IJICTR International Journal of Information & Communication Technology Research

Volume 5- Number 3- Summer 2013

A Dynamic Resource Allocation Method for Virtual Datacenter Resources Using CoCo-VM Approach

Mohammad Reza Ahmadi
Cyber Space Research Institute (CSRI)
Tehran, Iran
m.ahmadi@itrc.ac.ir

Received: December 24, 2012- Accepted: June 30, 2013

***Abstract*—A dynamic resource allocation method for virtual resources in virtualized data centers has been proposed in this paper. Since resource allocation with constraint in virtual environment is NP-hard, the solution has been focused on approximate methods based on immune system mechanism using agent based model and Cooperative Co-evolutionary Algorithm (CoCo-VM) that appears to perform well in finding a plausible answer [1]. The novelty of our approach lies in integrating an agent based greedy algorithm based on immune system functionality together with the cooperative co-evolutionary concept as an intelligent solution for virtual resource allocation in a large scale and distributed virtualized datacenters. Here, some mathematical analyses have been done to identify the parameters essential to assign a suggested allocation approach. Results of different evaluations in pure immune system (PIS)[1] and CoCo-VM methods demonstrate that, for the scenarios under consideration, the proposed resource allocation approach can significantly reduce resource consumption, increase number of successful services, and achieve higher performance.**

Keywords: **Virtual resource allocation, cooperative co-evolutionary algorithm, immune system, agent-based greedy algorithm**

I. INTRODUCTION

Virtualization is a technology that combines computer resources and provides different operating environments using methodologies like hardware and software partitioning or aggregation, partial or complete machine simulation, emulation, time-sharing, and more. More recently, virtualization at all levels has become important again as a way to improve system security, increase reliability and availability, reduce costs, create better adaptability to workload variations, support easier migration of virtual machines among physical machines, and prepare easy coexistence of legacy applications in data center applications. Virtualization provides flexible and manageable execution environments that are specialized resources for different applications which are using share resources and delivering an expected performance, security and isolation [2]-[4]. One of the important challenges in datacenters is resource allocation and dynamic resource management for virtualized resources. Resource allocation needs to not only guarantee enough virtual resources to meet the performance goals, but also prevents over-provisioning in order to reduce cost and allows concurrent hosting of many applications. Resource control functions are integrated in a data center at two different levels of abstraction: virtual machine and virtual resource pools. The central management is responsible for determining the necessary resources which are needed by each service based on framework of service level agreement.

[1] Pure immune system with centralized management and without modification functions.

In this case local controller minimizes leasing costs by avoiding over-provisioning for the applications running on the virtual machines. The key to service oriented resource allocation is the ability to efficiently find the minimum amount of resources that an application needs to meet the desired quality constraint [5]-[8]. We can consider the procedure as a classical 0/1 knapsack problem which could be represented as a specific instance of resource allocation. Consider a 0/1 knapsack problem with knapsack capacity, weight item and service quality. This is a multidimensional problem with different weight items and knapsack capacity with different vectors. We can add more dummy knapsacks to have a multiple knapsacks generalization, and these knapsacks can be described via different capacity vectors. As a result, the problem to be tackled is NP-hard. Here an approximate way to solve the formal problem is proposed [9-10]. To solve this problem, a dynamic resource management approach that enables automatic and adaptive resource provisioning in accordance with quality of service and Service Level Agreements (SLA) specifying dynamic tradeoffs of service quality and cost has been considered [11,12]. An intelligent technique to characterize the relationship between application workload and available virtual resources has been introduced. A prototype of the proposed resource management system has been deployed on a simulation test bed. Popular datacenter services have been generated by workload generator and applied to the system to evaluate the cooperative co-evolutionary greedy algorithm in computational process and virtual resource allocation. The rest of this paper is organized as follows. Section 2 provides general architecture of a virtual datacenter. Section 3 presents the proposed CoCo-VM algorithm. Section 4 introduces technical parameters for evaluation of the proposed algorithm. Section 5 presents experimental and result validation. Finally the paper has been concluded in section 6.

I. General Architecture of a Virtual Data Center

1.1. Basic Idea and System Architecture

The main functions and building blocks of a virtual data center are shown in figure.1. All entry services are collected in application service controller (ASC). Then, the requests are mapped to the internal format and forwarded to the central management controller (CMC). Moreover, any incoming service needs specific quality based on SLA agreement that should notify to the CMC. Both service type and service quality are the main criteria for decision making and resource allocation in CMC. On the other hand, all hardware systems are attached in a unified platform and mapped to a virtual resource pool environment which is supported by virtualization software. The virtual resource pool controller assigns specific service catalog of resources to each virtual machine. The resource assignment is managed by central management under the rules and quality agreement. Here, we explain the data structure and functional relations in the proposed model. At first, the application service controller crates a digital vector based on vector catalog format for each entry service and forwards a Request Vector (RV) to the central management controller. Management controller investigates the expected quality of service for the incoming request via SLA information. Considering the RV and QoS constraint, the central management creates a Job Vector (JV) for each service. On the other hand, the resource pool controller has assigned all virtual resources to the available virtual machines based on predefined rules and plan. Virtual management algorithm tries to find the best Service Vector (SV) for each JV within the available AVs. If the algorithm fails to find a suitable AV, it selects the best available vector with maximum fitness in the mismatching fields, and continues the process to fulfill the residual resources. In this step, the process will continue in other SVs to modify the resource value in all mismatching fields and obtain the acceptable fitness in all the possible fields. The final vector with proper resource value will send to the resource pool controller. This part modifies the available resources in SV and assigns a virtual machine for the corresponding JV. As a result, the JV and requested jobs are assigned to the best selected virtual machine.

