Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Ciência da Computação

Tese de Doutorado

Gerenciamento de uma Infraestrutura Híbrida de TI Dirigido por Métricas de Negócio

Paulo Ditarso Maciel Júnior

Campina Grande, Paraíba, Brasil
Junho de 2013

Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Ciência da Computação

Gerenciamento de uma Infraestrutura Híbrida de TI Dirigido por Métricas de Negócio

Paulo Ditarso Maciel Júnior

Tese submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande - Campus I como parte dos requisitos necessários para obtenção do grau de Doutor em Ciência da Computação.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Metodologia e Técnicas da Computação

Francisco Vilar Brasileiro, Ph.D.
(Orientador)

Campina Grande, Paraíba, Brasil

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

M152g Maciel Júnior, Paulo Ditarso.
Gerenciamento de uma infraestrutura híbrida de TI dirigido por métricas de negócio / Paulo Ditarso Maciel Júnior. – Campina Grande, 2013.
98 f. : il.

Tese (Doutorado em Ciência da Computação) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática.

"Orientação: Prof. Dr. Francisco Vilar Brasileiro".
Referências.

1. Sistemas Distribuídos. 2. Grades Computacionais. 3. Computação na Nuvem. I. Brasileiro, Francisco Vilar. II. Título.

CDU 004.03 (043)

"GERENCIAMENTO DE UMA INFRAESTRUTURA HÍBRIDA DE TI DIRIGIDO POR MÉTRICAS DE NEGÓCIO"

PAULO DITARSO MACIEL JUNIOR

TESE APROVADA EM 14/06/2013

FRANCISCO VILAR BRASILEIRO, Ph.D, UFCG
Orientador(a)

RAQUEL VIGOLVINO LOPES, D.Sc, UFCG
Examinador(a)

JOSÉ ANTÃO BELTRÃO MOURA, Ph.D, UFCG
Examinador(a)

ANTÔNIO TADEU AZEVEDO GOMES, Dr., LNCC
Examinador(a)

LUCIANO PASCHOAL GASPARY, Dr., UFRGS
Examinador(a)

CAMPINA GRANDE - PB

Resumo

Com o surgimento do paradigma de *computação na nuvem* e a busca contínua para reduzir o custo de operar infraestruturas de Tecnologia da Informação (TI), estamos vivenciando nos dias de hoje uma importante mudança na forma como estas infraestruturas estão sendo montadas, configuradas e gerenciadas. Nesta pesquisa consideramos o problema de gerenciar uma infraestrutura híbrida, cujo poder computacional é formado por máquinas locais dedicadas, máquinas virtuais obtidas de provedores de computação na nuvem e máquinas virtuais remotas disponíveis a partir de uma grade *peer-to-peer* (P2P) *best-effort*. As aplicações executadas nesta infraestrutura são caracterizadas por uma *função de utilidade* no tempo, ou seja, a utilidade produzida pela execução completa da aplicação depende do tempo total necessário para sua finalização. Tomamos uma abordagem dirigida a negócios para gerenciar esta infraestrutura, buscando maximizar o lucro total obtido. Aplicações são executadas utilizando poder computacional local e da grade *best-effort*, quando possível. Qualquer capacidade extra requerida no intuito de melhorar a lucratividade da infraestrutura é adquirida no mercado de computação na nuvem. Também assumimos que esta capacidade extra pode ser reservada para uso futuro através de contratos de curta ou longa duração, negociados sem intervenção humana. Para contratos de curto prazo, o custo por *unidade* de recurso computacional pode variar significativamente entre contratos, com contratos mais urgentes apresentando, geralmente, custos mais caros. Além disso, devido à incerteza inerente à grade *best-effort*, podemos não saber exatamente quantos recursos serão necessários do mercado de computação na nuvem com certa antecedência. Superestimar a quantidade de recursos necessários leva a uma reserva maior do que necessária; enquanto subestimar leva à necessidade de negociar contratos adicionais posteriormente. Neste contexto, propomos heurísticas que podem ser usadas por agentes planejadores de contratos no intuito de balancear o custo e a utilidade obtida na execução das aplicações, com o objetivo de alcançar um alto lucro global. Demonstramos que a habilidade de estimar o comportamento da grade é uma importante condição para estabelecer contratos que produzem alta eficiência no uso da infraestrutura híbrida de TI.

Palavras-chave: Sistemas Distribuídos, Grades Computacionais e Computação na Nuvem.

Abstract

With the emergence of the cloud computing paradigm and the continuous search to reduce the cost of running Information Technology (IT) infrastructures, we are currently experiencing an important change in the way these infrastructures are assembled, configured and managed. In this research we consider the problem of managing a hybrid high-performance computing infrastructure whose processing elements are comprised of in-house dedicated machines, virtual machines acquired from cloud computing providers, and remote virtual machines made available by a best-effort peer-to-peer (P2P) grid. The applications that run in this hybrid infrastructure are characterised by a utility function: the utility yielded by the completion of an application depends on the time taken to execute it. We take a business-driven approach to manage this infrastructure, aiming at maximising the total profit achieved. Applications are run using computing power from both in-house resources and the best-effort grid, whenever possible. Any extra capacity required to improve the profitability of the infrastructure is purchased from the cloud computing market. We also assume that this extra capacity is reserved for future use through either short or long term contracts, which are negotiated without human intervention. For short term contracts, the cost per unit of computing resource may vary significantly between contracts, with more urgent contracts normally being more expensive. Furthermore, due to the uncertainty inherent in the best-effort grid, it may not be possible to know in advance exactly how much computing resource will be needed from the cloud computing market. Overestimation of the amount of resources required leads to the reservation of more than is necessary; while underestimation leads to the necessity of negotiating additional contracts later on to acquire the remaining required capacity. In this context, we propose heuristics to be used by a contract planning agent in order to balance the cost of running the applications and the utility that is achieved with their execution, with the aim of producing a high overall profit. We demonstrate that the ability to estimate the grid behaviour is an important condition for making contracts that produce high efficiency in the use of the hybrid IT infrastructure.

Keywords: Distributed Computing, Grid Computing and Cloud Computing.

Agradecimentos

À minha família e todos os amigos que, direta ou indiretamente, torceram por mim e me ajudaram na elaboração deste trabalho.

Ao meu orientador Fubica pelos ensinamentos e, acima de tudo, pela paciência e perseverança em me orientar. Aos demais professores e funcionários da UFCG com quem tive contato, pela a oportunidade de conviver e aprender bastante. Aos colegas de IFPB, com quem tive a oportunidade de trabalhar, pela infinita fonte de ajuda.

Aos inúmeros amigos que fiz no LSD, pelos momentos mais prazerosos de toda essa minha jornada acadêmica. Vou evitar falar nomes para não ser injusto e esquecer de alguém, mas não poderia deixar de destacar que João Arthur, Marcus Carvalho e Nazareno Andrade foram três irmãos que ganhei ao longo desse período.

Ao Governo Brasileiro e agências de fomento CNPq e CAPES por financiarem os meus estudos.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Motivação

Um novo modelo de negócio, que já é uma realidade atualmente, está mudando a forma com a qual recursos e serviços de Tecnologia da Informação (TI) são oferecidos. Vivenciamos uma tendência de "TI como um serviço", onde a aquisição dos recursos de TI é realizada quando necessária e de qualquer lugar, e a cobrança se dá a partir da quantidade de recursos (ou serviços) que são realmente utilizados. À medida que a necessidade por serviços de computação aumenta a cada dia, o mercado de computação utilitária (do inglês *utility computing*) já é uma realidade para diversos departamentos de TI atualmente. O principal argumento daqueles que proveem este tipo de solução é a possibilidade da redução substancial do custo total de propriedade (*Total Cost of Ownership* - TCO) da infraestrutura de TI dos seus clientes. Este modelo de TI oferecida como um serviço tem sido chamado de *Computação na Nuvem* (do inglês *Cloud Computing*). Como evidência dessa tendência, está previsto que até 2015 mais de 75% das infraestruturas de TI de médias e grandes organizações serão adquiridos como serviço de provedores externos [Gartner Research 2007].

Vantagens econômicas citadas acima certamente desempenham um papel fundamental na adoção deste modelo de negócio, mas existem outros fatores importantes a serem considerados. Portanto, a adoção deste novo modelo deve ser gradual e planejada, para evitar possíveis obstáculos. Além disso, pelo menos parte da infraestrutura computacional desenvolvida localmente de uma empresa deverá provavelmente ser mantida em um primeiro passo, ao invés de executar toda sua computação em provedores de computação na nuvem. Como exemplos

hipotéticos destes cenários, apresentamos as seguintes situações. A retenção de alguma capacidade local pode servir para atenuar os efeitos das flutuações dos preços geradas a partir de uma instabilidade passageira no mercado de computação na nuvem. A migração dos serviços suportados por uma infraestrutura de TI dedicada para um provedor terceirizado pode sofrer uma forte resistência de funcionários internos. Para certos tipos de aplicações, a migração para provedores externos pode não apresentar um redução significativa nos custos. Ou ainda, pode ser desejado por algumas organizações que determinados tipos de aplicações sejam executadas na sua própria infraestrutura, o que pode incluir aplicações críticas para o negócio ou aplicações que processam dados sigilosos. Consequentemente, diversas companhias estão adotando uma estratégia na qual a capacidade da infraestrutura local é planejada para uma carga de trabalho base, e recursos ou serviços adicionais necessários para lidar com picos de carga de trabalho são adquiridos de provedores externos sob demanda. Esta abordagem tem sido chamada de *cloud-bursting* [InfoWorld 2010] [Marshall, Keahey e Freeman 2010] [Assuncao, Costanzo e Buyya 2009].

Contudo, o modelo baseado em computação utilitária não é a única alternativa para reduzir TCO. Dentre outras soluções possíveis, grades computacionais baseadas no modelo *peer-to-peer* (P2P) surgiram como uma forma simples de economia através da negociação de ciclos computacionais ociosos [Cirne et al. 2006]. Modelos de mercado baseiam-se na existência e eficiência de mecanismos financeiros, contábeis e de negociação de contratos. Entretanto, para diversos cenários em computação distribuída (e também fora da área da computação), a implementação de tais mecanismos é complexa, custosa ou ineficiente [Yochai 2004]. Por outro lado, nestas situações de complexidade, sistemas de compartilhamento podem ser eficientes, à medida que estes utilizam informações fracamente estruturadas e, portanto, de fácil obtenção. Além isso, podem fazer uso de mecanismos sociais para monitoração e execução de tarefas, e obter baixos custos marginais de transação [Yochai 2004]. Todavia, diferente das soluções baseadas em mercados, estes sistemas geralmente não oferecem garantias na qualidade do serviço provido. Não obstante, os mesmos têm sido utilizados com sucesso para aumentar a eficácia dos custos de infraestruturas de TI em diversos cenários, como por exemplo infraestruturas responsáveis pela execução de aplicações científicas [Brasileiro et al. 2007].

O modelo de *cloud-bursting* está se tornando bastante popular [InfoWorld 2010] e di-

versas organizações estão se movendo em direção a este tipo de arquitetura [Amazon AWS Case Studies]. Além disso, a capacidade ociosa da infraestrutura local pode ser utilizada para executar carga de trabalho em favor de outras organizações, em troca da possibilidade de usufruir da capacidade ociosa desta outra organização para executar tarefas locais em um instante de tempo futuro. Os diferentes componentes desta infraestrutura híbrida proverão diferentes garantias, das mais detalhadas restrições de qualidade de serviço, aos serviços oferecidos na base do "melhor esforço". Advogamos a ideia de que, em um futuro próximo, diversas infraestruturas de TI utilizarão não somente recursos providos por uma infraestrutura local dedicada, mas também recursos providos por provedores externos de computação na nuvem. Além do mais, esta infraestrutura híbrida pode tomar proveito de uma grade computacional P2P ao doar recursos ociosos para outros parceiros e receber de volta, quando há a necessidade de poder computacional maior do que oferecido localmente.

Esta conjectura é suportada pela nossa própria experiência com os usuários do *middleware OurGrid* (http://www.ourgrid.org/), que permite o desenvolvimento de grades P2P abertas. Este sistema é usado para fomentar a criação da comunidade OurGrid, que vem sendo utilizada como plataforma computacional em diferentes áreas de aplicação, incluindo engenharias, bioinformática, ciência da computação e aplicações financeiras [Araujo et al. 2005] [Wilter et al. 2005] [Duarte et al. 2006] [Osthoff et al. 2003]. Em particular, o OurGrid dá suporte ao trabalho cooperativo de uma comunidade de meteorologistas e hidrologistas, com representantes tanto na academia quanto em agências governamentais [Araujo et al. 2005] [Voorsluys et al. 2005]. Como resultado dessa cooperação, alguns membros dessa comunidade disponibilizam previsões diárias do tempo como um serviço público. A capacidade computacional necessária em tempos críticos (quando aplicações com restrições de tempo estão sendo executadas) é normalmente muito maior do que a capacidade necessária em outros tempos. Super provisionar a infraestrutura de TI destas agências públicas para lidar com a alta demanda em momentos críticos não é eficiente economicamente. Em geral, uma agência poderá ser capaz de obter alguma capacidade adicional do OurGrid a um custo insignificante em momentos críticos. Contudo, visto que o OurGrid trabalha em uma abordagem de "melhor esforço", a agência não pode confiar no OurGrid sempre que precisar de capacidade extra. Neste contexto, o poder computacional extra não obtido do OurGrid pode ser adquirido sob demanda a partir de provedores de computação na nuvem, prova-

velmente a um custo menor do que manter uma infraestrutura local dedicada para cumprir tais demandas. A presença da grade P2P então reduz a quantidade de poder computacional que uma agência necessitará comprar em tempos críticos. Dessa forma, uma infraestrutura híbrida de TI formada por todas as três fontes potenciais de recursos (local, provedor externo de computação na nuvem e uma grade P2P) é altamente desejada para este tipo de cenário.

Como muitos estudos na área de gerência de TI assumem que serviços são providos por somente uma destas fontes, e apenas recentemente cenários envolvendo *cloud-bursting* são considerados, questões interessantes de pesquisa surgem quando consideramos o ambiente híbrido descrito acima. Em contraste aos trabalhos existentes que focam no gerenciamento da infraestrutura do provedor de serviços na nuvem (por exemplo, [Popovici e Wilkes 2005]), neste trabalho consideramos o ponto de vista de um cliente de tais provedores que necessita executar aplicações científicas (*e-Science*). Mais especificamente, consideramos como este cliente (o gerente da infraestrutura híbrida) pode tirar o melhor proveito da capacidade local dedicada, ao mesmo tempo em que utiliza criteriosamente as outras duas fontes de recursos computacionais ou serviços, com o objetivo de executar aplicações do tipo *bag-of-tasks* [Cirne et al. 2006]. Dentro da grande área da gerência de TI, concentramos no aspecto de *planejamento* da infraestrutura híbrida [Thain, Tannenbaum e Livny 2005]; ou seja, dada a utilidade gerada por uma determinada aplicação e sua estimada carga de trabalho, como devemos planejar os contratos que serão estabelecidos com provedores de serviços na nuvem, visando balancear o custo de executar tal aplicação e a utilidade obtida por sua execução? Além disso, também investigamos como escalonar de forma eficiente os recursos, buscando amenizar os efeitos de decisões erradas tomadas a partir das incertezas durante a fase de planejamento; ou seja, dado o conjunto de recursos previamente planejados, como o escalonamento em tempo de execução pode minimizar os efeitos das incertezas inerentes ao sistema durante o planejamento?

1.2 Contribuições

Como principal contribuição deste trabalho, averiguamos a hipótese de que infraestruturas híbridas de TI podem tomar proveito do baixo custo da utilização de grades computacionais P2P, buscando maximizar a lucratividade através da execução de aplicações com restrições

de tempo em tais ambientes. De uma maneira geral, buscamos neste trabalho investigar estratégias que podem ser utilizadas por um gerente de uma infraestrutura de TI na hora de *planejar*, *reservar*, *alocar* e *investir* (ou *empregar*) os recursos provenientes do mercado de computação na nuvem. Advogamos que um bom planejamento dos recursos leva a um melhor cenário futuro de reserva de recursos; que por sua vez, pode ser melhorado a partir de alocações eficientes destes recursos.

De uma maneira geral, podemos destacar as seguintes contribuições alcançadas nesta pesquisa:

1. Modelamos o ambiente no qual a infraestrutura híbrida está inserida, em termos de seus componentes: a carga de trabalho submetida à infraestrutura híbrida e as funções de utilidade de suas aplicações; os recursos dedicados oferecidos pela infraestrutura local; o mercado de computação na nuvem através de seus provedores e modelos de tarifação; e a grade computacional P2P e sua respectiva qualidade de serviço.

2. Fornecemos fortes indícios de que um planejamento de capacidade eficiente pode lidar com incertezas durante a operação da infraestrutura híbrida, buscando reservar recursos futuros a preços mais acessíveis. Neste sentido, buscamos demonstrar que o planejamento de capacidade a longo prazo pode ser robusto a estas incertezas e indicar um cenário de reserva subótimo junto aos provedores na nuvem.

3. Investigamos como uma estratégia de alocação de recursos pode melhorar o lucro esperado a partir da fase de planejamento, através de um escalonamento eficiente dos recursos disponíveis em tempo de execução. Neste caso, podemos utilizar informações mais acuradas a respeito do sistema na hora de alocar efetivamente os recursos previamente reservados, buscando "corrigir" o efeito das incertezas na fase de planejamento.

4. Validamos a ideia de que a infraestrutura híbrida também pode tomar proveito da grade P2P com o intuito de reduzir os custo incorridos na execução de aplicações com restrições de tempo, envolvendo contratos de curta duração. Para isto, propomos heurísticas de planejamento de contratos com diferentes níveis de "confiança" na grade P2P *best-effort* no momento de estabelecer compromissos de curta duração com provedores de computação na nuvem.

Como resultado destas contribuições, apresentamos uma lista dos trabalhos publicados em veículos internacionais especializados:

- Alvaro Coêlho, Paulo Ditarso Maciel Jr., Flavio de Figueiredo, David Maia e Francisco Brasileiro. ***On the impact of choice in multi-service P2P grids***. In: BARTOLINI, C.; SAHAI, A.; SAUVé, J. P. (Ed.). Proceedings of BDIM 2008, 3rd IEEE/IFIP International Workshop on Business-Driven IT Management, April 7, 2008, Salvador, Brazil. [S.l.]: IEEE, 2008. p. 98-101. ISBN 978-1-4244-2191-6 [Coêlho et al. 2008].

- Álvaro Coêlho, Francisco Brasileiro e Paulo Ditarso Maciel Jr. ***Using heuristics to improve service portfolio selection in P2P grids***. In: Proceedings of the 11th IFIP/IEEE international conference on Symposium on Integrated Network Management. Piscataway, NJ, USA: IEEE Press, 2009. (IM?09), p. 438-444. ISBN 978-1-4244-3486-2. Disponível em: <http://dl.acm.org/citation.cfm?id=1688933.1689000> [Coêlho, Brasileiro e Maciel Jr. 2009].

- Marcus Carvalho, Renato Miceli, Paulo Ditarso Maciel Jr., Francisco Brasileiro e Raquel Lopes. ***Predicting the quality of service of a peer-to-peer desktop grid***. In: Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing. Washington, DC, USA: IEEE Computer Society, 2010. (CCGRID ?10), p. 649-654. ISBN 978-0-7695-4039-9. Disponível em: <http://dx.doi.org/10.1109/CCGRID.2010.50> [Carvalho et al. 2010].

- Raquel Vigolvino Lopes, Francisco Vilar Brasileiro e Paulo Ditarso Maciel Jr. ***Business-driven capacity planning of a cloud-based it infrastructure for the execution of web applications***. In: IPDPS Workshops. IEEE, 2010. p. 1-8. Disponível em: <http://dblp.uni-trier.de/db/conf/ipps/ipdps2010w.html# LopesBM10> [Lopes, Brasileiro e Maciel Jr. 2010].

- Paulo Ditarso Maciel Jr., Francisco Vilar Brasileiro, Raquel Vigolvino Lopes, Marcus Carvalho e Miranda Mowbray. ***Evaluating the impact of planning long-term contracts on the management of a hybrid IT infrastructure***. In: AGOULMINE, N. et al. (Ed.). Integrated Network Management. IEEE, 2011. p. 89-96. ISBN 978-1-

4244-9221-3. Disponível em: <http://dblp.uni-trier.de/db/conf/im/im2011.html# MacielBLCM11> [Maciel Jr. et al. 2011].

