Future Generation Computer Systems 68 (2017) 14–30

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

A computational model for ranking cloud service providers using hypergraph based techniques

Nivethitha Somu [a], Kannan Kirthivasan [b], Shankar Sriram V.S. [a,*]

[a] Centre for Information Super Highway (CISH), School of Computing, SASTRA University, Thanjavur, Tamil Nadu, India

[b] Department of Mathematics, SASTRA University, Thanjavur, Tamil Nadu, India

HIGHLIGHTS

- This paper presents a Hypergraph based Computational Model (HGCM) for ranking CSPs.
- HGCM uses hypergraph based technique (Minimum Distance-Helly Property (MDHP)) to evaluate the CSPs.
- MDHP exploits the relation between the Service Measurement Index (SMI) metrics to select appropriate CSPs based on CUs requirement.
- HGCM uses arithmetic residue and Expectation–Maximization (EM) algorithms to impute missing values.
- Complexity of HGCM has been reduced through exploitation of the Helly property.

ARTICLE INFO

Article history:
Received 9 December 2015
Received in revised form
22 July 2016
Accepted 9 August 2016
Available online 9 September 2016

Keywords:
Cloud service ranking
Hypergraph
Helly property
Minimum distance algorithm
Service Measurement Index (SMI)

ABSTRACT

In a cloud marketplace, the existence of wide range of Cloud Service Providers (CSPs) makes it hard for the Cloud Users (CUs) to find an appropriate CSP based on their requirements. The design of a suitable service selection framework helps the users in the selection of a suitable CSP, while motivating the CSPs to satisfy the assured Service Level Agreement (SLA) and enhance the Quality of Service (QoS). Existing service selection models employ random assignment of weights to the QoS attributes, replacement of missing data by random values, etc. which results in an inaccurate ranking of the CSPs. Moreover, these models have high computational overhead. In this study, a novel cloud service selection architecture, Hypergraph based Computational Model (HGCM) and Minimum Distance-Helly Property (MDHP) algorithm have been proposed for ranking the cloud service providers. Helly property of the hypergraph had been used to assign weights to the attributes and reduce the complexity of the ranking model, while arithmetic residue and Expectation–Maximization (EM) algorithms were used to impute missing values. Experimental results provided by MDHP under different case studies (dataset used by various research communities and synthetic dataset) confirms the ranking algorithm to be scalable and computationally attractive.

1. Introduction

'Cloud computing'—an internet-based technology has changed the way through which the computing resources were accessed and utilized by the users [1,2]. Based on the user's requirements, the Cloud Service Provider (CSP) models the requested resources and offers them in the form of cloud services. Computing resources can be modeled using the three cloud service models namely, Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) [3]. IaaS abstracts the physical hardware (server, network, etc.) in the form of virtual servers and storages, thereby provide the Cloud Users (CUs) with an environment to deploy, run and monitor them (e.g. Amazon Web Services (AWS), Google Compute Engine (GCE), Windows Azure, etc.). PaaS provides a platform on top of the abstracted hardware to develop cloud applications (e.g. Google App Engine, Apprenda, etc.). SaaS provides the entire application as a service, enabling the CUs to overlook the suspicions about the infrastructure, platform and application installation (e.g. Google Apps, Citrix GoToMeeting, Cisco WebEx, etc.). From the CUs viewpoint, IaaS offers greater flexibility and minimum application automation, in comparison with the other two service delivery models [4].

Support provided by the Information Technology (IT) infrastructure becomes mandatory for organizations up and down the

* Corresponding author.
E-mail addresses: nivethitha@sastra.ac.in (S. Nivethitha), kkannan@maths.sastra.edu (K. Kannan), sriram@it.sastra.edu (V.S. Shankar Sriram).

http://dx.doi.org/10.1016/j.future.2016.08.014
0167-739X/

scale. For large organizations, setting up and maintaining an IT infrastructure (servers, network cables, storage, cooling infrastructure, etc.) might be as easy as pie. Small and medium enterprises might perceive it as a 'hard nut to crack', since it requires skilled manpower (developers, network administrators, system administrators, etc.) and high capital investment on IT infrastructure. Cloud computing aims to deliver appropriate services based on Quality of Service (QoS) requirements at competitive costs to the CUs in different organizations [4]. CUs can access these services from anywhere, at any time on a 'Pay as You Go' fashion. Therefore, it is unnecessary for small and medium sized organizations to invest on hardware and manpower for delivering business services. To summarize, cloud computing offers overwhelming benefits (flexibility, disaster recovery, software updates, no capital investment, etc.) to a variety of business organizations, by providing liberation from the hitches in the task of setting up an IT infrastructure and enabling them to concentrate on an innovative way to enhance their business service values [3]. In order to exploit the various benefits offered by cloud computing, various organizations have started developing their own applications on cloud infrastructures. These merits in turn attract many organizations to move on to the cloud in order to provide their business solutions for a large-scale user community. However, moving an existing business model to cloud involves numerous challenges with respect to the unique requirements and characteristics of different applications.

Conventionally, computing resources have been purchased or leased from the data centers and the users were billed irrespective of their usage. Emergence of cloud computing enables the users to access computing as a basic utility similar to water, electricity, gas, etc. where the consumers pay only for the resource(s) utilized [4]. Evolution of this technology produces a wide range of public cloud services which varies with respect to their features, performance, and pricing levels. Nevertheless, identification of an appropriate CSP who can satisfy their QoS requirements becomes harder on the CUs end as there exists a trade-off between various functional and non-functional requirements. Hence, it is important for the CUs to evaluate and find a suitable CSP for a service request rather than discovering multiple CSPs.

Security and trust management are the two major thrust areas for future research in the field of cloud computing. Trust Management System (TMS) helps the CUs to find an appropriate CSP with expected QoS [5]. Trust of a service assessed through the existing trust evaluation systems insinuates the security and QoS level offered by various CSPs [6]. In general, TMS consists of components such as cloud service discovery, trust metrics selection and measurement, trust assessment, trust evolution and trust based ranking model to assess the trustworthiness of any CSP (Fig. 1) [7]. Out of these five components, ranking models which have been used to rank the CSPs, play a vital role in the entire life cycle of TMS. This paper, emphasizes the importance of service selection mechanism (ranking models) designed to prioritize CSPs based on their likelihood to the CUs requirements.

Selection of CSPs who match with the maximum set of functional and non-functional requirements requested by the CUs is a decision problem. This problem is similar to Multi-Criteria Decision-Making (MCDM), as complex decision making process involves multiple attributes and interdependent relationships among them [8]. In the present study, a novel cloud service selection architecture with Hypergraph based Computational Model (HGCM) and Minimum Distance-Helly Property (MDHP) ranking algorithm have been proposed to address the problem of service selection. The proposed MDHP ranking algorithm ranks and selects the most suitable CSPs from the available pool of CSPs. Further, Cloud Service Measurement Index Consortium—Service Measurement Index (CSMIC—SMI) has been used as standard metric [9] to assess different CSPs based on the CUs requirements. CSMIC—SMI was launched by Carnegie Mellon University as a standard measure to evaluate any service based on the user's requirements.

In this paper, Section 2 describes the CSMIC—SMI, novel cloud service selection architecture and quality model for cloud services. Section 3 enlightens the basics of hypergraph with its properties, HGCM, MDHP ranking algorithm along with its complexity analysis and missing data imputation techniques. Section 4 deals with the performance analysis of the MDHP algorithm using various case studies. Section 5 concludes the paper with future works.

2. Materials and methods

2.1. Service Measurement Index (SMI)

Traditional High Performance Computing (HPC) metrics and benchmarks, focusing on performance and cost [10] cannot be applied to cloud environment due to its distributed and dynamic nature. This prompted many standard bodies to frame benchmark tools like Information and Communication Technology Service Quality (ICTSQ), ISO/IEC 9126, Application Performance Index (APDEX), eSourcing Capability Model—Client Organizations (eSCM-CL) and SMI to evaluate different services [11]. SMI is a hierarchical model which consists of critical QoS characteristics or Key Performance Indicators (KPIs) to compare any type of services (non-cloud vs cloud services or cloud services among different CSPs) [9]. The entire metric space is divided into seven categories in which each category possess four or more attributes. Each attribute is further partitioned into sub-attributes or KPIs and so on. The major categories of the SMI metrics are discussed as follows:

(i) **Accountability**
Accountability can be defined as a collection of QoS attributes that are used to measure the specific characteristics of different CSPs. This measure plays a vital role in building up the trust of a CU on any CSP. In general, organizations prefers to store sensitive data or deploy any application in a CSP's premises with appropriate security measures and compliance. SMI considers auditability, compliance, ownership, etc. to measure accountability.

(ii) **Agility**
One of the major benefits offered by the cloud is the rapid enlargement and transformation of organizations resources with minimal disruption. Agility of a cloud service depends on adaptability, elasticity, portability, etc. which can be measured in terms of how quickly new features can be incorporated into an IT infrastructure.

(iii) **Assurance**
The quality of any service depends on the likelihood between the original performance and assurance provided during service negotiation. With respect to the cloud service, assurance specifies the CSP's adherence to the promised standards in the Service Level Agreement (SLA). SMI considers availability, reliability, resiliency, etc. as evaluation metrics for assurance, thereby confers how much a service is identical to the specifications in the SLA.

(iv) **Cost**
For organizations, cost is considered as a single quantifiable metric which plays a significant role among the various deciding attributes while moving to cloud. Hence, it is desirable to express cost with respect to the characteristics pertinent to an organization.

(v) **Performance**
CSP provides a number of services with respect to the requirements of various business and academic organizations. The performance of these services can be assessed with respect to suitability, interoperability, accuracy and so forth.

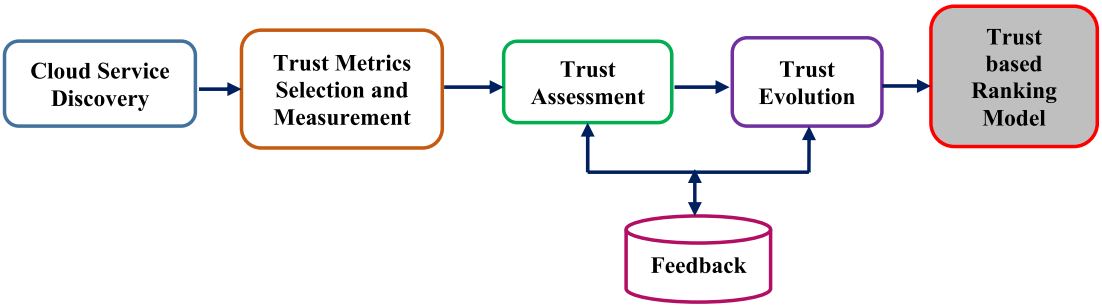


Fig. 1. Major components of TMS.

(vi) **Security and privacy**
Cloud computing can be simply viewed as a group of remote servers connected through internet to provide the required services to the CUs. It is obvious that the entire processing and storage of CU's data are hosted under the control of other organizations, based on their stringent security policies and compliance. Security and privacy are multi-dimensional attributes which includes confidentiality, data integrity, access control, etc.

(vii) **Usability**
For a Layman, the term 'cloud computing' projects a different view in each and every aspect. Therefore, the cloud service available to the CUs, must be easy to understand and use i.e. cloud service must be similar to what one does in his personal devices like desktop, laptop, mobile, etc. Usability of a cloud service depends on accessibility, learnability, transparency and operability.

2.2. Cloud service selection architecture

Fig. 2 represents the proposed cloud service selection architecture, which enable the CUs to find a suitable CSP based on their requirements. The MDHP ranking algorithm in the proposed cloud service selection architecture facilitates the CUs to evaluate the CSPs based on their adherence to the QoS requirements. Once a CU finds out an appropriate CSP, SLA is negotiated. Further the performance of the cloud services can be assessed through previous user experiences and log files (monitoring information). The proposed cloud service selection architecture consists of three layers namely Cloud Service Provider Layer (CSPL), Service Ranking Layer (SRL) and Cloud User Layer (CUL)

(i) ***Cloud Service Provider Layer (CSPL)***
It consists of diverse set of cloud service providers to offer one or more cloud services (IaaS, PaaS, SaaS, etc.). CSPL communicates with the other two layers for cloud service based interactions. In general, cloud services are indexed in several search engines and accessed through web portals which enable the user to find multitude of CSPs through a single keyword search. Performance of any service can be measured by evaluating the SMI—KPIs associated with each service. Some of the attributes such as availability, sustainability and cost may be common for all the service types.

(ii) ***Service Ranking Layer (SRL)***
Based on the requirements submitted in the CU layer, the SRL identifies the suitable CSP using the following components

a. **Cloud Service Registry and Discovery (CSRD)**
The CSPs register themselves in the middle tier of the proposed architecture by submitting the details of their services such as service id, service_type, etc. CSRD module in the SRL provides the list of CSPs based on the requirements submitted by the CUs. For example, if a user requests for a mail service, CSRD list out all the mail service providers. (e.g. Yahoo, Gmail, Hotmail, and Rediffmail)

b. **Monitoring Agent (MA)**
MA monitors and records the activities of the CSPs and CUs in a log based information system throughout the life time of the service. This information plays a significant role in performance evaluation of the CSPs and CUs based on the QoS assurance made during SLA negotiation [12].

c. **Evidence Base (EB)**
EB is a data store, where the CUs can provide their feedbacks (subjective) for the cloud services offered to them. Data from the EB and MA are pre-processed to extract the SMI metric values to assess the performance of the CSPs.

d. **Service Ranking Module (SRM)**
SRM in the proposed architecture measures the QoS levels provided by CSPs based on the information obtained from MA and EB for different service request. For example, if a CSP fails to provide the promised QoS, then the CSP will be penalized. Existence of this specific module in the architecture identifies appropriate CSPs who satisfy the CUs requirements to the maximum extent.

