Estimación del coste de aprovisionar instancias de cómputo para ejecutar aplicaciones *bag-of-task* en el cloud de Amazon

Pedro Álvarez, Sergio Hernández, Javier Fabra, Joaquín Ezpeleta

Instituto de Investigación en Ingeniería de Aragón (I3A)
Departamento de Informática e Ingeniería de Sistemas
Universidad de Zaragoza, España
{alvaper,shernandez,jfabra,ezpeleta}@unizar.es

Resumen Sin duda, el coste económico es un factor decisivo cuando se valora la posibilidad de ejecutar una aplicación en cloud. Hoy en día, estimar cuál es el precio a pagar no es sencillo si se trata de aplicaciones que requieren un elevado número de recursos de cómputo y almacenamiento. En este trabajo, se propone un método para minimizar el coste de ejecución de aplicaciones paralelas *bag-of-task* en un entorno de cómputo tipo cloud. Este método no sólo calcula una estimación del precio a pagar, sino que también determina qué recursos deberían ser contratados para minimizar ese precio. Una contribución importante de la solución es que considera la heterogeneidad de los proveedores cloud a nivel de catálogo de recursos, modelos y plazos de arrendamiento, opciones de pago, etc. La propuesta ha sido aplicada a un problema intenso en cómputo y ejecutado en el entorno de Amazon Elastic Compute Cloud (EC2).

Palabras clave: Computación en la nube, minimización de costes, aplicaciones paralelas, aprovisionamiento de recursos, Amazon EC2

1 Introducción

La mayoría de los proveedores de *cloud computing* ofrecen recursos de cómputo y almacenamiento conforme un modelo de pago por uso. Por lo tanto, desde la perspectiva de los usuarios que desean ejecutar sus aplicaciones software utilizando estos recursos, es necesario disponer de herramientas que permitan estimar de antemano el coste de sus ejecuciones. De inicio, conseguir que estas estimaciones sean precisas es una tarea compleja y requiere que sean combinadas las restricciones del usuario (el tiempo máximo de ejecución o el presupuesto disponible, por ejemplo), la naturaleza y los requisitos de la aplicación (políticas de acceso y almacenamiento de datos, nivel de paralelismo, decisiones de despliegue de sus componentes, etc.) y la heterogeneidad del proveedor cloud (tipos de recursos, opciones de arrendamiento, formas de pago, políticas de precios, etc.). Este último factor no sólo tiene una influencia muy

significativa en la complejidad de las estimaciones, sino que además plantea un problema adicional al usuario cuando éste planifica la ejecución de su aplicación en el cloud: dentro del amplio catálogo de recursos (sobre todo de cómputo) que ofrece un proveedor y teniendo en cuenta el rendimiento y el precio de estos recursos, ¿qué recursos son los más adecuados para ejecutar una aplicación?. La respuesta a esta cuestión tiene una consecuencia directa en el coste final e, incluso, una elección inadecuada suele conllevar un coste extra para el usuario que puede llegar a ser significativo cuando se ejecutan aplicaciones complejas. Por lo tanto, en el escenario actual no sólo resulta de interés estimar el coste total de ejecución, sino también minimizar ese coste aprovisionando los recursos más adecuados para cada aplicación concreta.

El problema previamente expuesto ha sido denominado por la comunidad científica como *aprovisionamiento basado en costes* de recursos cloud (en inglés *cost-driven provisioning*). Desde una perspectiva general, los trabajos que abordan este problema se pueden agrupar en tres categorías. El primer tipo de trabajos propone una simplificación del problema: el usuario decide qué recursos aprovisionar para ejecutar su aplicación en cloud y, una vez ha sido ejecutada, evalúa el coste de esa ejecución y lo adecuado de sus decisiones [1–3]. Obviamente, este modelo de solución basado en la ejecución directa, con los consiguientes costes, está lejos de lo deseable. Como alternativa, otros trabajos definen un modelo económico que estime el coste de ejecutar una aplicación utilizando unos recursos cloud concretos [4–6]. Estos modelos constan de fórmulas que combinan los precios de los recursos seleccionados con los parámetros operativos de la aplicación a ejecutar (tiempos de ejecución, volumen de datos de entrada y salida, requisitos de almacenamiento, operaciones de acceso a datos, etc.). Por lo tanto, exigen al usuario ser capaz de determinar estos parámetros como paso previo a la estimación del coste. A pesar de los esfuerzos, es complicado encontrar un modelo suficientemente general como para poder ser aplicado a cualquier tipo de aplicación (secuencial, paralela, un workflow, etc.) [7]. Finalmente, el tercer tipo de trabajos es el que resulta más interesante: en vez de estimar el coste de ejecución en base al uso de unos recursos concretos, pretenden minimizar ese coste determinando el aprovisionamiento de recursos más barato y que además garantice las restricciones del usuario y la aplicación. Sin embargo, en algunos casos estas propuestas ignoran la heterogeneidad de los recursos ofertados por el proveedor [8, 9] y, en otros, los modelos son integrados en mecanismos de *scheduling* con el fin de ayudar al sistema a decidir cuántos recursos necesita en cada instante minimizando el coste final de la ejecución [10, 11].

