Tesis defendida por

Luz María Lozano Guzmán

y aprobada por el siguiente Comité

Andrey Chernykh
Director del Comité

Carlos Alberto Brizuela Rodríguez
Miembro del Comité

Jesús Favela Vara
Miembro del Comité

José Eleno Lozano Rizk
Miembro del Comité

José Antonio García Macías

Coordinador
del Posgrado en Ciencias de la Computación

Dr. Jesús Favela Vara

Encargado del despacho de la
Dirección de Estudios de Posgrado

Septiembre 2013

CENTRO DE INVESTIGACIÓN CIENTÍFICA Y DE EDUCACIÓN SUPERIOR DE ENSENADA

Programa de Posgrado en Ciencias
en Ciencias de la Computación

Análisis experimental de estrategias de calendarización en la nube considerando calidad de servicio y eficiencia energética

Tesis
que para cubrir parcialmente los requisitos necesarios para obtener el grado de Maestro en Ciencias

Presenta:
Luz María Lozano Guzmán

Ensenada, Baja California, México
2013

Resumen de la tesis de Luz María Lozano Guzmán**,** presentada como requisito parcial para la obtención del grado de Maestro en Ciencias en Ciencias de la Computación.

Análisis experimental de estrategias de calendarización en la nube considerando calidad de servicio y eficiencia energética

Resumen aprobado por:

_____

Dr. Andrey Chernykh

En esta tesis nos centramos en el análisis de algoritmos energéticamente eficientes para calendarización en la nube, bajo el modelo de infraestructura como servicio (IaaS). Calendarizamos trabajos de computo de alto desempeño, los cuales llegan al sistema conforme el tiempo transcurre. El modelo de calendarización está basado en Niveles de Servicio (SL), esto para tener un control de admisión por parte del proveedor del servicio y tener garantías de calidad de servicio para el usuario. Cada nivel de servicio está asociado a un precio por unidad de tiempo de ejecución y un factor de holgura que proporciona el marco de tiempo en el cual el trabajo tiene que ser entregado al usuario, este marco de tiempo es conocido como fecha límite y es la forma en que están expresadas las garantías QoS. Así una vez que un trabajo es sometido al sistema el proveedor decide, si puede mantener las garantías de calidad de servicio, entonces acepta el trabajo y le proporciona al usuario una fecha máxima en la cual el trabajo finalizará su ejecución. La meta es encontrar algoritmos que proporcionen buen compromiso entre maximización de ganancia y minimización de energía. Nuestra experimentación y análisis muestra que balancear el consumo de energía por cada máquina puede dar un equilibrio entre consumo de energía del sistema y ganancia total de proveedor, manteniendo calidad de servicio.

Palabras clave**: Infraestructura como Servicio, Cómputo en la Nube, Niveles de Servicio, Calendarización, Eficiencia energética.**

Abstract of the thesis presented by Luz María Lozano Guzmán as a partial requirement to obtain the Master in Science degree in Computer Science

Experimental analysis of cloud scheduling strategies considering quality of services and energy efficiency

Abstract approved by:

_____

Dr. Andrey Chernykh

In this thesis we focus on the analysis of energy-efficient algorithms for online scheduling in Infrastructure as a Service (IaaS) type of Cloud, we work with HPC jobs. In order to have an admission control for the submitted jobs, the scheduling model is based on Service Levels (SL). We have quality of service guarantees for the user. Each service level is associated to a price per unit of execution time and slack factor that provides the time frame in which the work must be delivered to the user, this frame of time is known as deadline and the achievement of this deadline is how the QoS guarantees are expressed in this work. In the scheduling model, once a job is submitted to the system, the system provider has to decide whether the arriving job can be accepted or not, it depends on the QoS guarantees, if the system can maintain the quality of service guarantees for all the jobs already accepted and also give the required service to the arriving job, then the system accepts the job. The goal is to find algorithms that provide good compromise between profit maximization and energy consumption minimization. Our experiments and analysis have shown that if we have energy-efficiency information about the infrastructure, we can generate schedules where we can reduce the energy consumption while maintaining service quality.

Keywords: **Infrastructure as a Service, Cloud Computing, Service Levels, Scheduling, Energy Efficiency.**

Este trabajo está dedicado a Rafael Blanco
mi persona incondicional

Agradecimientos

Quiero agradecer primeramente a Dios por permitirme alcanzar esta meta.

A Rafael y a cada miembro de mi familia por estar para mí cada vez que he necesitado de ustedes.

Al Dr. Chernykh por guiarme en esta etapa de mi desarrollo académico, a los miembros de mi comité: M.I. Lozano, Dr. Brizuela y Dr. Favela por sus comentarios y consejos.

A los compañeros que dejaron cosas buenas en mi vida.

Agradecimiento especial para el Dr. Pecero y el equipo de la universidad de Luxemburgo por sus aportaciones a mi trabajo y por el soporte durante mi estancia en dicha universidad.

Finalmente a CICESE por la oportunidad y CONACyT por la beca otorgada para la realización de está investigación.

Contenido

	Página
Resumen español……	i
Resumen ingles……	ii
Dedicatorias……	iii
Agradecimientos……	iv
Lista de Figuras……	vii
Lista de Tablas……	ix
Capítulo 1. Introducción……	1
1.1 Cómputo en la nube……	1
1.1.1 Arquitectura de la nube……	1
1.1.2 Modelos de servicio de la nube……	3
1.1.3 Tipos de nubes……	4
1.1.4 Acuerdos de Nivel de Servicio……	5
1.2 Descripción del problema……	6
1.3 Justificación……	7
1.4 Objetivos……	7
1.4.1 Generales……	8
1.4.2 Específicos……	8
1.4.3 Preguntas de investigación……	9
Capítulo 2. Descripción del modelo……	10
2.1 Escenario IaaS……	10
2.2 Definición formal del problema……	12
2.2.1 Modelo de energía……	14
2.2.2 Métricas……	15
2.3 Metodología del análisis……	18
2.3.1 Degradación……	19
2.3.2 Perfiles de desempeño……	20
2.3.3 Pareto……	21
2.4 Trabajo relacionado……	22
Capítulo 3. Algoritmos de calendarización ……	27
3.1 Función de costos……	27
3.2 Parámetros del factor de competitividad……	29
3.3 Dinámica de realización de calendarios IaaS……	31
3.4 Política de aceptación……	32
3.5 Estrategias de asignación……	33
Capítulo 4. Configuración experimental……	35
4.1 Máquinas idénticas……	35
4.1.1 Múltiples niveles de servicio ……	35
4.2 Máquinas heterogéneas……	37
4.2.1 Heterogeneidad……	37
4.3 Cargas de trabajo……	39

Página
Capítulo 5. Análisis experimental y resultados…….…….. 42
5.1 Único nivel de servicio, una máquina ……. 42
5.2 Único nivel de servicio y múltiples máquinas idénticas…..… 44
5.2.1 Factor de competitividad SSL-MM……. 44
5.2.2 Consumo de energía SSL-MM……. 46
5.3 Múltiples niveles de servicio, múltiples máquinas idénticas…. 48
5.3.1 Factor de competitividad MSL-MM……. 48
5.3.2 Consumo de energía MSL-MM……. 51
5.4 Único nivel de servicio, múltiples máquinas heterogéneas..... 52
5.4.1 Factor de competitividad SSL-MM_h……. 52
5.4.2 Consumo de energía SSL-MM_h…….. 54
5.5 Múltiples niveles de servicio, múltiples máquinas…….…… 56
5.5.1 Factor de competitividad MSL-MM_h……. 56
5.5.2 Consumo de energía MSL-MM_h……. 58
5.6 Análisis: speed-aware y energy-aware…..……. 59
5.6.1 Ganancia…….……. 60
5.6.2 Consumo energético…….. 67
Capítulo 6. Análisis bi-objetivo…….…… 73
6.1 Análisis de degradaciones……. 73
6.2 Perfiles de desempeño……. 76
6.3 Pareto…….……. 78
6.3.1 Espacio de funciones objetivo y Frentes de Pareto…… 79
6.3.2 Cubrimiento de Conjunto…..…….……. 84
Capítulo 7. Conclusiones…….. 86
Referencias bibliográficas……..……. 88

Apéndices ……..... 91
A1. Configuración heterogénea para el sistema……..……. 92
A2. Ganancia total generada, gráficas por escenario……. 93
A3. Degradación de ganancia, gráficas por escenario…… 94
A4. Consumo de energía, gráficas por escenario…….. 95
A5. Degradación del consumo de energía, gráficas por escenario… 97

Lista de figuras

Figura		Página
1	Función de precios y función de costos de ejecución de trabajos……	29
2	Diagrama de calendarización a dos niveles.....	32
3	Histograma de velocidades y eficiencia energética.....	38
4	Factor de competitividad SSL-SM……	43
5	Ganancia generada y ganancia óptima, SSL-SM……	43
6	Factor de competitividad, SSL-MM……	44
7	Ganancia total generada con SSL-MM……..	45
8	Ganancia promedio por máquina, usando SSL-MM……..	46
9	Consumo total de energía, SSL-MM……	47
10	Degradación del consumo de energía, SSL-MM……	47
11	Factor de competitividad, MSL-MM……	49
12	Ganancia total generada con MSL-MM……..	49
13	Ganancia promedio generada por cada máquina, MSL-MM…..……..	50
14	Energía total consumida MSL-MM……..……	51
15	Degradación del consumo de energía MSL-MM……..……..	52
16	Factor de competitividad, SSL-MM-het……..……	53
17	Ganancia total generada con SSL-MM-het……..……..	53
18	Ganancia promedio por máquina, con SSL-MM-het……	54
19	Consumo de energía, SSL-MM-het……	55
20	Degradación del consumo de energía, SSL-MM-het……..	55

Página

Figura

21	Factor de competitividad, MSL-MM-het……	56
22	Ganancia total generada, MSL-MM-het……	57
23	Ganancia promedio generada por cada máquina, MSL-MM-het…….	57
24	Energía total consumida, MSL-MM-het……	58
25	Degradación del consumo de energía, MSL-MM-het……..…….	58
26(a)	Ganancia promedio por máquina al usar SLA con un nivel de servicio..	60
26(b)	Ganancia total generada, al usar SLA con un nivel de servicio……	61
26(c)	Degradación de ganancia, al usar SLA con un nivel de servicio……..	61
27(a)	Ganancia promedio generada por máquina al usar SLA con 2 SLs……	62
27(b)	Ganancia total generada al usar SLA con 2 SLs……	62
27(c)	Degradación de ganancia al usar SLA con 2 SLs……..	63
28(a)	Ganancia promedio generada por máquina al usar SLA con 3 SLs…..	63
28(b)	Ganancia total generada al usar SLA con 3 SLs……	64
28(c)	Degradación de ganancia al usar SLA con 3 SLs……..	64
29(a)	Ganancia promedio generada por máquina al usar SLA con 4 SLs…….	65
29(b)	Ganancia total generada al usar SLa con 4 SLs……..	65
29(c)	Degradación de ganancia al usar SLA con 4 SLs……	65
30(a)	Ganancia promedio generada por máquina al usar SLA con 5 SLs……	66
30(b)	Ganancia total genrada al usar SLA con 5 SLs……	66
30(c)	Degradación de ganancia al usar SLA con 5 SLs……..	66

Figura		Página
31(a)	Consumo total de energía usando SLA con un nivel de servicio……..	68
31(b)	Degradación del consumo energético usando SLA con un SL……..	68
32(a)	Consumo total de energía usando SLA con 2 SLs……..	69
32(b)	Degradación del consumo energético usando SLA con 2 SLs……..	69
33(a)	Consumo total de energía usando SLA con 3 SLs……..	70
33(b)	Degradación del consumo energético usando SLA con 3 SLs……..	70
34(a)	Consumo total de energía usando SLA con 4 SLs……..	71
34(b)	Degradación del consumo energético usando SLA con 4 SLs……..	71
35(a)	Consumo total de energía usando SLA con 5 SLs……..	72
35(b)	Degradación del consumo energético usando SLA con 5 SLs……..	72
36	Perfiles de desempeño con respecto a la ganancia generada, 8 estrategias evaluadas……..	77
37	Perfiles de desempeño con respecto de la energía consumida, 8 estrategias evaluadas.....	77
38	Perfiles de desempeño con respecto al promedio de energía y ganancia, 8 estrategias evaluadas.....	78
39	Espacio de funciones objetivo y frente de Pareto para la estrategia Random……..	80
40	Espacio de funciones objetivo y frente de Pareto para la estrategia MLp……..	80
41	Espacio de funciones objetivo y frente de Pareto para la estrategia FFit……..	81
42	Espacio de funciones objetivo y frente de Pareto para la estrategia Max_s……....	81

Figura		Página
43	Espacio de funciones objetivo y frente de Pareto para la estrategia Max_eff……	82
44	Espacio de funciones objetivo y frente de Pareto para la estrategia Max_eff……..	82
45	Espacio de funciones objetivo y frente de Pareto para la estrategia MCT_eff……	83
46	Espacio de funciones objetivo y frente de Pareto para la estrategia Min_e……..…	83
47	Frentes de Pareto de las ocho estrategias evaluadas……	84

Lista de tablas

Tabla		Página
1	Métricas secundarias……..	17
2	Estrategias de asignación…….	34
3	Distribución de niveles de servicio, sobre SLAs disponibles……..….	36
4	Detalles de velocidad y energía……..	37
5	Grids y bitácoras de trabajos sometidos…….….	39
6	Degradaciones y Ranking……..	73
7	Degradación y Ranking promedio de todos los SLAs……..	75
8	Cubrimiento de conjunto…….	85

Capítulo 1

Introducción

En este capítulo presentamos el entorno del cómputo en la nube, describimos conceptos importantes que se usan en esta investigación. Abordamos la problemática en la cual está centrada esta tesis, justificamos la investigación realizada y planteamos los objetivos.

1.1 Cómputo en la Nube

El cómputo en la nube es un término que surgió recientemente y ha tomado gran fuerza. El instituto nacional de estándares y tecnología de Estados Unidos (NIST) ha definido el cómputo en la nube como un modelo que permite, de manera ubicua y conveniente, proveer bajo demanda y a través de la red, recursos informáticos configurables, tales como: redes, almacenamiento, aplicaciones y servicios (Mell et al. 2011). Este modelo permite que los recursos computacionales sean liberados al cliente de manera rápida y sin necesidad de mucha interacción entre el cliente y el proveedor del servicio; normalmente el cliente no conoce la ubicación física de los recursos.

El modelo de cómputo en la nube se compone de una arquitectura con cinco características esenciales (servicio bajo-demanda, ubicuidad, elasticidad, recursos virtualizados y servicio medido), tres modelos de servicio (Software como servicio, Plataforma como servicio e Infraestructura como servicio) y tres tipos de implementación (públicas, privadas e híbridas) que presentamos en próximos capitulos.

1.1.1 Arquitectura de la nube

La arquitectura de la nube está basada en el principio del navegador web, donde se cuenta con dos terminales: usuario y proveedor. La terminal del usuario de la

nube es la interfaz donde el usuario accede a los recursos de la nube, generalmente a través de un navegador web. En la terminal del proveedor de servicios se encuentra la red de servidores que contiene las aplicaciones, almacenaje y demás recursos para ofrecer al usuario. A éste se le presenta la idea de que los recursos son ilimitados, es decir; capacidad de almacenaje ilimitada, todas las aplicaciones disponibles en el mercado, etc.

La nube tiene un sistema de administración centralizado, donde un servidor administra el sistema, equilibra la oferta para el cliente, ajusta las demandas, monitorea el tráfico y evita las congestiones. Dicho servidor es lo que comúnmente se conoce como el *middleware* de la nube.

El cómputo en la nube cuenta con cinco características esenciales:

1) Servicio bajo-demanda, el usuario puede abastecer de manera unilateral las capacidades de cómputo conforme a sus necesidades y sin necesidad de interacción personal con el dueño de la nube. Las capacidades de cómputo le serán entregadas en el momento que él las solicite, en la cantidad que el usuario requiere.
2) Ubicuidad, el servicio es ofrecido a través del internet y permite el acceso a través de plataformas heterogéneas, ya sea desde celulares, tablets, laptops o computadoras de escritorio.
3) Elasticidad, las capacidades computacionales pueden cambiar de manera elástica, ya sea crecer o decrecer. El usuario decide qué recursos quiere y en qué cantidad, así el usuario tiene la visión de que los recursos son ilimitados.
4) Recursos virtualizados, los recursos físicos son virtualizados para así tener la capacidad de entregar recursos a la medida de lo que el usuario requiere. Esto también permite que los servidores trabajen a su máxima capacidad, teniendo varias máquinas virtuales ejecutando aplicaciones similares o diferentes.

5) Servicio medido, los sistemas en la nube automáticamente monitorean, controlan y reportan el uso de los recursos, así se logra un manejo transparente en la utilización de recursos.

1.1.2 Modelos de servicio de la nube

Una vez que los recursos físicos están virtualizados y cada máquina es capaz de correr aplicaciones, ya sean iguales o diferentes, es posible hacer distinción entre los servicios a ofrecer al usuario, es así como la nube cuenta con tres principales modelos de servicio definidos por el NIST (Mell at al. 2011).

Software como servicio (SaaS) es el modelo de servicio que va dirigido a los usuarios de más alto nivel, usuarios de aplicaciones. El software como servicio ofrece aplicaciones informáticas bajo demanda, de manera que el usuario accede a la nube a través de diferentes interfaces tal como navegadores web o algún otro programa de interfaz. El usuario utiliza las aplicaciones, pero no administra ni controla la infraestructura de la nube, solamente se le permite la modificación de las opciones de la aplicación. Un ejemplo de este típo de servicio es google docs, donde el usuario puede modificar el documento desde diferentes plataformas y el documento se encuentra alojado en la nube.

Plataforma como servicio (PaaS) es el modelo de servicio dirigido a desarrolladores, ya que se ofrece bajo demanda, el entorno de desarrollo y almacenaje. El usuario accede a las herramientas de desarrollo (lenguajes de programación, librerías, etc.) y puede desarrollar aplicaciones sobre la nube, tiene derechos sobre la modificación de todo el entorno de desarrollo, pero al igual que el usuario de SaaS, no controla la infraestructura de la nube. Un ejemplo de PaaS es AppEngine, un entorno para desarrolladores basado en Python y Java. Los usuarios pueden tienen disponible el entorno cada vez que lo requieren.

Infraestructura como servicio (IaaS) es el modelo de servicio dirigido para usuarios de cómputo de alto desempeño, es decir, usuarios que requieren mucho poder de

procesamiento, almacenaje, etc. En general los usuarios de IaaS pueden desarrollar cualquier tipo de software, desde aplicaciones hasta sistemas operativos. El usuario de la infraestructura como servicio (IaaS) obtiene una máquina virtual, con las capacidades y características que requiere y puede manejar todas las opciones de configuración de la máquina virtual, desde el tipo de sistema operativo hasta qué aplicaciones instalar, etc. Pero no tiene acceso al control de la infraestructura de la nube. Un ejemplo de IaaS es Amazon EC2, quien ofrece diferentes tipos de IaaS a la medida de los usuarios: Reservaciones por año, reservaciones puntuales y reservaciones bajo demanda. Estos ejemplos los abordamos en la sección 2.1.

1.1.3 Tipos de nubes

Existen tres diferentes tipos de implantación de nubes, cada tipo tiene diferente disponibilidad de acceso a la nube.

En el caso de las *nubes públicas*, el proveedor es el propietario de la nube y se encarga de la administración de la misma. Los usuarios tienen acceso a los recursos por medio de una suscripción, ésta la realizan los usuarios a través de la web, normalmente. Una vez que el usuario se registra y hace la solicitud de recursos, el proveedor autoriza la entrega de los mismos y se procede con el servicio, el cual se provee por el periodo de tiempo establecido en el contrato. Una característica remarcable en éste tipo de implantación es que los recursos son usualmente en mayor cantidad. Un problema en este tipo de implantación de nube es la seguridad de los datos, el administrador debe garantizar a sus usuarios que sus datos están seguros.

En el modelo de *nube privada* los recursos son dedicados al cliente y el acceso a los mismos es manejado por el cliente. Éste tipo de implantación de nube hace referencia a redes o centros de procesamiento de datos propietarios que utilizan tecnologías características de cómputo en la nube, tales como la virtualización.

Así, partiendo de los principios del cómputo en la nube tradicional, se ofrecen los mismos servicios pero dentro de la propia estructura de la compañía. La característica distintiva es que éste tipo de nubes posea un nivel de seguridad muy alto.

El tercer tipo es el de las *nubes híbridas*, combina los modelos de nubes públicas y privadas, de manera que se busca aprovechar la ventaja de localización física de la información gestionada por las nubes privadas, con la facilidad de ampliación de recursos de las nubes públicas. Sin embargo, al fusionar los tipos de implantación también se heredan problemas tales como la necesidad de vigilar la privacidad y la protección de datos, al igual que en la nube pública.

1.1.4 Acuerdos de nivel de servicio

Al hacer uso de los recursos ofrecidos por las nubes se requiere la realización de un contrato donde se especifiquen las responsabilidades y derechos del usuario con respecto del servicio que está adquiriendo, así como la especificación de las responsabilidades y ganancias del proveedor de servicio. Dicho trabajo lo desempeñan los Acuerdos de Nivel de Servicio (SLA), los cuales son contratos virtuales que se encargan de establecer los términos en que se provee el servicio solicitado.

Los SLA deben contener la lista de servicios disponibles, así como la descripción completa de cada servicio, las métricas que aseguren que el proveedor está entregando el servicio en las condiciones y términos establecidos, así como un mecanismo que monitoree el servicio, las responsabilidades del proveedor y del usuario, también debe especificar qué pasa en caso de incumplimiento de contrato. Por último, la descripción de cómo el acuerdo va a cambiar conforme el tiempo trascurra.

Las responsabilidades del proveedor de servicios son establecidas como el nivel de Calidad de Servicio (QoS) que se debe mantener y la responsabilidad del

usuario se manifiesta como el pago por el servicio recibido. Normalmente las penalidades por incumplimiento de contrato son en la forma de remuneraciones económicas.

En IMB (2010) se presentan dos tipos de SLAs, el primer tipo se ofrece en nubes públicas y normalmente no son negociables, lo cual en determinados escenarios no es aceptable. El segundo tipo son los SLA personalizables, donde se permite la negociación de términos entre cliente y proveedor hasta llegar a un acuerdo satisfactorio para ambas partes. En éste tipo de SLA personalizables se presenta un mecanismo de negociación, lo cual es un tema en proceso de ser investigado para la obtención de resultados satisfactorios en negociaciones automáticas.

1.2 Descripción del problema

En la sección 1.1.2, hablamos sobre los modelos de servicio ofrecidos por la nube, uno de ellos, Infraestructura como Servicio (IaaS), representa el entorno en que se lleva a cabo la investigación presentada en esta tesis. El escenario genérico que usamos es el típico para la Infraestructura como servicio, donde el usuario de la nube accede al servicio a través de la web, ahí encuentra el contrato virtual SLA, donde puede seleccionar un nivel de servicio (SL) para el servicio que requiere, también se especifica el monto a pagar, el cual depende del SL seleccionado. Por parte del proveedor de servicios se establecen las garantías de calidad de servicio, en este trabajo son expresadas como el tiempo máximo de respuesta, lo cual también depende del nivel de servicio que el usuario seleccionó.

Una vez que el usuario está de acuerdo con los términos del SLA, se procede a la verificación de recursos. En esta etapa el proveedor tiene que asegurar que puede brindar el servicio sin afectar los contratos que ya aceptó, es decir, el proveedor puede otorgar el servicio requerido solo si puede cumplir con los requerimientos del contrato recién adquirido y no perder ningún otro contrato ya firmado.

La problemática que encontramos en este punto es la creciente demanda, cuando el proveedor tiene más demanda de servicios que recursos para proveer. Aquí requerimos algoritmos que implementen un control de admisión y así poder asegurar que cada contrato será cumplido en tiempo y forma de acuerdo a las garantías de QoS acordadas. Adicionalmente nos interesa reducir el consumo de energía desde el punto de vista de la asignación eficiente de recursos e incrementar la ganancia del proveedor.

1.3 Justificación

Existen varios estudios sobre asignación de recursos en el cómputo en la nube usando SLAs como mediadores entre cliente y proveedor, en ellos se busca la asignación eficiente de recursos, también hay varios trabajos en los que se busca reducir el consumo energético, pero hasta ahora no hemos encontrado entre la bibliografía, un trabajo en el que el estudio tenga en conjunto las siguientes características: uso de niveles de servicio, infraestructura con máquinas heterogéneas, ganancia para el proveedor y eficiencia energética en calendarización. El uso de niveles de servicio es un elemento clave en servicios que se proveen bajo-demanda, ya que es una forma de mantener prioridades, un reto es encontrar qué niveles de servicio usar para tener un manejo adecuado de los usuarios y de los recursos. El uso de infraestructura con máquinas heterogéneas es importante para tener una simulación más cercana al entorno real y el buscar la eficiencia energética en sistemas tales como la nube es un tema importante en más de un aspecto, en busca de las tecnologías verdes y el ahorro económico para el dueño de la nube.

1.4 Objetivos

En las siguientes secciones presentamos los objetivos de esta investigación. Partimos del objetivo general y desglosamos los objetivos específicos en términos de experimentación.

1.4.1 Objetivos generales

Diseño, implementación y análisis experimental de algoritmos bi-objetivo de calendarización para nubes tipo IaaS, los cuales buscan maximizar la ganancia para el proveedor de servicio, así como la reducción del consumo energético. Todo esto considerando restricciones de calidad de servicio bajo un modelo de máquinas heterogéneas.

1.4.2 Objetivos específicos4

Primer objetivo: evaluar una política de aceptación voraz para el modelo con múltiples niveles de servicio y múltiples máquinas paralelas idénticas (MSL-MM), usando la estrategia de asignación a la primera máquina disponible.

Segundo objetivo: evaluar el modelo con múltiples niveles de servicio y múltiples máquinas heterogéneas (MSL-MM-het) con diferentes características de velocidades y eficiencia energética, usando la estrategia de asignación a la primera máquina disponible.

Tercer objetivo: evaluar el modelo MSL-MM-het usando estrategias de asignación a todos los recursos disponibles: considerando diferentes niveles de información que requieren para tomar la decisión de asignación: *knowledge-free* (donde no tenemos información sobre los parámetros de las máquinas), *power-aware* (donde tenemos información acerca del consumo energético de las máquinas) y *speed-aware* (donde conocemos el factor de velocidad con que las máquinas procesan una unidad de tiempo). Para alcanzar este objetivo nosotros proponemos el diseño, implementación y análisis de varias estrategias de asignación.

Para hacer la evaluación, consideramos dos métricas principales: ganancia y consumo de energía.

Para cada etapa tenemos que:

- Definir escenarios de simulación,
- Definir y analizar cargas de trabajo para la simulación,
- Ejecutar las simulaciones de los diferentes escenarios,
- Analizar los resultados de las simulaciones.

1.4.3 Preguntas de investigación

Algunas preguntas de investigación que nos ayudan a realizar el análisis de los resultados de las simulaciones son:

- ¿Qué clase de garantías de desempeño pueden ser aseguradas cuando el calendarizador está basado en el modelo de niveles de servicio?
- ¿Cuál es la relación entre consumo de energía y desempeño?
- ¿Cómo el considerar el consumo de energía cambia la calidad de los calendarios?
- ¿Cómo el considerar máquinas heterogéneas puede incrementar la complejidad del modelo?

Capítulo 2

Descripción del Modelo

En este capítulo describimos el escenario particular de Infraestructura como servicio (IaaS) que usamos en la investigación, así como la descripción formal del modelo utilizado, también definimos formalmente los algoritmos de calendarización, la configuración experimental para cada uno de los algoritmos y la carga de trabajo utilizada en las simulaciones.

2.1 Escenario IaaS

El escenario típico de Infraestructura como Servicio consiste en la liberación de recursos computacionales virtualizados (máquinas virtuales) que incluye tanto hardware (almacenaje y servidores) como software (sistema operativo y archivos del sistema), la dinámica consiste en un usuario, quien demanda el servicio y un proveedor, quien libera el servicio.

El usuario accede a la nube a través de internet donde encuentra la interfaz de la nube, normalmente en la forma de un portal web, ahí puede elegir entre las diferentes formas de IaaS que ofrece el proveedor:

(1) Reservación por instancias (reserved instance), donde el usuario hace un pago único por cada instancia que quiere reservar, en esta modalidad la finalización de la reservación varía de un año a tres años y dependiendo de la utilización que el usuario pretende tener (ligera, media, intensa), también puede escoger qué tipo de instancia le acomoda mejor de entre una gama de tipos de instancias reservadas que ofrece el proveedor. Éste tipo de instancias puede dar al usuario un ahorro de entre 21% y 62% en un periodo de 3 años, en comparación con la utilización de instancias bajo-demanda (Amazon WS 2013).

(2) Instancias bajo-demanda (on-demand instances), son las instancias donde el usuario paga el servicio por horas y sin un compromiso a largo plazo, normalmente el usuario paga únicamente por el tiempo que está utilizando los recursos sin tener intervalos de tiempo sin utilización (Amazon WS 2013).

(3) Instancias puntuales (spot instances), las principales diferencias entre las instancias puntuales y las instancias bajo demanda son que las instancias puntuales pueden no ponerse en funcionamiento de forma inmediata y el precio por hora es variable. Son instancias de reservación para las que el usuario puede fijar el precio que desee. El precio puntual lo establece el proveedor y fluctúa en tiempo real en función de la oferta y la demanda de instancias puntuales. Si la oferta supera el precio puntual, la instancia puntual se lanza y se ejecuta hasta que el precio puntual es excedido por la oferta que haga otro usuario, esto genera una interrupción puntual, o hasta que el trabajo asignado en la instancia finaliza. Éste tipo de instancias son deseables para aplicaciones que tienen flexibilidad de ejecución (Amazon WS 2013).

El escenario de IaaS que nosotros usamos es un híbrido entre las instancias bajo-demanda y las instancias puntuales, ya que en nuestro modelo el usuario somete trabajos de HPC y no requiere que se le entregue una máquina virtual configurable, sino que requiere el poder de cómputo para ejecutar su trabajo y los trabajos permiten ser interrumpidos (característica necesaria en instancias puntuales), el usuario somete el trabajo que desea ejecutar y le asigna un nivel de servicio. El proveedor, en caso de aceptar el trabajo sometido, garantiza al usuario que antes de la fecha límite se le entregará su trabajo ya ejecutado. Así desde el punto de vista del usuario, se está rentando una instancia por horas, sin compromiso a largo plazo, pero el trabajo puede ser interrumpido al llegar otro trabajo con prioridad mayor, como sucede en las insatancias puntuales.

2.2 Definición formal del problema

En este trabajo consideramos un modelo de calendarización, donde los trabajos son sometidos al sistema conforme el tiempo transcurre (en línea). El usuario, quien somete los trabajos, selecciona un nivel de servicio (SL_l) para el trabajo que está sometiendo. Por otro lado el proveedor de servicios tiene que garantizar el nivel de servicio que el cliente está demandando. El nivel de servicio seleccionado para cualquier trabajo es tomado del conjunto de niveles de servicio, $SL = [SL_1, SL_2, \dots SL_l, \dots SL_k]$, que el proveedor ofrece. Cada nivel de servicio SL_l tiene asociado un costo u_l por unidad de tiempo de ejecución y un factor de holgura f_l el cual denota la urgencia con la que el trabajo se debe ejecutar. Teniendo que si un trabajo tiene urgencia alta, el costo será mayor en relación con una urgencia baja. Así que el usuario debe pagar un costo u_l por unidad de tiempo de ejecución, siendo $u_{max} = \max_l\{u_l\}$ el máximo costo y recibe una garantía de calidad de servicio (QoS) de que su trabajo se ejecutará en un marco de tiempo delimitado por el factor de holgura asociado al nivel de servicio seleccionado.

Consideramos que el número total de trabajos sometidos al sistema es n_r. Cada trabajo J_j del conjunto de trabajos sometidos $\mathcal{J}_r = [J_1, J_2, \dots, J_j, \dots, J_{n_r}]$ es descrito por la tupla (r_j, p_j, d_j, SL_j^l): $r_j \geq 0$ denota la fecha en que el trabajo fue liberado al sistema, p_j representa las unidades de tiempo de ejecución que el trabajo requiere, $SL_j^l \in SL$ es el nivel de servicio que el usuario escogió para el trabajo y d_j es la fecha límite, la cual es calculada como se muestra en la ecuación (1) y representa la fecha máxima en la que el trabajo debe ser entregado al usuario.

$$d_j = r_j + f_j^l \cdot p_j \tag{1}$$

Como podemos ver en la fórmula (1), el factor de holgura (f_j^l) es el parámetro que determina la fecha límite del trabajo. Podemos decir que $d_{max} = max_j\{d_j\}$ denota la máxima fecha límite en el sistema. Y la ganancia que el sistema obtiene por ejecutar este trabajo es calculada con la ecuación (2).

$$g_j = u_j^l \cdot p_j \tag{2}$$

Dado que estamos trabajando con un modelo de calendarización en tiempo-real, el tiempo de procesamiento p_J es conocido en el momento r_J. Para proceder a la calendarización de un trabajo, una vez que éste ha sido sometido al sistema, el proveedor tiene que decidir, antes de que cualquier otro trabajo llegue, si va a aceptar el trabajo nuevo o no. Para poder aceptar el trabajo J_j el proveedor debe asegurar que al menos una máquina en el sistema es capaz de procesarlo sin perder su fecha límite. Consideramos que podemos interrumpir la ejecución de un trabajo, para ejecutar algún otro trabajo que sea más urgente, es por eso que al aceptar un trabajo, el sistema debe asegurarse de que puede cumplir con la fecha límite del trabajo recién aceptado y también debe asegurar que no perderá la fecha límite de ningún otro trabajo previamente aceptado.

Una vez que el sistema ha aceptado un nuevo trabajo, el calendarizador usa alguna heurística para asignarle los recursos que requiere. Finalmente el conjunto de trabajos aceptados $\mathcal{J} = [J_1, J_2, \dots, J_n]$ es un subconjunto de $\mathcal{J}_r$ y n es el número de trabajos aceptados y ejecutados exitosamente.