(iii) ***Cloud User Layer (CUL)***
Wide range of users in the CUL, have different QoS requirements. The privacy and the identity of the users are preserved using Identity Management Service (IdM) [13]. The users in the CUL interacts with the SRL to submit service requests and feedbacks in order to evaluate the CSPs. SLA Manager is responsible for the service agreements and negotiations.

2.3. Quality model for cloud service models

Cloud services are delivered to its users in the form of IaaS, PaaS and SaaS through various cloud deployment models such as public cloud, private cloud, hybrid cloud and community cloud [2,3]. However, to assess the QoS provided by these services, some quantitative and qualitative metrics are required. In this paper, we adhere to the standard SMI metrics (KPIs) framed by the CSMIC. KPIs can be categorized as quantitative (metrics measured using some monitoring tools such as SolarWinds [14] and Nagios [15]) and qualitative (metrics inferred through user feedbacks and monitoring information logs) [16,17]. The usefulness and the practicability of these metrics are assessed through the software quality metrics (correlation, tracking, consistency, predictability, discriminative power and reliability) identified from the IEEE standard 1061 [18]. The importance and the definition for these attributes depend on the CUs and their application requirements. Quantitative metrics can be expressed using some standard definition, whereas the qualitative metrics are extracted from large trusted datasets. Fig. 3 depicts the classification of SMI KPIs into qualitative and quantitative metrics.

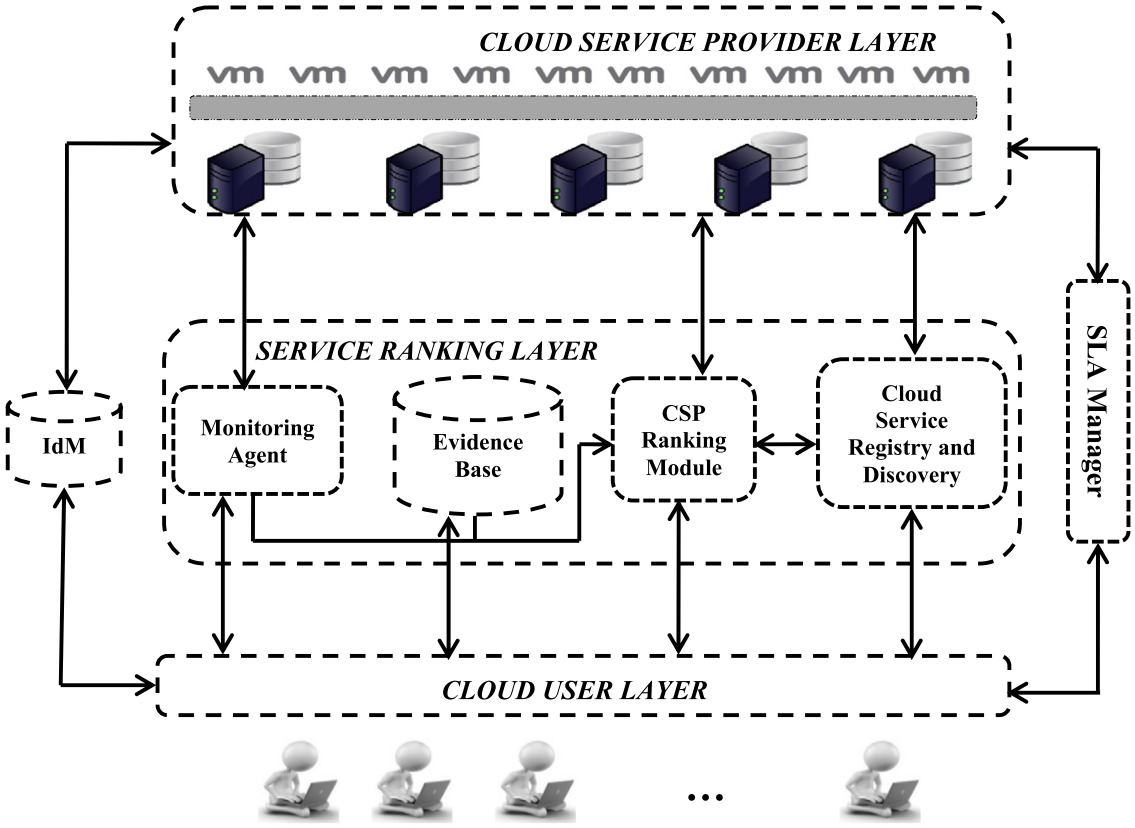


Fig. 2. Proposed cloud service selection architecture.

2.4. Related works

Trust is the foremost concern that needs to be addressed to realize the economic benefits offered by the cloud. Trust and security are inter-related as "*security includes concepts of trust and trust is a bundle before any secure and insecure communication takes place among the entities*" [7]. Among the components in the TMS, ranking service providers plays a vital role in selecting the trusted CSPs. In general, trust is computed using objective and subjective opinions, obtained as a result of CSP's adherence to the SLA and the level of CU's satisfaction on the service, respectively [16].

Massive growth in cloud computing technologies enables the researchers to focus their attention towards the performance evaluation of different CSPs and their services for diverse set of applications. This results in the development of cloud service evaluation and monitoring tools (performance and cost tradeoff) such as CloudCmp—performance and cost [19,20], CloudHarmony—performance [21], CloudMonitoring—performance [22], CloudRank – D – performance [23], CloudSleuth—performance [24], and CloudStone—performance [25]. The evaluation tool for web services cannot be used directly for cloud service as these services differ by nature.

Hoi Chan et al. [26] proposed an application and service provider mapping system for ranking service providers based on the Singular Vector Decomposition (SVD). The provider metrics from the repository is transformed and processed in the form of Provider—QoS matrix for SVD transformation in order to select a single CSP based on the CUs requirements. The service mapper approach does not include all qualitative metrics for evaluation and thereby provide limited potential for service comparison. P. Choudhury et al. [27,28] designed a Service Ranking System (SRS), which consists of static and dynamic states for ranking cloud services. The static ranking system, ranks cloud service by neglecting the CU requirements, whereas the dynamic ranking system uses a weighing scheme for seven key attributes such as availability, cost, reliability, security, throughput and user feedbacks to rank cloud services. Static SRS is a good ranking mechanism, while the dynamic SRS is inefficient as the ranking is entirely based on the key attributes and weight normalization. If any of these two moves beyond the boundary, the accuracy of the ranking system is reduced. S.C. Tejas Chauhan et al. [29,30] proposed a ranking approach which ranks CSPs by matching SLA parameters of the given CU's requirements with the CSPs provision. L. Qu et al. [31] proposed a ranking approach, based on the objective and subjective assessments with respect to the benchmarks and CUs feedbacks respectively. Z. Zheng et al. [32] proposed a ranking mechanism based on the prediction of qualitative metric values which are essential for comparing services. S.K. Garg et al. [33] proposed a ranking model called SMICloud which ranks the cloud service based on the Analytic Hierarchy Process (AHP) [34] and uses a standard developed by CSMIC known as SMI metrics. AHP is a MCDM approach which disintegrates complex and ill structured problems into simple decisions or primary components in the form of a hierarchical structure. The relative weights assigned to each component is based on their relative importance with other elements in the hierarchy. AHP has been an effective method in solving problems related to MCDM.

To summarize, the existing literature in ranking service providers represent the data in the form of a matrix and considers only some of the qualitative and quantitative metrics, which might end up with inaccurate results. Finding out the suitable service providers with minimum set of performance metrics does not arrive at a perfect solution. Besides, when the number of CSPs and CUs scale up, the matrix representation upsurges the execution time, exacerbating the complexity of the system. In the present work, the concept of hyperedges in the hypergraph have been introduced for data representation, as an alternate to the

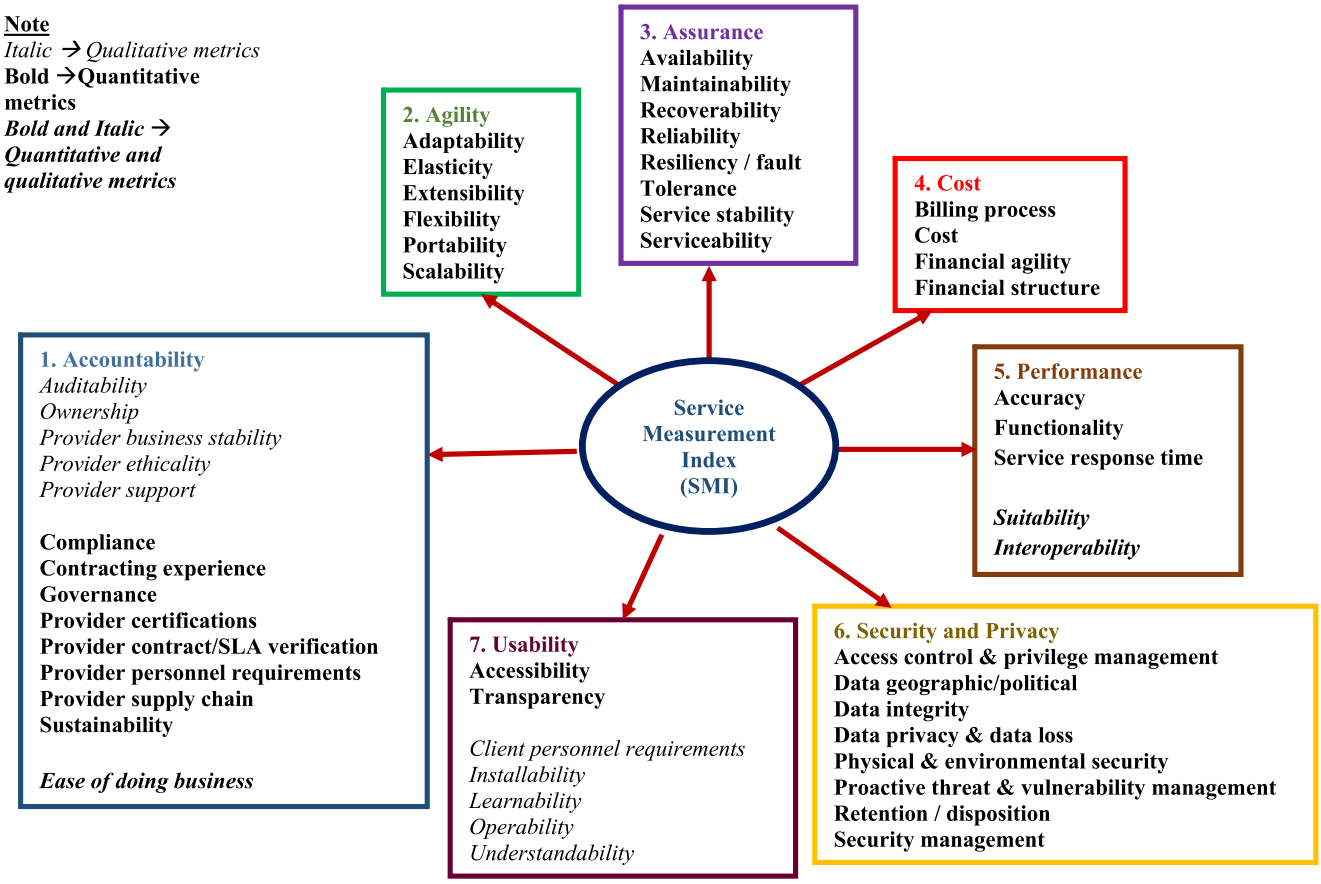


Fig. 3. SMI qualitative and quantitative attributes.

matrix representation, to reduce the complexity of the system. The reason for representing the dataset via hypergraph structure is that, the hyper relations prevailing among the attributes are more expressible via intersecting hyperedges rather than row/column vectors of the matrix. The connectivity between the CSPs are better expressed via the hyper relations among the SMI metrics. This work proposes a Hypergraph Computational Model (HGCM) and Minimum Distance-Helly Property ranking algorithm (MDHP) as a solution to the problem of CSP ranking and selection.