En este trabajo se propone un método para minimizar el coste de ejecución de una aplicación en cloud respetando un tiempo máximo definido por el usuario (este tiempo evita que se contraten recursos baratos dilatando excesivamente el final de la ejecución, por ejemplo). La diferencia con respecto otros trabajos existentes es que tiene en cuenta en sus decisiones los distintos tipos de recursos cloud ofertados por el proveedor y el rendimiento teórico de estos recursos desde el punto de vista de la aplicación a ejecutar. Además, el método determina cuántas unidades de cada tipo de recurso deberían ser aprovisionadas y durante

cuánto tiempo. Ante la dificultad de definir un método suficientemente general que estime el coste mínimo de cualquier tipo de aplicación, nuestra propuesta se ha acotado a aplicaciones paralelas de tipo *bag-of-task*. Para evaluar la solución propuesta, como proveedor de infraestructura cloud ha sido elegido *Amazon Elastic Compute Cloud* (EC2). Esta decisión viene justificada principalmente por dos motivos: por un lado, Amazon ofrece un entorno de computación maduro y estable, siendo según Gartner la solución líder en el mercado actual [12]; y, por otro lado, su catálogo de recursos de cómputo (llamadas *instancias de cómputo* en terminología Amazon) es muy amplio y con un abanico de precios muy variado en función del rendimiento de cada tipo de instancia, de las opciones y plazos de arrendamiento, de la forma de pago seleccionada por el usuario, de la localización geográfica de las instancias, etc. No obstante, el método es independiente del proveedor seleccionado, pudiendo ser aplicado incluso a soluciones que consideren la posibilidad de usar las instancias de cómputo de más de un proveedor.

Finalmente, el artículo consta de las siguientes secciones. En la sección 2 se describen las características de una aplicación *bag-of-task* y se discuten determinadas cuestiones a nivel de proveedor cloud que afectan al método de estimación propuesto. En la sección 3 se presenta el método para minimizar el coste de cómputo de una aplicación, y en la sección 4 se aplica sobre una aplicación concreta en el campo de la Web semántica. Por último, se discuten las principales conclusiones del trabajo y las líneas de trabajo futuro.

2 Consideraciones iniciales

En esta sección se discuten de forma breve algunas cuestiones que ayudan a entender el método propuesto. En primer lugar, se concreta el tipo de aplicación para el cual está pensado el método de minimización de costes. Posteriormente, se introducen las bases del método y ciertas precondiciones establecidas como consecuencia del funcionamiento habitual de los proveedores comerciales de infraestructura cloud.

Como se introdujo previamente, nuestro método pretende minimizar el coste de ejecución de aplicaciones paralelas de tipo *bag-of-task*. La principal característica de estas aplicaciones es que están compuestas por un elevado número de tareas similares (habitualmente, ejecutan el mismo código sobre datos de entrada diferentes) e independientes entre si. Por lo tanto, la naturaleza de las tareas favorece el paralelismo a gran escala, convirtiendo al cloud en un entorno atractivo para su ejecución gracias a la "ilusión" de recursos de cómputo infinitos que ofrece al usuario (esta disponibilidad no es real, dado que todos los proveedores comerciales establecen una cota límite de recursos a aprovisionar en cada instante y según el perfil de usuario que los solicita). En nuestro caso concreto, influenciados por el tipo de problemas que estamos resolviendo actualmente (por ejemplo, en el campo de la semántica y *Linked-data* [13]), estamos interesados en aplicaciones *bag-of-task* donde sus tareas son intensas en cómputo; es decir, el coste derivado de la transferencia de los datos de entrada y salida de cada tarea al recurso donde se ejecuta es insignificante

en relación al coste del procesamiento de esos datos. Por este motivo, el método presta especial atención al coste de las instancias de cómputo requeridas para resolver el problema.