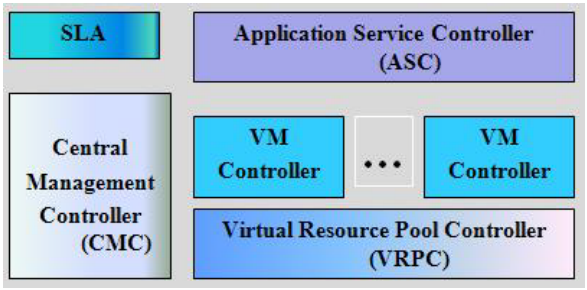

Fig.1. Proposed architecture and building blocks for a virtual datacenter

1.2. Building Blocks of the System

1.2.1. Virtual Recourse Pool Controller

In the resource pool controller, different physical hardware are collected in a unified pooling system and categorize in different resource types, then map to a unified virtual compartment. Partial of each virtual resources are assigned to different virtual machines. Each virtual machine supports dedicated service by employing partial resource of the pooling system. Virtualization software can create variety of virtual machine with different resource capacity. Data center's resource management should assign minimum resources to all operational virtual machines for acceptable performance. To achieve minimum resources to support specific service level agreement, central management controller should assign necessary resources to the related virtual machine. Central resource controller manages all dedicated virtual resources to optimize the resource distribution in the pooling environment. Since created

virtual machines are independent, a heterogeneous assignment is applicable. This approach can change or remove a service in a virtual machine without affecting the functionality of other machines.

1.2.2. Distributed Virtual Machine Controller

The main task of the virtual machine controller is operational management and reporting to the central management controller. Interaction between virtual machine controller and central management controller creates an opportunity for dynamic change of resource values to comply with the expected quality of service. Virtual machines start operation when they receive a job vector from central management controller and a resource vector from virtual resource pool controller.

1.2.3. Central Management Controller

Central management controller has responsibility for resource assignment to all virtual machines such that, each virtual machine can support related service with expected quality. This part receives the entry service vectors from application service controller and available resource vectors from virtual resource pool controller. The CMC creates the job vectors with QoS constraint and seeks to select the best available resource for each job using CoCo-VM algorithm. The proposed algorithm in this module uses a greedy approach based on immune system to select the best matching resource for all fields in the job vector. Moreover, the CMC should cooperate with virtual resource pool controller to adjust the necessary resources in the nominated virtual machines. The central management controller allocates virtual resources based on available resources and modifies the non-matching fields for maximum fulfillment. As a result, when a resource vector with acceptable resource field has been selected for a service, the job and selected resources are assigned to the nominated virtual machine for the requested operation.

II. PROPOSED COCO-VM ALGORITHM

2.1. Basic Concept

In this part, an optimization method based on immune system and genetic algorithm (GA) which has been supported by cooperative co-evolutionary technique has been introduced. Originally, immune system is a biological model and genetic algorithms are successful technique for optimization where repeatedly modifies with genetic operators in a searching area and seeking for the result with the best fitness. The co-evolution algorithm is an extended version of GA with multiple functions where includes several genetically isolated group that evolve in a parallel model. Individual member of each field collaborates with other members in similar fields and improves their fitness according to a specific objective function [13]-[15]. We have proposed the CoCo-VM algorithm using Jini-Grid environment based on figure 2. The system consists of several set of separated worker agents where each set manage an individual field and they are coordinating by a master agent. Each CoCo-VM group concentrates on a specific field of resources (e.g. CPU speed, available RAM, storage capacity, I/O ports and so on) where considering the bilateral relation among similar fields. The system has been implemented in a cluster of servers where all the tasks are distributed in different hosts with sharing functionality. Based on figure 2, the initial population of AVs is generated by capacity vector and is stored in the resource management controller. Each vector has several fields and evolves in all fields separately while there is cooperation among all similar fields. In the first level of process, a particular co-evaluation algorithm cooperates with other similar fields under a master agent management system. Each field generates a set of representative values which has the best fitness with JV fields. As a result, based on final representative set, the best RV for each JV are selected and stored in the control management system. This algorithm introduces a recursive procedure for selecting the best matching vector in the available resource pool. Acceptable quality variation and modification procedures are complementary functions for final resource allocation.

