Top-Down Pricing of IT Services Deals with Recommendation for Missing Values of Historical and Market Data

Aly Megahed[1(✉)], Kugamoorthy Gajananan[2], Shubhi Asthana[1], Valeria Becker[1], Mark Smith[3], and Taiga Nakamura[1]

[1] IBM Research - Almaden, San Jose, CA, USA
{aly.megahed, sasthan, beckerv, taiga}@us.ibm.com
[2] IBM Research – Tokyo, Tokyo, Japan
gajan@jp.ibm.com
[3] IBM Global Technology Services, North Harbour, Portsmouth, Hampshire, UK
marksmith@uk.ibm.com

Abstract. In order for an Information Technology (IT) service provider to respond to a client's request for proposals of a complex IT services deal, they need to prepare a solution and enter a competitive bidding process. A critical factor in this solution is the pricing of various services in the deal. The traditional way of pricing such deals has been the so-called bottom-up approach, in which all services are priced from the lowest level up to the highest one. A previously proposed more efficient approach and its enhancement aimed at automating the pricing by data mining historical and market deals. However, when mining such deals, some of the services of the deal to be priced might not exist in them. In this paper, we propose a method that deals with this issue of incomplete data via modeling the problem as a machine learning recommender system. We embed our system in the previously developed method and statistically show that doing so could yield significantly more accurate results. In addition, using our method provides a complete set of historical data that can be used to provide various analytics and insights to the business.

Keywords: Service analytics · IT service deals · Predictive analytics · Pricing services · Estimating prices · Data mining · Machine learning · Recommender systems

1 Introduction

Clients requiring complex Information Technology (IT) services typically submit a request for proposals that can fulfil their demands. Service providers have to respond with a proposed solution and enter a competitive bidding process trying to win the contract [1]. One of the critical factors in this process is the pricing of various services included in the proposal, though it is not the only factor for winning the deals [1, 2].

Traditionally, solutioners responsible for preparing the solution proposal followed the “bottom-up” pricing approach. In this approach, they obtain a quote by determining costs and prices of each detailed service component included in the solution and then summing them up [3, 4], given the hierarchical nature of IT services [6]. This is often a time consuming and resource-intensive task; because, in a complex service proposal, there could be thousands of items at the lowest level to be priced. In our previous work [3, 4], we proposed a “top-down” pricing method for IT services and demonstrated how this approach could lead to efficient pricing with adequate accuracy. The method makes use of historical data of prior deals as well as market data to determine the price using a minimal input from the user. It works via mining those data at the highest level [3] or the second hierarchical level [4]. The algorithm works in two main steps; peer selection and cost calculation, where after selecting similar peer historical and market deals, it mines these deals to estimate the costs and later the prices.

Previous results showed that this approach enables efficient pricing and that using the second hierarchical level (referred to as level two below) can improve estimation accuracy. However, one challenge was that not all level two subservices in the deal being priced are always included in the chosen peer deals to be mined. Since every deal can have different IT services demands, it can happen that some of the demanded services were not included in the chosen peer deals. The difference could become more significant as they are computed at granular service levels. In our previous work [4], we described a simple way to treat this problem; that is we assumed that a missing service at level two in a peer deal has a cost equal to the average of the costs of that same service in all the chosen peer deals that have that service.

In this work, we do a more advanced way of addressing this problem. That is, we formulate it as a recommendation problem [5] to complement the missing data, then proceed with the rest of our pricing algorithm. Figure 1 gives the overview of all mentioned approaches. We statistically show that our new approach is significantly more accurate to use than our two previous approaches. In addition, doing the recommendation can be used in several other applications rather than deal pricing. One important such applications is performing all kinds of analytics and trend analysis on historical and market data. Obtaining these analytical insights could help the business revise and reassess their competitiveness.

Therefore, the contributions of this paper are three-folds. First, we provide a novel formulation of the problem of missing services data in historical and market deals as a machine learning recommender system. Second, we show how embedding the results of this system in our prior pricing algorithm significantly enhances our pricing accuracy, statistically speaking. Thirdly, our approach can be directly used to enable all kinds of data analytics on the historical deal data; an insight of high business value.

The rest of this paper is organized as follows: in Sect. 2, we provide a review of the literature in the area of our problem. In Sect. 3, we present our new approach and show how it can be embedded in the pricing algorithm. We then show our numerical results in Sect. 4 and end the paper with the conclusions and ideas for future work in Sect. 5.

