Handling Uncertainty in Cloud Resource Management Using Fuzzy Bayesian Networks

Fahimeh Ramezani, Mohsen Naderpour, Jie Lu
Decision Systems and e-Service Intelligence (DeSI) Laboratory
Centre for Quantum Computation & Intelligent Systems (QCIS)
Faculty of Engineering and Information Technology, University of Technology Sydney (UTS)
PO Box 123, Broadway NSW 2007 Australia
{Fahimeh.Ramezani, Mohsen.Naderpour, Jie.Lu}@uts.edu.au

***Abstract*—The success of cloud services depends critically on the effective management of virtualized resources. This paper aims to design and implement a decision support method to handle uncertainties in resource management from the cloud provider perspective that enables underlying complexity, automates resource provisioning and controls client-perceived quality of service. The paper includes a probabilistic decision making module that relies upon a fuzzy Bayesian network to determine the current situation status of a cloud infrastructure, including physical and virtual machines, and predicts the near future state, that will help the hypervisor to migrate or expand the VMs to reduce execution time and meet quality of service requirements. First, the framework of resource management is presented. Second, the decision making module is developed. Lastly, a series of experiments to investigate the performance of the proposed module is implemented. Experiments reveal the efficiency of the module prototype.**

Keywords—Cloud computing, Fuzzy Bayesian networks, Resource management.

I. INTRODUCTION

Cloud systems have recently emerged as a new paradigm for the provision of computing infrastructure for a wealth of applications. They are largely being used to deliver services through the Internet for mainly economic and technical reasons. Thus, providers of software-based services, denoted as service providers, are freed from the burden of setting up and managing the hardware and/or software platforms required by their services. These resources are now provisioned by cloud platforms administrated by cloud providers[1].

Cloud systems are classified by the type of resources they offer: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). IaaS systems have arguably raised the greatest interest so far. These large scale services can be provided by applying shared virtualized cloud resources. In general, the cloud resources are provided as a collection of several proprietary processes in a virtual environment, called a virtual machine (VM). Fig. 1 shows the cloud architecture [2].

The recent surge in the popularity and usage of cloud computing services by both enterprise and individual consumers has necessitated the efficient and proactive management of data center resources that host services with a variety of characteristics. One of the major issues concerning both cloud service providers and consumers is real time resource management in response to highly unpredictable demands [3]. These unpredictable demands cause the workload of VMs to fluctuate dynamically, leading to imbalance in both the load and utilization of virtual and physical cloud resources. In light of these uncertainties, therefore, cloud providers need a decision support that will dynamically scale up and scale down resources without violating Service Level Agreements (SLA) while simultaneously ensuring adequate revenue.

Probability theory is the method of choice for dealing with uncertainty in many science and engineering disciplines. As a probability theory tool, the Bayesian Network (BN) model of a system is the compact representation of a joint probability distribution of the variables comprising the system [4]. In this sense, BNs can be used for building representative models of cloud resource management systems. This paper develops a BN-based predictive module to provide decision support for the virtualization layer to handle uncertainties in scaling utility computing resources up or down based on SLA parameters. As the module contains continuous variables such as CPU and

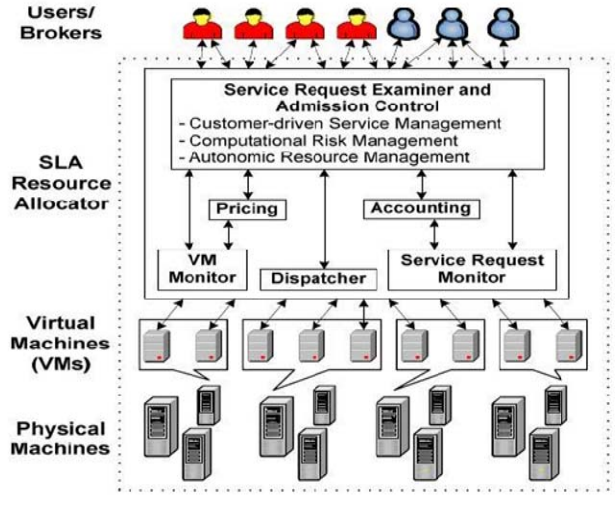
Users/ Brokers
Service Request Examiner and Admission Control
- Customer-driven Service Management
- Computational Risk Management
- Autonomic Resource Management
SLA Resource Allocator
Pricing
Accounting
VM Monitor
Dispatcher
Service Request Monitor
Virtual Machines (VMs)
Physical Machines

Fig.1. Cloud architecture.

memory utilizations, therefore, fuzzy BNs are relied on to complement probability theory with fuzzy sets to perform exact inferencing.

The paper is organized as follows: Section II describes the background and previous works related to this research. Section III develops the decision maker module, and is followed by experimental results in Section IV. The conclusion and future work are presented in Section V.

II. Background

A. Cloud computing resource management

Cloud computing has become increasingly popular in large-scale computing and data storage due to its ability to enable the sharing of computing resources that are distributed all over the world. In a cloud environment, the cloud provider uses virtualization technology to integrate dynamically scalable resources such as physical machines (PMs), data storage, and networks. In fact, virtualization technology is the backbone of the provisioning requirements of a cloud-based solution. A VM monitor or hypervisor manages and multiplexes access to the physical resources, maintaining isolation between VMs at all times. A VM, like a PM, has associated resource levels of CPU, memory, and input/output (I/O) devices, and each VM needs to be provisioned with an allocation of these resources. Resources can be overcommitted, however, and multiplexing them across VMs is the hypervisor's responsibility. As the physical resources are virtualized, several VMs, each of which is self-contained with its own operating system, can execute on a PM. The hypervisor, which arbitrates access to physical resources, can manipulate the extent of access to a resource (memory or CPU allocation to a VM, for example). The decoupling between physical and virtual resources provided by the hypervisor enables flexibility of resource provisioning for VMs. This decoupling also yields efficient state (more correctly memory state) capture of VMs, which enables the migration and restoration of VMs across PMs [5].