En cuanto a la infraestructura, nosotros consideramos m máquinas uniformes. El conjunto total de máquinas en el sistema es $M = [M_1, M_2, \dots, M_m]$. Cada máquina M_i es descrita por la tupla (s_i, eff_i) y caracterizada por el factor de velocidad de procesamiento s_i y la eficiencia energética eff_i. Al tiempo t, cuando un trabajo J es sometido al sistema, únicamente un subconjunto M^a de todas las máquinas puede aceptar el trabajo, ese conjunto de máquinas disponibles en el momento t es denotado como $M^a(t) = [M_1, M_2, \dots, M_{m^a}]$.

El tiempo de procesamiento de un trabajo j en una máquina i está dado por la ecuación (3).

$$p_j^i = p_j / s_i \tag{3}$$

2.2.1 Modelo de consumo de energía

Para modelar el consumo de energía nosotros consideramos que $P_i(t)$, potencia instantánea en watts (joules por segundo), del núcleo de una máquina M_i al tiempo t consiste de una parte constante: P^{idle}, que denota la potencia que el núcleo tiene cuando está encendido pero inactivo y una parte variable: P^{work} que depende de la carga de trabajo asignada al núcleo. La ecuación (4) muestra cómo calcular la potencia que el núcleo de una máquina M_i tiene en un determinado momento t.

$$P_i(t) = o_i(t) \cdot \left(P^{idle} + w_i(t)P^{work}\right) \tag{4}$$

Donde $o_i(t) = 1$ si el núcleo está encendido al momento t, de otra forma $o_i = 0$, y $w_i(t) = 1$ si el núcleo está trabajando al momento t, de otra forma $w_i(t) = 0$. La potencia que tiene todo el sistema al estar operando (P^{op}), está modelada por la ecuación (5), la cual es la sumatoria de la potencia que tienen los núcleos de las máquinas que forman el sistema.

$$P^{op}(t) = \sum_{i=1}^{m} P_i(t) \tag{5}$$

La energía total que consumen los núcleos mientras el sistema se encuentra operando es modelada por la ecuación (6), que es la sumatoria de potencia consumida durante la ejecución de todos los trabajos en el calendario.

$$E^{op} = \sum_{t=1}^{C_{max}} P^{op}(t), \tag{6}$$

$$E^{op} = \sum_{t=1}^{C_{max}} \sum_{i=1}^{m} o_i(t) \cdot \left(P^{idle} + w_i(t)P^{work}\right) \quad \text{con} \quad o_i(t) = \begin{cases} 1\ si\ encendida \\ 0\ si\ apagada \end{cases} \quad \text{y} \quad w_i(t) = \begin{cases} 1\ \ si\ \ trabajando \\ 0\ \ si\ \ ociosa \end{cases} \tag{7}$$

En el modelo de medición de consumo energético, nosotros únicamente consideramos el consumo energético de los núcleos de las máquinas del sistema, es por eso que a partir de aquí al hacer referencia al consumo total de energía estamos contando con que es el consumo total de los núcleos.

2.2.2 Métricas

Para poder medir el desempeño del sistema hemos establecido algunas métricas, las cuales separamos en dos grupos: métricas principales y métricas secundarias.

El objetivo de las métricas principales es medir las características más importantes del sistema y como uno de los objetivos es maximizar la ganancia para el proveedor de servicios de nube, nuestra primer métrica es el factor de competitividad (ρ) el cual fue adaptado al problema de calendarización en Grids por Schwiegelshohn et al. (2012) y representa la razón entre la ganancia generada por nuestro algoritmo y la ganancia óptima. El factor de competitvidad está definido por la ecuación (8).

$$\rho = \frac{\sum_{j=1}^{n}(u_j^l \cdot p_j)}{V^*} \leq 1 \tag{8}$$

Como podemos ver el cálculo de ρ requiere la ganacia óptima V^*, la cual en la mayoría de casos no es posible calcular, por lo que nosotros la aproximamos con una cota superior $\hat{V}^*$, expresada por la ecuación (9).

$$\hat{V}^* = min\left\{\sum_{j=1}^{n_r} p_j \cdot u_l \ , (\sum_{t=1}^{\tau} T_t \cdot m) \cdot u_{max}\right\} \tag{9}$$

Ésta cota superior es el mínimo entre: la ganancia generada al ejecutar todos los trabajos sometidos al sistema y la ganancia que se genera al ejecutar un marco de tiempo τ, sin intervalos ociosos y con el nivel de servicio más alto. En este

trabajo la evaluación del sistema se hace durante un intervalo de tiempo τ, expresado por la ecuación (10).

$$\tau = S_{segundos} \tag{10}$$

Así obtenemos la cota superior $\hat{V}^*$ para la ganancia óptima V^*. Dado que el factor de competitividad es la relación entre la ganancia obtenida con nuestro algoritmo y la ganancia óptima, usando la cota superior $\hat{V}^*$ en lugar de la ganancia óptima V^*, obtenemos la cota inferior $\check{\rho}$ para el factor de competitividad ρ, donde tenemos que $\check{\rho} \leq \rho$. Así podemos definir la cota inferior $\check{\rho}$ con la ecuación (11).

$$\check{\rho} = \frac{\sum_{j=1}^{n} p_j \cdot u_j^l}{min\left\{\sum_{j=1}^{n_r} p_j \cdot u_j^l \ , (\sum_{t=1}^{\tau} T_t \cdot m) \cdot u_{max}\right\}} \tag{11}$$

Por lo tanto el factor de competitividad que obtenemos en nuestro análisis experimental es una cota inferior al factor de competitividad. Por lo que el valor del factor de competitividad puede ser cualquier valor mayor al dato que nosotros presentamos en el análisis.

En los objetivos principales de esta investigación presentamos la eficiencia energética como uno de ellos, por lo que la segunda métrica principal es sobre el consumo de energía, dicha métrica está descrita por la ecuación (6), donde tenemos la sumatoria de la energía consumida por todo el sistema al ejecutar el calendario generado.

En varios trabajos (Quezada et al. 2010 y Tchernykh et al. 2010) para medir otros aspectos del sistema que también son importantes aunque no formen parte de los objetivos principales, lo usual es tener las métricas secundarias, las cuales están expresadas desde los puntos de vista de los diferentes personajes que interactúan en el escenario donde la problemática se desenvuelve: Sistema, proveedor de

servicios de nube y usuario de la nube. La Tabla 1 muestra las ecuaciones que definen algunas métricas secundarias junto con una descripción de cada métrica.

Tabla 1. Métricas secundarias

	Métrica	**Descripción**
1	Porcentaje de Trabajos Rechazados	$PRJ = \frac{n_r - n}{n_r} \cdot 100\%$
2	Desaceleración Promedio Acotada	$BSD = \frac{i}{n}\sum_{j=1}^{n} \frac{c_j - r_j}{\max\{10, p_j\}} \leq \max(f_l)$
3	Promedio de Interrupciones por Trabajo	$NI = \sum_{j=1}^{n} int(J_j)$
4	Eficiencia Energética en Calendarización	$SEE = \frac{W}{E^{op}}$ con $W = \frac{\sum_{j=1}^{n} p_j}{C_{max}}$
5	Ganancia Total	$G = \sum_{j=1}^{n} (u_j^l \cdot p_j)$

Desde el punto de vista del usuario si el sistema tiene un porcentaje de rechazo cercano a cero, el usuario tiene un alto nivel de confianza de que al someter su trabajo, éste será ejecutado exitosamente. Lezama et al (2012) definen el porcentaje de trabajos rechazados como la razón entre los trabajos sometidos con respecto de los trabajos aceptados y ejecutados exitosamente. Otra métrica útil para informar al usuario es la desaceleración promedio acotada (BSD), definida por Feitelson et al. (1997), la cual muestra el factor de holgura real que tienen los trabajos al momento de terminar su ejecución, como en éste trabajo aseguramos que $c_j \leq d_j$, tenemos que el valor de la desaceleración promedio siempre será a lo más igual que nuestro factor de holgura más grande.

Desde el punto de vista del sistema nos interesa medir el desempeño, en particular el número promedio de interrupciones por trabajo, es la forma en que nosotros podemos medir el exceso de trabajo generado para el sistema, aunque

en este modelo no consideramos que al interrumpir un trabajo se genera un costo, pero en un contexto más realista, el interrumpir un trabajo y delegar la ejecución a otra máquina genera un costo. La definición de esta métrica consiste en hacer el promedio de la sumatoria de las interrupciones que cada trabajo tiene. Así $int(J_j)$ es una función que nos dice el número de interrupciones que el trabajo J_j tuvo antes de terminar su ejecución.

La eficiencia energética eff_i de una máquina i está definida como el rendimiento por watt producido por la máquina dada. Esto mide la tasa de procesamiento que la máquina puede proporcionar por cada watt de potencia consumido $\sum_{j=1}^{n} p_j / E^{op}$. La medición de la eficiencia energética en calendarización se hace por medio de la métrica SEE. Ésta métrica busca encontrar la eficiencia energética obtenida al ejecutar el calendario en cuestión.

Desde el punto de vista del dueño del sistema es interesante saber cuál es la ganancia de ejecutar los trabajos aceptados. La métrica consiste en calcular la ganancia obtenida al ejecutar cada trabajo aceptado, tomando en cuenta el nivel de servicio asociado a cada trabajo, esto es importante porque un trabajo que se ejecuta con nivel de servicio alto, genera una ganancia mayor que un trabajo que tiene el mismo tiempo de ejecución y nivel de servicio más bajo.

2.3 Metodología del análisis

En este trabajo consideramos optimización bi-criterio: maximización de ganancia y minimización del consumo de energía. El problema se puede simplificar a un sólo objetivo a través de diferentes métodos de combinación (agregación) de objetivos. Existen también varias formas de modelar preferencias, por ejemplo, se les puede dar un criterio específico o asignar importancia relativa a cada criterio. Esto se hace definiendo un peso para cada criterio o clasificando y ordenando según su importancia.

Con el fin de proveer buen desempeño y minimizar el consumo de energía, realizamos algunos análisis que a continuación describimos: degradación (error relativo) de desempeño, perfil de desempeño, Pareto, y cobertura de conjunto.

2.3.1 Degradación

Queremos encontrar el algoritmo de calendarización que genere un mejor compromiso entre maximizar la ganancia y minimizar el consumo de energía. Por ello, para proveer una guía efectiva para escoger la mejor estrategia, realizamos un análisis de acuerdo con la metodología propuesta en Tsafrir et al. (2007), y usada para el problema de calendarización en grids en Ramírez-Alcaraz et al. (2011); Barrondo et al. (2012) y Tchernykh et al. (2013). En dichos trabajos introducen una solución que supone la misma importancia para cada métrica, lo cual es adecuado para nuestro análisis, puesto que nosotros consideramos la misma importancia para la maximización de ganancia y la minimización del consumo energético.

El concepto de degradación (error relativo) consiste en tomar el mejor valor obtenido por la métrica y usarlo como referencia al resto de valores obtenidos por la métrica, así para cada valor obtenido, sabemos cuántas veces es peor con respecto al mejor valor obtenido. La degradación en caso de minimización (energía) está dado por la ecuación 12 y en caso de maximización (ganancia) está dado por la ecuación (13)

$$\left(\frac{valor\ de\ la\ métrica}{mejor\ valor\ obtenido\ en\ la\ métrica} - 1\right) \tag{12}$$

$$\left(\frac{mejor\ valor\ obtenido\ en\ la\ métrica}{valor\ de\ la\ métrica} - 1\right) \tag{13}$$

Para realizar el análisis bi-criterio, primero, se evalúa el porcentaje de degradación de cada estrategia bajo cada métrica. Esto se hace relativo a la estrategia con

mejor desempeño por caso de estudio usando la ecuación 13 para el caso de maximización y la ecuación 12 para minimización. Posteriormente, se promedian estos valores y se clasifican las estrategias. La mejor estrategia es la que tenga la menor degradación y se clasifica como 1.

2.3.2 Perfiles de desempeño

Al presentar conclusiones basadas en promedios, pueden presentarse casos donde dominen pequeñas instancias del problema con alta desviación, por ejemplo, al tener el 10 soluciones que son 2 veces peor que la solución óptima y en otro caso tener 9 soluciones óptimas y una once veces peor que la óptima, al tener el promedio tenemos que ambos casos nos dan la misma solución, sin embargo la segunda opción puede ser mejor, dependiendo de lo que se busque solucionar. Para analizar estos efectos negativos de trabajar con promedios presentamos los perfiles de desempeño de las estrategias.

El perfil de desempeño $\delta(\sigma)$ es una función discreta, no-decreciente que presenta la probabilidad de que la razón de desempeño $\gamma = \frac{valor\ de\ la\ métrica}{mejor\ valor\ de\ la\ métrica}$ esté dentro de un factor σ, este concepto fue introducido por Dolan et al. (2002).

La ecuación 14 muestra como calculamos el perfil de desempeño de cada métrica, teniendo que $resultado$ es la degradación de cada solución encontrada y σ es el factor de degradación que evaluamos en cada caso, para saber qué porcentaje de las soluciones están por debajo de dicho factor. Una vez que tenemos el número de resultados que están por debajo del factor σ evaluado, dividimos entre el número total de experimentos $n_{experimentos}$.

$$\delta_{métrica} = \frac{|\{resultado : resultado \leq \sigma\}|}{n_{experimentos}} \tag{14}$$

Ya que tenemos el perfil de desempeño de cada métrica usamos la ecuación 15 para promediar los perfiles de desempeño de las dos métricas que evaluamos.

$$\bar{\delta} = \frac{\delta_{energia} + \delta_{ganancia}}{2} \quad (15)$$

La mejor estrategia tendrá un alto porcentaje de soluciones con baja degradación en cada una de las métricas, así como en el promedio de ambas.

2.3.3 Pareto

También queremos analizar y presentar la optimización de dos objetivos donde la solución no se limita a una única solución, sino a un conjunto de soluciones, llamado el conjunto óptimo de Pareto, una vez que tenemos dicho conjunto, utilizamos una métrica “Conjunto Cobertura” para evaluar el desempeño de nuestras estrategias, así podemos compararlas en términos de dominancia de Pareto.

Frente de Pareto: Contrario a la técnica de promedio de degradación, la optimización multi-objetivo no está restringida a encontrar una única solución para un problema dado, sino que se obtiene un conjunto de soluciones, este conjunto es llamado conjunto óptimo de Pareto. En nuestro caso, una solución puede representar el mejor caso en relación con el consumo de energía, mientras que otra solución puede ser la solución en relación a la máxima ganancia generada. La meta de la toma de decisiones es elegir la solución más adecuada del frente de Pareto. Nuestro objetivo es obtener un conjunto de soluciones que representen una buena aproximación al frente de Pareto. Hay dos características importantes en una buena técnica para encontrar soluciones: convergencia hacia el frente de Pareto y diversidad, esto para muestrear el frente tanto como sea posible.

Cubrimiento de conjunto: Para encontrar una mejor estrategia para un problema dado, teniendo en consideración un conjunto, necesitamos comparar sus distribuciones de aproximación para el frente de Pareto. A menudo, los resultados de diferentes optimizadores multi-objetivo se comparan a través de la observación

visual de los espacios de soluciones. Un enfoque más formal y estadístico se realizó en Zitzler et al. (1999) usando una métrica llamada conjunto cobertura.

Esta métrica, cubrimiento de conjunto SC(A,B), calcula la proporción de soluciones en B que están dominadas por A.

$$SC(A,B) = \frac{|\{b \in B;\ a \in A;\ a \leq b\}|}{|B|} \quad (14.)$$

Un valor de métrica SC(A,B) = 1 significa que todas las soluciones de B son dominadas por A, si SC(A,B) = 0 significa que ningún miembro de B es dominado por A. Así, un SC(A,B) más grande nos da un mejor frente de Pareto de A con respecto a B. Dado que el operador de dominancia no es simétrico, SC(A,B) no es necesariamente igual a 1-SC(B,A), y ambos SC(A,B) y SC(B,A) tienen que ser calculados para entender cuántas soluciones de A están cubiertas por B y viceversa.

2.4 Trabajo relacionado

El cómputo en la nube es un paradigma que está teniendo un gran impacto en el mundo real, en materia de negocios, por consecuencia se está realizando una considerable cantidad de investigación en torno a sus diferentes aspectos. Para establecer el estado del arte nosotros abordamos algunos trabajos relacionados con la calendarización de trabajos en la nube, donde se regulan diferentes aspectos tales como la asignación de recursos, es el caso del modelo propuesto por Garg et al. (2011) donde muestran un modelo de IaaS, en el cual asignan trabajos de cómputo de alto desempeño (HPC) en conjunto con trabajos transaccionales, tales como servicios web. Este tipo de asignación mixta permite al sistema tomar ventaja de los diferentes requerimientos de procesamiento ya que los trabajos HPC pueden utilizar los recursos que los trabajos transaccionales desperdician. Otra ventaja es que los trabajos HPC pueden ser suspendidos temporalmente mientras los trabajos transaccionales están en su uso máximo, es

decir, se pueden manejar las prioridades de ejecución para tener un mejor uso de los recursos. Así la aceptación de trabajos HPC requiere que haya máquinas que estén subutilizando recursos al ejecutar trabajos transaccionales, el proveedor ofrece garantía de calidad de servicio a los usuarios de trabajos HPC, en la forma de tiempo máximo de entrega (fechas límite).

La idea de asegurar al cliente que tendrá su trabajo completado antes de una determinada fecha, haciendo calendarización en entornos distribuidos, fue presentada por primer vez por Isam et al. (2003). Es el primer trabajo que presenta la implementación de un control de admisión. El tipo de calendarización realizada en dicho trabajo es en tiempo real duro, es decir, los trabajos son sometidos conforme el tiempo avanza y el calendarizador no tiene conocimiento de cuándo llegará cada trabajo, adicionalmente debe cumplirse con la ejecución del trabajo antes de su fecha límite. Los trabajos son uni-procesador y se usan dos heurísticas para la selección de trabajos, *Earliest Deadline First* dando prioridad al trabajo que tiene la fecha límite más próxima al momento en que se toma la decisión y *Least Laxity First* dando prioridad a los trabajos que tienen menos tiempo de holgura entre su tiempo de procesamiento restante y su fecha límite.

Adicionalmente se ha propuesto el uso de Acuerdos de Nivel de Servicio (SLA) como una herramienta para la calendarización de trabajos en sistemas múlti-usuario tales como Grids y Nubes. Sakellariou et al. (2008) definen un modelo de SLA que incluye términos como: fecha de inicio del primer trabajo, fecha de finalización del último trabajo, tiempo de reservación para la ejecución del trabajo, número de CPUs que requiere el trabajo para su ejecución y precio a pagar por la ejecución del trabajo. Estos parámetros son utilizados para calcular la prioridad con que el trabajo se va a calendarizar. Ellos permiten que la fecha límite sea excedida, pero al suceder esto se debe pagar una penalización.

En general existen varias investigaciones experimentales en torno al uso de SLA como herramienta para dar prioridad en la calendarización (Sakellariou et al. 2008, Sommers et al. 2007), pero no hay muchos resultados teóricos donde se muestre la eficiencia en el peor de los casos usando SLAs. En la mayoría de los casos de estudio se presentan problemas de calendarización en tiempo real con fechas límite dadas. Schwiegelshohn et al. (2012) presentan un estudio teórico sobre diferentes escenarios con una máquina (SM) y múltiples máquinas paralelas (PM), con trabajos que tienen asignado un nivel de servicio (SSL) y múltiples niveles de servicio (MSL). El análisis realizado se basa en el factor de competitividad, el cual es la razón entre la ganancia obtenida por el proveedor de servicio usando el algoritmo propuesto, sobre la ganancia óptima. En dicho trabajo proporcionan las cotas para el rendimiento en el peor de los casos para cuatro políticas de aceptación voraz, las cuales son llamadas SSL-SM, SSL-PM, MSL-SM y MSL-PM, también para dos políticas de aceptación elegible, E_MSL-SM y E_MSL-PM. Las seis políticas de aceptación están basadas en la adaptación del algoritmo Earliest-Due-Date-preemptive, el cual calendariza primero al trabajo que tiene la fecha límite más próxima al momento de decisión, permitiendo suspender un trabajo y reanudarlo después, ya sea en la misma máquina o en una diferente.

Lezama et al. (2012) presentan un análisis experimental basado en el modelo de Schwiegelshohn, donde usan los cuatro algoritmos voraces para la aceptación de trabajos sometidos, SSL-SM, SSL-PM, MSL-SM y MSL-PM. Cada vez que un usuario somete un trabajo elige un nivel de servicio para el trabajo en cuestión. Cada nivel de servicio tiene asociado un precio por unidad de tiempo de ejecución y un factor de holgura, el cual representa la capacidad que el trabajo tiene para esperar en el sistema sin ser ejecutado y no exceder su fecha límite. El precio por unidad de tiempo de ejecución va decreciendo conforme el nivel de servicio avanza, es decir, el nivel de servicio 1 es el más costoso y tiene el factor de holgura más pequeño.

La calendarización es en tiempo-real, es decir, los trabajos son sometidos al sistema conforme el tiempo transcurre y el calendarizador obtiene la información del trabajo hasta que éste llega al sistema. Una vez que el trabajo es conocido por el sistema, se calcula el tiempo de respuesta para el usuario (fecha límite del trabajo) en base al factor de holgura y al tiempo de ejecución que el trabajo requiere.

En el caso de SSL-SM y SSL-PM, el sistema permite al usuario tomar el único nivel de servicio disponible, y los trabajos son asignados a una sola máquina en el caso SSL-SM y en máquinas paralelas idénticas en el caso de SSL-PM. En los escenarios con MSL-SM y MSL-PM el usuario puede escoger entre diferentes niveles de servicio para cada uno de sus trabajos. Los autores también proponen una función de precios donde hacen la asociación del factor de holgura de un SLA a un precio por unidad de tiempo de ejecución. Toman en consideración que el usuario reciba calidad de servicio y que el proveedor no tenga perdidas al ofrecer el servicio.

Debido a que el cómputo en la nube es un tema relativamente nuevo, éste ha recibido mucha atención desde varios puntos de vista y es bien sabido que el consumo de energía es muy alto debido a que la nube ejecuta enormes tareas computacionales y provee servicios heterogéneos tal como exponen Kirschnick et al. (2010). Es por eso que en este trabajo estamos interesados en hacer un uso eficiente de la energía y encontramos que hay importantes investigaciones en materia de eficiencia energética en sistemas distribuidos multi-usuario tales como Grids y Nubes computacionales.

Zhou et al. (2012) presentan un escenario de cómputo en la nube, donde hay múltiples servidores que proveen diferentes tipos de servicios. Presentan la formulación del problema de calendarización como un problema de optimización y tratan el consumo de energía como una restricción de QoS. Proponen la solución utilizando calendarización dinámica aleatoria, centrándose en el cálculo

aproximado de la demanda del usuario, para ello usan un muestreo Monte Carlo y asignan una probabilidad a cada muestra.

También hay trabajos en los que se busca minimizar el consumo de energía en ambientes distribuidos multiusuario, desde el punto de la asignación eficiente de recursos, tal como Aragón et al. (2012) presentan una investigación en la que usan algoritmos energéticamente eficientes para calendarizar los trabajos en la Grid. El modelo de eficiencia energética que presentan reduce el consumo de energía por medio del encendido y apagado de componentes que han estado ociosos por un periodo de tiempo considerable. Para la evaluación experimental usan diferentes algoritmos de asignación de recursos y en el análisis de resultados encuentran que es posible reducir el consumo de energía manteniendo la calidad del calendario. Para nuestra investigación hicimos una adaptación del modelo de energía usado por Aragón et al.(2012).

Capítulo 3

Descripción del modelo

En este capítulo presentamos la función de costos utilizada para el cálculo de las ganancias generadas para el propietario de la nube, así como la descripción del factor de competitividad, continuamos con la dinámica para realizar un calendario para los entornos jerárquicos distribuidos tipo IaaS, también describimos los algoritmos utilizados en este trabajo.

3.1 Función de costos

En el planteamiento de los objetivos principales de esta investigación estipulamos el incrementar la ganancia para el proveedor de servicios de nube. Para ello, necesitamos hacer uso de una función para asignar un precio a la unidad de tiempo de ejecución en cada nivel de servicio. Un aspecto importante a considerar es que el precio de una unidad de tiempo de ejecución debe ser de acuerdo con el nivel de servicio asociado. Así cuando la holgura de un nivel de servicio crece, el precio por ejecutar una unidad de tiempo con dicho servicio, decrece. Es por eso que en este trabajo adoptamos la función de precios propuesta por Lezama et al. (2012). Esta función asocia el factor de holgura f_l de un SL_l a un precio por unidad de tiempo de ejecución u_l, tomando en consideración: (1) que el proveedor nunca debe tener pérdida por ejecutar un trabajo y (2) el usuario obtiene una calidad de servicio de acuerdo al precio que está pagando, esto implica que la función de precios debe ser decreciente.

Para el planteamiento de la función de costos consideremos como costo: lo que le cuesta al dueño de la nube ejecutar una unidad de tiempo y precio: es lo que el usuario paga por obtener la ejecución de una unidad de tiempo.

El principio en el cual se basa la función de costos es que el proveedor de servicio tiene en funcionamiento los recursos computacionales durante el periodo de tiempo en que los n_r trabajos son sometidos. Sin embargo, el número de trabajos que el sistema acepta y ejecuta exitosamente es n, donde $n \leq n_r$. Así podemos decir que cada trabajo sometido al sistema tiene un costo α para el proveedor de servicios de la nube y lo que buscamos es que el proveedor no tenga pérdidas al prestar el servicio, se requiere que el precio por ejecutar los n trabajos cubra los costos de mantener el sistema funcionando mientras los n_r trabajos son sometidos. La ecuación 16 representa dicha función de costos. Donde tenemos que el costo α que se requiere para mantener el sistema funcionando es multiplicado por la relación entre la sumatoria de tiempos de ejecución de los trabajos sometidos, con respecto a la sumatoria de trabajos aceptados y ejecutados bajo la configuración γ.

$$costo_\gamma = \alpha \frac{\sum_{i=1}^{n_r} p_i}{\sum_{j=1}^{n} p_j} \tag{16}$$

La propuesta de Lezama et al. (2012) para la función de precios, es una función lineal basada en la función de costos, donde el costo y el precio para trabajos ejecutados con nivel de servicio SL_1 y SL_{20} son iguales y son los puntos de partida y de finalización, respectivamente, para la función de costos.

La Figura 4 muestra la función de precios, así como la función de costos, ahí podemos observar que la distancia entre los valores de la función de precios y costos es la ganancia que el proveedor obtiene. Así podemos ver que para factores de holgura $f = 1$ y $f = 20$ la ganancia es cero. Mientras que la ganancia con $f = 2$ hasta $f = 11$ es de al menos de 22 unidades monetarias.

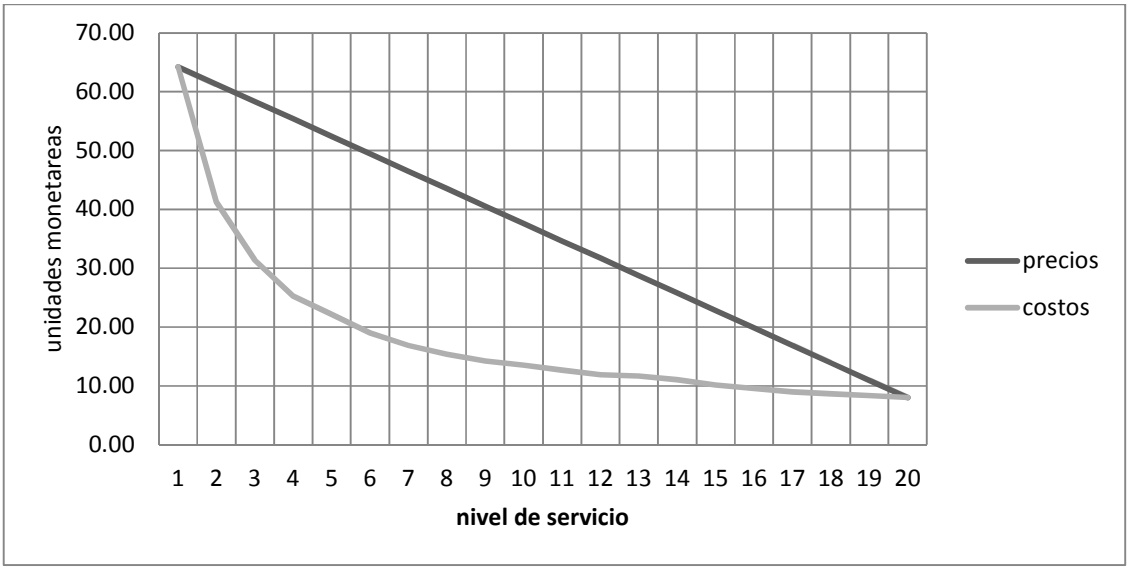


Figura 1. Función de precios y función de costos de ejecución de trabajos.

Una vez que tenemos definida las funciones de costos y de precios podemos calcular las cotas superiores para la ganancia óptima.

3.2 Descripción del factor de competitividad

El factor de competitividad es la relación entre la ganancia que generamos con nuestros algoritmos con respecto de la ganancia óptima posible, hemos mencionado que no es posible calcular la ganancia óptima, por lo que introdujimos una cota superior para aproximarla. Dicha cota $\hat{V}^*$ es calculada como el mínimo entre dos opciones de ganancia: 1) ganancia generada al ejecutar todos los trabajos sometidos y 2) cantidad de procesamiento en un marco de tiempo τ.

$$\hat{V}^* = min\{\sum_{j=1}^{n_r} p_j \cdot u_l \ , (\sum_{t=1}^{\tau} T_t \cdot m) \cdot u_{max}\} \quad (9)$$

La primer opción de aproximación de ganancia óptima es la más obvia, ya que es la ganancia que se genera al ejecutar todos los trabajos sometidos, donde cada trabajo genera una ganancia de acuerdo al nivel de servicio con el que fue

sometido, al ejecutar todos los trabajos sometidos estaríamos generando la ganancia óptima. Sin embargo cuando n_r es muy grande y la infraestructura tiene recursos limitados (1, 2, 4, etc. máquinas), no es posible ejecutar todos los trabajos sometidos debido a las restricciones de calidad de servicio, pues cada trabajo tiene una fecha límite para ser entregado. Es aquí donde requerimos otra opción para aproximar la ganancia óptima.

En nubes reales (amazon, Oracle) las ganancias son medidas en términos de marcos de tiempo (mensual, anual, etc), de ahí la propuesta para la segunda opción de aproximación donde evaluamos la ganancia que se puede generar en un marco de tiempo, si en dicho marco de tiempo ejecutamos trabajos sin tener intervalos con recursos ociosos. Pero recordemos que la ganancia no solo es sobre ejecutar más unidades de tiempo, sino ejecutar los trabajos que generan más ganancia. Tomando en cuenta estos dos factores, en la configuración de los experimentos que realizamos, fijamos el marco de tiempo τ en 7 días. Esto debido a que cada carga de trabajo utilizada, consiste en la simulación de 7 días de sumisión de trabajos. La ecuación 10 muestra la configuración de τ que utilizamos en la experimentación.

$$\tau = 7días \tag{10}$$

En esta segunda opción de aproximación buscamos ejecutar prioritariamente los trabajos sometidos con nivel de servicio alto, es decir, de los n_r trabajos sometidos buscamos ejecutar tantos trabajos como sea posible con nivel de servicio SL_1, si al ejecutar todos los trabajos con nivel de servicio SL_1 aún hay tiempo ocioso en τ, ejecutamos tantos trabajos como sea posible con SL_2 asociado y así hasta llegar a los trabajos con SL_{20}, los cuales tienen prioridad menor puesto que generan menor ganancia, esto bajo la suposición de que los tiempos de procesamiento de los trabajos en cuestión son iguales, pues no entraremos más a detalle con el problema de encontrar al trabajo que genera más ganancia real.

Otro aspecto de configuración para los experimentos es la asignación de nivel de servicio a cada trabajo sometido, para esto, en cada escenario tenemos una lista de usuarios $X = \{x_1, x_2, \dots, x_{100}\}$, cada usuario x_i tiene un nivel de servicio SL_l asociado, así como una lista de trabajos $L = \{J_{1,i}, J_{2,i}, \dots, J_{k,i}\}$ que somete. Así, cada trabajo $J_{j,i}$ que un usuario x_i somete al sistema, tiene asociado el nivel de servicio SL_l del usuario x_i.

Ahora que ya establecimos la cota $\hat{V}^*$, que aproxima la ganancia óptima y la dinámica de asignación de niveles de servicio a los trabajos, podemos iniciar con el análisis de los resultados para los diferentes casos de experimentación.

El primer experimento realizado consta de una configuración con múltiples máquinas paralelas idénticas. Donde contamos con 20 listas L_{SL}, donde los usuarios tienen asociado algún nivel de servicio del conjunto de niveles de servicio disponibles en cada escenario. Cada escenario cambia de número de niveles de servicio disponibles, en el primer escenario únicamente hay un SL disponible, en el segundo escenario hay dos SL disponibles y así hasta el último escenario, donde hay 20 niveles de servicio disponibles.

3.3 Calendarios en IaaS

La dinámica para la realización de un calendario global para la nube, consiste en dos niveles, aceptación y asignación de recursos: cuando un trabajo llega al sistema se verifica si podemos aceptarlo, esto es posible únicamente si hay al menos una máquina en el sistema capaz de ejecutar en tiempo y forma el trabajo sometido. Si existe al menos una máquina disponible para la ejecución del trabajo entonces se procede a la asignación de recursos, esto es, tenemos un subconjunto de máquinas que son capaces de ejecutar el trabajo en cuestión, y debemos decidir qué máquina lo ejecutará. La Figura 2 muestra la ilustración de la calendarización en dos niveles.

