Towards a Comprehensive Purchasing Model for Cloud Services*

Octavio Martín-Díaz, Pablo Fernandez, José María García, and Antonio Ruiz-Cortés

Dpto. Lenguajes y Sistemas Informáticos
ETS. Ingeniería Informática – Universidad de Sevilla
41012 Sevilla, España – Spain
omartindiaz@us.es, pablofm@us.es, josemgarcia@us.es, aruiz@us.es

Abstract. Cloud Service market has evolved into a complex landscape that challenges the decision making of consumers as they develop their purchasing process. In particular, we explore the case of cloud infrastructure (IaaS) providers as an example of heterogeneous variety of purchasing options and discounts; this variability represents an important drawback during the decision making process where there is a need to compare and select the best option.
In this work, we define a common model to describe purchasing models from different providers taking into account complex concepts such as discounts and purchasing dynamics variability. This purchasing model represents a first step towards the automated support of decision making problems during the purchasing process. In order to validate our approach we apply the model in a real case study of IaaS purchasing.

Keyworkds: Cloud Services, Purchasing Options, Decision Making

1 Introduction

In the last years, cloud services proliferation is gathering pace as the Software-as-a-Service model is becoming the predominant architectural paradigm. Amongst these services, the Infrastructure-as-a-service (IaaS) case is one of the most successfull cases where providers compete to offer competitive purchasing options. Specifically, IaaS purchasing aims to obtain virtual processing and storage resources from a Cloud provider, as a cheaper alternative for the procurement of a private computer infrastructure. In the typical purchasing scenario, the providers offer their purchasing options, so that the goal is to search and select "*the best one*".

Purchasing options put together configurations and their price for using for a period of time. It may include volume storage, data transfer, security, IP and

* This work has been co-funded by the Spanish Goverment under TAPAS (TIN2012-32273) project, and by the Andalusian Administration under THEOS (TIC-5906) and COPAS (P12-TIC-1867) projects.

load balancing, and a myriad of other properties. This results in a highly configurable service, since the actual number of combinations can be overwhelming [5]. Moreover, the purchasing options have a high degree of variability since most of providers publish their own purchasing options in different ways. As a consequence, such a variability represents a challenging process of monitoring and billing in order to get the total price to charge that should take into account the different configuration purchased, the purchasing dynamic itself (when and how the purchase was developed) and the potential discounts that should be applied.

In this context, in order to assist the purchasing process and estimate the potential costs there is a need of techniques and tools to support the great deal of variability amongst different providers. Currently, there are some tools implementing the search of an optimal configuration like CloudScreener [1] or several configurations for independent instances, like Cloudorado [2]. However, these tools provide a simplified comparison that does not include a comprehensive purchasing options but a subset of it (e.g. they do not include any discount modeling).

In this paper, we define a comprehensive model of purchasing options to find a commonality in different providers in order to provide support for a complete comparisons amongst IaaS offerings; this model represents a first step towards an automated support of the purchasing process.

The remaining text is organized as follows. Sect. 2 introduces the purchasing process, in order to characterize the problem. Next, Sect. 3 and 4 present the purchasing model for automating the purchasing process and a case of study for illustrative purpose. Later, Sect. 5 comments the related work. Finally, Sect. 6 presents our conclusions and future work.

2 Purchasing in IaaS

The purchasing process of cloud services consists of several steps: First, a user searches for the information published from different providers, going through the pricing they offer for purchasing different instance configurations. Next, once the provider is selected, the user purchases a number of instances in order to support its own business process. Periodically, the user is charged for the actual usage of the purchased instances, so that some discounts may be applied if certain conditions are fulfilled. In addition, payments in advanced and commitment to use a number of instances may make the user be eligible for much greater discounts.

The most common type of purchasing is on demand (i.e. the user only pays for the actual usage) without obligations or contractual relationship[3]. Pricing is usually specified as follows: (i) the configuration of instances includes properties such as the CPU number and type, the size of and storage, the operating system,

[1] See `http://www.cloudscreener.com/` for details.

[2] See `http://www.cloudorado.com/` for details.

[3] There are other types of purchasing, which are out of scope of the paper. As an example, Amazon also offers (i) spot instances, (ii) dedicated instances, and (iii) the reserved instance marketplace.