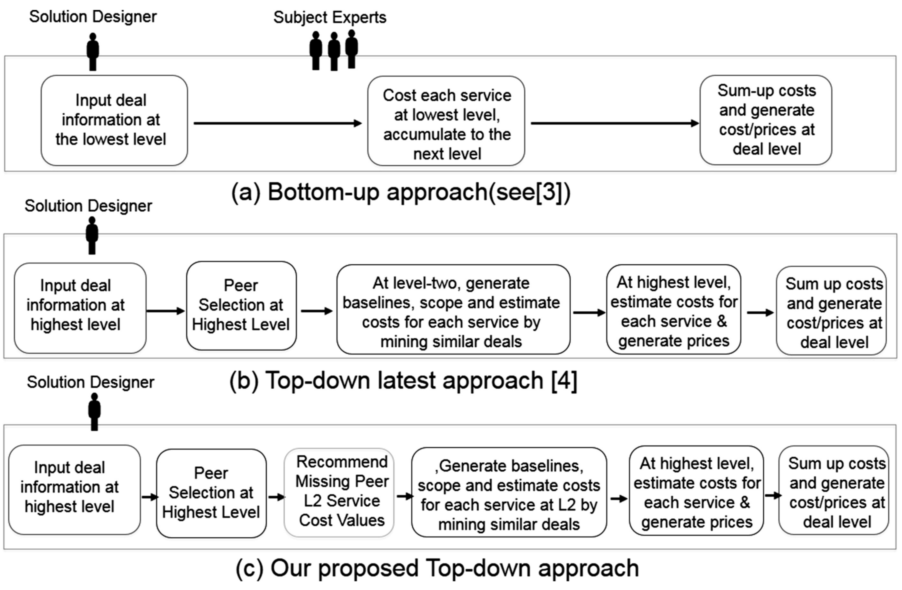
Solution Designer
Subject Experts
Input deal information at the lowest level
Cost each service at lowest level, accumulate to the next level
Sum-up costs and generate cost/prices at deal level
(a) Bottom-up approach(see[3])
Solution Designer
Input deal information at highest level
Peer Selection at Highest Level
At level-two, generate baselines, scope and estimate costs for each service by mining similar deals
At highest level, estimate costs for each service & generate prices
Sum up costs and generate cost/prices at deal level
(b) Top-down latest approach [4]
Solution Designer
Input deal information at highest level
Peer Selection at Highest Level
Recommend Missing Peer L2 Service Cost Values
,Generate baselines, scope and estimate costs for each service at L2 by mining similar deals
At highest level, estimate costs for each service & generate prices
Sum up costs and generate cost/prices at deal level
(c) Our proposed Top-down approach

Fig. 1. Overview of the bottom-up approach for pricing IT service deals, top-down latest approach in [4], and our proposed one

2 Related Work

References [1, 2, 7–9] provide an overview of the area of IT services deals and how competitive bidding process works. For pricing these deals, we refer the reader to our previous works in [3, 4]. This paper is an extension to the latter two papers. In [3], we proposed a method for top-down pricing of IT service deals, in which high level data for the included services in these deals are used. A peer selection algorithm along with a calculation logic was presented. Numerical results showed the validity of the hypothesis in the paper that mining historical data can lead to more accurate pricing than the business-traditionally used approach of using market benchmark data. The algorithm was enhanced in [4], where the data mining was done at level two of the services although the input is still at level one and thus the concept of top down pricing is preserved. Results showed that doing the pricing at such lower level can yield more accurate results. Note that there is no justification of pricing at a lower level, or doing it at the n^{th} level; since there is typically no known information at such lower levels when the deal is priced in the beginning of the reply to the request for proposal process. In addition, there are typically three, or at most four, levels of IT services in this type of business [3]. In this work, we extend the method in these latter two references and show the benefit of using a recommender system for augmenting missing data values.

Another related work in the literature is the work of Akkiraju et al. [9]. They presented a method for assessing the competitiveness of deals after being priced. That is, the deals need to be priced first in order for their method to work, rather than pricing the deals with minimal user input as the works in [3, 4] as well as the current one.

More literature about services pricing though not in the field of complex IT services deals can be seen in the studies in [10–20]. In [10], Li et al. provided a study of different pricing models for cloud storage. Their models focus on special characteristics of the cloud, e.g., storage types and configurations. Li et al. [11] studied queuing systems-based pricing models for other IT services. Ibrahim et al. [12] proposed a pricing framework for the pay-as-you-consume cloud computing service.

Basu et al. [13] developed optimal pricing strategies for cloud providers. Their method incorporates the utility of cloud users as a function of a set of parameters directly proportional to the utility and another set of parameters that have a negative effect on utility. In [14], Laatikainen and Ojala presented pricing models for software as a service (SaaS), in which they highlighted the relationship between architectural and pricing characteristics for this service. They showed that such relationship is tight when the value of firm proposition is at a high cloud maturity levels. Tawalbeh [15] stated an empirical study of a pricing method for mobile service providers driven by a cost based model. One of their main conclusions was that service providers should focus on market oriented pricing when their objectives are related to profitability, market share, and sales maximization.

For the sake of conciseness, we refer the reader to the other works for general services pricing [16–20]. In general, all these studies do not put in consideration the characteristics of IT services in complex deals as do our work and the prior two studies [3, 4] to it. Additionally, a literature survey for this class of these distantly related studies can be found in [21]. In the next section, we present our methodology in detail.

3 Methodology

In this section, we first present our notation in Sect. 3.1 followed by the formulation of our problem as a recommender system followed by the recommender algorithm that we used/applied to our real-world data in Sect. 3.2. We then show in Sect. 3.3 how we embed the results we can obtain from this modeling approach into our pricing algorithm.

3.1 Notation

We identify two categories of services that are included in any IT service deal in our context: regular services (referred to as services below) and common services. The regular services have baselines/units and their costs are independent of other services in a deal. Common services do not have baselines/units however their costs are dependent on different regular services included in the deal. Examples of regular services are databases and end user, and account management is a common service, for which the cost dependent on all regular services, which need some account management, in the deal.

We define any deal $d \in D$ by the tuple sets *(Meta Information, Services, Common Services), where D* is the set of deals (either historical or market benchmark), and Meta Information is the set of the meta-data of the deal, namely: *Meta Information = {Deal*

Outcome, Contract Year, Geography, Industry}. Deal Outcome is either won or lost (client did not pursue or provider withdraw from biding). *Contract year* is the calendar year at which delivery of the services will begin. *Geography* and *industry* refer to geographical location and industry of the client respectively.

We define the set of regular services as: *Services = {$Service_1$, ……, $Service_i$, ….., $Service_M$}*, where *M* is the cardinality of the set *Services*. Similarly, we specify the set of common services as: *Common Services = {Common $Service_1$, ……, Common $Service_j$, ….., $Service_N$}*, where *N* is the cardinality of the set *Common Services*.