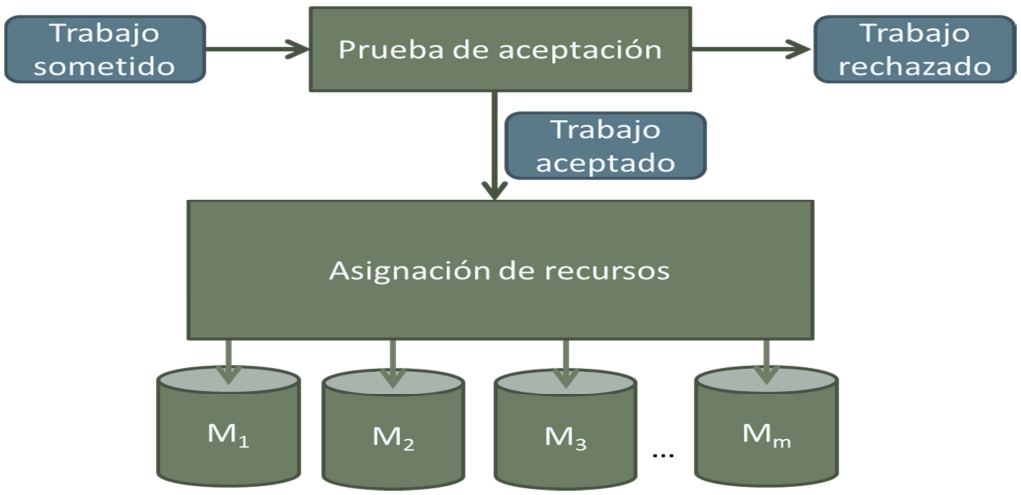


Figura 2. Diagrama de calendarización de dos niveles

3.4 Política de aceptación

La política de aceptación que estamos usando (MSL-MM) es un algoritmo voraz que está basado en el algoritmo de calendarización en-línea propuesto por Gupta et al. (2001) y funciona de la forma siguiente: tenemos el algoritmo MSL-MM sobre un conjunto de máquinas M, para cualquier trabajo J sometido al sistema, el algoritmo debe determinar si acepta o rechaza dicho trabajo J, para esto corre una prueba de disponibilidad en cada una de las máquinas en M. Cada máquina $m_i \, \epsilon \, M$ tiene una cola de trabajos en espera Q_i, en esta cola se tiene información tal como: (1) identificador del trabajo, (2) fecha límite y (3) tiempo de procesamiento restante; ya que es un algoritmo que permite suspender cualquier trabajo para darle prioridad a otro, el tiempo de ejecución restante para cada trabajo en la cola de espera no necesariamente es p_j.

La prueba de disponibilidad consiste en verificar la cola de trabajos en espera de la máquina m en cuestión y determinar si puede cumplir con la ejecución del trabajo J sin perder las fechas límite de los trabajos que ya tiene en su cola de espera. Si la máquina es capaz de ejecutar a J y cumplir con todos los trabajos de su cola de espera, entonces esa máquina se marca como disponible. Al final de la

prueba de disponibilidad tenemos un conjunto de máquinas disponibles M^a, si dicho conjunto de máquinas disponibles es diferente de conjunto vacío ($|M^a| \geq 1$), el trabajo J se acepta.

3.5 Estrategias de asignación

Una vez que un trabajo J ha sido aceptado y tenemos el conjunto M^a de máquinas disponibles para ejecutarlo, tenemos que decidir cuál máquina $m_i \in M^a$ va a ejecutar el trabajo en cuestión, para eso, en este trabajo propusimos estrategias de asignación de recursos, las cuales son descritas en la Tabla 2.

Las estrategias son caracterizadas por el nivel de información que requieren para tomar la decisión de asignación. Nosotros distinguimos dos niveles de información:

- Nivel 1: Sabemos el tiempo de ejecución del trabajo y la política de aceptación.
- Nivel 2: Además de conocer la información del nivel 1, conocemos la eficiencia energética, el consumo de energía por ejecutar el trabajo en cuestión y la velocidad con la que cada máquina ejecuta una unidad de tiempo.

Podemos ver de la Tabla 2 que comparamos tres grupos de estrategias:

- *knowledge-free*, no se tiene información de las máquinas para hacer la asignación de recursos
- *energy-aware*, conocemos el consumo energético de las máquinas y
- *speed-aware,* conocemos las velocidades de las máquinas

Tabla 2. Estrategias de asignación

Tipo	Estrategia	Nivel	Descripción
Knowledge-free	Rand	1	Asigna el trabajo j a la máquina con el número aleatorio generado con una distribución uniforme en el intervalo $[1..m]$
	FFit	1	Asigna el trabajo j a la primer máquina disponible y capaz de ejecutarlo (first fit)
	MLp	1	Asigna el trabajo j a la máquina con menos carga de trabajo al tiempo r_j: $\min_{i=1\ldots m^a}\{n_i\}$,
Energy-aware	MAX_eff	2	Asigna el trabajo j a la máquina con mayor eficiencia energética $\max_{i=1\ldots m^a}\{eff_i\}$ al tiempo r_j.
	MIN_E	2	Asigna el trabajo j a la máquina con menos consumo de energía al tiempo r_j : $\min_{i=1\ldots m^a}\{\sum_{t=1}^{r_j} P^{op}(t)\}$
	MCT_eff	2	Asigna el trabajo j a la máquina con mínimo tiempo de finalización de calendario que tenga mejor eficiencia energética $min\{\frac{c^i_{max}}{eff_i}\}$, donde $c^i_{max} = \max_{g_k=i}\{c^i_k\}$ y c^i_k es el tiempo de finalización del trabajo k en la máquina i.
Energy-speed-aware	MAX_seff	2	Asigna el trabajo j a la máquina con mejor compromiso entre rapidez y eficiencia energética: $\max_{i=1\ldots m^a}\{s_i * eff_i\}$
	MAX_s	2	Asigna el trabajo j a la máquina más rápida: $\max_{i=1\ldots m^a}\{s_i\}$

Capítulo 4

Configuración experimental

Para hacer la evaluación de los algoritmos propuestos, realizamos una serie de experimentos, los cuales fueron simulados en *teikoku Grid Scheduling* Framework (tGSF), el cual es un simulador estándar para Grids, (más detalles sobre tGSF son presentados por Carbajal et al., 2010). En el trabajo realizado por Lezama et al. (2012), tGSF fue modificado para la simulación del modelo IaaS en nubes computacionales. En este trabajo incluimos la implementación del modelo de energía y los algoritmos para la asignación de recursos.

En las próximas secciones describimos la configuración para cada uno de los experimentos realizados.

4.1 Máquinas idénticas

Los experimentos realizados en este trabajo constan de dos tipos de infraestructura: máquinas idénticas y máquinas heterogéneas. En esta sección presentamos la configuración con máquinas idénticas, es decir, todas las máquinas realizan el trabajo a la misma velocidad y todas cuentan con la misma eficiencia energética.

4.1.1 Múltiples niveles de servicio , política de aceptación voraz

Nuestro primer experimento es una réplica del experimento más realista presentado por Lezama et al. (2012). Donde para cada trabajo sometido, el usuario puede escoger entre 1 y 20 diferentes niveles de servicio. En cuestión de infraestructura tenemos diferentes cantidades de máquinas para ejecutar los trabajos.

Cada nivel de servicio tiene asociado un factor de holgura, nosotros usamos la misma dinámica usada por Lezama et al. (2012), donde los valores para el factor de holgura están directamente relacionados con el nivel de servicio: un nivel de servicio SL_i tiene un factor de holgura $f_i = i$, es decir, el SL_1 tiene $f_1 = 1$ y el nivel de servicio SL_2 tiene un factor de holgura $f_2 = 2$, y así sucesivamente hasta el SL_{20} tiene $f_{20} = 20$. En cada experimento se modifica el número de niveles de servicio que el usuario puede elegir en un SLA. La Tabla 3 muestra la distribución de SL sobre cada SLA disponible.

Tabla 3: distribución de niveles de servicio, sobre SLAs disponibles.

Número de SL disponibles en el SLA	**Factores de holgura disponibles en el SLA**
1	$SL_1 \rightarrow f_1 = 1$
2	$SL_1 \rightarrow f_1 = 1, SL_2 \rightarrow f_2 = 2$
3	$SL_1 \rightarrow f_1 = 1, SL_2 \rightarrow f_2 = 2, SL_3 \rightarrow f_3 = 3$
4	$SL_1 \rightarrow f_1 = 1, SL_2 \rightarrow f_2 = 2, SL_3 \rightarrow f_3 = 3, SL_4 \rightarrow f_4 = 4$
…	…
20	$SL_1 \rightarrow f_1 = 1, SL_2 \rightarrow f_2 = 2, SL_3 \rightarrow f_3 = 3, \ldots, SL_{19} \rightarrow f_{19} = 19, SL_{20} \rightarrow f_{20} = 20$

En términos de la infraestructura, el número de máquinas disponibles para la ejecución de los trabajos también es variable para cada experimento, tenemos incrementos en potencias de dos, desde 2^0 hasta 2^6 (1, 2, 4, 8, 16, 32, 64 y 128 para el último escenario).

Como mencionamos en la sección 5.1, la calendarización de los trabajos sometidos consta de dos niveles: prueba de aceptación y asignación de recursos. Para los experimentos realizados en este escenario usamos la política de aceptación voraz la cual definimos en la sección 5.2 y la estrategia utilizada para la asignación de recursos es asignación a la primera máquina disponible (First Fit).

4.2. Máquinas heterogéneas

La segunda parte de los experimentos la realizamos con infraestructura heterogénea, esto es, cada máquina cuenta con un factor de velocidad propio y una eficiencia energética que la caracteriza. En la siguiente sección describimos los parámetros de heterogeneidad utilizados.

4.2.1 Heterogeneidad

Para realizar la configuración heterogénea de las máquinas tomamos las infraestructuras presentadas por Aragón et al. (2012) las cuales fueron tomadas de grid reales. La Tabla 4 muestra la información de la infraestructura en la que basamos nuestra configuración.

Tabla 4: Detalles de velocidad y energía

Sitio	Procs	Consumo energético		Eficiencia energética	Velocidad
		P^{Idle}	$P^{Working}$		
DAS2—University of Amsterdam	64	17.8 W	35.35 W	1.36 MFLOPS/W	126 GFLOPS/s
DAS2—Delft University of Technology	64	17.8 W	35.35 W	1.36 MFLOPS/W	126 GFLOPS/s
DAS2—Utrecht University	64	17.8 W	35.35 W	1.36 MFLOPS/W	126 GFLOPS/s
DAS2—Leiden University	64	17.8 W	35.35 W	1.36 MFLOPS/W	126 GFLOPS/s
KTH—Swedish Royal Institute of Technology	100	17.8 W	26 W	1.75 MFLOPS/W	18.6 GFLOPS/s
DAS2—Vrije University Amsterdam	144	17.8 W	35.35 W	1.32 MFLOPS/W	230 GFLOPS/s
HPC2N—High Performance Computing Center North, Sweden	240	58.9 W	66 W	0.89 MFLOPS/W	481 GFLOPS/s
CTC—Cornell Theory Center	430	17.8 W	26 W	1.64 MFLOPS/W	88.4 GFLOPS/s
LANL—Los Alamos National Lab	1024	24.7 W	31 W	1.45 MFLOPS/W	65.4 GFLOPS/s

Normalizamos los datos de la Tabla 4 de forma que a la máquina más lenta se le asignamos factor de velocidad $s = 1$ y a la máquina con menor eficiencia energética le asignamos $eff = 1$, el resto de las máquinas tienen un factor de velocidad y una eficiencia energética relativa a la máquina más lenta y a la máquina menos eficiente energéticamente. Al analizar los resultados normalizados de la infraestructura, encontramos que la máquina más rápida es 59.78 veces más rápida que la máquina más lenta. En el caso de la eficiencia energética, la máquina más eficiente es 31.5 veces más.

Teniendo estos datos, usamos una distribución uniforme en el intervalo $[1,..,59.78]$, para la asiganación de velocidades a las máquinas en nuestra infraestructura y una distribución uniforme en el intervalo $[1,..,31.5]$ para la asiganación de eficiencia energética. Para las asignaciones del consumo energético usamos una distribución uniforme en el intervalo $[17.8, 58.9]$ watts para la asignación de potencia P^{idle} a los núcleos, y una función uniforme en el intervalo $[35.35, 66]$ para la asignación de potencia P^{work}. En el Apéndice A1 se muestra los detalles de la configuración heterogénea de la infraestructura.

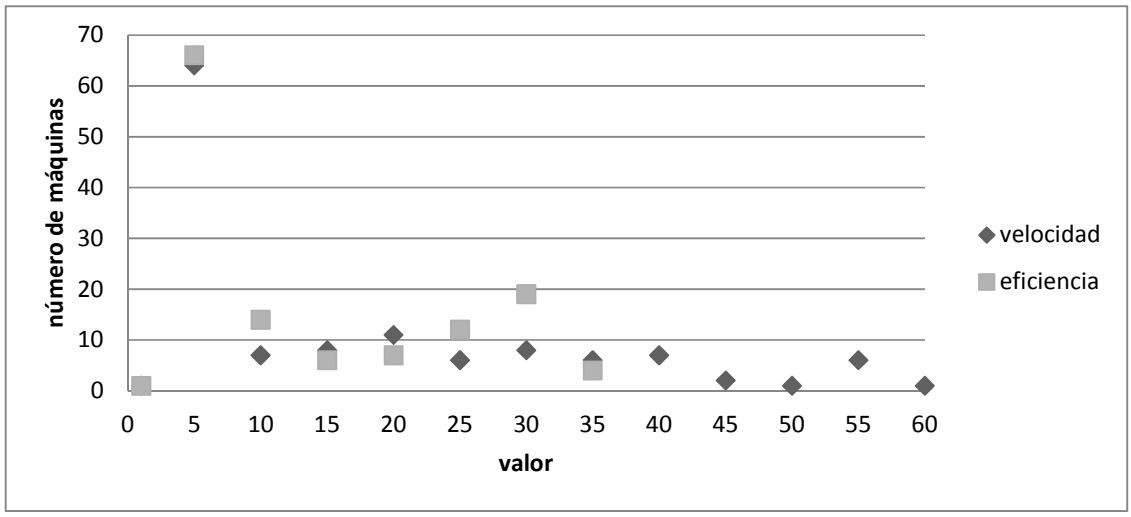


Figura 3. Histograma de velocidades y eficiencia energética de la infraestructura usada en la experimentación

4.3 Cargas de trabajo

Para tener un caso de estudio realista, en este trabajo empleamos cargas de trabajo obtenidas de bitácoras reales de Grids computacionales. Estas bitácoras nos dan una perspectiva de cómo nuestro esquema se desempeñaría con usuarios reales al implementarlo en nubes comerciales. Ya que estamos considerando que los trabajos sometidos en el modelo IaaS de la nube computacional, serán de características similares a los trabajos sometidos a los sistemas distribuidos actuales. Las bitácoras que utilizamos fueron creadas con las bitácoras tomadas de nueve Grids. La Tabla 5 muestra la lista de grids.

Tabla 5: Grids y bitácoras de trabajos sometidos

	Origen	**Bitácora**	**#Trab**	**#Usua**
1	DAS2 - University of Amsterdam	Gwa-t-1-anon_jobs-reduced.swf	1124772	333
2	DAS2 - Delft University of Technology			
3	DAS2 – Utrecht University			
4	DAS2 - Leiden University			
5	DAS2- Vrije University Amsterdam			
6	KTH - Swedish Royal Institute of Technology	KTH-SP2-1996-2.swf	28489	204
7	HPC2N - High Performance Computing Center North, Sweden	HPC2N-2002-1.1-cln.swf	527371	256
8	CTC - Cornell Theory Center	CTC-SP2-1996-2.1-cln.swf , jobs, users	79302	679
9	LANL -.Los Alamos National Lab	LANL-CM5-1994-3.1-cln.swf, jobs, users	201387	211
	Totales		**1961321**	**1683**

Utilizamos una mezcla de trabajos extraídos de estas bitácoras reales de sistemas en producción, las cuales se tomaron de *Parallel Workloads Archive* (Feitelson, D. G., 2008); posteriormente se aplicó el siguiente proceso para generar una carga más representativa:

Selección de bitácoras que tengan los siguientes datos válidos para todos los trabajos. Tiempo de llegada, número de procesadores solicitado, tiempo de ejecución y un identificador de usuario.

Normalización por zona horaria. Al crear la bitácora, se utilizó la zona horaria menor y se ajustaron todas las demás con respecto a ese valor. Para ello se modificaron los datos del encabezado de la bitácora correspondientes a la zona horaria, a la hora y al momento de inicio.

Exclusión de los primeros trabajos de cada bitácora. Esto se hace para descartar las condiciones iniciales del sistema para medir sólo el desempeño cuando éste esté estable (Frachtenberg, Feitelson, 2005).

Normalización por día de la semana. Se ajustó el inicio de trabajos válidos a las 0 horas del primer lunes y, como final, el último domingo registrado en cada una de las bitácoras.

Filtrado de trabajos no válidos. Aun cuando las bitácoras fueron previamente filtradas y adaptadas al formato estándar SWF (*Standard Workload Format*) propuesto inicialmente por Chapin *et al* (1999) y cuya versión actualizada se encuentra en Feitelson D.G. (2008), existen trabajos irrelevantes para esta investigación, por lo cual se eliminan los trabajos que tengan alguna de las siguientes características:

1. número de trabajo menor o igual a cero (job_number <= 0)
2. tiempo de llegada menor que cero (release_time < 0)
3. tiempo de ejecución menor o igual a cero (runtime <= 0)
4. número de procesadores asignados menor o igual a cero (allocated_processors <= 0)
5. tiempo estimado menor o igual a cero (requested_time <= 0)
6. Identificador de usuario menor o igual a cero (user_id <= 0)
7. Estatus fallido (status = 0)
8. Estatus fallido en la última ejecución parcial (status = 4)
9. Estatus cancelado por el usuario (status = 5)

Unión: Cuando todas las bitácoras han pasado por las operaciones previamente descritas, se procede a tomar los trabajos en orden no decreciente de su tiempo de llegada hasta completar el número de días requerido.

Paralelismo en los trabajos. Al hacer el análisis en las bitácoras, notamos que predominan los trabajos caracterizados por un bajo paralelismo, así que en este estudio nosotros consideramos que las máquinas en nuestro modelo tienen suficiente capacidad para procesarlos, así que abstraemos el paralelismo.

Usuarios. Dado que cada bitácora es independiente, los registros de usuarios también lo son; por esto se podría encontrar al usuario x en más de una de ellas, lo cual impediría identificar a cada usuario participante. Para esto, se modificaron los números de identificación de usuario por una secuencia global. Con el fin de tener una lista de usuarios manejable, nosotros redujimos el número de usuarios a 100. Así para cada escenario con los diferentes SLAs, asignamos a cada usuario un nivel de servicio disponible en el SLA, esto de forma aleatoria con una distribución uniforme.

Fechas límite: Cada trabajo pertenece a un usuario de la lista y cada usuario tiene asociado un nivel de servicio en cada escenario, así para cada trabajo nosotros calculamos su fecha límite en base al tiempo de procesamiento real y al nivel de servicio que tiene asociado el usuario que sometió el trabajo.

Capítulo 5

Análisis experimental y resultados

En este capítulo definimos los criterios para realizar el análisis de los experimentos realizados, también presentamos el análisis y los resultados obtenidos en la experimentación.

5.1 Único nivel de servicio, una máquina (SSL-SM)

El primer experimento que realizamos es una réplica de los experimentos realizados por Lezama et al. (2012). El objetivo de replicar estos experimentos es evaluar el comportamiento de la cota superior para la ganancia óptima, ya que en este trabajo se hizo un cambio de enfoque, Lezama et al. (2012) utilizan una cota dinámica dependiente de la fecha límite de los trabajos sometidos y nosotros usamos una cota fija dada por un intervalo de tiempo en el cual evaluamos el sistema.

En este experimento comparamos nuestros resultados con la cota inferior del factor de competitividad, demostrada por Schwiegelshohn et al. (2012). En la configuración cada SLA cuenta con un único nivel de servicio para elegir, por lo que todos los trabajos sometidos cuentan con el mismo nivel de servicio. En cada simulación el nivel de servicio va aumentando, así tenemos 20 simulaciones para el escenario de una única máquina (SSL-SM).

El primer análisis que presentamos es el factor de competitividad del algoritmo SSL-SM. La Figura 4 nos muestra que la ganancia que estamos generando con nuestro algoritmo es cercana a la ganancia óptima cuando usamos un único nivel de servicio y una máquina para la ejecución de la carga de trabajos. El factor de competitividad tiene valores mayores a 0.97, vemos que tiene su mínimo valor cuando tenemos nivel de servicio SL_1, lo cual es debido a que los trabajos

sometidos no tienen holgura para esperar por su ejecución y la asignación de recursos se tiene que realizar inmediatamente después de que el trabajo es sometido.

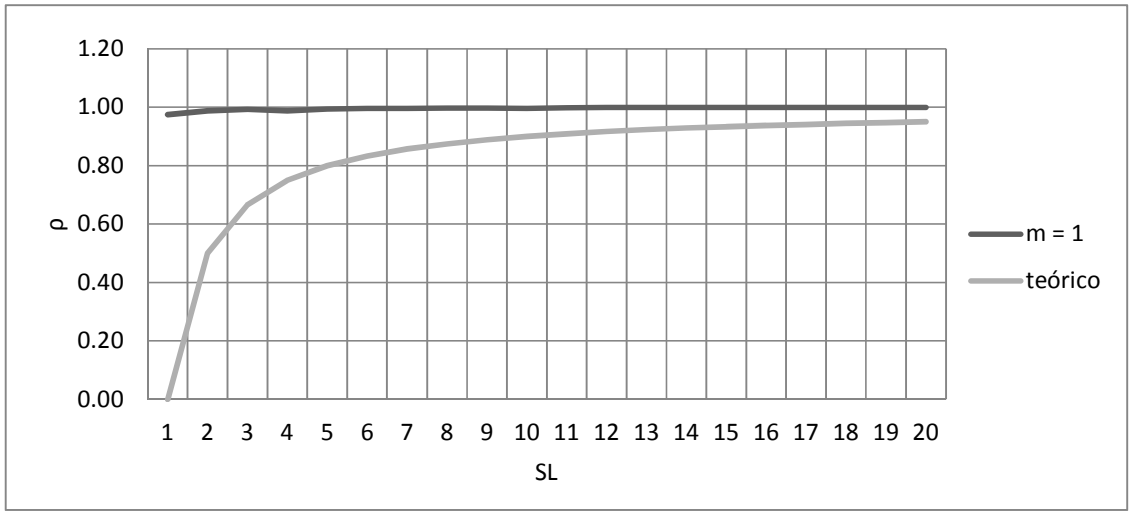


Figura 4. Factor de competitividad SSL-SM

La Figura 5 muestra la ganancia que obtenemos con el algoritmo SSL-SM, donde podemos ver que la diferencia máxima entre la ganancia óptima y la ganancia generada es de cerca de un millón de unidades, teniendo un factor de competitividad de 0.97.

Figura 5. Ganancia generada y ganancia óptima, SSL-SM

5.2 Único nivel de servicio, múltiples máquinas idénticas (SSL-MM)

En este análisis nos interesa encontrar la mejor combinación entre niveles de servicio y número de máquinas, es por eso que modificamos los modelos de simulación. En esta sección hacemos el análisis del escenario donde la infraestructura cuenta con múltiples máquinas paralelas identicas (MM), el número de máquinas se incrementa en potencia de 2. Seguimos usando 20 SLAs donde cada SLA tiene un nivel de servicio SL, tal como en la seccion anterior (SSL).

7.2.1 Factor de competitividad, SSL-MM

La figura 6 contiene los resultados del factor de competitividad obtenido en este experimento. Al analizarlo, encontramos que el factor de competitividad está por encima de los valores de 0.79 lo que nos dice que los diferentes modelos de infraestructura nos generan ganancia cercana a la ganancia máxima posible. En este punto del análisis es difícil decir qué configuración de infraestructura es mejor puesto que las 7 configuraciones presentan factor de competitividad con valores muy cercanos.

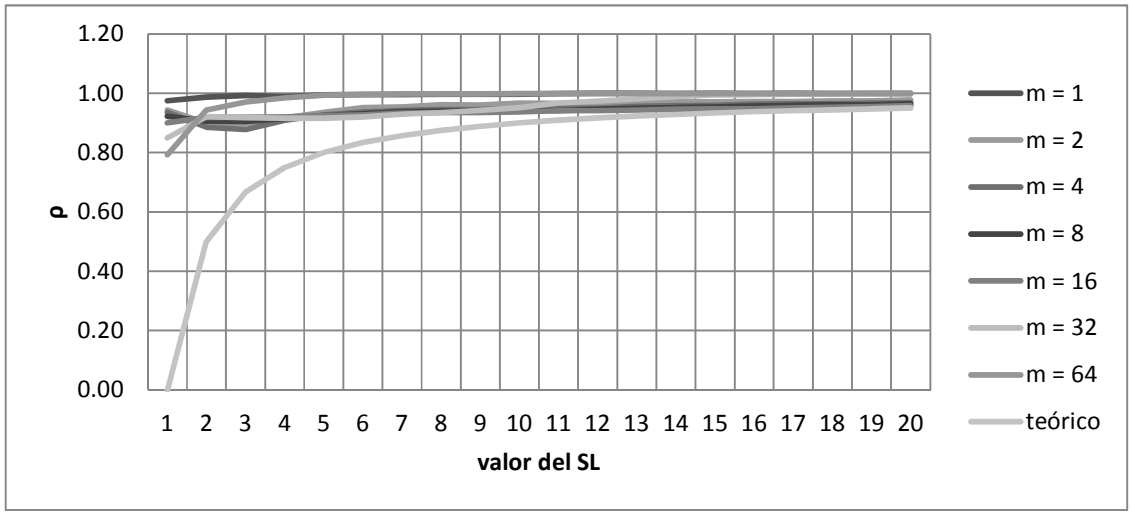


Figura 6. Factor de competitividad, SSL-MM

Para tener un mejor acercamiento a la ganancia neta que generamos en este experimento veamos la Figura 7, en la cual se muestra la ganancia total generada al incremental el factor de holgura asociado al nivel de servicio.

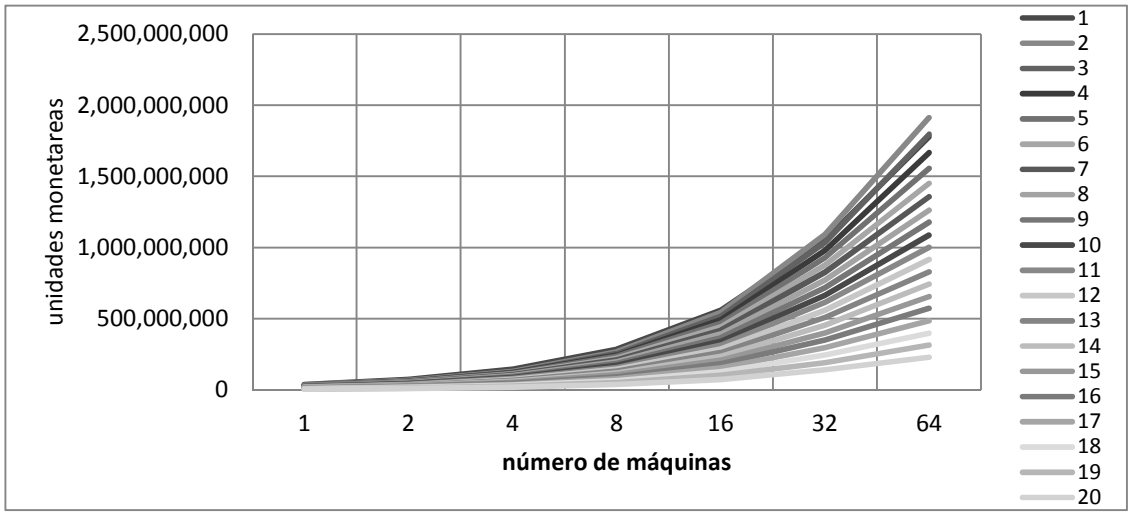


Figura 7. Ganancia total generada con SSL-MM

La mayor ganancia es generada con 64 máquinas y SL_1, mientras que al usar SL_{20}vemos que la ganancia no aumenta drásticamente con el incremento de máquinas en la infraestructura, esto es debido a que un trabajo, al ser ejecutado con SL_{20} genera 9 veces menos ganancia que al ser ejecutado con SL_1.

Un dato interesante, desde el punto de vista de las ganancias generadas, es saber si es rentable tener más máquinas en la infraestructura, ¿cómo decidir si son suficientes para dar un buen servicio al cliente, pero no demasiadas?, de manera que el proveedor no tenga recursos sub-utilizados. Para hacer dicho análisis presentamos en la Figura 8, la ganancia que en promedio genera cada máquina en las diferentes configuraciones de infraestructura. Aquí podemos ver que teniendo $m = 64$ y SL_{20} es el peor caso y se genera 25% menos ganancia que cuando $m = 1$.

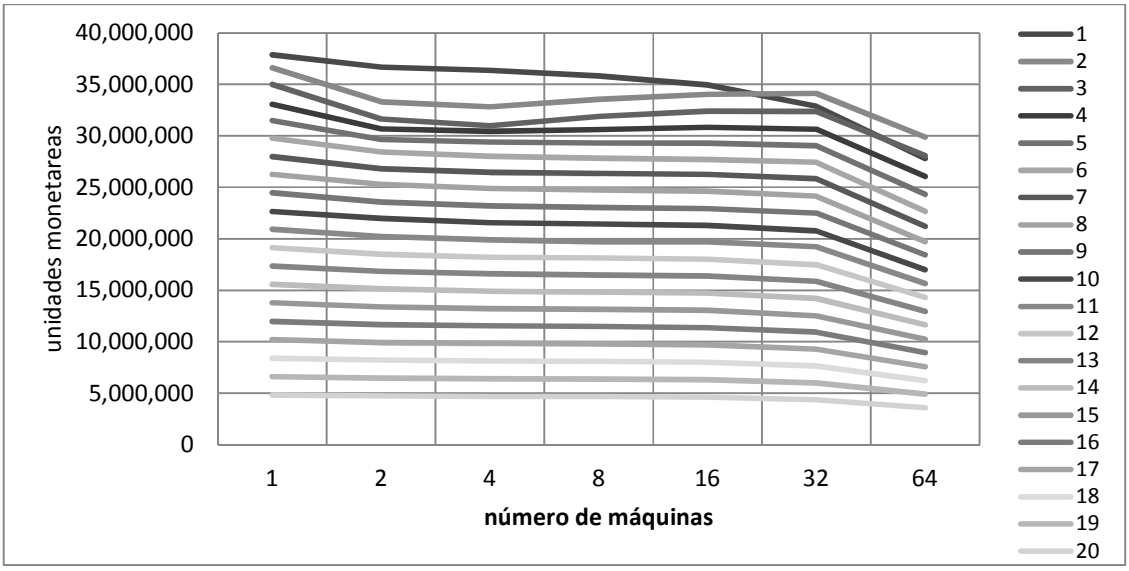


Figura 8. Ganancia promedio por máquina, usando SSL-MM

Hemos dicho que en ésta tesis tenemos dos objetivos principales: factor de competitividad, y consumo de energía así que en la siguiente sección analizamos el consumo de energía.

5.2.2 Consumo de energía, SSL-MM

En este experimento trabajamos con infraestructura homogénea, lo que significa que todas las máquinas consumen la misma cantidad de energía al ejecutar un determinado trabajo. La Figura 9 muestra la energía total que consumió cada modelo de infraestructura al ejecutar el calendario generado. Podemos ver que el crecimiento del consumo parece aumentar exponencialmente, lo cual es normal pues el número de máquinas crece exponencialmente. Para hacer la comparación objetiva del consumo de energía de los siete escenarios usamos el porcentaje de degradación. La Figura 10 muestra los resultados.

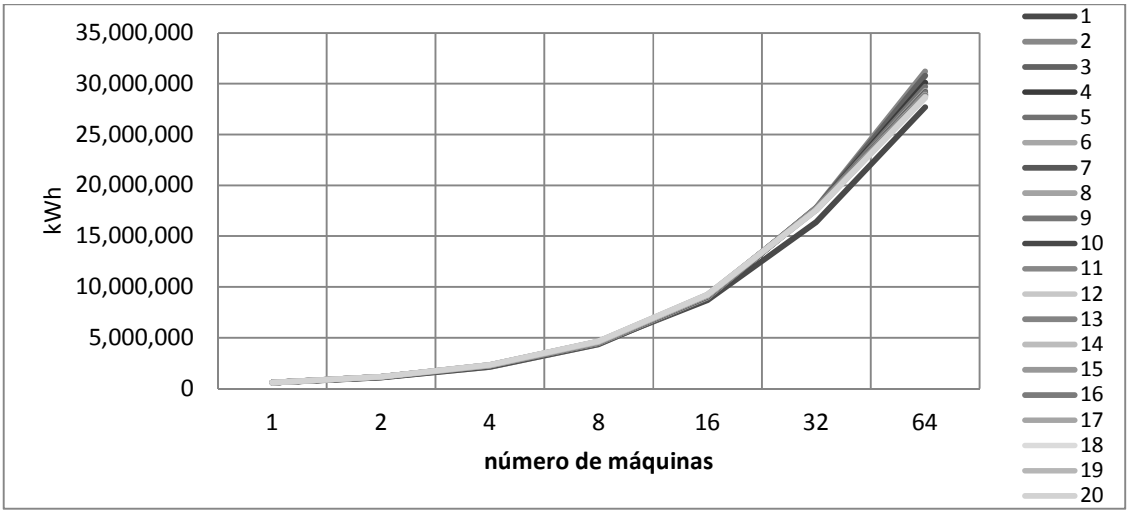


Figura 9. Consumo total de energía, SSL-MM

Para hacer la comparación objetiva del consumo de energía de los siete escenarios usamos el porcentaje de degradación. La Figura 10 muestra los resultados.