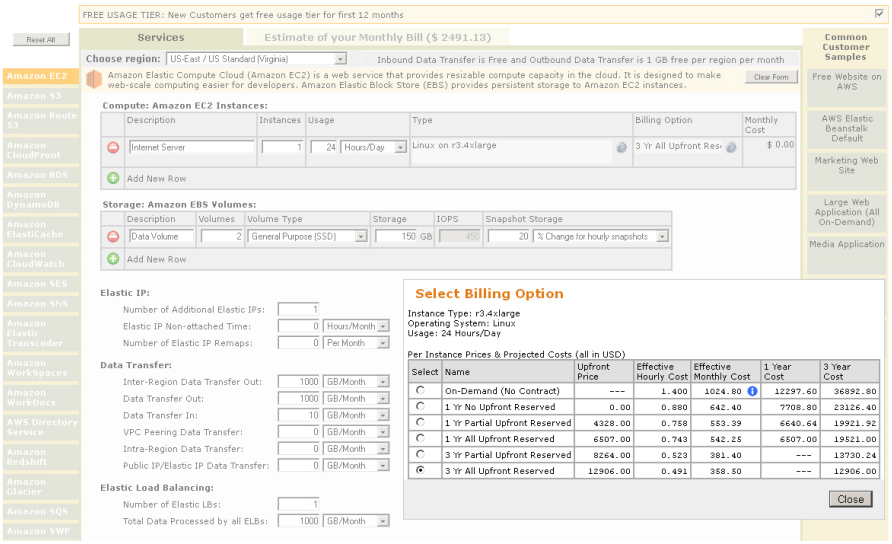

Fig. 1. Snapshot of the Amazon EC2 calculator

or the region on which the instance is executed, and (ii) the pricing is given in a tabular form whose columns are the configuration properties. Each row corresponds to a configuration and its basic price for a unit of time, either an hour or a month. Most of providers have a calculator in order to help users to select their preferred configurations. As an example, Fig. 1 depicts the case of the EC2 Calculator that is provided by Amazon in order to estimate the cost of a particular purchasing option (note the number of properties, denoting a highly configurable service).

From the purchasing dynamic perspective, Fig. 2 shows a proposed model in BPMN for the billing cycle that specify the different activities that correspond with charges or monitoring processes joint with the different events that are relevant. During a billing cycle, the monitoring of usage to be charged is usually done by counting hours or minutes, so that the total price to be charged is computed according to such usage; in some cases, providers define a minimum period of usage to be charged (e.g. Google establishes 10 minutes). In all the studied cases, billing cycle duration correspond with one month but there is a certain variability depending on the billing day that could correspond with a natural day (i.e. a given day on the calendar, usually the first) or the anniversary day (i.e. the day of the month in which the user began the relationship).

In the proposed billing cycle model, the events are associated with three different charges: First, on the contract event, one-time payments in advance may be charged (e.g. Rackspace prepayments, or Amazon reservation upfronts); next, on the billing cycle begin event, recurring payments in advance may be

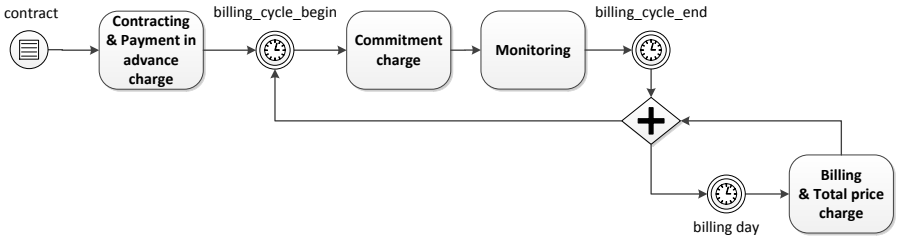

Fig. 2. Purchasing process with regards to the billing cycle

charged (e.g. Rackspace commitments); finally, on the billing cycle end event (or billing), the total price is usually charged.

