, we define three different deployment strategies; while in the small time-scale level, we formulate the problem using two categories of stochastic optimization models: the SEVO criterion and the SCCO criterion. SEVO criterion purchases the average measurement coverage and is sufficient to the applications where only the average performance is required, or/and where the flow rates are of little variability. SCCO criterion gives a probability bound or guarantee but requires more cost. If the applications are of sensitive (eg. security application) or/and the flow rates are of large variability, the SCCO criterion is preferred. A Hybrid Intelligent (HI) algorithm is utilized to solve the problems. The algorithm consists of two major components, namely, uncertain function approximation and genetic algorithm. The simulations and experiments demonstrate the effectiveness of our algorithm, i.e., only a small number of monitors are sufficient to maintain a high measurement coverage level in a high-speed network. Furthermore, we observe that, blindly increasing budget or the number of monitors has limited improvement on measurement coverage and will lead to ineffectiveness.

As a future work, we will extend the distributed sampling system with other (distributed) algorithms to support traffic information query service like on max, min, average flow rates.

8. References

[1] K. Claffy and S. McCreary, "*Internet Measurement and Data Analysis: Passive and Active Measurement*." http://www.caida.org/outreach/papers/1999/Nae4hansen/Nae4hansen.html, 1999.

[2] N. G. Duffield, C. Lund, and M. Thorup, "*Charging from sampled network usage*," in IMW, 2001: 245 -256.

[3] Cisco, "*Sampled Netflow Data Sheet*."

http://www.cisco.com/en/US/products/ps6601/products_data_sheet09186a0080081201.html.

[4] B.-Y. Choi, J. Park, and Z.-L. Zhang, "*Adaptive Random Sampling for Load Change Detection*," in SIGMETRICS, 2002.

[5] N. Duffield, C. Lund, and M. Thorup, "Estimating Flow Distributions from Sampled Flow Statistics," in *ACM SIGCOMM*, 2003:325 - 336.

[6] C. Estan, K. Keys, D. Moore, and G. Varghese, "*Building a Better NetFlow*," in ACM SIGCOMM, 2004.

[7] N. G. Duffield, C. Lund, and M. Thorup, "*Properties and Prediction of Flow Statistics from Sampled Packet Streams*," in IMW, 2002.

[8] K. Suh, Y. Guoy, J. Kurose, and D. Towsley, "*Locating Network Monitors: Complexity, Heuristics, and Coverage*," in INFOCOM, 2005.

[9] G. R. Cantieni, G. Iannaccone, C. Barakat, C. Diot, and P. Thiran, "Reformulating the monitor placement problem: Optimal Network-wide Sampling," Intel Research Technical Report, 2005.

[10] C. Hu, B. Liu, Z. Liu, S. Gao, and D. O. Wu, "*Optimal Deployment of Distributed Passive Measurement Monitors*," in ICC'06. Istanbul, Turkey, 2006.

[11] B. Liu, *Theroy and Practice of Uncertain Programming*: Heidelberg: Physica-Verlag, 2002.

[12] K. C. Claffy, G. C. Polyzos, and H.-W. Braun, "Application of Sampling Methodologies to Network Traffic Characterization," in *SIGCOMM*, 1993: 194 - 203.

[13] Cisco, "Cisco IOS Netflow Data Sheet." http://www.cisco.com/en/US/products/ps6601/products_data_sheet0900aecd80173f71.html.

[14] C. Estan and G. Varghese, "New Directions in Traffic Measurement and Accounting," in *SIGCOMM*, 2002: 323 - 336.

[15] N. Hohn and D. Veitch, "Inverting Sampled Traffic," in IMC, 2003.

[16] C. Fraleigh, C. Diot, B. Lyles, S. Moon, P. Owezarski, K. Papagiannaki, and F. Tobagi, "Design and Deployment of a Passive Monitoring Infrastructure," in *Proceeding of Passive and Active Measurement (PAM)*, 2001.

[17] NLANR, "NLANR: National Laboratory for Applied Network Research." http://www.nlanr.net.