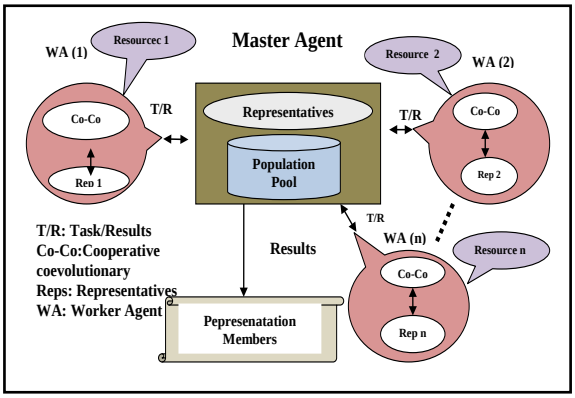

Fig.2. Functional diagram of CoCo –VM algorithm

2.2. Suggested Operational Function

In task of operation, the system creates a RV vector for each entry service. Central management controller creates a JV vector for each job considering RV and SLA constraint. The JVs evolves in the AVs for available resources and looking for the best matching vector which cover all the active fields in the available vectors. In the case of successful result, the algorithm will select the best matching vector (MVi) with the best fitness compared to the other members in AVs. For unsuccessful cases, the algorithm continues for residual resources in non matching fields of JV in the AV population. The process will continue to modify all the mismatching fields in a recursive model to get an acceptable fitness margin. In the case of final success, the system will send the MVi to the resource pool controller to update the AVi and forward the RVi to the same VM. In our prototype model, all vectors include several resource fields (i.e. CPU, Memory (M), Disk Capacity, Input/output Bandwidth, File system (FS), Network (Net)). As a result, the system will select the best matching vector with acceptable threshold which can

support a job vector with predefined constraints based on service SLA.

2.3. Data Structure in the Proposed Algorithm

In order to create a uniform model for all vectors, a set of digital windows is assigned for each field. The available resources are set to its corresponding fields where we assign null for the unused fields. We consider a similar vector as a mask filed which can apply to the pattern field where the value of "1" generates a corresponding bit and the value of "0" generates a don't care (*X*) bit. This structure will model a vector which binds a family of vectors with common characteristics. In description of the vectors, the *RV* represents groups of requested services (RV_i). Also, the *JV* represents groups of vectors for requested services with quality of service constraint. The constraints are according to service level agreement. Equation 1 introduces the RV and JV vectors where "*n*" refers to the number of services:

$$RV = \{RV_1, RV_2, RV_3, ..., RV_n\}$$
$$JV = \{RV_1 |_{QoS}, RV_2 |_{QoS}, RV_3 |_{QoS} .....RV_n |_{QoS}\} \quad (1)$$

In this model, a resource pool stores total capacity vector (CV) and the system distributes different virtual resources to different Virtual Machines through the Available Vectors (AVs). As a result, a set of expected job vectors (JVs) and a set of available resources (AVs) have been created for optimization algorithm. The algorithm tries to evolve in the set of AV and find the best options with acceptable fitness value. If the fitness in the selected value is more than acceptable threshold, the answer is unacceptable. In this case, the process will continue by selecting the best fitness vector for the residual values. Thus, the final answer could be summation of the basic and additive selected vectors. In this case, the virtual management control applies the final selection of the AVs to CV controller. Resource pool controller assigns a suitable VM and ask central management controller to forward the JV to the dedicated VM.

V M Algorithm
Match-Vector (resource) {
JV set = {
JV *1=10111011100xxx ….1 0 xx 1 1 1 0 x001100,* T=0.56
JV *2= 0x 00x 11xx1001.....xx1 x x x 1 0 x 01 x 1x,* T=0.71
JV *3= xx xxx1111xxx11 …. x x 1 1 x 0 x 01 100 1,* T=0.82
•
•
JV *n= xx 110100011x.x..... x x 0 x 1 1 x x 11 1xx,* T=0.56 };
For all AVi in Resource pool {
If the best Matching (JV, AV)>Threshold
If the best Matching (JV, Residual AVi) >Threshold
Continue;
}
Else **No- Available-Service;**
Match-Vector = SUM (AVi),
}

Fig.3. Process of VM algorithm

Figure 3 shows the process of VM algorithm for selection of the best matching vectors. To implement the algorithm, we have proposed a triple segment string schema for format of vectors. An X-Vector consists of a binary pattern including a pattern field, a mask field, service type and a real-value activation threshold (AT) based on figure 4.

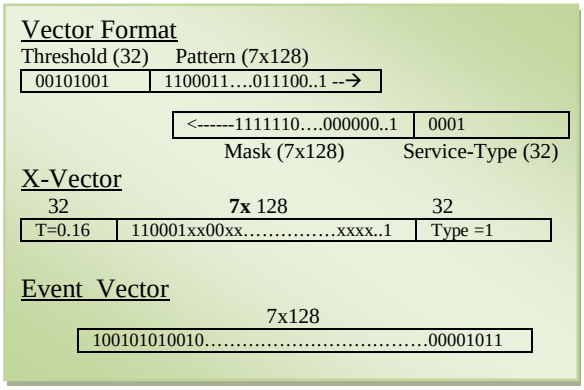

Fig.4. Format of the X-Vector.

In figure 4, format of the main vector, X-vector and Event vector have been shown. In the main vector, we have considered 32 bits for threshold field, 7X128 bits for pattern field, and 7X128 bits for the mask field and 32 bits for service type. Figure 5 shows available resource fields which have been considered for each vector. Note that string of vectors in figure 5 has 128 bits for each field. Also, the algorithm employs at least one vector for each job where the number of active resources depends on the requirements of the jobs. As a result, the aim of the algorithm is to find the best resource vector which covers all the resource constraint so that the selected AV vectors have the best matching with the requirement of the JV vectors.