3. Hypergraph based computational model for ranking cloud service providers

3.1. Hypergraph preliminaries

Let $Y = \{y_1, y_2, \ldots, y_n\}$ be a finite set. A hypergraph Y is a family $H = \{E_1, E_2, \ldots, E_n\}$ of subsets of Y such that [35,36]

$$E_i \neq \phi \tag{1}$$

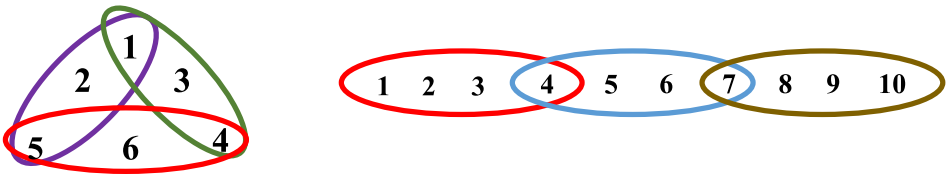
$$\bigcup E_i = Y; \quad i = 1, 2, \ldots, m. \tag{2}$$

The elements $y_1, y_2, \ldots, y_n$ of Y are called vertices and the sets $E_1, E_2, \ldots, E_m$ are called the hyperedges of the hypergraphs. A simple graph is a simple hypergraph, each of whose edges have a cardinality 2. A simple hypergraph or Sperner family is a hypergraph $H = \{E_1, E_2, \ldots, E_m\}$ such that $E_i \subset E_j$ implies $i = j$. Hypergraph provides exciting facilities to represent multiple attributes (dependent and independent) with n-ary relations, while graph structures provide binary relations between attributes. The representation of n-ary relations using hypergraph structures induces both topology and geometry of attributes in the form of hyperedges that can provide geometrical relations such as 4-bit neighborhood, 8-bit neighborhood, etc. [37]. While the topology is described by appropriate distance metric, geometry is described by the neighborhood relations. In this work, neighborhood relations among the hyperedges were represented using set notations. Set union and intersections are the operations that describe the n-ary relations between objects. Properties such as hypergraph cut, Helly, acyclic, etc. provide appropriate thresholds which should be maintained, to process the data in an elegant manner. Various hyperedges with different cardinalities evolve out of this process, describing the hierarchical structure of the assembled attributes. The recursive usage of Helly property results in achieving the same.

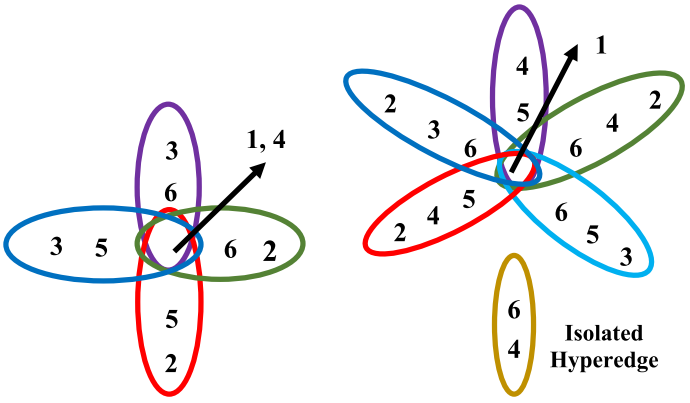
3.2. Helly property

In hypergraph, there exists an exciting property called Helly [38–40]. Consider a simple hypergraph H with hyperedges $E_1, E_2, \ldots, E_n$. The hypergraph H exhibits Helly property, if the intersection of E_j and $E_k (j, k \in J; J \subset \{1, 2, \ldots, n\})$ is non-empty. The necessary set theoretic operations used here are union and intersection of hyperedges. Fig. 4 represents the various cases involved in Helly property of the hypergraph. When a hypergraph has pairwise intersecting hyperedges, three cases arise.

***Case* 1**: *Pairwise intersecting hyperedges having an empty intersection* (Fig. 4(a)). This is due to the lack of Helly property. In image processing applications, while lack of Helly property corresponds to an edge, Helly property itself corresponds to a segment of an image. In our application, lack of Helly property corresponds to the service provider who does not satisfy the CUs requirements.



(a) Pairwise intersecting hyperedges without common intersection (lack of Helly property).



(b) Pairwise intersecting hyperedges with common intersection (exploits of Helly property).

(c) An example of an isolated hyperedge.

Fig. 4. Helly property.

***Case* 2**: *Pairwise intersecting hyperedges having a common intersection* (Fig. 4(b)). This is due to the existence of Helly property. In our application, this corresponds to a service provider who fulfills the CUs requirements.

***Case* 3**: There may not be any pairwise intersecting edges and such edges are called as *isolated edges* (Fig. 4(c)).

3.3. Representation of cloud dataset using hypergraph

Hypergraph representations are predominant in the field of image processing and clustering schemes. In this paper, to demonstrate the n-ary relations prevailing among the attributes of the CSPs, matrix form of data representation has been replaced by hyper relations (n-ary) in the hyperedges (group of attributes). Cloud dataset which consists of several SMI categories, attributes and KPIs were illustrated in the form of a hypergraph structure i.e. each hyperedge in the hypergraph represent the SMI metrics, while the vertices represent the CSPs. Notations used for CSMIC—SMI metrics are tabulated in Table 5.

Data representation using hyperedges in the hypergraph is based on

1. **Topology**—Distance metric between vertices
2. **Geometry**—Neighborhood relations between hyperedges.

By principle, sub-attributes are realized to be nearer to each other, which altogether form a hyperedge. The attributes that bear the same measurable unit may belong to a hyperedge, while the attributes that do not constitute isolated hyperedges. The pairwise intersecting hyperedges and the cluster strength (no of vertices in the cluster) are determined by the Euclidean distance metric with an appropriate threshold. Fig. 5 shows the two distance metrics used in this paper to process the data in the hypergraph structure, where $a = (x_1, y_1)$ and $b = (x_2, y_2)$

(i) **Inter cluster distance**—City Block distance metric [41] used between different clusters (SMI KPIs) is defined by

$$\mu\,(a, b) = \max\{|x_1 - x_2|\,, |y_1 - y_2|\}\,. \tag{3}$$

(ii) **Intra cluster distance**—Euclidean distance metric used within the cluster, i.e. between SMI attributes is defined by

$$d\,(a, b) = \ |a - b|\,. \tag{4}$$

3.4. Building neighborhood hyperedges

Let $CSP_i = \{CSP_1, CSP_2, \ldots, CSP_s\}$ and $CU_i = \{CU_1, CU_2, \ldots, CU_u\}$ be the respective sample sets of the CSPs and CUs used in HGCM. Each sample in the sample set CSP_i and CU_i has N attributes. CSP_i and CU_i have similar attribute values for S attributes, different attribute values for V attributes, missing attribute values for P attributes. Let J be a Cartesian product of CU_i and CSP_i, represented as $J = CU \times CSP = \{(CU_1, CSP_1), (CU_1, CSP_2)\,, \ldots, (CU_1, CSP_s)\,, (CU_2, CSP_1)\,, (CU_2, CSP_2), \ldots, (CU_2, CSP_s)\,, \ldots, (CU_u, CSP_1)\,, (CU_u, CSP_2)\,, \ldots, (CU_u, CSP_s)\}$; A_i, A_{ij} and A_{ijk} denotes the categories, attributes and KPIs respectively. With respect to Table 5, for Assurance (A_3), Availability (A_{51}), Service stability (A_{52}), Serviceability (A_{53}) are the KPIs, Upload Time (A_{521}), CPU (A_{522}) and Memory (A_{523}) are the sub-attributes. $v\left(CSP_{i_{Prov}}, A_i\right)$, $v\left(CSP_{i_{Prov}}, A_{ij}\right)$, $v\left(CSP_{i_{Prov}}, A_{ijk}\right)$, $v\left(CU_{i_{Req}}, A_i\right)$, $v(CU_{i_{Req}}, A_{ij})$ and $v(CU_{i_{Req}}, A_{ijk})$ denotes the CSPs provision ($CSP_{i_{Prov}}$) and the CUs requirement ($CU_{i_{Req}}$) values for each SMI metric. With the help of minimum distance metric, $\min\left|CSP_{i_{Prov}} - CU_{i_{Req}}\right|$ the CSP_i which are closer to the $CU_{i_{Req}}$ are formed as a hyperedge for each element in the SMI hierarchy. For example, to construct a hyperedge for Accountability (a_1), compute the minimum distance between $CSP_{i_{Prov}}$ and $CU_{i_{Req}}$ from the dataset $v\,(CSP_1, 4)$, $v\,(CSP_2, 8)$, $v\,(CSP_3, 4)$, $v\,(CSP_4, 5)$, $v\,(CSP_5, 5)$, $v\,(CU_1, 4)$. Based on the minimum distance metric, the distance between the CSPs provision and the CU's requirement is found to be {0, 4, 0, 1, 1}. CSP_i closer to the CU's requirement form the vertices of the hyperedge (Accountability) i.e. CSP_1, CSP_3, CSP_4 and CSP_5.

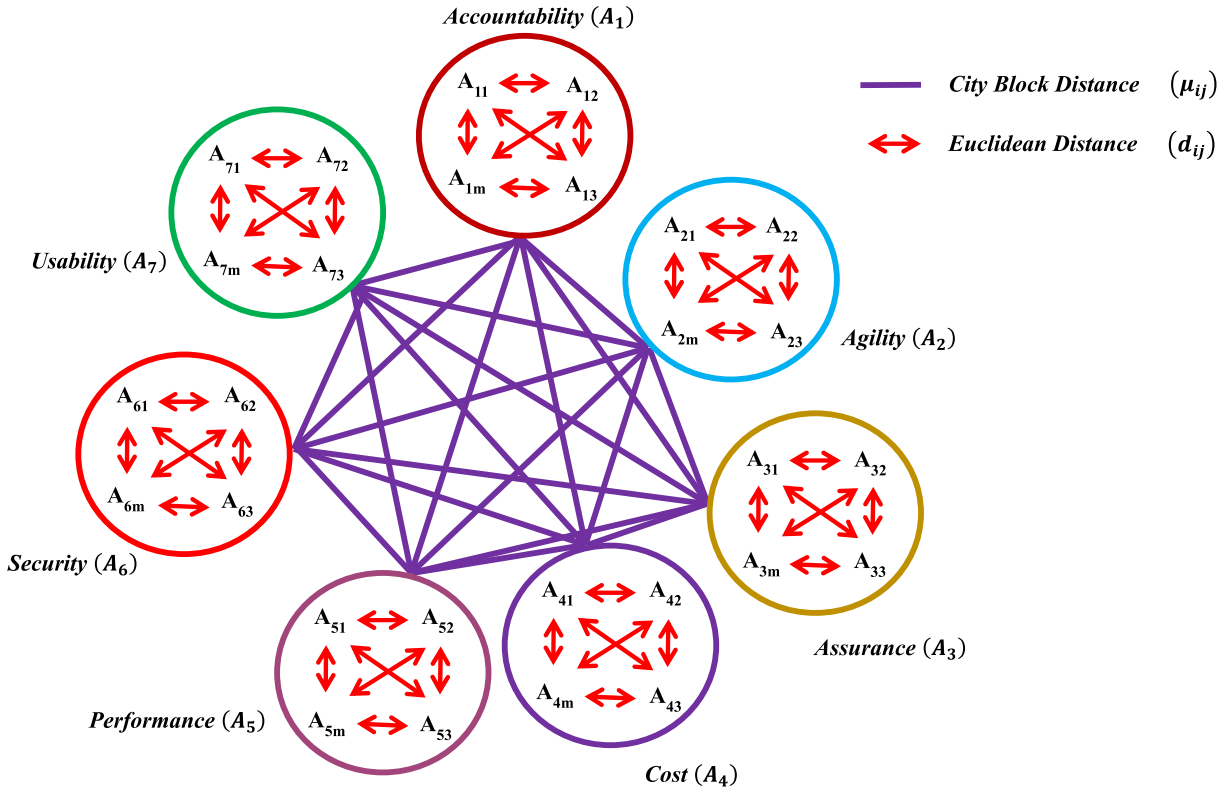


Fig. 5. Distance metrics used for processing the dataset.

3.5. Ranking of cloud services using minimum distance Helly property algorithm

The most probable solution to address the problem of service selection is to rank the CSPs based on their adherence to the CUs requirements. In this paper, we propose a novel HGCM and MDHP ranking algorithm to rank and select the suitable CSPs (Algorithm 1). Fig. 6 depicts the process involved in HGCM for ranking cloud services. Data represented in the form of matrix was subjected to Chi-Square test for checking the degree of independency among the attributes [42]. The hyper relations (relation defined based on distance metric and a threshold) among the SMI metrics were represented via hyperedges, where its intersections reveal the inter relations between them. A minimum distance algorithm was invoked to arrange the attributes in a hierarchical order. Neighborhood relations between the hyperedges have been established through fixing a minimum threshold using minimum distance algorithm. The recursive use of Helly property influence the order of CSPs, to select the best CSP.

While processing the data, dependency among the SMI attributes was identified using Chi-Square test. To examine the independent nature of the chosen attributes, Chi-Square test had been conducted with $(r-1, c-1)$ degrees of freedom, where r and c represent the number of rows and columns respectively. From the dataset used in the case studies, it was observed that a minimum of 65% of the attributes were found to be dependent on each other, even at a level of 10% significance. Hence, the interrelation between the attributes need to be utilized for ranking CSPs.