The cloud provider constructs VMs, which need to be rapidly elastic on demand. A cloud user outsources jobs to the cloud provider and only pays according to service usage. The cloud provider negotiates a SLA with the cloud user which covers the properties of cloud resources and the cloud user's requirements, and guarantees quality of service (QoS). SLAs are used to ensure that service quality is sustained and includes QoS parameters such as response time, reliability, and availability. When a job requires processing, a cloud provider constructs the necessary VMs, called a resource set, and provides them as a service to the cloud user. The "sizing" process, based on the resource-usage profiles of applications or estimates to meet load requirement and QoS characteristics, is used to determine the initial provisioning-levels of a VM to meet the agreed SLA [6]. Changes in the workload conditions of VMs can lead to "hot spots" — insufficient resources provisioned to meet demand —or "cold spots" — provisioned resources are not utilized efficiently [5]. The cloud provider needs to dynamically allocate jobs to a specific set of VMs in order to meet the SLA and address hot or cold spot situations.

Efficiently allocating scarce resources on a large scale between competing interests in decentralized cloud computing networks is a challenge for resource managers [7]. Virtualization technology has improved utilization and system load balancing by enabling VM migration in two ways (1) suspend /resume and (2) live migration. The suspend/resume VM migration approach has three steps: pause the original VM, copy the VM's related data (memory pages and processor state) to a new host PM, and eventually resume the VM on the destination host [8-10]. In this method, applications running on the VM are stopped and are then unavailable until the migration process has been completed and all data transferred to the new destination. Because of the long VM downtime in the suspend /resume method, live migration applied by such tools such as VMotion [11] and Xen [12] is proposed to reduce the downtime. In this way, the VM's run-time memory state files are pre-copied (migrated) from the source host to the destination host while the VM is still working. The framework of resource management is illustrated in Fig. 2, and as can be seen, includes two major modules:

- *Resource Monitor*: is responsible for monitoring the resource status of both VMs and PMs, including resource utilization, VM configuration information (workload characteristics, VCPU number, memory size, and image size).
- *Decision-Maker*: is a key module in the framework of resource management. The module has intelligence to predict and make efficient strategies such as migration, or to expand strategies. The hypervisor relies upon the decision maker module and executes the strategies it determines. If the strategy is migration, the hypervisor chooses the right target PM from the candidate PMs.

Various approaches based on machine learning techniques have been developed to address resource management issues. Gong et al. utilized short-term signal processing techniques and discrete-time Markov chain for long-term resource demand predictions [13]. Bayesian Network (BN) models were used by Qiang et al. [14] to characterize the normal states of cloud computing systems and to detect failures in an unsupervised learning manner, and by Bashar [3] to scale cloud resources up and down in a supervised learning manner. On-demand resource forecasting based on pattern matching was developed by Caron et al. [15]. A cloud resource scaling system was also proposed to make scaling decisions [16]. A cloud resource prediction and provisioning scheme to predict future demand

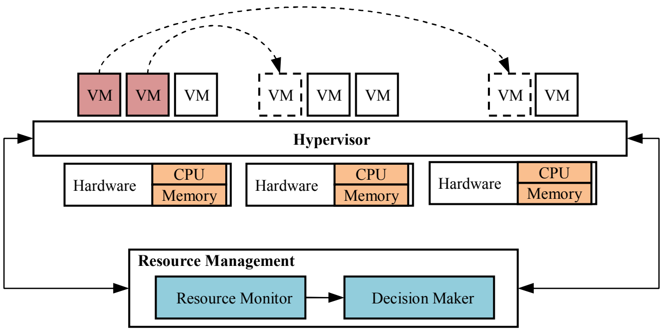

Fig. 2. The resource management framework.

and perform proactive resource provisioning for cloud applications has been suggested by Fang et al. [17].

B. Fuzzy systems

Fuzzy logic is a concept for dealing with uncertainty, vagueness, or imprecise problems that uses membership functions with values between 0 and 1. Unlike conventional set theory based on Boolean logic, a particular object or variable in fuzzy set theory based on fuzzy logic has a degree of membership in a given set that may be anywhere in the range of 0 (completely not in the set) to 1 (completely in the set).

Definition 1 (Fuzzy set)[18]: Fuzzy set A is defined in terms of a universal set X by a membership function that assigns to each element $x \in X$ a value $\mu_A(x)$ in the interval [0,1], i.e. $A: X \rightarrow [0,1]$.

Definition 2 (α-cut) [18]: Let A be a fuzzy set in the universe X, $\alpha \in (0,1]$. The α-cut or α-level set of the fuzzy set A is the crisp set A_α defined by:

$$A_\alpha = \{\, x \in X | \mu_A(x) \geq \alpha\} \tag{1}$$

Definition 3 (Fuzzy number) [18]: A fuzzy set A in $\mathbb{R}$ satisfies the following conditions:

- A is normal,
- A_α is a closed interval for every $\alpha \in (0,1]$,
- the support of A is bounded.