CPU	M	Disk	IN-B	OUT-B	FS	Net
110…1	1100…1	100..1	100…1	110…0	100..1	000...1

Format of Virtual Resources

Fig.5. Resource format

III. TECHNICAL PARAMETERS FOR EVALUATION OF THE PROPOSED ALGORITHM

In this section, some mathematical analysis to calculate the best matching vectors with maximum fitness has been given. Furthermore, some key parameters for evaluation of the proposed system have been introduced.

3.1. Maximum Similarity

In this part, a mathematical analysis for obtaining the maximum matching vectors has been given [7]-[9]. In Co-Co algorithm, when a JV evolves in one AV field, it cooperates with similar fields in AV group. Thus, the selected vector has the best fitness in all related fields. In our prototype model, we have prepared a set of available resources which are assigned by resource pool controller. Each set of fields may fit with specific service. The aim is to generate a set of AVs which can support a set of JVs under the QoS constraint. To implement this model, we have proposed a triple segment string schema including the threshold field, 128x7 bits pattern field and finally the service type field as mentioned in sec 3.3. For the

selected vectors with the best fitness, we should calculate the match-strength factor. We define *S* as the match-strength factor between two binary strings of *x* and *y* where $x \in JV$ and $y \in RV$ with size of 128x7 bits. Value of *S* can simply obtain by comparison of similar position bits in *x* and *y* based on Eq. 2:

$$S(x,y) = \sum_{i=1}^{l} \begin{cases} 1 & if [x_i] \equiv [y_i] \quad or \quad [x_i] \neq X \\ 0 & else \end{cases} \tag{2}$$

Where *l*=128x7 and *X* refers to don't care value. In order to find the maximum fitness for RV_j, we obtain a member which has the maximum match-strength based on Eq.3.

$$S_{\max_i}(JV_i, AV_i) = S(JV_i, AV_i) \Big| S(JV_i, AV_j) \geq S(JV_i, AV_j) \quad for \quad i=1....a \quad for \quad i,j=1.....n \tag{3}$$

Where a refers to number of vectors and i and j refer to number of fields in each vector. In set of vectors which are generated for the existing JVs, a representative member introduces a member which has the best fitness for each service. We assume service vector *SV* has the maximum match-strength (best fitness) in set of AV for job vector JV_i where:

$$SV = \sum_{i=1}^{a} AV_i \tag{4}$$

Thus, the selected service vector has the best fitness with actual job vector; the result vector is the best answer for supporting the service.

3.2. Validating the Parameters

For validation of the results in the proposed model, probability of available resource ratio (PAR) and false accuracy ratio (FAR) have been introduced as the two evaluation parameters**.** Probability of available resources refers to a successful selection of resource where the probability of false accuracy ratio refers to the probability of selecting a resource that cannot meet the expected QoS. In order to calculate the probability of right selection, we have defined the following parameters:

T = A threshold level which is obtained by dividing the decimal value of threshold field in x-vector to the vector length. The obtained result gives a real value between zero and one.

Hit = Selection of enough resource which refers to summation of all successful vectors for JVs.

We obtain the hit value based on Eq. 5.

$$Hit = \sum_{i=1}^{a} \begin{cases} 1 & if \quad [S_{\max_i}(JV, RV_i) / Z(m_i)] \geq T_i \\ 0 & else \end{cases} \tag{5}$$

Where Ti is the threshold for member m_i and $Z(m_i)$ is obtained based on Eq. 6.

$$Z(m_i) = \sum_{j=1}^{128} \begin{cases} 1 & Maskbit(j)=1 \\ 0 & else \end{cases} \tag{6}$$

Probability of detection ratio is calculated based on Eq. 7.

$$PDR = Hit/a \tag{7}$$

On the other hand, the false rate (F) is sum of the vectors that are supported for expected QoS based on Eq.8.

$$F = \sum_{i=1}^{a} \sum_{j=1}^{b} \begin{cases} 1 & if \quad [S(JV_i, RV_j) / Z(m_i)] \geq T_i \\ 0 & else \end{cases} \tag{8}$$

Probability of false ratio is calculated based on Eq. 9.

$$PFR = F/b \tag{9}$$

Moreover, the false accuracy ratio (FAR) is obtained based on the following equation:

$$FAR = \frac{PFR}{PFR + PDR} \tag{10}$$

Equations (7) and (10) are introduced the two important evaluation parameters for resource allocation procedure. Considering the two parameters, we have introduced successful ratio as a key parameter for system evaluation based on Eq.11.

$$SR = PDR - FAR \tag{11}$$

These parameters are used for validation of the results in sec. 5.