Initially, CSRD module in the SRL reduced the number of comparisons, by filtering out the CSPs, based on the service delivery model requested by the CUs. To filter out the untrusted CSPs, some form of evaluation based on the SMI metrics needs to be carried out over the CSPs registered with the CSRD module. SMI metric values were obtained by processing the data from the MA and EB using intelligent techniques (Arithmetic Residue-Probabilistic Neural Network [43]) to extract the maximum information contained in it. The evaluation was carried out based on the $CSP_{i_{Act}}$, obtained during data acquisition (SMI metric), rather than the promised value in the SLA. Using these datasets, the actual value ($CSP_{i_{Act}}(A_i)$) provided by the CSP (difference between the CSP promised value ($CSP_{i_{Prom}}$) and the provisioned value ($CSP_{i_{Prov}}$)) for each SMI attribute were determined.

$$(CSP_{i_{Act}}(A_i)) = \left(CSP_{i_{Prom}} - CSP_{i_{Prov}}\right) \forall CSP_{i_{Prov}}, \quad CU_{i_{Req}} \in A_i \tag{5}$$

where

$A_i = \{A_{ij} : j = 1, 2, \ldots, m\}$, where $A_{ij} = \{A_{ijk} : k = 1, 2, \ldots, n\}$, $i = 1, 2, \ldots, l$

$CU_{i_{Req}}$—CUs requirements.

CSPs with $CSP_{i_{Act}}$ lesser than δ (δ is the threshold which differs with respect to each SMI metric) has been considered for further progress. CSPs with high $CSP_{i_{Act}}$ were penalized and those who have been penalized for a certain number of times were blacklisted. This type of formulation reduce the number of untrusted CSPs in a cloud ecosystem.

Once the untrusted CSPs were filtered out from the process, $\frac{s}{2}$ CSPs were identified by likelihood computation between 's' CSPs provisions and 'u' CUs requirements. This likelihood measure can be modeled as a minimum distance problem, i.e. given two points at different locations, the distance between them is calculated. From 's' CSPs, a set of $\left(\frac{s}{2}\right)$ CSPs which has the maximum likelihood of satisfying the CUs requirements were represented by,

$$Hyperedge_{A_i} \leftarrow \text{Min}\{|CSP_{i_{Prov}} - CU_{i_{Req}}|\} \forall CSP_{i_{Prov}}, \quad CU_{i_{Req}} \in A_i \tag{6}$$

where

$A_i = \{A_{ij} : j = 1, 2, \ldots, m\}$, where $A_{ij} = \{A_{ijk} : k = 1, 2, \ldots, n\}$, $i = 1, 2, \ldots, l$.

$Hyperedge_{A_i}$—Top $\frac{s}{2}$ CSPs who adhere to the CUs requirements for each SMI attribute

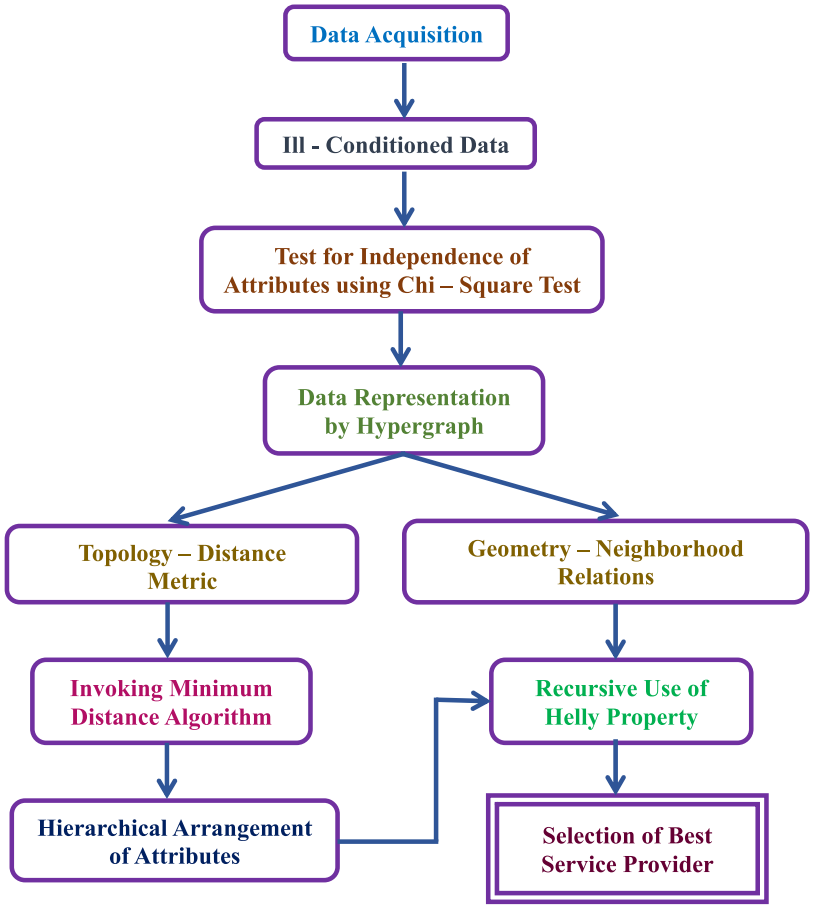


Fig. 6. Hypergraph based Computational Model (HGCM).

$CSP_{i_{Prov}}$—Resource provisioned by the CSPs
$CU_{i_{Req}}$—CUs requirements.

For example, if the level of security requested by the CU was 4 and the provisions given by (CSP_1, CSP_2, CSP_3) were (4, 6, 8), then CSP_1 is the most suitable provider than the others due to the minimum distance between the provisions and the CU's requirements ($CSP_1 = 0$; $CSP_2 = 2$; $CSP_3 = 4$).

Repeated application of minimum distance metric over the SMI hierarchy (KPIs, attributes and sub-attributes) resulted in the formation of neighborhood hyperedges. As explained in Section 3.2, the existence of Helly property on the pairwise intersecting hyperedges results in a common intersection, representing the most prominent vertex among the neighborhood hyperedges. With respect to the CSP ranking problem, the recursive application of Helly property on the neighborhood hyperedges generated for the SMI metrics, figures out the most suitable CSP. Thus, the use of hypergraph representation in this regard provides a novel solution to the problem of ranking service providers.

To enhance the reader's understandability on the working of the MDHP ranking algorithm, let us consider a QoS dataset with 3 CSPs (S_1, S_2, S_3) and 2 CUs (CU_1, CU_2) for 3 SMI categories (Table 1). Missing value in the table for the KPI (Agility: Capability: Disk) in $CU_{2_{Req}}$ had been imputed using arithmetic residue as the number of CSPs was 3 (Algorithm 1: Step 2).

The minimum distance metric was applied between the $CSP_{i_{Prov}}$ and $CU_{i_{Req}}$, to construct the neighborhood hyperedges for each SMI metric. The vertices of these neighborhood hyperedges represent the CSPs with the maximum likelihood to the CUs requirements (Table 2). Use of Helly property on the neighborhood hyperedges generated for the SMI attributes, resulted in a common intersection point which contains the CSPs satisfying the CUs requirements (Fig. 7(a) and Fig. 8(a)). The CSPs that adhere to the CU_1 and CU_2 requirements with respect to each SMI metric is given in Table 3. The overall ranking of the CSPs with respect to the SMI hierarchy involves the recursive application of Helly property on the neighborhood hyperedges built for each SMI category (Fig. 7(b) and Fig. 8(b)).

3.6. Recursive use of Helly property to find the suitable cloud service provider

Fortunately, pairwise dependency among the attributes form major relation between the vertices of the hypergraph (CSPs). Helly property discusses an important issue of partitioning the hypergraph representation into two different domains: (i) data which satisfy the Helly property and (ii) the data which lack Helly property. This partitioning mainly helps us in ranking the CSPs (repeated application of Helly property). Likelihood measure using the minimum distance metric and hypergraph representation of each SMI metric provides an insight to the $\frac{s}{2}$ CSPs who can satisfy the CUs requirements. To select a suitable CSP, the most prominent Helly property of hypergraph had been exploited and used recursively on the SMI standard measure hierarchy. Helly property exists, where all pairwise intersecting edges had a common intersection point (Fig. 4(b)). Application of Helly property on the unique hypergraph representation of each SMI metric pointed out the CSPs who fulfilled the user's requirements.

Application of Helly property on the constructed hyperedges, tri-partitions the entire dataset (tri-partition of hypergraph—assigning weights to the SMI metrics) into,

(i) Pairwise intersecting hyperedges with common intersection (*High priority*)

Algorithm 1: Minimum Distance - Helly Property (MDHP) Algorithm

Input:
D_{EB}: Data from EB;
D_{MA}: Data from MA;
CSP_i: List of CSPs;

Output:
$CSP_{i_{Prov}}$: SMI attribute values of the CSPs;
$CU_{i_{Req}}$: SMI attribute values of the CUs;
$Hyperedge_{A_i}$: Hyperedges for SMI Metrics;
$R_i \leftarrow \{R_1, R_2, \ldots, R_L\}$: Rank of the CSPs;

Step 1: (Extract the SMI metric values for CSP and CU)
for each CSP_i do
  $\boldsymbol{CSP_{i_{Act}}(A_i) \leftarrow \left(CSP_{i_{Prom}}(A_i) - CSP_{i_{Prov}}(A_i)\right)}$
  // $A_i \leftarrow \{A_{ij}: j = 1,2, \ldots, m\}$; $A_{ij} \leftarrow \{A_{ijk}: k = 1,2, \ldots, n\}$, *i = 1, 2, …, l*
  while $(\boldsymbol{CSP_{i_{Act}} < \delta})$ do
    for each SMI Metric do
      $\boldsymbol{CSP_{i_{Prov}} \leftarrow v(A_i, A_{ij}, A_{ijk})}$ // $CSP = \{CSP_1, CSP_2, \ldots CSP_s\}$
      $\boldsymbol{CU_{i_{Req}} \leftarrow v(A_i, A_{ij}, A_{ijk})}$ // $CU = \{CU_1, CU_2, \ldots CU_u\}$
    end
  end
end

Step 2: Imputing Missing Values
if $(CSP_i \leq 3)$
  $\boldsymbol{A_i} \leftarrow \frac{\sum_{i=1}^{n}\left(\frac{CSP_i(A_i)}{100+\epsilon}\right)_i}{Max\,(CSP_i(A_i)) - Min\,(CSP_i(A_i))}$
  // $CSP_i(A_i)$ – Available SMI attribute values; ∈ - Residue assumption to find missing value; $\boldsymbol{A_i}$ – Imputed value for the missing SMI attribute value
else
  CALL ***Expectation_Maximization*** $\boldsymbol{(CSP_{i_{Prov}})}$***;***
end

Step 3: Procedure Expectation Maximization $\boldsymbol{(CSP_{i_{Prov}})}$
  // (link: *https://github.com/roshank/EMAlgo/blob/master/em.rb*)

Step 4: Construction of Neighborhood Hyperedges
for each SMI metric do
  $\boldsymbol{Hyperedge_{A_i} \leftarrow Min\{|CSP_{i_{Prov}} - CU_{i_{Req}}|\} \forall\, CSP_{i_{Prov}}, CU_{i_{Req}} \in A_i}$
  //Kiviat graph representation to analyze the QoS provided by each CSPs with respect to the SMI KPIs prioritization.
end

Step 5: Ranking CSPs
for each SMI metric do
  $\boldsymbol{R_i \leftarrow \bigcap_{i \in CSP_i} Hyperedge_{A_i}}$ // Class 1, Class 2 and Class 3
end

Step 6: Return
  return $\boldsymbol{R_i}$;

Table 1
QoS dataset example—3 CSPs; 2 CUs.

KPIs	Attributes	Sub-attributes	$CSP_{1_{Prov}}$	$CSP_{2_{Prov}}$	$CSP_{3_{Prov}}$	$CU_{1_{Req}}$	$CU_{2_{Req}}$
Accountability			9	7	5	5	7
Agility	Capacity	CPU (GHz)	9.8	12.8	7.2	6.4	11.6
		Memory (GB)	17	5	22	20	*4.6 (Imputed value)*
		Disk (GB)	5420	684	550	540	650
	Elasticity	Response time (S)	80–120	580–780	80–120	60–120	560–700
Security			9	7	5	5	7

(ii) Pairwise intersecting hyperedges with no common intersection (*Medium priority*)
(iii) Isolated hyperedges (*Low priority*).