Definition 4 (Fuzzy random variable): Let $(\Omega, \mathcal{F}, \mathrm{P})$ be a probability space, $\mathrm{F}(\mathbb{R})$ the set of fuzzy numbers in $\mathbb{R}$ with compact supports and W is a mapping $\Omega \rightarrow \mathrm{F}(\mathbb{R})$. Then W is a fuzzy random variable if and only if given $\omega \in \Omega$, $\mathrm{W}_\alpha(\omega)$ is a random interval for any $\alpha \in [0,1]$ where $\mathrm{W}_\alpha(\omega)$ is an α-level set of the fuzzy set W(ω) [19].

Note: A subset of Euclidean space $\mathbb{R}^\mathrm{n}$ is called compact if it is closed and bounded. For example, in $\mathbb{R}$, the closed unit interval [0,1] is compact.

Definition 5 (Fuzzy state): Let the crisp state set $\mathcal{P}$ consist of the states $p_1, p_2, \ldots, p_n$. Then, each *fuzzy state* can be written as a vector $q = [q_1, q_2, \ldots, q_n]$, where $q_i \in [0,1]$. This way, each fuzzy state can be considered as a possibility distribution or alternatively as a fuzzy set $q \in \mathcal{F}(\mathcal{P})$, ($\mathcal{F}(\mathcal{P})$ the set of all fuzzy subsets defined for $\mathcal{P}$) determining the degree q_i by which the system participates in each crisp state p_i, provided it is in the current fuzzy state q [20].

C. Bayesian networks

A BN is defined as a couple: $\mathcal{G} = ((N, A), \mathcal{P})$, where (*N,A*) represents the graph; *N* is a set of nodes; *A* is a set of arcs; $\mathcal{P}$ represents the set of probability distributions that are associated with each node. When a node is not a root node, the distribution is a conditional probability distribution that quantifies the probabilistic dependency between that node and its parents [21]. A discrete random variable *X* is represented by a node $n \in N$ with a finite number of mutually exclusive states. States are defined on $S_n\colon \{s_1^n, s_2^n, \ldots, s_M^n\}$. The set $\mathcal{P}$ is represented with Conditional Probability Tables (CPT), and each node has an associated CPT. For instance, if n_i is a parent of n_j, and the nodes n_i and n_j are defined over the sets $S_{n_i}\colon \{s_1^{n_i}, s_2^{n_i}, \ldots, s_M^{n_i}\}$ and $S_{n_j}\colon \{s_1^{n_j}, s_2^{n_j}, \ldots, s_L^{n_j}\}$, the CPT of n_j is then defined as a matrix by the conditional probabilities $P(n_j | n_j)$ over each n_j state knowing its parents states (n_i) [22]:

$$P(n_j | pa(n_j)) = \begin{bmatrix} p(n_j = s_1^{n_j} | n_i = s_1^{n_i}) & \ldots & p(n_j = s_L^{n_j} | n_i = s_1^{n_i}) \\ \vdots & \vdots & \vdots \\ p(n_j = s_1^{n_j} | n_i = s_M^{n_i}) & \cdots & p(n_j = s_L^{n_j} | n_i = s_M^{n_i}) \end{bmatrix} \tag{2}$$

Various inference algorithms can be used to compute marginal probabilities for each unobserved node, given information on the states of a set of observed nodes; the junction tree algorithm is a classic example of such an algorithm. Inference in BN then allows us to take into account any state variable observation (an event) to update the probabilities of the other variables. When observations are given, this knowledge is integrated into the network and all the probabilities are updated accordingly. Hard evidence of the random variable *X* indicates that the state of the node $n \in N$ is one of the states $S_n\colon \{s_1^n, s_2^n, \ldots, s_M^n\}$. Nevertheless, when this knowledge is uncertain, soft evidence can be used. Soft evidence for a node *n* is defined as evidence that enables the updating of the prior probability values for the states of *n* [23].

A dynamic BN (DBN) is a BN that includes a temporal dimension. This new dimension is managed by time-indexed random variables X_i, which are represented at time step *k* by a node $n_{(i,k)} \in N$ with a finite number of states $S_{n_i}\colon \{s_1^{n_i}, s_2^{n_i}, \ldots, s_M^{n_i}\}$. Several time stages are represented by several sets of nodes and an arc that links two variables belonging to different time slices represents a temporal probabilistic dependence between these variables. DBNs then allow us to model random variables and their impact on the future distribution of other variables. Defining this impact as transition probabilities between the states of the variable at time step *k-1* and those at time step *k* leads to the definition of CPTs that are relative to inter-time slices. With this model, the future slice (*k*) is conditionally independent of the past given the present (*k-1*) [24, 25].

III. The Decision Maker Module

In the imbalanced load situation, additional resources (CPU/memory) will be allocated to overloaded VMs to expand them and enhance their performance, if the original host PM has idle resources available. Otherwise, these VMs will be migrated to another PM using suspend/resume or live migration strategy for stronger computation power, larger memory, fast communication capability, or energy savings [26]. If an overloaded VM is large, it would be reasonable to migrate it to a new physical host. However, VM migration is not an optimal solution when the overloaded VM is large. This results in dirty memory due to the pre-copy process, utilizes a large amount of memory in the primary and new host PM, causes the VM to slow down during migration process, carries the risk of losing the most recent customer activities, and is cost- and time-consuming. The resume/suspend migration strategy not only has live migration shortcomings, but also causes a long VM downtime. Taking the above description into