IV. Experimental & Results Validation

This section presents several evaluation scenarios to show actual behavior of the CoCo-VM algorithm and to demonstrate characteristics and advantages of the proposed resource management method. To the best of my knowledge there is no prior work using a genetic algorithm modeling approach to virtual data center resource management. The following briefly summarizes other work with some common elements with this paper's approach. Rule-based systems: This approach uses a set of event-condition-action rules (defined by system experts) that are triggered when some precondition is satisfied (e.g., when some metrics exceed a predefined threshold). For example, the HP-UX Workload Manager [16] allows the relative CPU utilization of a resource partition to be controlled within a user specified range, and the approach of Rolia [17]. Observes resource utilization (consumption) by an application workload and uses some "fixed" threshold to decide whether current allocation is sufficient or not for the workload. With the growing complexity of systems, even experts are finding it difficult to define thresholds and corrective actions for all possible system states. In [18] the CPU shares are dynamically allocated with the goal to optimize a global utility function, under varying workload levels, and in [19] the proposed architecture involves different Application Environments (AEs), each one comprising several physical machines bounded together. Each AE serves different classes of transactions, and server could be moved from

one AE to another, to optimize a global utility function which is based on the performance metrics of the AEs, like response time and throughput. The proposed solver searches for the optimal number of physical servers for each AE, with a beam search algorithm. In continue, using different evaluation scenario, I am going to show the effect of the proposed algorithm compare to the pure genetic algorithm rather than comparing with different non-intelligent methods.

4.1. Prototype System in our Experiments

In prototype system, five (HP Proliant Blade 45p) servers with windows operating system, HP storage system, virtualization software, jini-grid platform and Matlab software supporting MDCE[2] have been installed. Twenty virtual machines have been created in a virtualization platform. A database system with 10.000 vectors has been created to store resource vectors with seven fields of CPU, Memory (M), Disk Capacity, Input/output Bandwidth, File system (FS), and Network capacity (Net). Service generator and SLA rules with constraints have been installed. On jini software platform the CoCo-VM algorithm has been installed and configured. The process for assigning the requested vectors to entry services in simulation environment has been activated where the content of this vector has been updated in CMC based on expected quality of service.

4.2. Functions of the Prototype System

In this section, architecture and functional procedure of the prototype system have been discussed. Collection of hardware systems creates a resource pool environment to support necessary resources for all virtual machines. A digital vector with seven fields has been introduced to support resource assignment. Figure 5 shows a sample of the proposed resource vector. Initial resources have been applied to the resource vectors and stored in the DB system. On the other hand, the entry services are mapped to a uniform format and initialized based on type of service and quality. Based on service specification and resource constraint, the algorithm creates a job vector for each service. In operation, the algorithm tries to find a suitable virtual machine which can support all fields of the job vector. If a suitable virtual machine has been found, the system starts its operation. Otherwise, the algorithm will continue in a recursive process to modify the unacceptable resource fields. This process will continue until the algorithm obtains a resource vector with acceptable fitness value for that service vector. The functional process is based on procedure in figure 3. This procedure clarifies the process and shows the possible options in resource allocation algorithm.

[2] Matlab Distributed Computing Engine.

On the other hand, to evaluate the proposed system, two evaluation scenarios have been used. In the first scenario, system response and cooperation of the members for a converged solution have been evaluated. In the second scenario, successful service rate and resource utilization together with system performance have been measured.

4.3. Behavior of the Prototype System

In order to confirm behavior of CoCo-VM algorithm and contribution level of the existing members for convergence, an evaluation scenario has been developed. For technical specification of the prototype, please refer to sections 2 and 3. In this scenario initial values for system response time are obtained during several trial tests and optimization procedures. In this scenario a Jini-grid platform with selected data set has been adopted. The process starts from worker agents in one group where each agent initiates its activity by generating a random preliminary population with size of 128x7. During the process, if matching rate has stagnated, new members are added to the system and any unproductive members will be tagged for future use. Once required number of generations has been produced or the specific conditions are met, the process will terminate. Figure 6 shows progress of fitness (matching rate) in different generations. In operation environment, several groups of parameters (CPU,RAM,….) cooperate together to cover the whole existing population. Any progress in cooperation will improve the matching rate; otherwise, the progress will stagnate and we need to add new members for more generations. It should be noted that by adding a new group, contribution among the members may downfall for several steps. In this case, matching rate will reduce and the system may deviate from its expected track. This transit behavior will recover quickly by improving the cooperation among existing and new members.

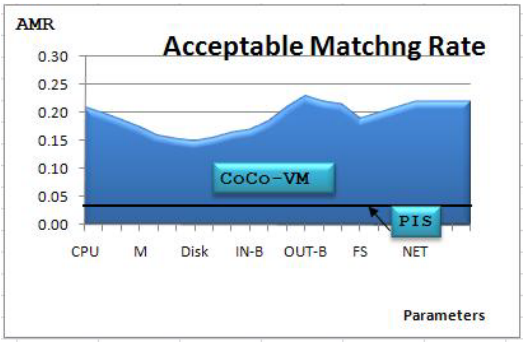

Fig. 6. System response for different generations

As a result, the system does not stagnate or deviate to a non-convergent response that is a vital advantage for the algorithm. On the other hand, we have evaluated the non-cooperation effect of members on system response. Figure 7 shows the non-cooperation rate among the members for 100 consecutive generations. The results show that system learning increases matching rate while decreases the non-

cooperation rate. Results of both evaluations show that CoCo-VM system has a positive cooperation among the members and leads the results to a converged solution. Moreover, the training will improve member's cooperation and reduce unexpected deviation in system response.