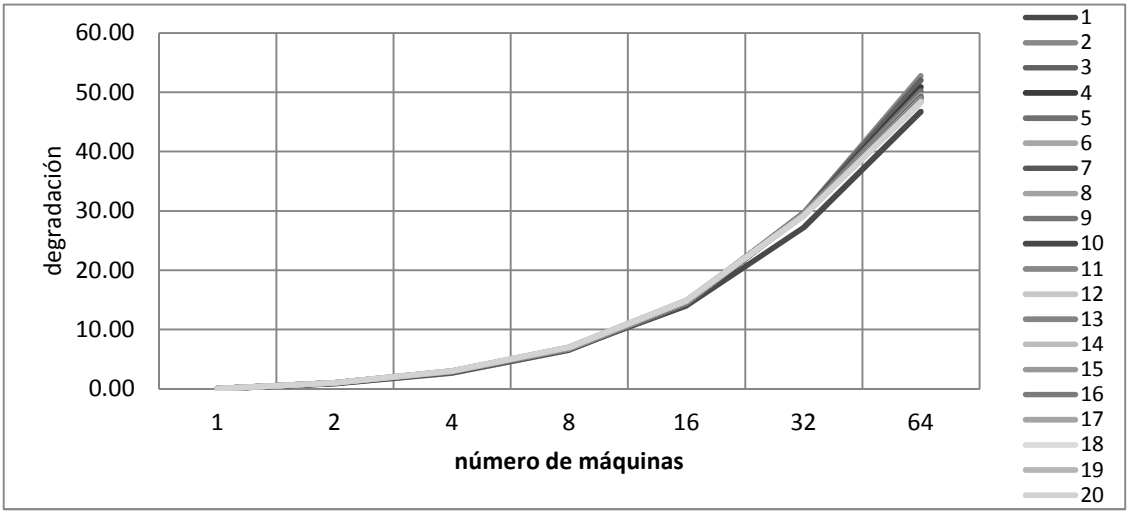


Figura10. Degradación del consumo de energía, SSL-MM

Podemos ver que al tener $m = 64$ el resultado es aproximadamente de 53 veces peor que cuando tenemos $m = 1$. Lo cual nos dice que al incrementar 64 veces el número de máquinas en la infraestructura, el consumo de energía se incrementa 53 veces, esto nos dice que en este escenario homogéneo, donde las máquinas

son paralelas e idénticas, tenemos una mejor utilización de energía al incrementar el número de máquinas.

7.3 Múltiples niveles de servicio, Múltiples máquinas idénticas, MSL-MM.

Estamos en la búsqueda de escenarios cada vez más realistas por ello en este experimento analizamos un escenario donde la infraestructura del sistema consta de múltiples máquinas paralelas idénticas (MM) y la configuración de cada SLA cuenta con múltiples niveles de servicio (MSL). Los algoritmos que utilizamos para la aceptación de trabajos es el algoritmo de aceptación voraz y la asignación de recursos se realiza a la primer máquina disponible.

7.3.1 Factor de competitividad, MSL-MM

En este experimento esperamos tener factores de competitividad menores a los obtenidos con SSL-MM, ya que al mezclar niveles de servicio en una SLA, la ganancia generada decrece. La Figura 11 muestra los resultados del factor de competitividad que obtenemos. Podemos observar que al aumentar el número de máquinas en el sistema, el factor de competitividad mejora considerablemente. Lo cual es deseable y comprensible, pues con 32 y 64 máquinas nos da mayor posibilidad de ejecutar trabajos con nivel de servicio alto, al momento que el trabajo es sometido.

Podemos ver que de acuerdo a los valores del factor de competitividad (ρ) la mejor ganancia la tenemos cuando usamos la infraestructura con 64 máquinas, pues los valores están por arriba de 0.90 y es el mejor factor a partir del SLA 3, sin embargo, eso puede decirnos que hay recursos subutilizados. Dado que desde el punto de vista de la ganancia global, el tener recursos funcionando sin ser utilizados, provoca que la ganancia decrezca, en figuras siguientes analizaremos también la ganancia total generada.

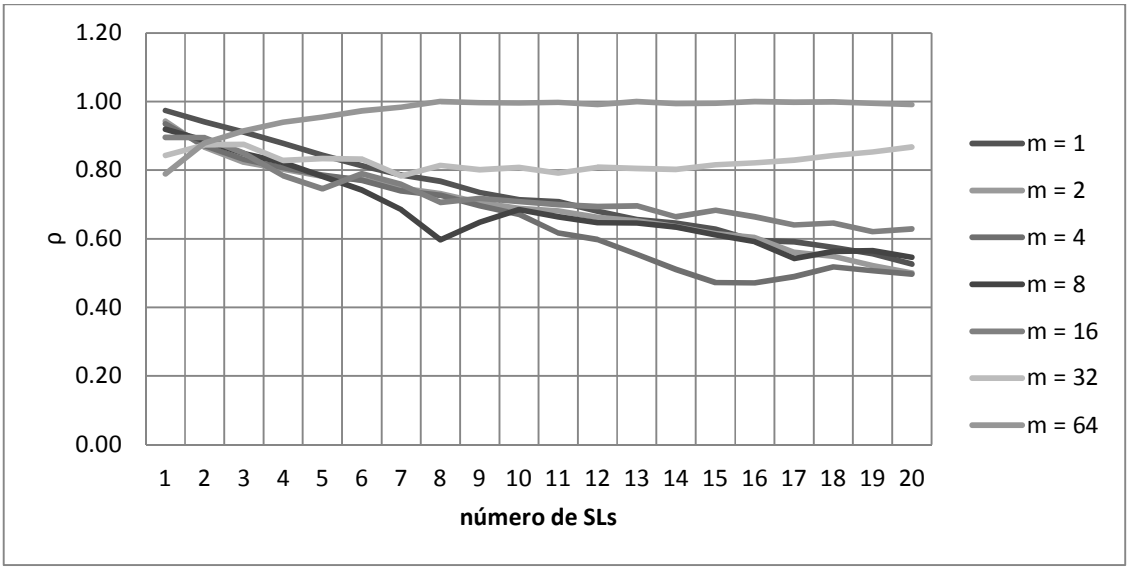


Figura 11. Factor de competitividad, MSL-MM

La Figura 12 nos muestra la ganancia que se genera con cada escenario de infraestructura, usando cada una de las SLA, donde en cada SLA se encuentra una gama de niveles de servicio.

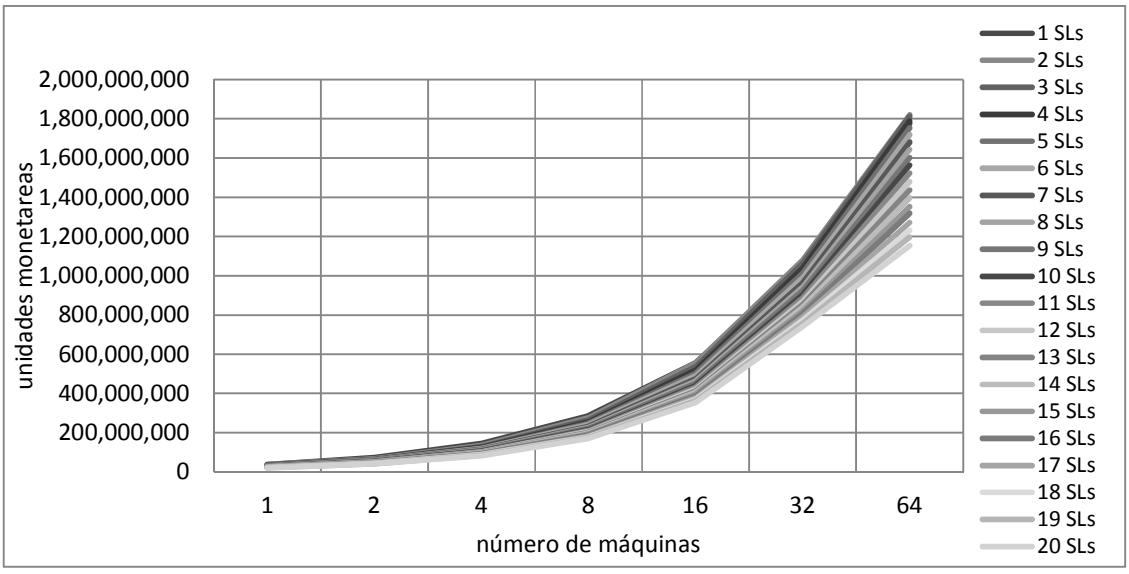


Figura 12. Ganancia total generada con MSL-MM

Aunque tenemos factores de competitividad menores que los obtenidos con SSL-MM, vemos que la ganancia total generada es mejor con MSL-MM, y es más evidente con SLAs mayores, lo cual es correcto, pues en SSL-MM el SLA 20

ejecuta todos los trabajos con SL_{20}, lo cual genera menos ganancia que ejecutar con SL de 1 a 20, como lo hacemos con MSL-MM. Ahora veamos en la Figura 13 la ganancia que genera cada máquina, en cada uno de los modelos de infraestructura. Notemos que cuando tenemos una única máquina ejecutando los trabajos, la ganancia oscila de ser la mayor ganancia al usar un único SL, hasta ser la segunda peor cuando usamos 20 SLs. Sin embargo un punto a resaltar es que al usar 64 máquinas la ganancia promedio es la peor generada en los 20 modelos de SLAs, esto debido a que ejecutamos prácticamente todos los trabajos sometidos, pero tenemos recursos ociosos.

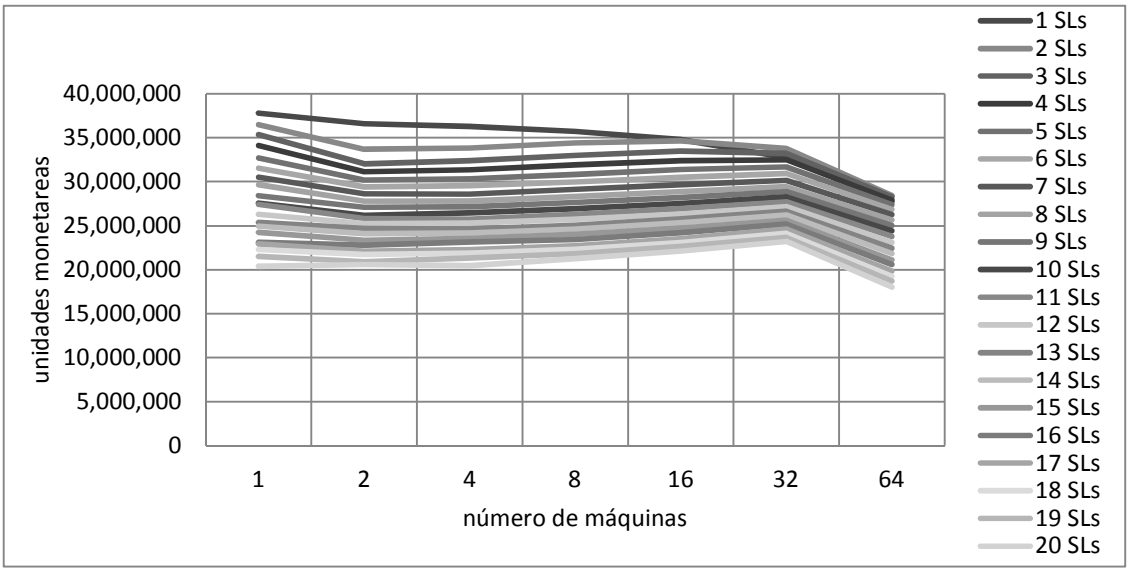


Figura 13. Ganancia promedio generada por cada máquina, MSL-MM

Algo que también podemos notar en estas gráficas de ganancia, es que conforme aumentamos el número de SLs en una SLA la ganancia decrece y esto se debe a que ejecutamos más tiempo pero a un precio menor, lo cual genera menor ganancia que ejecutar menos tiempo a costo mayor. Esto abre el análisis para encontrar el mejor compromiso entre número de SLs en un SLA y mayor ganancia.

5.3.2 Consumo de energía, MSL-MM

Ahora analicemos el consumo de energía en este segundo experimento homogéneo. La Figura 14 muestra el consumo total de energía. Si comparamos el consumo total de energía del experimento con SSL y este MSL vemos que el consumo de energía es muy parecido en ambos pues el nivel de servicio no tiene impacto en el consumo de energía.

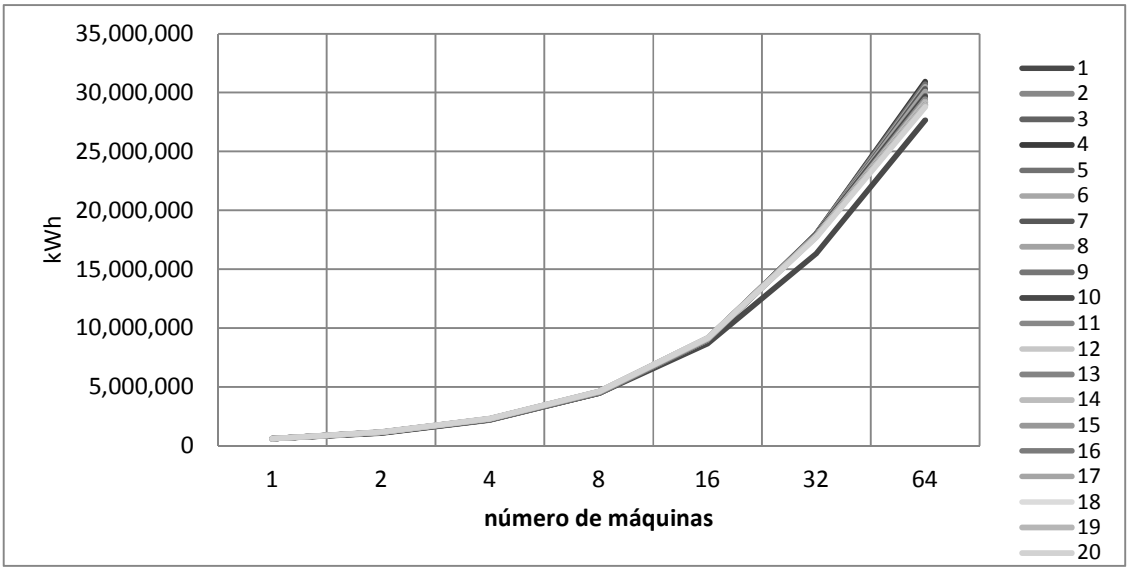


Figura 14. Energía total consumida MSL-MM

Ahora vemos en la Figura 15 que la degradación del consumo de energía se mantiene, es decir, al tener 64 máquinas el consumo de energía es 53 veces más grande que cuando tenemos una máquina en la infraestructura. Así en cuestión de energía la mejor de entre las opciones de número de máquinas en la infraestructura, es tener 64 máquinas ya que al tener 32 máquinas la degradación está entre 27% y 30%.

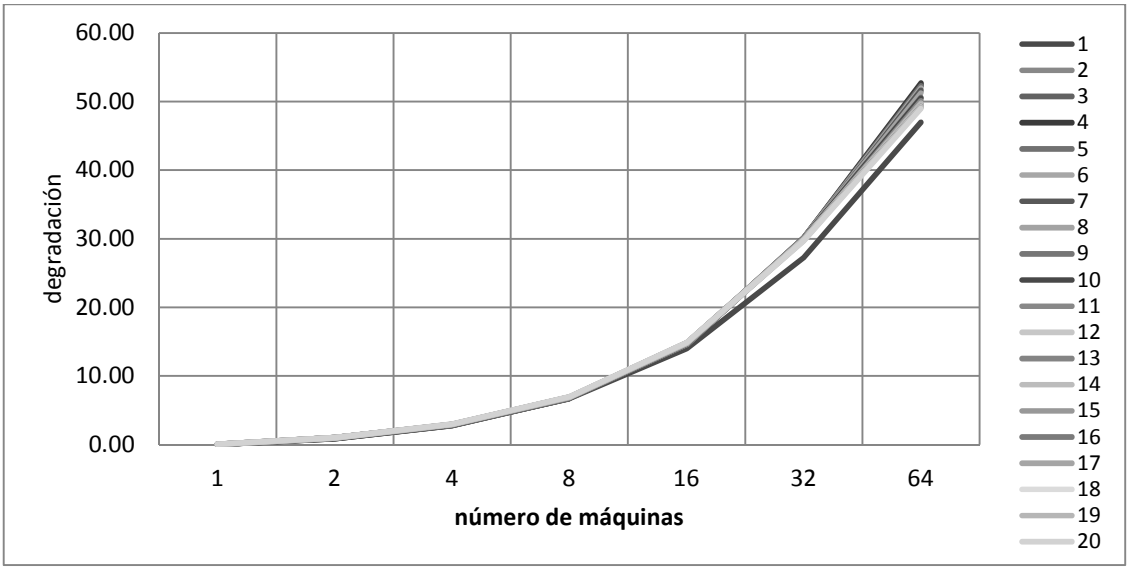


Figura 15. Degradación del consumo de energía MSL-MM

5.4 Único nivel de servicio, múltiples máquinas heterogéneas, SSL-MM-het

En este análisis nos interesa medir el consumo de energía, ahora que introdujimos las máquinas heterogéneas, también nos interesa saber el impacto que la heterogeneidad causa en el calendario. Analizaremos el factor de competitividad para saber el impacto en términos de ganancia y el consumo de energía, ya que queremos reducir el consumo de energía a través de la asignación eficiente de los recursos.

La configuración para el experimento está conformada por SLAs con un único nivel de servicio y la infraestructura cuenta con múltiples máquinas heterogéneas.

5.4.1 Factor de competitividad, SSL-MM-het

La Figura 16 muestra el factor de competitividad alcanzado para diferente número de máquinas. Ahí podemos ver que el factor de competitividad está por encima de los valores de 0.90 lo cual representa una mejoría con respecto del factor de competitividad generado con el mismo algoritmo en máquinas homogéneas. La mejor opción, en este experimento, es cuando usamos 64 máquinas, pues la

ganancia que generamos con el algoritmo SSL-MM es tan cercana a la ganancia óptima que ρ tiene valor de 1 al usar precisión de dos cifras.

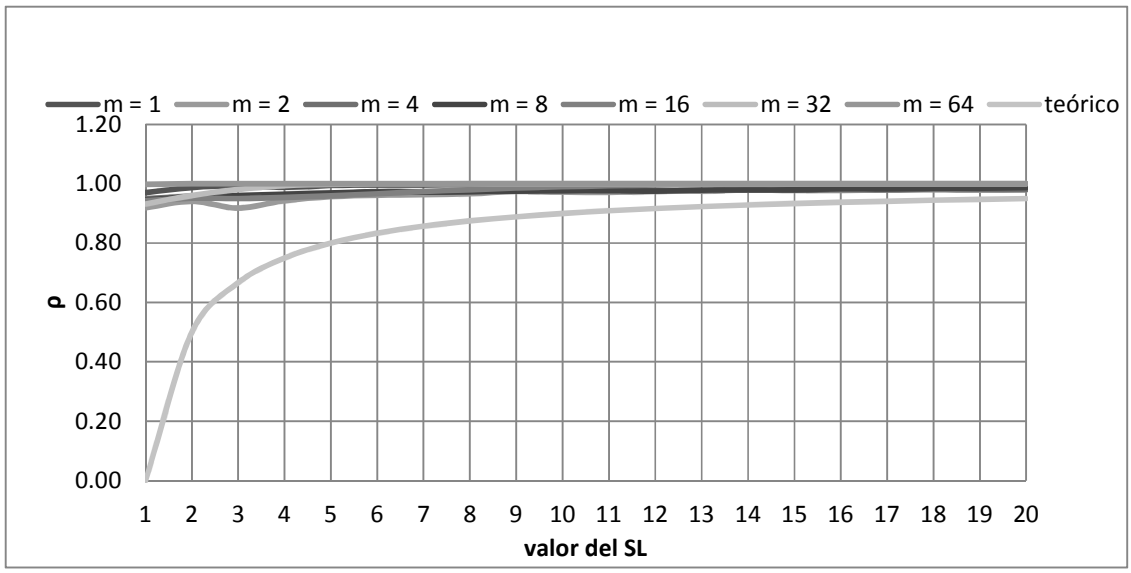


Figura 16. Factor de competitividad, SSL-MM-het

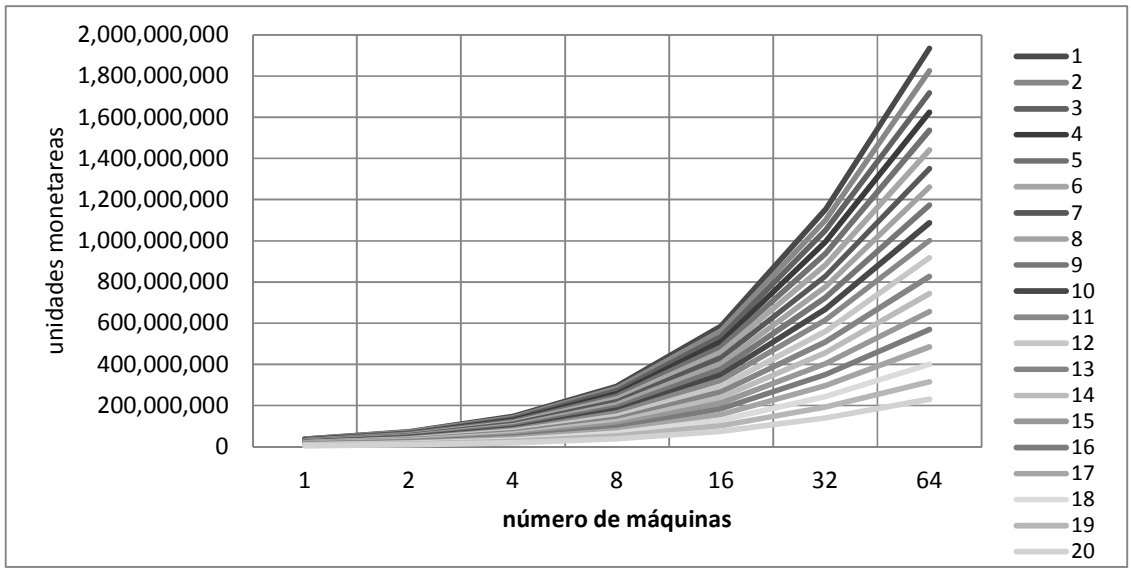


Figura 17. Ganancia total generada con SSL-MM-het

Hemos visto que el factor de competitividad alcanzado es muy cercano a 1, con todos los casos de estudio así que para tener un mejor acercamiento a la ganancia neta que generamos en este experimento veamos la Figura 17, en la cual presentamos la ganancia total generada al incrementar el factor de holgura

asociado al nivel de servicio. Esta gráfica tiene la misma tendencia que la generada en el caso homogéneo, sin embargo los valores son más grandes en el caso heterogéneo, esto es debido a que en este experimento hay máquinas que ejecutan los trabajos más rápido que otras, así podemos generar más ganancia que cuando las máquinas son idénticas.

Ahora veamos en la Figura 18, la ganancia promedio que genera cada máquina en las diferentes configuraciones de infraestructura. Aquí podemos ver que la ganancia promedio que cada máquina genera, teniendo $m = 64$, en el peor caso, está 150,000 unidades por debajo que cuando $m = 1$. Lo que representa 23% menos ganancia, lo que se traduce a recursos sub-utilizados.

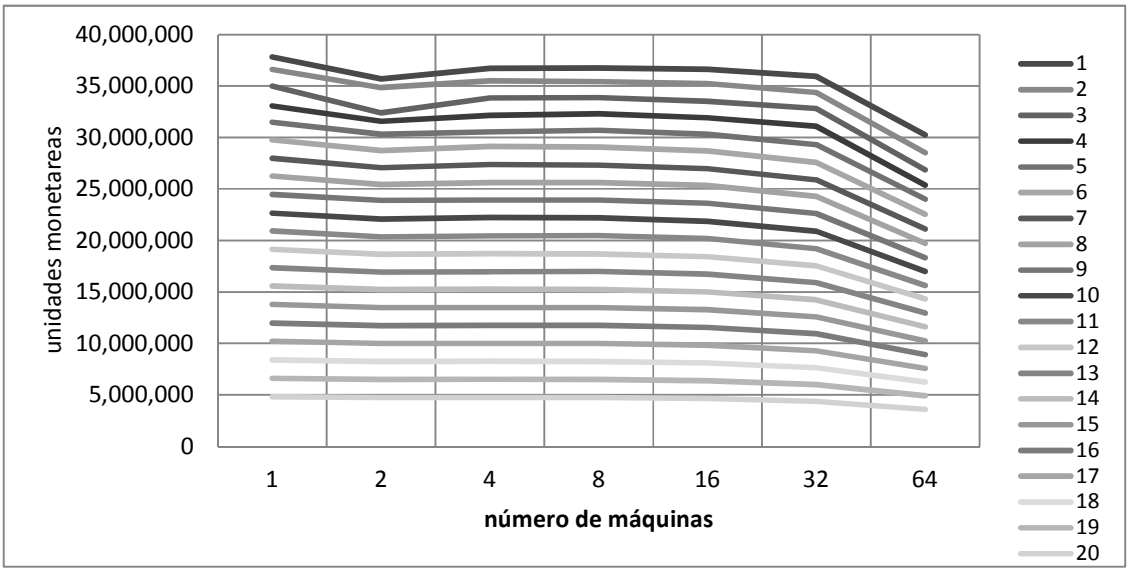


Figura 18. Ganancia promedio por máquina, con SSL-MM-het

5.4.2 Consumo de energía, SSL-MM-het

Ahora analicemos el consumo de energía, para ello tenemos las figuras 19 y 20 con el consumo total y con la degradación del consumo energético, respectivamente.

Es evidente que en torno al consumo de energía, la heterogeneidad de la infraestructura, tiene un gran impacto ya que el comportamiento es muy diferente que en el caso homogéneo

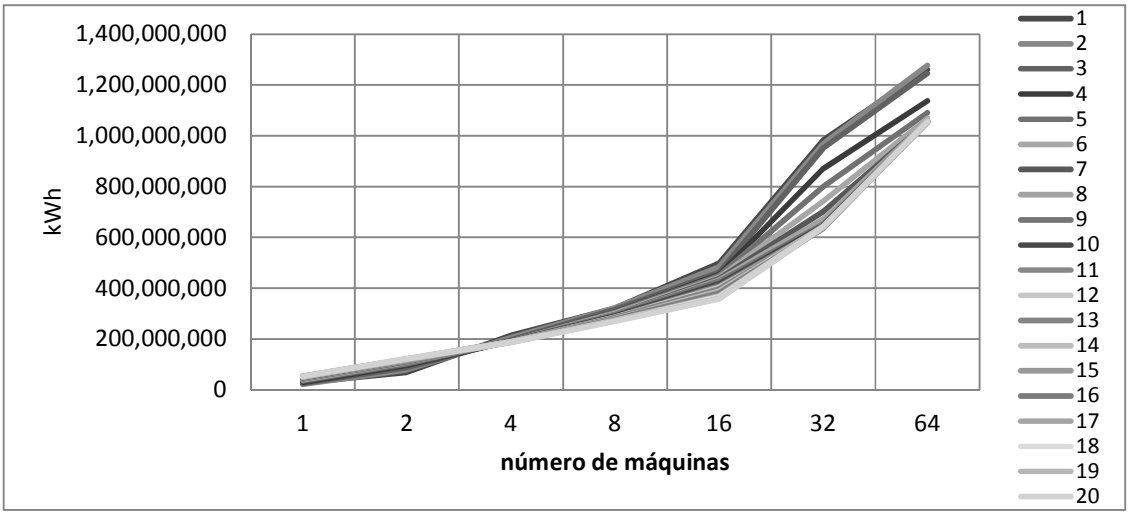


Figura 19. Consumo de energía, SSL-MM-het

Vemos que con 64 máquinas tenemos una degradación desde 110 hasta 140, es normal por la diversidad de consumo energético entre las máquinas, pues hay máquinas que consumen hasta 5.5 veces más energía que la menos energía consume.

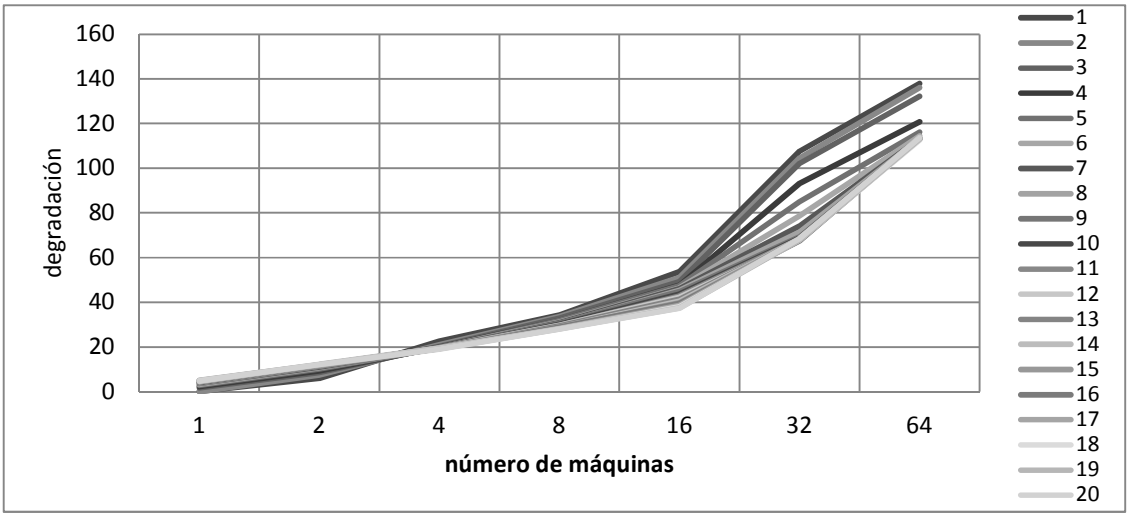


Figura 20. Degradación del consumo de energía, SSL-MM-het

5.5 Múltiples niveles de servicio, Múltiples máquinas heterogéneas MSL-MM-het.

Este experimento cuenta con la misma configuración que el experimento presentado en la sección 7.3, en cuanto a niveles de servicio se refiere, con la diferencia de la heterogeneidad en la infraestructura.

5.5.1 Factor de competitividad, MSL-MM-het

La Figura 21 muestra los resultados del factor de competitividad que obtenemos en este primer experimento con MSL-MM heterogéneo, cada línea de la gráfica muestra el resultado de las diferentes configuraciones de la infraestructura. Aquí podemos observar que al aumentar el número de máquinas en el sistema, el factor de competitividad mejora considerablemente tal como en el caso homogéneo.

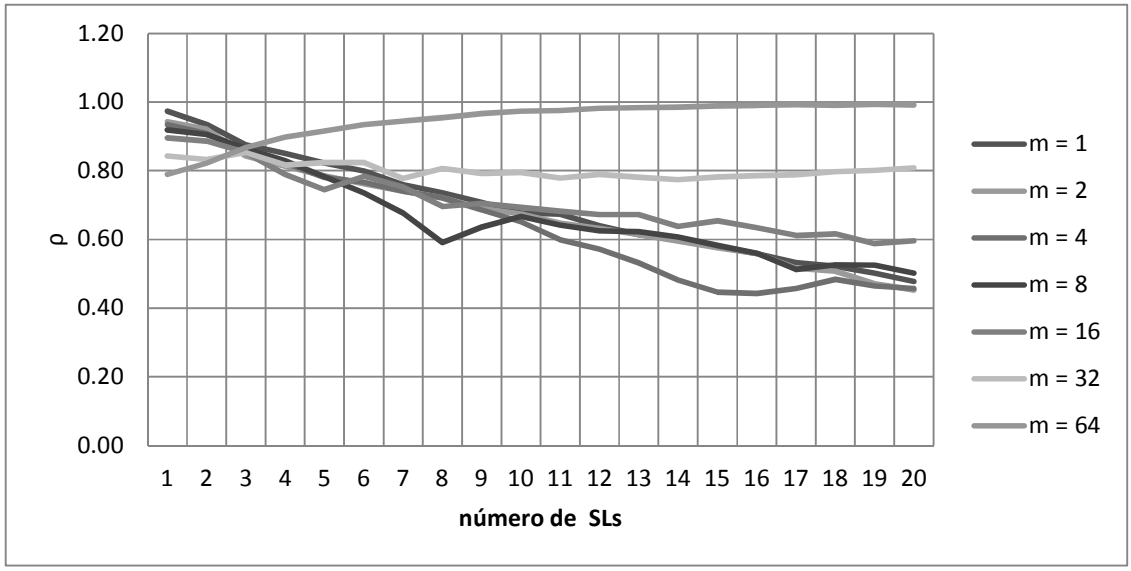


Figura 21. Factor de competitividad, MSL-MM-het

La Figura 22 nos muestra la ganancia que se genera con cada modelo de infraestructura, usando cada una de las SLA, aquí podemos ver que la tendencia de la ganancia generada continúa como en el caso homogéneo, donde las SLAs con mayor número de SLs genera menor ganancia.

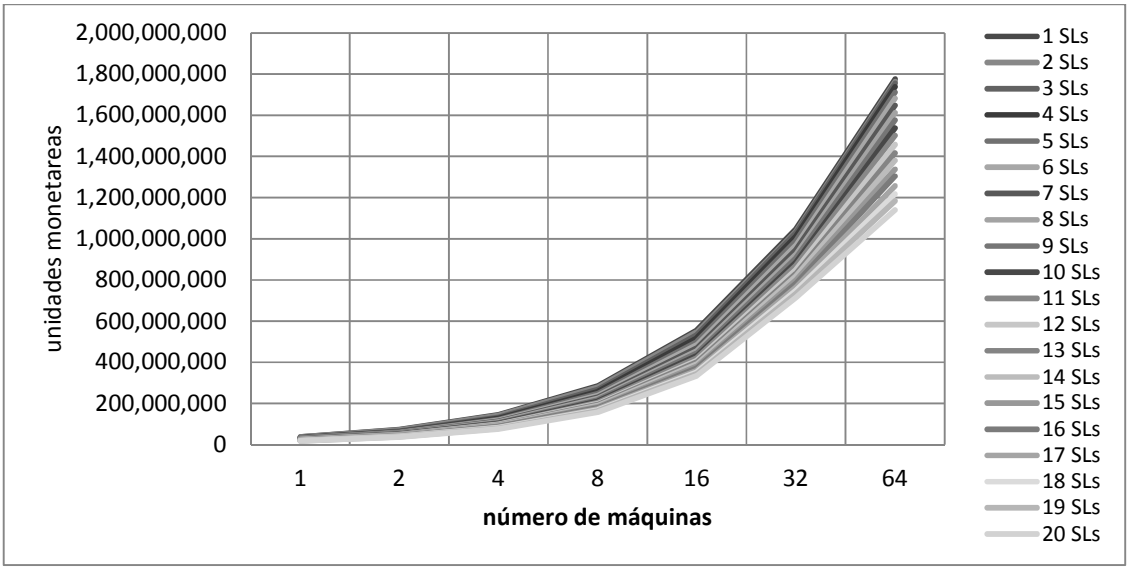


Figura 22. Ganancia total generada, MSL-MM-het

En la Figura 23 se presenta la ganancia promedio que genera cada máquina, en cada uno de los escenarios. Y vemos que con 32 y 64 máquinas la productividad por máquina decrece al igual que en el caso homogéneo.