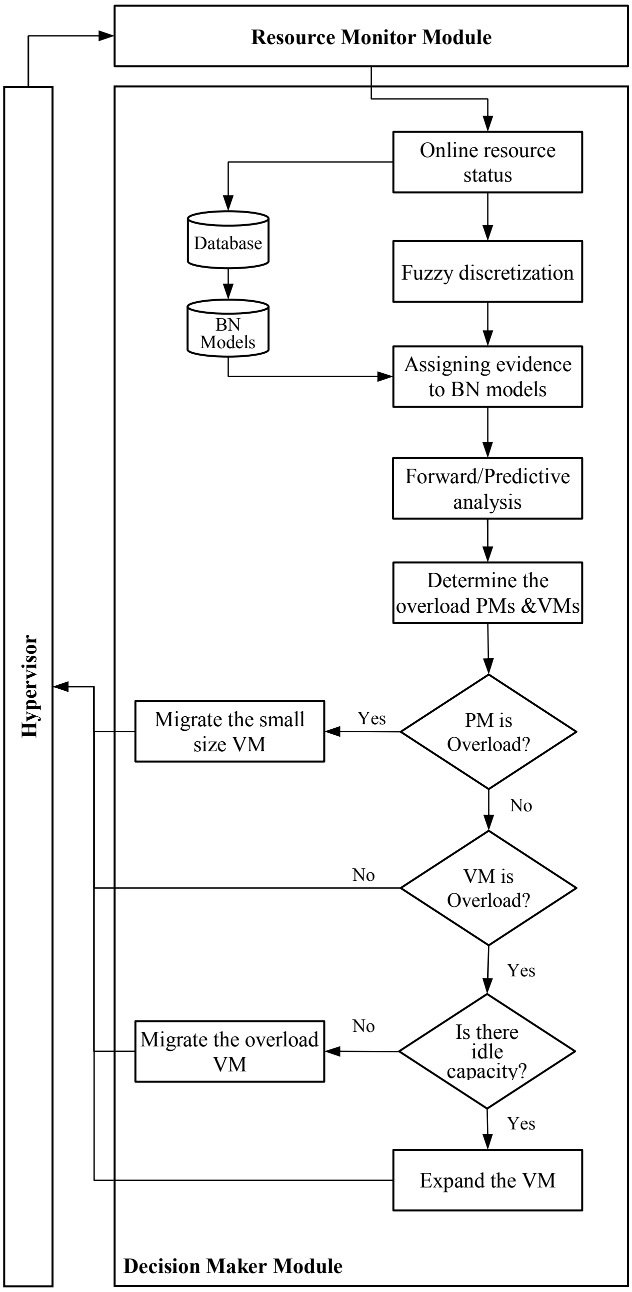

Fig. 3. Structure of the decision maker module.

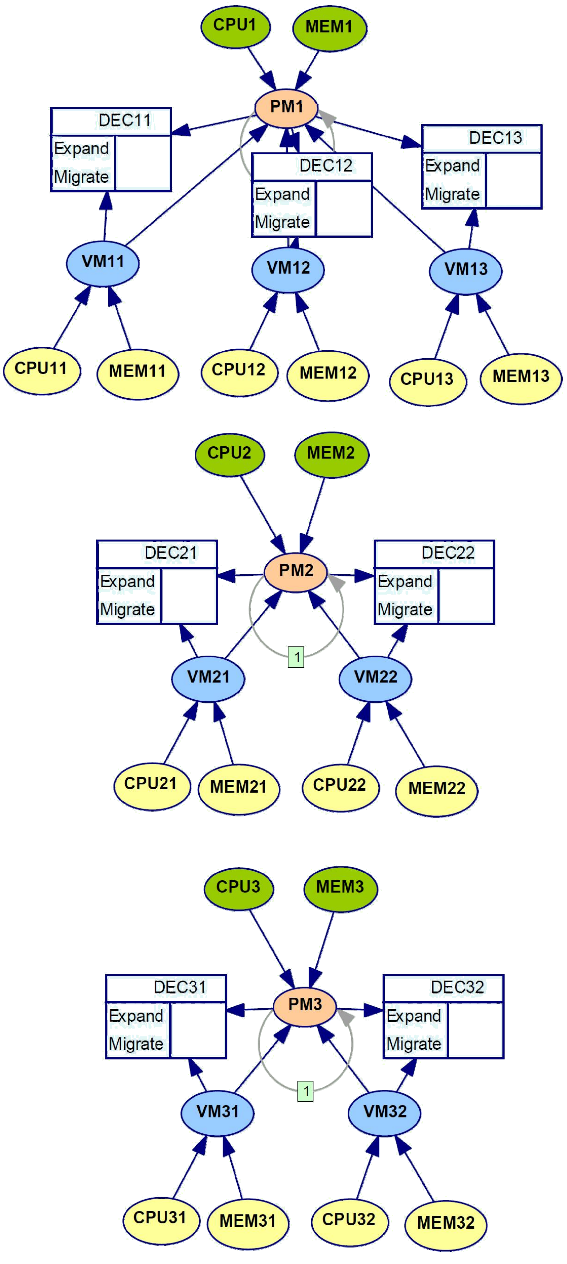

Fig. 4. Three DBN models.

account, the structure of the decision maker module is presented in Fig. 3.