The Analytic Hierarchy Process in [34] was replaced by minimum distance algorithm based tri-partition technique of the hypergraph. The CSPs were ranked based on the following set of rules,

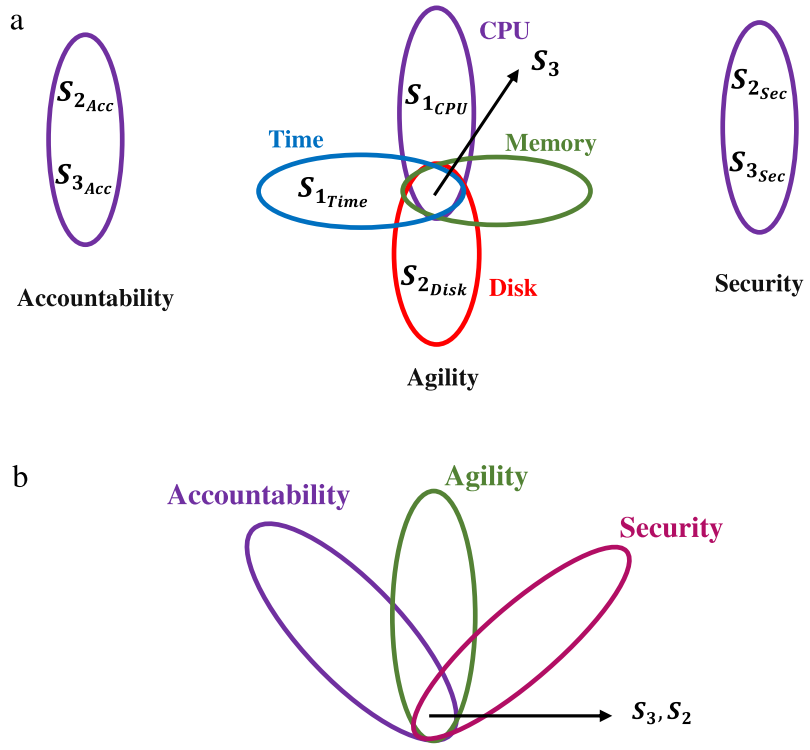


Fig. 7. Recursive use of Helly property for CSP ranking—CU_1; (a) SMI attributes (b) SMI KPIs.

Table 2
Construction of neighborhood hyperedges.

KPIs	Attributes	Sub-attributes	$Hyperedges_{CU_1}$			$Hyperedges_{CU_2}$		
			$CSP_{1_{Prov}}$	$CSP_{2_{Prov}}$	$CSP_{3_{Prov}}$	$CSP_{1_{Prov}}$	$CSP_{2_{Prov}}$	$CSP_{3_{Prov}}$
Accountability (A_1)			4	2	0	2	0	–
Agility (A_2)	Capacity (A_{21})	CPU (GHz) (A_{211})	3.4	6.4	0.8	–	1.19	–
		Memory (GB) (A_{212})	–	–	2	12.4	0.4	17.4
		Disk (GB) (A_{213})	4880	144	10	4770	34	–
	Elasticity (A_{22})	Response time (S) (A_{221})	20 – 0	–	20 – 0	–	20 – 80	–
Security (A_3)			4	2	0	2	0	–

Table 3
CSPs ranking for each SMI KPIs and attributes.

SMI KPIs	CSP ranking—CU_1	CSP ranking—CU_2
Accountability	S_3, S_2	S_2, S_1
Agility	$\boldsymbol{S_3, S_1}$	$\boldsymbol{S_2, S_1}$
Capacity	$\boldsymbol{S_3, S_2, S_1}$	$\boldsymbol{S_2, S_1}$
CPU	S_3, S_1	S_2
Memory	S_3	S_2, S_1
Disk	S_3, S_2	S_2, S_1
Elasticity	$\boldsymbol{S_1, S_3}$	$\boldsymbol{S_2}$
Response time	S_1, S_3	S_2
Security	$\boldsymbol{S_3, S_2}$	$\boldsymbol{S_2, S_1}$

(i) The number of vertices that belong to the common intersection of the pairwise intersecting hyperedges are categorized to be *Class* 1. The common intersecting clusters with descending order of vertices were ranked as $R_1, R_2, \ldots, R_M$.
(ii) Pairwise intersecting hyperedges with no common intersection were categorized to be *Class* 2 and ranked as $R_{M+1}, R_{M+2}, \ldots, R_N$ according to descending order of vertices contained in it.
(iii) *Class* 3 symbolizes the isolated hyperedges with descending order of vertices, ranked as $R_{N+1}, R_{N+2}, \ldots, R_L$.

To summarize, the recursive use of Helly property on the hyperedges (SMI KPIs) resulted in the selection of suitable CSPs (vertices) who satisfy the CUs requirements in an efficient manner.

3.7. Pre-processing algorithm for imputing missing values

In general, missing values had been estimated using data imputation techniques. In [34], missing data with respect to $CSP_{i_{Prov}}$ had been replaced by random assignments and missing values in $CU_{i_{Req}}$ had not been taken into consideration. Even under this circumstance, AHP provide the results with reasonable accuracy, due to the strength of relative metric introduced therein. However, in MDHP ranking algorithm for the sequence of data taken into consideration, a simple normalization procedure with hypothetical arithmetic residue $\in$ was sufficient to find the missing value (Algorithm 1). This is feasible because of the tri-partitioning ranking algorithm (MDHP) based on hypergraph structure.

However, for large sized data, imputation of missing values was possible through Expectation–Maximization (EM) algorithm (Algorithm 1: Step 3) [44]. In this paper, for case study 2 EM algorithm had been used to obtain missing values, where for a given data size 'n', the values were imputed by fitting the data into a class of distributions (In our case, normal distribution is taken into consideration). The missing data were imputed with the maximum value by considering the parameter distribution of the variable for a given estimated parameter in the Expectation step (E step). EM algorithm was executed 3 times with respect to case study 2, for finding the optimal missing values by tuning σ (standard deviation) of the normal distribution.

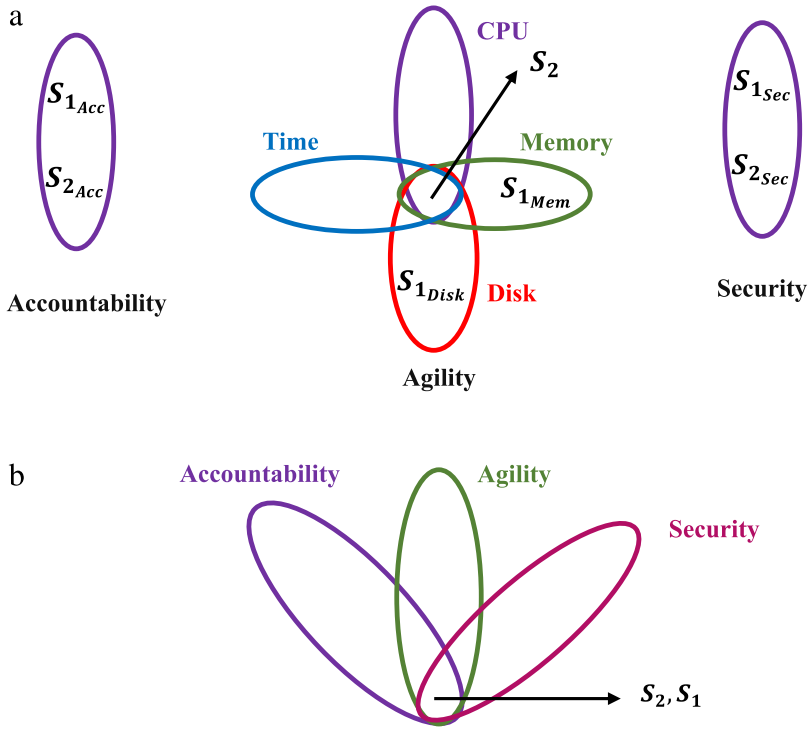


Fig. 8. Recursive use of Helly property for CSP ranking—CU_2; (a) SMI attributes (b) SMI KPIs.

Table 4
QoS dataset of Amazon EC2 (S_1), Windows azure (S_2), Rackspace (S_3), private cloud using OpenStack (S_4) and private cloud using eucalyptus (S_5).

SMI metrics			Dataset—evaluation studies [45–47]			Dataset—private cloud set up SASTRA university		User requirement
KPI's	Attributes	Sub-attributes	Service 1 (S_1)	Service 2 (S_2)	Service 3 (S_3)	Service 4 (S_4)	Service 5 (S_5)	
Accountability	*Level* (0–10)		4	8	4	5	5	4
Agility	*Capacity*	CPU	9.6	12.8	8.8	12.8	9.6	6.4 GHz
		Memory	15	14	15	16	14	10 GB
		Disk	1690	2040	630	500	400	500 GB
	Elasticity	Time	80–120	520–780	20–200	120–180	120–180	60–120 s
Assurance	*Availability*		99.95%	99.99%	100%	99.99%	99.90%	99.99%
	Service Stability	Upload time	13.6	15	21	12.1	10.5	–
		CPU	17.9	16	23	8	6	–
		Memory	7	12	5	18	14	–
	Serviceability	Free support	0	1	1	1	1	–
		Type of support	24/7, Diagnostic tools, phone, urgent response	24/7, Diagnostic Tools, Phone, Urgent Response	24/7, Phone, urgent response	8/6, Phone	8/6, Phone	24/7, Phone
Cost	*On-Going Cost*	VM cost	$0.68	$0.96	$0.96	–	–	<1$/hour
		Data						
		Inbound	10	10	8	10	10	100 GB/month
		Outbound	11	15	18	11	11	200 GB/month
		Storage	12	15	15	12	10	1000 GB
Performance	*Service response time*	Range	80–120	520–780	20–200	120–180	120–180	60–120 s
		Average value	100	600	30	100	100	–
Security	*Level* (0–10)		4	8	4	5	5	4

3.8. Computational complexity for MDHP ranking algorithm

This section discusses the computational complexity of the MDHP ranking algorithm used by the HGCM for ranking CSPs. Initially, a stable hierarchical structure was built for all the CUs requests. The computational complexity is due to the minimum distance algorithm used to compute the likelihood between the CSPs provisions and the CUs requirements. Let s be the number of CSPs and u be the number of CUs; N_L be the number of attributes at level L and n_{Li} be the number of sub-attributes at level $L-1$ of the ith attribute. To compute the likelihood between the CSPs and the CUs, the MDHP begins its processing from the KPIs of each level L and gets aggregated towards the SMI categories.

To compute the minimum distance between s CSPs and u CUs, $s*u$ computations were carried out for N_L attributes, which resulted in the computational complexity of $O(n^2 + su)$ and $O(n^2 + sN)$ respectively. Thus the total computational complexity of the MDHP ranking algorithm was *Minimum* $[(n^2 + su), (n^2 + sN)]$. The

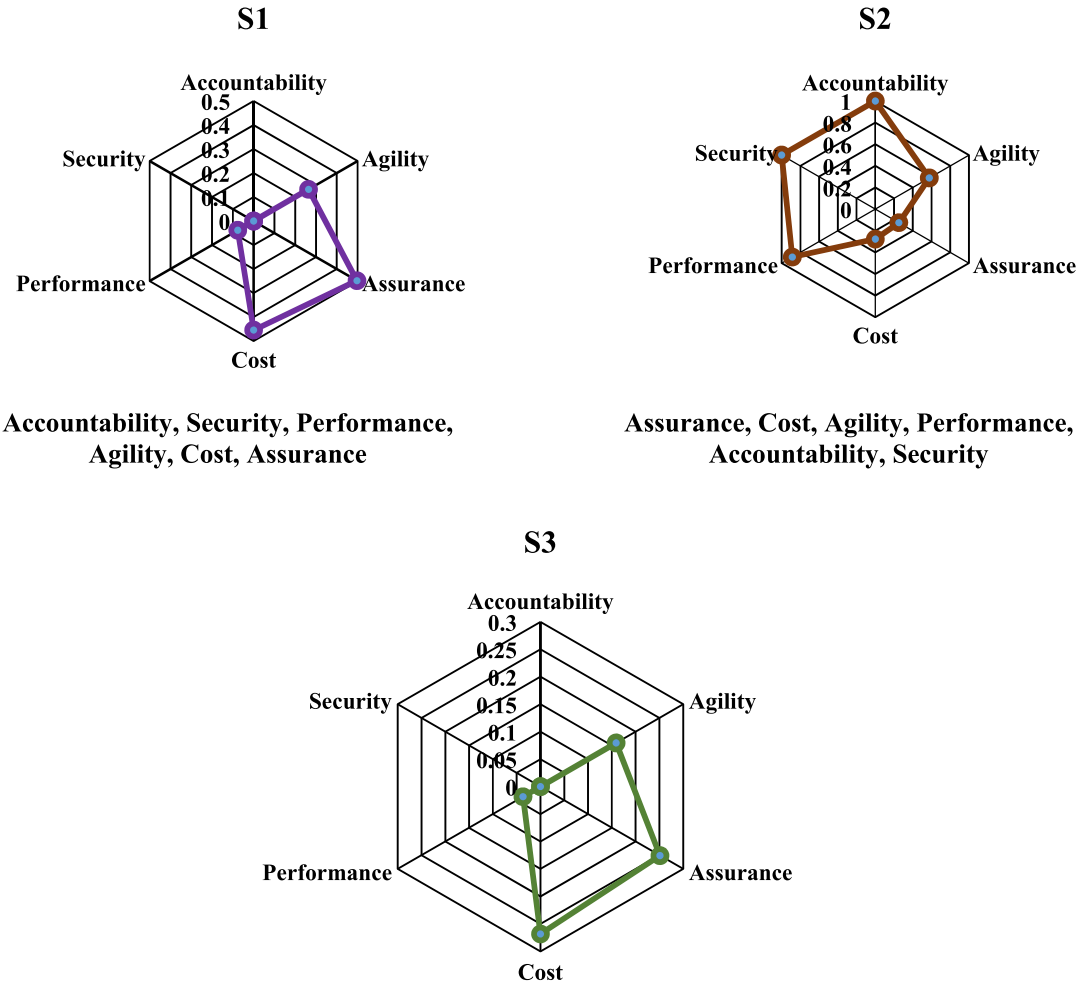


Fig. 9. Kiviat graph for Amazon EC2, windows azure and rackspace.

worst case computational complexity of the ranking algorithm was *Maximum* $[(n^2 + su),\ (n^2 + sN)] = (n^2 + sN)$, as sN was greater than su.