Por otro lado, el método propuesto requiere conocer el rendimiento de las distintas instancias de cómputo ofertadas por el proveedor cloud. Este rendimiento se define como el número de tareas que es capaz de ejecutar por unidad de tiempo. Dado que la mayoría de los proveedores comerciales facturan conforme un modelo *full-hour billing model*, habitualmente la unidad de tiempo seleccionada es la hora. En el cálculo del rendimiento suponemos que la instancia de cómputo está dedicada por completo a la ejecución de tareas. Obviamente, este valor de rendimiento es dependiente del problema a resolver y puede ser calculado de forma experimental o por medio de simulaciones. Además, estos valores de rendimiento también sirven para calcular el coste de ejecutar una tarea en cada tipo de instancia concreta. No obstante, este coste no sólo depende del rendimiento, sino también del modelo de arrendamiento seleccionado. Es frecuente que un proveedor ofrezca la posibilidad de contratar una misma instancia de cómputo en base a diferentes modelos y plazos de arrendamiento y opciones de pago. Éste es el caso, por ejemplo, de Amazon EC2: una instancia de cómputo de tipo *m3.xlarge* puede ser arrendada por horas, o por plazos de 1 o 3 años, siendo diferente su precio en cada caso particular. Por lo tanto, en estos casos, un coste diferente por tarea será calculado para cada uno de las distintas opciones de arrendamiento existentes.

No obstante, el rendimiento de una instancia de cómputo en cloud también se ve afectado por una serie de factores difíciles de medir. El primero de estos factores es consecuencia del hecho de que los proveedores de infraestructura cloud ofrecen sus instancias de cómputo como máquinas virtuales. En nuestro método hemos supuesto que dos instancias de un mismo tipo tienen el mismo rendimiento. Sin embargo, existen ciertos trabajos que afirman que dos instancias de un mismo tipo pueden ofrecen rendimientos muy diferentes como consecuencia del uso de técnicas de virtualización y de la heterogeneidad del hardware subyacente [14]. En nuestro método, los efectos de este problema los hemos tratado de minimizar calculando rendimientos medios de las instancias por medio del uso de técnicas de experimentación. Por otro lado, un segundo factor está relacionado con el tiempo que transcurre desde que se solicita al proveedor una instancia de cómputo hasta que ésta está en disposición de comenzar a ejecutar las tareas. Aunque no existen estudios que cuantifiquen este tiempo, en escenarios donde las instancias se contraten por periodos de tiempo muy pequeños, podría tener una influencia significativa en el coste. Una forma de estimar esta influencia podría ser con herramientas que facilitasen la monitorización de la actividad de las instancias. No obstante, a día de hoy, estas herramientas no existen. En nuestro método, ante la dificultad de estimar este tiempo de penalización, se ha decidido presuponer que no influye en el rendimiento.

3 Minimización del coste de las instancias de cómputo

Nuestro método de minimización determina la combinación de instancias de cómputo que resulta más barata para ejecutar una aplicación en el entorno Amazon EC2. Como parte de la solución se tiene en cuenta el tiempo máximo establecido por el usuario y los distintos tipos de instancias, modelos de arrendamiento y opciones de pago ofertados por Amazon.

El coste total de cada instancia depende de dos factores. El primero es el precio a pagar por la propia instancia. En el caso de contratar instancias bajo demanda (*on-demand instances*) este precio depende fundamentalmente del tipo de instancia (generación de la instancia, región geográfica, o cararterísticas técnicas, entre otros) y del tiempo de utilización del recurso. Por otro lado, si se contratan instancias reservadas (*reserved instances*), el cálculo del precio es más complejo, dado que depende del plazo de reserva (*1-year reserved instances* o *3-year reserved instances*), del modelo de uso de la instancia (*light*, *medium*, y *heavy usage*) y de la opción de pago seleccionada (*All upfront*, *Partial Upfront*, o *No Upfront*), entre otros. En base a los criterios anteriores, el precio final de las instancias reservadas puede ser fijo, variable en función de su utilización o mixto. Por otro lado, el segundo factor que influye en el coste de una instancia es el precio a pagar por el almacenamiento local de información. En nuestro caso particular, el producto *Amazon Elastic Block Store* (EBS) proporciona esta capacidad de almacenamiento. Su precio depende del espacio de almacenamiento contratado (en *Giga-Bytes*, GB) y el tiempo que está siendo usada la instancia.