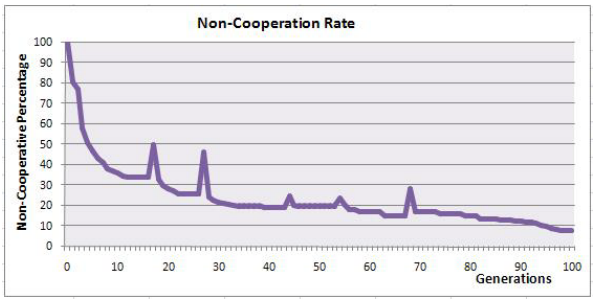

Fig.7. the non-cooperative rates in new generations

4.4. Acceptable Threshold Rate

Traditional resource allocation systems have been focused on an exact matching or zero deviation $(\delta = 0)$, this is an essential point which has been changed by new intelligent systems which introduce the concept of a fitness value. In fact, the acceptable affinity rate limits the system to an acceptable deviation[3] level where the maximum deviation refers to minimum matching boundary. As the systems become more intelligent, the acceptable deviation rate affects the decision criteria where the certain level of similarity replaces with exact matching condition. Thus, the intelligent system has capability to concentrate within a reasonable boundary which increases alternative choices and flexibility. In our system, introduction of any acceptable margin increases opportunity for more available resources which ultimately may increase probability of successful service rate and service quality. In pure immune system (PIS), resources must satisfy exact matching criteria which are required by a service; i.e. virtual machines with non-matching resources are ignored. On the other hand, for CoCo-VM, allocation is based on acceptable fitness value criteria. Since sensitivity of each resource is different in each service, upper and lower thresholds may consider different for each field in the vector. This is an advantage of the algorithm for accepting a reasonable variation of the resources which are allocated to virtual machines. As a result, the resource allocation in CoCo-VM has more flexibility and diversity compare to PIS system. Moreover, the searching functions in CoCo-VM are able to trace in the vectors which are close to the expected resources and have an acceptable variation compared to the PIS system. This characteristic expands the border of activity for the algorithm and increases more opportunity for available virtual machines. To manage system specification, a set of popular services and sensitivity of them have been investigated. In this evaluation based on sensitivity of each resource for certain quality, maximum acceptable deviation has been tested and the final values are depicted in figure 8.

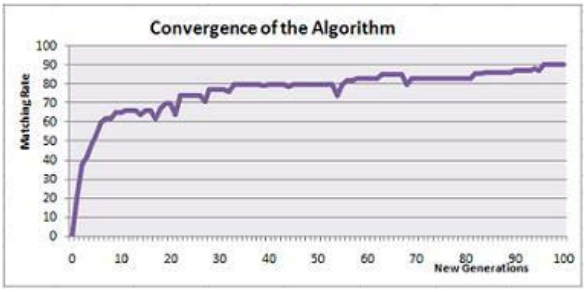

Fig. 8. Acceptable Matching Rate in CoCo-VM

Figure 8 shows the maximum border (zero means 100% matching and maximum refers to limit of acceptable mismatching) for most popular resources in CoCo-VM prototype system, where the minimum border is limited to zero. Results show that CoCo-VM has more flexibility in operation and can concentrate on the vectors with less degree of similarity as well as more accuracy compare to the PIS model.

4.5. Successful Service Rate (SSR)

Two important factors for resource assignment are stability of service-level and probability of successful service rate regardless of service population. Stability in supporting the services refers to capability of the algorithm to prevent any degradation in number of successful services especially in high service rate. Moreover, Successful service rate defines how an algorithm is effective to handle an entry services in the framework of the quality constraint. Selecting a virtual machine with enough resource fields in the JV vector such that, the selected resources can meet the predefined quality constraint is the policy of the proposed resource allocation algorithm. To confirm this advantage, two evaluation scenarios using PIS and CoCo-VM resource allocation algorithms have been implemented in a simulation environment. In the simulation system, any mismatching of the resource fields which causes over capacity (extra capacity) or under capacity (lack of enough capacity) assignment may cause degradation in successful service rate or affect the resource consumption level. This attribute in the algorithm improves efficiency using a modification procedure to adjust the proper resource assignment.

[3] Variance (δ)

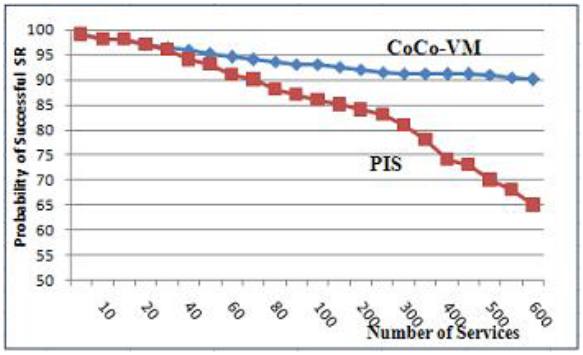

Fig.9. Successful Service Rate (SRS)