4. Case study—ranking CSPs using hypergraph based computational model

Various case studies presented in this paper mainly focus on the SMI metrics. Service indices of the SMI metrics were calculated using the QoS data of the various CSPs (public and private cloud set up).

4.1. Case study 1—dataset used by various research communities

The QoS data obtained from various evaluation studies for the three IaaS CSPs (Amazon EC2, Windows Azure and Rackspace) had been used to validate the computational model (HGCM) [45–47]. The service index of the three CSPs were computed for a set of 6 KPIs, 9 attributes and 15 sub-attributes. Unavailable QoS data with respect to the SMI attributes were randomly assigned to each CSP, whereas for the CUs, missing data were imputed using the data imputation techniques given in Section 3.7. The QoS dataset of various IaaS CSPs were arranged in the hierarchical structure of SMI metrics (Table 4).

To determine a suitable CSP, the distance between the CSPs provisions $\left(CSP_{i_{Prov}}\right)$ and the CUs requirements $\left(CU_{i_{Req}}\right)$ were computed using Eq. (6) with respect to each SMI metric. For this case study, let us assume that S_1, S_2 and S_3 were the three IaaS CSPs listed out by the CSRD module in the proposed cloud service selection architecture. A kiviat graph was generated based on the likelihood computations, provide an analysis on the CSP rankings for various SMI categories: Accountability ($S_1 < S_3 < S_2$); Agility ($S_3 < S_1 < S_2$); Assurance ($S_3 < S_2 < S_1$); Cost ($S_3 < S_2 < S_1$); Performance ($S_3 < S_1 < S_2$); Security ($S_1 < S_3 < S_2$) (Fig. 9).

The execution of MDHP algorithm was initiated with the construction of neighborhood hyperedges from the bottom level of the SMI hierarchy. For agility (category), initial hypergraph representation started from the CPU, memory, disk and response time (KPIs); the representation was aggregated towards the capacity and elasticity (attributes). CSPs which have $CSP_{i_{Prov}}$ with minimum distance from the $CU_{i_{Req}}$ constitute the vertices of the hyperedge constructed for each SMI metric. For example, the hyperedge created for $S_{n_{UT}}$ (Assurance: Service Stability: Upload Time) comprised of S_1 and S_2 as its vertices. When a SMI category had more than one attribute or sub-attribute, the hyperedges were created for each one of them and the Helly property was applied on the pairwise intersecting neighborhood hyperedges. The recursive application of Helly property on the neighborhood hyperedges with respect to each element in the SMI hierarchy resulted in a common intersection point indicating the suitable CSPs for each element of SMI (Fig. 10). For cost (Category), the application of Helly property on the intersection of hyperedges

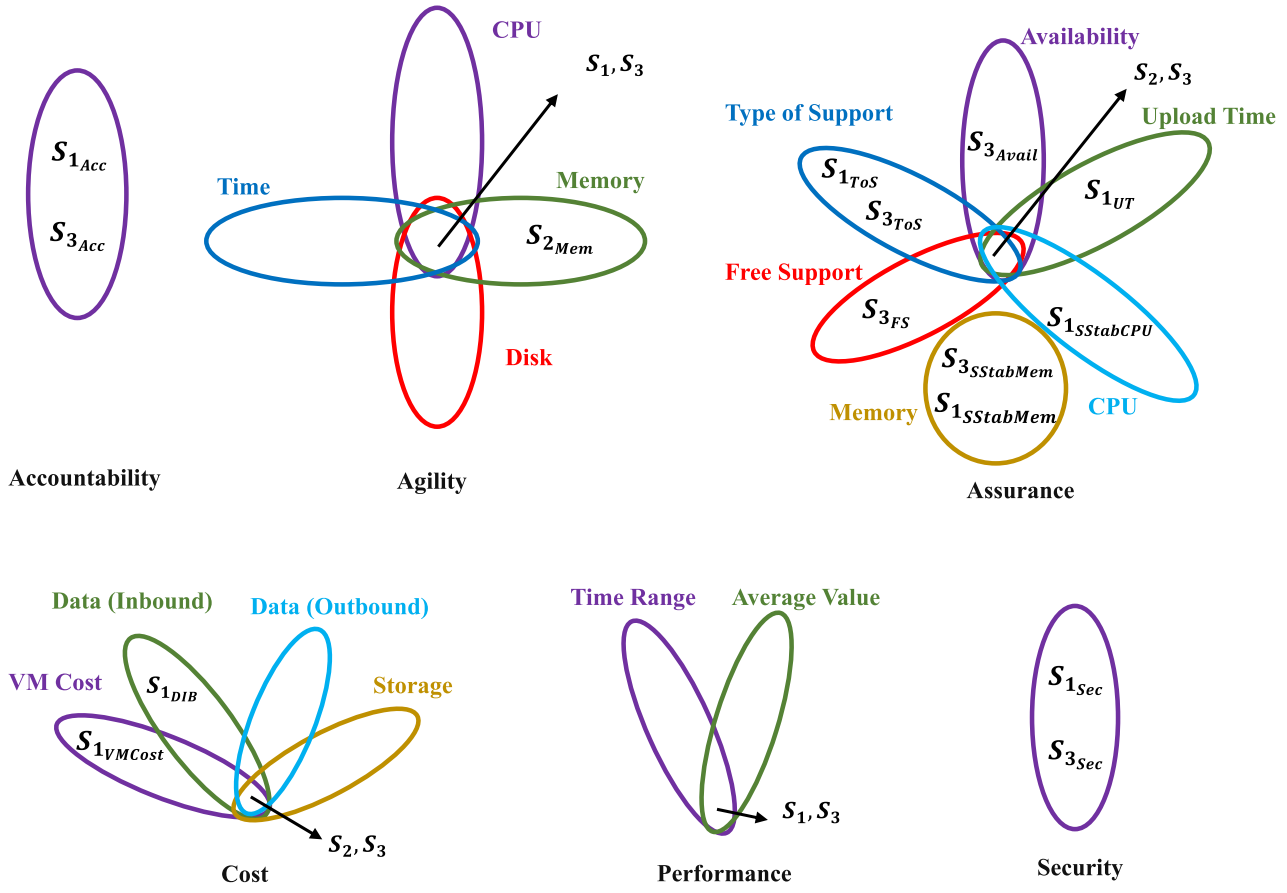


Fig. 10. CSPs ranking with respect to SMI attributes—case study 1.

(VM Cost, Inbound data, outbound data and Storage) resulted in two common intersection points (S_2, S_3). At the SMI category level, the application of Helly property on the intersection of hyperedges resulted in a common intersection point (S_3), representing the most suitable CSPs (Fig. 11). Based on the ranking rules given in Section 3.6, the CSPs were ranked as $S_3 < S_1 < S_2$, with S_3 as the most suitable CSP. The CSPs were ranked with respect to each attribute in the SMI hierarchy, based on their ability to satisfy the CU's requirements (Table 5).

4.2. Case study 2—-dataset obtained from the private cloud set up at SASTRA university

Case study 1 compared and ranked the three IaaS providers: Amazon EC2, Windows Azure and Rackspace using MDHP ranking algorithm. To test the applicability and scalability of the algorithm on various cloud environments, MDHP was applied to the SMI QoS dataset obtained from the private cloud set up (using Eucalyptus and OpenStack) at SASTRA University (Table 4). Collectively, five CSPs and one user's requirements were considered for evaluation. Missing values in the CU's requirement were imputed using the data imputation techniques given in Section 3.7.

The execution of MDHP ranking algorithm was initiated from the leaf node of the SMI hierarchy. To construct the neighborhood hyperedge for each SMI metric, the minimum distance metric given in Eq. (6) was invoked. HGCM displayed the adherence level of CSPs to the CUs requirements with respect to the SMI KPIs using kiviat graph (Fig. 12). Kiviat graph provides an analysis on the CSP rankings based on Accountability ($S_1 < S_3 < S_4 < S_5 < S_2$), Agility ($S_5 < S_3 < S_1 < S_4 < S_2$), Assurance ($S_4 < S_2 < S_3 < S_1 < S_5$), Cost ($S_4 < S_5 < S_1 < S_3 < S_2$), Performance ($S_3 < S_1 < S_4 < S_5 < S_2$) and Security ($S_1 < S_3 < S_4 < S_5 < S_2$).

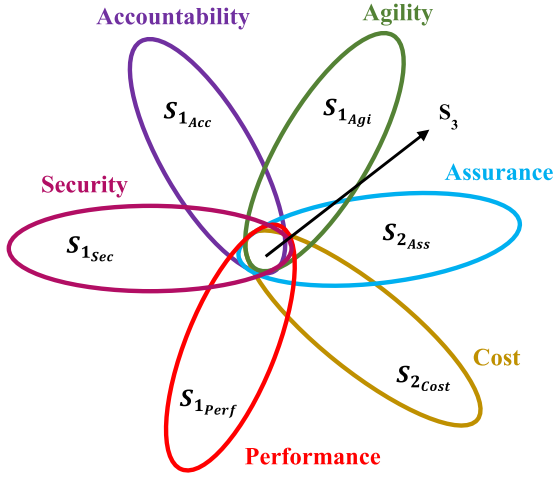


Fig. 11. CSPs ranking with respect to SMI KPIs—case study 1.

The vertices of the hyperedge constructed for the elements in the SMI hierarchy, containing the CSPs with minimum distance with the $CU_{i_{Req}}$ (Fig. 13). For example, the hyperedge for $S_{n_{Mem}}$ contained $S1_{Mem}$, $S2_{Mem}$, $S3_{Mem}$ and $S5_{Mem}$, excluding $S4_{Mem}$ which was farther away from $CU_{i_{Req}}$. The application of Helly property on the hypergraph structure constructed for performance (category), indicated $S1_{Mem}$, $S3_{Mem}$, $S4_{Mem}$ and $S5_{Mem}$ as the suitable CSPs (Fig. 14). As a whole, with the application of Helly property on the neighborhood hyperedges constructed for the SMI metrics, resulted in a common intersecting vertices, which were ranked as $S_3 < S_4 < S_5 < S_1 < S_2$ based on the rules given in Section 3.6.

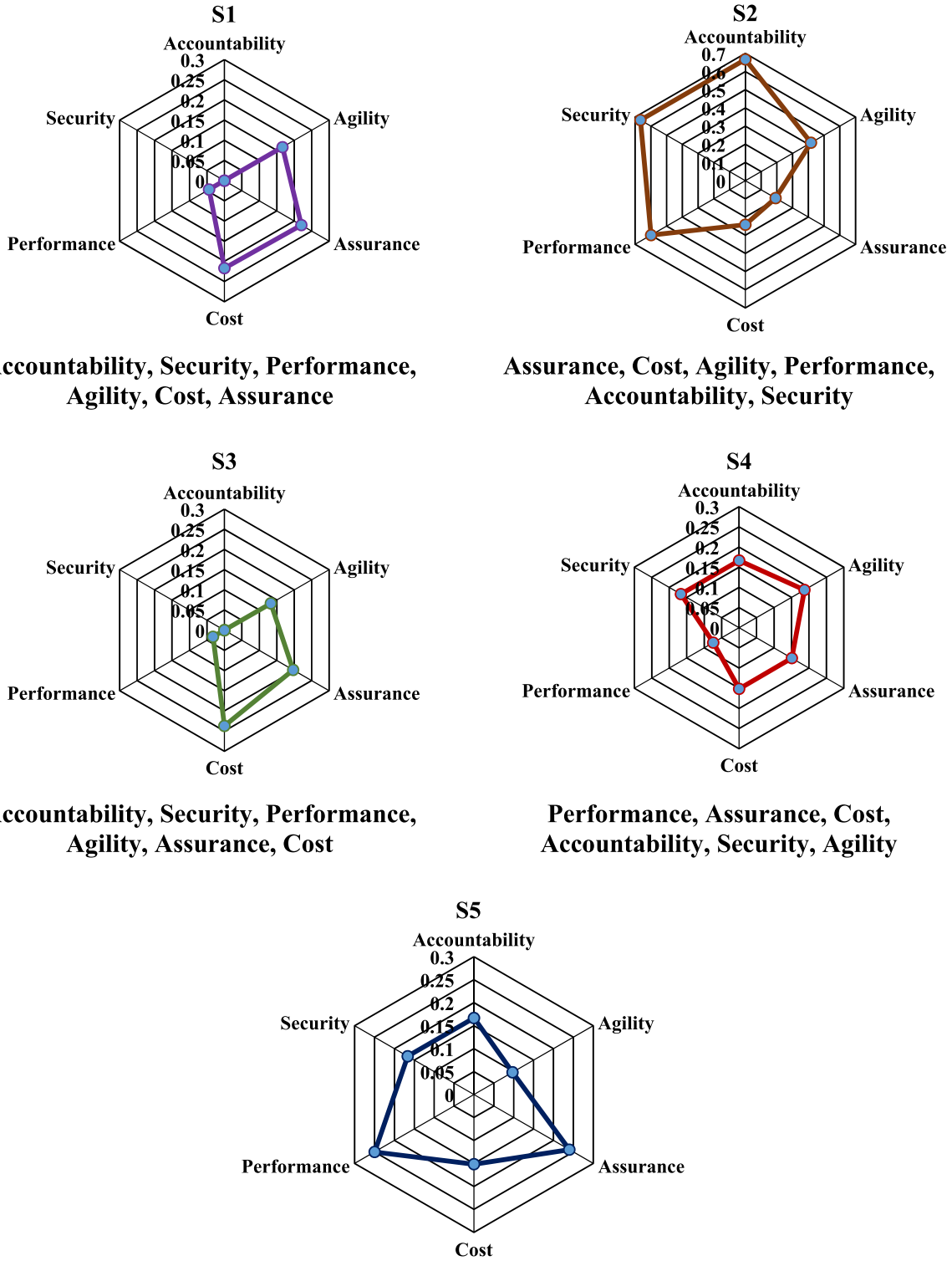


Fig. 12. Kiviat graph for amazon EC2, windows azure, rackspace, private cloud using OpenStack and eucalyptus.