Let us define (a) any regular service as $Service_i \in Services$, where $i = \{1, \ldots, M\}$, by the tuple *(Baseline, Cost, Price)*, and (b) any common service as *Common* $Serivce_j \in Common\ Services$, where $j = \{1,\ \ldots,\ N\}$, by the tuple *(Percentage of Total Cost, Cost, Price)*.

We further decompose each regular service $Service_i$, into a set of level two services: $L2Services = \{L2Service_1, \ldots, L2Service_a, \ldots, L2Service_P\}$ where $a = \{1, \ldots, P\}$, and *P* is the cardinality of the set *L2Services.* Similar to level one service $Service_i$, we define any $L2Service_a$ with the tuple *(L2Baseline, L2Cost, L2Price)* where cardinality *P* may vary for different level one services.

Similarly, we break down each common service *Common* $Service_j$ into a set of level two common services $L2\ Common\ Services = \{L2Common\ Service_1, \ldots, L2Common\ Service_b, \ldots, L2Service_Q\}$ where $b = \{1,\ \ldots,\ Q\}$, and *Q* is the cardinality of the set *L2Common Services.* Similar to level one service *Common* $Service_j$, we define any *L2Common* $Service_b$ with the tuple *(L2Percentage of Total Cost, L2Cost, L2Price)* where cardinality *Q* may vary for different level-one common services.

Finally, we define any scenario *S* to be a new deal to be priced, by the tuple sets *{Meta Information, $Service_s$, Common $Service_s$}.* The following are the inputs for our approach: the elements of the set *Meta* $Information_s$, the values of *Baselines* for each $service_k \in Services_s$ and the scope values for each *Common* $Service_l \in Common\ Services_s$. The output of our approach are: the estimated *Cost* and *Price* for each element of the sets $Services_s$ and *Common* $Services_s$, and thus the total cost and price of $Scenario_s$. In the following section, we briefly describe the details of our approach.

3.2 Formulating Our Problem as a Recommendation System

In a typical recommendation system, there are "users" and "items". The "ratings" of some user-item pairs are known while the rest are unknown [22]. An example of this is the movie recommendation. In that problem, there are some users who have seen some movies and rated them, but not all users have seen all movies. The movies' provider would like to predict the rating of users for the movies that haven't seen so that he can recommend to them those movies that they would have rated highly had they seen them. Another example is in online retailers, where the "users" are buyers and the "items" are the products that they can purchase.

There are generally two main classes of recommender systems; content-based recommendations and collaborative recommendations [5]. In the former one, the user will be recommended items that are similar to the ones that he/she preferred in the past.

In the latter one, the user will be recommended items that were preferred in the past by people with similar tastes to him/her.

Now, we look into our problem. Considering the historical and market deals as users and the services at any level as items/movies, one can see the mapping between the two problems. Figure 2 shows this analogy.

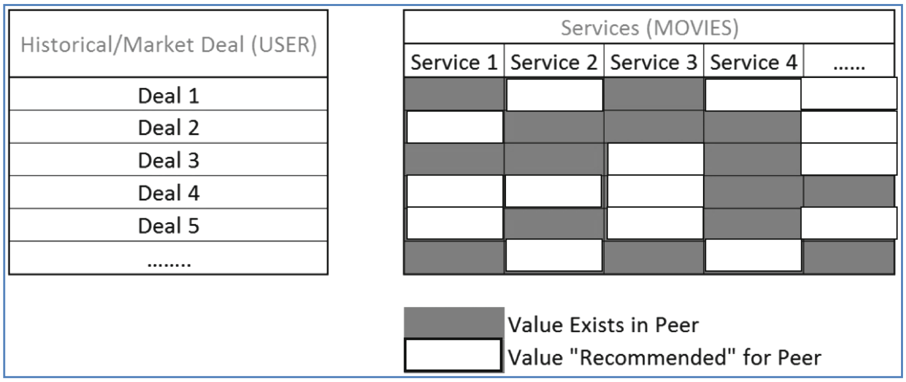

Fig. 2. Formulation of our problem as a recommender system

We note that the content-based recommendation is the one that better suits our application. This is because, in our pricing algorithm, we select peers at the highest level first and then perform cost mining for the preselected peers. Thus, the recommendation of missing data will be performed on the already filtered set of similar deals to our deal that we are trying to price. Thus, the collaborative recommendation is not applicable since we are already using a subset of "similar" deals. This observation was confirmed by a set of preliminary experiments that we did. Basically, we did a standard machine learning experiment, where we divided our data set into training and testing sets. Then, we trained several recommender systems on the training data set and applied them to the testing one. We found that the context based ones (also known as "item recommendation") give more accurate results on both sets. That is our method identified similar deals better than collaborative filtering; as it uses our expertise knowledge of problem structure. Thus, we decided to embed it in our approach.

We now provide an overview of the context-based recommender that we use. The basic idea is that we compute a similarity s between every item i that user u has no preference for yet and for every item j that he/she has a preference for. Then, u's preference for j, weighted by s is added to a running average. Lastly, the top items ranked by weighted average is returned. Note that, in our problem, that last step is not relevant; since we do not *recommend* the some services among the missing ones for each deal, but we are rather interested in only coming up with a score for item i (which would be that of j weighted by s. For a more detailed explanation of context-based recommendation, we refer the reader to [23]. In the next subsection, we show how/where we exactly embed this system in our pricing approach.