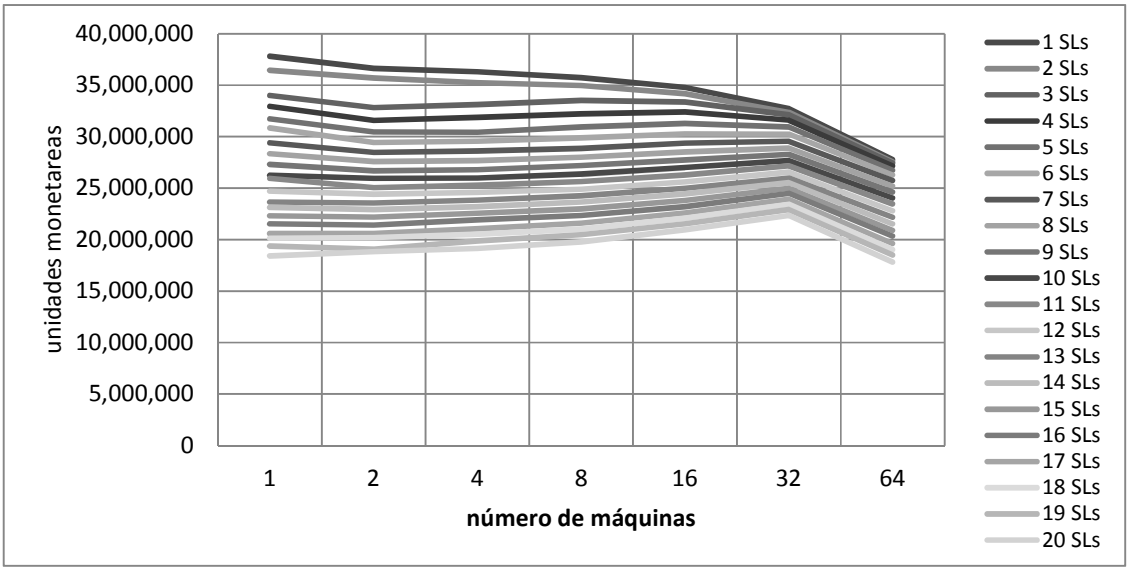


Figura 23. Ganancia promedio generada por cada máquina, MSL-MM-het

5.5.2 Consumo de energía, MSL-MM-het

En las figuras 24 y 25 tenemos el consumo total de energía y la degradación del consumo de energía, respectivamente, para este experimento con MSL-MM-het.

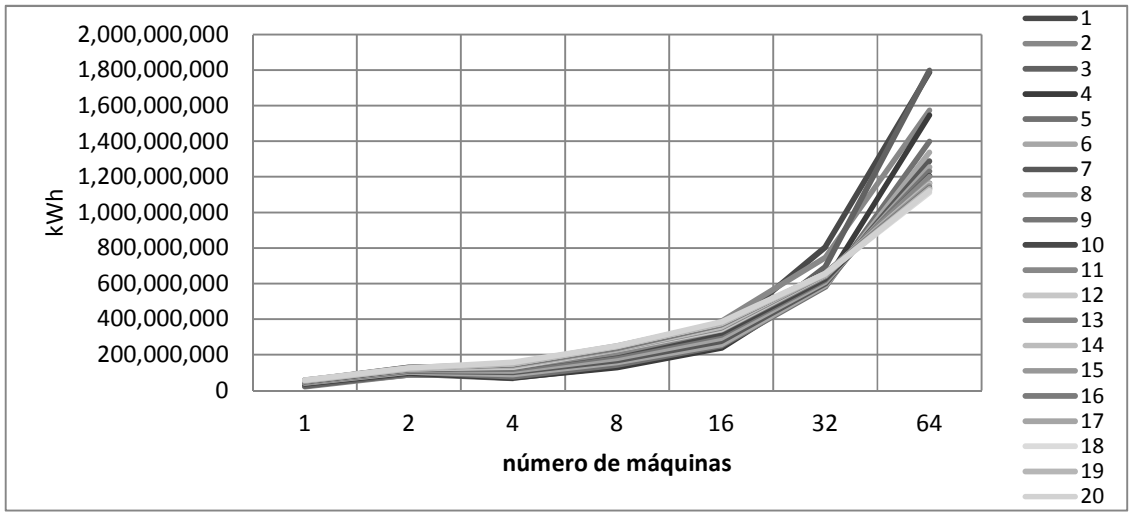


Figura 24. Energía total consumida, MSL-MM-het

En este experimento al usar 64 máquinas, tenemos degradación del consumo energético con valores de 100 a 160.

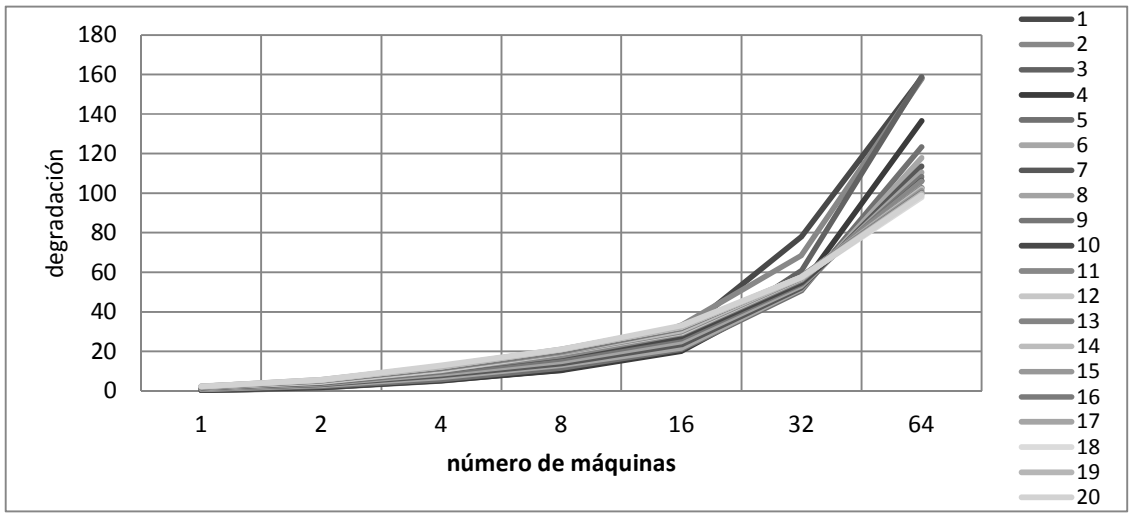


Figura 25. Degradación del consumo de energía, MSL-MM-het

En general vemos que en los experimentos con infraestructura heterogénea, tenemos un consumo energético más elevado, lo cual es normal por la diversidad de consumo energético entre las máquinas. Ahora en cuestión de tipos de SLA, tenemos que al usar SLA con SSL obtenemos ganancia por debajo que cuando usamos SLAs con MSL, esto es debido a que un SLA con MSL va a generar ganancia de acuerdo a los diferentes precios por cada SL, en tanto que SLAs con SSL generan ganancia con un costo homogéneo.

En relación al consumo energético hemos visto que usando SLAs con SSL, y la infraestructura con 64 máquinas tenemos degradación del consumo energético entre 110 a 140, en tanto que SLAs con MSL, usando también 64 máquinas, la degradación va de 100 a 160.

5.6 Análisis experimental, caso heterogéneo (speed-aware y energy-aware)

En esta sección analizamos los resultados experimentales del caso heterogéneo usando las estrategias de asignación. Buscamos encontrar la estrategia que genere mejores valores para nuestros dos principales objetivos: ganancia y consumo de energía. Mantenemos el escenario de infraestructura con múltiples máquinas heterogéneas y múltiples niveles de servicio, sin embargo en el análisis de los resultados anteriores notamos que tener 20 niveles de servicio no es adecuado para un escenario real ya que para el usuario esperar 20 veces el tiempo que su trabajo tarda en ejecutarse no es factible. En escenario realistas, una SLA no ofrece tantos niveles de servicio, así que para el análisis de ésta sección redujimos a 5 el número máximo de niveles de servicio, así tenemos 5 diferentes SLAs. Las 5 primeras SLAs, donde tenemos niveles de servicio $SL = SL_1, SL_2, SL_3, SL_4 \, y \, SL_5$.

5.6.1 Ganancia, MSL-MM-het

En esta sección vamos a analizar la ganancia obtenida al ejecutar cada calendario generado por cada una de las estrategias: tenemos 8 estrategias de asignación, 8 configuraciones de infraestructura con diferente número de máquinas y 5 SLAs.

En las Figuras 26 (a,b,c) tenemos la ganancia promedio generada por cada máquina, seguido la gráfica de ganancia generada y la tercer gráfica es la degradación de ganancia generada. Las tres gráficas contienen los datos generados en los 8 escenarios de infraestructura al usar las 8 estrategias de asignación cuando los trabajos son sometidos bajo el SLA_1.

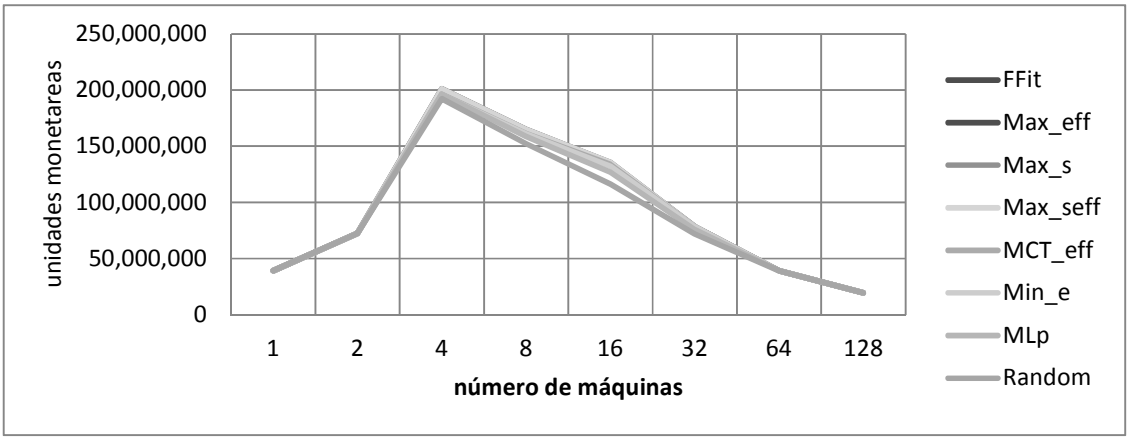


Figura 26(a). Ganancia promedio por maquina al usar SLA con un nivel de servicio

Podemos ver que en promedio la mejor ganancia la obtenemos cuando en la infraestructura tenemos únicamente 4 máquinas, después de eso la ganancia que en promedio genera cada máquina decrece. En la ganancia total que se genera con cada estrategia, vemos que usando 8,16 y 32 máquinas se ve diferencia entre la ganancia que genera cada estrategia teniendo que la asignación aleatoria es la que menos ganancia genera, cuando tenemos 64 y 128 máquinas, las estrategias pierden efecto pues ya no importa a qué máquina asignemos aceptamos casi todos los trabajos y generamos la ganancia óptima.

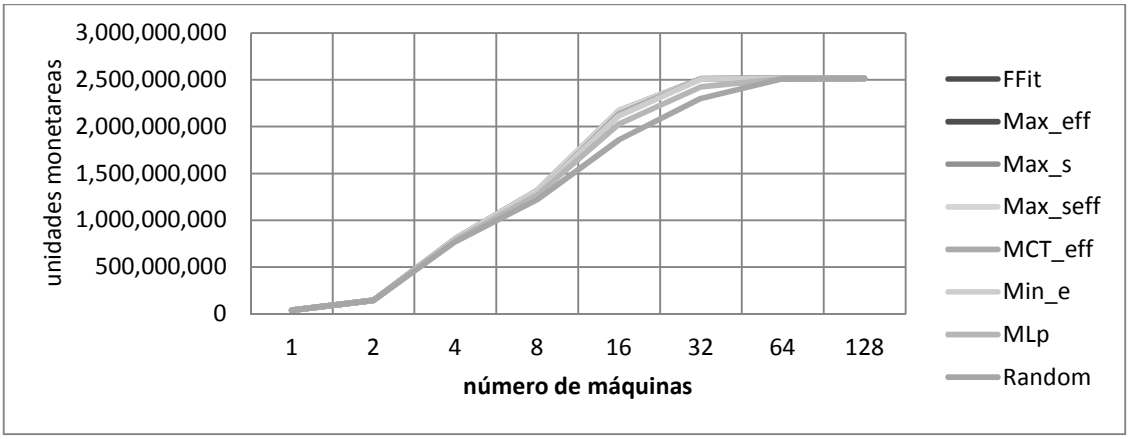


Figura 26(b). Ganancia total generada, al usar SLA con un nivel de servicio

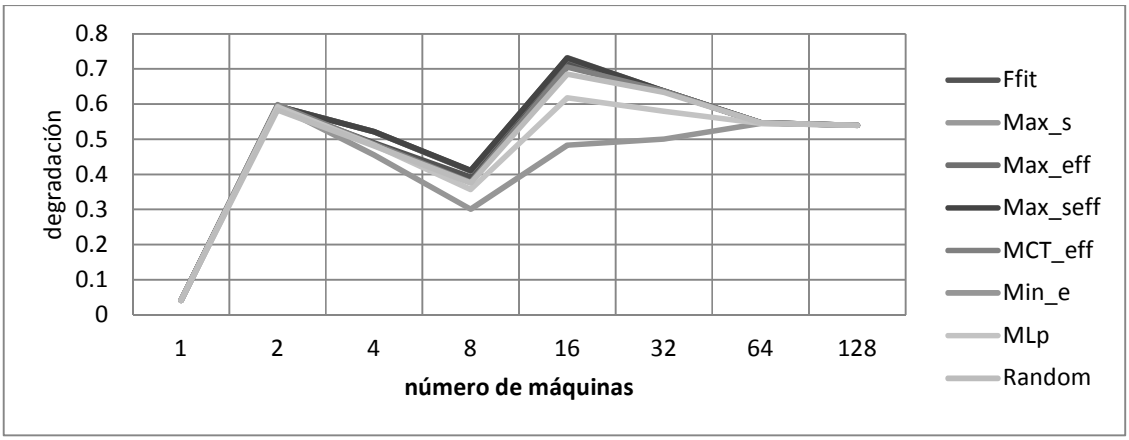


Figura 26(c). Degradación de ganancia, al usar SLA con un nivel de servicio

En el análisis de las degradaciones vemos que con 8, 16 y 32 máquinas la estrategia que asigna a la máquina con mínimo consumo de energía, al momento de la asignación tiene mejor degradación.

Ahora analicemos las Figuras 27 (a,b,c), que muestran la ganancia promedio por máquina, ganancia total y degradación de ganancia, respectivamente. Esto al usar SLA con dos niveles de servicio.

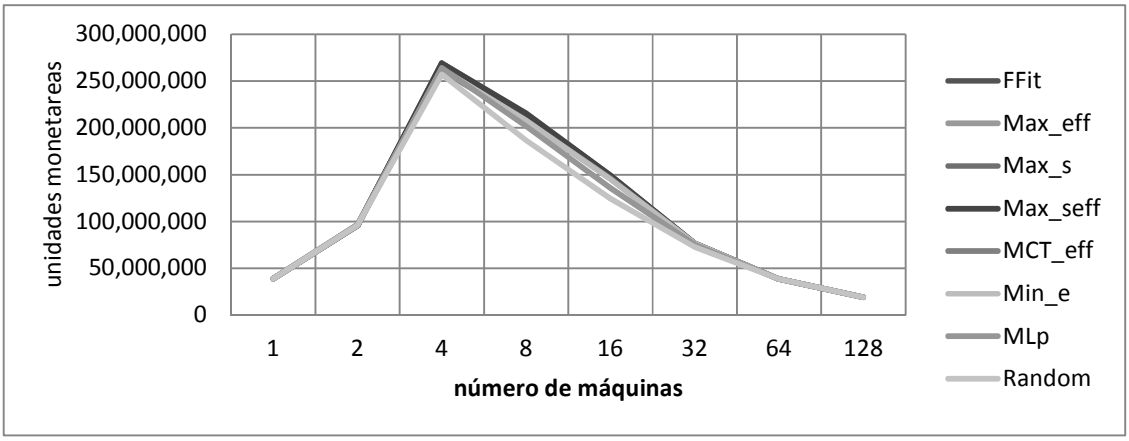


Figura 27(a). Ganancia promedio generada por máquina al usar SLA con 2 SLs

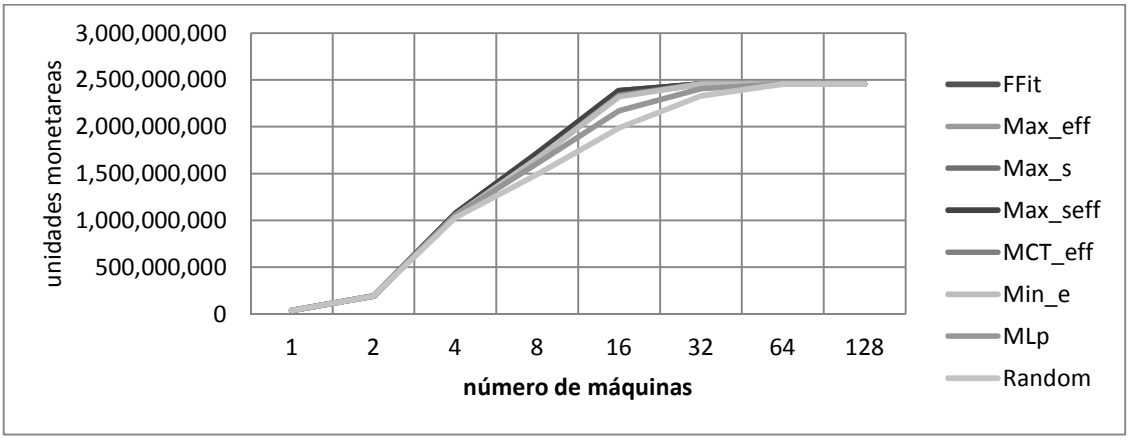


Figura 27(b). Ganancia total generada al usar SLA con 2 SLs

Podemos ver que las tendencias continúan como al usar SLA1, en promedio cada máquina genera más ganancia cuando tenemos 4 máquinas. En la ganancia total las estrategias de asignación tienen impacto únicamente en escenarios con 8,16 y 32 máquinas. La estrategia Min_e sigue manteniendo una mejor degradación.

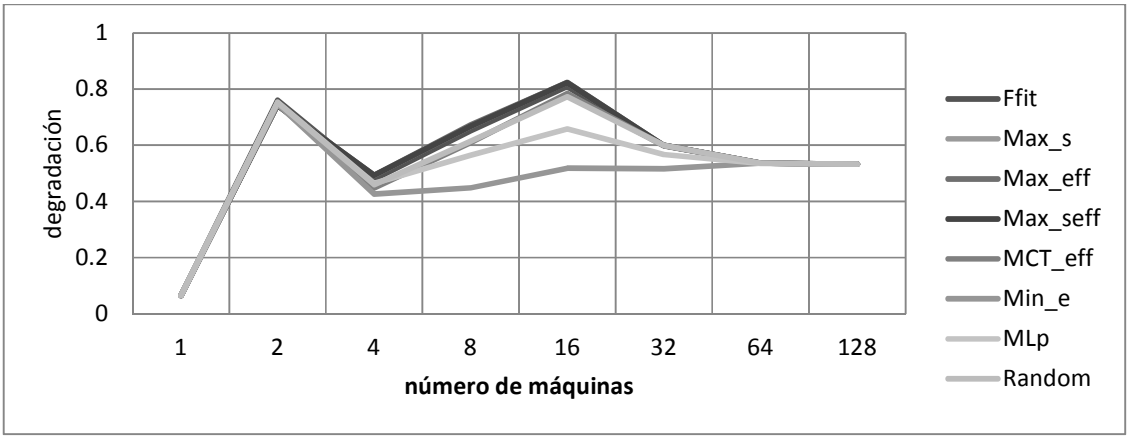


Figura 27(c). Degradación de ganancia al usar SLA con 2 SLs

Las Figuras 28 (a,b,c) muestran los resultados de los experimentos realizados con SLA con tres niveles de servicio, la primer figura muestra la ganancia que en promedio genera cada máquina, seguido de la figura que presenta la ganancia total generada y finalmente la degradación de la ganancia generada. En este experimento vemos que la estrategia Min_e sigue con la tendencia de mejor ganancia generada, aunque la ganancia decrece conforme el número de SLs crece en el SLA.

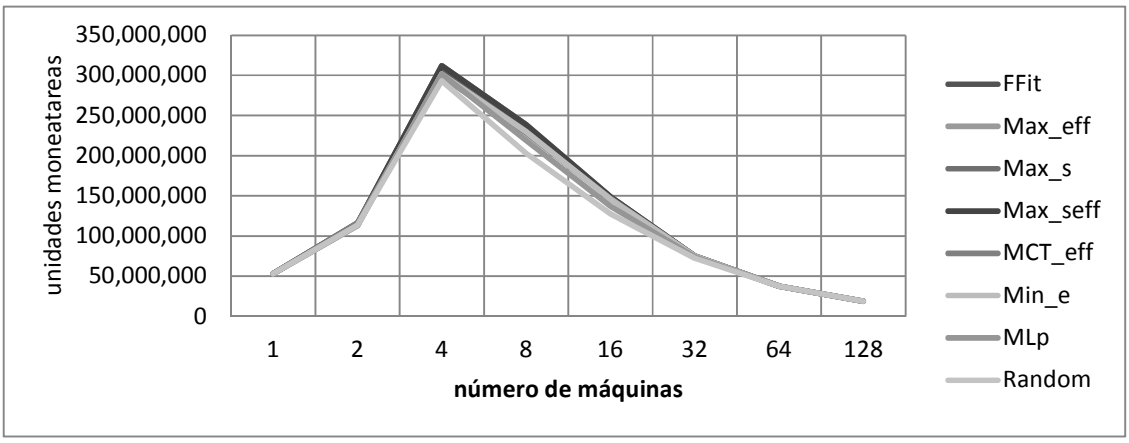


Figura 28(a). Ganancia promedio generada por máquina al usar SLA con 3 SLs

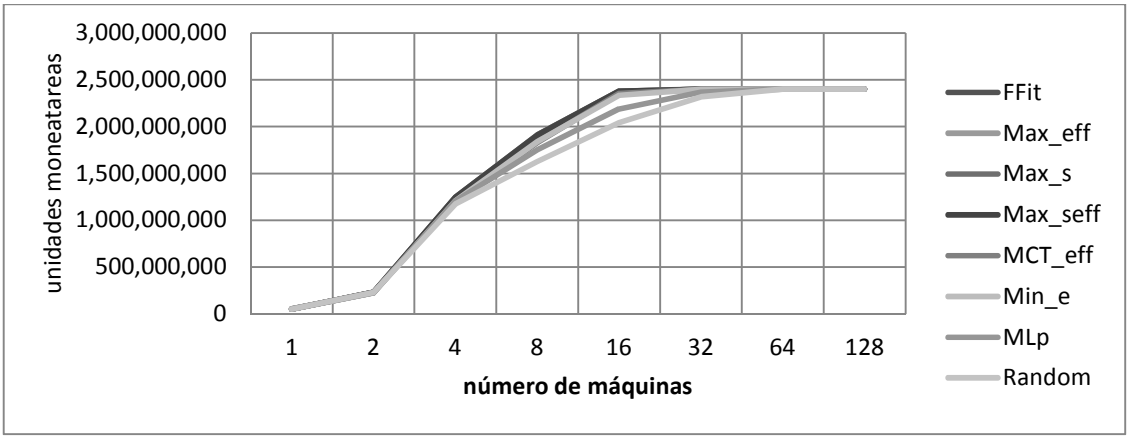


Figura 28(b). Ganancia total genrada al usar SLA con 3 SLs

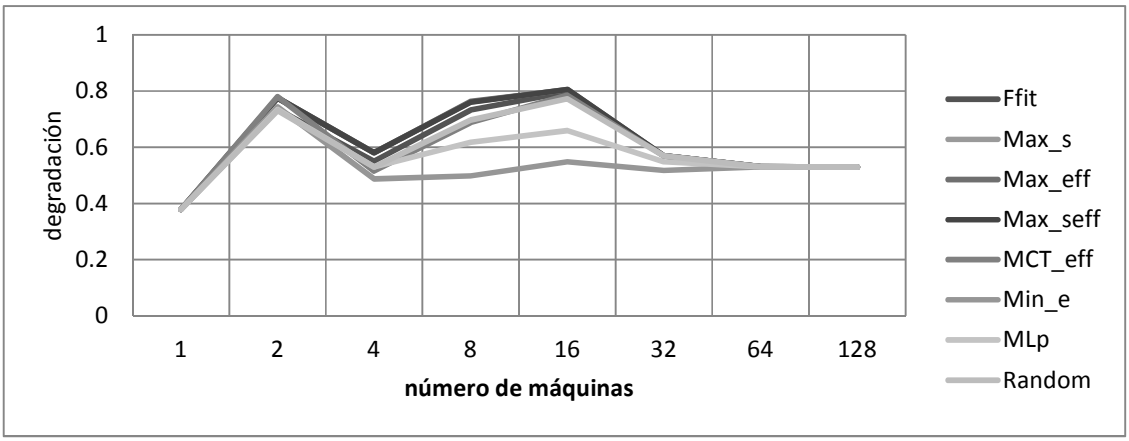


Figura 28(C). Degradación de ganancia al usar SLA con 3 SLs

El siguiente conjunto de figuras que presentamos, Figuras 29 (a, b, c), muestra la ganancia promedio que cada máquina genera, la ganancia total generada y la degradación de ganancia con cada estrategia, al ejecutar los trabajos con SLA con 4 niveles de servicio. En estas gráficas podemos ver que no hay ningún cambio drástico con respecto de los resultados que hemos obtenido al ejecutar los trabajos con SLA 1, 2 y 3, las tendencias son las mismas aunque los números son ligeramente menores en cuanto a la ganancia total generada.

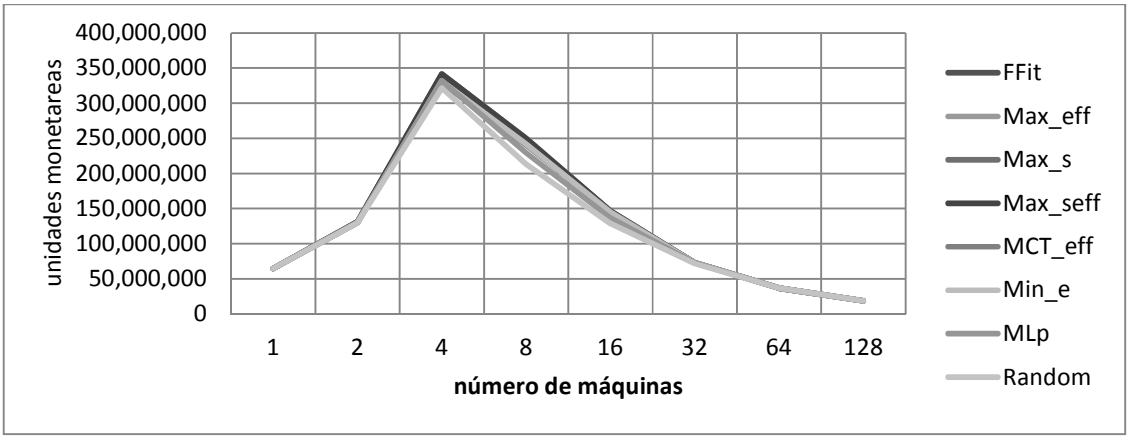


Figura 29(a). Ganancia promedio generada por máquina al usar SLA con 4 SLs

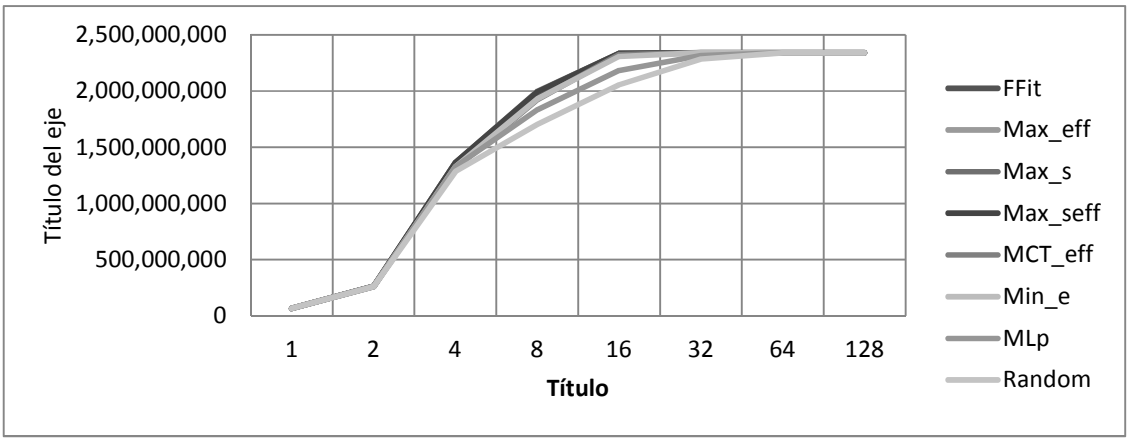


Figura 29(b). Ganancia total generada al usar SLa con 4 SLs

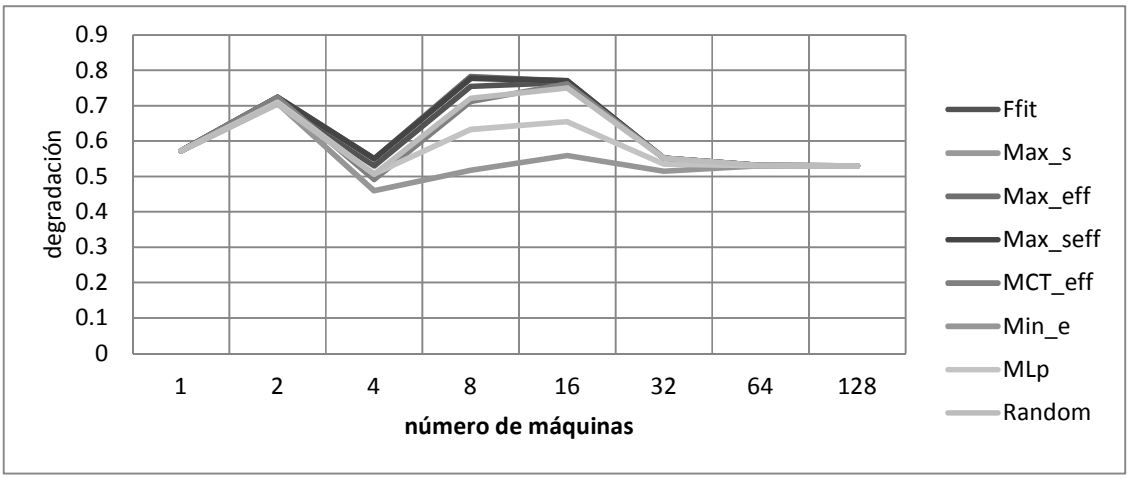


Figura 29(c). Degradación de ganancia al usar SLA con 4 SLs

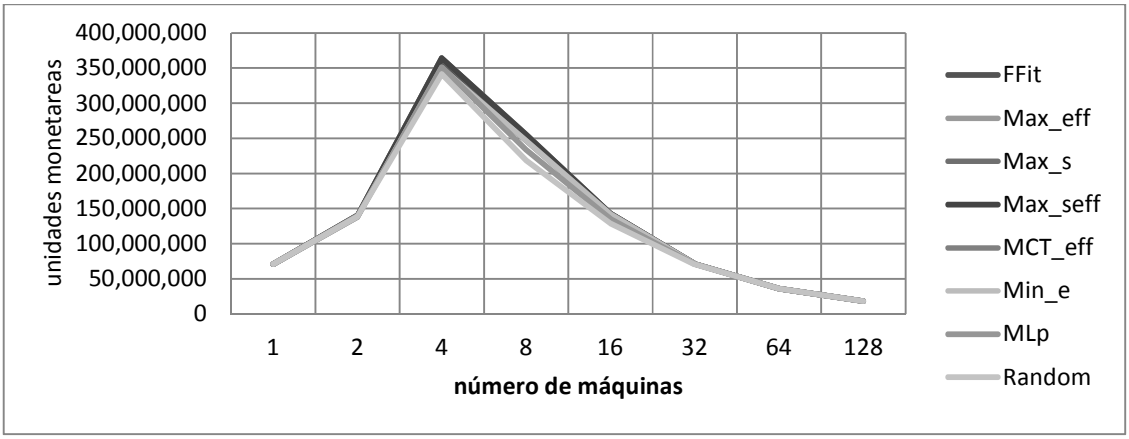


Figura 30(a). Ganancia promedio generada por máquina al usar SLA con 5 SLs

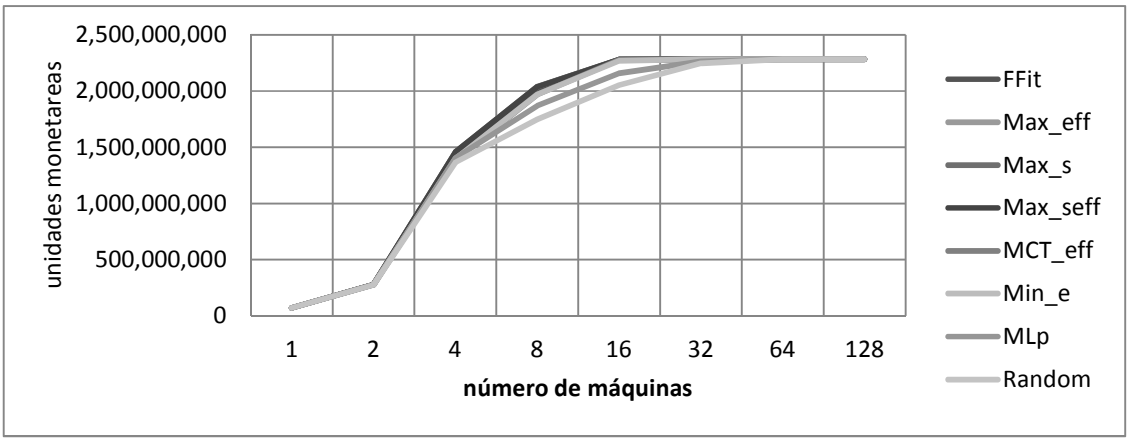


Figura 30(b). Ganancia total generada al usar SLA con 5 SLs

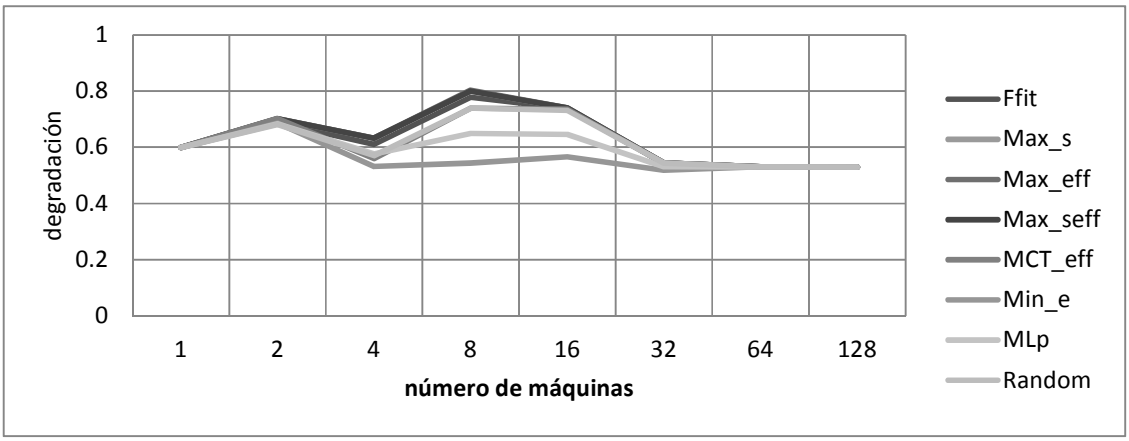


Figura 30(c). Degradación de ganancia al usar SLA con 5 SLs

Por último tenemos el conjunto de figuras 30 (a,b,c), que muestran la ganancia promedio por máquina, la ganancia total y la degradación de ganancia que

obtenemos al usar SLA con 5 SLs. En general en estos experimentos encontramos que en relación a la ganancia promedio que genera cada máquina el escenario con 4 máquinas tiene un desempeño sobresaliente, 3 veces mayor que los escenarios con 1, 64 y 128 máquinas. En materia de ganancia total generada, vemos que en los escenarios con $m = 1$, $m = 2$, $m = 64$ y $m = 128$, las estrategias de asignación no tienen gran impacto, esto debido, en el caso de $m = 1$ y $m = 2$, a que realmente no hay diversidad de máquinas para la asignación y en el caso de $m = 64$ y $m = 128$, se debe a que aceptamos prácticamente todos los trabajos sin importar la estrategia que usamos. En los escenarios con $m = 8$, $m = 16$ y $m = 32$ las estrategias que generan más ganancia son Max_eff, Max_s y Max_seff, pero la diferencia con respectro de Min_e es pequeña. En cuanto a la degradación en los escenarios con 8 y 16 máquinas es donde se puede ver diferencia notable, notemos que la constante es que la asignación a la máquina con mínimo consumo de energía (Min_e), tiene la mejor degradación en los cinco escenarios de SLAs.