Otra cuestión importante en nuestro método es el rendimiento de cada instancia de cómputo. Este valor es dependiente del problema concreto a resolver y, en nuestro caso, se ha decidido calcular aplicando técnicas de experimentación. Por ejemplo, en la siguiente sección se aplica el método de costes a una aplicación concreta para el etiquetado semántico de documentos. En este caso particular, el concepto de tarea consiste en extraer de un documento sus palabras clave (llamadas *términos*) y, posteriormente, etiquetarlas semánticamente en base a una ontología de referencia. El rendimiento de una instancia será definido como el número de términos que es capaz de etiquetar por unidad de tiempo (por hora, por ejemplo). Dado que hay instancias que tienen más de un procesador, en esos casos, se ejecutan tantas tareas paralelas como procesadores disponibles, tratando de aprovechar al máximo la capacidad de cómputo de la instancia.

El problema a resolver ha sido formulado como una función con dos conjuntos de variables: el primero está formado por las distintas instancias lógicas que pueden ser contratadas para ejecutar una aplicación (una instancia lógica corresponde con un tipo de recurso concreto contratado bajo ciertas condiciones de arrendamiento y pago), y el segundo establece para cada tipo de instancia lógica el tiempo que tendrá que estar trabajando. El objetivo es minimizar el coste respectando el tiempo máximo de ejecución establecido por el usuario de la aplicación. La ecuación 1 define el problema formulado como un problema de programación lineal mixta que incluye los siguientes elementos:

- Cada instancia lógica está definida por las opciones de arrendamiento (plazo de contratación, modelo de uso y forma de pago) y el tipo de instancia concreta (número de cores, velocidad de la CPU, memoria, almacenamiento, etc.). PM e IT representan el número de opciones de arrendamiento y de instancias consideradas para ejecutar la aplicación, respectivamente. $N = (n_{11}, n_{12}, \dots, n_{ij}, \dots)$ son variables enteras que indican cuántas unidades de cada instancia lógica son necesarias para ejecutar la aplicación (para cada n_{ij}, el primer subíndice representa una opción de arrendamiento concreta y el segundo un tipo de instancia) y $T = (t_{11}, t_{12}, \dots, t_{ij}, \dots)$ el tiempo, en horas, que esas instancias serán utilizadas.
- T_{max} es el tiempo máximo de ejecución definido por el usuario de la aplicación, y W_{app} la carga de trabajo que implica ejecutar la aplicación (por ejemplo, en la aplicación de etiquetado semántico el número de términos a anotar).
- $CH = (ch_{11}, ch_{12}, \dots, ch_{ij}, \dots)$ es una matriz de números reales que determina el coste por hora de utilización de las diferentes instancias lógicas ($euro/hora$); mientras que, $CF = (cf_{11}, cf_{12}, \dots, cf_{ij}, \dots)$ es una segunda matriz de números reales que determina los costes fijos de esas instancias ($euro$), si los hubiera.
- $TPH = (tph_1, tph_2, \dots, tph_j, \dots)$ es un vector de números reales que determina el rendimiento de cada tipo de instancia.
- $size$ es la capacidad de disco contratada para cada instancia de cómputo (en GB), y c_{EBS} el precio por hora a pagar por ese almacenamiento local (en $GB/hora$).

$$
\begin{aligned}
\text{minimizar } & \sum_{i=1}^{PM}(\sum_{j=1}^{IT}(size \cdot c_{EBS} + ch_{ij}) \cdot t_{ij} + cf_{ij}) \cdot n_{ij} \\
\text{sujeto a } & \sum_{i=1}^{PM}\sum_{j=1}^{IT} t_{ij} \cdot n_{ij} \cdot tph_j \geq W_{app} \\
& 0 \leq n_{ij}, i \in \{1..\text{PM}\}, j \in \{1..\text{IT}\} \\
& 0 \leq t_{ij} \leq T_{max}, i \in \{1..\text{PM}\}, j \in \{1..\text{IT}\} \\
& t_{ij} \leq 24 \cdot 365, i = \text{Res-1-año}, j \in \{1..\text{IT}\} \\
& t_{ij} \leq 24 \cdot 3 \cdot 365, i = \text{Res-3-años}, j \in \{1..\text{IT}\}
\end{aligned}
\tag{1}
$$