3.3 Embedding the Recommender System in Our Pricing Algorithm

Our previous studies [3, 4] extensively describe our top down pricing approach. In this paper, we provide a summary of the approach using similar definitions used in our previous work and in the above Subsect. 3.1.

Our approach contains the following steps: selecting peer deals, calculating scope and baselines for services at level two, recommending missing service cost values in peers at level two, and estimating costs/prices at level two and aggregating them to compute costs/prices at level one. In the following sub sections, we briefly describe each step.

Selecting Peer Deals. For each regular and common service of a scenario, our approach selects a set of historical and market benchmark deals as peers to draw the unit cost values of the service from them. Our approach compares the *Meta Information*$_s$ *(Deal Outcome, Contract Year, Geography, and Industry)* of the scenario to that of all historical and market benchmark deals to select the matching ones. The reason behind the choice of suitable Meta Information are explained in detail in our previous work [3].

Once deals are selected based on the Meta information match, our approach sorts the deals based on two different criterias separately defined for regular and common services.

For each regular service $\forall$Service $k \in Services_s$, our approach adopts a criteria based on *baselines proximity*. We denote $Baseline\ Proximity_{dsk}$ be the baseline proximity between deal $d \in D$ and scenario S for $Service\ k \in Services_s$ and define it as:

$$Baseline\,Proximity_{dsk} = |Baseline\ for\ Service_i\ of\ deal\,d - Baseline\ for\ Service\,k\,of\ scenario\,S|$$

For each common service $\forall Common$ Service$l \in Common\ Services_s$, our approach sorts the selected deals based on a different proximity which is denoted as $Common\ Service\ Proximity_{dsl}$ (the proximity between deal $d \in D$ and scenario S for $Common\ Service\ l \in Common\ Services$) and defined as follows:

$$Common\,Service\,Proximity_{dsl} = |Sum\,of\ Costs\,of\ regular\,services\ for\ deal\,d - Sum\,of\ costs\,of\ regular\,services\ for\ our\,scenario\,S|$$

We refer the readers to [3] for detailed explanation of the proximity criterias defined above.

Calculating Scope and Baselines for Services at Level Two. Note that our approach requires scope and baseline values for services at level one. Hence the approach estimates scope and baselines values for services at level two. Each $\forall$Service $k \in Services_s$ has many $L2Service_{ka} \in L2Services_s$. To decide which of them are in-scope, our approach rely on a set of predefined business rules. To calculate the baselines for $L2Service_{ka} \in L2Services_s$, the method uses the peer deal selected for the corresponding level one Service $k \in Services_s$ from market benchmark data. We denote

$p_m \in D$ as the market peer deal for a Service $k \in Services_s$ of a scenario S, $Baseline_{pma}$ as the baseline of the corresponding L2Serviceka of peer p_m, and $Baseline_{pmi}$ as the corresponding level one Servicei of peer p_m. Then the baselines for $L2Service_{ka} \in L2Services_s$ can be defined as:

$$L2Baseline_{ka} = \frac{L2BaseLine_{pma} * Baseline_k}{Baseline_{pmi}} \tag{1}$$

Recommending Missing Service Cost. We report on how our approach finds the recommended service cost values for selected peers for services of a scenario. For each service, Service $k \in Services_s$ of a scenario S, let us assume that there are selected peer deals $p_h \in D$ where $h = \{1, \ldots, H\}$, and H is the number of selected peers of that particular Service $k \in Services_s$. For each $L2Service_{ha} \in L2Services_h$, of the corresponding service from each peer, let us denote the cost as $L2Cost_{ha}$ which may be missing for some peers. For the peers that do not have the $L2Cost_{ha}$, our approach uses a recommender algorithm to estimate cost values from the pool of selected peers $p_h \in D$. Note that these selected peers of particular Service $k \in Services_s$ are similar to each other with respect to (a) their Meta information, and (b) baseline proximity for regular services. Note also that we implicitly assume, either here or in our overall methodology, that historical data is available. This is a quite realistic assumption practically speaking in this domain area.

Estimating Costs/Prices. We describe how our approach estimates the costs for each regular and common service for both the historical data and market benchmark views.

Cost Calculation for Regular Services of a Scenario. For each $L2Service_{ka}$ of Service $k \in Services_s$, our approach retrieves the unit costs of that level two service in each of its sorted peer deals and then compute the n^{th} Percentile of these peer unit costs. For $L2Service_{ka}$, we denote the resulting unit cost as $L2Unit - Cost_{ka}$ and its cost is computed as follows:

$$L2Cost_{ka} = L2Unit{-}Cost_{ka} * L2Baseline_{ka} \tag{2}$$

Finally, our approach computes the cost of the Service $k \in Services_s$, as follows:

$$Cost_{ka} = \sum\nolimits_{a \in L2Services_k} L2Cost_{ka} \tag{3}$$

Cost Calculation for Common Services of a Scenario. For each $L2Common\ Service_{s,l,b}$ of $Common\ Servicel \in Common\ Services_s$, our approach calculates the percentage of the cost for that level two service to the overall cost of the deal in each of its sorted peer deals. Then it use that percentage as is without any normalization and applies the l^{th} Percentile to the set of percentages of these peer percentages values to get the percentage of that service to the total cost of our scenario S. For each $L2Common\ Service_{s,l,b}$, we denote the resulting percentage as $L2P_{s,l,b}$.