In addition to the base prices of the purchasing options, providers offer a variety of different discount programs. In order to exemplify the variability of the discounts models, in the following we describe three different cases:

- Amazon offers the *reserve instance volume discounts* program[4]. On purchasing a large amount of reserved instances, if the total list price (the expected expense which is obtained by summing up these and precedent reservations) exceeds a given quantity, the user will be eligible for savings (both upfront and monthly prices) in subsequent reservations of instances.
- Google offers the *sustained use discounts* program[5] based on the actual usage of instances. The user is eligible for incremental discounts according to the usage thresholds which had been reached.
- Rackspace offers up to three discount programs[6]: (i) Volume discount, i.e. *spend more and save*. On billing, savings are applied according to the current charge; (ii) Commitment discount, i.e. *commit to a term, save even more*. The user commits to spend an amount for a term, being eligible for greater savings, according to the commitment amount and the duration of term. On the beginning of each billing cycle, the user is charged for the commitment amount, being savings applied (if any). On billing, the user is monthly charged the spent over, that is to say, the exceeds over the commitment amount. Volume discounts can be in turn applied on spent over, provided that conditions are fulfilled. (iii) Prepayment discount, i.e. *prepay for the entire term, save the most*. The user prepays the total amount committed for a term, being eligible for greatest savings. On billing, the user is monthly charged the spent over, that is to say, the exceeds over the com-

[4] See `http://aws.amazon.com/ec2/purchasing-options/reserved-instances/` for details on Amazon reserved instances volume discounts.

[5] See `https://cloud.google.com/compute/pricing` for details on Google sustained use discounts.

[6] See `http://www.rackspace.com/cloud/servers/discounts` for details on Rackspace discounts for cloud servers.

mitment amount. Volume discounts can be in turn applied on spent over, provided that conditions are fulfilled. In all cases, savings are set by means of a logarithmic scale according to the total amount which is being charged.

3 Purchasing Model

Figure 3 shows our purchasing model consisting of five different modules: configuration, pricing, monitoring, billing, and discounts. These modules and their interdependencies support the whole purchasing process, as described in Sec. 2.

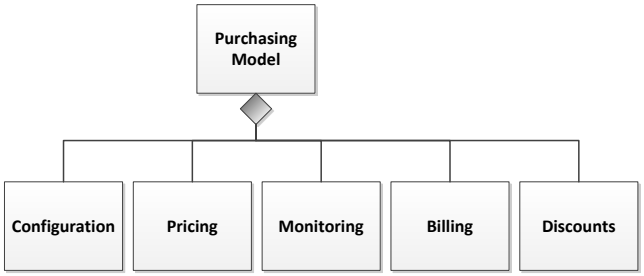

Fig. 3. Purchasing model modules

The configuration module provides facilities to describe computational characteristics of cloud services, so that we can define which particular IaaS instance we are interested in with respect to its instance type (i.e. CPU, memory, storage and other computational features), its operating system, and the region to be deployed in, among other characteristics. Depending on the instance configuration, the pricing module can be used to describe all the pricing options offered by a cloud service provider, whereas the monitoring module registers the actual usage of that instance, and hence enabling the total pricing to be computed. Figure 4 shows a conceptual global view of these three modules, including the relationships between the main concepts of those modules, depicted with a gray background. Empty arrows represent *is-a* relationships (e.g. a Base Price is a Price), diamonds associates a compound with its components (e.g. Configuration is composed of Instance Type, Region and Operating System), and filled arrows show uni-directional dependencies (e.g. the Total Pricing depends on the Base Pricing, the Extra Price, and the Usage Time).

Focusing on the pricing module, the Base Pricing for purchasing instances relates an instance Configuration and a Purchasing Option with a Base Price. The Base Price Rate, which establishes how often the Base Price is charged, is also specified, since some purchasing models offer different prices for each available rate. Optionally, depending on the purchasing option chosen (e.g. *on*

