Probabilistic Qualitative Preference Matching in Long-Term IaaS Composition

Sajib Mistry[1(✉)], Athman Bouguettaya[1], Hai Dong[2], and Abdelkarim Erradi[3]

[1] School of Information Technologies, University of Sydney, Sydney, Australia
{sajib.mistry,athman.bouguettaya}@sydney.edu.au
[2] School of Science, RMIT University, Melbourne, Australia
hai.dong@rmit.edu.au
[3] Department of Computer Science and Engineering, Qatar University, Doha, Qatar
erradi@qu.edu.qa

Abstract. We propose a qualitative similarity measure approach to select an optimal set of probabilistic Infrastructure-as-a-Service (IaaS) requests according to the provider's probabilistic preferences over a long-term period. The long-term qualitative preferences are represented in probabilistic temporal CP-Nets. The preferences are indexed in a k-d tree to enable the multidimensional similarity measure using tree matching approaches. A probabilistic range sampling approach is proposed to reduce the large multidimensional search space in temporal CP-Nets. A probability distribution matching approach is proposed to reduce the approximation error in the similarity measure. Experimental results prove the feasibility of the proposed approach.

1 Introduction

IaaS providers (e.g., Amazon and Windows Azure) offer Virtual Machines (VMs) as services in a cloud market [1]. IaaS services (i.e., configurations of VMs) are usually customized to fit the requirements of consumers. Consumers (e.g., universities, governments, and Software-as-a-Service (SaaS) providers) are more likely to require long-term IaaS services according to their business goals and budget constraints. A typical IaaS request includes functional attributes, such as CPU, memory, and network units, and Quality of Services (QoSs) attributes, such as availability, throughput, response time and price [1]. *The IaaS composition is defined as to select an optimal set of custom consumer requests that maximizes the revenue and profit of the provider* [8,16].

The provider's long-term business strategies are typically qualitative in nature. For example, the provider may have a promotional strategy (discounted prices for services) in the first year. In the following years, it may have profit-maximization strategies considering the market completion. Similarly, long-term consumer requests are usually variable over a time period and qualitative in nature. For example, a consumer may prefer an IaaS service that has higher throughput in the first year. While in the second year, the consumer may find throughput is less important and may require price-sensitive services.

M. Maximilien et al. (Eds.): ICSOC 2017, LNCS 10601, pp. 256–271, 2017.
https://doi.org/10.1007/978-3-319-69035-3_18

In the *qualitative IaaS composition*, the acceptance or rejection of an incoming request should follow the business strategies of the provider as accepted requests are committed for the whole period [9]. A key limitation of exiting approaches is that the business strategies need to be deterministic, i.e., the provider should have 100% confidence to determine future changes in advance. Another limitation is that consumers are not allowed to represent their preferences in a qualitative manner.

We consider probabilistic qualitative IaaS requests from the consumers and probabilistic qualitative business strategies of the providers in the long-term composition. Here, consumers provide their probabilistic IaaS service requests based on their predicted business needs. For example, a university may calculate the required IaaS services based on the number of students and staffs for the first year. However, there is a 40% chance that the number of students and staffs will increase in two folds in the second year. Hence, there is 60% probability that the consumer's preference will remain similar and 40% probability to be changed in the second year. Similarly, providers' business strategies are constructed based on different environment variables such as available resources and number of consumers. For example, business strategies are constructed assuming a fixed size of resources for a long-term period in [9]. However, such an assumption is hardly applicable in the real world as available resources tend to be probabilistic rather than deterministic. For example, the provider may invest in increasing new resources or sell a part of existing resources to other providers in the following years [5]. Similarly, the future demand for IaaS services is probabilistic in nature and hard to predict with 100% confidence.

We assume that an IaaS provider has already developed its long-term probabilistic qualitative service delivery preferences. It receives different long-term probabilistic qualitative service requirements from different consumers. Note that, how the probabilities are determined is out of the scope of this paper. *Our target is to find the optimal set of requests where their probable preferences are best matched with the provider's uncertain preferences.* We have identified the following research challenges in the probabilistic long-term IaaS composition:

- **Probabilistic temporal preference representation:** We require not only an intuitive tool for structuring the provider's qualitative preferences but also a support for assigning temporal transition probabilities. For example, a provider may prefer providing CPU based services over Network based services in the first year. In the second year, there is 80% probability that provider will stick to its existing preference order, but there is 20% probability to deliver services with a different preference order. The semantics of preferences may not be static during the whole period of composition. For example, 10ms response time is treated as a high QoS in this year, but it may become a moderate QoS in the next year due to an upgrade of the hardware in the market.
- **Probabilistic qualitative similarity measure:** Upon receiving probabilistic qualitative temporal preferences from the consumers, we have to quantify their similarity measure with the provider's temporal qualitative preferences.

However, as we are considering long-term composition, each time segments should have several probable temporal preferences and each of them may have several probable temporal preferences in the next temporal segments. Hence, the number of temporal sequences or orders of preferences might be large for the whole composition period. It is computationally inefficient to compare every pair of sequences for the similarity measure. We require a probabilistic similarity matching approach that can approximate to the optimal result using fewer number of comparisons.

We represent preferences in probabilistic Temporal CP-Nets (PrTempCP-Net), where dynamic TempCP-Nets have a transition probability matrix among composition intervals. The dynamic semantics of the preferences are indicated using a Conditional Preference Table (CPT) [3] of the PrTempCP-Net. We assume that the dynamic semantics of preferences are global across the consumers for simplicity. However, we transform semantics of consumers' preferences to match the dynamic semantics of the provider's preferences and apply composition aggregation rules [16] for the similarity measure. Moreover, the induced preference graph [13] from TempCP-Net is indexed in a multidimensional k-d tree [2] to effectively match with the attributes of the consumer preferences.