- Paulo Ditarso Maciel Jr., Flavio de Figueiredo, David Maia, Francisco Vilar Brasileiro e Álvaro Coêlho. ***On the planning of a hybrid IT infrastructure***. In: IEEE/IFIP Network Operations and Management Symposium: Pervasive Management for Ubioquitous Networks and Services, NOMS 2008, 7-11 April 2008, Salvador, Bahia, Brazil, p. 496-503. Disponível em: <http://dblp.unitrier.de/db/conf/noms/noms2008.html# MacielFMBC08> [Maciel Jr. et al. 2008].

- Paulo Ditarso Maciel Jr., Francisco Brasileiro, Ricardo Araújo Santos, David Candeia, Raquel Lopes, Marcus Carvalho, Renato Miceli, Nazareno Andrade, Miranda Mowbray. ***Business-driven short-term management of a hybrid IT infrastructure***. Journal of Parallel and Distributed Computing, Academic Press, Inc., Orlando, FL, USA, v. 72, n. 2, p. 106-119, fev. 2012. ISSN 0743-7315. Disponível em: <http://dx.doi.org/10.1016/j.jpdc.2011.11.001> [Maciel Jr. et al. 2012].

Primeiramente, realizamos estudos que caracterizaram o impacto da escolha dos serviços oferecidos em uma grade computacional P2P, visando uma maior colaboração entre os participantes do sistema. Estes estudos buscaram mostrar que uma escolha mais criteriosa de tais serviços pode favorecer uma maior interação entre pares mutuamente interessados. Os resultados foram publicados nos artigos [Coêlho et al. 2008] e [Coêlho, Brasileiro e Maciel Jr. 2009].

Também realizamos uma modelagem analítica da qualidade de serviço oferecida pelo OurGrid, gerando um modelo matemático que representa a quantidade de recursos que serão recebidos da grade computacional em um instante futuro de tempo. A eficiência do modelo de predição foi avaliada a partir de experimentos, através da comparação entre resultados obtidos com o modelo matemático e resultados obtidos via simulação, utilizando-se um registro de carga de trabalho de um sistema real disponível na Internet. A descrição detalhada do trabalho realizado e dos resultados obtidos foram publicados no artigo [Carvalho et al. 2010].

Caracterizamos o ambiente híbrido considerando contratos de longa duração com provedores de computação na nuvem e o resultado desta caracterização foi a elaboração de modelos analíticos que serviram para avaliar o ambiente proposto. Investigamos como planejar estes contratos visando a execução de aplicações e-Science. Ou seja, o gerenciamento da infraestrutura híbrida considera o planejamento de todos os contratos que deverão ser realizados com os provedores durante um ano a frente. Contratos a longo prazo são vantajosos tanto na perspectiva dos provedores, já que é uma forma de aumentar as garantias de operação estável permanente; quanto na perspectiva do cliente, já que o protege contra flutuações nos preços e indisponibilidade de recursos. A partir dos resultados obtidos, podemos dizer que temos indícios de que um planejamento a longo prazo da infraestrutura híbrida é essencial em termos de melhorar a lucratividade esperada. Estes resultados foram publicados nos artigos [Lopes, Brasileiro e Maciel Jr. 2010] e [Maciel Jr. et al. 2011].

Caracterizamos o ambiente híbrido para execução de aplicações e-Science considerando que contratos de curta duração (em torno de um dia) podem ser estabelecidos com provedores de computação na nuvem. Mais especificamente, propomos heurísticas que podem ser utilizadas por um agente planejador de contratos com um intuito de balancear os custos de rodar aplicações e-Science e os rendimentos obtidos pela execução das mesmas, produzindo, no geral, lucros substancialmente altos. Demonstramos que a habilidade de estimar o comportamento da grade computacional é uma condição importante para estabelecer contratos que produzem uma alta eficiência no uso da infraestrutura híbrida. Os resultados desta avaliação envolvendo contratos de curta duração com provedores de computação na nuvem podem ser encontrados nos artigos [Maciel Jr. et al. 2008] e [Maciel Jr. et al. 2012].

1.3 Estrutura da Tese

Nesta seção buscamos explicar a sequência lógica de realização do trabalho apresentada nos próximos capítulos.

Dividimos esta pesquisa em duas abordagens de gerenciamento, considerando um planejamento a longo e curto prazo. A partir do momento em que o esquema de reserva começou a ser empregado pelos provedores de computação na nuvem, decidimos investigar como o gerenciamento envolvendo esse tipo de interação entre clientes e provedores deve ser rea-

lizado. Nos preocupamos sempre em validar a ideia de que a infraestrutura híbrida de TI pode tomar proveito do baixo custo envolvido quando utilizamos uma grade P2P *best-effort*. Para tal, buscamos elaborar estratégias que visam um planejamento a longo prazo para a infraestrutura híbrida. Mais especificamente, com o gerenciamento a longo prazo buscamos demonstrar como gerentes de TI podem planejar infraestruturas híbridas de computação, a partir de diversas formas de aquisição de recursos e diferentes modelos de tarifação empregados no mercado de computação na nuvem atualmente.

Já no gerenciamento a curto prazo, propomos um novo modelo de tarifação que considera contratos de reserva de recursos a curto prazo. Este novo modelo de tarifação pode proporcionar uma maior flexibilidade na hora de estabelecer contratos com os provedores. Com o aumento na procura pela computação na nuvem e, consequentemente, na competição entre os diversos provedores, novos modelos como este se tornarão uma realidade neste tipo de mercado. É importante ressaltar que os modelos de negócio dos provedores de computação na nuvem vêm sendo rapidamente ampliados nos últimos anos e que este novo modelo de tarifação de curto prazo pode surgir como uma alternativa viável na oferta de novos serviços. Podemos dizer então que trabalho foi realizado sempre procurando responder as seguintes perguntas referentes à infraestrutura híbrida.

- *Vale a pena?*
- *Como devemos planejá-la?*
- *Podemos utilizar melhor os recursos?*
- *Podemos melhorar o cenário de reserva existente?*

Dessa forma, organizamos o restante deste trabalho de maneira a seguir.

Primeiramente, apresentamos uma revisão bibliográfica envolvendo o estado da arte para a realização deste trabalho no Capítulo 2.

No Capítulo 3 apresentamos um arcabouço conceitual com uma modelagem do ambiente híbrido proposto. Esta modelagem servirá para avaliarmos como diferentes formas de gerenciamento podem melhorar a eficiência, em termos econômicos, na operação da infraestrutura híbrida. A Seção 3.2 descreve o modelo analítico proposto para a infraestrutura híbrida, em

termos de seus componentes. Definimos, de uma forma macro, o problema de gerenciar o ambiente híbrido proposto na Seção 3.3.

No Capítulo 4 investigamos como o gerenciamento envolvendo diferentes modelos de tarifação entre clientes e provedores de computação na nuvem deve ser realizado. Mais especificamente, validamos a ideia de que um planejamento criterioso da infraestrutura híbrida pode alcançar uma maior lucratividade quando utilizamos contratos de longa duração com os provedores. A Seção 4.2 apresenta o modelo do sistema e formaliza o problema do gerenciamento a longo prazo de uma infraestrutura híbrida de TI. Devido a sua complexidade, dividimos o problema do gerenciamento em dois problemas menores: (i) planejamento da capacidade, cuja execução é realizada *off-line* e define os contratos a longo prazo que devem ser estabelecidos; e (ii) escalonamento em tempo de execução, que de fato realiza a alocação de partes da carga de trabalho nos recursos apropriados, buscando maximizar o lucro na operação da infraestrutura. Discutimos o planejamento da capacidade na Seção 4.3 e abordamos o problema do escalonamento na Seção 4.4. Na Seção 4.5 apresentamos uma avaliação de desempenho das estratégias propostas, com o intuito de promover a adoção de tais estratégias por gerentes que desejam maximizar o lucro com o planejamento e alocação de recursos de uma infraestrutura híbrida de TI. Por fim, apresentamos algumas considerações finais sobre o gerenciamento a longo prazo na Seção 4.6.

No Capítulo 5 buscamos responder a primeira pergunta levando em consideração compromissos através de contratos de curta duração entre provedores de computação na nuvem e clientes. Acreditamos que compromissos de curta duração proporcionam uma flexibilidade maior para os clientes deste mercado. Na Seção 5.2 definimos a aplicação e os elementos que compõem a infraestrutura híbrida de TI, e descrevemos formalmente o problema encontrado. Apresentamos heurísticas para o planejamento dos contratos na Seção 5.3, e uma avaliação destas heurísticas em termos de lucratividade na Seção 5.4. Apresentamos algumas considerações finais sobre o planejamento de contratos de curta duração na Seção 5.5.

No Capítulo 6 apresentamos as nossas considerações finais a respeito do trabalho realizado na Seção 6.1 e perspectivas de trabalhos futuros na Seção 6.2.

Por fim, no Apêndice A apresentamos um sumário da notação utilizada nos modelos propostos.

Capítulo 2

Revisão Bibliográfica

Apresentamos neste capítulo uma revisão bibliográfica envolvendo os assuntos estudados para a realização deste trabalho. Os trabalhos foram divididos em algumas áreas como indicado abaixo, mas podemos primeiramente destacar dois artigos básicos para a fundamentação teórica desta tese. Vaquero et al. [Vaquero et al. 2008] apresentam alguns conceitos básicos de computação na nuvem e as principais características desse paradigma. Além disso, os autores ressaltam as características em comum entre a área de grades computacionais e computação na nuvem, bem como as diferenças mais importantes. Já Galante e Bona [Galante e Bona 2012] investigam o estado da arte de uma das principais características da computação na nuvem, a "elasticidade". Os autores investigam as principais soluções encontradas no mercado atualmente e apresentam uma classificação baseada nas principais características levantadas. Ademais, os autores apresentam alguns dos principais trabalhos acadêmicos encontrados nesta área.

Planejamento de Capacidade e Escalonamento de Recursos

Como mencionado anteriormente, *Cloud-burst* significa o desenvolvimento dinâmico de uma infraestrutura de serviço (software/hardware) que opera em recursos internos de uma organização e provedores de computação na nuvem, para resolver o problema de picos na demanda computacional. Diversos trabalhos foram propostos nesta direção, tais como os apresentados por Marshall, Keahey e Freeman [Marshall, Keahey e Freeman 2010] e Assunção, Costanzo e Buyya [Assunção, Costanzo e Buyya 2009], onde as estratégias de escalonamento buscam reduzir o tempo de execução das aplicações através da aquisição de recursos adicionais a

partir de um provedor na nuvem. Em outras palavras, eles apresentam diferentes estratégias que aumentam a capacidade total disponível para executar uma aplicação, acrescentando aos recursos locais aqueles adquiridos sob demanda de provedores de computação na nuvem. No geral, o processo de aquisição dos recursos adicionais é desencadeado a partir de variações na carga de trabalho computacional.

Existem algumas similaridades entre o trabalho desenvolvido nesta pesquisa e o trabalho proposto por Popovici e Wilkes [Popovici e Wilkes 2005], e Yu, Buyya e Tham [Yu, Buyya e Tham 2005], bem como alguns pontos onde nosso trabalho complementa os mesmos. Popovici e Wilkes concentraram-se na operação de um provedor de serviços, enquanto Yu, Buyya e Tham concentraram-se na interação entre um cliente e um provedor de uma grade de serviço. Nosso trabalho, por outro lado, investiga a interação de um cliente com provedores de serviços, uma grade computacional P2P e um provedor de computação na nuvem.

Popovici e Wilkes [Popovici e Wilkes 2005] propõem uma abordagem orientada a mercado para gerenciar um provedor de serviços visando resolver o problema de quais requisições (clientes) o provedor deve aceitar. Eles estendem o trabalho desenvolvido por Chun e Culler [Chun e Culler 2002], e Irwin, Grit e Chase [Irwin, Grit e Chase 2004], levando em consideração um provedor que oferece serviços baseado em *jobs* (tarefas computacionais) para seus clientes, e aluga sua infraestrutura de um provedor especializado em prover recursos. A dificuldade em selecionar as requisições está na suposição de que o provedor de serviços não tem completa certeza da disponibilidade dos recursos necessários para cumprir as requisições. Os autores definem heurísticas tanto para o controle de admissão quanto para o escalonamento, que levam em consideração os riscos da incerteza na disponibilidade dos recursos e buscam maximizar o lucro obtido pelo provedor de serviços. Neste trabalho consideramos um agente planejador de contratos para uma infraestrutura híbrida de TI, e as incertezas surgem a partir da carga de trabalho a ser executada, da natureza "melhor-esforço" da grade P2P e das variações no mercado de computação na nuvem.

Yu, Buyya e Tham [Yu, Buyya e Tham 2005] propõem um algoritmo de escalonamento que minimiza os custos de execução de *workflows* (sequências de tarefas conectadas) em uma grade, considerando os prazos especificados pelos usuários. A instabilidade da grade traz a incerteza no ambiente. De fato, esta incerteza é tratada com o re-escalonamento das tarefas que falham, movendo-as para outros computadores disponíveis na grade. Predições da

disponibilidade e informações sobre os custos dos recursos contratados da grade são usadas para selecionar os contratos mais apropriados e para minimizar os custos. Nosso trabalho usa uma estratégia similar, mas aplica esta estratégia em um contexto diferente.

Zhu e Agrawall desenvolveram métodos para alocação dinâmica de recursos provenientes de grades [Zhu e Agrawal 2009] e da nuvem [Zhu e Agrawal 2012], cuja abordagem geral não é específica para nenhum destes ambientes, mas podem ser utilizados também em ambientes híbridos. Os métodos propostos para alocação de recursos em tempo real não apenas se adaptam à falha de tarefas alocadas em recursos não confiáveis, como também à mudança de requisitos imposta pela atualização em componentes de serviços. O principal objetivo dos autores é otimizar uma "função de benefício independente do tempo", ao mesmo tempo em que o *deadline* e (opcionalmente) um limite financeiro são respeitados. Os autores apresentam resultados da aplicação dos métodos propostos em aplicações adaptativas reais. Nosso trabalho é mais específico do que o proposto pro Zhu e Agrawall uma vez que assumimos que os requisitos dos recursos para a aplicação são conhecidos de antemão; entretanto, como buscamos maximizar a lucratividade ao invés do benefício, nossa abordagem pode ser capaz de encontrar soluções para alocação dos recursos que levam a maiores lucros.

Restrições de Tempo e Funções de Utilidade

Neste trabalho, estamos interessados particularmente em aplicações científicas que possuam restrições de tempo para serem executadas. Existem vários exemplos de aplicações distribuídas nos quais os clientes (ou usuários) necessitam de garantias nas alocações de recursos e no tempo de resposta de suas execuções. Exemplos de áreas de domínio destes tipos de aplicações incluem: medicina remota [Camorlinga e Schofield 2006]; controle em tempo real de instrumentos e fluxo de tráfego aéreo [McGough et al. 2007] [Weigang, Dib e Cardoso 2004]; processamento de tráfego multimídia e *streaming* de vídeo [Kuntschke et al. 2006] [Sawchuk et al. 2003]; previsão do tempo [Plale et al. 2005] [Araujo et al. 2005]; e experimentos de e-Science, tais como os realizados na física de alta energia e astronomia [Plaza et al. 2006] [CERN-LHCC-02-26]. Alguns esforços foram direcionados para o suporte à execução de aplicações distribuídas com restrições de tempo [Zhang e Sivasubramaniam 2001] [Funk, Goossens e Baruah 2001], e para melhorar as garantias de qualidade do serviço oferecido por sistemas de computação distribuída de alto desempenho através de

reserva antecipada de recursos (*advance reservations*) [Farooq, Majumdar e Parsons 2007] [Farooq, Majumdar e Parsons 2006] [Foster e Roy 2000]. Contudo, é necessário um melhor entendimento dos requisitos de qualidade de serviço destas aplicações com restrições de tempo.

Outro problema é a falta de conhecimento em como gerar funções que expressam a "utilidade" que os clientes esperam receber de suas aplicações dado o tempo que elas levam para serem executadas. Estas são as chamadas *Funções de Utilidade* (do inglês, *Utility Functions*). Alguns trabalhos têm sido desenvolvidos para determinar estas funções, assumindo que a utilidade das aplicações é geralmente representada por uma função degrau ou de decaimento linear com o tempo [Mills e Dabrowski 2008] [Yu, Buyya e Tham 2005] [Popovici e Wilkes 2005] [Irwin, Grit e Chase 2004].

Aplicações Científicas na Nuvem

Muitos trabalhos concentram-se em características específicas da aplicação, como por exemplo, o escalonamento de workflows [Pandey et al. 2010] ou aplicações *bag-of-tasks* (onde as tarefas são completamente independentes umas das outras) [Silva, Veiga e Ferreira 2008]; enquanto outros trabalhos concentram-se em propor infraestruturas para IaaS (*Infrastructure-as-a-Service*) [Maciel Jr. et al. 2012] ou SaaS (*Software-as-a-Service*) [Kim et al. 2009].

Pandey et al. [Pandey et al. 2010] apresentam e avaliam a heurística *Particle Swarm Optimization* para o escalonamento de *workflow* de aplicações científicas no ambiente de computação na nuvem. A heurística proposta emprega conceitos simples de algoritmos genéticos, e o seu principal objetivo é minimizar o custo total da execução do workflow. Os autores consideram tanto os custos de execução quanto os de transmissão de dados. Os resultados apresentados demonstram uma melhoria se comparados com estratégias convencionais tais como: *earliest-finish-time*, *earliest-starting-time* ou *highest-processing-capabilities*.

Silva, Veiga e Ferreira [Silva, Veiga e Ferreira 2008] consideram aplicações *bag-of-tasks* de curta duração e propõem uma heurística para escalonar estas tarefas em provedores na nuvem. A heurística considera uma verba limitada para o escalonamento das tarefas, bem como uma abordagem que busca agilizar a execução das aplicações. Além disso, a heurística tenta encontrar o número ideal de instâncias no provedor na nuvem, dado que os tempos de execução das tarefas não são conhecidos previamente. Os resultados mostram que a

quantidade de instâncias no provedor de serviço é próxima da quantidade para o caso onde esses tempos são conhecidos.

Kim et al. [Kim et al. 2009] propõem uma infraestrutura similar àquela proposta em [Maciel Jr. et al. 2012], onde os recursos são formados por uma infraestrutura local dedicada, instâncias em provedores de computação na nuvem e recursos obtidos de uma grade computacional. O ambiente é proposto para dar suporte a uma comunidade de médicos que compartilham imagens a partir de diferentes grupos de pesquisa. Duas estratégias de escalonamento são propostas, a primeira busca finalizar as aplicações tão logo possível, enquanto a segunda busca melhorar o tempo de resposta sem exceder a verba previamente direcionada.

Modelos de Tarifação e Mercado de Computação Utilitária

Provedores de computação na nuvem necessitam de modelos de negócio robustos. Rappa [Rappa 2004] apresenta uma visão geral do que o modelo de negócio para o mercado de computação na nuvem deve conter, levando em consideração características tais como: necessidade, confiabilidade, usabilidade e escalabilidade. Este modelo é semelhante aos modelos de negócio utilizados no provimento de serviços públicos como água, telefone, acesso à Internet e eletricidade. Os modelos de tarifação para recursos de computação na nuvem utilizados neste trabalho são baseados nos modelos reais empregados pelos grandes provedores atualmente em operação, onde o preço final é formado por um preço da reserva do recurso mais o preço pelo uso efetivo deste.

Buyya et al. [Buyya et al. 2002] indicam alguns modelos econômicos para ajustar os preços dos serviços baseado na oferta e demanda, que incluem mercado de *commodities*, preço fixo e leilões. Os autores descrevem a arquitetura de um sistema e políticas para o gerenciamento dos recursos em infraestruturas de grades computacionais, que levam em consideração os vários modelos de tarifação possíveis. No nosso trabalho, assumimos um modelo de tarifação genérico onde o custo de um ciclo computacional[1] em um provedor é composto de uma taxa de reserva mais uma taxa de consumo. Diferentes perfis de provedores podem ser mapeados em diferentes taxas de reserva e consumo. Detalhes do modelo de

[1]Um ciclo computacional é definido neste trabalho como uma unidade de computação por unidade de tempo, podendo ser mapeado para diferentes unidades empregadas como, por exemplo, uma instância virtual, um nó em um *cluster* ou em uma grade, um núcleo processador em um desktop, etc.

tarifação são apresentados nos próximos capítulos deste documento.

Diferente da abordagem que tomamos no nosso trabalho, Rogers e Cliff [Rogers e Cliff 2012] propõem uma solução que busca otimizar os recursos para o provedor de computação na nuvem. Baseada em um componente externo (*broker*), a solução proposta busca atender as demandas dos clientes ao mesmo tempo em que otimiza o provisionamento de recursos dos provedores de serviço. Como especificado acima, o foco do nosso trabalho é buscar soluções de gerenciamento que ajudem clientes de provedores na nuvem a planejarem suas infraestruturas de maneira mais econômica. Contudo, embora não seja foco do nosso trabalho uma investigação envolvendo a perspectiva dos provedores, acreditamos que soluções envolvendo as duas abordagens podem ser interessantes como trabalhos futuros.