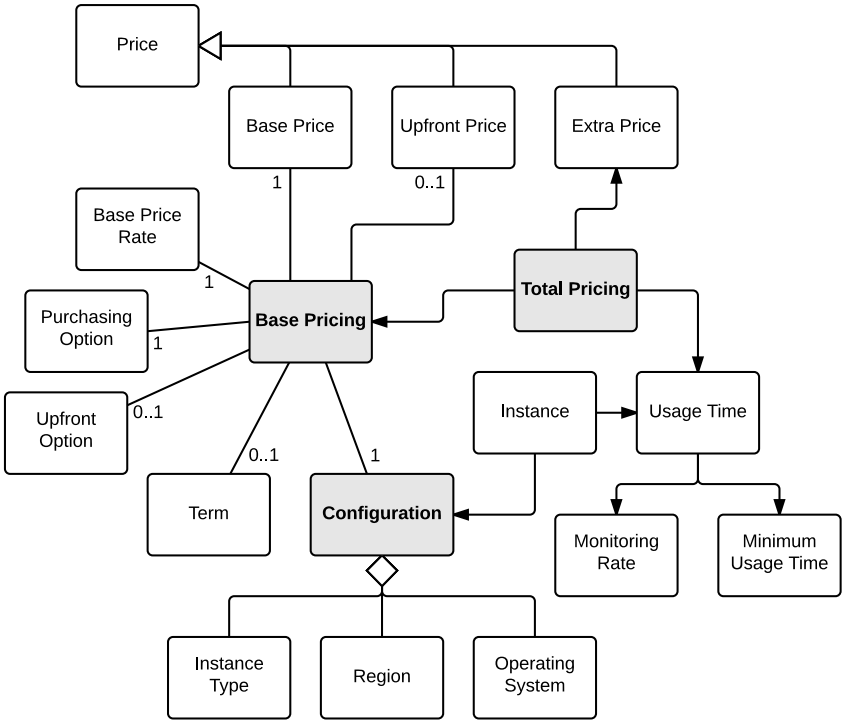

Fig. 4. Configuration, pricing and monitoring model

demand or *reserved* in Amazon EC2), an Upfront Price can be established, which will depend in turn on the actual Upfront Option chosen (e.g. one can choose from none, partial or total upfront payment in Amazon EC2) and on the contract Term (i.e. the number of months or years that the contract will last). Extra price refers to those properties which are not included in the configuration by all providers, such as region or operating system, but their prices are given apart.

Monitoring can be carried out either hourly or minutely, and a minimum time charged can be added. For each instance, the number of hours or minutes of usage is monitored. This usage time, along with the base pricing applied plus the extra costs are used to compute the total pricing that will be billed.

Although not shown in Fig. 4, the billing module describes the billing cycle period, which is usually a month, so that the billing day can be either a natural day or the anniversary day. Different kinds of charges can be identified. First, the total price obtained from the monitoring module, which is the payment per usage at the end of a billing cycle. Second, a payment in advance, which is a one-time or recurring payment that corresponds to a flat rate per usage. And finally, a commitment charge, which is not properly a payment, but a obligation

to make it in due time. The events when these charges can be applied are (1) the contract, (2) the begin of the billing cycle, and (3) the end of the billing cycle, or simply billing event.

Finally, the discounts module uses several types variables obtained from previous modules, such as the monitored usage of instances, the expected expense, the different charges, and the savings for discounts to be applied. We define the discounts by means of ECA rules (event, condition, action rules). Each ECA rule consists of:

- The event of a billing cycle which triggered it, in order to define the instant in which the discount is to be set, or applied, or both.
- The condition on variables to be fulfilled for the action to be effective.
- The action can consist of (1) setting the savings (on contract), or (2) applying the savings (on the current charge), or (3) both.

As an example, we define the discounts by Amazon. Regarding Amazon, the ECA discount rule is triggered on the reservation of instances (as the contract). If the total list price (as the expected expense) exceeds a quantity (as we illustrate later in the example) then the savings are set for the user in case of subsequent reservations (the contract).

ON RESERVATION
IF TOTAL_LIST_PRICE >= $500000
THEN SET SAVINGS = 5% **FOR** USER

ON RESERVATION
IF TOTAL_LIST_PRICE >= $4000000
THEN SET SAVINGS = 10% **FOR** USER

Fig. 5. Rules discounts for Amazon

4 Case of Study

In order to illustrate the purchasing process, we introduce the following case of study of an user with a demanding requirements of computing and selects Amazon EC2 reserved instances options according to the following scheduling:

- On January the 15^{th}, the user reserves 10 instances of the "d2.8xlarge" configuration for a year. It is published in the website to have 36 virtual CPUs, 116 ECUs (EC2 computed unit), memory of 255 GiB, storage of 24 per 2000 Gb HDD, with Linux operating system, deployed in the East US Region. Its reservation has an all upfront of $23616 and monthly price of $0, for a full amount of $236160.