Although long-term IaaS composition is a preference maximization combinatorial optimization problem [8], *we only focus on probabilistic similarity matching approach in the composition.* We apply a brute force approach to generate all possible combinations of IaaS requests. Instead of comparing all preference sequences in the PrTempCP-Net (a computationally inefficient matching process), we propose a novel probabilistic range sampling approach. The ranges are selected in a way so that they approximate to an optimal spectrum of similarity deviations from any random preference sequences matching. We use the Kolmogorov-Smirnov test (K-S test) [4] as a statistical distribution matching algorithm to determine the weight of a given preference range in the similarity measure. The weighted similarity measures of all the preference range samples are aggregated to determine the highest matched, i.e., the optimal set of requests.

2 Related Work

Graphical models are proposed to represent user-preferences where relative ordering among preference attributes are determined by economic variables such as cost and profit [14]. A Conditional Preference Network (CP-Net) [3] is a dependency graph that represents consumers' preferences qualitatively. A CP-Net based graphical model is proposed for the service composition from the consumers' perspective [12]. The composition approach from incomplete consumer preferences [13] performs preference amendment, i.e., the similar consumer detection and historical preference voting. Graph based similarity measure are applied to find the optimal composition in web service compositions [7]. A deterministic temporal CP-Net is proposed to represent the provider's long-term qualitative preferences [9]. *To the best of our knowledge, exiting research does not consider probabilistic qualitative preferences in the long-term IaaS composition.*

Cluster sampling and multistage sampling are applied to generalize the results to the target population [10]. Convenience sampling and probabilistic range sampling are nonprobability sampling techniques which approximate a sample of subjects/units from a population [15]. It is useful especially when randomization is impossible like when the population is very large [15]. Kolmogorov-Smirnov (K-S) test is efficient to measure the similarity between the probability distributions of two samples [4]. *To the best of our knowledge, statistical analysis is yet to be applied to reduce the large search space and to perform similarity measure in the probabilistic qualitative IaaS composition.*

3 Motivation: Probabilistic Qualitative IaaS Composition

Let us assume, a new IaaS provider starts offering virtual CPU services associated with QoS of availability for simplicity. Consumer A and B are interested in using services from the provider. We assume that both the provider and consumers have same semantic interpretation of the qualitative preferences on CPU, availability, and price for simplicity. We represent the semantic levels as high, moderate, and low, as shown in Fig. 1(a).

The CP-Net can elegantly represent these qualitative preferences. For example, an arc from "CPU" to "availability" means the preference of "availability" depends on the preference of "CPU" units. The provider may have different business strategies represented in CP-Nets. For example, the provider prefers to provide high-quality services with relatively lower prices, to build its reputation in the market. Hence, the provider decides that "availability" of a service is the most important attribute, followed by "CPU" and "price". $CP1$ is the corresponding CP-Net for reputation building (Fig. 1(b)). In $CP1$, the "high" availability has a higher priority than the "moderate" availability, i.e., $A1 \succ A2$. Note that, the "low" availability ($A3$) is not in the provider's preference in $CP1$. The choice of availability dictates the choice of CPU units. Finally, the price of the service is chosen based on the selection of the levels of availability and CPU units. As this is a reputation building phase, the provider will not charge "high" price ($P1$) while providing "moderate" CPU units ($C2 : P2 \succ P3$). In $CP1$, the most preferred service provision is $(A1, C1, P1)$ and the least preferred choice is $(A2, C1, P3)$. Similarly, $CP2$ and $CP3$ capture the profit maximization and risk management strategies respectively (Fig. 1(b)). In $CP2$, the most preferred service provision is $(P1, C3, A3)$ and the least preferred service is $(P2, C2, A2)$ expressing the preference on the higher price. In $CP3$, the most preferred service provision is $(C3, P1, A3)$ and the least preferred service provision is $(C2, P3, A3)$.

Consumers may have different qualitative preferences represented in CP-Nets based on their requirements. In Fig. 1(c), $CP4$ captures the "availability sensitive" preferences. Here, consumers do not prefer "low" availability and are able to pay "high" price for "high" availability and CPU units. $CP5$ captures the "price sensitive" preferences where consumers do not prefer "high"-priced services and are satisfied with"low" CPU and availability if the service price is "low". $CP6$ captures the "CPU sensitive" preferences. Here, consumers do not

prefer "low" CPU and are able to pay "high" price for "high" CPU units and availability. $CP7$ also captures the "availability sensitive" preferences. It decides CPU and price values based on "low to moderate" availability preferences. In $CP7$, the highest preferred service is $(A2, C1, P2)$ and the least preferred service is $(A3, C1, P3)$.

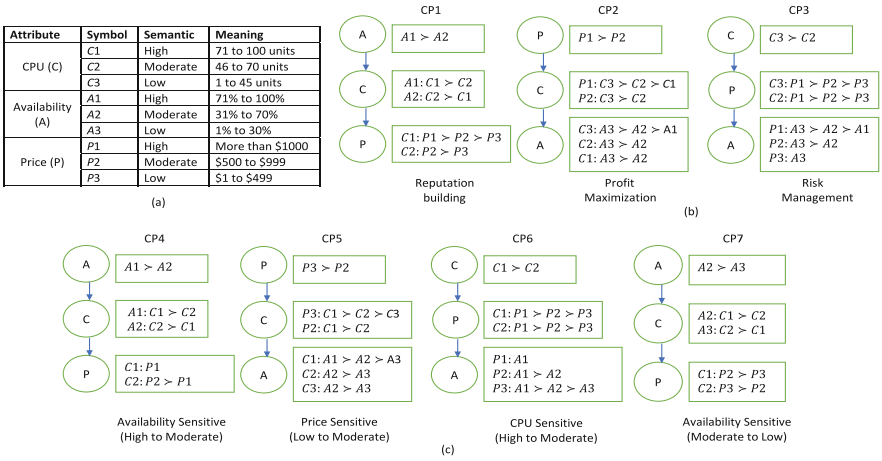
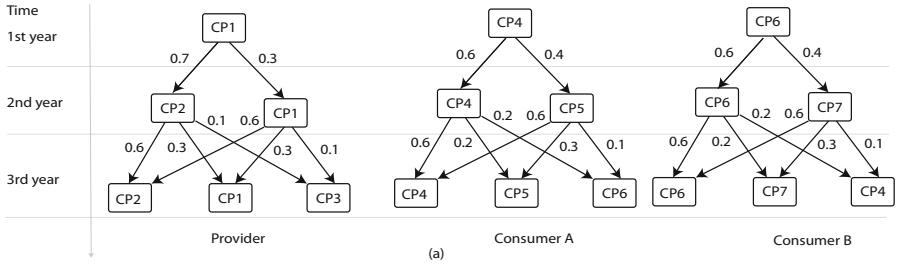