A. BN models

The DBN models based on relationships between VMs and PMs are developed. For instance, three DBN models of three PMs and related VMs are illustrated in Fig. 4. The states of the VM and PM nodes are defined as Underload and Overload. The states of the resource variables, i.e. CPU and Memory, are determined as Low and High. The CPTs of the VM and PM nodes are set based on the "OR gate" or "AND gate" definitions. The temporal arcs are represented by connecting the PM nodes to themselves. The interpretation is that the PM is Overload when the status persists for a short time. The decision nodes are added for every VM containing two states: Expand and Migrate. The former refers to the expansion strategy, i.e. changing CPU share or memory allocation at the current PM, and the former indicates the migration of the VM to other physical nodes.

B. BN model evaluations

Development of the proposed decision maker module is dependent on the BN models. The evaluation of BN models can therefore be investigated by sensitivity analysis, according to the following three axioms [18]:

- A slight decrease/increase in the prior probabilities of each parent node should result in a relative decrease/increase of the posterior probabilities of the child node.
- Given the variation in the subjective probability distributions of each parent node, the magnitude of influence of the parent node on the values of the child node should remain consistent.
- The magnitude of the total influence of the combination of probability variations from x attributes (evidence) on the values should be always greater than the probability variations from the set of x-y (y ∈ x) attributes (sub-evidence).

To validate the proposed DBN models, the parameters used need to be closely monitored over a long period, therefore the above axioms are useful for partial validation.

C. Continuous variable discretization

As the online resource variables are continuous, a discretization process is required to use them in BNs. In general, mapping a continuous variable to a discrete variable can be achieved with a crisp set or a fuzzy set. However, fuzzy sets provide a smooth option [18]. Now suppose $X = \{X_1, X_2, \dots, X_n\}$ is the set of variables in a BN. If the variable X_i is a continuous variable then it can be transformed into a fuzzy random variable W_i. The corresponding set U_i can be utilized to map the variable X_i to fuzzy states:

$$U_i = \{\hat{X}_{i1}, \hat{X}_{i2}, \dots, \hat{X}_{im}\} \tag{3}$$

where $\hat{X}_{ij}$ is the *j*-th fuzzy state and *m* denotes the number of the fuzzy states, and fuzzy state $\hat{X}_{ij}$ can be defined as follows:

$$\hat{X}_{ij} = \{\mu_{\hat{X}_{ij}}(x) | x \in \boldsymbol{X}_i\} \tag{4}$$

where $\mu_{\hat{X}_{ij}}(x)$ is the membership function of fuzzy state $\hat{X}_{ij}$, $\boldsymbol{X}_i$ is the frame of X_i, and x denotes the value of variable X_i. For a continuous variable X_i, its condition probability in the BN with its parent can be replaced by $P(W_i|Pa(W_i))$:

$$P(X_i|Pa(X_i)) \rightarrow P(W_i|Pa(W_i)) \tag{5}$$

where W_i is the corresponding fuzzy random variable defined by X_i. For a node without parents, soft evidence is equivalent to modifying its prior probability; otherwise, soft evidence on a variable X_i is represented by a conditional probability vector $P(X_i = x|H_i)$ for *i=1,2,...,m*, where H_i denotes the hypothesis that the true state is the *i*-th state. To simplify the inference process for continuous variables, consider the fuzzy random variable W_i with states $\{\hat{X}_{i1}, \hat{X}_{i2}, \dots, \hat{X}_{im}\}$. Define H_j, *j=1,2,...,m* as hypotheses that W_i is in state $\hat{X}_{ij}$. The results of fuzzy function member $\mu_{\hat{X}_{ij}}(x)$ *j=1,2,...,m* form the soft evidence vector:

$$e = \{\mu_{\hat{X}_{i1}}(x), \mu_{\hat{X}_{i2}}(x), \dots, \mu_{\hat{X}_{im}}(x)\} \tag{6}$$

The $\mu_{\hat{X}_{ij}}(x)$ is approximately considered to be equivalent to the condition probability $P(\mu_{\hat{X}_{ij}}|X_i = x)$ Then the soft evidence vector can be defined as:

$$e = \{P(W_i = 1|H_1), P(W_i = 1|H_2), \dots, P(W_i = 1|H_m)\} \tag{7}$$

where $P(W_i = 1|H_j)$ represents that the observed value of W_i is "1" if the state is $\hat{X}_{ij}$, which is indeed the probability $P(\mu_{\hat{X}_{ij}}|X_i = x)$.

The shapes of the membership functions are defined as a combination of trapezoidal and triangular numbers to simplify the operation and increase sensitivity in a number of bounds. For example, the CPU utilization [%] in terms of operation can be partitioned into fuzzy states Low and High. Its membership function is defined as follows, as well as being shown in Fig. 5:

$$\mu_{CPU(L)}(x) = \begin{cases} 1 & 0 \le x \le 50 \\ (80 - x)/30 & 50 < x \le 80 \\ 0 & x > 80 \end{cases} \tag{8}$$

$$\mu_{CPU(H)}(x) = \begin{cases} 0 & 0 \le x < 50 \\ (x - 50)/30 & 50 \le x < 80 \\ 1 & x \ge 80 \end{cases} \tag{9}$$

D. Predictive analysis

In the forward method or predictive analysis, the workload probability of any VM or PM is calculated on the basis of their prior probabilities and online resource states called evidence *E*, thus yielding the posteriors. This new information usually becomes available during the operational life of the cluster:

$$P(X|E) = \frac{P(X,E)}{P(E)} = \frac{P(X,E)}{\sum_X P(X,E)} \tag{10}$$

The static BN models are extended to DBN models by introducing relevant temporal dependencies that capture the dynamic behaviors of the cluster variables between representations of the static network at different times. Two types of dependency are distinguished in the DBNs: contemporaneous and non-contemporaneous.