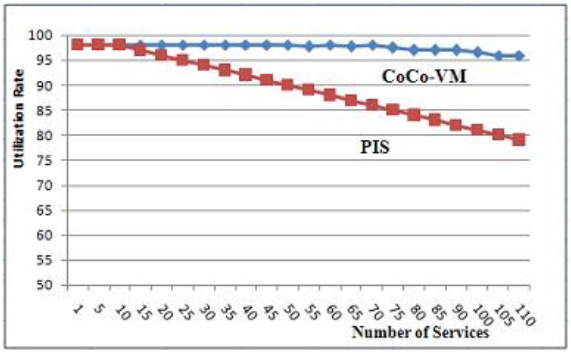

Fig.10. Resource utilization

Figure 9 compares probability of successful service rate in different number of services. As depicted in this figure, both algorithms in low service rate show similar results with high probability of success. But, when number of entry services is increases, probability of successful service rate will decreases gradually. Degradation in successful service rate for PIS model is faster than CoCo-VM method. This attribute will improve when number of service increase step by step. On the other hand, in CoCo-VM method, service degradation rate is more slowly compare to PIS model. This advantage is results of the greedy function of algorithm and the concept of threshold with an acceptable deviation rate instead of an exact matching function. Moreover capability of the algorithm for selecting the best matching for resources and using modification functions are other advantage of the proposed algorithm. Thus, the algorithm can improve successful service rate that is an important factor especially in high service rate.

4.6. Resource Utilization Rate (RUR)

In this part, we have compared resource utilization rate using PIS and CoCo-VM algorithms. In PIS model, assigning resources for virtual machines are limited to the available resources. There is not any alternative value for each field or any procedure for modification in the JV vector. On the contrary, in the proposed algorithm, the system will trace all the available fields in each vector to find the non-matching fields. In those fields, the available resources are not enough to comply with necessary resource. Then, the algorithm concentrates for finding other possible options or activating the resource modification procedure. Procedure will continue until the algorithm obtains an acceptable resource with suitable fitness value for the mismatching fields. This mechanism increases resource utilization level. It should be noted that this factor is more noticeable in high service rate. On the other hand, the existing physical hardware is an important factor for available physical resources and a bottleneck for virtual resource capacity. Based on results in figure 10, level of resource utilization rate in the proposed method is higher than PIS model. This advantage is due to greedy procedure of the algorithm, sticking for finding different alternatives and calling modification procedures. It should be noted that limitation in hardware resources is the main constraint for the algorithm and lack of available hardware can limit the volume of available virtual resources. On the other hand, the service level agreement is a key point for class of service and the scale of consuming resources.

4.7. System Response Time (SRT)

System delay is an important constraint for any computing machine. In our evaluation, response time is summation of training phase plus decision process. Training phase is a pre-operational period and prepares the system for actual operation. The resource assignment period should be negligible compared to the training phase. On the other hand, there is a compromise between accuracy and decision interval. It is obvious that any improvement in accuracy rate may influence the decision time interval. For analysis, we have compared two different scenarios of PIS system and CoCo-VM system which running multi agent algorithm with modification capability. Figure 11 compares the response time for both systems. As is shown in this figure, in a limited number of resource groups, PIS procedure starts with low response time. Along the process, by increasing number of resource population, the CoCo-VM response will increase smoothly while it increases more quickly in PIS system. This analysis confirms that even though the probability of successful rate and resource utilization has been increased in CoCo-VM method, the average response time is lower than PIS model. This advantage is more noticeable especially in high service rate. It should be noted that this advantage originate from capability of training procedure and efficient indexing in the proposed algorithm.

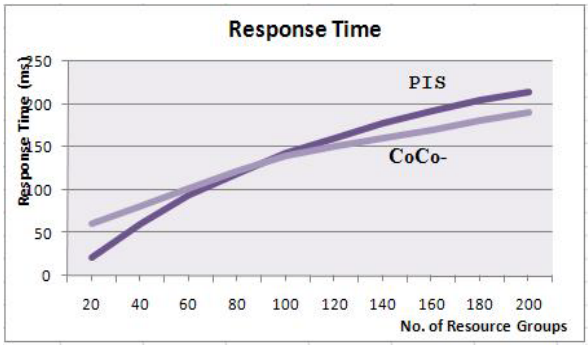

Fig.11. Comparison of average response time

4.8. Results Validation

In order to validate the evaluation results in the proposed system, we have investigated using predictive accuracy metric. Predictive accuracy and error rate are two important parameters for precise evaluation of the prototype systems which are obtained based on results of sec 4. In validation procedure, we have used 10-fold cross-validation technique. The technique involves randomly dividing the complete data set into 10 disjoint sets with equal size where we use one subset as a test set and the others as a training set. We have given predefined parameters for the random resources and execute the procedures for several consecutive trials. In initial process, the CoCo-VM uses a training data to learn and generate different resources for all JV vectors of virtual machines. Once the process has completed, the input services are applied to the testing data and the successful services are measured. In experiment, each class of resources has to recognize one field with specific service requirement (i.e. CPU, HD,…) and ignores other fields. The predictive accuracy is computed through repeating the process for consecutive trials. The CoCo-VM executes the procedures for 200 consecutive generations. Here, we execute the two selected systems for 200 trials and applied the testing data to calculate the average successful service rate. The final successful service for CoCo-VM and PIS method are summarized in table 1.

TABLE 1. PREDICTIVE ACCURACY COMPARISON FOR COCO-VM AND PIS METHODS

Assignment Method	Average Successful Rate
CoCo-VM	**0.93±0.625**
PIS Method	**0.85±0.363**

This table shows that probability of predictive accuracy value is 0.93 ±0.625 percent for CoCo-VM and 0.85 ±0.363 percent for PIS method. Results show that the CoCo-VM is able to achieve higher percentage of successful resource assignment compare to PIS system.