Let us describe how to calculate the cost values for each $L2Common\ Service_{s,l,b}$. Let us define the total cost of all services in our scenario S as

$$Sum_{s,all} = Sum_{s,com} + Sum_{s,reg} \tag{4}$$

Where $SUM_{s,all}$ is the total cost of the scenario (sum of the costs for all services, both regular and common ones); $Sum_{s,reg}$ is the sum of the costs for the regular services-Service $k \in Services_s$; $SUM_{s,com}$ is the sum of the costs for the common services- *Common* Service $l \in Common\ Services_s$ and computed as follows

$$Sum_{s,com} = \sum\nolimits_{l \in Common\ Services_s} Cost_{s,l} \tag{5}$$

Where $Cost_{s,l}$ refers to the cost of $Common\ Service\ l \in Common\ Services_s.Cost_{s,l}$ can be further defined using the level two cost values as follows:

$$Cost_{s,l} = \sum\nolimits_{b \in L2Common\ Services_{s,l}} Cost_{s,l,b} \tag{6}$$

Now we replace $Cost_{s,l}$ in Eq. (5) with the definition in Eq. (6) which lead to the following:

$$Sum_{s,com} = \sum\nolimits_{l \in Common\ Services_s} \sum\nolimits_{b \in L2Common\ Services_{s,l}} Cost_{s,l,b} \tag{7}$$

Now we have that for each $L2Common\ Services_{s,l,b}$ in our scenario S:

$$Cost_{s,l,b} = Sum_{s,all} * L2P_{s,l,b} \tag{8}$$

Finally we transform the above set of linear equations to a standard form as follows:

$$\begin{aligned} \left(L2P_{s,1,1} - 1\right) * Cost_{s,1,1} + L2P_{s,1,1} * Cost_{s,1,2} + \ldots + \\ L2P_{s,1,1} * Cost_{s,1,B_1} + L2P_{s,1,1} * Cost_{s,2,1} + \ldots + \\ L2P_{s,1,1} * Cost_{s,2,B_2} + \\ L2P_{s,1,1} * Cost_{s,L,B_L} = -L2P_{s,1,1} * Sum_{s,reg} \end{aligned} \tag{10}$$

$$\begin{aligned} L2P_{s,1,2} * Cost_{s,1,1} + (L2P_{s,1,2} - 1) * Cost_{s,1,2} + \ldots + \\ L2P_{s,1,2} * Cost_{s,1,B_1} + L2P_{s,1,2} * Cost_{s,2,1} + \ldots + \\ L2P_{s,1,1} * Cost_{s,2,B_2} + \\ L2P_{s,1,2} * Cost_{s,L,B_L} = -L2P_{s,1,2} * Sum_{s,reg} \end{aligned} \tag{11}$$

$$\begin{aligned} L2P_{s,l,b} * Cost_{s,1,1} + (L2P_{s,1,2} - 1) * Cost_{s,1,2} + \ldots + \\ L2P_{s,l,b} * Cost_{s,l,B_1} + L2P_{s,l,2} * Cost_{s,2,1} + \ldots + \\ L2P_{s,1,1} * Cost_{s,2,B_2} + \\ (L2P_{s,l,b} - 1) * Cost_{s,L,B_L} = -L2P_{s,L,B} * Sum_{s,reg} \end{aligned} \tag{12}$$

Where L refers to the cardinality of the set $Common\ Services_s$; $B_1, B_2, \ldots B_L$ are cardinalities of the sets $L2Common\ Sevices_1, \ldots., L2Common\ Services_L$. Our approach solves these equations straightforwardly as they fulfil the requirement for such set of linear equations (see, for instance, [24]). By solving above equations, the approach computes the cost of each common service at level-two ($Cost_{s,l,b}$) per year.

We refer the readers to [3] to understand the difference between the cost estimations for historical and market benchmark perspectives.

To aggregate the costs at deal level, our approach add the costs for services at level two to get the costs at level one. Then, it sums up all level one service costs to compute the cost at deal or scenario level. To estimate the price, our approach adds a chosen arbitrary gross profits to the estimated costs. Figure 3 shows the overall overview of our approach. We also note that market data follows the same structure as historical data, with the difference in its source; market data are from market rates rather than historical deals. Therefore, the exact same aforementioned method for calculating the historical price point applies for calculating the market price point.

Note that we can straightforwardly embed our outputted prices in a prediction model as the one in [2, 3] in order to assess the probability of winning the deal at

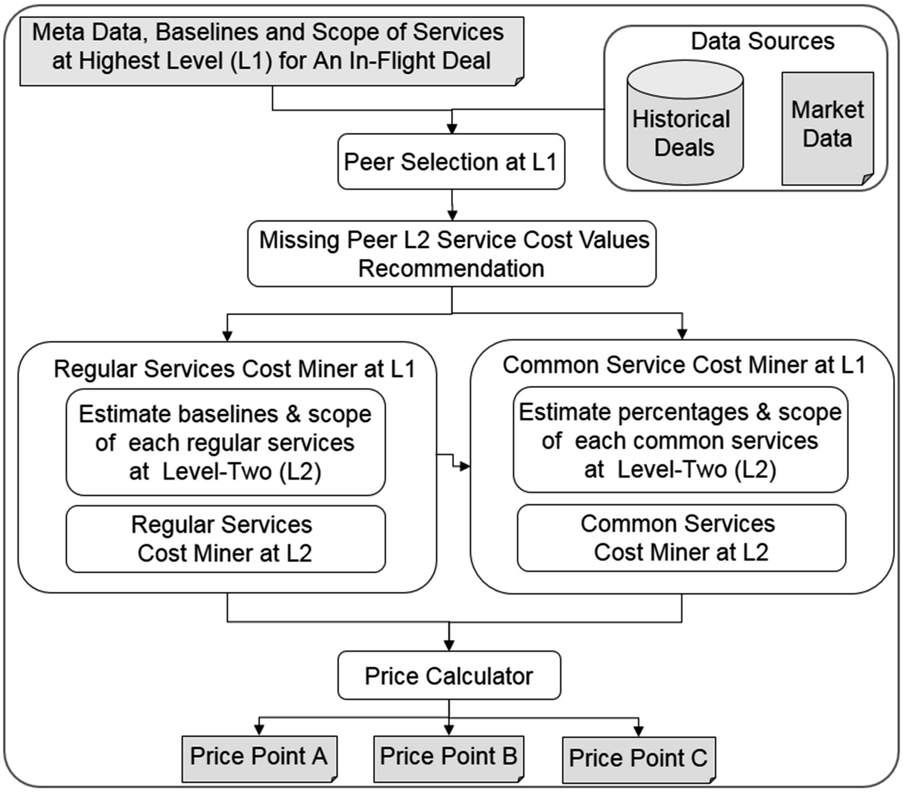
Meta Data, Baselines and Scope of Services at Highest Level (L1) for An In-Flight Deal
Data Sources
Historical Deals
Market Data
Peer Selection at L1
Missing Peer L2 Service Cost Values Recommendation
Regular Services Cost Miner at L1
Estimate baselines & scope of each regular services at Level-Two (L2)
Regular Services Cost Miner at L2
Common Service Cost Miner at L1
Estimate percentages & scope of each common services at Level-Two (L2)
Common Services Cost Miner at L2
Price Calculator
Price Point A
Price Point B
Price Point C