Contemporaneous dependencies refer to arcs between nodes that represent variables within the same time period, e.g. arcs between one PM and corresponding VMs. Non-contemporaneous dependencies refer to arcs between nodes which represent variables at different times, e.g. arcs between one PM and itself. The DBN models are defined as a pair $(\mathrm{B_1}, \mathrm{2TBN})$ where B1 is a BN which defines the prior distribution $\mathrm{P(X_1)}$ and 2TBN is a two-slice temporal BN with

$$\mathrm{P(X_t|X_{t-1})} = \prod_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{P}\left(\mathrm{X_t^i|Pa}\left(\mathrm{X_t^i}\right)\right) \tag{11}$$

where $\mathrm{X_t^i}$ is a node at time slice t and $\mathrm{Pa}\left(\mathrm{X_t^i}\right)$ is the set of parent nodes which can be in time slice *t* or in time slice *t*-1. The nodes in the first slice of a 2TBN have no parameters associated with them, but each node in the second slice has an associated conditional probability table for discrete variables, which defines $\mathrm{P}\left(\mathrm{X_t^i|Pa}\left(\mathrm{X_t^i}\right)\right)$ for all $\mathrm{t} > 1$. The semantics of the DBN models are defined by "unrolling" the 2TBN until there are T time-slices. The resulting joint distribution is then given by [24]:

$$P(X_{1:T}) = \prod_{t=1}^{T}\prod_{i=1}^{n} P\left(X_t^i | Pa(X_t^i)\right) \tag{12}$$

IV. EXPERIMENTAL EVALUATION

To implement the decision maker module, a cloud simulation environment called CloudSim toolkit [27] and a library of C++ classes named SMILE (Structural Modeling, Inference, and Learning Engine) [28] are utilized.

A. *Experiment setup*

CloudSim, which allows the extension and definition of policies in all the components of the software stack, is a customizable research tool that can handle the complexities arising from simulated environments. SMILE is used to implement BNs in intelligent systems. The experimental setup is composed of three PMs that run seven VMs in total. The properties of PMs and VMs are summarized in Tables 1 and 2 respectively.

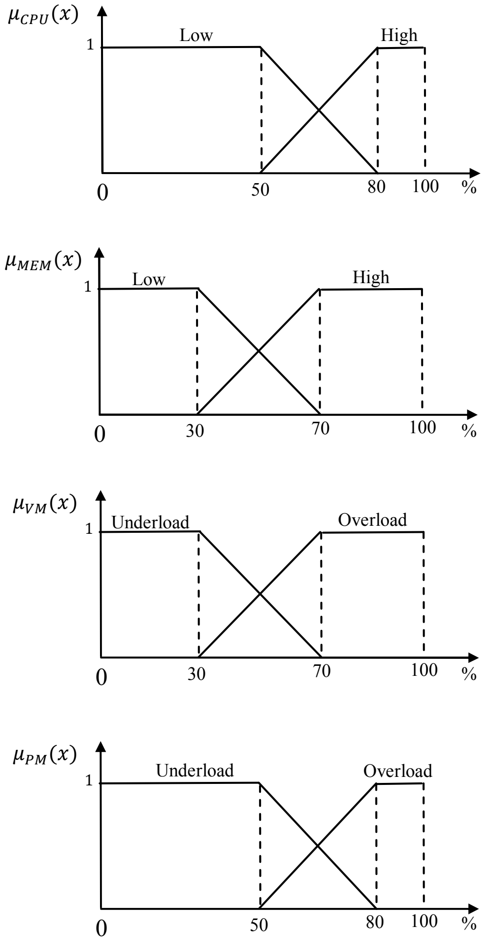

Fig. 5. The membership functions of variables.

TABLE I. PM PROPERTIES

Host ID	MIPS	Memory (MB)	OS	VMM
1	1000	2048	Linux	Xen
2	1000	2048	Linux	Xen
3	1000	2048	Linux	Xen

TABLE II. VM PROPERTIES

PM ID	VM ID	MIPS	VM image size	VM memory (Ram)	Bandwidth	Number of CPUs
1	11	250	1000	512	1000	1
1	12	300	1000	256	1000	1
1	13	250	1000	512	1000	1
2	21	250	1000	512	1000	1
2	22	250	1000	512	1000	1
3	31	300	1000	256	1000	1
3	32	250	1000	512	1000	1

B. *Analysis and results*

The prior probability of the PMs is set to 1 for the Underload state and 0 for the Overload state, and it is assumed that none of the PMs is initially Overload. The time interval is set to three minutes, and a scenario which contains sixty time intervals is considered. The use of resources is randomly produced and assigned to the DBN models. For instance, Fig. 6 illustrates the monitored CPU utilization of each VM of PM1 over a 180-minute period. Each VM exhibits a time-varying utilization pattern with interspersed peaks. The figure also