The execution time of the MDHP ranking algorithm was analyzed by varying the number of CSPs and CUs, while keeping the number of SMI attributes constant. Fig. 15 illustrates the performance of the existing AHP and the proposed MDHP ranking algorithm with respect to the execution time. MDHP ranking algorithm ranks and selects the best CSPs amongst 1000 CSPs with less execution time. This was due to the reduced inherent time complexity provided by the hypergraph technique.

5. Conclusion and future work

Cloud computing spans across various organizations to satisfy the computational demands put forth by diversified users. Tremendous growth in the area of cloud computing had led to increase in the number of CSPs, service offerings and user expectations. Hence, finding an appropriate CSP fulfilling the QoS requirement of the CUs, remains a challenge. To overcome this

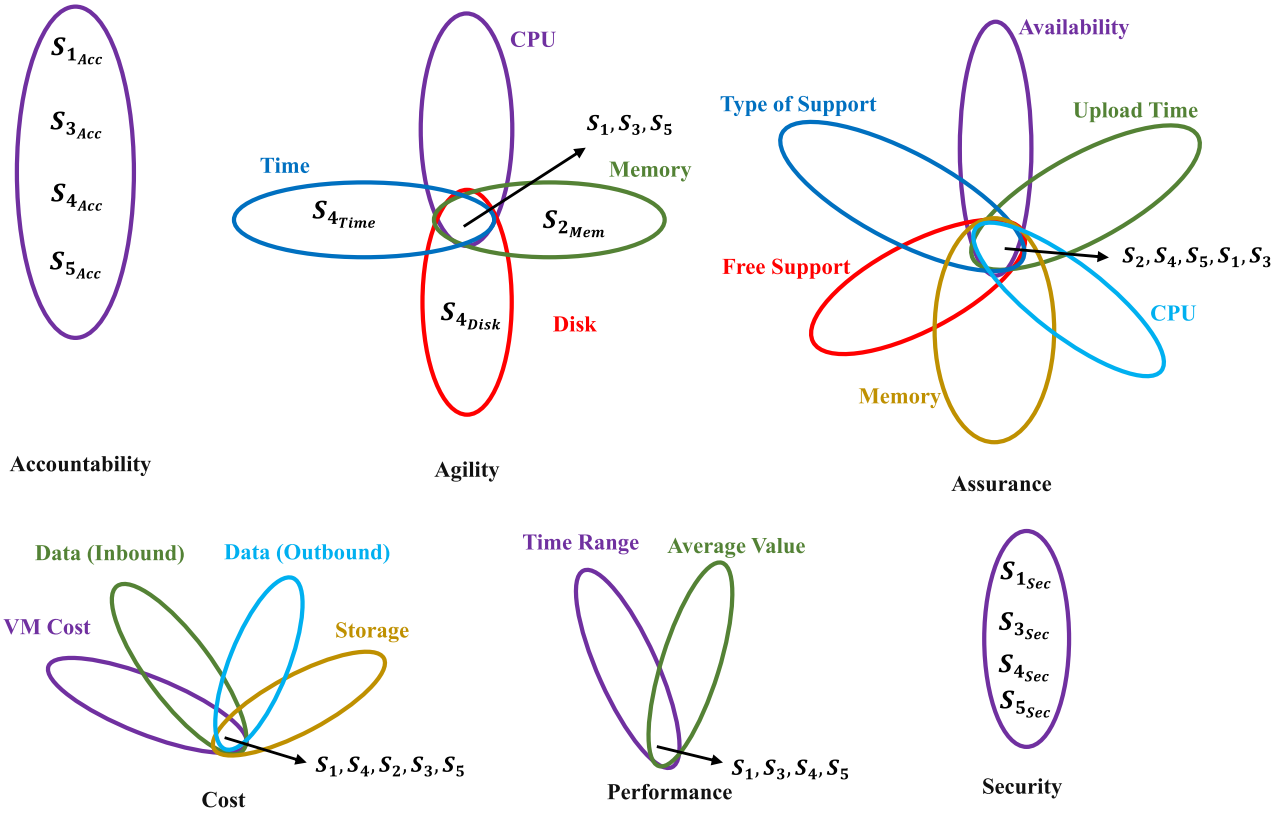


Fig. 13. CSPs ranking with respect to SMI attributes—case study 2.

Table 5
CSPs ranked with respect to each SMI metric—case study 1 and 2.

KPIs and attributes	Notations	CSPs	
		Case study 1—evaluation studies	Case study 2—private cloud
Accountability	$S_{n_{Acc}}$	S_1, S_3	S_1, S_3, S_4, S_5
Agility	$\mathbf{S_{n_{Agi}}}$	$\mathbf{S_1, S_3}$	$\mathbf{S_1, S_3, S_5}$
Capacity	$\mathbf{S_{n_{Cap}}}$	S_1, S_3	S_1, S_3, S_5
CPU	$\mathbf{S_{n_{CPU}}}$	S_3, S_1	S_3, S_1, S_5
Memory	$\mathbf{S_{n_{Mem}}}$	S_2, S_3, S_1	S_2, S_5, S_1, S_3
Disk	$\mathbf{S_{n_{Disk}}}$	S_3, S_1	S_4, S_3, S_1
Elasticity	$\mathbf{S_{n_{Elas}}}$	S_1, S_3	S_1, S_3, S_4, S_5
Time	$\mathbf{S_{n_{Time}}}$	S_1, S_3	S_1, S_3, S_4, S_5
Assurance	$\mathbf{S_{n_{Ass}}}$	$\mathbf{S_2, S_3}$	$\mathbf{S_2, S_4, S_5, S_1, S_3}$
Availability	$\mathbf{S_{n_{Avail}}}$	S_2, S_3	S_2, S_4, S_3
Service stability	$\mathbf{S_{n_{SStab}}}$	S_1, S_2	S_1, S_2, S_4, S_5
Upload time	$\mathbf{S_{n_{UT}}}$	S_1, S_2	S_5, S_4, S_1
CPU	$\mathbf{S_{n_{SStabCPU}}}$	S_2, S_1	S_5, S_4, S_2
Memory	$\mathbf{S_{n_{SStabMem}}}$	S_3, S_1	S_3, S_1, S_2
Serviceability	$\mathbf{S_{n_{Serv}}}$	S_2, S_3	S_2, S_3, S_4, S_5, S_1
Free support	$\mathbf{S_{n_{FS}}}$	S_2, S_3	S_2, S_3, S_4, S_5
Type of support	$\mathbf{S_{n_{ToS}}}$	S_3, S_1, S_2	S_1, S_2, S_3
Cost	$\mathbf{S_{n_{Cost}}}$	$\mathbf{S_2, S_3}$	$\mathbf{S_1, S_4, S_5, S_2}$
On-going cost	$\mathbf{S_{n_{OGCost}}}$	S_2, S_3	S_1, S_4, S_5, S_2
VM Cost	$\mathbf{S_{n_{VMCost}}}$	S_1, S_2, S_3	S_4, S_5, S_1
Data	$\mathbf{S_{n_{Data}}}$	S_2, S_3	S_1, S_2, S_4, S_5
Inbound	$\mathbf{S_{n_{DIB}}}$	S_1, S_2, S_3	S_1, S_2, S_4, S_5
Outbound	$\mathbf{S_{n_{DOB}}}$	S_3, S_2	S_3, S_2, S_4, S_5, S_1
Storage	$\mathbf{S_{n_{Storage}}}$	S_2, S_3	S_2, S_3, S_1, S_4
Performance	$\mathbf{S_{n_{Perf}}}$	$\mathbf{S_1, S_3}$	$\mathbf{S_1, S_3, S_4, S_5}$
Service response time	$\mathbf{S_{n_{SRT}}}$	S_1, S_3	S_1, S_3, S_4, S_5
Range	$\mathbf{S_{n_{SRTRange}}}$	S_1, S_3	S_1, S_3, S_4, S_5
Average value	$\mathbf{S_{n_{SRTAvgVal}}}$	S_3, S_1	S_3, S_4, S_5, S_1
Security	$\mathbf{S_{n_{Sec}}}$	$\mathbf{S_1, S_3}$	$\mathbf{S_1, S_3, S_4, S_5}$
Usability	$\mathbf{S_{n_{Use}}}$	–	–

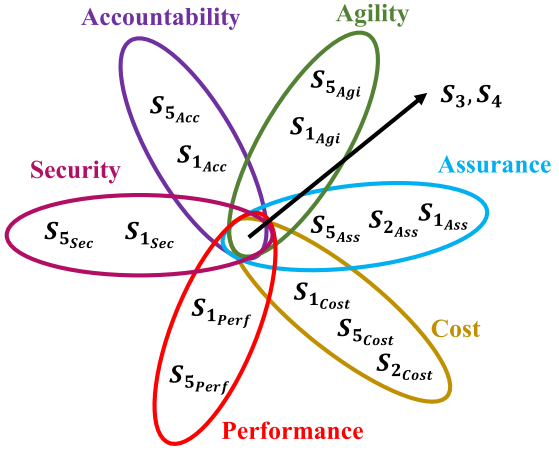


Fig. 14. CSPs ranking with respect to SMI KPIs—case study 2.

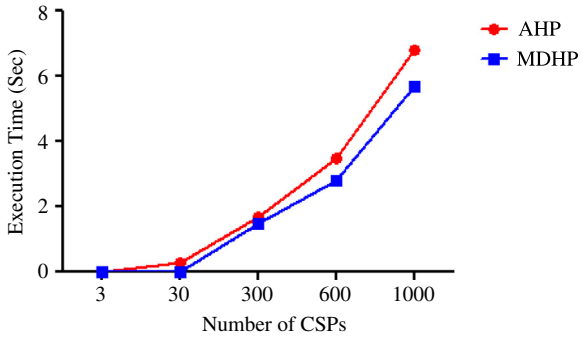


Fig. 15. Execution time of AHP and MDHP.

issue, a CU needs to identify and measure some standard set of performance metrics relevant to their application. Therefore, CSMIC proposed a set of standard metrics known as SMI to compare different cloud services based on various KPIs.

In the current study, ranking models based on CSPs adherence to the users requirements had been proposed to find suitable CSP. A 3 – tier cloud service selection architecture with Hypergraph based Computational Model (HGCM) and Minimum Distance-Helly Property (MDHP) ranking algorithm in the service ranking layer was proposed to measure and quantify the SMI QoS attributes thereby facilitating the CUs to rank the cloud services. This study had addressed the issue of missing values by introducing arithmetic residue and EM algorithm. Observations from the case studies proved that the MDHP ranking algorithm is efficient with reduced time complexity. HGCM enables the CSPs to analyze themselves by comparing with other CSPs and to enhance the level of satisfaction experienced by the CUs.

Thus MDHP algorithm in the HGCM framework was found to be scalable and computationally attractive for various domains using big data analytics such as VLSI design [48], image processing [41], stock market data analysis for ranking volatility estimates, TMS for finding trustworthy CSPs, graph optimization, DNA sequencing and weather forecasting, validation of metadata quality in Geographic Information System (GIS) and other time series problems.

Acknowledgments

The first and third author thank the Department of Science and Technology, India for INSPIRE Fellowship (Grant No: DST/INSPIRE Fellowship/2013/963) and Fund for Improvement of S&T Infrastructure in Universities and Higher Educational Institutions (SR/FST/ETI-349/2013) for their financial support. The second author thanks the Department of Science and Technology—Fund for Improvement of S&T Infrastructure in Universities and Higher Educational Institutions Government of India (SR/FST/MSI-107/2015) for their financial support. We would like to express our gratitude towards the unknown potential reviewers who have agreed to review this paper and provided valuable suggestions to improve the quality of the paper.