Embora os trabalhos mencionados acima sejam relacionados com o desenvolvido nesta pesquisa, acreditamos que o nosso é o primeiro que propõe a utilização de uma grade P2P como uma alternativa economicamente viável para o provisionamento de recursos em uma infraestrutura híbrida, dentro do contexto da execução de aplicações *bag-of-tasks* com carga de trabalho variável e restrições de tempo. Nosso trabalho, diferente dos apresentados neste capítulo, investiga aspectos de gerenciamento desta infraestrutura híbrida a partir da incerteza na quantidade necessária de recursos, objetivando maximizar o lucro na operação da mesma. Outro aspecto que diferencia o nosso trabalho dos demais apresentados é com relação ao escopo do gerenciamento empregado. Enquanto outros trabalhos apresentam apenas estratégias de gerenciamento de curto prazo, neste trabalho apresentamos estratégias que podem ser empregadas tanto para o gerenciamento a curto, quanto a longo prazo. Por fim, pelo melhor do nosso conhecimento, fomos os primeiros a apresentar um modelo de reserva para o mercado de computação na nuvem, e sugerir uma alternativa desta modalidade para contratos de curta duração.

Capítulo 3

Arcabouço Conceitual

3.1 Introdução

De acordo com Nicholas Carr [Carr 2008], é improvável que os departamentos de TI das corporações sobrevivam por muito tempo, pelo menos na sua forma habitual que conhecemos. Companhias oferecendo *computação utilitária* estão substituindo os departamento de TI, mais ou menos como aconteceu com empresas de energia elétrica no início do século XX. Nesta direção, estamos vivenciando um rápido crescimento no esforço de estabelecer um mercado de computação utilitária *"na nuvem"*, com diversas companhias comerciais provendo atualmente recursos virtuais sob demanda, tais como infraestrutura, armazenamento de dados e serviços de *software*. No contexto da computação na nuvem, o termo "serviço" pode representar qualquer tipo de recurso, desde infraestrutura física para execução de aplicações, a serviços viabilizados via *software*.

Neste ponto, é importante trazermos uma breve discussão sobre os conceitos da *computação utilitária* e da *computação na nuvem*. Apesar do conceito de computação utilitária ter surgido antes, consideramos que a computação na nuvem é um termo mais abrangente e que a computação utilitária pode ser vista, em alguns casos, como uma implementação da computação na nuvem [Hwang, Dongarra e Fox 2011]. De uma maneira bem simples, a computação utilitária envolve o pagamento sob demanda de recursos computacionais tais como *hardware*, *software* ou largura de banda. Ou seja, o que antes era visto como produto, agora é oferecido como serviço. Obviamente, computação utilitária pode ser implementada sem computação nuvem (e vice-versa). Como exemplo, podemos citar um supercomputa-

dor que tem o seu processamento "fatiado" entre vários usuários, mas não necessariamente: utiliza virtualização de recursos; oferece acesso de qualquer lugar; provê alta escalabilidade e elasticidade dos recursos; etc. Entretanto, por simplicidade, consideramos os dois termos permutáveis deste ponto em diante no texto.

Como mencionamos no Capítulo 1, aqueles que promovem este tipo de mercado advogam que os benefícios de uma abordagem de computação na nuvem incluem a flexibilidade para aumentar e reduzir capacidades à vontade, pagando-se apenas pelos recursos realmente utilizados. Além disso, a economia de escala em operar grande centros de dados significa que é possível aos provedores de serviços oferecerem alguns tipos de recursos mais baratos do que se fossem providos localmente nas organizações individuais. Estes fatores juntos podem resultar em reduções substanciais nos custos de operação de uma infraestrutura de TI. Por outro lado, buscar o modelo de computação utilitária não é a única forma de reduzir custos. Outra possibilidade de adquirir recursos computacionais extras a um baixo custo é através de grades computacionais P2P, como sugerido por Cirne et al.[Cirne et al. 2006]. Este tipo de sistema possibilita uma economia simples para a negociação de recursos ociosos. Entretanto, sistemas P2P geralmente se baseiam no comportamento recíproco de seus participantes, e normalmente não oferecem indicações sobre a qualidade do serviço que será provido. Diferentemente, provedores de computação utilitária tipicamente oferecem garantias de qualidade do serviço (como por exemplo, disponibilidade de pelo menos 99.9% do tempo ao longo de um determinado período), e recompensam os clientes caso estas garantias não sejam cumpridas.

No geral, proprietários e gerentes de TI podem tomar vantagem a partir de uma infraestrutura híbrida que se beneficia de ambas as estratégias para aquisição de recursos. Neste sistema híbrido, poder computacional pode ser obtido não apenas localmente em uma infraestrutura dedicada, mas também através de provedores de computação na nuvem e recursos recebidos de uma grade P2P "best-effort". Salientamos que a grade P2P utiliza um mecanismo de alocação de recurso no qual, quanto mais recursos são doados para a grade por um *peer*, mais este *peer* pode esperar receber de volta da grade no futuro. Uma instância real desse tipo de grade é o *OurGrid*, que será modelado nos próximos capítulos desta tese. Contudo, entendemos que os modelos apresentados ao longo deste trabalho são suficientemente genéricos, e poderiam ser estendidos para outros tipos de grades computacionais, como por

exemplo, grades oportunistas. Para isto, basta uma caracterização apropriada da quantidade de recursos esperada do sistema em um determinado intervalo de tempo.

No mercado de computação na nuvem, a reserva de recursos pode ser a forma adequada de mitigar os riscos enfrentados por ambos, provedores e clientes. Enquanto provedores enfrentam o risco de operarem suas infraestruturas com capacidade sub-utilizada, clientes enfrentam o risco de, quando necessário, não ter os recursos disponíveis a preços razoáveis. Assim, contratos de reserva são vantajosos tanto na perspectiva dos provedores, já que é uma forma de aumentar as garantias de operação estável permanente; quanto na perspectiva do cliente, já que serve como uma proteção contra flutuações nos preços e indisponibilidade de recursos. Nesta direção, em adição aos dois modelos distintos de provisionamento de recursos sob demanda (*On-demand* e *Spot*), a *Amazon.com Inc.* (o maior provedor de IaaS no mercado atualmente) oferece a opção de reserva de recursos para o seu serviço de computação na nuvem *Elastic Compute Cloud* (EC2). Através deste serviço, recursos podem ser reservados por longos períodos de tempo (pelo menos um ano de duração). Usuários pagam uma taxa única para a reserva de recursos e são cobrados pela utilização dos recursos a uma taxa reduzida, quando comparada a outros modelos de tarifação oferecidos pela *Amazon.com* [Amazon AWS 2010] [Vogels 2009]. Em suma, estes modelos são diferenciados pelo esquema de tarifação utilizado e sua qualidade de serviço associada:

1. *On-demand market.* Este primeiro modelo permite a alocação de recursos dedicados sob demanda a qualquer momento, sujeita apenas à atual disponibilidade de recursos do provedor. O usuário (cliente) é cobrado apenas pelo tempo que ele realmente utilizou o recurso.

2. *Spot market.* O segundo modelo é baseado em lances de oferta pelos recursos. Um recurso é instanciado (alocado) para uso quando o lance ofertado pelo recurso é maior do que um preço mínimo estabelecido pelo provedor de IaaS. Este preço é variável a partir do nível de disponibilidade do provedor. O cliente é cobrado pelo valor atual estabelecido, que é tipicamente menor do que o preço cobrado no *On-demand market.*

3. *Reservation market.* O último modelo de aquisição de recursos permite a reserva de uma instância (recurso) por um longo período de uso (tipicamente um ou mais anos) através do pagamento de uma taxa inicial de reserva. Quando instâncias previamente

reservadas são utilizadas, o cliente é cobrado pelo uso dos recursos a uma taxa inferior, quando comparada aos outros esquemas de tarifação.

Podemos notar que a *Amazon.com* representando um modelo de sucesso de computação na nuvem, oferece uma boa variedade nos seus modelos de tarifação, o que pode ser crucial para oferecer o esquema que mais se adequa a cada tipo de cliente no mercado. Contudo, vislumbramos que novos modelos de tarifação surgirão a medida que o mercado de computação na nuvem se diversifica cada vez mais, além dos (novos e atuais) provedores que sempre buscam oferecer novidades a partir de novas soluções. Desta forma, não será surpresa o surgimento de novos esquemas de tarifação neste emergente mercado de computação utilitária. Dentro deste contexto, apresentamos neste capítulo uma modelagem do ambiente híbrido proposto com o intuito de oferecer um arcabouço conceitual para os dois próximos capítulos. A partir desta modelagem, avaliamos como diferentes formas de gerenciamento podem melhorar a eficiência na operação da infraestrutura híbrida, em termos econômicos. Mais especificamente, esta modelagem servirá de base para nossa avaliação de como gerenciar a infraestrutura proposta, dispondo dos esquemas de tarifação mais utilizados no mercado atualmente (reserva e sob demanda) e considerando contratos com reserva de longa duração (Capítulo 4); e também, como gerenciar a infraestrutura híbrida levando em consideração a possibilidade de contratos com reserva de curta duração com os provedores de computação na nuvem (Capítulo 5). Apresentamos na Seção 3.2 a modelagem do sistema, através da carga de trabalho demanda e de seus componentes; e na Seção 3.3 uma definição macro dos problemas inerentes às incertezas do ambiente híbrido proposto.

3.2 Modelo do Sistema

Tratamos o problema da perspectiva de um COO (*Chief Operations Officer*) de uma corporação que deve executar uma carga de trabalho[1] ($\mathcal{W}$) durante um longo período (tipicamente anual). A carga de trabalho consiste de múltiplas aplicações de curta duração do tipo *Bag-of-Tasks*. Um conjunto de provedores de IaaS ($\mathcal{P}$) estão à disposição do COO, cujo trabalho é decidir os contratos ($\mathcal{K}$) que devem ser estabelecidos com os provedores, e o escalonamento

[1]Gostaríamos de lembrar que no Apêndice A apresentamos um sumário da notação utilizada nos modelos propostos.

apropriado ($\mathcal{S}$) para alocação da carga de trabalho na infraestrutura disponível, visando maximizar o lucro obtido. A melhor escolha para ($\mathcal{S}$) vai depender dos valores de ($\mathcal{W}$) e ($\mathcal{K}$).

Certamente não é realístico esperar que o COO saberá precisamente qual será a carga de trabalho para o ano a frente. Assumimos que o COO planejará quais contratos serão estabelecidos baseando-se em uma estimativa da carga de trabalho e ajustará o escalonamento das aplicações em resposta às variações desta estimativa. Demonstraremos nos próximos capítulos que, através da implementação do nosso modelo, um bom desempenho pode ser alcançado mesmo com uma estimativa não muito acurável da carga de trabalho.

Nas duas subseções seguintes apresentamos, respectivamente, a modelagem para a carga de trabalho demandada ao sistema e a modelagem dos componentes que formam a infraestrutura híbrida.

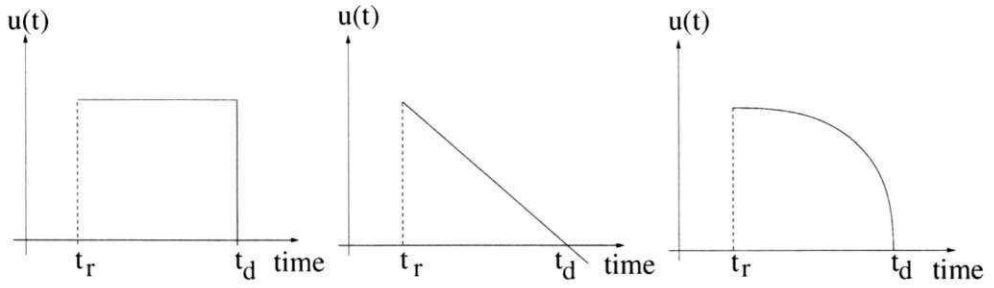
3.2.1 Carga de Trabalho

Assumimos que durante um período de tempo Δt (tipicamente na ordem de um dia), existe apenas uma única aplicação crítica a ser executada. Caracterizamos cada aplicação da carga total de trabalho $\mathcal{W} = \{\mathcal{A}_1, ..., \mathcal{A}_m\}$ que será computada na infraestrutura híbrida pela tupla $\mathcal{A}_i = \langle d_i, tr_i, u_i(\Delta t)\rangle, 1 \leq i \leq m$, onde d_i representa uma indicação da demanda computacional da aplicação i, expressada em termos de ciclos computacionais necessários para sua execução; tr_i é o instante de tempo, dentro do período Δt, quando a aplicação está pronta para ser executada (por exemplo, o instante de tempo em que os dados necessários para sua execução tornam-se disponíveis); e $u_i(\Delta t)$ é a função de utilidade da aplicação. Esta função especifica a utilidade[2] total obtida pelo proprietário da infraestrutura híbrida de TI a partir da execução da aplicação, com uma função do tempo no qual a execução é finalizada. Claramente, $u_i(\Delta t)$ é definida apenas para $t \geq tr_i$.

A Figura 3.1 mostra alguns exemplos de funções de utilidade para aplicações com restrições de tempo. Para tais aplicações, a utilidade é representada como uma função decrescente no tempo. Por exemplo, a Figura 3.1(a), definida por *Função Degrau*, representa aplicações cuja utilidade é positiva e constante tão logo sua execução finalize antes de um prazo

[2]Definimos utilidade como um termo genérico que pode representar o ganho obtido por uma determinada aplicação. Entendemos que esse fator de ganho pode ser mapeado para valores financeiros de forma simples, a partir de dados reais fornecidos, por exemplo, por gerentes de infraestruturas de TI.

determinado (td), definido como o *deadline* de uma aplicação. Caso a execução finalize após o *deadline*, a utilidade cai para zero. A *Função Exponencial*, ilustrada na Figura 3.1(b), decresce exponencialmente e tende a zero com o atraso cada vez maior na finalização da aplicação. Por último, a *Função Linear* na Figura 3.1(c) começa positiva e decresce linearmente com o aumento no tempo de conclusão da aplicação, tornando-se negativa caso esse tempo ultrapasse o *deadline*. A utilidade negativa, neste caso, pode representar uma penalidade aplicada se a aplicação não for executada até um prazo.



(a) AoN (b) LDuD (c) GDuD

Figura 3.1: Exemplos hipotéticos de funções de utilidade para aplicações com restrições de tempo.

Consideramos aplicações que seguem o seguinte padrão comum: a aplicação i não obtém nenhuma utilidade se a sua execução finalizar após um *deadline* predefinido (td_i), ou seja, $u_i(t) = 0, \forall t, t > td_i$. Três aplicações diferentes são consideradas. A primeira, definida como *All-or-Nothing* (AoN), possui apenas uma única restrição de finalizar sua execução antes de um *deadline*. Para a segunda, definida como *Linear-Decay-until-a-Deadline* (LDuD), a utilidade cai linearmente até uma utilidade zero em td_i. Finalmente, consideramos uma aplicação intermediária definida como *Graceful-Degradation-until-a-Deadline* (GDuD), cuja utilidade decresce exponencialmente com o tempo. As duas primeiras aplicações são modeladas pela seguinte função de utilidade:

$$u_i(t) = \begin{cases} a \cdot (t - tr_i) + b, & \text{se } tr_i \leq t \leq td_i; \\ 0, & \text{caso contrário,} \end{cases}$$

com $b > 0$ e $a = 0$ para a aplicação AoN; e $a = -b/(td_i - tr_i)$ para a aplicação $LDuD$. A terceira aplicação é modelada pela seguinte função de utilidade:

$$u_i(t) = \begin{cases} a^t + b, & \text{se } tr_i \leq t \leq td_i; \\ 0, & \text{caso contrário,} \end{cases}$$

com $b > 0$ e $a < -1$.

A escolha destas classes de aplicações é suportada pela nossa própria experiência com os usuários do *middleware* OurGrid (http://www.ourgrid.org/). Por exemplo, previsão do tempo é um exemplo de aplicação do tipo AoN, executada por usuários do OurGrid. É tipicamente executada em grupos de tarefas compostos por um grande número de simulações, o que torna a sua execução apropriada para infraestrutura híbrida aqui proposta. Esta atividade é normalmente executada diariamente, sua inicialização ocorre apenas depois que os dados de entrada estão disponíveis e deve finalizar até um prazo determinado (um *deadline* imposto por uma rede de TV que paga pelo serviço de previsão do tempo, por exemplo). Completar toda simulação bem antes do *deadline* não traz nenhuma utilidade extra para o usuário. Outro cenário típico para execução de aplicações *e-Science* é o processamento destas aplicações durante a noite ou finais de semana. Neste cenário os usuários planejam a execução de suas aplicações paralelas (tipicamente simulações Monte Carlo ou processamento de grandes volumes de dados) para ocorrerem enquanto eles não estão no trabalho. Desta forma, quando retornarem ao trabalho os resultados já estarão prontos e poderão analisar os mesmos sem maiores atrasos. Neste caso, o usuário trabalha com a perspectiva de tempo descrita em seguida. Idealmente, a execução já deve estar finalizada quando o usuário retornar ao trabalho, o que pode não ser prevista precisamente, mas que no geral tem um limite superior de tempo. Até este ponto, a utilidade da aplicação não deve diminuir muito e deve permanecer próxima ao valor máximo. Contudo, deste ponto em diante, e para um período de tempo relativamente curto, a utilidade começa a decrescer bruscamente, até que o tempo após o qual o usuário considera que o cálculo demorou muito e utilidade é zero. A utilidade de todas as aplicações que são executadas seguindo este padrão podem ser expressadas pela definição de uma função GDuD. Para o tipo de aplicações cujo usuário segue uma perspectiva similar ao padrão apresentado acima, mas que por questões diversas e extra execução das aplicações a utilidade decai de maneira ainda mais brusca, podem ser representadas a partir de uma função LDuD.

3.2.2 A Infraestrutura Híbrida de TI

Definimos uma infraestrutura híbrida de TI tal que, durante o período de tempo Δt, é capaz de prover ciclos computacionais adquiridos da infraestrutura local dedicada, da grade computacional P2P e de provedores de computação na nuvem. A Figura 3.2.2 ilustra os componentes da infraestrutura híbrida de TI.

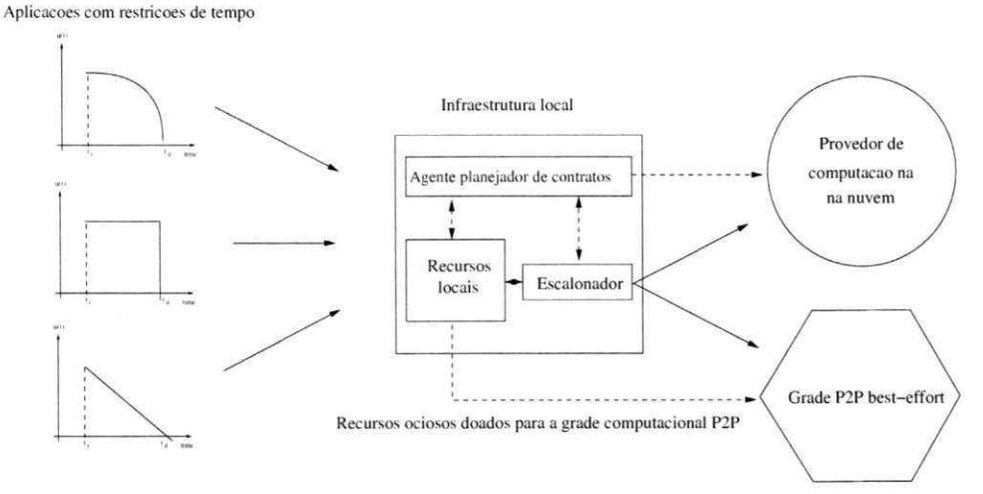


Figura 3.2: Componentes da infraestrutura híbrida de TI.