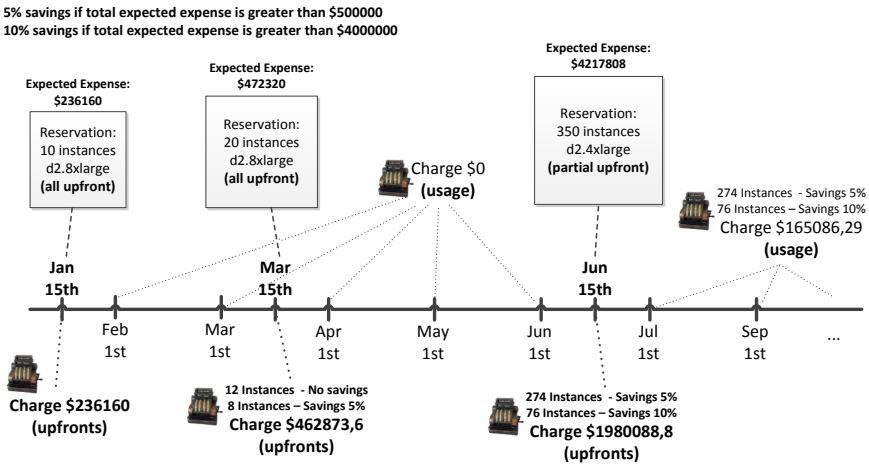

Fig. 6. Amazon discounts case of study

- On March the 15^{th}, the user reserves another 20 instances of the "d2.8xlarge" configuration for a year, for a full amount of $472320.
- On June the 15^{th}, the user reserves 350 instances of the "d2.4xlarge" configuration for a year. It is published to have 16 vCPUs, 56 ECUs, memory of 122 GiB, storage of 12 per 2000 Gb HDD, with Linux operating system, deployed in the East US Region. Its reservation has a partial upfront of $6024 and monthly price of $502,24 for a full amount of $4217808.

In Amazon, the discount rule is only triggered on reservation (a *contract event*). The billing cycle events, as shown in Fig. 6, occur as follows:

- On January the 15^{th}, the reservation date, the user will be immediately charged for $236160.
  - As no threshold for discounts is reached, so that the rule condition is not fulfilled, then the discount rule action is not executed.
- On February and March the 1^{st} (a pair of *billing day events*), the user will not be charged any amount since the monthly price was $0.
- On March the 15^{th}, another reservation date, the user will be immediately charged for $462873,6 which corresponds to:
  - For the first 12 instances, the charge is for $283392. In full, the user sums up a total expected expense of $519552:
    - The first threshold for discounts is reached, so that the rule condition is fulfilled, then the discount rule action is executed so that the user becomes eligible for savings of 5% in subsequent reservations (contracts).
  - For the remaining 8 instances, the user is charged for only $179481,6 after savings of 5% had been applied.

- On April, May, and June the 1^{st}, (some *billing day events*), the user will not be charged any amount since the monthly price was $0.
- On June the 15^{th}, another reservation date, the user will be immediately charged for $1980088,8 which corresponds to:
  - For the first 274 instances, the charge is for only $1568047,2 after savings 5% had been applied. In full, the user sums up a total expected expenses of $4010421,1:
    - The second threshold for discounts is reached, so that the rule condition is fulfilled, then the discount rule action is executed so that the user becomes eligible for savings of 10% in subsequent reservations (contracts).
  - For the remaining 76 instances, the user is charged for only $412041,6 after savings of 10% had been applied.
- From July the 1^{st} on (this and subsequent *billing day events*) the user will be charged monthly for $165086,29 which corresponds to:
  - For 274 instances, the user is charged for $130733,07 after savings of 5% had been applied.
  - For 76 instances, the user is charged for $34353,22 after savings of 10% had been applied.

5 Related Work

A purchasing model involves a characterization of the cloud services description and the purchasing process. In this context, the problem of service description modelling has been extensively addressed in the literature from semantic driven approaches [9] to traditional interface-driven alternatives [3]; in both cases, the combination of functional and non-functional service properties and configuration have been also developed in different formal languages [1] [6]. Traditionally, the applicability of these models to different scenarios were either simplified prototypes or intra-organizational scenarios whose pricing models were simplified to a single cost property in most of cases and, to the best of our knowledge, no formal characterization of complex purchasing models has been developed. As an example, in [11], authors present a formal language for cloud services with a pricing model that does not include a variable purchasing process nor extensible discount rules. As we introduce in this work, an analysis of a real open market of services (i.e. the cloud service market) shows a complex variability of purchasing models where the processes varies and different combination of purchasing options arises; in particular there is a rich combination of discount rules in the different providers studied. Similar limitations can be found in other related works such as [8] where authors present an initial review of different pricing models for cloud services that shows simple cost model scenarios from different authors where elements such as purchasing processes variability or discounts are not discussed.