5.6.2 Consumo energético

En esta sección presentamos el análisis del consumo de energía. En cada figura tenemos el resultado de los experimentos realizados en los escenarios presentados en la sección anterior. La Figura 31(a) muestra el consumo total de energía y la Figura 31(b) muestra las degradaciones del consumo de energía. En la Figura 31(a) vemos que la tendencia en todas las estrategias es incremental para cada escenario, pero del escenario de 64 máquinas al escenario de 128 máquinas el consumo es prácticamente el mismo, esto se debe a que en el escenario con 64 máquinas aceptamos prácticamente todos los trabajos sometidos, así que en el escenario con 128 máquinas, el consumo de energía no aumenta, pues las máquinas que no tienen carga de trabajo están apagadas y por lo tanto no consumen energía.

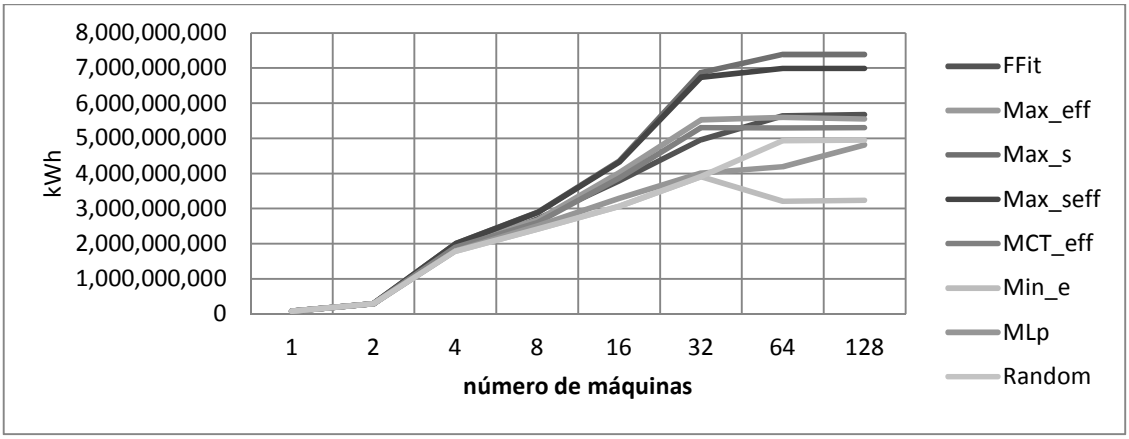


Figura 31(a). Consumo total de energía usando SLA con un nivel de servicio

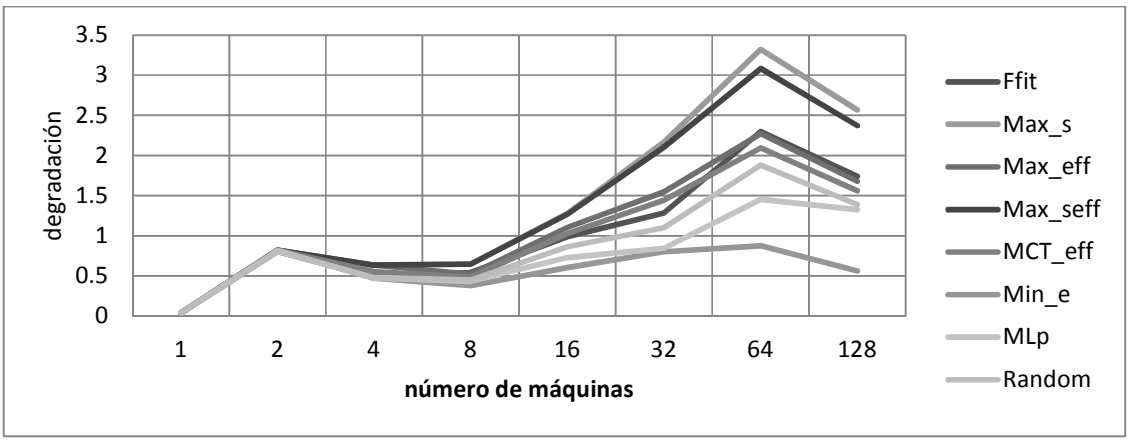


Figura 31(b). Degradación del consumo energético usando SLA con un SL

Las Figura 32(a) y 32(b) muestran el caso de experimentación al cambiar de SLA, aquí vemos los resultados con SLA 2. Las tendencias son parecidas a las obtenidas con SLA 1, en este caso tenemos consumo de energía ligeramente más elevado por lo tanto degradaciones ligeramente mayores que en el caso con SLA 1, esto debido a que procesamos más tiempo al tener dos niveles de servicio, en comparación al tiempo de procesamiento con un nivel de servicio.

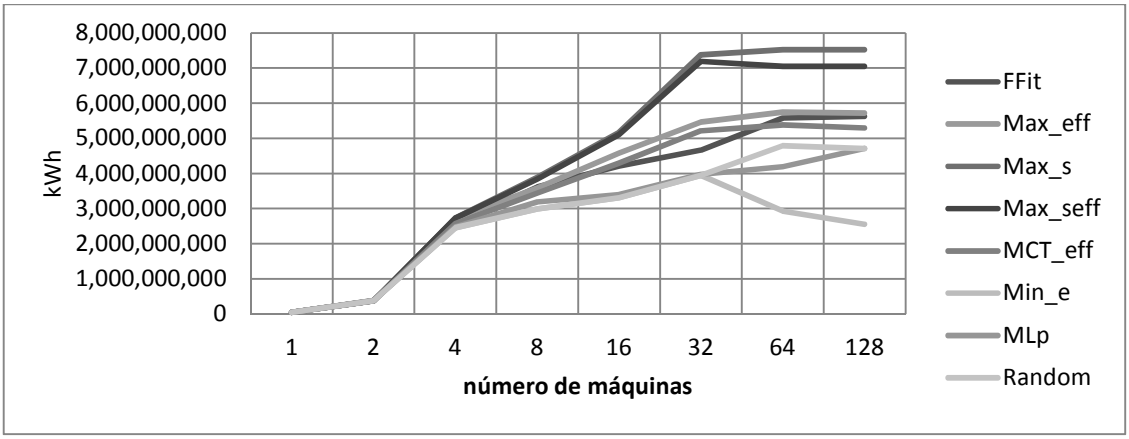


Figura 32(a). Consumo total de energía usando SLA con 2 SLs

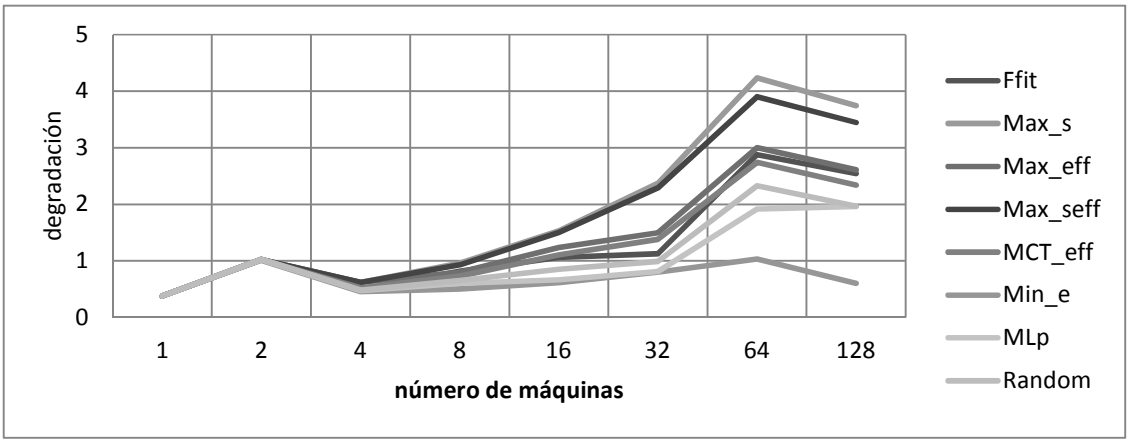


Figura 32(b). Degradación del consumo energético usando SLA con 2 SLs

En las Figuras 33(a) y 33(b) presentamos los resultados de ejecutar los trabajos con SLA que cuentan con 3 niveles de servicio. Las tendencias son parecidas, en el consumo total de energía tenemos que la asignación a la máquina con menor consumo de energía tiene un desempeño casi 4 veces mejor que la estrategia que asigna a la máquina más rápida, esto al tener 128 máquinas, cuando tenemos 32 máquinas en general la diferencia no es tan grande, sin embargo, Min_e presenta una ahorro de casi la mitad de energía.

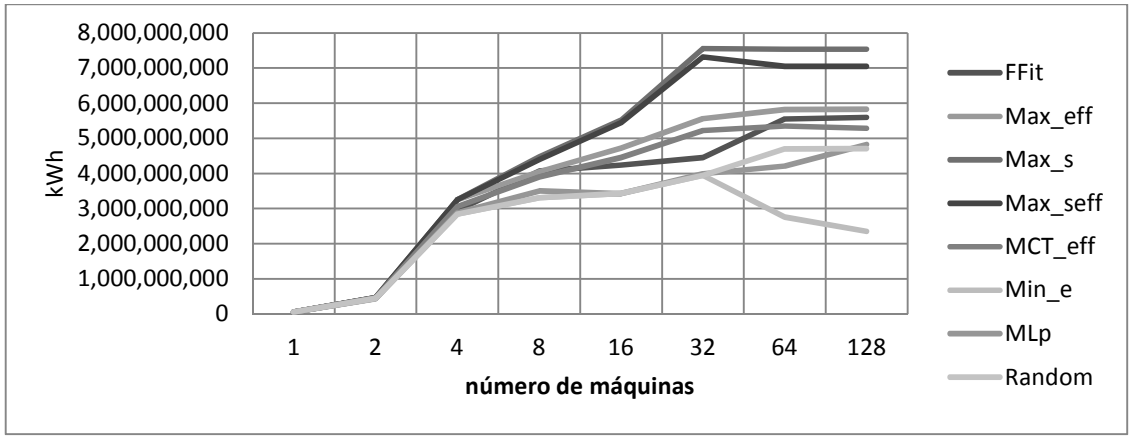


Figura 33(a). Consumo total de energía usando SLA con 3 SLs

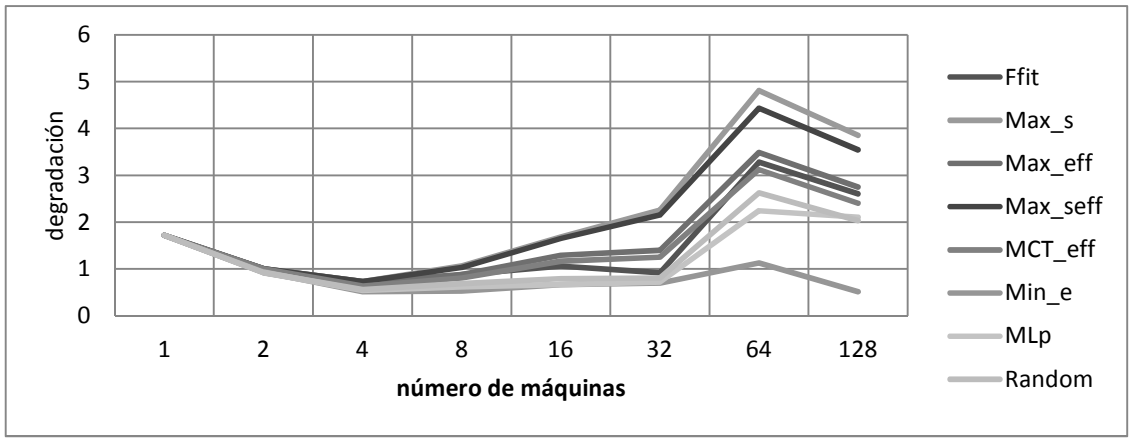


Figura 33(b). Degradación del consumo energético usando SLA con 3 SLs

En las Figuras 34(a) y 34(b) presentamos los resultados de ejecutar los trabajos con SLA 4. En estos experimentos usando SLA con 4 niveles de servicio, las tendencias del consumo y degradación de energía, siguen siendo parecidas, los número se incrementan un poco, pero el comportamiento general es similar.

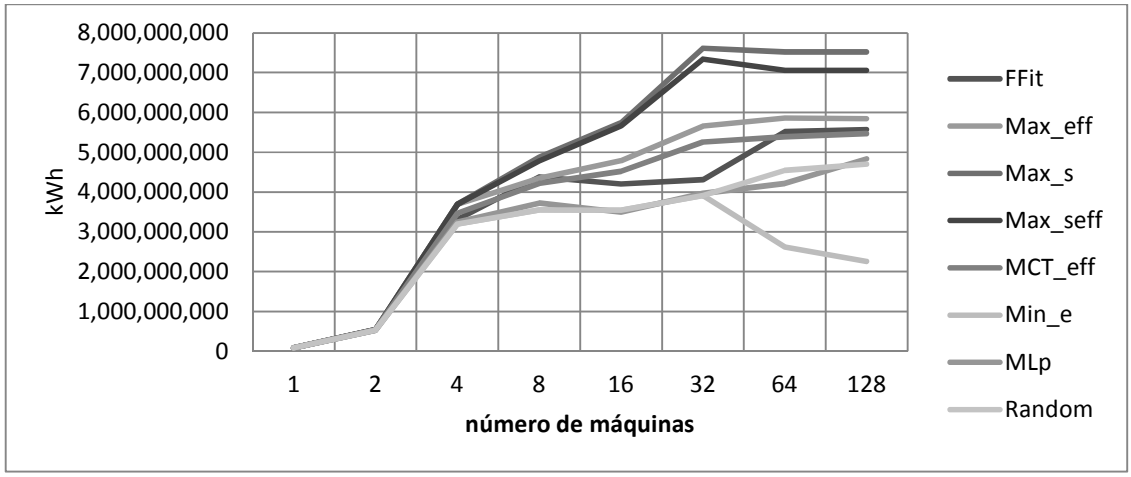


Figura 34(a). Consumo total de energía usando SLA con 4 SLs

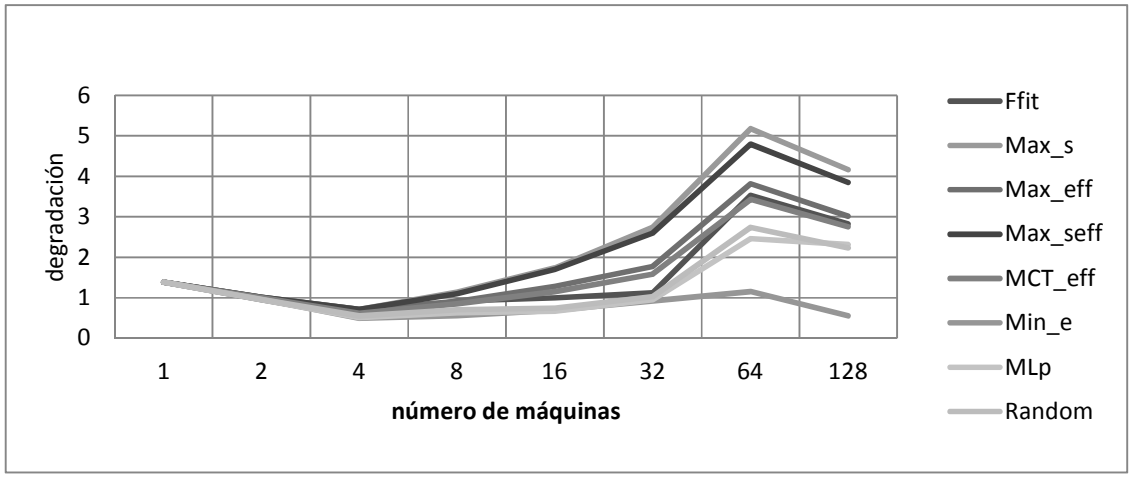


Figura 34(b). Degradación del consumo energético usando SLA con 4 SLs

Pasemos al análisis del último escenario, donde usamos SLA con 5 niveles de servicio, dichos resultados están expresados en las figuras 35(a) y 35(b) aquí vemos que la tendencia sigue: conforme aumentamos el número de niveles de servicio en la SLA, el consumo de energía crece ligeramente y la degradación que presenta cada escenario es un poco mayor que el escenario con SLA menor.

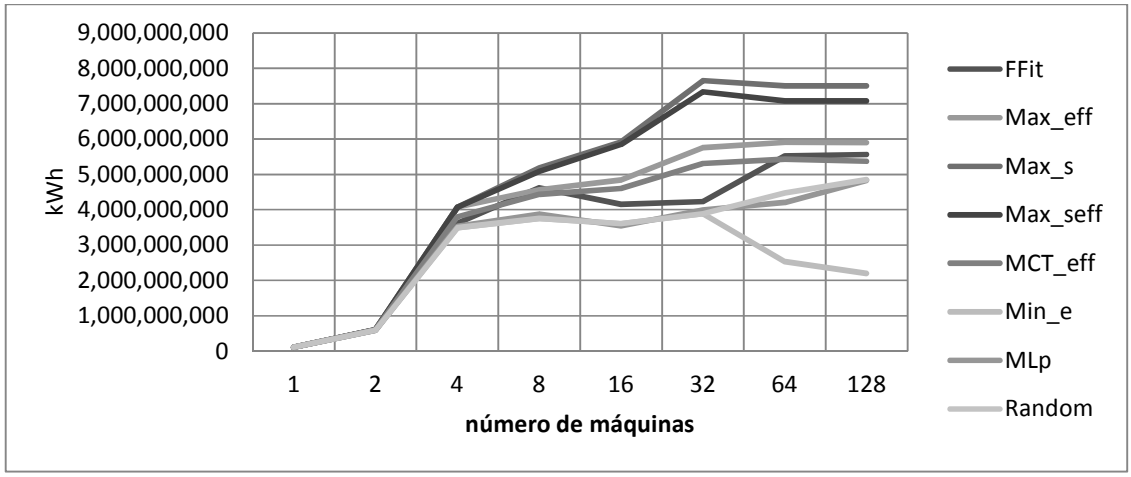


Figura 35(a). Consumo total de energía usando SLA con 5 SLs

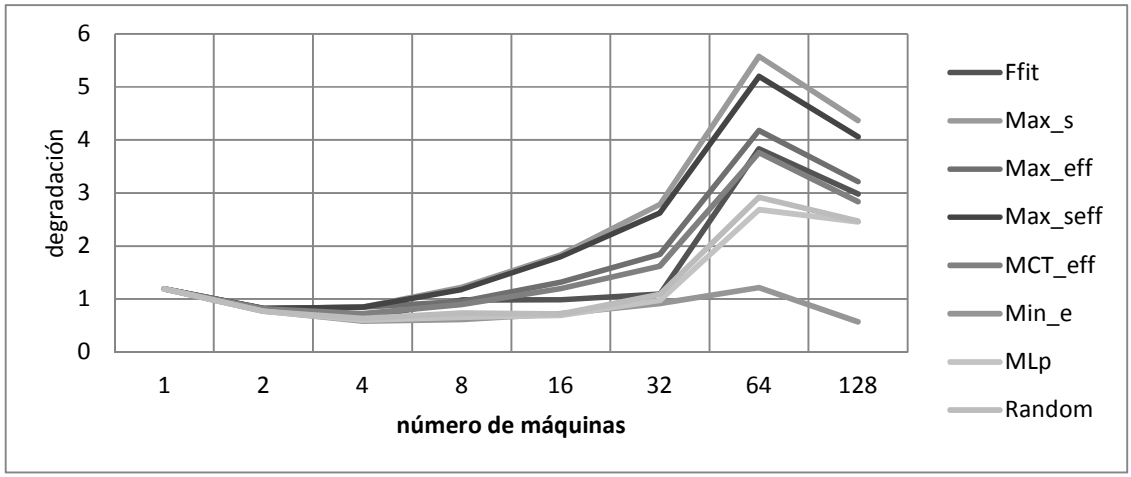


Figura 35(b). Degradación del consumo energético usando SLA con 5 SLs

En general hemos visto hasta ahora que las degradaciones en el escenario con 128 máquinas decrecen con respecto de la degradación del escenario con 64 máquinas, esto se debe a que al tener mayor número de máquinas las estrategias tienen mayor diversidad para la asignación de recursos, en general todas las estrategias toman ventaja de la diversidad. Sin embargo, Min_e es la estrategia que indiscutiblemente da mejor resultado para minimizar el consumo de energía.

Capítulo 6

Análisis bi-objetivo

En capítulos anteriores presentamos análisis por cada métrica: ganancia y energía por separado, pero nos interesa encontrar la estrategia que genere los mejores resultados teniendo el mejor compromiso entre generación de ganancia y consumo energético. Para realizar dicho análisis hacemos uso de la técnica de degradaciones y ranking (la sección 6.1), Perfil de desempeño (la sección 6.2), Espacio de funciones objetivo y frentes de Pareto (la sección 6.2.1) y cubrimiento de conjunto (la sección 6.2.2).

6.1 Análisis de degradaciones

Una vez que tenemos los resultados de la experimentación de cada escenario (cada escenario consiste de la ejecución de las 30 cargas de trabajo en cada una de las 8 configuraciones con diferente número de máquinas), calculamos las degradaciones para cada escenario y hacemos el promedio de degradaciones de los 8 escenarios, así obtenemos la tabla 6,

Tabla 6. Degradaciones y Ranking

SLA	Estrategia	Ganancia	Energía	Promedio	Ranking G	Ranking E	Ranking
SLA 1	FFit	0.49	1.02	0.76	5	5	5
	Max_s	0.50	1.43	0.97	7	8	8
	Max_eff	0.50	1.08	0.79	6	6	6
	Max_seff	0.50	1.37	0.94	8	7	7
	MCT_eff	0.49	1.00	0.75	4	4	4
	Random	0.49	0.88	0.68	3	3	3
	Min_e	0.43	0.57	0.50	1	1	1
	MLp	0.47	0.76	0.62	2	2	2

Tabla 6. Degradaciones y ranking (continuación)

SLA	Estrategia	Ganancia	Energía	Promedio	Ranking G	Ranking E	Ranking
SLA2	FFit	0.55	1.29	0.92	5	5	5
	Max_s	0.56	1.86	1.21	6	8	8
	Max_eff	0.56	1.40	0.98	8	6	6
	Max_seff	0.56	1.76	1.16	7	7	7
	MCT_eff	0.54	1.28	0.91	4	4	4
	Random	0.54	1.09	0.81	3	3	3
	Min_e	0.47	0.68	0.58	1	1	1
	MLp	0.52	0.98	0.75	2	2	2
SLA 3	FFit	0.60	1.50	1.05	5	4	4
	Max_s	0.62	2.14	1.38	7	8	8
	Max_eff	0.62	1.66	1.14	8	6	6
	Max_seff	0.62	2.03	1.33	6	7	7
	MCT_eff	0.60	1.51	1.06	4	5	5
	Random	0.59	1.27	0.93	3	3	3
	Min_e	0.53	0.84	0.68	1	1	1
	MLp	0.57	1.19	0.88	2	2	2
SLA 4	FFit	0.59	1.52	1.06	5	4	4
	Max_s	0.60	2.32	1.46	7	6	6
	Max_eff	0.60	1.75	1.18	8	8	8
	Max_seff	0.60	2.20	1.40	6	7	7
	MCT_eff	0.59	1.61	1.10	3	5	5
	Random	0.59	1.30	0.94	4	3	3
	Min_e	0.54	0.84	0.69	1	1	1
	MLp	0.57	1.24	0.91	2	2	2
SLA 5	FFit	0.60	1.58	1.09	5	4	4
	Max_s	0.61	2.44	1.53	7	6	8
	Max_eff	0.61	1.85	1.23	8	8	6
	Max_seff	0.61	2.32	1.46	6	7	7
	MCT_eff	0.59	1.68	1.14	4	5	5
	Random	0.59	1.36	0.97	3	3	3
	Min_e	0.56	0.85	0.70	1	1	1
	MLp	0.58	1.29	0.93	2	2	2

En ésta tabla 6 para cada grupo de experimentos, cambiamos el SLA usado y hacemos el promedio de la degradación de los experimentos en cada escenario.

Por cada configuración de SLA, tenemos las 8 estrategias evaluadas y para cada estrategia el promedio de degradación de ganancia, energía y el promedio de las dos degradaciones. En las tres últimas columnas tenemos el posicionamiento que cada estrategia tiene con respecto a la mejor degradación. *Ranking G* es el posicionamiento en torno a la degradación de ganancia, *Ranking E* se refiere al posicionamiento en relación a la degradación de consumo energético y *Ranking* es el posicionamiento que cada estrategia tiene al promediar las degradaciones de ganancia y consumo de energía.

Podemos ver que la estrategia que asigna a la máquina que tiene menor consumo de energía (Min_e), en las cinco variaciones de experimentación con SLA, obtiene el mejor posicionamiento, tanto en ganancia como en consumo energético. La tabla 7 muestra las degradaciones y posicionamientos del promedio de los cinco SLAs.

Tabla 7: Degradación y Ranking promedio de todos los SLAs

Estrategia	Ganancia	Energía	Promedio	Ranking G	Ranking E	Ranking
FFit	0.57	1.38	0.98	5	4	4
Max_s	0.58	2.04	1.31	7	8	8
Max_eff	0.58	1.55	1.06	8	6	6
Max_seff	0.58	1.94	1.26	6	7	7
MCT_eff	0.56	1.42	0.99	4	5	5
Random	0.56	1.18	0.87	3	3	3
Min_e	0.51	0.76	0.63	1	1	1
MLp	0.54	1.09	0.82	2	2	2

Vemos los posicionamientos promedio, aquí notamos que la mejor estrategia de asignación de recursos es la que asigna a la máquina que tiene menor consumo de energía (Min_e), esto nos genera en promedio mejor ganancia y también menor consumo energético, el buen comportamiento de ésta estrategia se debe a que hace un balanceo de carga entre las máquinas y dicho balanceo de cargas lo

realiza en base al historial de carga de la máquina, ya que una máquina en un determinado momento, puede tener menor consumo de energía debido a varias razones, entre ellas, que no ha tenido carga asignada o que el consumo de energía por la ejecución de unidad de tiempo es menor, ambas situaciones provocan que el balance de la carga sea adecuado tanto, para generar ganancia mayor, como para tener menor consumo de energía. La segunda mejor estrategia en promedio es la que asigna a la máquina que tiene menor número de trabajos asignados para su ejecución, esto significa que también se busca hacer un balance de carga, pero éste balance es en relación al trabajo asignado y aún no ejecutado, mientras que en Min_e, es en base historial del trabajo realizado.

Hasta este momento el análisis nos dice que si no tenemos información sobre la velocidad de las máquinas ni sobre su eficiencia energética, lo mejor es asignar a la máquina que tiene menor número de trabajos asignados. Pero si contamos con la información de velocidades y eficiencia energética, la mejor opción es asignar a la máquina que ha consumido menos energía al momento de la decisión.

6.2 Perfil de desempeño

Como mencionamos en la sección 3.1.1 de la metodología del análisis, el hacer conclusiones basados en resultados de promedios puede tener algunos aspectos negativos. Para minimizar dichos aspectos negativos en esta sección presentamos los perfiles de desempeño de las estrategias evaluadas.

La Figura 36 muestra los perfiles de desempeño de las 8 estrategias con respecto de la métrica de ganancia generada. En esta figura podemos ver que si queremos que la solución sea 2 veces peor que la solución óptima, cualquier estrategia que usemos nos dará 90% de las soluciones. Sin embargo, podemos ver que Min_e nos da mayor porcentaje de soluciones que son menores a 1.3, 1.4,1.5 y 1.6 veces peor que la solución óptima. Esto nos dice que Min_e nos da mayor porcentaje de mejores soluciones que cualquier otra estrategia.

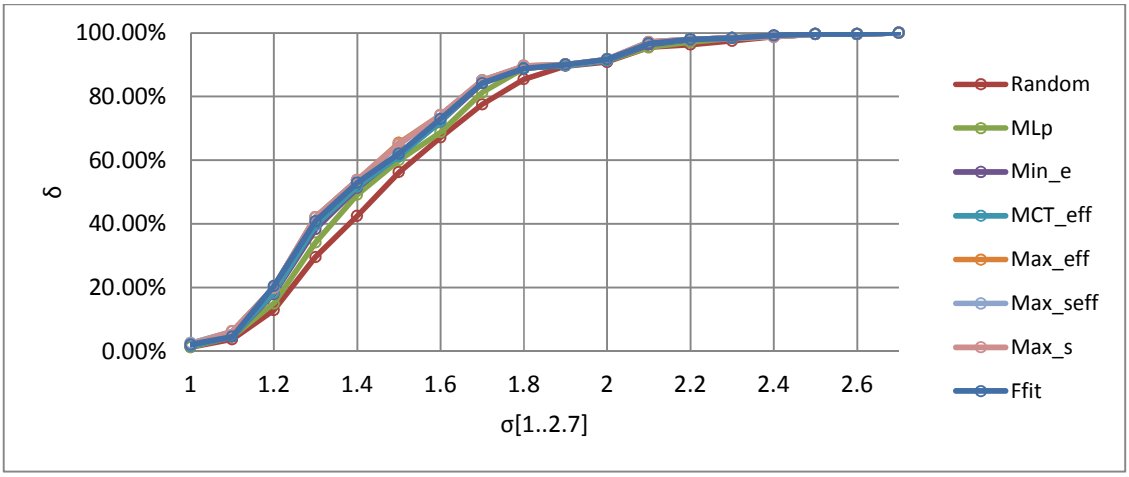


Figura 36. Perfiles de desempeño con respecto a la ganancia generada, 8 estrategias evaluadas

En la figura 37 vemos los perfiles de degradación de las 8 estrategias con respecto del consumo de energía.