Assumimos que $\mathcal{P} = \langle \mathcal{P}_1, ..., \mathcal{P}_n \rangle$ representa o conjunto dos provedores disponíveis. Denotamos um único provedor através da tupla $\mathcal{P}_j = \langle pr_j, pu_j(t) \rangle, 1 \leq j \leq n$, onde pr_j é a taxa fixa de reserva que o provedor $\mathcal{P}_j$ cobra pela reserva de uma unidade de recurso por uma unidade de tempo, e $pu_j(t)$ é o preço cobrado em t pelo uso de uma unidade de recurso por unidade de tempo, para cada instante t dentro do período de reserva. Esta dependência entre o tempo e o preço pode refletir, por exemplo, sazonalidade nos preços. Representamos um planejamento de capacidade $\mathcal{K} = \langle \langle \mathcal{K}_1(t), tb_1, te_1 \rangle, ..., \langle \mathcal{K}_n(t), tb_n, te_n \rangle \rangle$ como uma sequência de contratos estabelecidos com cada um dos provedores disponíveis, onde a função $\mathcal{K}_j(t)$ retrata a quantidade de unidades de recursos reservada para uso a partir do provedor $\mathcal{P}_j$ em todos os instantes de tempo t dentro do período de reserva definido por $[tb_j, te_j]$[3]. Assim, a cada instante de tempo $t, tb_j \leq t \leq te_j$, o contratante não pode utilizar mais do que $\mathcal{K}_j(t)$

[3]Tipicamente, $\mathcal{K}_j(t)$ é constante no tempo, mas preferimos modelar esta característica de uma maneira mais genérica.

unidades de recursos do provedor $\mathcal{P}_j$. Definimos, então, o custo de reserva de um contrato $\mathcal{K}$ por:

$$\mathcal{R}(\mathcal{K}) = \sum_{j=1}^{n} pr_j \cdot \int_{tb_j}^{te_j} \mathcal{K}_j(t)dt. \tag{3.1}$$

É importante observar que um planejamento compreende os contratos com todos os provedores, mas não implica que um contrato é realmente estabelecido com cada um dos provedores. Um contrato $\mathcal{K}_j(t)$ tal que $\mathcal{K}_j(t) = 0, \forall t, tb_j \leq t \leq te_j$, terá um custo de reserva igual a zero e não disponibilizará recursos para uso futuro, tornando-se essencialmente nulo. Essa modelagem, no entanto, simplifica a formalização do problema.

Com o modelo apresentado acima, podemos definir diferentes tipos de provedores, de acordo com os preços de reserva e uso especificados em seus respectivos contratos. Por exemplo, se $pr_j \neq 0$ e $pu_j(t) \neq 0$, então $\mathcal{P}_j$ provê reserva de recursos, a qual antecede a utilização dos mesmos. Por outro lado, se $pr_j = 0$ e $pu_j(t) \neq 0$, então $\mathcal{P}_j$ provê apenas recursos sob demanda, sem a necessidade de reservar previamente. Neste caso, $\mathcal{K}_j(t)$ é a quantidade máxima de recursos que podem ser adquiridos do provedor sob demanda $\mathcal{P}_j$ no tempo t. Assumimos também que o custo de operar a infraestrutura local dedicada (adquirir *hardware* e *software*, instalações, consumo de energia, pessoal, etc) pode ser uniformemente amortizado durante o período de reserva. Então, considerando $pr_j \neq 0$ e $pu_j(t) = 0$ para todo t, podemos modelar a infraestrutura local. Finalmente, a grade P2P pode ser modelada com um provedor em que $pr_j = pu_j(t) = 0$ para todo t, visto que não existe a reserva de recursos na grade e seu custo de uso é desprezível se comparado com o custo dos demais provedores. Portanto, para efeito de simplificação da modelagem, consideramos o custo de utilizar o *OurGrid* desprezível ao longo deste trabalho. Contudo, é importante salientarmos que existe um custo operacional para manter/oferecer serviços a partir de grades computacionais e que, em alguns casos, pode ser alto se comparado ao custo de aquisição de recursos.

Definimos $[tb_j, te_j]$ como o intervalo de tempo no qual o gerenciamento da infraestrutura será planejado, e consideramos que $\forall j, \mathcal{K}_j(t) = 0$ se $t < tb_j$ ou $t > te_j$. Além disso, consideramos que uma política de escalonamento $\mathcal{S} = \{uc_j^i(t), \forall i, j, t | 1 \leq i \leq m, 1 \leq j \leq n, tb_j \leq t \leq te_j\}$ define como os ciclos disponíveis a partir do contrato $\mathcal{K}$ são utilizados para computar a carga de trabalho $\mathcal{W}$, sujeita a:

$$\forall j, t, 1 \leq j \leq n, tb_i \leq t \leq te_i, \sum_{i=1}^{m} uc_j^i(t) \leq \mathcal{K}_j(t); \tag{3.2}$$

e,

$$\forall i, 1 \leq i \leq n, \sum_{j=1}^{n} \int_{tr_i}^{te_j} uc_j^i(t)dt = d_i. \tag{3.3}$$

Observe que $uc_i^j(t)$ define o número de ciclos do provedor $\mathcal{P}_j$ que deve ser utilizado para computar a aplicação $\mathcal{A}_i$ no tempo t.

Representamos o instante de tempo da finalização de uma aplicação $\mathcal{A}_i$ por tc_i. Assim, a utilidade obtida com a execução da carga de trabalho $\mathcal{W}$ a partir da política de escalonamento $\mathcal{S}$ é dada por:

$$\mathcal{U}(\mathcal{W}, \mathcal{S}) = \sum_{i=1}^{m} u_i(tc_i - tr_i). \tag{3.4}$$

Por outro lado, o custo incorrido pela execução da carga de trabalho $\mathcal{W}$ a partir de $\mathcal{S}$ é dado por:

$$\mathcal{C}(\mathcal{W}, \mathcal{S}) = \sum_{i=1}^{m} \sum_{j=1}^{n} (\int_{tr_i}^{tc_i} uc_j^i(t) \cdot pu_j(t)dt). \tag{3.5}$$

Definimos o lucro da infraestrutura híbrida como a diferença entre a utilidade obtida a partir da execução das aplicações, e a soma dos custos de provisionamento da infraestrutura durante todo o período e o custo de processar realmente a carga de trabalho. Logo, o lucro ($\wp$) obtido a partir de um determinado planejamento $\mathcal{K}$ para processar $\mathcal{W}$ sob o escalonamento $\mathcal{S}$ é dado por:

$$\wp(\mathcal{W}, \mathcal{K}, \mathcal{S}) = \mathcal{U}(\mathcal{W}, \mathcal{S}) - [\mathcal{R}(\mathcal{K}) + \mathcal{C}(\mathcal{W}, \mathcal{S})]. \tag{3.6}$$

3.3 Definição do Problema

No ambiente proposto, o escalonamento das aplicações que formam a carga de trabalho $\mathcal{W}$ é precedido pela aquisição dos recurso, o que chamamos de planejamento da capacidade [Thain, Tannenbaum e Livny 2005]. Portanto, o principal problema atacado é como executar o planejamento dos recursos, ou seja, quando e quanto de recursos extra deverão ser contratados junto aos provedores de computação na nuvem. De uma maneira geral, precisamos

elaborar algoritmos que um agente de *software* possa utilizar para definir, automaticamente, quais contratos devem ser estabelecidos com os provedores externos de recursos, tal que a execução das aplicações possam finalizar antes de seus respectivos *deadlines* e com o maior lucro possível.

A grande dificuldade em planejar os recursos para uso futuro advém das incertezas inerentes ao ambiente proposto, como por exemplo, incertezas na quantidade de recursos adquirida da grade computacional P2P, no comportamento futuro da carga de trabalho demandada e na variação dos preços no mercado de computação na nuvem. No geral, contratos que são estabelecidos com antecedência custam menos, porém, menos precisa é a informação sobre a quantidade extra de computação necessária.

O outro problema atacado é como escalonar de maneira mais eficiente as aplicações $\mathcal{A}_i$, uma vez que os recursos estejam previamente reservados. Consideramos que no escalonamento das aplicações as informações a respeito do sistema são mais precisas. Assim, a dificuldade é encontrar o melhor tempo para finalizar as aplicações (tc_i). Ou seja, a partir de tr_i o escalonador deve determinar o tempo ideal para finalizar a aplicação $\mathcal{A}_i$ que maximize o lucro obtido. Se por um lado finalizar cada aplicação o quanto antes aumenta a utilidade obtida, por outro lado, utilizar os recursos com preços mais baixos diminui os custos incorridos na execução. Em alguns casos, pode ser mais vantajoso utilizar os recursos locais e provenientes da grade por mais tempo, ao invés de utilizar todos os recursos disponíveis para finalizar a aplicação o quanto antes, inclusive os recursos mais caros disponibilizados pelos provedores externos de computação na nuvem.

Dada esta descrição geral dos problemas a serem atacados, assumimos duas abordagens de gerenciamento para a infraestrutura híbrida de TI. Nos próximos dois capítulos apresentamos, respectivamente, duas abordagens de gerenciamento: uma considera reservas de longo prazo e outra considera reservas de curto prazo. Mais especificamente, o gerenciamento a longo prazo considera um cenário de reserva dos recursos com um ano de duração; enquanto o gerenciamento a curto prazo considera um cenário com reservas diárias dos recursos. Para cada uma destas abordagens, apresentamos os problemas específicos de cada cenário de reserva e como o modelo foi mapeado para resolver estes problemas.

Capítulo 4

Gerenciamento a Longo Prazo

4.1 Introdução

No mercado de computação na nuvem, a reserva de recursos pode ser a forma adequada para mitigar os riscos enfrentados por ambos provedores e clientes. Enquanto provedores enfrentam o risco de operarem suas infraestruturas com capacidade sub utilizada, clientes enfrentam o risco de, quando necessário, não ter os recursos disponíveis a preços razoáveis. Assim, contratos de reserva são vantajosos tanto na perspectiva dos provedores, já que é uma forma de aumentar as garantias de operação estável permanente; quanto na perspectiva do cliente, já que serve como uma proteção contra flutuações nos preços e indisponibilidade de recursos. Nesta direção, a *Amazon.com* (indiscutivelmente o maior provedor de IaaS no mercado atualmente) oferece a opção de reserva de recursos para o seu serviço de computação na nuvem *Elastic Compute Cloud* (EC2). Através deste serviço os recursos podem ser reservados por longos períodos de tempo (pelo menos um ano de duração). Usuários pagam uma taxa única para a reserva de recursos e são cobrados pela utilização dos mesmos a uma taxa reduzida, se comparada ao modelo de tarifação sob demanda.

Esta realidade nos levou a investigar o problema de planejar a capacidade da infraestrutura híbrida de TI através de contratos de longa duração estabelecidos com provedores de computação na nuvem. Consideramos uma abordagem dirigida a negócios e propomos uma estratégia tanto para o planejamento de contratos a longo prazo, quanto para utilização eficiente dos recursos, visando maximizar o lucro obtido pela execução de uma carga de trabalho que chega à infraestrutura híbrida, durante um longo período de tempo.

A dificuldade em planejar a infraestrutura híbrida advém das incertezas inerentes ao ambiente, associadas à demanda de trabalho e ao comportamento da grade *best-effort*. Por esse motivo, planejamos a capacidade necessária para computar a carga de trabalho de um ano de duração considerando o padrão da demanda *esperada*, bem como a *esperada* qualidade de serviço da grade P2P. Para a alocação dos recursos, a dificuldade surge em determinar a melhor estratégia de utilização dos recursos reservados, visando alcançar um gerenciamento lucrativo da infraestrutura híbrida. Por exemplo, decidir se é melhor finalizar as aplicações o quanto antes (o que significa uma melhor utilidade) e a qualquer custo; ou usar os recursos mais baratos para computar a aplicação, mesmo que isto signifique finalizar a aplicação próximo do seu *deadline*.

O restante deste capítulo está organizado como segue. A Seção 4.2 apresenta uma instanciação do modelo do sistema apresentado anteriormente e formaliza o problema do gerenciamento a longo prazo de uma infraestrutura híbrida de TI. Devido à sua complexidade, dividimos o problema do gerenciamento em dois problemas menores: (i) planejamento da capacidade, cuja execução é realizada *off-line* e define os contratos a longo prazo que devem ser estabelecidos; e (ii) escalonamento em tempo de execução, que de fato realiza a alocação de partes da carga de trabalho nos recursos apropriados, buscando maximizar o lucro na operação da infraestrutura. Discutimos o planejamento da capacidade na Seção 4.3 e abordamos o problema do escalonamento na Seção 4.4. Na Seção 4.5 apresentamos uma avaliação de desempenho das estratégias propostas, com o intuito de promover a adoção de tais estratégias por gerentes que desejam maximizar o lucro com o planejamento e alocação de recursos de uma infraestrutura híbrida de TI. Por fim, apresentamos algumas considerações finais sobre o gerenciamento a longo prazo na Seção 4.6.

4.2 Instanciação do Modelo e Definição do Problema

Consideramos, neste capítulo, que recursos podem ser adquiridos de provedores de *IaaS* sem intervenção humana, cada um com seu próprio esquema de tarifação e sua qualidade de serviço associada. Mais especificamente, consideramos um cenário de provedores que oferecem recursos tais como os principais serviços encontrados atualmente no mercado de computação na nuvem, através de recursos oferecidos sob demanda ou através de reservas

de longa duração. Deixamos para uma análise futura os serviços oferecidos por um mercado de leilão dos recursos (*spot market*).

O problema que buscamos resolver é como otimizar o lucro da execução de uma infraestrutura híbrida de TI. Visto que o custo de usar uma unidade de recurso é normalmente calculado para um intervalo de tempo (Δt), com o uso do recurso por uma fração deste intervalo sendo cobrado como se tivesse sido utilizado por todo o intervalo, mapeamos todas as funções dependentes do tempo de sua forma continua descrita no capítulo anterior para uma discreta. Dessa forma, consideramos que o período para o qual a capacidade é primeiramente planejada e depois utilizada para um contrato j é compreendido de $T_j = (te_j - tb_j)$ intervalos de tempo, com uma hora de duração cada. Além disso, como descrito no capítulo anterior, $\mathcal{K}_j(t)$ é a quantidade de recursos providos por $\mathcal{P}_j$ que estão disponíveis durante o t-(ésimo) intervalo de tempo, e $pu_j(t)$ é o valor cobrado pela utilização destes recursos durante o mesmo intervalo de tempo. Assumimos também que o instante de tempo em que uma aplicação $\mathcal{A}_i$ está pronta para execução (tr_i) e o tempo em que finaliza (tc_i) ocorrem sempre no início de um destes intervalos discretos de tempo. Portanto, para todas as aplicações, $\Delta t = tc_i - tr_i$ é um número natural e para cada $\Delta t \in I\!N$ existe um valor $u_i(\Delta t)$ que define a utilidade obtida quando a execução de $\mathcal{A}_i$ finaliza em tc_i. A partir destas suposições, formalizamos o problema como segue:

Maximize

$$\wp(\mathcal{W}, \mathcal{K}, \mathcal{S}) = \mathcal{U}(\mathcal{W}, \mathcal{S}) - [\mathcal{R}'(\mathcal{K}) + \mathcal{C}'(\mathcal{W}, \mathcal{S})], \tag{4.1}$$

onde

$$\mathcal{R}'(\mathcal{K}) = \sum_{j=1}^{n} (pr_j \cdot \sum_{t=1}^{T_j} \mathcal{K}_j(t)), \tag{4.2}$$

e,

$$\mathcal{C}'(\mathcal{W}, \mathcal{S}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{t=tr_i}^{tc_i} (uc_j^i(t) \cdot pu_j(t)), \tag{4.3}$$

sujeito a

$$\forall j, 1 \leq j \leq m, \sum_{i=1}^{m} uc_j^i(t) \leq \mathcal{K}_j(t) \tag{4.4}$$

e

$$\forall i, j, 1 \leq i \leq n, 1 \leq j \leq m, \sum_{j=1}^{n} \sum_{t=tr_i}^{tc_i} uc_j^i(t) = d_i. \tag{4.5}$$

Em outras palavras, o problema é como encontrar, para uma dada carga de trabalho $\mathcal{W}$ e um conjunto de provedores $\mathcal{P}$, o planejamento $\mathcal{K}_{opt}$ que conduz $\wp(\mathcal{W}, \mathcal{K}_{opt}, \mathcal{S}(\mathcal{W}, \mathcal{K}_{opt}))$ ao máximo valor alcançável. É importante notar que para uma determinada carga de trabalho $\mathcal{W}$ e planejamento $\mathcal{K}$ podem haver diversas políticas de escalonamento $\mathcal{S}$ possíveis, levando a diferentes utilidades e custos associados. Por conseguinte, calcular o lucro máximo obtido por um dado planejamento $\mathcal{K}$ requer solucionar um problema de otimização para encontrar o escalonamento ótimo $\mathcal{S}_{opt}(\mathcal{W}, \mathcal{K})$ para o planejamento $\mathcal{K}$ e carga de trabalho $\mathcal{W}$. Para este tipo de otimização multiobjetivo, nem sempre é trivial (ou mesmo possível) encontrar a solução ótima [Miettinen 1999].

Em adição à complexidade trazida pela formulação do problema em termos da dupla otimização, existem outras dificuldades em solucioná-lo. As dificuldades adicionais são todas relacionadas às incertezas nas informações disponíveis no tempo em que as otimizações precisam ser executadas, ou seja, no ato de planejamento da capacidade. Primeiramente, exceto em casos muito específicos, não é possível antecipar precisamente a carga de trabalho que será processada durante todo o período de interesse. Depois, a capacidade futura disponível em um provedor $\mathcal{P}_j$ pode não ser facilmente estimada quando os recursos não são reservados antecipadamente (i.e. quando $pr_j = 0$). Em termos mais concretos, não é fácil prever com antecedência quantos recursos estarão disponíveis, em um provedor de serviços sob demanda, ou de uma grade P2P *best-effort*. Por fim, no geral, não é possível predizer os preços futuros dos recursos adquiridos de um provedor que oferece serviços sob demanda (ou seja, aqueles onde $pr_j = 0$ e $pu_j(t) > 0$).

No intuito de superar essas dificuldades, tomamos uma abordagem *dividir-e-conquistar*. Dividimos a solução do problema em dois momentos distintos no tempo. Em um primeiro momento, o do planejamento da capacidade de longo prazo, solucionamos um problema

de otimização simplificado que utiliza informação disponível naquele momento e faz suposições sobre a informação que está faltando. Então, no momento real da execução da carga de trabalho, otimizamos a execução utilizando quaisquer recursos disponíveis neste momento. É importante perceber que a disponibilidade de parte destes recursos depende do planejamento "subótimo" que foi executado antes. Assim, o planejamento tenta estabelecer as melhores reservas possíveis, considerando as informações disponíveis naquele momento, enquanto o escalonamento tenta "corrigir", em tempo de execução, as limitações do planejamento subótimo. Nas próximas duas seções descrevemos os algoritmos de planejamento subótimo da capacidade e de escalonamento em tempo de execução propostos para resolver o problema formalizado acima.

4.3 Planejamento Subótimo de Capacidade

Por efeito de simplificação, consideramos um sistema com apenas uma grade P2P, um único provedor local, e alguns provedores de computação na nuvem que oferecem tanto ciclos reservados quanto sob demanda. Modelamos cada provedor na nuvem como dois provedores separados, um que oferece apenas ciclos sob demanda e outro apenas ciclos reservados. Definimos o provedor $\mathcal{P}_1$ como a grade P2P, então $pr_1 = 0$ e $pu_1(t) = 0, \forall t, 1 \leq t \leq T_1$. O provedor $\mathcal{P}_2$ representa a capacidade local, assim $pr_2 > 0$ e $pu_2(t) = 0, \forall t, 1 \leq t \leq T_2$. E do provedor $\mathcal{P}_3$ até $\mathcal{P}_n$ representamos aqueles que proveem serviços reservados, seguidos pelos que provêem serviços sob demanda. Assumimos que em qualquer instante t, provedores estão ordenados em ordem crescente de preço de uso (ou seja, $\forall k, j, 1 \leq j, k \leq n, pu_k(t) < pu_j(t) \longrightarrow k < j, \forall t, 1 \leq t \leq T_j$), e o desempate é baseado na ordem crescente do preço de reserva. Devido às incertezas nos preços de uso e na disponibilidade dos recursos que podem ser obtidos dos provedores que oferecem serviços sob demanda, tomamos duas abordagens alternativas de aproximação. Na primeira, desconsideramos os recursos sob demanda. Modelamos isso considerando, em todos os intervalos de tempo, o preço de uso destes provedores como infinito e a quantidade de ciclos reservados igual a zero. Na outra abordagem consideramos que o preço de uso e a disponibilidade são constantes e baseados no conhecimento acessível na hora que o planejamento é executado (por exemplo, os valores atuais na hora do planejamento ou um valor médio durante algum período passado). Nos dois

casos, contudo, assumimos que os preços de uso para os recursos sob demanda são sempre maiores do que os preços cobrados pelos recursos reservados.