Fig. 1. (a) Semantic representation of preferred service attributes, (b) A provider's qualitative preferences, (c) Consumers' qualitative preferences

The provider's business strategies probably change in the long-term period (Fig. 2(a)). For example, the provider is determined to apply the reputation building strategy ($CP1$) in the first year. In the second year, the profit maximization strategy ($CP2$) has a 60% chance to be applied, because the number of consumers may turn out lower than expected. Hence, there is 40% probability to continue the reputation building strategy ($CP1$) from the first year. Similarly, the risk management strategy ($CP3$) has a 10% chance to be applied in the third year due to possible hardware failures in the aging infrastructure. Uncertainties around consumers' qualitative preferences are also a natural phenomena in a long-term period (Fig. 2(a)). For example, consumer A may forecast a 60% chance of using "availability-sensitive" services ($CP4$) only for the three year period. It also predicts that there is a 40% chance to use "price-sensitive" services ($CP4$) due to a possible economic recession in business. The temporal changes in qualitative preferences and their transition probabilities from one CP-Net to another CP-Net are captured in a probabilistic temporal CP-Net model denoted as PrTempCP-Net. In Fig. 2(a), the provider uses $\{CP1, CP2$ and $CP3\}$, the Consumer A uses $\{CP4, CP5$ and $CP6\}$, and Consumer B uses $\{CP6, CP7$ and $CP4\}$ to build their PrTempCP-Nets.

Here, all possible compositions in a brute force manner are $\{A\}$, $\{B\}$ and $\{A, B\}$. The aggregated CPU and availability requirements of Consumer A and



(a)

Probable run-time temporal preference sequences			Similarity Index	
Provider (P)	Consumer A	Consumer B	Similarlity (P,A)	Similarlity (P,B)
1st: CP1, CP2, CP2	1st: CP4, CP4, CP4	1st: CP6, CP6, CP6	High	Almost High
1st: CP1, CP2, CP2	2nd: CP4, CP4, CP5	2nd: CP6, CP6, CP7	Low	Moderate
2nd: CP1, CP2, CP1	1st: CP4, CP4, CP4	1st: CP6, CP6, CP6	High	Almost High
2nd: CP1, CP2, CP1	2nd: CP4, CP4, CP5	2nd: CP6, CP6, CP7	Low	Moderate

(b)

Fig. 2. (a) Probabilistic Temporal CP-Nets (b) Runtime similarity index

B are greater than the provider's maximum resource limit, we select either $\{A\}$ or $\{B\}$ as the best composition. Each PrTempCP-Net in Fig. 2(a) has 6 sequences of CP-Nets with different probabilities. First, we apply a greedy approach and match the highest probable sequences (Provider: $(CP1, CP2, CP2)$, Consumer A: $(CP4, CP4, CP4)$, Consumer B: $(CP6, CP6, CP6)$) in Fig. 2(b). Here, $CP2$ is highly matched with $CP4$ and $CP6$ as consumers are able to pay "high" prices for "high" availability and CPU units. However, $CP1$ is better matched with $CP4$ (higher availability in both preferences) than $CP6$ (CPU-sensitive preferences). Hence, A is better matched (high) than B (almost high) for highest probable CP-Net sequences. Next, we compare the provider's highest sequences $(CP1, CP2, CP2)$ with second highest sequences of Consumer A $(CP4, CP4, CP5)$ and Consumer B $(CP6, CP6, CP7)$. As $CP5$ does not prefer higher priced services, but $CP2$ does prefer the opposite, the similarity measure between the provider and Consumer A is lower than the similarity measure between the provider and Consumer B. The similarity measure of the first two probable sequences are described in Fig. 2(b). Although $\{A\}$ is best matched with the highest probable sequences, B has the best averaged similarity in all the sequences (it never goes low in similarity measure). Hence, the greedy approach may not be applicable in runtime. If there are m CP-Nets and t time segments in a PrTempCP-Net, $O(m^{t^2})$ are required to find the optimal composition. It may not be feasible to compare all the sequences for large m and t values. *Hence, we apply probabilistic statistical sampling and matching techniques to reduce the search space in runtime.*

4 Probabilistic Temporal CP-Net

We require not only an intuitive tool for structuring the probabilistic qualitative preferences, but also a support for a matching process. We model the long-term

preferences as probabilistic temporal CP-Net (PrTempCP-Net). PrTempCP-Net is defined as 6-tuple $< V, M, N, I, I_0, P(.,.) >$ where:

- $V = \{X_1, ..., X_n\}$ represents a set of functional and non-functional attributes. Typical functional attributes are CPU (C), Network bandwidth (NB), and Memory (M), and QoS attributes are Availability (A), Response time (RT), Throughput(TP) and Price(P).
- $M = \{CP_1, CP_2, ..., CP_m\}$ is a finite set of CP-Nets. A CP-Net in the interval I_k, CP^{I_k} is a directed graph G over V whose nodes are annotated with conditional preference tables $CPT(X_i)$ for each $X_i \in V$. Each conditional preference table $CPT(X_i)$ describes the qualitative preferences over the values of the variable X_i given every combination of parent values. For example, in $CP1$, the $CPT(C)$ contains $\{A1, A2\}$ while preferences are made over $\{C1, C2\}$ (Fig. 1(b)). A CP-Net generates a total ordered ($\succeq$) preference ranking over the set of service configurations: $o_1 \succeq o_2$ means that a configuration o_1 is equally or more preferred than o_2. We use $o_1 \succ o_2$ to denote the fact that provisioning or consuming service o_1 is more preferred than o_2 (i.e., $o_1 \succeq o_2$ and $o_2 \not\succeq o_1$), while $o_1 \sim o_2$ denotes that the provider's or consumers' preference is indifferent between the configurations o_1 and o_2 (i.e., $o_1 \not\succeq o_2$ and $o_2 \not\succeq o_1$).
- $N = \{Sem\_Table_1, Sem\_Table_2, ..., Sem\_Table_n\}$ is a finite set of semantic tables. Sem_Table_k represents the k^{th} semantic interpretations over ranges of the variable X_i. Figure 1(a) is such a semantic table that maps 70–100 units of CPU as a "high" CPU value.
- $I = \{I_1, I_2, ..., I_t\}$ is the finite set of intervals. Here, the total composition time, T is divided into t intervals where, $T = \sum_{i=1}^{t} I_i$.
- I_0 represents the starting interval in the matching process of a composition which is defined by the provider or consumer.
- $P(CP_s, Sem\_Table_s, I_s | \acute{CP}_s, Sem\_\acute{T}able_s, \acute{I}_s)$ is the probability to choose a particular service preference CP_s in interval I_s with the corresponding semantic table (Sem_Table_s) from service preference $\acute{CP}_s$ which is applied in interval $\acute{I}_s$ with the semantic table $Sem\_\acute{T}able_s$. We assume that all probabilities are generated before the composition. In Fig. 2(a), the probability to transit from ($CP1$, first year) to ($CP2$, second year) is 0.7.

A probabilistic TempCP-Net produces different deterministic TempCP-Nets based on I_0. A deterministic TempCP-Net is generated by applying transition probabilities to a CP-Net in an interval. Usually, the matching process is performed from left to right, i.e., first interval to second interval and so on. Here, the first interval is set as I_0. For example, {($CP1$, 1st Year), ($CP2$, 2nd year), ($CP2$, 3rd year)} is the highest probable deterministic TempCP-Net as the transition probabilities are 0.7 and 0.7 respectively. The set of consequences $o \succ \acute{o}$ of an acyclic TempCP-Net constitutes a partial order over the service configuration. This partial order can be represented by an acyclic directed graph, referred to as the *induced preference graph*. The nodes of the induced preference graph correspond to the complete assignments to the variables of the network. There is an edge from node $\acute{o}$ to node o iff the assignments at $\acute{o}$ and o differ only in the

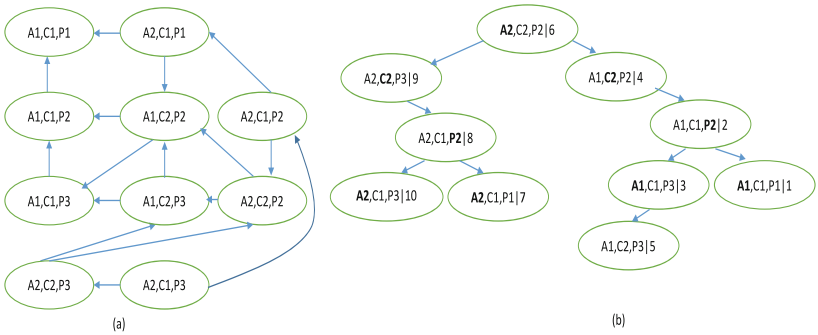


Fig. 3. (a) Induced preference model, (b) k-d tree indexing

value of a single variable X. Given the values assigned by *ó* and o to $Pa(X)$, the value assigned by o to X is preferred to the value assigned by *ó* to X. Figure 3(a) depicts the induced preference graph of $CP1$. There is no outgoing edge from $(A1, C1, P1)$ as it is the most preferred request configuration. Similarly, there is no incoming edge to $(A2, C1, P3)$ as it is the least preferred configuration. If n is the number of attributes in the TempCP-net and q is the number of output configurations in an interval, the time complexity for ordering queries in an interval is $O(nq^2)$.

5 TempCP-Net Matching Using *k*-d Tree Indexing

First, we perform similarity measure between two deterministic TempCP-Nets, $A = \{(CP_A^1, Sem\_Table_A^1, I_1), ..., (CP_A^m, Sem\_Table_A^m, I_m)\}$ and $B = \{(CP_B^1, Sem\_Table_B^1, I_1), ..., (CP_B^m, Sem\_Table_B^m, I_m)\}$. We consider it as a base to match probabilistic TempCP-Nets. We assume that the temporal lengths of the CP-Nets are same in each TempCP-Nets. CP-Nets within the same interval are matched and the similarity measure is averaged over the number of intervals (m) as follows:

$$Sim(A, B) = \frac{\sum_{i=1}^{m} Sim_{A,B}(CP_A^i, Sem\_Table_A^i, CP_B^i, Sem\_Table_B^i)}{m} \quad (1)$$