La restricción $\sum_{i=1}^{PM}\sum_{j=1}^{IT} t_{ij} \cdot n_{ij} \cdot tph_j \geq W_{app}$ representa que la capacidad de cómputo contratada será suficiente para completar la ejecución de la aplicación. Es una desigualdad porque en la mayoría de los escenarios contratar el cómputo exacto para una determinada carga de trabajo es imposible (por ejemplo, en el caso de la aplicación semántica que se presenta a continuación). Por otro lado, $t_{ij} \leq 24 \cdot 365, i = \text{Res-1-año}, j \in \{1..\text{IT}\}$ impone que las instancias lógicas reservadas por un plaza de 1 año pueden ser usadas únicamente durante 1 año y, análogamente, $t_{ij} \leq 24 \cdot 3 \cdot 365, i = \text{Res-3-años}, j \in \{1..\text{IT}\}$ impone

la correspondiente restricción para las instancias reservadas por un plazo de 3 años.

Este problema de programación lineal mixta ha sido programado utilizando el lenguaje AMPL (*A Mathematical Programming language*, *http://ampl.com/*) y ejecutado por medio de las herramientas *filterSQP solver* y *NEOS server* (*Network-enabled Optimization System*) [15].

4 Caso de estudio

En [13] programamos una aplicación paralela *bag-of-task* para el etiquetado semántico de distintos recursos académicos (tesis doctorales, publicaciones científicas, revistas, etc.). En aquella experiencia la aplicación fue ejecutada utilizando los recursos de cómputo de una infraestructura clúster y dos infraestructuras Grid, ambas de naturaleza académica. El objetivo de esta sección es aplicar el método de minimización de costes propuesto para estimar qué combinación de instancias de cómputo sería la más adecuada para ejecutar determinados casos de prueba de la aplicación, y cuál sería el coste de contratar esas instancias.

Desde el punto de vista de la aplicación, como se mencionó con anterioridad, el rendimiento de las instancias de cómputo ofertadas por Amazon EC2 va a ser calculado por medio de la experimentación. Dado el amplio catálogo de instancias y opciones de arrendamiento ofertadas por el proveedor y el coste que supone realizar la experimentación, se ha decidido restringir el tipo de instancias que serán usadas en estos casos de prueba. De esta forma, se trabajará con instancias M1 (*m1.small*, *m1.medium*, *m1.xlarge* y *m1.2xlarge*), instancias M3 (*m3.xlarge* y *m3.2xlarge*), e instancias de alto rendimiento de E/S (*i2.xlarge* e *i2.2xlarge*). Además, en los experimentos se fija la región Amazon a la que pertenecen estas instancias, concretamente, la región *US West Oregon*. El hecho de seleccionar preferentemente modelos *xlarge* de esta región concreta es porque en un trabajo preliminar se concluyó que este tipo de instancias ofrecían un buen rendimiento para este problema de etiquetado. La tabla 1 muestra el rendimiento obtenido experimentalmente para cada una de las instancias mencionadas. Para cada instancia se detalla el número de cores, el tiempo que cuesta etiquetar un término y, por último, los términos que podrían ser etiquetados por hora.

Tabla 1. Rendimiento de las instancias de Amazon EC2 para el problema de etiquetado (precios del 1 de enero del 2015)

Tipo de instancia	m1.small	m1.medium	m1.large	m1.xlarge	m3.xlarge	m3.2xlarge	i2.xlarge	i2.2xlarge
Número de cores	1	1	2	4	4	8	4	8
Tiempo ejecución (seg/térm)	152.38	59.03	41.39	38.97	38.04	37.69	33.80	33.04
Capacidad cómputo (térms/hora)	23.63	60.99	173.96	369.52	378.55	764.13	4526.04	871.67

Por otro lado, se estimó que a nivel de disco local cada instancia de cómputo iba a necesitar 70 GB para almacenar la ontología utilizada para el proceso de anotación y los datos intermedios del etiquetado. El coste de EBS es de 0.776€ por GB-mes.

Supongamos que se pretende ejecutar la aplicación para etiquetar 10 millones de términos (W_{app}) en un tiempo máximo de 15 días (T_{max}). El problema de minimización da como resultado que el precio mínimo de las instancias de cómputo será de 5,784.33€. La combinación de instancias lógicas corresponde con 36 instancias *m3.2xlarge* bajo demanda contratadas durante 15 días y 1 instancia también *m3.2xlarge* bajo demanda contratada durante 5 días y 7 horas. Con estas instancias, el número de términos que podrían llegar a etiquetarse es de $10,000,148.37$ términos (el pequeño exceso se debe a que las instancias bajo demanda son aprovisionadas durante horas completas).