Assumimos também que a capacidade local é conhecida e constante ao longo do tempo; e denotamos este valor por $\mathcal{K}_2$. Dessa forma, resolver o planejamento se resume a definir a quantidade de ciclos que precisam ser reservados dos provedores que oferecem o serviço de reserva, para assim maximizar o lucro esperado. Também por simplicidade, consideramos que a quantidade de ciclos reservados não varia ao longo do tempo, ou seja, para $2 < j \leq 2 + (n-2)/2$, $\mathcal{K}_j(t)$ é independente de t e denotamos este valor por $\mathcal{K}_j$.

Presumimos que a carga de trabalho $\mathcal{W}$ utilizada para realizar o planejamento é uma boa estimativa da carga real de trabalho que será submetida à infraestrutura híbrida. Se existir uma boa caracterização de uma distribuição que representa a carga de trabalho para o longo período assumido na reserva, então $\mathcal{W}$ pode ser uma amostra gerada a partir desta distribuição.

Nos referimos ao *peer* responsável pela negociação dos recursos da infraestrutura dedicada ociosa na grade P2P como o *peer* local. Estimamos o número de ciclos disponíveis na grade P2P no começo do intervalo de tempo t ($\mathcal{K}_1(t)$) através da função $b(t)$, que registra o balanço no tempo t das interações passadas do *peer* local com todos os outros *peers* na grade P2P. A função de balanço $b(t)$ é dada por:

$$b(t) = b(t-1) + (\mathcal{K}_2 - \sum_{i=1}^{m} uc_2^i(t-1)) \cdot p_d \cdot p_c - \sum_{i=1}^{m} uc_1^i(t-1), \tag{4.6}$$

onde p_d é a probabilidade de existir uma demanda para o ciclo que está sendo doado, e p_c é a probabilidade que, uma vez doado, o ciclo foi consumido por um *peer* "colaborador"[1]. Assim, o valor da função de balanço no início do intervalo de tempo t é o valor do início do intervalo anterior, mais a quantidade de recursos doados para colaboradores, menos a quantidade de recursos consumida da grade também no intervalo de tempo anterior.

Para tornar o problema de otimização tratável, evitamos a dupla otimização simplificando o escalonamento assumido no processo de planejamento da capacidade. Por esta razão, assumimos que todos os recursos reservados disponíveis no intervalo de tempo t são utilizados

[1]Um *peer* colaborador é aquele que participa por completo no sistema, consumindo e também doando recursos. Um *peer* "carona" (conhecido também como *free-rider*), por sua vez, é aquele que apenas consome recursos do sistema [Andrade et al. 2007].

para computar todas as aplicações de $\mathcal{W}$ não finalizadas até t e estão prontas para execução em t ou mesmo antes. Consideramos também que aplicações que ficam prontas para execução mais cedo têm prioridade sobre aplicações que "chegam" depois. Por esse motivo, é possível que uma aplicação $\mathcal{A}_i$ pronta para execução em tr_i, seja computada apenas em $tr'_i, tr'_i > tr_i$, visto que todos os recursos disponíveis em tr_i estão sendo consumidos por outras aplicações $\mathcal{A}_j$, cujos $tr_j < tr_i$. Isto faz com que os tempos em que as aplicações finalizam sejam tão rápidos quanto possível, podendo ser calculados em função da quantidade de recursos reservados em cada provedor. Baseados nesta suposição, temos o tempo de finalização de uma aplicação $\mathcal{A}_i$ dado por:

$$tc_i = tr'_i + \left\lceil \frac{max(0; d_i - [\sum_{j=2}^{n} \mathcal{K}_j + b(tr'_i)] - \sum_{j=1}^{n} \sum_{i'=1}^{i-1} uc_j^{i'}(tr'_i)))}{\sum_{j=2}^{n} \mathcal{K}_j} \right\rceil \tag{4.7}$$

com $tr'_i = tr_i + max(0; tc_{i-1} - tr_i)$. Ou seja, o tempo para conclusão da aplicação $\mathcal{A}_i$ é o tempo em que a mesma começou realmente a ser executada (ou ainda, o tempo em que a última aplicação foi finalizada) mais o tempo necessário para executar a sua demanda (d_i). O tempo para executar a demanda d_i depende da quantidade de recursos disponíveis nos provedores e do crédito na grade, menos a quantidade de recursos utilizada para executar as aplicações anteriores.

Especificamos o escalonamento subótimo $\mathcal{S}(\mathcal{W}, \mathcal{K})$ pelos valores uc_j^i para $i, j, 1 \leq i \leq m, 1 \leq j \leq n$, e estes valores são dados como segue:

$$uc_j^i(t) = 0 \ \ \forall t, t < tr'_i \vee t > tc_i; \tag{4.8}$$

$$uc_1^i(tr_i) = min\{b(tr_i) - \sum_{i'=1}^{i-1} uc_1^{i'}(tr_i); max(0; d_i - (\mathcal{K}_2(tr_i) - \sum_{i'=1}^{i-1} uc_2^{i'}(tr_i)))\}; \tag{4.9}$$

$$uc_1^i(t) = 0 \ \ \forall t, tr_i < t \leq tc_i; \tag{4.10}$$

$$uc_j^i(t) = min\{(\mathcal{K}_j(t) - \sum_{i'=1}^{i-1} uc_j^{i'}(t)); \ \ (d_i - \sum_{t'=tr_i}^{t-1} \sum_{j'=1}^{n} uc_{j'}^i(t') - \sum_{j''=1}^{j-1} uc_{j''}^i(t))\},$$
$$\forall t, tr_i \leq t \leq tc_i, j \neq 1. \tag{4.11}$$

A estratégia para o planejamento de capacidade que propomos aqui busca maximizar o lucro obtido através da operação da infraestrutura híbrida, escolhendo apropriadamente a quantidade de recursos que deve ser reservada junto aos provedores de computação na nuvem. A formulação acima nos permite usar um otimizador simples para encontrar o número de recursos que devem ser reservados em cada provedor na nuvem, a fim de maximizar o lucro esperado.

4.4 Escalonamento em Tempo de Execução

Como discutimos anteriormente, o escalonamento que foi projetado assumindo o planejamento subótimo precisa ser reavaliado em tempo de execução. A cada intervalo de tempo t quando uma decisão de escalonamento de recursos precisa ser tomada, existem poucas incertezas ofuscando as melhores escolhas a serem feitas. Em particular, a carga real de trabalho para ser processada (ao menos a carga das aplicações que estão prontas para execução em t), a quantidade de recursos que podem ser obtidos a partir da grade P2P e dos provedores de serviços sob demanda, e o preço destes serviços em t são todos conhecidos.

Propomos então o seguinte algoritmo de escalonamento. Definimos $\mathcal{W}(t) \subseteq \mathcal{W}$ como o sub-conjunto de aplicações que estão prontas para execução no tempo t, ou seja, $\mathcal{W}(t) = \{\mathcal{A}_i | tr_i = t\}$. No começo de cada intervalo t, o escalonador executa um procedimento de otimização que calcula o melhor tempo para finalização (tc_i) de todas as aplicações $\mathcal{A}_i \in \mathcal{W}(t)$. Esta otimização considera apenas as aplicações que estão prontas para execução no começo de cada intervalo de tempo t e assume que a mesma quantidade de recursos que podem ser adquiridos dos provedores sob demanda no intervalo de tempo t podem também ser adquiridos no intervalo seguinte pelo mesmo preço oferecido em t. Considerando uma carga de trabalho anual composta de aplicações de curta duração (diária), esta suposição é razoável, visto que os tempos reais calculados para finalização das aplicações (tc_i) estão relativamente próximos do momento em que a otimização é executada. Pode não ser o caso quando a carga de trabalho é composta de aplicações de longa duração. Neste trabalho, portanto, nos concentramos em aplicações de curta duração. Além disso, a otimização considera que até $\mathcal{K}_1(t)$ ciclos podem ser recebidos de volta da grade P2P no tempo t e nenhum ciclo é usado da grade nos intervalos de tempo seguintes. Novamente, assumimos que os provedores são ordenados

pelo preço crescente de uso (ou seja, $\forall k, j, 1 \leq j, k \leq n, pu_k(t) < pu_j(t) \rightarrow k < j$). Logo, o otimizador prioriza a utilização de provedores mais baratos. Para cada intervalo de tempo t, recursos são escalonados seguindo os valores de $uc_j^i(t)$ definidos pelo otimizador.

4.5 Avaliação de Desempenho

Nesta seção avaliamos o desempenho do planejamento subótimo de capacidade e do escalonador em tempo de execução propostos nas seções anteriores. Primeiramente, descrevemos como o modelo foi implementado e como os experimentos foram conduzidos. Em seguida, descrevemos os cenários avaliados e como os parâmetros do sistema foram instanciados. No fim desta seção, apresentamos os resultados obtidos.

4.5.1 Implementação do Modelo e Descrição dos Experimentos

Implementamos um otimizador de acordo com o modelo proposto com o intuito de avaliar o problema apresentado na Seção 4.2. Seu funcionamento é explicado como segue. Dada uma carga de trabalho $\mathcal{W}$, o conjunto de provedores $\mathcal{P}$ e um planejamento $\mathcal{K}$, o algoritmo escolhe qual é a melhor hora para finalizar cada aplicação $\mathcal{A}_i$ entre tr_i e seu respectivo *deadline* (td_i), considerando todos os recursos disponíveis. A quantidade de recursos disponível depende da quantidade de recursos reservada com os provedores de computação na nuvem, bem como da quantidade de recursos obtida da grade P2P. O otimizador executa o algoritmo de planejamento experimentando diferentes opções para a quantidade de recursos reservada em cada provedor, com o objetivo de encontrar a opção que alcança o maior lucro esperado.

Os experimentos foram conduzidos como explicado a seguir. Na fase de planejamento, consideramos um *cenário base* (daqui em diante referenciado como *baseline*), que representa um conjunto particular de valores escolhidos para certos parâmetros do sistema (demanda de carga de trabalho, QoS da grade P2P e preço de uso dos recursos na nuvem sob demanda). Justificamos a escolha destes parâmetros na próxima subseção. Então, introduzimos "pertubações" nos valores utilizados no *baseline* na fase de execução para representar o efeito de erros na predição destes parâmetros. Dessa forma, a otimização na fase de planejamento é realizada assumindo um dado cenário (*baseline*), enquanto que a otimização na fase de escalonamento dos recursos é executada assumindo outro (uma dada perturbação do *baseline*).

Para cada cenário variado calculamos um planejamento ótimo ($\mathcal{K}_{optimal}$) e o lucro associado executando a fase de planejamento com os valores perturbados, ao invés do *baseline*. Podemos então quantificar o impacto que estimativas erradas para os parâmetros do sistema causam no lucro alcançado utilizando a seguinte métrica de eficiência:

$$\mathcal{E}(\mathcal{K}) = 100 \cdot \frac{\wp(\mathcal{W},\mathcal{K},\mathcal{S}))}{\wp(\mathcal{W},\mathcal{K}_{optimal},\mathcal{S})}, \tag{4.12}$$

a qual expressa o lucro obtido pelo cenário de reserva obtido na fase de planejamento com o um percentual do lucro obtido pelo planejamento ótimo para aquele determinado cenário.

4.5.2 Descrição dos Cenários de Avaliação

Assumimos um sistema com 5 provedores, compreendendo: uma grade P2P ($\mathcal{P}_1$), a infraestrutura local dedicada ($\mathcal{P}_2$) e 3 provedores de computação na nuvem, dois deles oferecendo serviços a partir de reservas ($\mathcal{P}_3$ e $\mathcal{P}_4$) e um oferecendo serviços sob demanda ($\mathcal{P}_5$).

Para avaliar a incerteza no preço do serviço oferecido pelo provedor de recursos sob demanda ($\mathcal{P}_5$) modelamos uma flutuação nos preços futuros através do erro na predição dos mesmos. Assumimos o preço praticado pelo serviço Amazon EC2 na época de execução dos experimentos (USD$0.10) [Amazon AWS 2010] [Vogels 2009] para o *baseline*, e para os cenários variados a partir do *baseline* assumimos um erro na predição do custo do serviço sob demanda dentro do conjunto $\{-20\%, -10\%, 0\%, +10\%, +20\%\}$. Ou seja, avaliamos cenários onde o preço do serviço sob demanda estava de 20% a 10% mais barato do que o *baseline*, a até 10% e 20% mais caro. Consideramos que o preço do uso sob demanda é também constante ao longo do tempo. Quanto aos serviços de reserva, os dois provedores diferem de maneira que um oferece uma taxa de reserva mais barata, enquanto o outro oferece uma taxa de uso mais barata. Modelamos estes provedores seguindo o esquema de tarifação empregado no serviço EC2 da *Amazon.com* em 2010, para reserva da menor instância por um período de um ano [Amazon AWS 2010] [Vogels 2009]. Configuramos um provedor ($\mathcal{P}_4$) com o mesmo esquema de preços do EC2, enquanto o outro ($\mathcal{P}_3$) tem um custo de uso que é metade do empregado pelo EC2 e um custo de reserva duas vezes maior. Em ambos os casos, o preço de uso é constante ao longo do tempo, ou seja, $pu_j(t) = pu_j, \forall t, j, 1 \leq t \leq T_j \wedge 2 < j \leq 2 + (n-2)/2$. Consideramos que, por ra-

zões que vão além do simples custo de gerenciar a infraestrutura, uma mínima quantidade de recursos está disponível localmente ($\mathcal{K}_2$). Em nossos experimentos, consideramos que a capacidade da infraestrutura local é equivalente a uma fração de 40% da capacidade média por hora necessária para executar a carga de trabalho anual esperada. Calculamos o custo de reserva da infraestrutura local assumindo que o custo de operar tal infraestrutura pelo período de um ano é maior do que o custo de reservar e exaustivamente utilizar a mesma quantidade de recursos junto ao EC2. Para isto, utilizamos um *fator de ineficiência* ι, ou seja, $pr_2 = \iota \cdot (pr_3 + pu_3) \cdot \mathcal{K}_2 \cdot T_2$. Nos cenários avaliados assumimos que $\iota = 5$. Admitimos que os recursos disponíveis a partir do mercado sob demanda a cada intervalo de tempo são limitados a 20 instâncias, seguindo novamente as restrições impostas pela *Amazon.com* para negociação automática de instâncias no EC2.

Para a grade P2P, assumimos que no começo de cada experimento não existem créditos deixados no sistema, ou seja, $b(1) = 0$. Admitimos que $p_c \cdot p_d = 0.72$ durante a fase de planejamento da capacidade do *baseline*. Durante o escalonamento em tempo de execução, entretanto, consideramos um erro na estimativa de $p_c \cdot p_d$, dentro do seguinte conjunto de valores $\{-20\%, -10\%, 0\%, +10\%, +20\%\}$. O valor 0.72 foi estimado em Carvalho et al. [Carvalho et al. 2010] a partir de registros de execução de aplicações reais.

Assumimos que a carga de trabalho executada na infraestrutura proposta é compreendida de aplicações científicas de processamento intensivo e do tipo *Bag-of-Tasks*. Para estas aplicações, os parâmetros utilizados no *baseline* foram obtidos de registros do *Grid Workloads Archive* [Iosup et al. 2008] (GWA-T3 - NorduGrid). Cada execução é "alimentada" por registros de um grupo particular de usuários, de uma determinada grade presente no arquivo. Mais especificamente, agrupamos as aplicações em *Bag-of-Tasks* e coletamos informações sobre os tempos de submissão e execução das tarefas. Em cada experimento a fase de planejamento é alimentada por aplicações de usuários do registro real (na forma de *Bag-of-Tasks*), enquanto na fase de escalonamento utilizamos uma versão "distorcida" do registro real. Obtemos a carga de trabalho utilizada na fase de escalonamento pela introdução de um erro percentual fixo na demanda d_i de cada aplicação $\mathcal{A}_i$ que compreende a carga total $\mathcal{W}$. Os valores que utilizamos para "perturbar" a carga de trabalho estão dispostos no conjunto $\{-40\%, -20\%, 0\%, +20\%, +40\%\}$.

Assumimos que a utilidade obtida pela execução de cada aplicação $\mathcal{A}_i$ tem um valor

máximo M_i quando $tc_i = tr_i$ e decai linearmente a medida que tc_i aumenta, chegando ao valor zero em $tc_i = td_i$, onde td_i é o *deadline* do aplicação $\mathcal{A}_i$. Formalizando esta função de utilidade temos:

$$u_i(tc_i - tr_i) = max\{0, M_i \cdot (1 - (tc_i - tr_i)/(td_i - tr_i))\}. \tag{4.13}$$

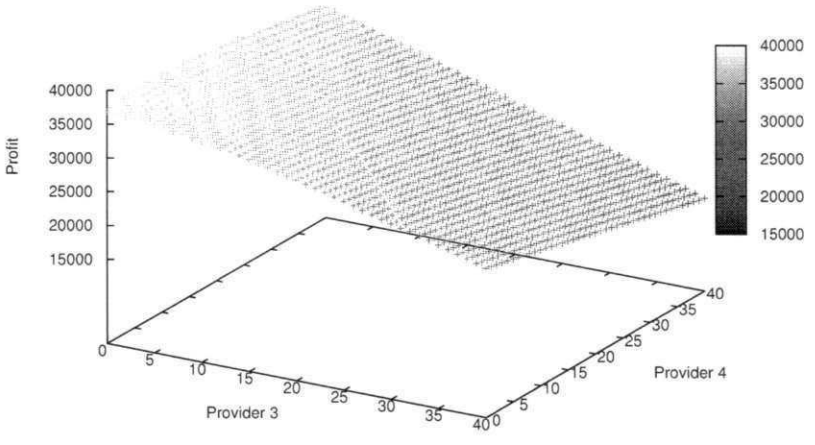
Na ausência de dados reais para definirmos funções de utilidade, seguimos a proposta de C. Lee e A. Snavely [Lee e Snavely 2007] para geração sintética de tais funções. Mais especificamente, definimos M_i como $(10 \cdot pr_2 \cdot d_i)$ (ou seja, a utilidade máxima é equivalente a 10 instâncias locais vezes a demanda da aplicação); e $td_i - tr_i = d_i$ (ou seja, se houver apenas um único recurso disponível em tr_i, a aplicação $\mathcal{A}_i$ finalizará exatamente no *deadline*).

4.5.3 Resultados Analíticos

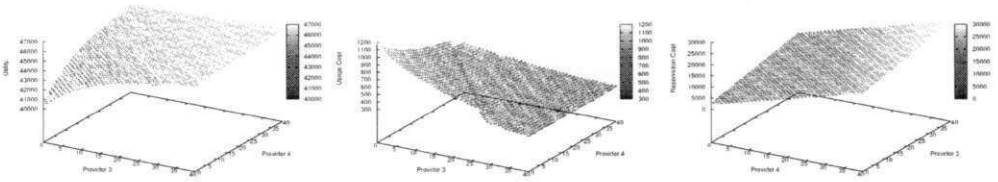
Na fase de planejamento, consideramos todos os valores possíveis para a quantidade de recursos reservada, variando de 0 a 40 em cada provedor. Em outras palavras, executamos uma busca exaustiva nos dois provedores, variando de um cenário com nenhuma reserva, até a quantidade máxima definida de 40 instâncias em cada um dos provedores. A saída desta fase é um conjunto com a quantidade de recursos que devem ser reservados nos provedores $\mathcal{K}_3$ e $\mathcal{K}_4$, para maximizar o lucro esperado.

Na Figura 4.1 plotamos o lucro, a utilidade e o custo obtidos para todas as combinações de valores de $\mathcal{K}_3$ e $\mathcal{K}_4$, levando em consideração o cenário *baseline*. Os eixos x e y representam, respectivamente, os recursos reservados nos provedores $\mathcal{K}_3$ e $\mathcal{K}_4$; enquanto o eixo z ilustra o valor obtido (lucro, utilidade ou custo) para cada combinação de (x, y). A partir da Figura 4.1(a) podemos notar a importância da fase de planejamento, visto que o lucro obtido pode variar significantemente dependendo da quantidade de recursos reservados. O lucro é de fato calculado da utilidade recebida pela execução da carga de trabalho usando tantos recursos quanto necessários da combinação (x, y) (Figura 4.1(b)), menos o custo de usar os recursos requeridos para finalizar a carga de trabalho (Figura 4.1(c)) mais o peço por reservar os recursos (Figura 4.1(d)).