Fig. 3. Overall overview of our proposed pricing approach

different pricing points. We refer the reader to the details in these two references for the prediction model; since it is out of scope of the present work.

4 Evaluation

In this section, we first present our evaluation setup bed in Sect. 4.1 and then report some numerical results that show the usefulness of using our new approach.

4.1 Evaluation Setup

From an industrial data repository of an IT service provider, we retrieved 30 random historical deals with their complete cost structure (at level one and level two) information. For each of the deal, our test bed generated a corresponding scenario by using the deal's meta-data and baselines and scopes of level one services. The test bed further generated the cost estimation for the services of the scenarios by invoking the pricing approach described in Sect. 3. In addition, to compare with our previous approach, the test bed also generated the cost estimations for the services of the same scenarios, however by invoking the earlier version of top down pricing algorithm with recommending missing cost values of services of peer deals. The selected deals to create scenarios were excluded from being selected as peers when invoking the pricing algorithms. Figure 4 shows an overview of our test bed.

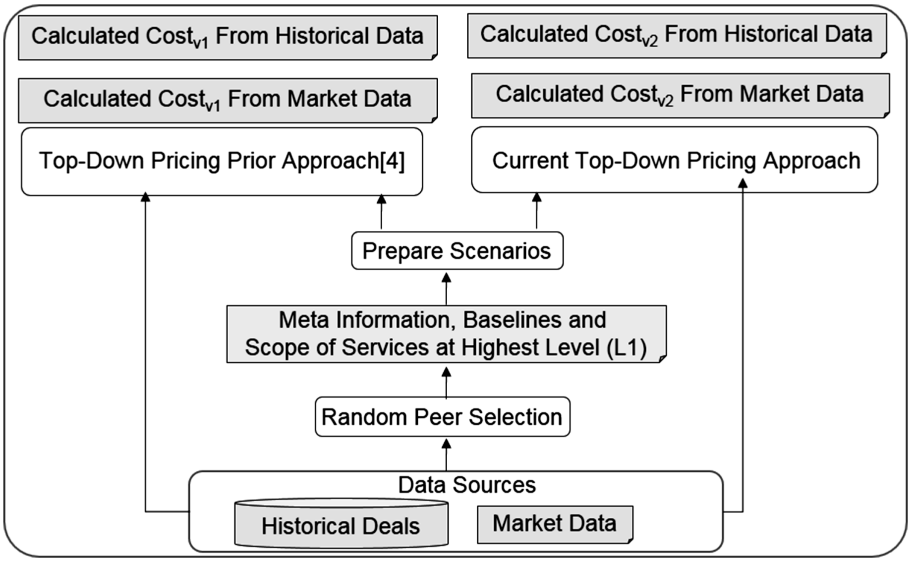
Calculated Cost$_{v1}$ From Historical Data
Calculated Cost$_{v2}$ From Historical Data
Calculated Cost$_{v1}$ From Market Data
Calculated Cost$_{v2}$ From Market Data
Top-Down Pricing Prior Approach[4]
Current Top-Down Pricing Approach
Prepare Scenarios
Meta Information, Baselines and Scope of Services at Highest Level (L1)
Random Peer Selection
Data Sources
Historical Deals
Market Data

Fig. 4. Overview of our evaluation test bed

4.2 Numerical Results

For recommending missing values, we use the item-based recommender algorithm implementation in Apache Mahout [25]. More specifically, we use the Pearson Correlation based Item Similarity algorithm [26]. That is, similarity between any two services $u \in L2Services$ and $w \in L2Services$ is calculated from the below equation:

$$\begin{aligned} &Pearson^{'}s\ Similarity\ Correlation\ Coefficient\ (w, u) \\ &= \frac{\sum_{i \in L2Services}\left(c_i - c_{avg,w}\right) * \left(c_i - c_{avg,u}\right)}{\sqrt{\sum_{i \in L2Services}\left(c_i - c_{avg,w}\right)^2 * \sum_i \left(c_i - c_{avg,u}\right)^2}} \end{aligned} \tag{13}$$

Where, c_i is the cost of service $i \in L2Services \backslash \{u, w\}$, while $c_{avg,w}$ is the average cost of service $w \in L2Service$ among all peer deals that have this service. $c_{avg,u}$ is the same but for service $u \in L2Service$. Note that if the denominator in Eq. (13) is zero, we set the corresponding coefficient to 1.