Como un segundo ejemplo consideremos el caso de etiquetar 100 millones de términos en 8 meses (240 días). En este escenario el precio mínimo sería de 53,913.77€ y la combinación de instancias lógicas constaría de 22 instancias *m3.2xlarge* reservadas durante el plazo de 1 año y pagadas por anticipado (pago *All-upfront*) y 1 instancia *m3.2xlarge* bajo demanda contratada durante 183 días y 8 horas. En este ejemplo, la recomendación de contratar instancias reservadas por el plazo de 1 año cuando sólo se van a utilizar 8 meses llama rápidamente la atención. Este resultado está justificado porque el precio de las instancias reservadas puede llegar a experimentar un descuento de hasta el 75% respecto el precio de las instancias bajo demanda; es decir, en este caso, resulta más económico "desperdiciar" 4 meses de cómputo de 22 instancias que optar por un modelo bajo demanda de compra de instancias. Por otro lado, este tipo de resultados abre nuevas líneas de trabajo con el fin de aprovechar estos excesos de cómputo ya pagados: podría usarse para reducir el coste de ejecutar otros procesos de etiquetado posteriores (a estos procesos se les podría incluso cobrar un precio simbólico por el consumo del exceso, devolviendo este importe como un descuento no previsto al usuario que contrató originalmente las instancias reservadas), o podría utilizarse para volver a procesar términos que hayan sido incorrectamente etiquetados (gestión de fallos), entre otros ejemplos.

5 Conclusiones y trabajo futuro

A día de hoy es difícil predecir cuál es el coste de ejecutar una aplicación paralela utilizando los recursos de cómputo y almacenamiento de un proveedor cloud. En este trabajo se ha propuesto un método para estimar este coste en el caso particular de ejecutar una aplicación *bag-of-task* en el entorno de computación Amazon EC2. El método no sólo determina una estimación del coste mínimo de la ejecución, sino también calcula qué instancias de cómputo deberían ser arrendadas y durante cuánto tiempo. Aunque es muy probable que el coste resultante sea inferior al coste que implicaría una ejecución real de la aplicación (en la sección 2 se introdujeron ciertos factores que incrementarán el coste final, pero cuyo impacto es difícil de predecir), proporciona una idea del precio mínimo a pagar para resolver un problema y de las instancias que podrían ser aprovisionadas en base a criterios de rendimiento y precio. En nuestro caso particular, hemos aplicado el método propuesto para estimar el coste de

ejecutar una aplicación de etiquetado semántico para distintas cargas de trabajo proporcionadas por el usuario.

Una discusión interesante es en qué medida este método puede ser reutilizado para estimar el coste de ejecución de otros modelos de aplicación y/o proveedores de infraestructura cloud. En el caso de las aplicaciones, el método está muy ligado a las características de las aplicaciones *bag-of-task* intensas en cómputo. Se ha valorado la posibilidad de extender las ecuaciones de costes para el caso de aplicaciones *bag-of-task* intensas en datos, pero encontrar una fórmula general para cualquier aplicación es una tarea compleja debido al abanico de posibilidades en torno a la distribución, almacenamiento y acceso a los datos. Evidentemente, otros modelos de aplicación (por ejemplo, formadas por tareas dependientes entre si, como un *workflow*) requieren de ecuaciones de coste alternativas. Por otro lado, la decisión de definir instancias lógicas en el método de estimación abre la posibilidad de considerar otros tipos de máquinas ofertadas por distintos proveedores cloud (*Google Cloud Platform* o *Windows Azure*, por ejemplo); es decir, sería posible aplicar el método a otros proveedores concretos o a una combinación de varios proveedores (en este último caso, un subconjunto de las tareas programadas sería ejecutado por cada proveedor; esto es posible gracias a que se tratan de tareas independientes).