Explicando o mais simples primeiro, o custo de reserva cresce linearmente em ambas as direções com o aumento na quantidade de recursos reservados, como esperado. Entretanto,



(a) Lucro



(b) Utilidade (ou rendimento) (c) Custo de utilização (d) Custo de reserva

Figura 4.1: Resultados do planejamento para o *baseline*, levando em consideração a possibilidade de reservar até 80 instâncias (40 no $\mathcal{P}_3$ e 40 no $\mathcal{P}_4$) e ainda utilizar até 20 instâncias sob demanda (no $\mathcal{P}_5$).

apesar desta linearidade no custo, podemos notar que o preço de reserva para o provedor $\mathcal{K}_3$ impacta mais no custo final visto que o seu preço de reserva é maior do que o do $\mathcal{K}_4$. Por outro lado, o custo de uso para os recursos reservados sofre maior influência pelo provedor $\mathcal{K}_4$. No geral, quanto mais recursos são usados do provedor $\mathcal{K}_4$, maior é o custo de uso para cada ponto fixo do provedor $\mathcal{K}_3$, como podemos observar no gráfico. Além disso, podemos notar uma descontinuidade na Figura 4.1(c). Isto acontece a partir do momento em que é mais vantajoso utilizar recursos dos provedores $\mathcal{P}_3$ e/ou $\mathcal{P}_4$, ao invés do provedor sob demanda $\mathcal{P}_5$, que apresenta um custo de uso mais alto. A utilidade obtida pela execução da carga de trabalho é quase plana a partir de certa quantidade de recursos reservados, porque a partir do momento que a carga de trabalho é computada por completa, não importa o quanto de recursos extra estão disponíveis se não são necessários. Em outras palavras, uma vez que a carga de trabalho é executada com $tc_i = tr_i$ para todas as aplicações A_i, a utilidade não crescerá mais. Assim, a utilidade é a máxima de um certo ponto em diante.

Para o cenário definido como *baseline* o algoritmo de planejamento propôs como reserva o conjunto $(0, 8)$; ou seja, o maior lucro foi obtido quando não foi reservada nenhuma instância do provedor $\mathcal{P}_3$ e com 8 instâncias reservadas no provedor $\mathcal{P}_4$.

Com o intuito de mensurar o impacto nos resultados do planejamento de erros nas estimativas da carga de trabalho, do preço no mercado sob demanda e na QoS da grade, executamos experimentos varrendo todos os parâmetros descritos na subseção anterior. Os valores da eficiência obtidos para a reserva $(0, 8)$ são apresentados na Tabela 4.1. Esta tabela apresenta apenas os resultados para os cenários nos quais os erros na predição do preço do serviço sob demanda e da QoS da grade P2P estão no conjunto $\{0\%, \pm 20\%\}$, uma vez que os outros resultados foram bastante similares.

Podemos ver a partir da tabela que erros na predição do preço do serviço sob demanda e da QoS da grade P2P trazem um impacto pequeno nos resultados obtidos; e que, quando temos uma boa estimativa da carga de trabalho ($\Delta\mathcal{W}$ é pequeno), o algoritmo de planejamento resulta em uma boa eficiência. Além disso, percebemos que o desempenho do algoritmo de planejamento é próximo do planejamento ótimo em todos os cenários avaliados.

Comparamos também as nossas estratégias de planejamento e escalonamento com algumas estratégias que usam abordagens simples de provisionamento de recursos ao invés de um processo de planejamento. Para isto, executamos experimentos comparando os resul-

Tabela 4.1: Eficiência da estratégia sugerida pelo algoritmo de planejamento $\mathcal{E}(0,8)$ para diferentes cenários, variando os erros na predição: da carga de trabalho, do preço do serviço sob demanda e da QoS oferecida pela grade P2P.

ΔW	Variação no preço de -20%	Variação no preço de 0%	Variação no preço de $+20\%$
	Erro na predição da grade de -20%		
-40%	96.82	96.97	97.12
-20%	99.55	99.62	99.69
0%	100.00	99.98	99.97
$+20\%$	99.75	99.69	99.62
$+40\%$	99.14	99.05	98.94
	Erro na predição da grade de 0%		
-40%	96.67	96.81	96.95
-20%	99.46	99.54	99.61
0%	100.00	100.00	100.00
$+20\%$	99.80	99.74	99.67
$+40\%$	99.25	99.16	99.07
	Erro na predição da grade de $+20\%$		
-40%	96.02	96.15	96.27
-20%	99.38	99.47	99.56
0%	100.00	100.00	100.00
$+20\%$	99.86	99.80	99.74
$+40\%$	99.33	99.26	99.17

tados da eficiência $\mathcal{E}(0,8)$ com sete outros cenários de reserva: $\mathcal{E}(0,0)$, $\mathcal{E}(34,0)$, $\mathcal{E}(0,34)$, $\mathcal{E}(17,17)$, $\mathcal{E}(14,0)$, $\mathcal{E}(0,14)$ e $\mathcal{E}(7,7)$. A primeira estratégia não reserva nada. As três próximas reservam um total de 34 instâncias, que é a capacidade média por unidade de tempo requerida para executar a carga de trabalho esperada no ano, menos a capacidade disponível localmente. As últimas três reservam um total de 14 instâncias e assumem que 20 instâncias podem ser adquiridas a partir do mercado sob demanda. Estas reservas apresentam estratégias "ingênuas", mas comumente utilizadas no provisionamento de recursos. Apresentamos os resultados nas Tabelas 4.2, 4.3 e 4.4, explicadas a seguir. Por simplicidade, estas Tabelas apresentam apenas os resultados para o preço do serviço sob demanda do *baseline*, que é a coluna do meio da Tabela 4.1.

A partir da Tabela 4.2 podemos observar que a eficiência obtida pelo algoritmo de pla-

nejamento através da reserva $(0, 8)$ é maior do que a eficiência obtida com a reserva $(0, 0)$ na maioria dos cenários avaliados. As exceções são os cenários onde a carga atual de trabalho é 40% menor do que a carga esperada e um cenário onde a carga é 20% menor. Nestes casos é possível alcançar uma maior eficiência utilizando apenas recursos locais, da grade e sob demanda, sem a necessidade de reservar nenhum recurso extra. Podemos explicar este comportamento simplesmente pelo fato de que, quanto menor for a demanda da carga de trabalho, menos são necessários parar executá-la. Contudo, mesmo nos cenários onde $\mathcal{E}(0, 0)$ é maior do que $\mathcal{E}(0, 8)$, a diferença não é tão grande entre as duas eficiências.

Tabela 4.2: Comparação entre a estratégia sugerida pelo algoritmo de planejamento $\mathcal{E}(0, 8)$ e a estratégia de não reservar nenhum recurso.

$\Delta\mathcal{W}$	$\mathcal{E}(0,0)$	$\mathcal{E}(0,8)$
Erro na predição da grade de -20%		
-40%	100.00	96.97
-20%	99.46	99.62
0%	98.12	99.98
$+20\%$	93.42	99.69
$+40\%$	91.02	99.05
Erro na predição da grade de 0%		
-40%	100.00	96.81
-20%	99.53	99.54
0%	98.21	100.00
$+20\%$	93.65	99.74
$+40\%$	91.29	99.16
Erro na predição da grade de $+20\%$		
-40%	100.00	96.15
-20%	99.68	99.47
0%	98.35	100.00
$+20\%$	93.88	99.80
$+40\%$	91.59	99.26

Na Tabela 4.3, podemos ver que os resultados da eficiência para a reserva do algoritmo de planejamento $(0, 8)$ são todos melhores do que os resultados apresentados pelas estratégias que reservam 34 instâncias. Para os cenários com $\Delta\mathcal{W} = -40\%$, as eficiências obtidas

usando a reserva $(0, 8)$ são mais do que o dobro das eficiências obtidas com a reserva $(34, 0)$. Além disso, todos os resultados para estratégias que reservam algum recurso do provedor $\mathcal{P}_3$ são piores do que os resultados das estratégias que reservam apenas recursos do provedor $\mathcal{P}_4$. Isto acontece porque $pr_3 + pu_3 > pr_4 + pu_4$ e $pr_3 > pr_4$; e como resultado, o custo total de reservar e utilizar uma instância de $\mathcal{P}_3$ é sempre maior do que o custo de reservar uma instância de $\mathcal{P}_4$ e utilizá-la durante o mesmo intervalo de tempo.

Tabela 4.3: Comparação entre a estratégia sugerida pelo algoritmo de planejamento $\mathcal{E}(0, 8)$ e as estratégias que reservam 34 instâncias.

ΔW	$\mathcal{E}(34, 0)$	$\mathcal{E}(0, 34)$	$\mathcal{E}(17, 17)$	$\mathcal{E}(0, 8)$
Erro na predição da grade de -20%				
-40%	45.44	77.40	61.58	96.97
-20%	62.20	86.94	74.83	99.62
0%	72.36	92.21	82.53	99.98
$+20\%$	79.54	95.59	87.84	99.69
$+40\%$	83.28	97.67	90.49	99.05
Erro na predição da grade de 0%				
-40%	45.11	76.82	61.03	96.81
-20%	62.07	86.73	74.64	99.54
0%	72.24	92.04	82.40	100.00
$+20\%$	79.47	95.51	87.76	99.74
$+40\%$	83.18	97.56	90.39	99.16
Erro na predição da grade de $+20\%$				
-40%	44.49	75.64	60.09	96.15
-20%	61.78	86.38	74.27	99.47
0%	72.14	91.95	82.28	100.00
$+20\%$	79.41	95.42	87.70	99.80
$+40\%$	83.07	97.43	90.26	99.26

Por fim, na Tabela 4.4 apresentamos os resultados da estratégia baseada no algoritmo de planejamento em comparação com as estratégias que reservam 14 instâncias de recursos. Como podemos observar, os resultados para a reserva $(0, 8)$ são todos melhores do que os resultados das estratégias que reservam recursos do provedor $\mathcal{P}_3$ — $(14, 0)$ e $(7, 7)$ — uma vez que o mesmo tem um alto custo de reserva. Porém, os resultados para o cenário de reserva

$(0, 14)$ são melhores para os casos onde a demanda da carga de trabalho foi subestimada. Para os casos onde a demanda é superestimada, ambas as reservas $(0, 14)$ e $(0, 8)$ têm eficiência próxima da ótima.

Tabela 4.4: Comparação entre a estratégia sugerida pelo algoritmo de planejamento $\mathcal{E}(0, 8)$ e as estratégias que reservam 14 instâncias.

$\Delta\mathcal{W}$	$\mathcal{E}(14, 0)$	$\mathcal{E}(0, 14)$	$\mathcal{E}(7, 7)$	$\mathcal{E}(0, 8)$
Erro na predição da grade de -20%				
-40%	80.03	93.25	86.65	96.97
-20%	87.50	97.80	92.66	99.62
0%	91.26	99.55	95.41	99.98
$+20\%$	93.18	99.97	96.58	99.69
$+40\%$	94.12	99.99	97.06	99.05
Erro na predição da grade de 0%				
-40%	79.94	92.99	86.47	96.81
-20%	87.36	97.63	92.50	99.54
0%	91.22	99.49	95.36	100.00
$+20\%$	93.19	99.97	96.58	99.74
$+40\%$	94.13	100.00	97.06	99.16
Erro na predição da grade de $+20\%$				
-40%	79.22	92.01	85.62	96.15
-20%	87.21	97.41	92.31	99.47
0%	91.15	99.41	95.28	100.00
$+20\%$	93.18	99.95	96.57	99.80
$+40\%$	94.13	100.00	97.07	99.26

Como podemos observar dos resultados apresentados, erros na predição de 20% no preço dos serviços sob demanda e na qualidade do serviço da grade não impactam significativamente nas eficiências obtidas. Estimativas ruins para a demanda de carga de trabalho podem levar a situações onde uma estratégia "ingênua" de provisionamento de recursos alcance melhores resultados. Contudo, para todos os cenários avaliados nos quais a demanda de carga de trabalho é a esperada ($\Delta\mathcal{W} = 0\%$), a abordagem proposta é mais eficiente em termos de lucratividade. Mais importante, em todos os cenários avaliados, o lucro obtido pela reserva encontrada no algoritmo de planejamento é próximo do lucro ótimo obtido em cada cenário,

com uma diferença máxima de aproximadamente 4%.

4.5.4 Resultados Simulados

O modelo analítico utilizado na subseção anterior apresentas fortes indícios de que um planejamento criterioso e um escalonamento eficiente podem melhorar substancialmente o lucro obtido ao computar aplicações científicas infraestrutura híbrida proposta. Entretanto, o modelo analítico traz fortes simplificações com relação à grade computacional, uma vez que não foi modelada a "volatilidade" dos recursos disponibilizados, ou pelo OurGrid, ou pelos provedores na nuvem. Uma vez escalonada uma tarefa na grade, o modelo assume que a mesma será finalizada sem interrupção. Além disso, consideramos uma carga de trabalho derivada de registros de grades computacionais dedicadas, cuja demanda pode variar muito daquela submetida às grades computacionais oportunistas, como o OurGrid. Para avaliar o impacto desta suposição nos resultados apresentados anteriormente, implementamos um modelo de simulação que utiliza uma modelagem de carga de trabalho mais realística. Este modelo também considera a volatilidade dos recursos que são providos pela grade P2P e pelo provedor de computação na nuvem. Os primeiros se tornam indisponíveis a qualquer momento (caso o usuário da máquina doada à grade passe a utilizá-la) e os últimos estão disponíveis apenas durante o período de duração do contrato. O modelo de simulação implementa as fases de planejamento e escalonamento dos recursos descritas nas seções anteriores. É importante ressaltar que testes de *software* foram realizados para validar o funcionamento do simulador implementado.

Modelo Simulado

Assim como no modelo analítico, na simulação os recursos computacionais são providos por máquinas que estão disponíveis na infraestrutura local, recebidas da grade computacional P2P e possivelmente adquiridas do provedor de computação na nuvem. Uma diferença fundamental do modelo analítico para o modelo simulado é que, no segundo, modelamos a carga de trabalho levando em consideração as principais características dos *Jobs* que são submetidos ao *OurGrid*. Além disso, instanciamos os parâmetros do modelo simulado com valores obtidos em estudos recentes que buscam caracterizar o comportamento de grades

computacionais oportunistas [Carvalho 2011]. É importante lembrar que, para o modelo analítico, instanciamos os parâmetros da carga de trabalho com valores obtidos a partir de registros reais de utilização em grades computacionais dedicadas.

Modelamos cada aplicação que forma a carga de trabalho $\mathcal{W} = \{\mathcal{A}_1, ..., \mathcal{A}_m\}$ por uma tupla $\mathcal{A}_i = \langle n_i, \bar{w}_i, tr_i, u_i(\Delta t)\rangle$, onde n_i é o número de tarefas paralelas independentes que compõem a aplicação e $\bar{w}_i$ é a média do tempo de processamento destas tarefas. Da mesma forma que no modelo analítico, tr_i e $u_i(\Delta t)$ representam, respectivamente, o instante de tempo dentro de Δt quando a aplicação está pronta para ser executada e a sua função de utilidade. Diferente do modelo analítico, a demanda computacional de cada aplicação $\mathcal{A}_i$ é representada no modelo simulado pela quantidade de tarefas paralelas a serem executadas e pelo tempo de processamento das tarefas. No modelo simulado implementado, consideramos que, para cada tarefa que compõe a carga total de trabalho, é sorteado o tempo de processamento da mesma segundo uma distribuição de probabilidade.

Outra diferença em relação ao modelo analítico é que, no modelo simulado, consideramos uma caracterização mais realística da grade P2P, onde as máquinas oriundas da grade possuem um comportamento modelado a partir de registros de utilização do próprio *OurGrid*. A modelagem deste comportamento leva em consideração: (i) como é contabilizado o crédito de doação para grade; (ii) o momento de chegada das máquinas da grade; e (iii) a volatilidade das máquinas devido ao uso por parte de seus usuários. Para a modelagem do crédito contabilizado (i), consideramos o valor de doação das máquinas locais em termos de minuto-máquina. Por exemplo, se a infraestrutura local dispões de 5 máquinas ociosas por um tempo de 5 horas, para este período, soma-se um crédito de $(1500 \cdot p_d \cdot p_c)$ minutos-máquina, onde p_d e p_c são, respectivamente, as probabilidades de existir uma demanda para o recurso que está sendo doado e, uma vez doado, de que o recurso foi consumido por um *peer* "colaborador", assim como foram definidas na Equação 4.6. Assumimos que os recursos da grade são solicitados no instante de chegada de um *Job* (ii) e que a quantidade de máquinas solicitada é menor ou igual ao crédito total de doação dividido pelo tempo médio de operação de uma máquina na grade (iii). Continuando o exemplo anterior para que fique ainda mais claro, considerando as 5 primeiras horas de doação e $(p_d \cdot p_c = 0.72)$, teremos um crédito de 1080 minutos-máquinas na chegada do primeiro *Job*. Considerando também que o tempo médio de operação de uma máquina da grade é de 30 minutos, o número total de máquinas

disponíveis na grade seria $(1080/30 = 36)$. Desta forma, até 36 máquinas poderiam ser utilizadas da grade computacional para executar o primeiro *Job*. Devemos notar que isto não significa um aumento efetivo de 36 máquinas disponíveis, uma vez que existe a volatilidade das mesmas. No modelo simulado desenvolvido, para cada uma destas máquinas na grade é sorteado, a partir de uma distribuição probabilística, o seu tempo de vida (operação). Caso este tempo seja menor do que o tempo de processamento da tarefa alocada para esta máquina, a mesma será "devolvida" ao seu usuário, a tarefa será preemptada e retornará para a fila de execução. Uma vez que uma tarefa é preemptada, um novo recurso pode ser solicitado da grade caso exista crédito disponível.

Para avaliar os resultados do modelo simulado utilizamos a métrica de eficiência definida na Equação 4.12.

Descrição dos Cenários Avaliados

O modelo de simulação implementa o problema de otimização definido na Seção 4.2 e os cenários de avaliação descritos na Subseção 4.5.2, com algumas alterações que serão explicadas a seguir.

Assumimos um sistema com 4 provedores, compreendendo: uma grade P2P ($\mathcal{P}_1$), a infraestrutura local dedicada ($\mathcal{P}_2$) e 2 provedores de computação na nuvem, um oferecendo serviços a partir de reservas ($\mathcal{P}_3$) e um oferecendo serviços sob demanda ($\mathcal{P}_4$). Por simplificação e diferente do modelo analítico, não consideramos dois provedores de computação na nuvem oferecendo recursos a partir de reserva. Pelos resultados obtidos com o otimizador analítico, temos indícios de que, para os cenários avaliados analiticamente, é melhor contratar recursos dos provedores que oferecem melhores preços de reserva e que o preço de uso não impacta tanto nos valores contratados. Desta forma, a otimização descrita na seção anterior busca pelo melhor cenário de reserva no provedor de computação na nuvem que oferece este serviço ($\mathcal{P}_3$). Seguindo as restrições impostas pela *Amazon.com* para negociação automática de instâncias no EC2, consideramos que até 20 instâncias podem ser reservadas pelo período de 1 ano a frente.

Modelamos uma flutuação nos preços futuros através do erro na predição dos mesmos e assumimos o preço praticado pelo serviço Amazon EC2 na época de execução dos experimentos (USD$0.08) para o *baseline*, e para os cenários variados a partir do *base-*

line assumimos um erro na predição do custo do serviço sob demanda dentro do conjunto $\{-20\%, 0\%, +20\%\}$. Consideramos que o preço do uso sob demanda é também constante ao longo do tempo. Modelamos o provedor de reserva também seguindo o esquema de tarifação empregado no serviço EC2 da *Amazon.com* na época dos experimentos. Consideramos em nossos experimentos que a infraestrutura local é composta por 2 máquinas dedicadas e que os recursos disponíveis a partir do mercado sob demanda a cada intervalo de tempo são limitados a 20 instâncias, seguindo novamente as restrições impostas pela *Amazon.com* para negociação automática de instâncias no EC2.

Assumimos que a carga de trabalho executada na infraestrutura proposta é compreendida de aplicações científicas de processamento intensivo e do tipo *Bag-of-Tasks*. Para estas aplicações, definimos a chegada de uma única aplicação diária e que o número de tarefas (n_i) de todas as aplicações $\mathcal{A}_i$ que formam a carga de trabalho é fixo em cada cenário simulado. Consideramos que para o *baseline* o número de tarefas é igual a 100 e, desta maneira, obtemos a carga de trabalho utilizada na fase de escalonamento pela introdução de um erro percentual fixo no número total de tarefas a serem executadas n_i. Os valores que utilizamos para "perturbar" a carga de trabalho estão dispostos no conjunto $\{-40\%, -20\%, 0\%, +20\%, +40\%\}$, ou seja, *Jobs* com 60, 80, 100, 120 e 140 tarefas, respectivamente. Por simplicidade, escolhemos fixar o número de tarefas das aplicações que compõem a carga total de trabalho e introduzir aleatoriedade através de tempos de processamento de cada tarefa. Com isto, controlamos o fator aleatório nos cenários avaliados, para uma comparação justa entre os mesmos.