The purchasing problem can be also analyzed from an economic and quality perspectives. On the one hand, from an economic perspective, some studies over

the cost of cloud services have been developed [2] [10] [4]; these works focus on the analysis of investments to support the decision making during the purchasing process but they do not address how the different purchasing options are formally expressed. On the other hand, from a quality perspective, in [7] authors provide a comprehensive model to automate the ranking of different providers but define a simplified version of cost and do not introduce a real purchasing option variability in their scenario.

6 Conclusions and Future Work

While Cloud Computing market is growing, the purchasing model is becoming increasingly complex. Specifically, the process of selecting and comparing offers from different providers result in a time-consuming and error-prone task. In this paper we aim to establish a first step towards the automation of the support in the purchasing process. To this end, this paper outlines a comprehensive purchasing model for Cloud Infrastructure that takes into account all the aspects included in real-world offerings.

Some consequent actions are foresight as future work, specifically: to design a concrete language to help in the modeling phase; to develop a set of tools for analyzing the model and provide an automated the comparison among different offers; finally we will study the applicability of this model as part of a more general idea of service level agreement that has being addressed by authors in previous works.

References

[1] Alain Andrieux, Karl Czajkowski, Asit Dan, Kate Keahey, Heiko Ludwig, Toshiyuki Nakata, Jim Pruyne, John Rofrano, Steve Tuecke, and Ming Xu. Web services agreement specification (ws-agreement). In *Open Grid Forum*, volume 128, 2007.

[2] Slaven Brumec and Neven Vrček. Cost effectiveness of commercial computing clouds. *Information Systems*, 38(4):495–508, 2013.

[3] Erik Christensen, Francisco Curbera, Greg Meredith, Sanjiva Weerawarana, et al. Web services description language (wsdl) 1.1, 2001.

[4] Se-Hak Chun and Byong-Sam Choi. Service models and pricing schemes for cloud computing. *Cluster Computing*, 17(2):529–535, 2014.

[5] Jesús García-Galán, Pablo Trinidad, Omer F. Rana, and Antonio Ruiz-Cortés. Automated Configuration Support for Infrastructure Migration to the Cloud. *Future Generation Computer Systems*, 2015.

[6] Jose M. Garcia, Carlos Pedrinaci, Manuel Resinas, Jorge Cardoso, Pablo Fernandez, and Antonio Ruiz-Cortes. Linked usdl agreement: Effectively sharing semantic service level agreements on the web. In *Web Services (ICWS), 2015 IEEE 22nd International Conference on*. IEEE, 2015.

[7] Saurabh Kumar Garg, Steve Versteeg, and Rajkumar Buyya. A framework for ranking of cloud computing services. *Future Generation Computer Systems*, 29(4):1012–1023, 2013.

[8] Sahil Kansal, Gurjeet Singh, Harish Kumar, and Sakshi Kaushal. Pricing models in cloud computing. In *Proceedings of the 2014 International Conference on Information and Communication Technology for Competitive Strategies*, page 33. ACM, 2014.

[9] Carlos Pedrinaci, Jorge Cardoso, and Torsten Leidig. Linked usdl: a vocabulary for web-scale service trading. In *The Semantic Web: Trends and Challenges*, pages 68–82. Springer, 2014.

[10] Bhanu Sharma, Ruppa K Thulasiram, Parimala Thulasiraman, Saurabh K Garg, and Rajkumar Buyya. Pricing cloud compute commodities: a novel financial economic model. In *Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012)*, pages 451–457. IEEE Computer Society, 2012.

[11] Rafael Brundo Uriarte, Francesco Tiezzi, and Rocco De Nicola. Slac: A formal service-level-agreement language for cloud computing. In *Utility and Cloud Computing (UCC), 2014 IEEE/ACM 7th International Conference on*, pages 419–426. IEEE, 2014.