To use Mahout library, first our approach prepares the Mahout-compliant data. For that, the approach maps each peer deal's service data in the form: "dealID, serviceID, costValue", which is in the form of "userID, itemID, prefValue". Then, the approach invokes Mahout library's ItemSimilarity function to build the correlation map for the services from peer deals and GenericItemBasedRecommenderBuilder function to recommendations. Note that this all is done (the call to the Mahout library) in the appropriate step in our method, as explained in the previous section.

For each of our latest work in [4] and this work, we generated two cost points out of two perspectives; historical and market data. Note that we do our comparisons for costs values since prices are calculated by adding user chosen gross profits.

Now, we define the following errors for each $Service\ k \in Service$ and $Common\ Service\ l \in Common\ Services_s$:

$$\begin{aligned} &Ver1\_Error_{historical\ data} \\ &\quad = |Calculated\ Cost_{i,v1}\ From\ Historical\ Data_{scenario_s} \\ &\quad - Actual\ Cost_{i,deal_s}| \end{aligned}$$

$$\begin{aligned} &Ver1\_Error_{market\ data} = \\ &\quad |Calculated\ Cost_{i,v1}\ From\ Market\ Data_{scenario_s} \\ &\quad - Actual\ Cost_{i,deal_s}| \end{aligned}$$

$$\begin{aligned} &Ver2\_Error_{historical\ data} = \\ &\quad |Calculated\ Cost_{i,v2}\ From\ Historical\ Data_{scenario_s} \\ &\quad - Actual\ Cost_{i,deal_s}| \end{aligned}$$

$$\begin{aligned} &Ver2\_Error_{market\ data} = \\ &\quad |Calculated\ Cost_{i,v2}\ From\ Market\ Data_{scenario_s} \\ &\quad - Actual\ Cost_{i,deal_s}| \end{aligned}$$

Where $Veri\_Error_{historicaldata}$ is the absolute difference between the actual cost and that calculated using level *i,* where *i = 1* (the previous work [4]) or *i = 2* (this current work). Similarly, $Veri\_Error_{marketdata}$ shares the same definition but for market data.

Then, just like in [3, 4], we validate the same result for this work. That is; we obtain the results in Table 1 that show the confirming results that there is a significant increase in accuracy when historical data are mined to estimate costs compared to market ones. The paired t-test of hypothesis here (see [26, 27]) is as follows:

$$H_o : \mu_D = 0$$

$$H_1 : \mu_D < 0$$

Where,

$$\mu_D = \mu_{\{historical\,data\,error\}} - \mu_{\{market\,data\,error\}}$$

And, μ_D is the difference between the mean of the historical data errors ($Ver2\_Error_{historicaldata}$) and that of market data ($Ver2\_Error_{marketdata}$).

Next, we compare the costs obtained by our new pricing algorithm to that of the previous one in [4]. We use the difference between the errors of the two algorithms for historical data only. The reason for not performing this for market data is that typically, market data is complete and usually there is only one market deal for each geography setting. Thus, there is no need to apply our method for recommending missing values to that latter case.

Table 1. Hypothesis test result for difference between historical data and market data using the method in this paper

Service number	P-value	Reject Ho at significance level of 0.1
1	0.000	X
2	0.046	X
3	0.090	X
4	0.002	X
5	0.015	X
6	0.027	X

Table 2 shows the results of that comparison. One can see that our claim is justified; that is, statistically speaking, for almost all services in our study, performing calculations using the method presented in this paper is more accurate than the previous one in [4] (which was shown to outperform the previous results in [3]). The test of hypothesis in Table 2 is as follows:

$$H'_o : \mu'_D = 0$$

$$H'_1 : \mu'_D < 0$$

Where, $\mu'_D = \mu_{Ver_2} - \mu_{Ver_1}$

Here, μ'_D is the difference between the mean of the calculations for historical data errors of our current approach (e.g., $Ver1\_Error_{historical\ data}$) and that of our previous work (i.e., $Ver1\_Error_{historical\ data}$).

Table 2. Hypothesis test result for the difference between calculations using the method in this paper and those in [4] for historical data

Service number	P-value	Reject Ho at significance level of 0.1
1	0.007	X
2	0.084	X
3	0.095	X
4	0.073	X
5	0.16	
6	0.052	X

Lastly, note that using a paired t-test is justified using the same argument in [3, 4]. We also refer the reader to the texts in [26, 27] for more details on these tests. Also, note that the two-sided test is significant (H_o is rejected) for all the test shown above, and thus justifies doing the one sided tests whose results are shown in the three mentioned tables. Lastly, we note that we used a significance level of 0.1 and that at an 0.05 significance level, results will vary slightly as one can see in Table 2.

5 Conclusion and Future Work

In this paper, we presented an enhanced top-down pricing method for IT services deals. Our approach models the problem of missing values in the historical data, that is mined to estimate the costs for the deals to be priced, as an item-based recommender system. Using such system, we augment the missing values and embed the resulting complete set of data in the top-down pricing approach we proposed before. We showed that doing so could yield significant increase in the accuracy of services pricing, statistically speaking. Additionally, using the resulted complete set of data, one can perform more analytics to gain business insights and recommendations for the business. We also showed that our results still agree with the hypothesis we proposed in our previous works; that is, statistically speaking, using historical data could yield significantly more accurate results compared to the traditional business usage of market data.