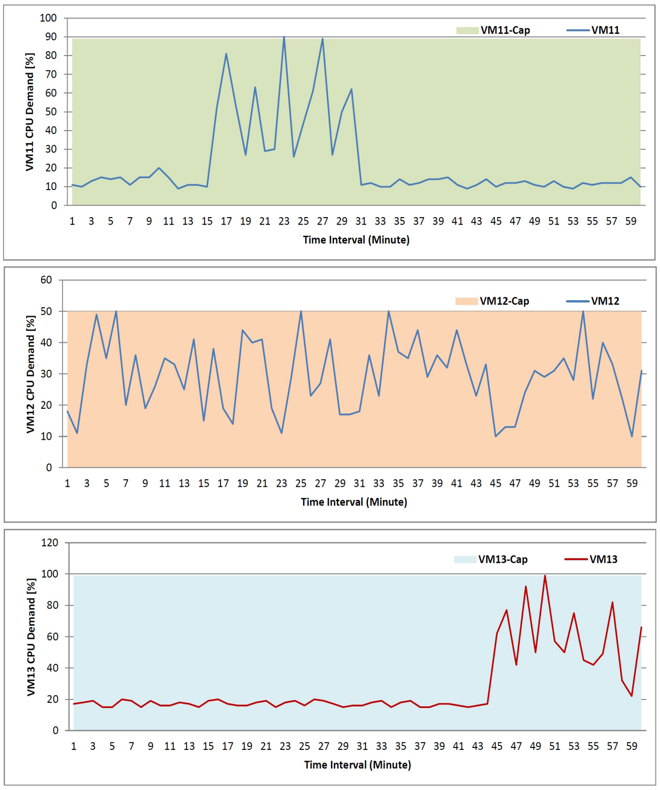

Fig. 6. The VM capacity requirements of PM1.

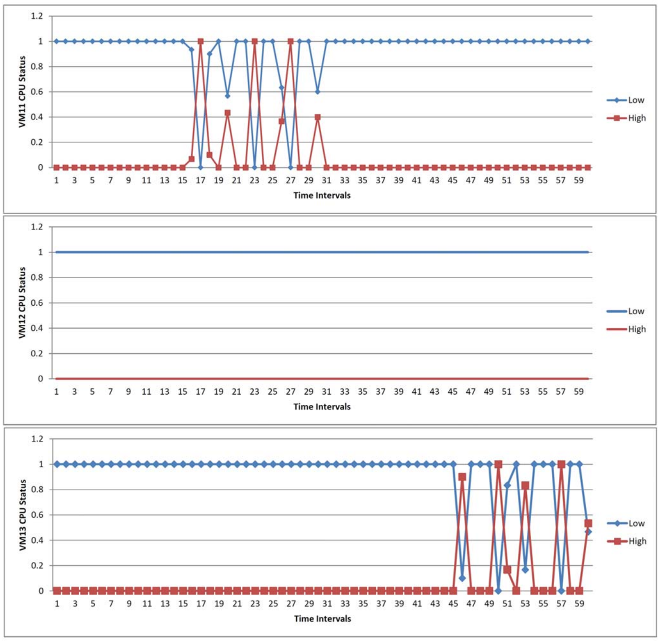

Fig. 7. The fuzzy states of VM CPUs.

depicts a simple capacity bound required by each VM that is based on conservatively satisfying each instantaneous peak, i.e., the capacity required for each VM is set as the maximum demand observed historically.

By assigning the fuzzy partitioning values of observable variables to the DBN models, the posterior probabilities of the VMs and PMs are calculated. For example, Fig. 7 shows the fuzzy states of VM CPUs, and Fig. 8 presents the VM posterior probabilities of PM1. As can be seen, there is a sharp increase in the overloading probability of VM11 at interval 13. In addition the posterior probability of PM1 shows that it is not overloaded in this interval. Therefore, the decision maker module takes the advantage of the posterior probabilities and decides to expand the VM11 as there is idle resource in PM1. At interval 27, VM13 is overloaded and PM1 is also overloaded, so the decision maker module decides to migrate VM12 because it is the smallest in this PM. There is no idle resource for its expansion, therefore migrating it will assist in meeting the terms of the SLA.

V. Conclusion and Future Work

Virtualization technology is used to increase resource utilization and reduce operating costs in IaaS clouds. Two key mechanisms for flexible resource utilization offered by virtualization are: a) allocating resources dynamically to virtual machines (i.e., changing CPU share or memory allocation), and b) migrating virtual machines to other physical nodes. However, with this flexibility comes uncertainty in managing resources. Consequently, this paper presents a cloud resource management framework which has two modules: Resource monitor and Decision maker. A structure for the decision maker module is proposed, and fuzzy Bayesian networks are implemented to handle some of the uncertainty. Lastly, the theoretical approach is adopted in a cloud simulation environment to investigate its performance.

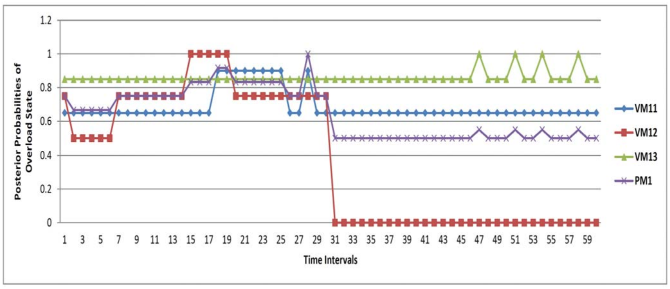

Fig. 8. The posterior probabilites of VM workloads in PM1.

As illustrated in the structure of the decision maker module, the database stores the status of cloud resources over time, therefore, the first direction of our future work will be to consider the use of stored data in the training of the DBN models. The second direction will be to improve the decision maker module to determine a new host for the required VM migration.

Acknowledgment

The work presented in this paper was supported by the Australian Research Council (ARC) under Discovery Project DP140101366.

References

[1] L. Rodero-Merino, L. M. Vaquero, V. Gil, F. Galán, J. Fontán, R. S. Montero, et al., "From infrastructure delivery to service management in clouds," Future Generation Computer Systems, vol. 26, pp. 1226-1240, 2010.