The induced preference graph enables similarity measure between two CP-Nets constructed with the same semantic table (i.e., $Sim_{A,B}$). Each tuple $(s_1, ..., s_n)$ in the induced preference graph of CP_A^i is linearly traversed over the induced preference graph of CP_B^i. The similarity measure is then defined as the averaged number of traversals required to search all tuples (time complexity $O(n^2)$) [12]. Here, a lower value indicates a higher similarity index. Considering the tuple $(s_1, ..., s_n)$ as a multidimensional vector, we improve the matching process using the k-d tree [2]. The k-d tree is a binary tree in which every node is a k-dimensional point (Fig. 3(b)). Every non-leaf node can be thought of as implicitly generating a splitting hyperplane that divides the space into two parts, known

as half-spaces. Points on the left and right sides of this hyperplane are represented by the left and right subtree of that node respectively. We use the canonical method to construct the k-d tree [2]:

- The selection of splitting planes follows a cycle as the construction algorithm moves down on the tree. For example, in Fig. 3(b), the root is an "Availability-aligned" plane, the root's children both have "CPU-aligned planes", the root's grandchildren have "Price-aligned" planes, the root's great-grandchildren have again "Availability-aligned" planes, and so on.
- As all the n points are available from the induced preference graph, we insert points by selecting the median of the points being put into the subtree, with respect to their coordinates in the axis being used to create the splitting plane. This would result in a balanced k-d tree construction in $O(n\ log(n))$ times [2]. Each node in the k-d tree is annotated with its respective preference order from the induced graph. For example, the root node $(A2, C2, P2)$ is annotated with the preference ranking 6 in Fig. 3(b).

At first, CP^i_A and CP^i_B are indexed in corresponding k-d trees. We apply semantic transformation to one of the k-d trees as follows:

- **Semantic Transformation:** if $Sem\_Table^i_A \neq Sem\_Table^i_B$ and the average value of "high" semantics in $Sem\_Table^i_A$ is greater than the average value of "high" semantics in $Sem\_Table^i_B$, semantic transformation is applied to CP^i_B. The average value of a semantic "X" in range $[a, b]$ is calculated as $Avg(X) = \frac{(a+b)}{2}$. For all "X" in CP^i_B, if $Avg(X)$ is within the range $[\acute{a}, \acute{b}]$ of a semantic "Y" in $Sem\_Table^i_A$, "X" is replaced with "Y" in CP^i_B. If $Avg(X)$ is below the "low" semantic in $Sem\_Table^i_A$ (i.e., no range found), "X" is replaced with "low" in CP^i_B. For example, if "high" availability ($A1$) of CP^i_A is ranged in [80,100] (avg. 90) and "high" availability ($A1$) of CP^i_B is ranged in [60,90] (avg. 75), $A1$ of CP^i_B is semantically transformed to "moderate" $A2$ as it is ranged in [60,80] in $Sem\_Table^i_A$.

We start the matching process $Sim_{AB}(CP^i_A, CP^i_B)$ using the indexed k-d trees after the semantic transformation. We consider each tuple $(s_1, ..., s_n)$ of CP^i_B as search points. Starting with the root node of CP^i_A, a search point (ranking r_b) moves down on the tree recursively, in the same way that it would if the search point was being inserted. If the search point is matched with a node, it returns the annotated ranking value, r_a. For example, the search point $(A2, C1, P3)$ of rank 1 in CP^i_B returns rank 10 in CP^i_A using only 4 comparisons. A non-matched search point returns L which is a large number indicating the lowest ranking. The normalized difference between r_a and r_b indicates a similarity measure (Eq. 2). In the previous example, it indicates a dissimilarity as the non-negative normalized difference between r_a and r_b is 0.9. $Sim_{AB} = 0$ indicates the highest match and $Sim_{AB} = 1$ indicates the lowest match, i.e., dissimilarity. The time complexity of the k-d tree based similarity measure in an interval is $O(n\ log(n))$.

$$Sim_{AB}(CP^i_A, CP^i_B) = \frac{abs|r_a, r_b|}{max(r_a, r_b)} \mid \forall\ r_a \in CP^i_A \text{ and } r_b \in CP^i_B \qquad (2)$$

6 Similarity Measure Between PrTempCP-Nets

The similarity measure between probabilistic temporal CP-Nets should be reflective of a matching between runtime temporal CP-Nets. Let us assume there are two probabilistic temporal CP-Nets (PA and PB) and two random deterministic or runtime tempCP-Nets, A and B are generated from PA and PB respectively. If $Sim(A, B) = \alpha$, then the similarity measure between PrCP-Nets, $Sim(PA, PB) = \beta$ indicates that the difference $|\alpha - \beta|$ has a higher probability to be less than the standard deviation. Based on the prediction of the possible runtime CP-Nets, two approaches could be applied for the similarity measure between PrTempCP-Nets:

- *Greedy approach:* The most runtime likelihood sequences of CP-Nets are generated from PA and PB and are matched using Eqs. 1 and 2 in this approach. We define the following recursive procedure:
  1. Base case: $O(0) = \{\phi\}$ denotes the empty sequence at no interval and the total probability $TP(0) = 1$.
  2. Recursion: $O(n) = \{CP^n, O(n-1)\}$ denotes the maximum likelihood sequence where, $TP(n) = P(X, CP^{n-1}) \times TP(n)$ is maximum for $X = CP^n$.
- *Brute force approach:* It is not guaranteed that the similarity measure with the greedy approach has a higher probability to be less than the standard deviation from all possible sequences in PrTempCP-Nets. Hence, the brute force approach generates all possible sequences of deterministic TempCP-Nets from PA and PB and perform pair-wise similarity measure using Eqs. 1 and 2. If q is the total number of comparisons, the probabilistic similarity measure is calculated as the averaged mean value:

$$Sim(PA, PB) = \frac{\sum_{i=1}^{q} Sim_i(A, B)}{q} \mid \forall\ A \in PA,\ B \in PB \tag{3}$$

We apply statistical analysis and sampling techniques to reduce the large number of comparisons in the brute force approach and to approximate the similarity measure within the standard deviation. The approach consists of two steps: (a) *probabilistic range sampling* to compress CP-Nets into fewer numbers, (b) *reducing approximation error* by applying deviations in probability distributions using the K-S test.