Por último, respecto al trabajo futuro, los esfuerzos a medio plazo se centran en las siguientes cuestiones. Primero, sería deseable la búsqueda de técnicas alternativas a la experimentación como medio de evaluar el rendimiento de las instancias de cómputo para un determinado problema. El uso de técnicas de simulación (utilizando el simulador *CloudSim* [16], por ejemplo) podría ser una alternativa, evitando el coste económico que implica la experimentación. Segundo, también sería interesante encontrar funciones de distribución que modelen la variabilidad en el rendimiento de las instancias que introduce el uso de la virtualización y, posteriormente, estudiar cómo integrar en el método estas funciones. Finalmente, es necesario validar las estimaciones de coste obtenidas utilizando herramientas que monitoricen el rendimiento real de las instancias. No obstante, hoy en día, el estado de estas herramientas es aún inmaduro.

Agradecimientos

Este trabajo está siendo financiado por el proyecto TIN2014-56633-C3-2-R del Programa Estatal de Investigación, Desarrollo e Innovación orientada a los Retos de la Sociedad (Plan Estatal de Investigación Científica y Técnica y de Innovación, 2013-2016) del Ministerio de Economía y Competitividad, por el proyecto UZ-2014-TEC-04 de la Universidad de Zaragoza, y por la Beca de Formación de Profesorado Universitario FPU12/04775.

Bibliografía

1. Deelman, E., Singh, G., Livny, M., Berriman, B., Good, J.: The cost of doing science on the cloud: The montage example. In: Proceedings of the 2008

ACM/IEEE Conference on Supercomputing. SC '08, Piscataway, NJ, USA, IEEE Press (2008) 50:1–50:12
2. Juve, G., Deelman, E., Berriman, G., Berman, B., Maechling, P.: An evaluation of the cost and performance of scientific workflows on amazon ec2. Journal of Grid Computing **10**(1) (2012) 5–21
3. Turcu, G., Foster, I., Nestorov, S.: Reshaping text data for efficient processing on amazon ec2. Sci. Program. **19**(2-3) (April 2011) 133–145
4. Truong, H.L., Dustdar, S.: Cloud computing for small research groups in computational science and engineering: Current status and outlook. Computing **91**(1) (January 2011) 75–91
5. De Alfonso, C., Caballer, M., Alvarruiz, F., Moltó, G.: An economic and energy-aware analysis of the viability of outsourcing cluster computing to a cloud. Future Generation Computer Systems **29**(3) (March 2013) 704–712
6. Kashef, M.M., Altmann, J.: A cost model for hybrid clouds. In: Proceedings of the 8th international conference on Economics of Grids, Clouds, Systems, and Services. GECON'11 (2012) 46–60
7. Truong, H.L., Dustdar, S.: Composable cost estimation and monitoring for computational applications in cloud computing environments. In: Proceedings of the International Conference on Computational Science. ICCS '10 (2010) 2175–2184
8. Malawski, M., Juve, G., Deelman, E., Nabrzyski, J.: Cost- and deadline-constrained provisioning for scientific workflow ensembles in iaas clouds. In: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis. SC '12, Los Alamitos, CA, USA, IEEE Computer Society Press (2012) 22:1–22:11
9. Trivedi, N., Chudasama, D.: Dynamic resource provisioning for deadline and budget constrained application in cloud environment. International Journal Computer Technology and Applications **4**(3) (2013) 462–465
10. Rodríguez, M.A., Buyya, R.: Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds. IEEE Transactions on Cloud Computing **2**(2) (2014) 222–235
11. Mao, M., Li, J., Humphrey, M.: Cloud auto-scaling with deadline and budget constraints. In: Proceedings of the 11th IEEE/ACM International Conference on Grid Computing. (2010) 41–48
12. Gartner: Gartner magic quadrant methodology. Available at http://www.gartner.com/technology/research/methodologies/research_mq.jsp (2014)
13. Fabra, J., Hernández, S., Otero-García, E., Vidal, J.C., Lama, M., Álvarez, P.: A practical experience concerning the parallel semantic annotation of a large-scale data collection. In: The 9th International Conference on Semantics Systems. I-SEMANTICS '13 (2013) 65–72
14. Schad, J., Dittrich, J., Quiané-Ruiz, J.A.: Runtime measurements in the cloud: Observing, analyzing, and reducing variance. Proc. VLDB Endow. **3**(1-2) (September 2010) 460–471
15. Czyzyk, J., Mesnier, M.P., Moré, J.J.: The neos server. IEEE Journal on Computational Science and Engineering **5**(3) (July 1998) 68–75
16. Wickremasinghe, B., Calheiros, R., Buyya, R.: Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications. In: Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications. AINA '10 (2010) 446–452