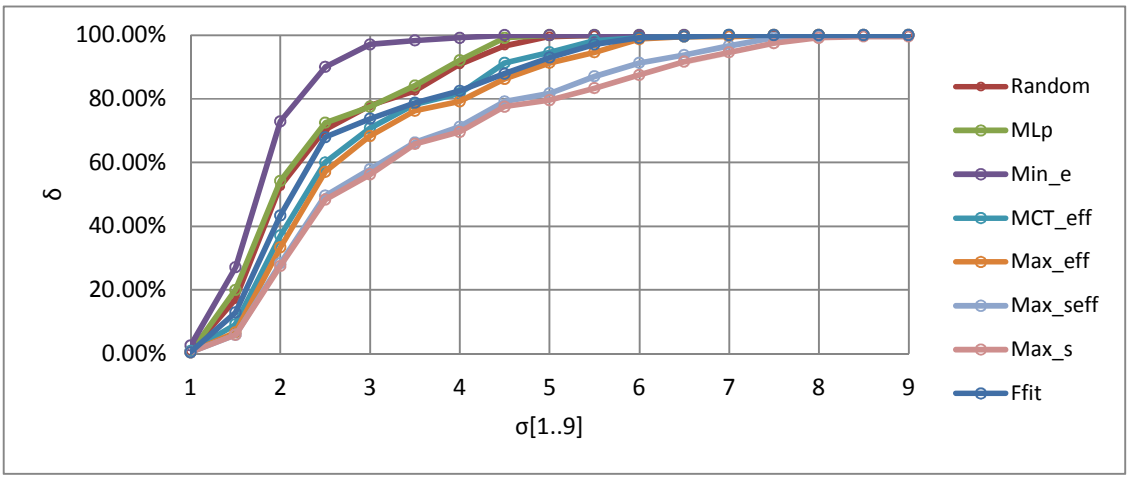


Figura 37. Perfiles de desempeño con respecto de la energía consumida, 8 estrategias evaluadas.

En los perfiles de desempeño de esta métrica es muy clara la mejoría que hay al usar la estrategia Min_e, pues el 97% de soluciones que genera son a lo más 3 veces peor que la solución óptima. Mientras que estrategias como Max_s genera únicamente 56% de soluciones con dicha calidad. Vemos que efectivamente MLp es la segunda mejor estrategia la cual genera 77% de soluciones con la calidad en cuestión.

Ahora analicemos la Figura 38, la cual presenta los perfiles de desempeño de las 8 estrategias al promediar las dos métricas: energía y ganancia.

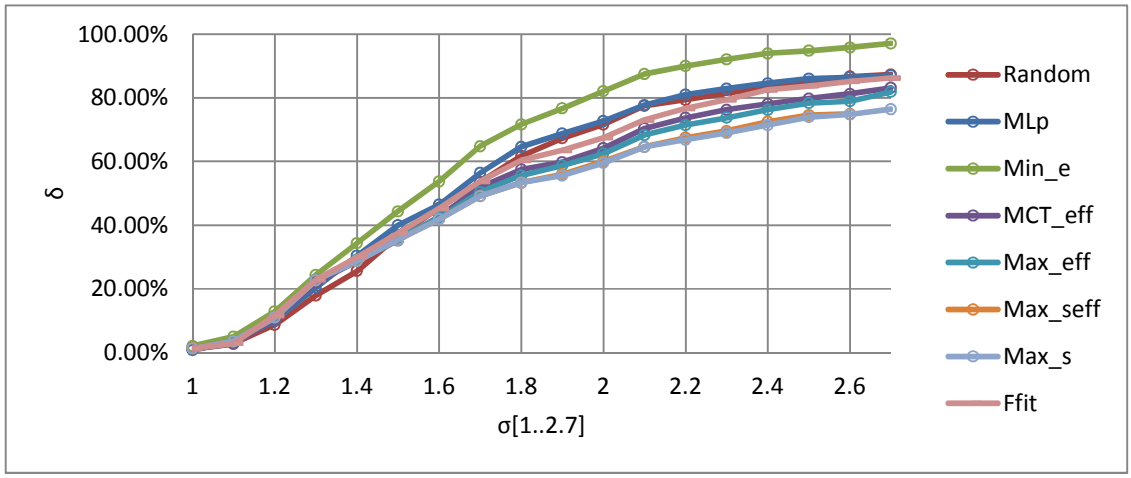


Figura 38. Perfiles de desempeño con respecto al promedio de energía y ganancia, 8 estrategias evaluadas.

Para realizar el promedio de las dos métricas establecimos al factor σ en el rango $[1\,..\,2.7]$, ya que factores mayores son despreciables desde el punto de vista de ganancia. Podemos ver que la estrategia Min_e 82% de las soluciones con calidad de a lo más dos veces peor que la solución óptima. La estrategia que le sigue es MLp generando 72% de soluciones con dicha calidad, sin embargo la estrategia de asignación aleatoria esta muy cerca generando 71% de soluciones con la misma calidad. Pero ahora podemos ver que ya sea por soluciones o por promedio, la estrategia Min_e está generando buenos resultados.

6.3 Pareto

En esta sección, comparamos los algoritmos en términos de optimalidad de Pareto. Una solución forma parte del conjunto óptimo de Pareto si ninguna otra solución es mejor en términos de todos los objetivos que se evalúan. En nuestro caso, denominamos una solución como óptima de Pareto, si ninguna otra solución examinada es mejor en ambos objetivos: energía y ganancia. El conjunto de dichas soluciones es referido como frente de Pareto y cualquier solución que no

forme parte del frente de Pareto puede ser considerada como de menor calidad que las que sí están incluidas.

La selección de soluciones incluidas en el frente de Pareto depende de las preferencias del sistema: si algún objetivo es considerado más importante que los demás, entonces se da preferencia a las soluciones que se desempeñan mejor en dicho objetivo incluso si los valores del objetivo secundario no están entre los mejores obtenidos. Nosotros usamos la métrica *Cubrimiento de conjunto* presentada por Ziztler (1999), la cual permite comparar dos algoritmos en términos de dominancia de Pareto y así evaluar el desempeño de un algoritmo con respecto del otro.

6.3.1 Espacio de funciones objetivo y frentes de Pareto

Nuestra meta es calcular un conjunto de soluciones para aproximar el frente de Pareto de cada una de las estrategias utilizadas (FFit, Max_s, Max_eff, Max_seff, MCT_eff, Min_e, Random, MLp), así obtenemos 8 aproximaciones de frentes de Pareto. Para ello tomamos cada una de las soluciones generadas en cada experimento, lo que implica tener 30 experimentos por 8 escenarios con diferente número de máquinas y las 5 SLAs, así tenemos $30 * 8 * 5 = 1200$ soluciones, de esas 1200 soluciones encontramos las no dominadas y formamos el frente de Pareto aproximado, esto para cada estrategia. A partir de ahora y a lo largo de nuestro análisis, a las aproximación del frente de Pareto, la llamaremos frente de Pareto sabiendo que se trata de una aproximación.

La Figura 39 muestra el espacio de funciones objetivo (1200 soluciones) y el frente de Pareto para la estrategia de asignación aleatoria (Random).

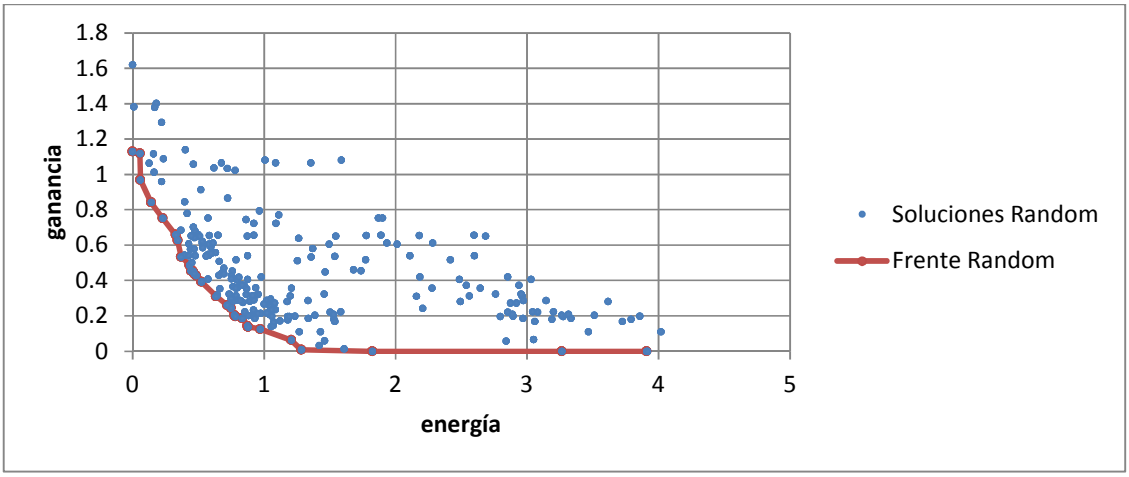


Figura 39. Espacio de funciones objetivo y frente de Pareto para la estrategia Random

La Figura 40 muestra el espacio de funciones objetivo y el frente de Pareto para la estrategia MLp, asignación a la máquina que tiene menor número de trabajos asignados en la cola de espera.

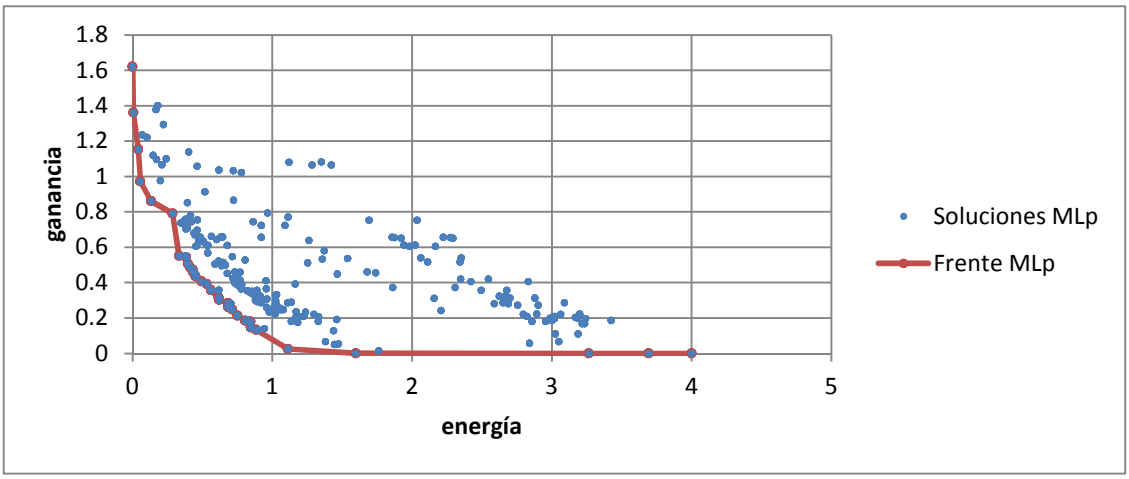


Figura 40. Espacio de funciones objetivo y frente de Pareto para la estrategia MLp

La Figura 41 muestra el espacio de funciones objetivo y el frente de Pareto para la estrategia FFit, asignación a la primera máquina disponible. En este espacio de funciones objetivo se nota un patrón en las soluciones, esto se debe a que se tiende a sobre cargar una determinada máquina, en este caso a la primer máquina que puede aceptar el trabajo.

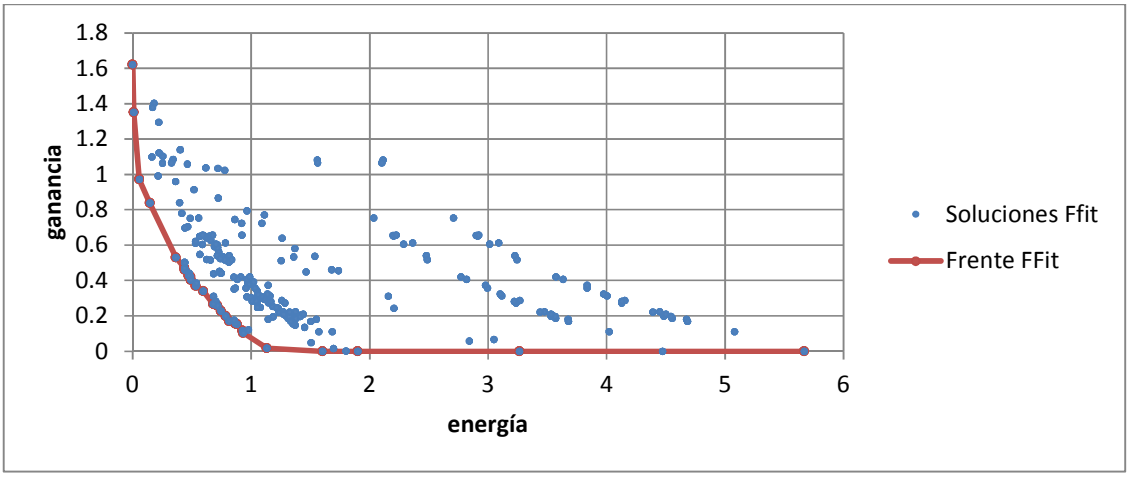


Figura 41. Espacio de funciones objetivo y frente de Pareto para la estrategia FFit

La Figura 42 muestra el espacio de funciones objetivo y el frente de Pareto para la estrategia Max_s, asignación a la máquina más rápida disponible. En esta gráfica también podemos ver que las soluciones muestran un patrón repetitivo y es que esta estrategia también tiende a saturar a una determinada máquina, la máquina más rápida.

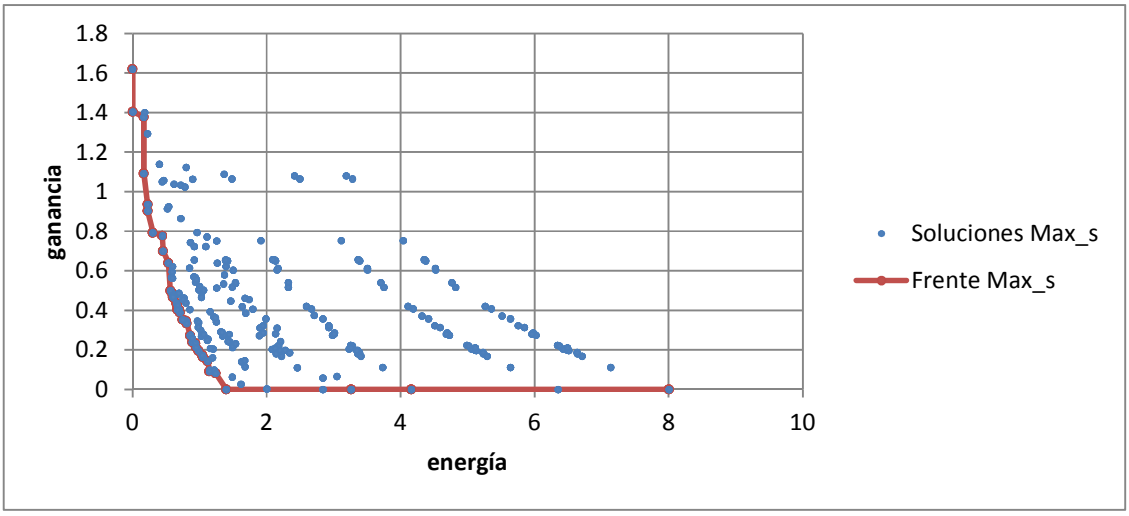


Figura 42. Espacio de funciones objetivo y frente de Pareto para la estrategia Max_s

La Figura 43 muestra el espacio de funciones objetivo y el frente de Pareto para la estrategia Max_eff, asignación a la máquina más eficiente energéticamente hablando. Dado que esta estrategia también tiene de a sobre cargar una

determinada máquina, vemos que el espacio de funciones objetivo también presenta patrones en su comportamiento.

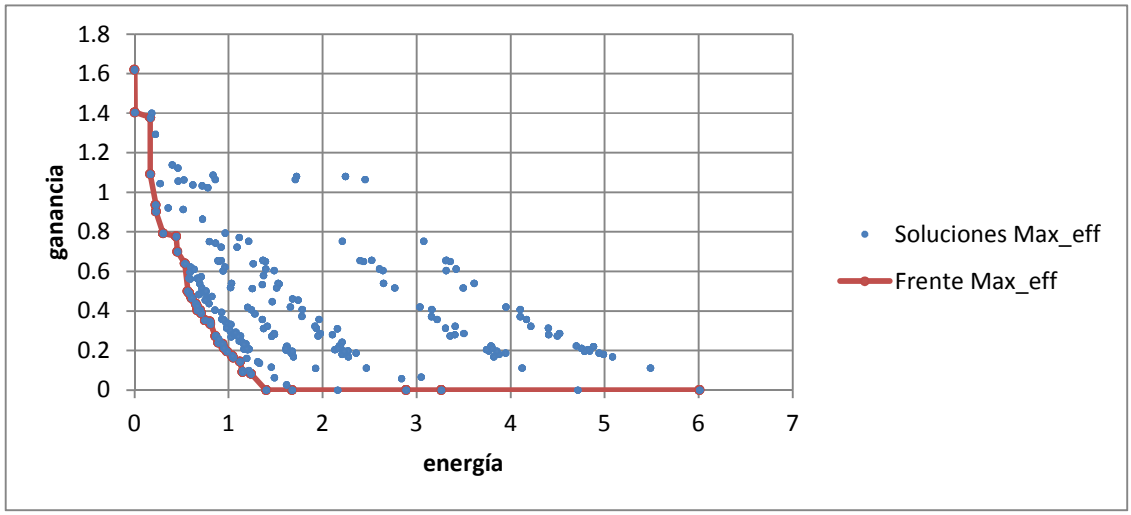


Figura 43. Espacio de funciones objetivo y frente de Pareto para la estrategia Max_eff

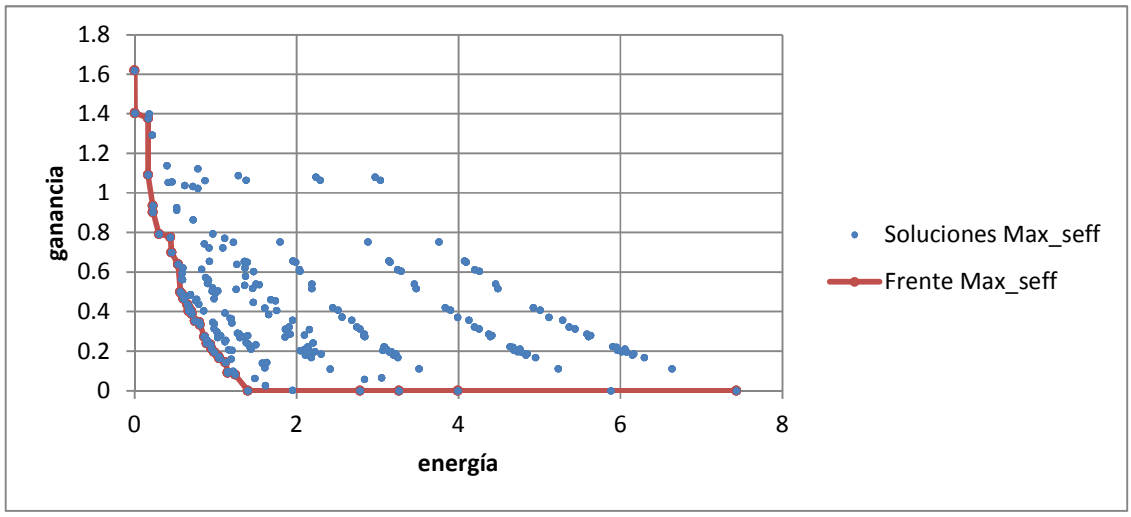


Figura 44. Espacio de funciones objetivo y frente de Pareto para la estrategia Max_eff

En la figura 44 tenemos el espacio de funciones objetivo y el frente de Pareto para la estrategia Max_seff, asignación a la máquina que tenga mejor compromiso entre rapidez y eficiencia energética. Esta estrategia también forma un patrón en el espacio de funciones objetivo. La siguiente figura (45) muestra el espacio de funciones objetivo y el frente de Pareto para la estrategia MCT_eff, asignación a la

máquina que tiene mejor compromiso entre mínimo tiempo de finalización y eficiencia energética. Esta estrategia tiene un buen desempeño, en cuanto a distribución de carga entre las máquinas, es por eso que aquí no se forman patrones repetitivos en el espacio de funciones objetivo.

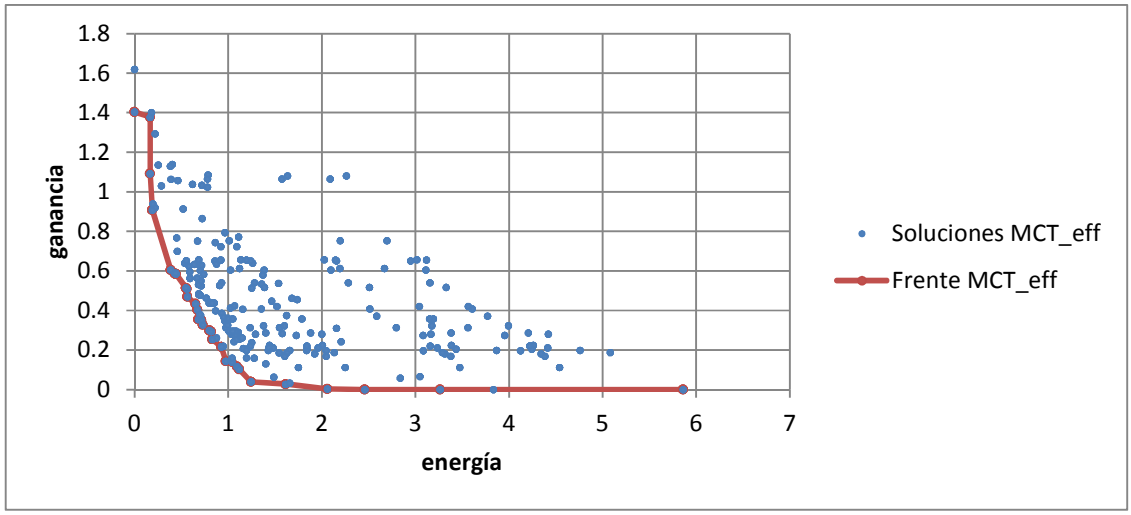


Figura 45. Espacio de funciones objetivo y frente de Pareto para la estrategia MCT_eff

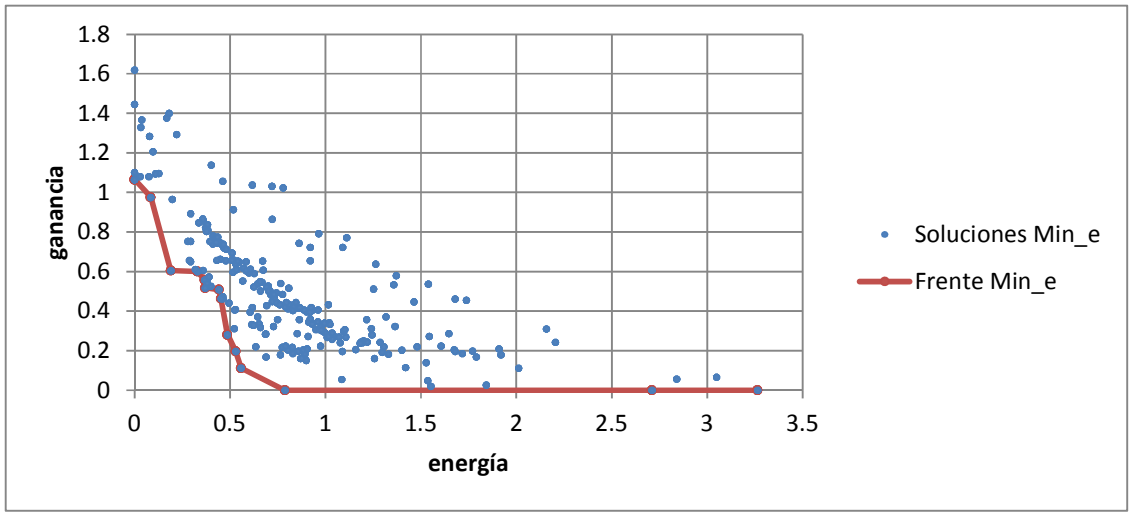


Figura 46. Espacio de funciones objetivo y frente de Pareto para la estrategia Min_e

En la Figura 46 vemos el espacio de funciones objetivo y el frente de Pareto generado por la estrategia Min_e, en estas gráficas por separado, podemos notar que el espacio de funciones objetivo y frente de Pareto de Min_e, están más

aglomerados en la esquina inferior izquierda, lo que significa que tendrán mejor dominancia sobre los frentes de otras estrategias. Para confirmar esto veamos la Figura 47, la cual muestra las ocho aproximaciones a los frentes de Pareto, generados por las estrategias. Aquí podemos ver que el frente generado por Min_e cubre en gran parte a los frentes de las otras estrategias.

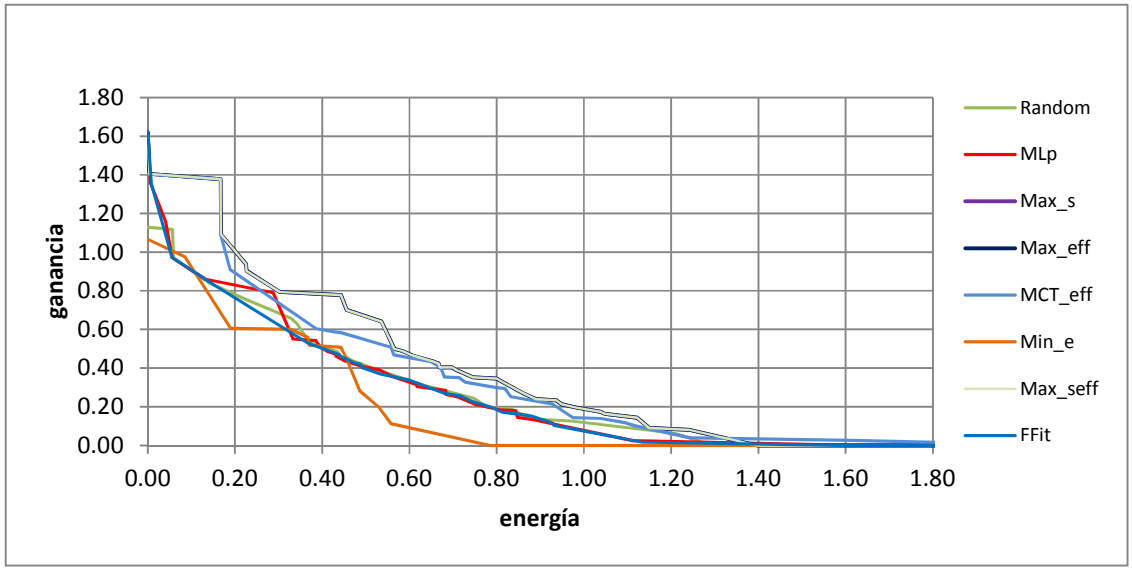


Figura 47. Frentes de Pareto de las ocho estrategias evaluadas

6.3.2 Cubrimiento de conjunto

En esta sección presentamos los resultados de usar la métrica cubrimiento de conjunto (Cubrimiento de conjunto). Para cada uno de los ocho frentes de Pareto calculamos la métrica conjunto cobertura, los resultados son presentados en la tabla 8.

Las filas de la tabla muestran los valores de la métrica para la dominancia de la estrategia A con respecto de la estrategia B y las columnas indican la dominancia de B con A. Las dos últimas columnas indican el promedio de dominancia para cada fila y el Ranking asignado a cada algoritmo, en base a la dominancia

promedio. De igual forma las dos últimas filas muestran promedio de dominancia y ranking de la estrategia en cada columna.

Tabla 8: Cubrimiento de conjunto

	FFit	Max_s	Max_eff	Max_seff	MCT_eff	Min_e	Random	MLp	**Promedio**	**Ranking**
FFit	1.00	0.92	0.92	0.92	0.93	0.40	0.39	0.26	**0.72**	**2**
Max_s	0.19	1.00	1.00	1.00	0.20	0.13	0.11	0.16	**0.47**	**4**
Max_eff	0.19	1.00	1.00	1.00	0.20	0.13	0.11	0.16	**0.47**	**4**
Max_seff	0.19	1.00	1.00	1.00	0.20	0.13	0.11	0.16	**0.47**	**4**
MCT_eff	0.11	0.70	0.68	0.71	1.00	0.13	0.07	0.13	**0.44**	**5**
Min_e	0.70	0.97	0.95	0.97	0.93	1.00	0.68	0.74	**0.87**	**1**
Random	0.26	0.95	0.92	0.95	0.90	0.33	1.00	0.26	**0.70**	**3**
MLp	0.33	0.95	0.89	0.92	0.90	0.33	0.39	1.00	**0.72**	**2**
Promedio	**0.37**	**0.94**	**0.92**	**0.93**	**0.66**	**0.33**	**0.36**	**0.36**		
Ranking	**3**	**7**	**5**	**6**	**4**	**1**	**2**	**2**		

Vemos que en el renglón de Min_e tenemos que en promedio domina el 87% de los frentes de las otras estrategias y en la columna de Min_e en promedio es dominada en 33% por los frentes de las otras estrategias. Así el ranking de las estrategias, por renglón, cuanto mayor es el porcentaje de cobertura, mayor es el ranking, pues esto implica que el frente es mejor y para el rankin por columnas entre menor es el promedio de dominancia, mejor es el frente.

Así de acuerdo a la métrica cubrimiento de conjunto, la estrategia que da mejor compromiso entre maximización de ganancia y minimización de consumo de energía, es Min_e, seguida de MLp que tiene posición 2. Este resultado es congruente con los resultados que encontramos en el análisis basado en las degradaciones promedio, así como el resultado encontrado de acuerdo a los perfiles de desempeño.

Capítulo 7

Conclusiones

Hoy en día el cómputo bajo demanda está tomando fuerza y es necesario utilizar estrategias adecuadas para satisfacer las demandas tanto de los clientes como de los administradores. Del punto de vista de los administradores el requerimiento va desde la maximización de ganancia, teniendo control de admisión para satisfacer calidad de servicio, hasta minimizar el consumo de energía. Los requerimientos de los usuarios es recibir el cómputo que requieren a un costo adecuado, teniendo garantías de calidad de servicio.

En este trabajo abordamos la calendarización de trabajos de cómputo de alto desempeño en la nube bajo el modelo de Infraestructura como Servicio de Nube. Hacemos uso de Acuerdos de Nivel de Servicio para establecer las responsabilidades de las dos partes involucradas: proveedor y usuario del servicio de nube. Las responsabilidades del usuario son expresadas como el monto a pagar por el servicio que recibirá y las del proveedor, ofrecer el servicio manteniendo las garantías de QoS por las que el usuario pagó. La calendarización se lleva a cabo en dos niveles: aceptación y asignación de recursos. Para la aceptación usamos un algoritmo voraz y propusimos diferentes estrategias de asignación de recursos en el entorno de IaaS. El objetivo es maximizar la ganancia y minimizar el consumo de energía, así como encontrar el mejor compromiso entre niveles de servicio en una SLA.

Aquí realizamos la simulación de varios experimentos, donde los escenarios fueron evolucionando de ser escenarios simples a más realistas. Para el análisis que realizamos, utilizamos optimalidad de Pareto, donde decimos que: tanto la maximización de ganancia como la minimización del consumo energético tienen la misma prioridad.

En la experimentación más realista, donde usamos SLAs con múltiples niveles de servicio e infraestructura con múltiples máquinas heterogéneas, encontramos dos características deseables:

- Podemos minimizar el consumo de energía.
- Podemos mantener la calidad de los calendarios, es decir, generar la máxima ganancia.

Dichas ventajas las logramos usando la asignación a la máquina que tiene menor consumo de energía al momento de la decisión (Min_e). Esto es debido a que en la calendarización en línea, el asignar el trabajo a la máquina que tiene menor consumo energético genera un balance de asignación de trabajos, pues en determinado momento una máquina puede tener consumo energético bajo por diferentes razones:

- La máquina no tiene trabajos para ejecución.
- La máquina tiene bajo consumo energético por la ejecución de los trabajos.
- La máquina tiene buen compromiso entre rapidez y consumo energético.

Adicionalmente encontramos que cuando no tenemos información sobre la infraestructura, la mejor opción es asignar a la máquina que tiene menos trabajos en su cola de espera (MLp).

Referencias

Amazon web services (2013). Recuperado de: http://aws.amazon.com

Aragón Ayala, Alonso Mitza (2012). *Diseño y análisis experimental de algoritmos de calendarización de bajo consumo de energía para Grids jerárquicos de dos niveles*. (Tesis de maestría, Centro de Investigación Científica y de Educación Superior de Ensenada, Baja Claifornia). Recuperada de: http//:www.biblioteca.cicese.mx

Barrondo Corral, Aritz (2012). *Eficiencia energética de algoritmos de calendarización para Grids de computadoras personales punto-a-punto*. (Tesis de maestría, Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California). Recuperada de: http//:www.biblioteca.cicese.mx

Chapin, S. J., Cirne, W., Feitelson, D. G., Jones, J. P., Leutenegger, S. T., Schwiegelshohn, U. (1999). *Benchmarks and Standards for the Evaluation of Parallel Job Schedulers*. Proceedings of the Job Scheduling Strategies for Parallel Processing, pp 67-90. Springer-Verlag.

Dolan, E. D. y Moré, J. J. (2002). *Benchmarking optimization software with performance profiles*. Mathematical Programming, vol.91: pp 201-213. Springer-Verlag.

Dolan, E. D., Mor_e, J. J., y Munson, T. S. (2006). *Optimality Measures for Performance Profiles*. SIAM J. on Optimization, vol.16: pp 891-909. Society for Industrial and Applied Mathematics.

Feitelson, D.(2008). Parallel Workloads Archive. Recuperado de*: http://www.cs.huji.ac.il/labs/parallel/workload/*

Feitelson, D., Rudolph, L., Schwiegelshohn, U., Sevcik, Kenneth C., & Wong, P.(1997). *Theory and practice in parallel job scheduling.* Proceedings of the Job Scheduling Strategies for Parallel Processing, pp 1-34. Springer-Verlag.