6.1 Probabilistic Range Sampling of PrTempCP-Net

Stratified sampling is an effective technique where the solution space embraces a number of distinct categories, the whole solution space can be organized into separate "strata" [15]. Each stratum is then sampled as an independent subspace, out of which individual elements can be randomly selected [15]. Due to different probability distributions in a PrTempCP-Net, we can apply stratified sampling to compress CP-Nets into fewer numbers, where each "starta" is a

probability range. We create the set of m probability ranges, denoted as RG, where each interval in a range is $\frac{1}{m}$. If $m = 5$, the set of probability ranges are $\{[0, 0.2), [0.2, 0.4), [0.4, 0.6), [0.6, 0.8), [0.8, 1.0)\}$. For each probability range in RG, we compress CP-Nets with the same probability interval. Total $|RG|$ numbers of deterministic TempCP-nets are created from a PrTempCP-Net. Given a probability range $[x, y]$, we apply weighted aggregation to compress the CP-Nets. For each interval I in a PrTempCP-Net, we filter CP-Nets where their probabilities are within the range $[x, y]$. For example, if the probability range is [0,0.4], the filtered provider's PrTempCP-Net is $\{(CP1, I_1), (CP1, I_2), (CP1, CP3, I_3)\}$ in Fig. 2. Note that, $CP2$ is excluded because its probabilities is out of the range [0,0.4].

We aggregate CP-Nets as a compression mechanism in each interval to create the deterministic TempCP-Nets. Pairwise aggregation order is applied for multiple CP-Nets. For example, CP^a, CP^b, and CP^c are aggregated as $((CP^a + CP^b) + CP^c)$. The aggregation procedure of CP^a and CP^b along with associated probabilities P^a and P^b uses tuple aggregation rules [9] as follows:

1. CP^a and CP^b are transformed into their corresponding k-d trees where each node is a tuple $(x_1, x_2, ..., x_n)$.
2. Select tuples from the same level of the k-d trees. For example, both roots of k-d trees are selected in the first level. If N tuples are selected, we apply the following weighted summation rule for resource attributes (x) and weighted maximization rule for QoS attributes (y):

$$\text{Summation: } \bar{x_i} = \sum_{i=1}^{N} P^i \times x_i, \text{where } x_i \in \{C, M, NB, RT, P, Rank\} \quad (4)$$

$$\text{Maximization: } \bar{y_i} = max(P^i \times y_i), \forall\ i \in [1, N] \text{ where } y_i \in \{A, TP\}$$

3. Starting from the first level, the aggregation is performed in every level and the corresponding ranking in attached with each tuple.

The m probability ranges are applied in both PA and PB. If P^i_{mean} is the mean probability of the i^{th} probability range, the similarity measure is calculated as follows:

$$Sim(PA, PB) = \frac{\sum_{i=1}^{m^2}(P^i_{mean} \times Sim_i(A, B))}{m^2} \mid \forall\ A \in PA,\ B \in PB \quad (5)$$

6.2 Reducing Approximation Error in Sim(PA, PB) Using K-S Test

The similarity measure between TempCP-Nets with higher probability ranges is given higher weight in the computation of total similarity measure between PrTempCP-Nets in Eq. 5. It is based on the heuristic that the probability distribution of attributes in a higher probability range TempCP-Net is significantly greater than the probability distribution of attributes in a lower probability range TempCP-Net. Hence, a change in the probability distribution may not change

the similarity measure in the runtime. For example, the similarity measure with the probability range [80,100] is not expected to change to a similarity measure with the probability range [0,20] in runtime. However, such heuristic may not be applicable when probability distributions of TempCP-Nets are close to each other.

We apply Kolmogorov-Smirnov test (K-S test) [4] to find the closeness of probability distribution of TempCP-Nets which are filtered with probabilistic range sampling. Given two TempCP-Nets A and B and an attribute x, we first derive the cumulative probability distribution $F_A(x)$ and $F_B(x)$. *The null hypothesis is that both the preferences are generated by the same distribution.* The null hypothesis is tested in the K-S test with two values $L_{m,n}$ and $L_{m,n,\alpha}$ defined in Eq. 6. Here $L_{m,n}$ is the maximum difference in the cumulative distribution functions and $L_{m,n,\alpha}$ is the critical value from Kolmorogov distribution functions [4]. α is the confidence level to reject the null hypothesis. According to the recommendation in [4], we reject the null hypothesis (at significance level α) if $L_{m,n} > L_{m,n,\alpha}$. For example, $\alpha = 0.05$ gives 95% confidence to reject the null hypothesis.

$$
\begin{aligned}
&L_{m,n} = max_x|F(x) - G(x)| \\
&L_{m,n,\alpha} = c(\alpha)\sqrt{\frac{m+n}{mn}} \\
&c(\alpha) = \text{the inverse of the Kolmorogov distribution at } \alpha
\end{aligned}
\tag{6}
$$

Let us assume, $Avg_A(\alpha)$ is the averaged similarity measure in the pairwise probability distributions between A and rest of the TempCP-Nets. A higher $Avg_A(\alpha)$ indicates that A is highly similar with other distributions and it has higher chance to change in runtime. Hence, the initial probability which is attached to A should consider such changing probability to reduce the approximation error. Hence, we update the Eq. 5 using (α) as follows:

$$
Sim(PA, PB) = \frac{\sum_{i=1}^{m^2} \frac{P^i_{mean} \times Sim_i(A,B)}{max(Avg_A(\alpha), Avg_B(\alpha))}}{m^2} \mid \forall\ A \in PA,\ B \in PB \tag{7}
$$

7 Experiments and Results

As our focus is not on the optimization of IaaS composition, the optimal composition is selected by the brute-force combinatorial optimization. It generates all combinations of consumers' PrTempCP-Nets along with the brute-force similarity measure. The brute-force similarity measure compares all possible TempCP-Nets with the provider's PrTempCP-Net. We compare the efficiency of the proposed similarity measure with the greedy approach to find the optimal composition in a fewer number of comparisons between TempCP-Nets. All the experiments are conducted on computers with Intel Core i7 CPU (2.13 GHz and 4GB RAM). Java is used to implement the algorithms.