There are multiple directions for future work. One direction is to further automate the user-input percentile step of our algorithm; as this could potentially improve the pricing accuracy. Another direction for future work is to apply some of the more sophisticated machine learning recommenders that use the context of both the users/items in the prediction. Lastly, applying this method to other general services that have a tinder process like ours, might be another direction for future research.

References

1. Greenia, B.D., Qiao, M., Akkiraju, R.: A win prediction model for IT outsourcing bids. In: Service Research and Innovation Institute Global Conference, pp. 39–42 (2014)
2. Megahed, A., Ren, G., Firth, M.: Modeling business insights into predictive analytics for the outcome of IT service contracts. In: Proceedings of the 12th IEEE International Conference on Services Computing (SCC), pp. 515–521 (2015)
3. Megahed, A., Gajananan, K., Abe, M., Jiang, S., Smith, M., Nakamura, T.: Pricing IT service deals: a more agile top-down approach. In: Barros, A., Grigori, D., Narendra, N.C., Dam, H.K. (eds.) Service-Oriented Computing. LNCS, vol. 9435, pp. 461–473. Springer, Heidelberg (2015)
4. Gajananan, K., Megahed, A., Nakamura, T., Abe, M., Smith, M.: A top-down pricing algorithm for IT service contracts using lower level service data. In: Proceedings of the 13th IEEE International Conference on Services Computing (2016, to appear)
5. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans. Knowl. Data Eng. **17**(6), 734–749 (2005)
6. Gamma, N., Do Mar Rosa, M., Da Silva, M.: IT services reference catalog. In: IFIP/IEEE International Symposium on Integrated Network Management (IM), pp. 764–767 (2013)
7. Yin, P., Nezhad, H.R., Megahed, A., Nakamura, T.: A progress advisor for IT service engagements. In: Proceedings of the 12th IEEE International Conference on Services Computing, pp. 592–599 (2015)
8. Megahed, A., Yin, P., Nezhad, H.R.: An optimization approach to services sales forecasting in a multi-staged sales pipeline. In: Proceedings of the 13th IEEE International Conference on Services Computing (2016, to appear)
9. Akkiraju, R., Smith, M., Greenia, D., Jiang, S., Nakamura, T., Mukherjee, D., Pusapaty, S.: On pricing complex IT service solutions. In: Service Research and Innovation Institute Global Conference, pp. 55–64 (2014)
10. Li, N., Zhang, L., Xu, P., Wang, L., Zheng, J., Guo, Y.: Research on pricing model of cloud storage. In: Proceedings of the 9th IEEE World Congress on Services, pp. 412–419 (2013)
11. Li, Z., Li, M.: A hierarchical cloud pricing system. In: Proceedings the IEEE 9th World Congress on Services, pp. 403–411 (2013)
12. Ibrahim, S., He, B., Jin, H.: Towards pay-as-you-consume cloud computing. In: Proceedings of the 8th IEEE International Conference on Service Computing, pp. 370–377 (2011)
13. Basu, S., Chakraborty, S., Sharma, M.: Pricing cloud services—the impact of broadband quality. Omega **50**, 96–114 (2015)
14. Laatikainen, G., Ojala, A.: SaaS architecture and pricing models. In: Proceedings of the 11th IEEE International Conference on Service Computing, pp. 597–604 (2014)
15. Tawalbeh, M.: The impact of marketing-oriented pricing on product mix pricing strategies – an empirical study on the mobile telecommunication providers in Jordan. Int. J. Econ. Commer. Manag. **III**(1), 1–18 (2015)
16. De Medeiros, R.W.A., Rosa, N.S., Pires, L.F.: Predicting service composition costs with complex cost behavior. In: Proceedings of the 12th IEEE International Conference on Service Computing, pp. 419–426 (2015)
17. Avlonitis, J.G., Indounas, A.K.: Pricing objective and pricing methods in the service sector. J. Serv. Mark. **19**(1), 47–57 (2005)
18. Indounas, K., Avlonitis, G.J.: Pricing objectives and their antecedents in the services sector. J. Serv. Manag. **20**(3), 342–374 (2009)

19. Xu, L., Jennings, B.: A cost-minimizing service composition selection algorithm supporting. In: Proceedings of the 7th IEEE International Conference on Service Computing, pp. 402–408 (2010)
20. Gaivoronski, A.A., Becker, D.: Differentiated service pricing on social networks using stochastic optimization. In: Proceedings of the 8th IEEE International Conference on Service Computing, pp. 386–393 (2011)
21. De Medeiros, R.W.A., Rosa, N.S., Campos, G.M.M., Pires, L.F.: A survey of cost accounting in service-oriented computing. In: Proceedings of the 10th IEEE World Congress on Services, pp. 77–83 (2014)
22. Pazzani, M.J., Billsus, D.: Content-based recommendation systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) Adaptive Web 2007. LNCS, vol. 4321, pp. 325–341. Springer, Heidelberg (2007)
23. De Gemmis, M., Semeraro, G.: Content-based recommender systems: state of the art and trends. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (eds.) Recommender Systems Handbook, pp. 73–105. Springer US, New York (2011)
24. Adhikari, M.R., Adhikari, A.: Text Book of Linear Algebra: Introduction to Modern Algebra. Allied Publisher Pvt Ltd., Hyderabad (2005)
25. Apache Mahout by Apache Software Foundation (2016). http://apache.org. Accessed 22 May 2016
26. Montgomery, D., Runger, G.: Applied Statistics and Probability for Engineers, 5th edn. Wiley, Hoboken (2010)
27. Walpole, R., Myers, R., Myers, S., Ye, K.: Probability and Statistics for Engineers and Scientists, 9th edn. Pearson, London (2011)