Frachtenberg, E., & Feitelson, D. (2005). *Pitfalls in parallel job scheduling evaluation*. Proceedings of the 11th international conference on Job Scheduling Strategies for Parallel Processing, pp 257-282. Spreinger-Verlag

Garg, S., Gopalaiyengar, S. (2011). *SLA-based Resource Provisioning for Heterogeneous Workloads in a Virtualized Cloud Datacenter*. Algorithms and Architectures for Parallel Processing, pp 371–384. Springer-Verlag

Gupta, B. D., Palis, M. A.(2001). *Online real-time preemptive scheduling of jobs with deadlines on multiple machines*. Journal of Scheduling, vol.4: pp 297–312. Springer

Hirales Carbajal, A., Tchernykh, A., Roblitz, T., & Yahyapour, R.(2010). *A Grid simulation framework to study advance scheduling strategies for complex workflow applications*. Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on, IEEE Comput. Soc, pp 1–8.

IBM.(2010). Review and summary of cloud service level agreements. Recuperado de: http://public.dhe.ibm.com/software/dw/cloud/library/cl-rev2sla-pdf.pdf, pp 1–10.

Iosup, A., Li, H., Jan, M., et al.(2008). *The Grid Workloads Archive*. Future Generation Computer Systems, vol.24: pp 672–686. Elsevier

Islam, M., Balaji, P., Sadayappan, P., & Panda, D. K.(2003). *QoPS : A QoS based scheme for Parallel Job Scheduling*. Job Scheduling Strategies for Parallel Processing, pp 252–268. Springer

Kirschnick, J., Alcaraz Calero, J.M., Wilcock L, Edwards, N. (2010). *Toward an Architecture for the Automated Provisioning of Cloud Services*. IEEE Communications Magazine vol.48: pp124 –131.

Lezama Barquet, Anuar (2012). *Experimental analysis of a HPC model of infrastructure as a service cloud*. (Tesis de maestría en ciencias. Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California). Recuperada de: http//:www.biblioteca.cicese.mx

Mell, P., Grance, T.(2011). *Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology*. Recuperado de: http://csrc.nist.gov/publications/PubsSPs.html#800-145.

Quezada Pina, A. Tchernykh, A. González García, J. L. Hirales Carbajal, A. Ramírez Alcaraz, J. M. Schwiegelshohn, U. Yahyapour, R. Miranda López, V. (2012). *Adaptive parallel job scheduling with resource admissible allocation on two-level hierarchical grids*. Future Generation Computer Systems, vol.28: pp. 965-976. Elsevier Science

Ramírez Alcaraz, J. M., Tchernykh, A., Yahyapour, R., Schwiegelshohn, U., Quezada-Pina, A., González-García, J. L., y Hirales-Carbajal, A. (2011). *Job allocation strategies with user run time estimates for online scheduling in hierarchical grids*. Journal. Grid Computing., vol.9: pp 95-116.

Sakellariou, R., Yarmolenco, V. (2008) *Job Scheduling on the Grid: Towards SLA-Based Scheduling in Advances in Parallel Computing*. High Performance Computing and Grids in Action, vol 16: pp 207-222. IOS press

Schwiegelshohn, U., Tchernykh, A.(2012*). Online Scheduling for Cloud Computing and Different Service Levels*. 26th IEEE International Parallel & Distributed Processing Symposium, pp 1061–1068.

Sommers, J. et al. (2007) *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication* . ACM SIGCOMM'07, pp 109-120

Tchernykh, A. Schwiegelsohn, U. Yahyapour, R. Kuzjurin, N. (2010). *Online Hierarchical Job Scheduling on Grids with Admissible Allocation*. Journal of Scheduling, vol.13: pp. 545—552. Springer-Verlag

Tchernykh, A. Pecero Sanchez, J. E., Barrondo Corral, Schaeffer, A. E. (2013). *Adaptive Energy Efficient Scheduling in Peer-to-Peer Desktop Grid*s. Future Generation Computer Systems. Elsevier

Tsafrir, D., Etsion, Y., y Feitelson, D. G. (2007*). Backfiling using system-generated predictions rather than user runtime estimates*. IEEE Trans. Parallel Distrib. Syst., vol.18: pp 789-803.

Zitzler, E.: (1999). *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. (Tesis de doctorado. Swiss Federal Institute of Technology, Zurich, Zuiza). Recuperada de: http://www.tik.ee.ethz.ch/~sop/publicationListFiles/zitz1999a.pdf

Apéndices

Figura A1. Configuración de velocidades y eficiencia energética de la infraestructura

$Máquina$	P^{idle}	P^{work}	eff	s
1	20	101	1.45	1
2	67	71	4.7	5.63
3	30	94	29.14	35.36
4	74	114	29.76	54.79
5	52	46	29.84	28.64
6	28	69	27.28	25.86
7	47	103	21.77	31.72
8	36	102	6.97	8.76
9	56	66	16.18	18.98
10	27	35	25.57	15.47
11	28	48	20.69	15.25
12	18	66	28.55	23.46
13	22	52	25.82	18.65
14	64	40	29.75	30.3
15	27	109	26.32	34.95
16	30	132	26.34	41.67
17	92	32	14.24	16.89
18	56	101	15.66	23.62
19	78	62	4.56	5.52
20	83	103	12.79	22.63
21	66	190	20.62	51.21
22	93	90	30.6	54.86
23	81	165	22.9	54.8
24	59	88	8.69	11.87
25	22	46	12.16	7.85
26	29	63	15.23	13.44
27	85	37	13.05	15.16
28	75	159	23.64	53.87
29	36	95	27.68	35.45
30	77	105	6.32	10.38
31	48	72	6.81	7.43
32	63	69	9.65	11.92
33	46	111	23.38	35.74
34	35	72	29.89	31.32

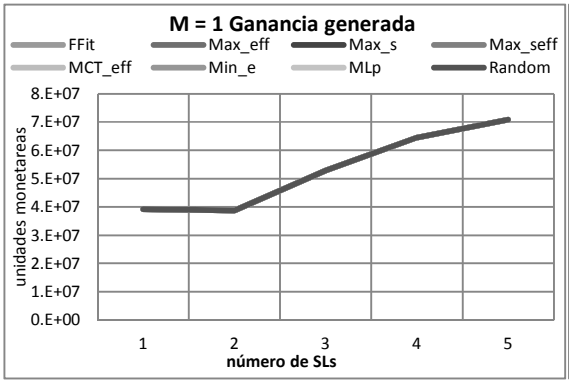
$Máquina$	P^{idle}	P^{work}	eff	s
35	71	113	28.9	52.03
36	38	46	26.12	21.42
37	63	54	23.96	27.31
38	73	123	8.91	16.25
39	71	87	31.18	48.29
40	90	83	11.71	19.19
41	47	69	25.03	28.32
42	30	111	18.37	25.03
43	87	103	3.6	5.66
44	37	40	22.19	16.61
45	79	88	27.23	44.44
46	73	73	21.97	31.17
47	51	78	29.59	37.37
48	32	90	17.69	20.82
49	80	100	18.94	32.97
50	66	44	22.86	24.46
51	71	113	22.01	39.35
52	96	90	16.29	29.14
53	55	87	8.07	10.58
54	53	105	3.89	5.16
55	59	115	7.26	11.56
56	88	62	8.13	11.26
57	96	110	9.76	18.83
58	40	100	26.38	36.07
59	58	136	30.49	57.95
60	62	54	2.05	1.66
61	95	103	6.31	11.26
62	30	95	30.6	37.47
63	43	105	10.78	15.04
64	36	107	21.45	29.79
65	74	29	29.19	29.43
66	73	55	3	3.04
67	90	112	1.33	1.44
68	95	112	1.65	2.14

Figura A1. Configuración de velocidades y eficiencia energética de la infraestructura (continuación)

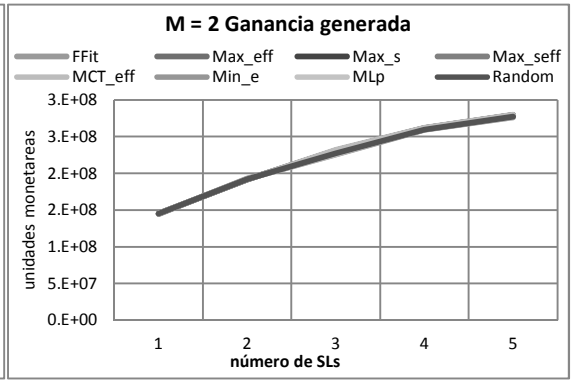
$Máquina$	P^{idle}	P^{work}	eff	s
69	102	136	1.1	1.14
70	91	71	2.4	2.88
71	113	135	1.64	2.54
72	111	116	2.13	3.43
73	55	54	2.01	1.51
74	54	35	4.81	3.73
75	64	133	1.27	1.28
76	104	125	2.18	3.58
77	89	45	3.04	3.24
78	94	144	2.13	3.6
79	65	131	1.93	2.57
80	97	86	1.94	2.41
81	27	136	2.91	3.74
82	36	118	1.93	2.02
83	49	62	2.59	2.19
84	61	58	2.43	2.15
85	65	43	3.85	3.49
86	93	54	1.88	1.85
87	45	38	2.74	1.76
88	68	115	2.08	2.67
89	33	99	3.43	3.71
90	56	148	2.19	3.2
91	47	23	6.21	3.91
92	42	126	1.68	1.78
93	33	125	2.42	2.85
94	112	142	1.95	3.38
95	29	29	2.45	1.06
96	71	107	1.89	2.26
97	97	91	1.26	1.2
98	85	88	1.28	1.15

$Máquina$	P^{idle}	P^{work}	eff	s
99	73	17	5.01	3.95
100	126	124	2.02	3.49
101	35	22	5.16	2.59
102	54	109	2.4	2.9
103	28	18	4.1	1.6
104	28	88	1.63	1.17
105	62	146	1.16	1.12
106	37	63	3.41	2.79
107	72	111	1.85	2.25
108	74	79	3.22	3.98
109	89	84	2.88	3.91
110	30	118	2.15	2.26
111	119	130	1.96	3.34
112	50	141	1.77	2.2
113	75	129	2.42	3.67
114	97	123	1.87	2.87
115	85	141	1.26	1.44
116	29	83	1.69	1.2
117	78	21	4.5	3.84
118	53	54	3.26	2.83
119	96	61	1.45	1.3
120	87	69	2.79	3.38
121	143	131	1.58	2.64
122	59	35	4.72	3.85
123	48	94	1.66	1.47
124	79	75	2.57	3.01
125	19	135	1.79	1.8
126	22	112	1.93	1.75
127	79	119	2.09	2.91
128	58	133	2.46	3.51

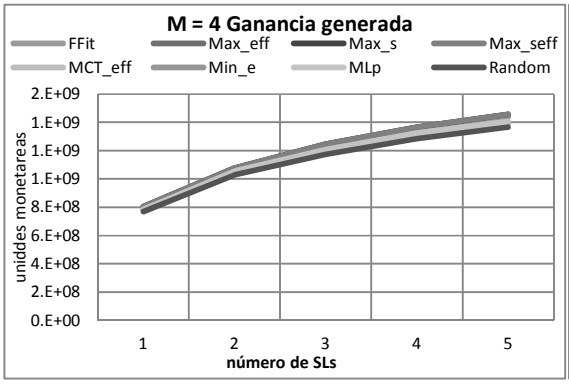
Figura A2. Ganancia total generada por escenario



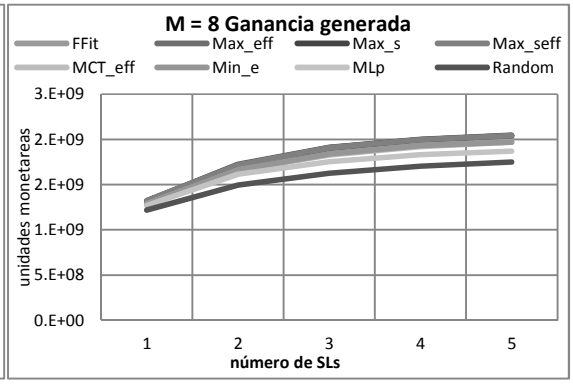
A2(a). Ganancia generada por cada SLA, m=1



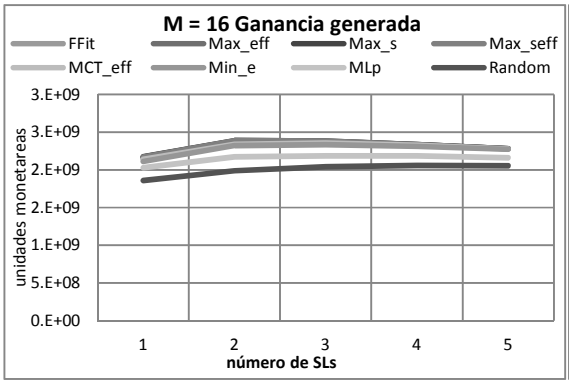
A2(b). Ganancia generada por cada SLA, m=2



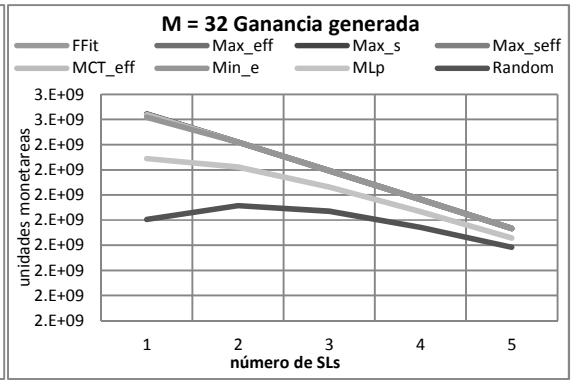
A2(c). Ganancia generada por cada SLA, m=4



A2(d). Ganancia generada por cada SLA, m=8

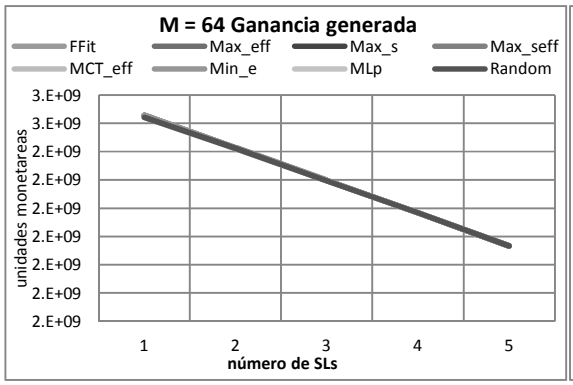


A2(e).Ganancia generada por cada SLA, m=16

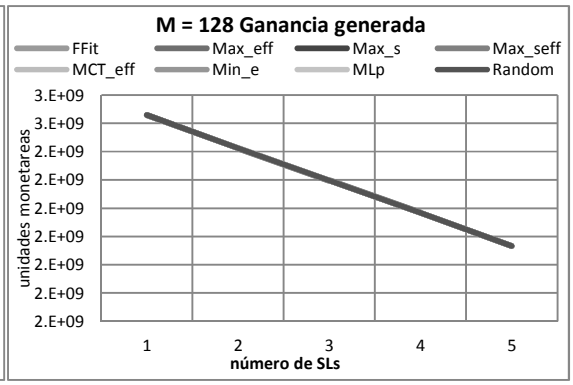


A2(f). Ganancia generada por cada SLA, m=32

Figura A2. Ganancia total generada por escenario (continuación)

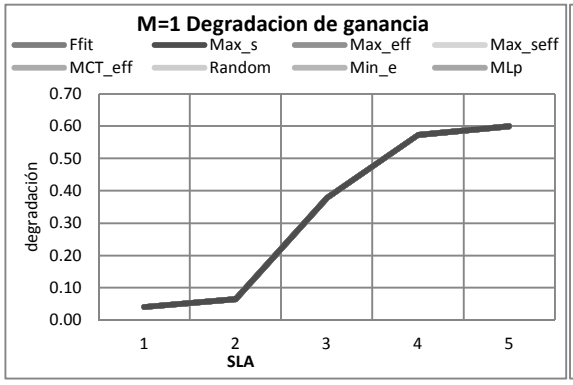


A2(g) Ganancia generada por cada SLA, m=64

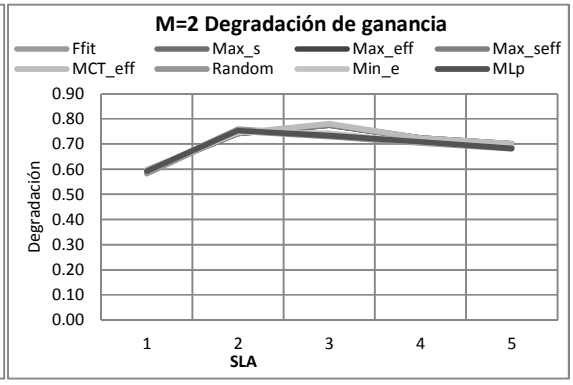


A2(h).Ganancia generada por cada SLA, m=128

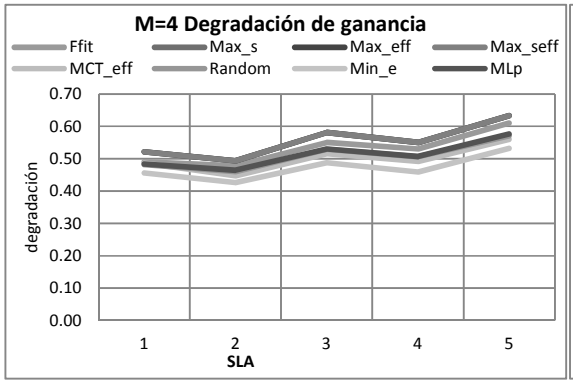
Figura A3. Degradación de ganancia por cada SLA en cada escenario con diferente número de máquinas



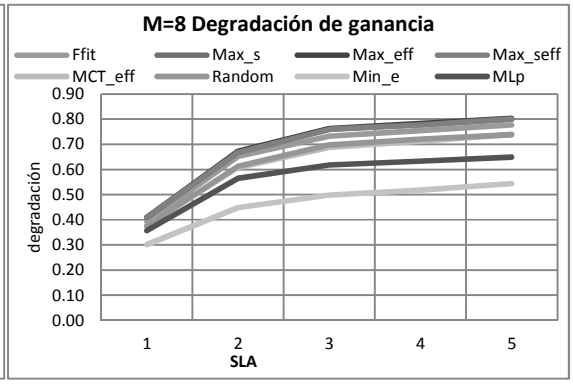
A3(a). Degradación de ganancia m=1



A3(b). Degradación de ganancia m=2

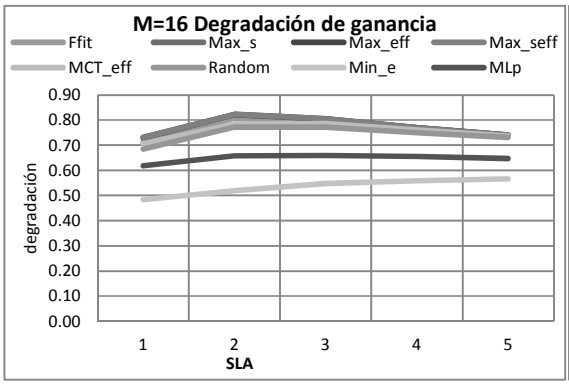


A3(c). Degradación de ganancia m=4

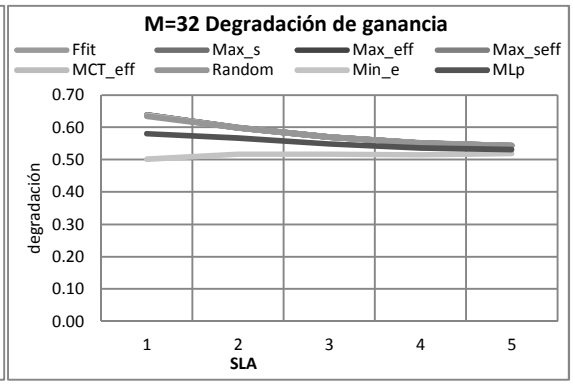


A3(d). Degradación de ganancia m=8

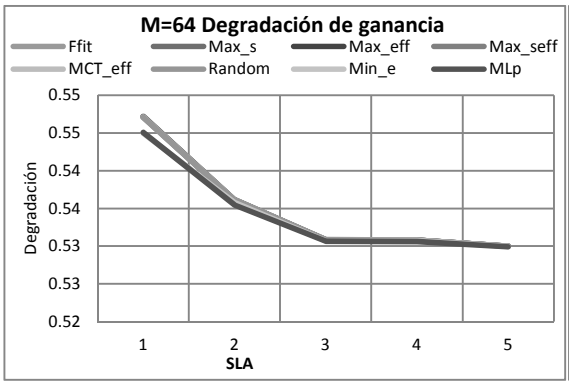
Figura A3. Degradación de ganancia por cada SLA en cada escenario con diferente número de máquinas (continuación)



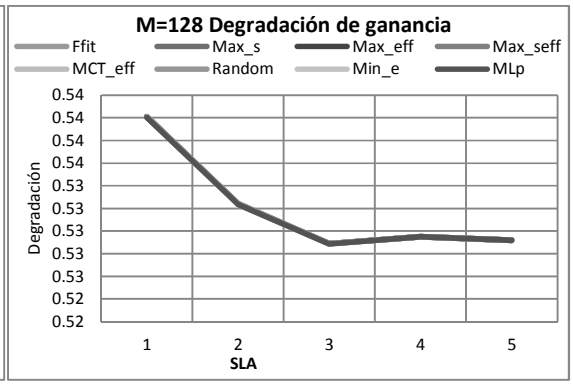
A3(e). Degradación de ganancia m=16



A3(f). Degradación de ganancia m=32

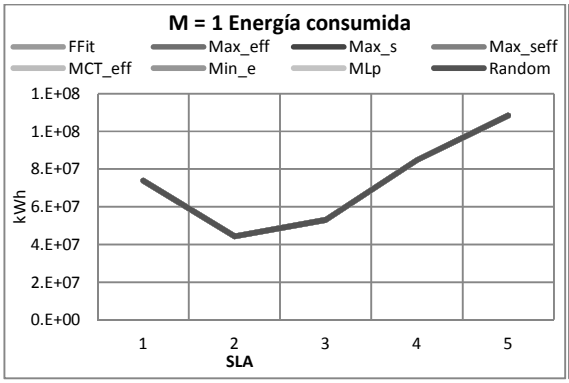


A3(g). Degradación de ganancia m=64

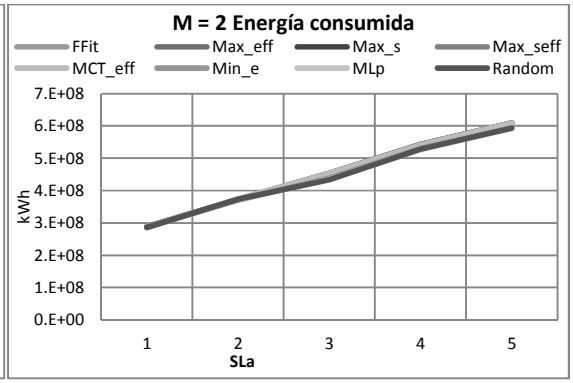


A3(h). Degradación de ganancia m=128

Figura A4. Consumo de energía, por cada escenario con diferente número de máquinas

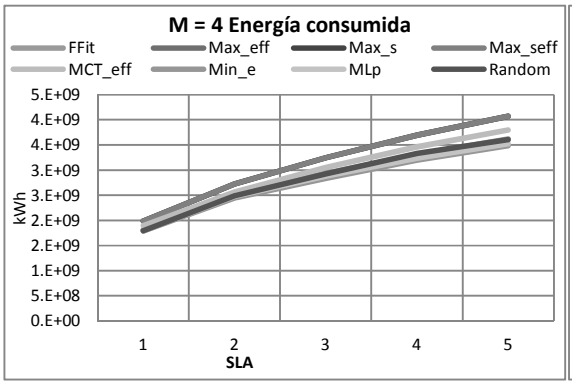


A4(a). Consumo total de energía m=1

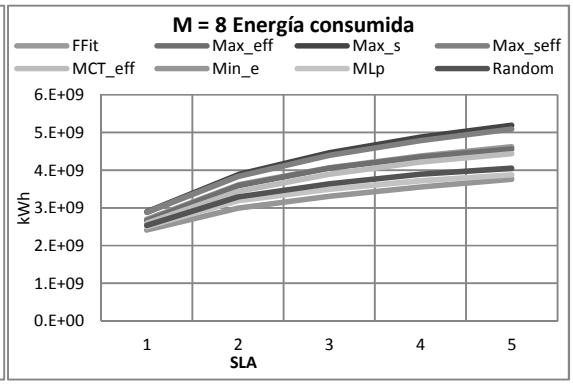


A4(b). Energía total consumida m=2

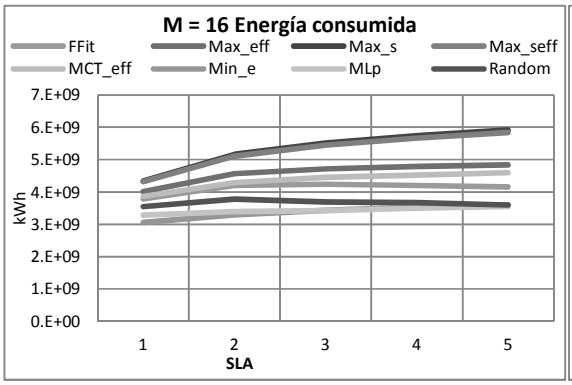
Figura A4. Consumo de energía, por cada escenario con diferente número de máquinas (continuación)



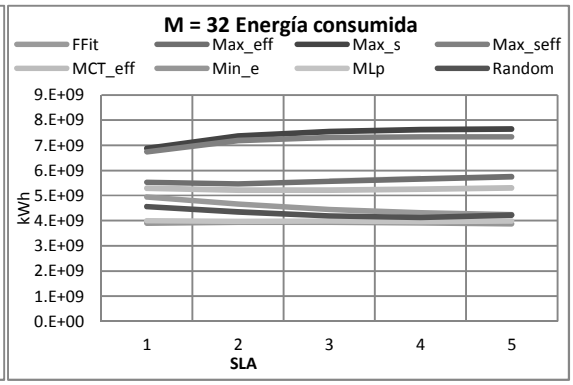
A4(c). Consumo total de energía m =4



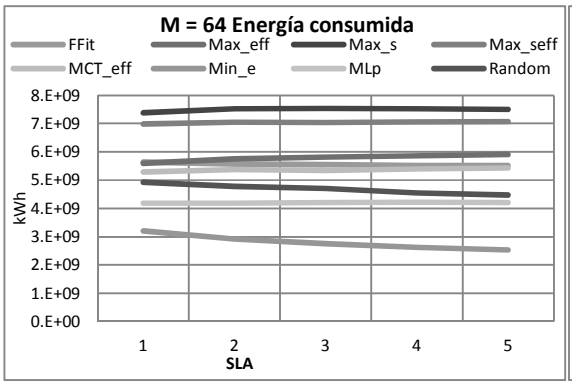
A4(d). Consumo total de energía m=8



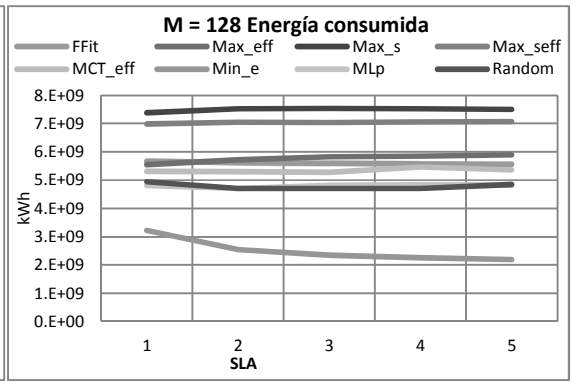
A4(e). Consumo total de energía m=16



A4(f). Consumo total de energía m=32

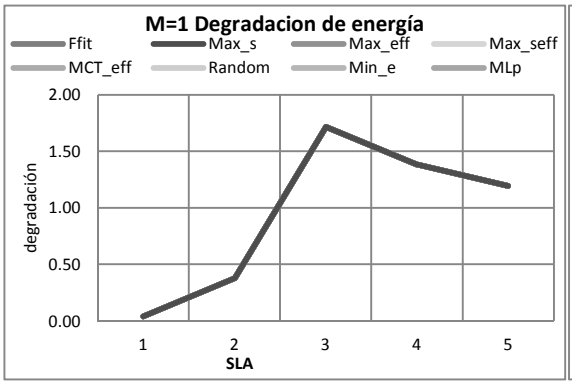


A4(g). Consumo total de energía m=64

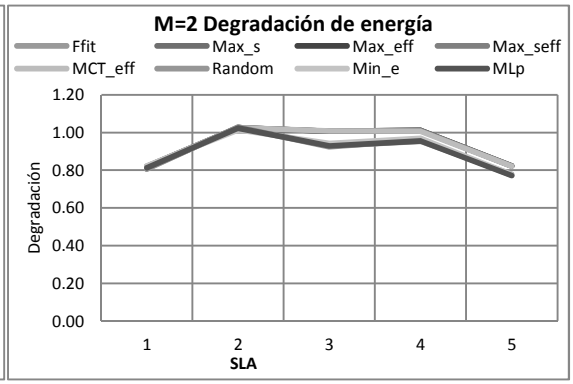


A4(h). Consumo total de energía m=128

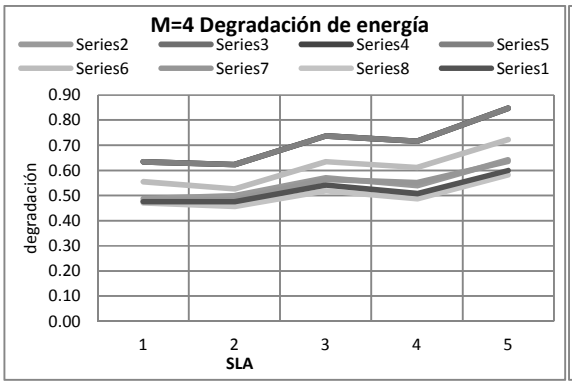
Figura A5. Degradación del consumo de energía, por cada escenario con diferente número de máquinas



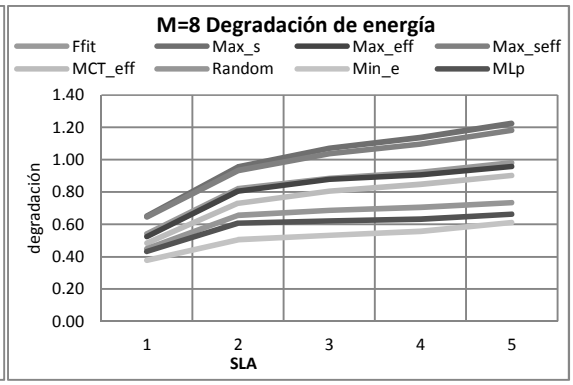
A5(a). Degradación del consumo energético m=1



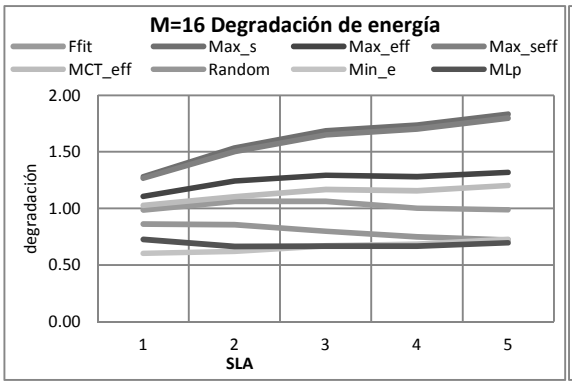
A5(b). Degradación del consumo energético m=2



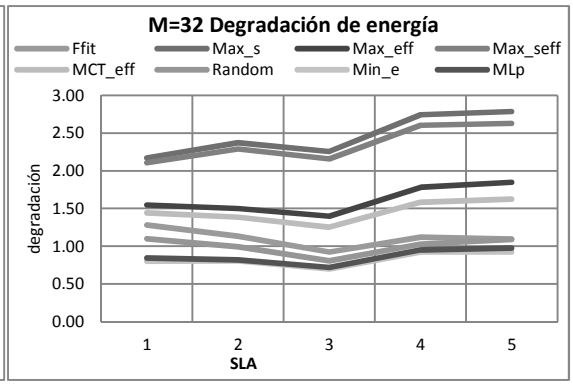
A5(c). Degradación del consumo energético m=4



A5(d). Degradación del consumo energético m=8

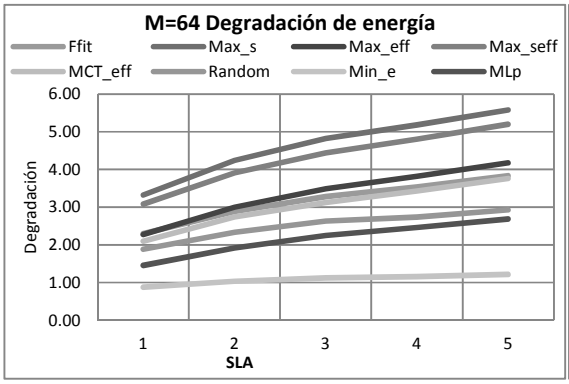


A5(e). Degradación del consumo energético m=16

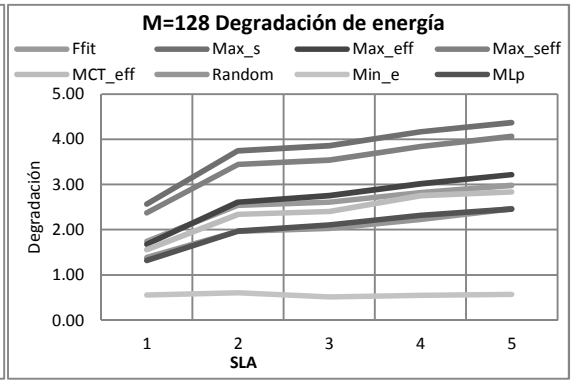


A5(f). Degradación del consumo energético m=32

Figura A5. Degradación del consumo de energía, por cada escenario con diferente número de máquinas (continuación)



A5(g). Degradación del consumo energético m=64



A5(h). Degradación del consumo energético m=128