V. CONCLUSION & FUTURE DISCUSSION

This paper proposes a dynamic resource allocation method based on Cooperative Co-evolutionary Algorithm for virtual resource allocation in datacenter applications. The algorithm integrates an agent based greedy function based on immune system together with the cooperative co-evolutionary concept as a successful solution for virtual resource allocation in a large scale and distributed virtualized datacenter. Mathematical analyses and evaluation formula have been done to identify the parameters essential to assessing the proposed allocation approach. A prototype system has been developed in a simulation environment to compare the proposed method with standard version of the genetic algorithm. Results of different evaluations in a simulation environment demonstrate that the proposed approach can significantly reduce resource consumption, increase service capability, and achieve higher performance. In continue, there is an opportunity to compare effectiveness of the proposed algorithm with other non-intelligent method using compatibility matching technique between virtual and real environment.

REFERENCES

[1] Thomas C. Perry, Joseph C. Hartman, "An approximate dynamic programming approach to solving a dynamic, stochastic multiple knapsack problem" International Transactions in Operational Research, Vol.16, No. 3, 2009, pp.347–359.

[2] P. Xiong, Z. Wang, G. Jung, C. Pu, “Study on performance management and application behavior in virtualized environment”, IEEE Network Operations and Management NOMS 2010, April 2010, ISSN1542-1201, pp. 841-844 .

[3] E. Kalyvianaki, T. Charalambous and S. Hand : "Resource Provisioning for Multi-Tier Virtualized Server Applications", Computer Measurement Group Journal, Spring 2010.

[4] V. Anitha and N. Keerthana, “An Overview: Security in Virtualization Technology”. *Journal of Artificial Intelligence, 6: 112-116, 2013.*

[5] Ahmad N. Quttou**m,** “A collusion-resistant mechanism for autonomic resource management in Virtual Private Networks”, Computer Communication Journal, Vol. 33, No. 17, 2010.

[6] Gast N Keller, H. Lutfiyya, “Dynamic Resource Management in Virtualized Environments through Virtual Server Relocation”, International Journal on Advances in Software, 2010.

[7] H. Liu, H. Jin, X.Liao, C. Yu and C. Zhong Xu,,"Live Virtual Machine Migration via Asynchronous Replication and State Synchronization", IEEE Transactions on Parallel and Distributed Systems, 2010.

[8] YangSun Lee and YunSik Son, “A Study on the Smart Virtual Machine for Executing Virtual Machine Codes on Smart Platforms”, International Journal of Smart Home , Vol.6, No.4, October , 2012.

[9] D. Bertsimas and R. Demir.,”An approximate dynamic programming approach to multidimensional knapsack problems”, Manage. Sci., Vol.48, No.4, 2002, pp. 550–565.

[10] P. C. Chu and J. E. Beasley,” A genetic algorithm for the multidimensional knapsack problem”, Journal of Heuristics, Vol.4, No.1, 1998, pp. 63–86.

[11] Gast N Keller, Hanan Lutfiyya, “Dynamic Resource Management in Virtualized Environments through Virtual Server Relocation”, International Journal on Advances in Software, Vol. 3, No. 3, 2010. pp. 333-350.

[12] T. Lust, J. Teghem, “The multi-objective multidimensional knapsack problem: a survey and a new approach International Transactions in Operational Research”, Volume 19, Issue 4, pages 495–520, July 2012.

[13] D. F. Ciocan and, V. Farias, “Model Predictive Control for Dynamic Resource Allocation” *Mathematics of Operations Research* August 2012, 37:501.

[14] I. Aydin, M. Karakose, E. Akin, “A multi-objective artificial immune algorithm for parameter optimization in support vector machine”, Applied Soft Computing Journal, Volume 11, Issue 1, January 2011, Pages 120–129.

[15] M. R. Ahmadi and D. Maleki, “An Intrusion Detection Technique Using Co-Co Immune System for Distributed Networks (CoCo-ISD)’’ International Journal of Computer

Science and Network Security, Vol.8, No.4, April 2008, pp.160-169.

[16] HP-UX Workload Manager, http://docs.hp.com/en/5990-8153/ch05s12.html

[17] J. Rolia "Configuring Workload Manager Control Parameters for Resource Pools," 10th IEEE/IFIP Network Operations and Management Symposium, 2006.

[18] D. A. Menasce and M. N. Bennani, "Autonomic virtualized environments" In ICAS '06: Proceedings of the International Conference on Autonomic and Autonomous Systems, IEEE Computer Society, 2006.

[19] M. N. Bennani and D. A. Menasce, "Resource allocation for autonomic data centers uses analytic performance models," In ICAC '05: Proceedings of the Second International Conference on Automatic Computing, IEEE Computer Society, 2005.

Mohammad Reza Ahmadi received his B.Sc. and M.Sc. degrees in Electrical Engineering and Communication Systems from K.N.T. University of Technology in 1986 and 1990 respectively. He received his Ph.D. degree in Communication Networks from Tokyo Institute of Technology, Tokyo, in 1997. Currently he is a project manager and researcher in IT department of Cyber Space Research Institute. His research interests are resource optimization in data centers, virtualization and cloud computing techniques, network security focus on intrusion detection systems.