[18] A. Moore, J. Hall, E. harris, C. Kreibech, and I. Pratt, "Architecture of a Nerwork Monitor," in *Passive and Active Measurement Workshop (PAM)*, 2003.

[19] C. Hu, B. Liu, Z. Liu, S. Gao, and D. O. Wu, "Optimal Deployment of Distributed Passive Measurement Monitors," presented at ICC'06, Istanbul, Turkey, 2006.

[20] L. Li, M. Thottan, B. Yao, and S. Paul, "Distributed Network Monitoring with Bounded Link Utilization in IP Networks," in *INFOCOM*, 2003,**2**: 1189 - 1198.

[21] X. Tang and J. Xu, "On Replica Placement for QoS-aware Content Distribution," in *INFOCOM*, 2004, **2**: 806 - 815.

[22] S. Jamin, C. Jin, A. R. Kurc, D. Raz, and Y. Shavitt, "Constrained Mirror Placement on the Internet," in *INFOCOM*, 2001, **1**:31 -40.

[23] S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang, "On the Placement of Internet Instrumentation," in *INFOCOM*, 2000,**1**: 295 - 304.

[24] K. Thomson, "Wide-area Traffic patterns and Characteristics," *IEEE Networks*, 1997.

[25] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot, "Packet-level traffic measurements from the Sprint IP backbone," *IEEE Network*, 2003,**17**: 6 -16,.

[26] S. Haykin, *Neural Networks - A Comprehensive Foundation*. New York: Macmillan College Publishing Company, 1994.

[27] S. Khuller, A. Moss, and J. S. Naor, "The Budgeted Maximum Coverage Problem," *Information Processing Letters*, 1999, **70**: 39 - 45,

[28] M. Mitchell, *An Introduction to Genetic Algorithms*, Reprint ed: The MIT Press, 1998.

[29] C. R. Reeves and J. E. Rowe, *Genetic Algorithms - Principles and Perspectives : A Guide to GA Theory* 1st ed: Springer, 2002.

[30] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*: Addison-Wesley Professional, 1989.

[31] G. Rudolph, "Convergence Analysis of Canonical Genetic Algorithms," *IEEE Trans. On Neural Networks*, 1994, **5**: 96-101.

[32] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot, "Traffic Matrix Estimation: Existing

Techniques and New Directions " *ACM SIGCOMM Computer Communication Review*, 2002, **32**: 161 -174.

[33] S. Bhattacharyya, C. Diot, J. Jetcheva, and N. Taft, "Geographical and Temporal Characteristics of Inter-POP Flows: View from a Single POP," *European Transactions on Telecommunications*, 2002, **13**: 5-22,.

[34] H. Fang, M. Kodialam, T. V. Lakshman, and Z. Hui, "Fast, memory-efficient traffic estimation by coincidence counting," in *INFOCOM*, 2005, **3**: 2080-2090.

[35] B. M. Waxman, "Routing of Multipoint Connections," *IEEE Journal on Selected Areas in Communications*, 1988, **6**: 1617-1622.

[36] G. Varghese and C. Estan, "The Measurement Manifesto," *ACM SIGCOMM Computer Communication Review*, 2004 , **34**: 9-14.

[37] J. P. Cohoon, S. U. Hedge, W. N. Martin, and D. Richards, "Punctuated Equilibria: A Parallel Genetic Algorithm," in *Internatioanal Conference on Genetic Algorithms*, 1987.

[38] M. Tomassini, "Parallel and Distributed Evolutionary Algorithms," in *Evolutionary Algorithms in Engineering and Computer Science*. Chichester: John Wiley & Sons, 1999.

[39] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *2004 IEEE International Joint Conference on Neural Networks*, 2004, **2**: 985-990.

[40] M.-B. Li, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "Fully Complex Extreme Learning Machine," *Neurocomputing*, 2005, **68**: 306-314.

[41] "*The Basic MATLAB Codes of ELM*." http://www.ntu.edu.sg/home/egbhuang/ELM_Codes.htm.