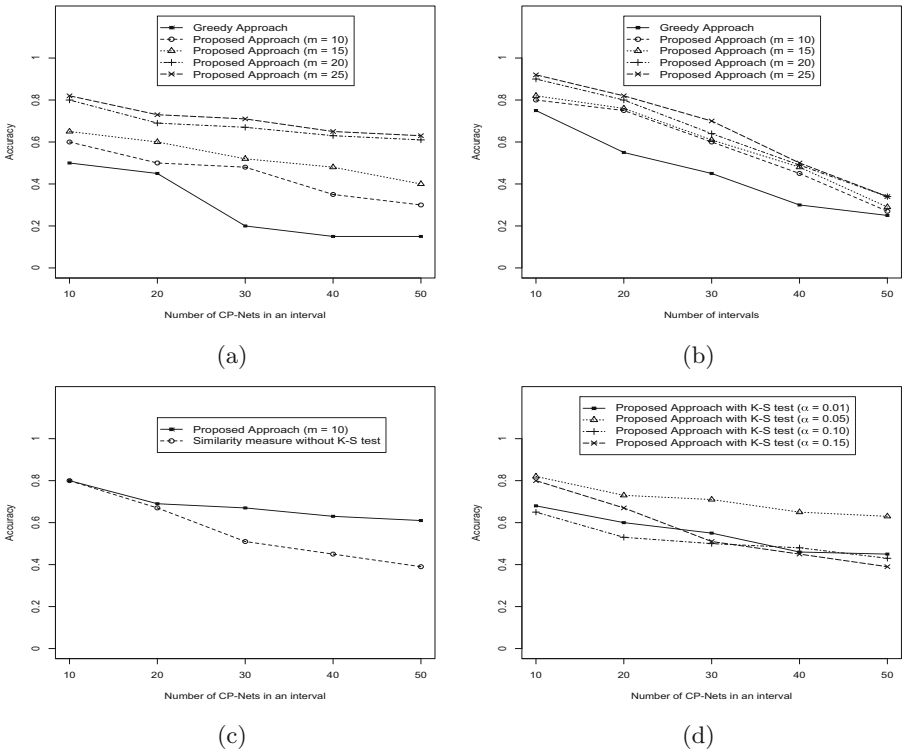


Fig. 4. (a) Accuracy in different m values, (b) Accuracy in scalable intervals, (c) Importance of K-S tests, (d) Accuracy in different α values

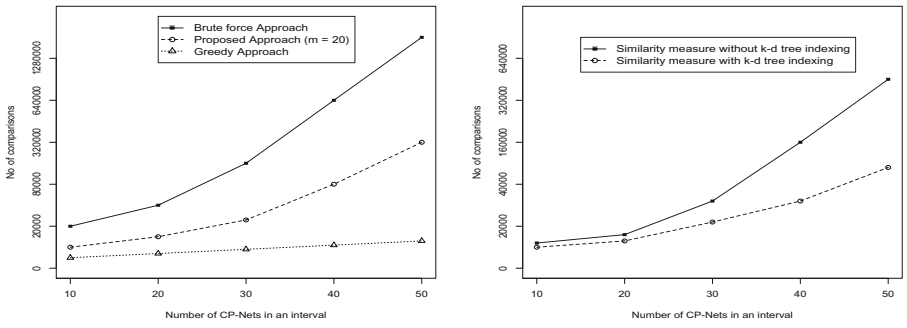
7.1 Data Description

We create the PrTempCP-Nets using Google Cluster resource utilization [11], real world cloud QoS performance [6], and synthetic price. Google Cluster data include CPU and Memory utilization and allocation time series of 70 jobs over a 1-month period. The real world QoS data [6] include two time series (i.e., response time and throughput) for 100 cloud services over a 6-month period. We randomly pick 70 Google Cluster jobs and make one-to-one mapping with the 100 sets of QoS data. A 6-month request is extended to a 12-month request using ARIMA model [17] with a confidence score. We create 50 such long-term requests from one Google Cluster job with random confidence scores in the range (0, 100]. Each TempCP-Net has 12 monthly intervals and each interval contains different CP-Nets where dependencies among the attributes are randomly generated from the same segment of 50 long-term requests. The probabilities in the transition matrix are mapped with the confidence scores which are used to generate the long-term preferences. The generated PrTempCP-Nets are separated into 10 groups (G_1 to G_{10}). In a group, a random PrTempCP-Net is considered as the

provider's business strategy and the rest 6 PrTempCP-Nets are considered as consumers' preferences. For the K-S test, we set $\alpha = 0.05$.