O tempo de processamento de cada tarefa que compõe uma aplicação $\mathcal{A}_i$ é atribuído, através de uma distribuição normal de probabilidade, no instante de chegada de cada *Job* diário. Consideramos um tempo médio de processamento $\bar{w}_i$ de 47 minutos e um desvio padrão de 26 minutos. Estes valores foram modelados a partir de registros de utilização de grades computacionais oportunistas [Carvalho 2011].

Assumimos que a utilidade obtida pela execução de cada aplicação $\mathcal{A}_i$ tem um valor máximo M_i quando $tc_i = tr_i$ e decai linearmente a medida que tc_i aumenta, chegando ao valor zero em $tc_i = td_i$, onde td_i é o *deadline* do aplicação $\mathcal{A}_i$. Formalizando esta função de utilidade temos:

$$u_i(tc_i - tr_i) = max\{0, M_i \cdot (1 - (tc_i - tr_i)/(td_i - tr_i))\}. \quad (4.14)$$

Na ausência de dados reais para definirmos funções de utilidade, seguimos a proposta de C. Lee e A. Snavely [Lee e Snavely 2007] para geração sintética de tais funções. Mais especificamente, definimos M_i como $(n_i \cdot pr_3 \cdot 1.5)$, ou seja, a utilidade máxima é equivalente a reservar, no provedor de computação na nuvem, o número de instâncias necessários para computar o *Job* (aplicação $\mathcal{A}_i$) vezes um fator de ineficiência de 1.5. Definimos também que o *deadline* de uma aplicação $\mathcal{A}_i$ é fixo e tem um valor de $td_i - tr_i = 840$ minutos (14 horas).

Para a grade P2P, assumimos que $p_c \cdot p_d = 0.72$ durante a fase de planejamento da capacidade do *baseline*. Durante o escalonamento em tempo de execução, entretanto, consideramos um erro na estimativa de $p_c \cdot p_d$, dentro do seguinte conjunto de valores $\{-20\%, 0\%, +20\%\}$. O valor 0.72 foi estimado em Carvalho et al. [Carvalho et al. 2010] a partir de registros de execução de aplicações reais. O tempo de operação (tempo de vida) das máquinas da grade é modelado a partir de uma distribuição uniforme entre 15 e 25 minutos, o que leva a um tempo médio de operação de 20 minutos. Durante todo o período de simulação em que as máquinas locais não estiverem sendo utilizadas para executar um *Job* as mesmas são doadas para a grade P2P.

Avaliação dos Resultados do Modelo Simulado

Cada cenário de simulação foi executado 30 vezes para extração dos valores médios de eficiência. Para decidir que tipo de teste estatístico deveríamos utilizar para comparar os resultados, avaliamos se os valores das eficiências para cada experimento seguiam uma distribuição normal. Executamos o teste *Shapiro-Wilk* [Shapiro e Wilk 1965] e constatamos a normalidade dos dados obtidos. Utilizamos o teste *t-test* [Jain 2008] para extrair os valores médios e os intervalos de confiança das eficiências em cada cenário avaliado, com um nível de confiança de 95%.

Para o cenário definido como *baseline*, o algoritmo de planejamento propôs que (6) instâncias fossem reservadas; ou seja, o maior lucro foi obtido quando foram reservadas 6 instâncias junto à $\mathcal{P}_3$. A Tabela 4.5 apresenta os valores da eficiência para os cenários avaliados. Podemos ver a partir da tabela que impacto na eficiência devido à variação nos erros da predição da grade P2P e do preço dos recursos sob demanda é pequeno. Além disso,

da mesma forma que no modelo analítico, uma grande variação na carga de trabalho esperada não impacta significativamente a eficiência obtida. Por estes resultados, conjecturamos novamente que os algoritmos de planejamento e escalonamento propostos são robustos em relação às incertezas inerentes ao ambiente.

Tabela 4.5: Eficiência da estratégia sugerida pelo algoritmo de planejamento $\mathcal{E}(6)$ para diferentes cenários, variando os erros na predição: da carga de trabalho, do preço do serviço sob demanda e da QoS oferecida pela grade P2P.

ΔW	Variação no preço de -20%	Variação no preço de 0%	Variação no preço de $+20\%$
	Erro na predição da grade de -20%		
-40%	95.03	97.17	97.39
-20%	100.00	96.86	97.08
0%	100.00	100.00	100.00
$+20\%$	97.96	97.19	96.36
$+40\%$	96.26	95.29	94.27
	Erro na predição da grade de 0%		
-40%	95.02	97.17	97.39
-20%	100.00	96.86	97.08
0%	100.00	100.00	100.00
$+20\%$	97.96	97.19	96.36
$+40\%$	96.26	95.29	94.26
	Erro na predição da grade de $+20\%$		
-40%	95.02	97.17	97.39
-20%	100.00	96.86	97.08
0%	100.00	100.00	100.00
$+20\%$	97.96	97.19	96.36
$+40\%$	96.26	95.29	94.27

Realizamos testes estatísticos (*t-test*) para comparar os valores obtidos quando variamos o erro na predição da grade (ou no preço do recurso sob demanda) e confirmar se são realmente diferentes. Para todos os casos, a hipótese de que a diferença das médias não é igual a 0 foi confirmada verdadeira, com um nível de significância de 0.05.

A Tabela 4.6 apresenta uma comparação entre o cenário de reserva sugerido pelo planejamento (6 instâncias) e casos extremos como: não reservar nada ou reservar o que for possível junto ao provedor de computação nuvem. Pelos resultados pode-se novamente in-

ferir que um planejamento criterioso da infraestrutura híbrida traz vantagens econômicas em sua operacionalidade por um período de um ano.

Tabela 4.6: Comparação entre a estratégia sugerida pelo algoritmo de planejamento $\mathcal{E}(6)$ e as estratégias de não reservar nenhum recurso ou reservar tudo.

ΔW	$\mathcal{E}(0)$	$\mathcal{E}(20)$	$\mathcal{E}(6)$
Grid Prediction Error of -20%			
-40%	80.25	37.59	96.17
-20%	80.64	63.69	96.86
0%	78.16	74.30	100.00
$+20\%$	72.95	77.93	97.19
$+40\%$	66.58	78.97	95.29
Grid Prediction Error of 0%			
-40%	80.18	38.46	96.17
-20%	80.70	63.93	96.86
0%	78.16	74.38	100.00
$+20\%$	72.93	77.94	97.19
$+40\%$	66.54	79.02	95.29
Grid Prediction Error of $+20\%$			
-40%	80.23	44.85	96.17
-20%	80.73	63.97	96.86
0%	78.23	74.41	100.00
$+20\%$	72.89	77.97	97.19
$+40\%$	66.52	79.02	95.29

4.6 Considerações a Respeito do Gerenciamento a Longo Prazo

Apresentamos neste capítulo uma estratégia para o planejamento e escalonamento da capacidade a longo prazo da infraestrutura híbrida. Pelos resultados obtidos, concluímos que esta estratégia pode alcançar uma maior eficiência do que algumas estratégias que usam abordagens mais simples de provisionamento. Além disso, o algoritmo proposto alcança resultados sempre próximos do melhor resultado para um dado cenário, mesmo com estimativas ruins

para a carga de trabalho, o preço do serviço sob demanda e a qualidade de serviço da grade P2P.

Como trabalho futuro pretendemos caracterizar melhor a demanda da carga de trabalho, com o objetivo de instanciar melhor os parâmetros do modelo. Além disso, pretendemos avaliar como informações mais precisas dos recursos disponíveis podem ser utilizadas no processo de escalonamento para melhorar a eficiência. Por exemplo, como instâncias do *spot-market* da Amazon EC2 [Amazon AWS 2010] poderiam ser utilizadas para executar as aplicações e como estratégias diferentes de ofertar o *spot-price* poderiam levar em consideração os preços atuais de reserva e sob demanda, bem como a qualidade de serviço da grade.

Capítulo 5

Gerenciamento a Curto Prazo

5.1 Introdução

No capítulo anterior propomos uma abordagem de gerenciamento dirigida a negócios para maximizar a lucratividade obtida a partir da execução de aplicações com restrições de tempo em uma infraestrutura híbrida de computação na nuvem, envolvendo contratos de longa duração com provedores. Esta infraestrutura suporta a execução de aplicações que possuem uma função de utilidade associada. A execução de uma aplicação não tem nenhuma utilidade até que seja finalizada, e execuções que finalizam o quanto antes podem ganhar mais utilidade do que execuções que finalizam mais tarde. Além disso, existe um *deadline* no qual as aplicações devem ser finalizadas. Caso contrário, uma penalidade (ou utilidade negativa) pode ser imposta. Quando a capacidade necessária para computar a aplicação é maior do que a capacidade local dedicada, recursos podem ser obtidos a partir de provedores de computação na nuvem capazes de oferecer IaaS. Por outro lado, quando existem mais recursos locais disponíveis do que o necessário para computar a aplicação, uma opção é doar estes recursos para uma grade P2P visando aumentar quantidade provável de recursos disponíveis na grade em um futuro próximo.

No estudo anterior, contudo, consideramos que recursos de provedores de computação na nuvem podem ser adquiridos através de reservas em contratos de longa duração e negociados sem intervenção humana. Em contraste aos recursos sob demanda que podem ser adquiridos sem compromisso prévio entre cliente e provedor, estes recursos reservados têm uma alta probabilidade de estarem disponíveis quando necessários, o que torna-se um requisito

crucial para diversas aplicações com restrições de tempo. Por outro lado, reservar uma quantidade muito grande de recursos por um longo período de tempo pode incorrer em um custo desnecessário para execução das aplicações, diminuindo assim a eficiência da infraestrutura híbrida em termos de lucratividade.

Como mencionamos anteriormente, o modelo de tarifação utilizado pelos provedores de computação na nuvem é de fundamental importância para o gerenciamento dos contratos que devem ser estabelecidos entre provedores e clientes. Atualmente, diversos modelos de tarifação são empregados neste mercado de computação utilitária, desde os mais simples onde o cliente paga à medida que for utilizando os recursos, até os mais complexos envolvendo uma negociação a partir de que o cliente está disposto a pagar pelo recurso. Outra característica importante é a QoS envolvida com a disponibilidade dos recursos. Modelos baseados em reserva são adotados por provedores como uma forma de garantir para os clientes, antecipadamente, a disponibilidade dos recursos. Neste capítulo apresentamos um modelo de tarifação baseado em reserva que envolve contratos de curta duração entre provedores e clientes. É importante salientar que propomos primeiramente este modelo no artigo [Maciel Jr. et al. 2008], quando, até então, nenhum provedor público de computação na nuvem empregava a reserva de recursos como modelo de tarifação. Uma versão estendida deste trabalho está disponível no artigo [Maciel Jr. et al. 2012].

O modelo de tarifação proposto neste capítulo é baseado em reservas de curta duração, onde o preço final é composto por uma taxa de reserva cujo valor varia dependendo da urgência do contrato, e uma taxa de uso a partir da utilização efetiva dos recursos reservados. Acreditamos que este modelo proporciona uma maior flexibilidade na hora de estabelecer os contratos e que, com o aumento na procura pela computação na nuvem e, consequentemente, na competição entre os diversos provedores, modelos como este se tornarão cada vez mais comuns neste tipo de mercado.

O restante deste capítulo está organizado da seguinte maneira. Na Seção 5.2 instanciamos a aplicação e os elementos que compõem a infraestrutura híbrida de TI para contratos de curta duração; e descrevemos formalmente o problema encontrado. Apresentamos heurísticas para o planejamento dos contratos na Seção 5.3; e uma avaliação destas heurísticas em termos de lucratividade na Seção 5.4. Apresentamos algumas considerações finais sobre o planejamento de contratos de curta duração na Seção 5.5.

5.2 Instanciação do Modelo e Definição do Problema

Nesta seção apresentamos uma instanciação para o modelo do sistema e a definição do problema atacado no contexto do gerenciamento a curto prazo. Primeiramente, assumimos algumas ligeiras modificações do modelo do sistema proposto anteriormente. A primeira modificação diz respeito a como são representados os elementos que compõem a infraestrutura híbrida. Diferente do modelo apresentado no Capítulo 3, a infraestrutura local e a grade P2P são representadas pelo número de ciclos computacionais disponíveis por instante de tempo, como definido abaixo. A outra modificação envolve o modelo de tarifação que, para o gerenciamento a curto prazo, foi necessário introduzir o tempo de estabelecimento de cada contrato com um provedor de computação na nuvem. Também de forma diferente do modelo anterior, contratos serão estabelecidos para as aplicações que formam a carga de trabalho, ao invés de contratos por provedores. Por fim, apresentamos também uma definição do problema de escalonamento que precisamos resolver.

5.2.1 A Infraestrutura Híbrida de TI e o Modelo de Tarifação

Definimos uma infraestrutura local dedicada e uma grade computacional P2P que, durante um período de tempo Δ, é capaz de prover $\mathcal{L}(\Delta)$ e $\mathcal{G}(\Delta)$ ciclos computacionais, tal que:

$$\mathcal{L}(\Delta) = \int_{\Delta} i(t) \cdot dt, \tag{5.1}$$

$$\mathcal{G}(\Delta) = \int_{\Delta} g(t) \cdot dt, \tag{5.2}$$

onde $i(t)$ e $g(t)$ representam, respectivamente, o número de ciclos disponíveis no instante de tempo t nas máquinas locais dedicadas e na grade P2P.

Assumimos que todos os ciclos obtidos do mercado de computação na nuvem são adquiridos de um único provedor que oferece um modelo de tarifação com reservas de curta duração. Dessa forma, consideramos um sistema com 3 provedores, compreendendo: uma grade P2P ($\mathcal{P}_1$), a infraestrutura local dedicada ($\mathcal{P}_2$) e 1 provedor de computação na nuvem ($\mathcal{P}_3$).

O custo total de operar a infraestrutura híbrida durante um período de tempo Δ é a soma dos custos de manter os recursos locais, doar/receber recursos de/para uma grade P2P, e de

usufruir os recursos providos na nuvem. Definimos que o custo de manter os recursos locais dedicados é formado a partir de um custo fixo para cada ciclo computacional disponível na infraestrutura, se utilizado ou não, representado pelo custo de reserva por ciclo (pr_2). O custo de manter a infraestrutura dedicada durante Δ é, portanto[1]

$$\gamma_{\mathcal{L}}(\Delta) = pr_2 \cdot \int_{\Delta} i(t)dt. \tag{5.3}$$

Naturalmente, existe um custo de doar recursos para a grade; principalmente devido ao consumo extra de energia e a qualquer esforço extra necessário para garantir a segurança antes que programas possam ser executados em favor de outras pessoas, utilizando capacidade ociosa da infraestrutura local do doador. Todavia, estes custos são muitas vezes bem menores se comparados a outros custos de operação e manutenção da infraestrutura local (por exemplo, hardware, software, hospedagem e administração de sistemas, etc). Portanto, consideramos novamente desprezível o custo de doar ciclos ociosos para a grade.

Os custos de obter recursos de computação na nuvem depende do modelo de tarifação empregado pelo provedor. Na época em que este trabalho foi realizado, existiam poucos provedores de computação na nuvem que disponibilizavam um procedimento de negociação de recursos completamente automatizado. A maioria usava um modelo de tarifação simples no qual instâncias (ou máquinas) virtuais eram "levantadas" e desligadas de acordo com as necessidades dos clientes, que pagavam um preço fixo para cada hora ou fração de hora efetivamente utilizada. Nenhuma reserva era necessária, mas apenas um número limitado de máquinas virtuais poderiam ser instanciadas simultaneamente por um único cliente. Caso o cliente precisasse de um número maior de máquinas virtuais, este deveria ter contratado previamente através de uma negociação "homem-a-homem" com o provedor [Amazon AWS 2010].

Contudo, alguns provedores de computação na nuvem, tal como a própria Amazon, disponibilizam atualmente um modelo de tarifação que incorpora a noção de reservas a longo prazo de recursos antes de sua utilização. Neste caso, um preço mais baixo é cobrado pela utilização dos recursos se o cliente tiver previamente reservado-os [Vogels 2009]. O mercado de computação na nuvem está evoluindo rapidamente e acreditamos que é uma suposi-

[1]Lembramos que o custo de uso para o provedor local não é considerado ($pu_2(t) = 0$), uma vez que assumimos o custo total de operação da infraestrutura local representado pelo seu custo de reserva.

ção fraca considerarmos que os modelos de tarifação empregados atualmente se tornarão os padrões para o futuro. De fato, a utilização de reservas pela Amazon é uma indicação que, à medida que cada vez mais clientes entram neste mercado, provedores buscarão por mais informações a respeito da provável carga de trabalho dos clientes, visando uma operação eficiente de suas infraestruturas. Predizemos que reservas eventualmente se tornarão uma característica comum nos modelos de tarifação de serviços na nuvem e clientes estarão cada vez mais interessados em interfaces que proveem alguma automação no processo de reserva e negociação dos contratos. Também acreditamos que a competição entre provedores forçará que estes disponibilizem a seus clientes mais flexibilidades na hora de estabelecer contratos.

Baseados nestas predições, definimos um modelo de tarifação que julgamos ser o mais provavelmente adotado nos cenários que consideramos neste trabalho, onde agentes planejadores estabelecem automaticamente contratos de curta duração em favor dos seus clientes. O modelo de tarifação inclui dois tipos de taxas: um preço pela reserva dos recursos e outro pelo consumo. Além disso, é razoável pensar que a urgência de um contrato é um fator crítico para estabelecer a taxa de reserva dos recursos. Em geral, contratos que são estabelecidos somente pouco antes dos recursos serem consumidos tendem a terem custos de reserva por ciclo mais expressivos do que contratos estabelecidos com maior antecedência. Para um tempo fixo de reserva e consumo, consideramos que o custo de reserva é diretamente proporcional ao número de ciclos que estão sendo reservados. O número máximo de ciclos que serão consumidos em um determinado tempo t é o mesmo reservado para o tempo t, podendo, entretanto, ser menor ou igual à quantidade reservada. O preço de reserva é cobrado para todos os ciclos reservados, se consumidos ou não. Porém, o preço de utilização dos recursos é cobrado apenas pelos ciclos efetivamente usados. Assumimos que o preço de consumo associado a um contrato é diretamente proporcional ao número de ciclos consumidos naquele contrato. É importante notar que este modelo de tarifação é semelhante ao que a Amazon disponibiliza (e que foi utilizado no capítulo anterior), mas, em nosso trabalho, consideramos contratos para períodos de tempo mais curtos do que os negociados pela Amazon (um dia ao invés de um ano, por exemplo) e que nossos contratos são estabelecidos automaticamente independente da quantidade de ciclos reservada. Para modelar esta reserva de curta duração e o sua forma de tarifação vamos alterar o modelo de contratos que podem ser estabelecidos com o provedor de computação na nuvem, definido no Capítulo 3 (Subseção

3.2.2).

Consideramos que diariamente n_i contratos podem ser estabelecidos apenas com o provedor na nuvem ($\mathcal{P}_3$) e representamos o conjunto desses contratos como uma tupla $\mathcal{K}_i = \langle\langle \mathcal{K}_1^i, ts_1^i, tb_1^i, \beta_1^i\rangle, ...\langle \mathcal{K}_{n_i}^i, ts_{n_i}^i, tb_{n_i}^i, \beta_{n_i}^i\rangle\rangle$, onde $\mathcal{K}_j^i$ é o número de ciclos reservados junto à $\mathcal{P}_3$ no contrato j, para execução de $\mathcal{A}_i$; ts_j^i é o tempo no qual o contrato j é estabelecido; tb_j^i é o tempo no qual os ciclos estão disponíveis para serem utilizados; e β_j^i é um número entre 0 e 1 que expressa um relacionamento entre as taxas de reserva e consumo por ciclo, que é utilizado para calcular o custo dos ciclos da forma detalhada em seguida. Visto que não faz sentido para o cliente reservar ciclos para utilização fora do intervalo $[tr_i, td_i]$ para qualquer aplicação $\mathcal{A}_i$, consideramos que tb_j^i encontra-se neste intervalo de tempo para todos os contratos estabelecidos junto à $\mathcal{P}_3$ ($\mathcal{K} = \{\mathcal{K}_i,\ \forall i \in \{1, m\}\}$). Quando um contrato é estabelecido, o provedor concorda em cumprí-lo e o cliente (via seu agente automático) concorda em pagar o preço combinado. Assumimos que todos os contratos em conformidade com os interesses do provedor e cliente são estabelecidos com sucesso e devidamente honrados pelas partes envolvidas.