[2] F. Ramezani, L. Jie, and F. Hussain, "An online fuzzy decision support system for resource management in cloud environments," presented at the IFSA World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS), Edmonton, Canada, 2013.

[3] A. Bashar, "Autonomic scaling of cloud computing resources using BN-based prediction models," in 2nd IEEE International Conference on Cloud Networking (CloudNet), San Francisco, CA, 2013, pp. 200-204.

[4] A. Oztekin, "A generalized hybrid fuzzy-Bayesian methodology for modeling complex uncertainty," PhD thesis, The State University of New Jersey, New Jersey, 2009.

[5] M. Mishra, A. Das, P. Kulkarni, and A. Sahoo, "Dynamic resource management using virtual machine migrations," IEEE Communications Magazine vol. 50, pp. 34-40, 2012.

[6] Y. Ma, S. Jang, and J. Lee, "Ontology-based resource management for cloud computing," presented at the Third International Conference in Intelligent Information and Database Systems:, Daegu, Korea, 2011.

[7] G. Wei, A.V.Vasilakos, Y. Zheng, and N. Xiong, "A game-theoretic method of fair resource allocation for cloud computing services," J Supercomput, vol. 54, pp. 252-269, 2010.

[8] M. Kozuch and M. Satyanarayanan, "Internet suspend/resume," in Proceedings Fourth IEEE Workshop on Mobile Computing Systems and Applications, 2002., 2002, pp. 40-46.

[9] C. P. Sapuntzakis, R. Chandra, B. Pfaff, J. Chow, M. S. Lam, and M. Rosenblum, "Optimizing the migration of virtual computers," ACM SIGOPS Operating Systems Review, vol. 36, pp. 377-390, 2002.

[10] A. Whitaker, R. S. Cox, M. Shaw, and S. D. Gribble, "Constructing services with interposable virtual hardware," in Proceedings of the 1st Symposium on Networked Systems Design and Implementation (NSDI), 2004, pp. 169-82

[11] M. Nelson, B. H. Lim, and G. Hutchins, "Fast transparent migration for virtual machines," in USENIX Annual Technical Conference, 2005, pp. 391-394.

[12] C. Clark, K. Fraser, S. Hand, and G. H. Jacob, "Live migration of virtual machines," in Proceedings of 2nd ACM/USENIX Symposium on Network Systems, Design and Implementation (NSDI), 2005, pp. 273-286.

[13] Z. Gong, X. Gu, and W. John, "PRESS: PRedictive Elastic ReSource Scaling for cloud systems," presented at the International Conference on Network and Service Management (CNSM) Niagara Falls, ON, 2010.

[14] G. Qiang, Z. Ziming, and F. Song, "Ensemble of Bayesian predictors for autonomic failure management in cloud computing," presented at the 20th International Conference on Computer Communications and Networks (ICCCN), Maui, HI, 2011.

[15] E. Caron, F. Desprez, and A. Muresan, "Forecasting for grid and cloud computing on-demand resources based on pattern matching," presented at the IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom), Indianapolis, IN, 2010.

[16] M. Z. Hasan, E. Magana, A. Clemm, L. Tucker, and S. L. D. Gudreddi, "Integrated and autonomic cloud resource scaling," presented at the IEEE Network Operations and Management Symposium (NOMS), Maui, HI, 2012.

[17] F. Wei, L. ZhiHui, W. Jie, and C. ZhenYin, "RPPS: A novel resource prediction and provisioning scheme in cloud data center," presented at the IEEE Ninth International Conference on Services Computing (SCC), Honolulu, HI, 2012.

[18] M. Naderpour, J. Lu, and G. Zhang, "An intelligent situation awareness support system for safety-critical environments," Decision Support Systems, vol. 59, pp. 325-340, 2014.

[19] H. Kwakernaak, "Fuzzy random variables I: Definitions and theorems," Information Sciences, vol. 15, pp. 1-29, 1978.

[20] M. Naderpour, J. Lu, and G. Zhang, "A fuzzy dynamic bayesian network-based situation assessment approach," presented at the 22nd IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Hyderabad, India, 2013.

[21] P. Weber and L. Jouffe, "Complex system reliability modelling with dynamic object oriented Bayesian networks," Reliability Engineering & System Safety, vol. 91, pp. 149-162, 2006.

[22] M. Naderpour and J. Lu, "A situation analysis decision support system based on dynamic object oriented Bayesian networks," Journal of Software, vol. 9, 2014.

[23] M. Naderpour, J. Lu, and G. Zhang, "An abnormal situation modeling method to assist operators in safety-critical systems," Reliability Engineering & System Safety, vol. 133, pp. 33-47, 2015.

[24] M. Naderpour, J. Lu, and G. Zhang, "The explosion at Institute: Modeling and analyzing the situation awareness factor," Accident Analysis & Prevention, vol. 73, pp. 209-224, 2014.

[25] M. Naderpour, J. Lu, and G. Zhang, "A situation risk awareness approach for process systems safety," Safety Science, vol. 64, pp. 173-189, 2014.

[26] H. Jin, W. Gao, S. Wu, X. Shi, X. Wu, and F. Zhou, "Optimizing the live migration of virtual machine by CPU scheduling," Journal of Network and Computer Applications, vol. 34, pp. 1088-1096, 2011.

[27] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. s. A. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software: Practice and Experience, vol. 41, pp. 23-50, 2010.

[28] Decision System Laboratory, "SMILE (Structural Modeling, Inference, and Learning Engine)," University of Pittsburgh, 1998.