7.2 Efficiency of the Proposed Probabilistic Range Sampling

We consider the brute force similarity measure as our baseline. Let us assume, s optimal compositions are returned from m groups by the brute force approach. However, r compositions are optimal from the m returned compositions using greedy or the proposed approach. Hence, we compute the accuracy of a similarity measure as $\frac{r}{s}$ in the range [0,1]. Here, 1 means the perfect accuracy. Figure 4(a) depicts the accuracy of the proposed probabilistic sampling with different numbers of probability ranges (m). We find that the proposed approach is more accurate when higher numbers of probability ranges are used to sample. There are no significant improvement in accuracy after $m = 20$. The greedy approach performs similar to the proposed approach when the number of CP-Nets is lower in the PrTempCP-Net. We find that the proposed approach is significantly accurate than the greedy approach for higher numbers of CP-Nets in Fig. 4(a). Figure 4(b) depicts the scalablity of the proposed approach in long-term compositions. We find that the accuracy is relatively lower when the number of intervals is increased. The proposed approach does not perform better than the greedy approach when the number of intervals are set to 50. If each interval represents a month, the proposed approach is applicable in a 4-year long composition which is acceptable in the real world. Figure 4(c) depicts the importance of reducing approximation error using K-S tests. We find that K-S tests are unnecessary when the number of CP-Nets is lower in an interval. However, it improves the accuracy significantly for a higher number of CP-Nets. Figure 4(d) depicts the importance of appropriate significance level (α) in K-S tests in the proposed similarity measure. We find that $\alpha = 0.5$, i.e., 95% confidence interval is appropriate as it maximizes the similarity measures than other values.



(a) (b)

Fig. 5. (a) Time complexity, (b) Significance of k-d tree indexing

7.3 Time Complexity Analysis

Although the brute force approach is more accurate, it is not appreciable in runtime due to its exponential nature (Fig. 5(a)). We find that the greedy approach is the most time efficient which is linearly correlated with the number of CP-Nets in an interval. However, the time complexity of proposed approach is quadratic and related to the value of m. For $m = 20$, the proposed similarity measure takes around 75% less time than the brute force approach. Figure 5(b) depicts the importance of k-d tree indexing. For a large number of CP-Nets in TempCP-nets, k-d tree reduces the number of comparisons by the factor $\frac{log(n)}{n}$.

8 Conclusion

We represent the long-term qualitative preferences using a novel probabilistic temporal CP-Nets in the IaaS composition. We propose sampling and probabilistic distribution matching in the similarity measure between PrTempCP-Nets. Although the greedy approach is the most time-efficient, the proposed approach is significantly accurate than the greedy approach and has an acceptable runtime efficiency. In the future work, we explore an efficient optimization process in relation with PrTempCP-Nets.

Acknowledgements. This research was made possible by NPRP 7-481-1-088 grant from the Qatar National Research Fund (a member of The Qatar Foundation). The statements made herein are solely the responsibility of the authors.

References

1. Armbrust, M., Fox, A., Griffith, R.: Above the clouds: a berkeley view of cloud computing. Technical Report. University of California, Berkeley (2009)
2. Bentley, J.L.: Multidimensional binary search trees used for associative searching. Commun. ACM **18**(9), 509–517 (1975)
3. Boutilier, C., Brafman, R.I., Domshlak, C., Hoos, H.H., Poole, D.: Cp-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. J. Artif. Intell. Res. **21**, 135–191 (2004)
4. Fasano, G., Franceschini, A.: A multidimensional version of the kolmogorov-smirnov test. Roy. Astron. Soc. **225**(1), 155–170 (1987)
5. Goiri, Í., Guitart, J., Torres, J.: Economic model of a cloud provider operating in a federated cloud. Inf. Syst. Front. **14**, 827–843 (2012)
6. Jiang, W., Lee, D., Hu, S.: Large-scale longitudinal analysis of soap-based and restful web services. In: Proceedings of ICWS, pp. 218–225 (2012)
7. Limthanmaphon, B., Zhang, Y.: Web service composition with case-based reasoning. In: Proceedings of ADC, pp. 201–208. ACS (2003)
8. Mistry, S., Bouguettaya, A., Dong, H., Qin, A.K.: Metaheuristic optimization for long-term IaaS service composition. IEEE TSC **PP**(99), 1 (2016)
9. Mistry, S., Bouguettaya, A., Dong, H., Erradi, A.: Qualitative economic model for long-term IaaS composition. In: Sheng, Q.Z., Stroulia, E., Tata, S., Bhiri, S. (eds.) ICSOC 2016. LNCS, vol. 9936, pp. 317–332. Springer, Cham (2016). doi:10.1007/978-3-319-46295-0_20

10. Puzicha, J., Hofmann, T., Buhmann, J.M.: A theory of proximity based clustering: Structure detection by optimization. Pattern Recogn. **33**(4), 617–634 (2000)
11. Reiss, C., Wilkes, J., Hellerstein, J.L.: Google cluster-usage traces: format + schema, Technical report. Google Inc., Mountain View, CA, USA (2011)
12. Santhanam, G.R., Basu, S., Honavar, V.: Web service substitution based on preferences over non-functional attributes. In: Proceedings of SCC, pp. 210–217 (2009)
13. Wang, H., Shao, S., Zhou, X., Wan, C., Bouguettaya, A.: Preference recommendation for personalized search. Knowl.-Based Syst. **100**, 124–136 (2016)
14. Wang, H., Zhang, J., Sun, W., Song, H., Guo, G., Zhou, X.: WCP-nets: a weighted extension to CP-nets for web service selection. In: Liu, C., Ludwig, H., Toumani, F., Yu, Q. (eds.) ICSOC 2012. LNCS, vol. 7636, pp. 298–312. Springer, Heidelberg (2012). doi:10.1007/978-3-642-34321-6_20
15. Wang, J.F., Stein, A., Gao, B.B., Ge, Y.: A review of spatial sampling. Spat. Stat. **2**, 1–14 (2012)
16. Ye, Z., Bouguettaya, A., Zhou, X.: QoS-aware cloud service composition based on economic models. In: Liu, C., Ludwig, H., Toumani, F., Yu, Q. (eds.) ICSOC 2012. LNCS, vol. 7636, pp. 111–126. Springer, Heidelberg (2012). doi:10.1007/978-3-642-34321-6_8
17. Zhang, G.P.: Time series forecasting using a hybrid arima and neural network model. Neurocomputing **50**, 159–175 (2003)