References

[1] Frank John Krautheim, Building trust into utility cloud computing (Ph.D. diss.), University of Maryland, 2010.
[2] Felix N. Njeh, Cloud computing: An evaluation of the cloud computing adoption and use model (Ph.D. diss.), Bowie State University, 2014.
[3] Peter Mell, Tim Grance, The NIST Definition of Cloud Computing, 2011.
[4] R. Buyya, C. Yeo, S. Venugopal, J. Broberg, I. Brandic, Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility, Future Gener. Comput. Syst. 25 (6) (2009) 599–616.
[5] Mohamed Firdhous, Osman Ghazali, Suhaidi Hassan, Applying bees algorithm for trust management in cloud computing, in: Bio-Inspired Models of Networks, Information, and Computing Systems, Springer Berlin, Heidelberg, 2012, pp. 224–229.
[6] D. Marudhadevi, V. Neelaya Dhatchayani, V.S. Shankar Sriram, A trust evaluation model for cloud computing using service level agreement, Comput. J. (2014) 1–8.
[7] Sabu M. Thampi, Bharat Bhargava, Pradeep K. Atrey (Eds.), Managing Trust in Cyberspace, CRC Press, 2013.
[8] Milan Zeleny, James L. Cochrane, Multiple Criteria Decision Making, University of South Carolina Press, 1973.
[9] CSMIC Consortium, Service Measurement Index Version 1.0, Carnegie Mellon University Silicon Valley Moffett Field, CA USA, 2011, pp. 1–8.
[10] C. Binnig, D. Kossmann, T. Kraska, S. Loesing, How is the weather tomorrow?: towards a benchmark for the cloud, in: Proceedings of the Second International Workshop on Testing Database Systems, RI, USA, 2009.
[11] P.M.A.C. Costa, Evaluating cloud services using multicriteria decision analysis (Dissertation), 2013.
[12] Jagpreet Sidhu, Sarbjeet Singh, Compliance based trustworthiness calculation mechanism in cloud environment, Procedia Comput. Sci. 37 (2014) 439–446.
[13] V.Neelaya Dhatchayani, V.S. Shankar Sriram, Trust aware identity management for cloud computing, Int. J. Inf. Comm. Technol. 6 (3–4) (2014) 369–380.
[14] Robert John Hennan, John G. Roane, Security monitoring tool for computer network. US Patent 7,904,456, issued March 8, 2011.
[15] Wolfgang. Barth, Nagios: System and Network Monitoring, No Starch Press, 2008.
[16] Lie Qu, Yan Wang, Mehmet A. Orgun, Ling Liu, Athman Bouguettaya, Cloud service selection based on contextual subjective assessment and objective assessment, in: Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems, International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 1483–1484.
[17] Sushil Singh, Deepak Chand, Trust evaluation in cloud based on friends and third party's recommendations, in: Engineering and Computational Sciences (RAECS), 2014 Recent Advances in, IEEE, 2014.
[18] IEEE Standards Association and Others, IEEE STD 1061–1998, IEEE standard for a software quality metrics methodology, 1998.
[19] X.Y. Ang Li, Srikanth Kandula, Ming Zhang, CloudCmp: Comparing Public Cloud Providers, Proceedings of the 10th ACM SIGCOMM conference on Internet measurement, 2011, pp. 1–14.
[20] X.Y. , Srikanth Kandula, Ming Zhang, CloudCmp:Shopping for a Cloud Made Easy, 2nd USENIX Workshop on Hot Topics in Cloud Computing 2010, pp. 1–7.
[21] CloudHarmony, 2009, http://www.cloudharmony.com.
[22] Z. Rehman, O.K. Hussain, S. Parvin, F.K. Hussain, A Framework for User Feedback Based Cloud Service Monitoring, Sixth International Conference on Complex, Intelligent, and Software Intensive Systems 2012, pp. 257–262.
[23] J. Chunjie LUO1, LeiWang Zhen, Gang LU, Lixin Zhang, Cheng-Zhong XU, Ninghui Sun, CloudRankD: Benchmarking and Ranking Cloud Computing Systems for Data Processing Applications, Front. Comput. Sci. 6 (4) (2012) 347–362.
[24] T. Abubakr, Cloud Sleuth, 2011, https://www.cloudsleuth.net.
[25] V. Sobel, S. Subramanyam, A. Sucharitakul, J. Nguyen, Cloudstone: Multi-Platform, Multi-Language Benchmark and Measurement Tools for Web 2.0, First Workshop on Cloud Computing and Its Application, Chicago, IL, October 22–23, 2008, pp. 1–6.
[26] H. Chan, Ranking and Mapping of Applications to Cloud Computing Services by SVD, 2010, pp. 362–369.
[27] P. Choudhury, M. Sharma, K. Vikas, T. Pranshu, V. Satyanarayana, Service ranking systems for cloud vendors, Adv. Mater. Res. 433–440 (2012) 3949–3953.
[28] S. Preeti Gulia, Automatic Selection and Ranking of Cloud Providers using Service Level Agreements, Int. J. Comput. Appl. 72 (11) (2013).
[29] S.C. Tejas Chauhan, M. Vikas Kumar, Bhise, Service level agreement parameter matching in cloud computing, 2011.
[30] Cirrocumulus: A Semantic Framework for Application and core Services Portability across Heterogeneous Clouds, project at Kno-e-sis Center at Wright State University, 2010.
[31] L. Qu, Y. Wang, M. Orgun, Cloud Service Selection Based on the Aggregation of User Feedback and Quantitative Performance Assessment, 2013, pp. 152–159.
[32] Z. Zheng, X. Wu, Y. Zhang, M. Lyu, J. Wang, QoS ranking prediction for cloud services, J. IEEE Trans. Parallel Distrib. Syst. 24 (6) (2012) 1213–1222.
[33] S.K. Garg, S. Versteeg, R. Buyya, A framework for ranking of cloud computing services, Future Gener. Comput. Syst. 29 (4) (2013) 1012–1023.
[34] O.S. Vaidya, S. Kumar, Analytic hierarchy process: an overview of applications, European J. Oper. Res. 169 (1) (2006) 1–29.
[35] Claude. Berge, Graphs and Hypergraphs Vol. 7, North-Holland Publishing Company, Amsterdam, 1973.
[36] Claude. Berge, Hypergraphs: Combinatorics of Finite Sets. Vol. 45, Elsevier, 1984.
[37] R. Dharmarajan, K. Kannan, A hypergraph-based algorithm for image restoration from salt and pepper noise, AEU-Int. J. Electron. Commun. 64 (12) (2010) 1114–1122.
[38] Alain Bretto, Hocine Cherifi, Driss Aboutajdine, Hypergraph imaging: an overview, Pattern Recognit. 35 (3) (2002) 651–658.
[39] A. Bretto, H. Cherifi, A noise cancellation algorithm based on hypergraph modeling, in: Digital Signal Processing Workshop Proceedings, IEEE, 1996, pp. 5–8.
[40] Alain Bretto, Hocine Cherifi, Stéphane Ubéda, An efficient algorithm for Helly property recognition in a linear hypergraph, Electron. Notes Theor. Comput. Sci. 46 (2001) 177–187.
[41] Kirthivasan Kannan, B. Rajesh Kanna, Chandrabose Aravindan, Root mean square filter for noisy images based on hyper graph model, Image Vis. Comput. 28 (9) (2010) 1329–1338.
[42] David S. Moore, Chi-Square Tests. No. Mimeograph. purdue univ lafayette ind dept of statistics, 1976.
[43] A. Sivakumar, K. Kannan, A novel feature selection technique for number classification problem using PNN—A plausible scheme for boiler flue gas analysis, Sensors Actuators B 139 (2) (2009) 280–286.
[44] Yihua Chen, Maya R. Gupta, Em demystified: An Expectation-Maximization Tutorial, University of Washington, 2010.
[45] A. Li, X. Yang, S. Kandula, M. Zhang, CloudCmp: comparing public cloud providers, in: Proceedings of the 10th Annual Conference on Internet Measurement, Melbourne, Australia, 2010.
[46] J. Schad, J. Dittrich, J. Quiane-Ruiz, Runtime measurements in the cloud: observing, analyzing, and reducing variance, Proc. VLDB Endow. 3 (1–2) (2010) 460–471.
[47] A. Iosup, N. Yigitbasi, D. Epema, On the performance variability of production cloud services, in: Proceedings of IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, CA, USA.
[48] George Karypis, et al., Multilevel hypergraph partitioning: applications in VLSI domain, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 7 (1) (1999) 69–79.

Glossary

Highlights

HGCM: Hypergraph based Computational Model
MDHP: Minimum Distance-Helly Property
CSP(s): Cloud Service Provider(s)
CU(s): Cloud User(s)
SMI: Service Measurement Index
EM: Expectation–Maximization

Abstract

SLA: Service Level Agreement
GIS: Geographic Information System

Keywords

TMS: Trust Management System

Introduction

IaaS: Infrastructure as a Service
PaaS: Platform as a Service
SaaS: Software as a Service
AWS: Amazon Web Services
GCE: Google Compute Engine
QoS: Quality of Service
MCDM: Multi-Criteria Decision-Making
CSMIC: Cloud Service Measurement Index Consortium
HPC: High Performance Computing
ICTSQ: Information and Communication Technology Service Quality
APDEX: Application Performance Index
eSCM-CL: eSourcing Capability Model-Client Organizations
ISO: International Organization for Standardization
KPI(s): Key Performance Indicators

Cloud service selection architecture

CSRD: Cloud Service Registry and Discovery
MA: Monitoring Agent
EB: Evidence Base
SRM: Service Ranking Module
IdM: Identity Management

Related works

SVD: Singular Vector Decomposition
SRS: Service Ranking System
AHP: Analytic Hierarchy Process

Building neighborhood hyperedges

$CSP_i = \{S_1, S_2, \ldots, S_S\}$: Set of CSPs
$CU_i = \{U_1, U_2, \ldots, U_U\}$: Set of CUs
A_i: SMI KPIs
A_{ij}: SMI Attributes
A_{ijk}: SMI Sub-attributes
$CSP_{i_{Prov}}$: CSPs Provision
$CU_{i_{Req}}$: CUs Requirements
$v\left(CSP_{i_{Prov}}, A_i\right)$: CSPs Provision of SMI KPIs
$v\left(CSP_{i_{Prov}}, A_{ij}\right)$: CSP Provision of SMI Attributes
$v\left(CSP_{i_{Prov}}, A_{ijk}\right)$: CSPs Provision of SMI Sub-attributes
$v\left(CU_{i_{Req}}, A_i\right)$: CUs Requirement of SMI KPIs
$v\left(CU_{i_{Req}}, A_{ij}\right)$: CUs Requirement of SMI Attributes
$v\left(CU_{i_{Req}}, A_{ijk}\right)$: CUs Requirement of SMI Sub-attributes

Ranking cloud services using MDHP algorithm

$CSP_{i_{Prom}}$: CSP Promised Value
$CSP_{i_{Act}}(A_i)$: Difference between the CSP promised value ($CSP_{i_{Prom}}$) and the provisioned value ($CSP_{i_{Prov}}$) for each elements in the SMI hierarchy
$Hyperedge_{A_i}$: Top $\frac{M}{2}$ CSPs who adhere to the CUs requirements for each SMI attributes

Algorithm

D_{EB}: Data from Evidence Base
D_{MA}: Data from Monitoring Agent
$R_i \leftarrow \{R_1, R_2, \ldots, R_L\}$: Rank of the CSPs

Pre-processing algorithm for imputing missing values

E step: Expectation Step
σ: Standard Deviation

Complexity analysis

N_L: Number of attributes at level L
n_{Li}: Number of sub-attributes at level $L-1$ of the ith attribute

Nivethitha Somu is a Full time Ph.D. Scholar at School of Computing, SASTRA University, Thanjavur, INDIA. She is a member of Ramanujan Mathematical Society (Mem. No: 1198). She received her Master's degree in Science from Anna University, Chennai, INDIA in 2011. She also received her Master's degree in Technology from SASTRA University, Thanjavur, INDIA in 2013. Her current research interests include trust management, energy aware workload consolidation and secure live migration in cloud.

Kannan Kirthivasan is a Professor in the Department of Mathematics, SARTRA University, Thanjavur, INDIA. He obtained his Bachelor's and Master's degrees from the University of Madras, India, in 1980 and 1982, respectively. He also received his Bachelor's and Master's degrees in Education from Madurai Kamaraj University, India, in 1984 and 1986 respectively. He obtained his M.Phil degree in Mathematics from Regional Engineering College, Tiruchirapalli, India, in 1988. He was conferred Ph.D. in Mathematics in the area of Computational Fluid Dynamics by Alagappa University, Karaikudi, India, in 2000. He has been in Academia for the past 25 years. His specific areas of interest include Combinatorial Optimization, Artificial Neural Networks and Hypergraph-based Image Processing.

Shankar Sriram V.S. is an Associate Professor in School of Computing, SASTRA University, Thanjavur, Tamil Nadu, INDIA. He received his Bachelor's degree in Science from Madurai Kamraj University, Madurai, INDIA in 1997. He obtained his Master's degree in Computer Applications from Madurai Kamraj University, Madurai, INDIA in 2000. He also received his Master's degree in Engineering from Thapar University, Punjab, INDIA in 2004. He was conferred Ph.D. in Information and Network Security from Birla Institute of Technology, Mesra, INDIA in 2010. He has been in the Academia for the past 15 years. His current area of research includes Information and network security, Cryptography, MANETS, Steganography and Cloud computing.