Utilizamos o seguinte modelo para o custo de c ciclos conforme o contrato $\mathcal{K}_j^i$, onde $0 \leq c \leq \mathcal{K}_j^i$. Denotamos pr_3 como o custo total para o cliente (composto por ambas as taxas, reserva e consumo) de reservar um único ciclo no tempo tp_i e consumí-lo até td_i, onde tp_i é o mais cedo que um contrato pode ser estabelecido para executar a aplicação $\mathcal{A}_i$, e td_i é o *deadline* para sua execução. Em outras palavras, consideramos um preço fixo por ciclo (pr_3) que abrange o custo de reserva do mesmo. Assumimos que este custo é independente da escolha do contrato sob o qual esta reserva é feita. (Obviamente este contrato tem que ser $\langle 1, tp_i, td_i, \beta_j^i\rangle$ para algum β_j^i, então a sentença anterior significa que o custo pr_3 é independente do valor deste β_j^i.) Definimos $\varphi_j^i(ts_j^i, tb_j^i)$ como uma função do tempo de estabelecimento do contrato j e do tempo de consumo, que reflete como os custos de reserva variam a partir destes tempos. Consideramos $\varphi_j^i(ts_j^i, tb_j^i)$ como uma função estritamente decrescente da diferença entre os tempos de consumo e estabelecimento do contrato. Assim, esta função é maior para contratos mais urgentes e assume um valor 1 quando esta diferença é tão grande quanto possível, ou seja, $\varphi_j^i(tp_i, td_i) = 1$ (quando $ts_j^i = tp_i$ e $tb_j^i = td_i$). Dessa forma, diferentes perfis de provedores podem ser modelados a partir de diferentes instanciações para β_j^i, pr_3 e $\varphi_j^i(ts_j^i, tb_j^i)$. Com isso, o custo de utilizar c ciclos sob um contrato

j estabelecido com o provedor de computação na nuvem para execução da aplicação $\mathcal{A}_i$ é dado por:

$$\gamma_j^i(c) = pr_3 \cdot \{[\beta_j^i \cdot \mathcal{K}_j^i \cdot \varphi_j^i(ts_j^i, tb_j^i)] + [(1 - \beta_j^i) \cdot c]\}, \tag{5.4}$$

que representa a soma dos custos de reservar $\mathcal{K}_j^i$ ciclos e consumir c dos ciclos reservados.

Notamos que, quando ambos os custos (reserva e consumo) são considerados, não é sempre verdade que dado quaisquer dois contratos estabelecidos ao mesmo tempo, (i) se os dois reservam o mesmo número de ciclos, é mais barato usar o contrato menos urgente, ou que (ii) se os dois reservam os ciclos para consumirem ao mesmo tempo, e reservam ao menos tantos ciclos quanto são utilizados naquele tempo, é mais barato usar o contrato que reserva menos ciclos. Como contra-exemplos, suponhamos que $tr_i < t_1 < t_2 < td_i$ e temos 3 contratos: $\langle 1, tp_i, t_1, 0.25\rangle$, $\langle 1, tp_i, t_2, 0.875\rangle$ e $\langle 2, tp_i, t_2, 0.25\rangle$. Então $\gamma_2(1)$ é maior do que ambos $\gamma_1(1)$ e $\gamma_3(1)$. Contudo, dado quaisquer dois contratos estabelecidos ao mesmo tempo e com o mesmo valor de β_j^i, (i) e (ii) acontecem. Permitimos esta flexibilidade em nosso modelo de tarifação geral, mas nas seções de avaliação assumimos que o provedor de serviços fixa um valor β e permite apenas o estabelecimento de n_i contratos nos quais $\beta_j^i = \beta$ $\forall i \in \{1, m\}$ e $\forall j \in \{1, n_i\}$. Dessa forma, em nossas avaliações (i) e (ii) acontecem; e a escolha do contrato pelo cliente corresponde em decidir: o tempo de estabelecimento do contrato, o tempo que os ciclos reservados serão consumidos, e o número de ciclos reservados. Notamos também que, se $\beta = \beta'$, $ts_j^i = ts_j^{i'}$ e $tb_j = tb_j^{i'}$, então para todo $c \leq \mathcal{K}_j^i$, $c' \leq \mathcal{K}_j^{i'}$ temos $\gamma_j^i(c) + \gamma_j^{i'}(c') = \gamma_j^{i''}(c + c')$, onde o lado direito da igualdade representa o contrato $\langle(\mathcal{K}_j^i + \mathcal{K}_j^{i'}), ts_j^i, tb_j^i, \beta\rangle$. Acontece que, quando calculamos custos em nossas avaliações assumimos sem perda de generalidade que dado qualquer par de tempos (t_1, t_2), ao menos um contrato é estabelecido no tempo t_1 para ciclos serem consumidos no tempo t_2.

Consideramos que o cliente (que é o gerente da infraestrutura híbrida) executa um agente de planejamento [Thain, Tannenbaum e Livny 2005] responsável pelo estabelecimento dos contratos com o provedor de computação na nuvem. Este agente planejador inicia a operação no tempo tp_i, que é o tempo mais cedo possível para o estabelecimento de contratos para aplicação $\mathcal{A}_i$. Assumimos que o planejamento dos contratos pertencentes à $\mathcal{K}_i$ é o resultado da execução do agente planejador para o período de tempo $\Delta t_i = [tp_i, td_i]$. $\mathcal{K}_i$ é um conjunto que contém os contratos que foram estabelecidos entre o cliente e o provedor

de computação na nuvem para cada uma das aplicações $\mathcal{A}_i$ pertencentes à carga de trabalho $\mathcal{W}$. Como consideramos uma única aplicação diária e contratos de curta duração, $\mathcal{K}$ é renovado diariamente para atender a cada uma das m aplicações que formam a carga de trabalho. Aproveitando uma generalização do modelo apresentado no Capítulo 3 onde vários contratos podem ser estabelecidos para execução de cada uma das aplicações $\mathcal{A}_i$, definimos $\mathcal{S} = \{uc_j^i(t), \forall i, j \mid 1 \leq i \leq m, 1 \leq j \leq n_i, tb_j^i \leq t \leq tc_j^i\}$ como o registro de utilização para execução da carga de trabalho $\mathcal{W}$ a partir do planejamento $\mathcal{K}$; onde $uc_j^i = \int_{tb_j^i}^{te_j^i} uc_j^i(t)dt$ representa todos os ciclos consumidos no contrato j pela execução da aplicação $\mathcal{A}_i$, desde o início do consumo (tb_j^i) até a sua completude (te_j^i). O custo incorrido dos ciclos reservados por um planejamento $\mathcal{K}$ e utilizados como indicado por $\mathcal{S}$ é dado por:

$$\gamma(\mathcal{W}, \mathcal{K}, \mathcal{S}) = \sum_{i=1}^{m} \sum_{j=1}^{n_i} \gamma_j^i(uc_j^i). \tag{5.5}$$

Daqui em diante, usaremos a notação t^- (respectivamente t^+) referindo-se como o instante de tempo que é infinitesimalmente anterior (respectivamente posterior) a algum instante de tempo t. Durante o intervalo de tempo $[tp_i, tr_i)$, nenhum ciclo é consumido da grade ou do provedor na nuvem. Isto acontece porque não existe computação crítica para ser executada neste intervalo de tempo e, portanto, não é necessário para o cliente buscar recursos na grade ou no provedor externo. Durante este período, qualquer ciclo ocioso da infraestrutura local dedicada é oferecido à grade P2P. Por simplicidade, assumimos que para qualquer t com $tp_i \leq t < tr_i$, todos os $\int_{tp_i}^{tr_i^-} i(t) \cdot dt$ ciclos estão ociosos e, portanto, são doados para a grade.

Consideramos agora o número de ciclos que estarão disponíveis durante Δ a partir das três componentes da infraestrutura híbrida. Definindo tc_i como o tempo que a aplicação $\mathcal{A}_i$ completa sua execução, então o número de ciclos que estarão disponíveis da infraestrutura local para executar a aplicação é dado por $\int_{tr_i}^{tc_i} i(t)dt$.

Por outro lado, o número de ciclos que estarão disponíveis a partir da grade P2P durante Δt_i dependerá da quantidade de recursos que foram doados previamente e da qualidade de serviço que a grade é capaz de oferecer. Esta qualidade de serviço está relacionada com a probabilidade de receber de volta os "favores" oferecidos a outros pares na grade dentro de um certo horizonte de tempo. Estes favores são doações de ciclos. Tão rapidamente os favores são pagos de volta, melhor é a qualidade de serviço da grade. Neste trabalho

definimos a qualidade de serviço da grade provida para a aplicação $\mathcal{A}_i$ como:

$$\Phi_i = \frac{\int_{tr_i}^{td_i} g(t)dt}{\int_{tp_i}^{tr_i^-} i(t)dt}. \tag{5.6}$$

Finalmente, o número de ciclos disponíveis a partir do provedor na nuvem no tempo t é definido pelos contratos existentes no planejamento $\mathcal{K}$, e é dado por:

$$\sum_{\{\mathcal{K}_i \in \mathcal{K} | tb_j^i = t\}} \mathcal{K}_j^i. \tag{5.7}$$

5.2.2 Definição do Problema

O problema que precisamos resolver é como escalonar $\mathcal{A}_i$ na infraestrutura híbrida, tal que o *lucro* de seu proprietário seja maximizado. Definimos o lucro da execução de uma aplicação $\mathcal{A}_i$ na infraestrutura híbrida como a diferença entre a utilidade obtida por executar $\mathcal{A}_i$ e o custo de operar a infraestrutura durante Δt_i.

Novamente por simplicidade, assumimos que a aplicação tem uma carga de trabalho com uma granularidade extremamente "fina" e suficientemente paralela, ao ponto que, em qualquer instante de tempo t é possível, se necessário, escalonar tarefas simultaneamente utilizando ciclos disponíveis nos recursos locais, no provedor na nuvem e na grade P2P. Começando em tr_i, o escalonador usa todos os ciclos disponíveis até que a execução da aplicação $\mathcal{A}_i$ seja finalizada, usando tanto mais ciclos locais quanto possível (já que o custo destes ciclos é incorrido se utilizados ou não); também quanto mais possível dos ciclos disponíveis na grade (cujos custos são desprezíveis); e, finalmente, ciclos disponíveis do provedor na nuvem. Seguindo este algoritmo de escalonamento, sob a suposição de que dois contratos em $\mathcal{K}$ especificam os mesmos tempos de consumo tb_j^i, o registro de utilização $\mathcal{S}$ associado com a execução de $\mathcal{W}$ com o planejamento $\mathcal{K}$ é tal que para todo $uc_j^i \in \mathcal{S}$,

$$uc_j^i = max(0, min(\mathcal{K}_j^i, d_i - \sum_{\{uc_{j'}^i \in \mathcal{S} | tb_{j'}^i < tb_j^i\}} uc_{j'}^i - \int_{tr_i}^{tb_j^i} (i(t)+g(t))dt)). \tag{5.8}$$

Neste cenário que propomos, o escalonamento da aplicação é precedido pela reserva dos ciclos do provedor de computação na nuvem. Isto torna o escalonamento relativamente simples, mas requer uma solução para o problema de executar o planejamento, ou seja, decidir

quando e quantos ciclos reservar. Planejamentos distintos levarão não apenas a diferentes custos, mas também a diferentes cenários de disponibilidade de ciclos para executar a aplicação e, dessa forma, a dois diferentes valores de tc_i. Por sua vez, valores diferentes para tc_i levam a diferentes utilidades. De forma resumida, utilizando o fato de que ciclos obtidos da grade P2P possuem custo desprezível, o lucro obtido por um planejamento $\mathcal{K}$ para carga de trabalho $\mathcal{W}$, que permite a finalização de uma aplicação $\mathcal{A}_i$ no tempo tc_i (onde $tr_i \leq tc_i \leq td_i) \forall\ i \in \{1, m\}$ é:

$$\wp(\mathcal{W}, \mathcal{K}, \mathcal{S}) \quad = \quad \mathcal{U}(\mathcal{W}, \mathcal{S}) - \gamma_{\mathcal{L}}(td_m - tp_1) - \gamma(\mathcal{W}, \mathcal{K}, \mathcal{S}), \tag{5.9}$$

onde

$$\mathcal{U}(\mathcal{W}, \mathcal{S}) = \sum_{i=1}^{m} u_i(tc_i - tr_i) \tag{5.10}$$

e $\mathcal{S}$ é o registro de utilização resultante da aplicação do algoritmo de escalonamento sob $\mathcal{K}$. O objetivo do algoritmo de planejamento é encontrar $\mathcal{K}$ que maximize $\wp(\mathcal{W}, \mathcal{K}, \mathcal{S})$.

5.3 Algoritmos de Planejamento

Nesse capítulo propomos alguns algoritmos de planejamento que podem ser usados para maximizar o lucro obtido quando da execução de uma aplicação na infraestrutura híbrida. Para tal, o algoritmo deve primeiramente estimar o número de ciclos que serão obtidos do provedor de computação na nuvem e então estabelecer o(s) contrato(s) que irá(ão) maximizar o lucro.

O número total de ciclos extra necessários do provedor na nuvem para completar a execução de uma aplicação $\mathcal{A}_i$ em algum instante de tempo tc_i é dado por:

$$ec_i(tc_i) = max(0, d_i - \int_{tr_i}^{tc_i} (i(t) + g(t))dt). \tag{5.11}$$

Visto o tempo relativamente curto no qual a aplicação está em execução, é razoável assumir que $i(t)$, para $tr_i \leq t \leq tc_i$, é conhecido no tempo tp_i, quando o planejamento começa. Infelizmente, devido à incerteza inerente da grade P2P, pode não ser possível no tempo tp_i predizer $g(t)$ de forma precisa para $tr_i \leq t \leq tc_i$.

Notamos que, se é possível no tempo tp_i estimar precisamente $ec_i(t)$ para qualquer t em $[tr_i, td_i]$, então para o modelo do sistema definido neste capítulo com a suposição adicional que β_j^i é o mesmo para todos os contratos $\mathcal{K}$ que podem ser estabelecidos, é simples encontrar um planejamento que maximize o lucro. Existe um planejamento consistindo apenas de um contrato j para o qual o lucro da aplicação $\mathcal{A}_i$ é máximo, $\mathcal{K}_i = \langle ec_i(tb_j^i), tp_i, tb_j^i, \beta \rangle$. Um simples otimizador (analítico) pode ser utilizado para encontrar o tempo tb_j^i que define este contrato. Por exemplo, para a aplicação AoN cuja utilidade é positiva e constante para qualquer tb_j^i, $tr_i \leq tb_j^i \leq td_i$, (e assumindo ainda que apenas contratos com $\beta_j^i = \beta$ podem ser estabelecidos), é fácil perceber que o contrato necessário é $\mathcal{K}_i = \langle ec_i(td_i), tp_i, td_i, \beta \rangle$, porque de todos os contratos que podem ser estabelecidos para maximizar a utilidade, este é o contrato menos urgente reservando o menor número de ciclos, e então o que tem o menor custo.

Na abordagem esboçada acima, subestimar o número de ciclos que serão recebidos da grade leva a uma superestimação de $ec_i(t)$ e a reserva de mais ciclos do que necessário. Por sua vez, isto pode levar a custos de reserva maiores do que se apenas os ciclos que realmente seriam utilizados tivessem sido reservados, e neste caso o lucro alcançado pode não ser o máximo. (É possível que este possa ser o máximo se os ciclos extra da grade possibilitar em finalização da aplicação mais cedo do que o esperado, resultando em uma utilidade maior.) Por outro lado, superestimar o número de ciclos que serão recebidos da grade leva a uma subestimação de $ec_i(t)$ e, como resultado, no tempo tb_j^i ainda existirá parte da aplicação $\mathcal{A}_i$ a ser computada. Haverá, portanto, custos inesperados ao comprar ciclos adicionais do provedor na nuvem necessários para completar a execução da aplicação (caso possível), e a aplicação finalizará depois do que esperado e então produzirá também uma utilidade menor do que a esperada. Assim, o lucro será menor do que o esperado pelo agente planejador no tempo em que foram estabelecidos os respectivos contratos, e é possível que uma escolha diferente de contratos poderá levar a lucros maiores.

Visto que, geralmente, não é possível predizer com precisão o número de ciclos que estarão disponíveis da grade computacional em um intervalo de tempo futuro, recorremos a heurísticas que tentam obter lucros mais próximos do máximo alcançável quanto possível. Estas heurísticas assumem que $\beta_j^i = \beta$ é o mesmo para todos os contratos em $\mathcal{K}$ que podem ser estabelecidos, e também que para qualquer ts_j^i, tb_j^i com $tp_i \leq ts_j^i < tb_j^i$, $tr_i \leq tb_j^i \leq td_i$

e qualquer inteiro positivo c, é possível estabelecer um contrato $\mathcal{K}_i = \langle c, ts_j^i, tb_j^i, \not\emptyset$.

De todas as heurísticas possíveis, nos concentramos naquelas que fazem no máximo dois contratos no intervalo de tempo $\Delta t_i = [tp_i, td_i]$. Em particular, assumimos que um contrato ($\mathcal{K}_i = \langle \mathcal{K}_1^i, ts_1^i, tb_1^i, \not\emptyset$) é sempre estabelecido o mais cedo possível, no tempo tp_i, enquanto o segundo contrato ($\mathcal{K}_i = \langle \langle \mathcal{K}_1^i, ts_1^i, tb_1^i, \not\emptyset, \langle \mathcal{K}_2^i, ts_2^i, tb_2^i, \not\emptyset \rangle \rangle$), quando necessário, é estabelecido no instante de tempo em que os ciclos reservados no primeiro contrato foram usados, ou seja, $ts_2^i = tb_1^i$. Descrevemos a razão para isto em seguida. A heurística executa um otimizador para encontrar o conjunto de contratos $\mathcal{K}_i$ que maximiza o lucro, dada uma estimativa gc_i' (que depende de tb_1^i) do número de ciclos que serão recebidos da grade no intervalo $[tr_i, tb_1^i]$. No tempo tb_1^i a heurística pode avaliar o quão precisa foi esta predição. Se o número de ciclos que foram recebidos da grade neste intervalo de tempo (denotamos por gc_i) é maior ou igual do que gc_i', então a execução da aplicação será finalizada no tempo (ou mesmo antes de) tb_1^i, e não será necessário providenciar mais recursos. Por outro lado, se $gc_i' < gc_i$, um contrato adicional pode ser necessário para completar a execução da aplicação. Neste ponto a heurística executa uma nova otimização, desta vez considerando apenas a carga de trabalho ainda não processada. Para evitar o estabelecimento de um terceiro contrato, a heurística toma uma abordagem conservativa e assume que nenhum ciclo será recebido da grade antes de finalizar a execução da aplicação.

Neste trabalho avaliamos diferentes "sabores" em cima deste quadro (comportamento) geral de heurísticas que diferem uma das outras em como elas estimam gc_i'. Discutiremos cada uma delas em seguida.

Omniscient Heuristic. Esta heurística produz o planejamento ótimo. Assumimos neste caso que a heurística tem acesso a um "oráculo" que é capaz de predizer gc_i com precisão, ou seja, supomos que $gc_i' = gc_i$. A heurística portanto sempre estabelece um único contrato com a quantidade precisa de ciclos extra necessários do provedor de computação na nuvem para finalizar a carga de trabalho. Logo, esta heurística sempre encontra o lucro máximo para a infraestrutura híbrida.

Averse Heuristic. Esta heurística tem um comportamento completamente averso ao risco de confiar na grade *best-effort*. Portanto, ela sempre assume que $gc_i' = 0$. Em outras palavras, a heurística sempre estabelece um primeiro contrato com todos os ciclos necessários para computar cada aplicação da carga de trabalho, ou seja, a diferença entre as demandas

d_i e os ciclos que serão providos pela infraestrutura local no respectivo intervalo de tempo. Obviamente, neste caso também não há a necessidade de estabelecer um segundo contrato. Embora esta heurística estime que nenhum ciclo será recebido da grade, o algoritmo de escalonamento utiliza os ciclos que de fato forem providos pela grade, potencialmente reduzindo o número de ciclos consumidos do provedor na nuvem.

Oblivious Heuristic. Esta heurística é indiferente à existência da grade, e portanto também assume que $gc'_i = 0$. Além disso, quando avaliamos esta heurística desprezamos $g(t)$ e assumimos que $g(t) = 0$ durante Δt_i. Em outras palavras, assumimos que o algoritmo de escalonamento não utiliza nenhum ciclo disponível na grade. Novamente, nesta caso apenas um único contrato é estabelecido. Sob nossas suposições em quais contratos podem ser estabelecidos e sobre o modelo de tarifação, esta heurística realiza a escolha ótima de contratos para uma infraestrutura híbrida que não tem acesso à uma grade P2P. Assim, quando comparamos as outras heurísticas com esta, podemos mensurar o valor que a grade P2P adiciona à infraestrutura híbrida de TI.

Predictive Heuristic. Esta heurística utiliza um oráculo para obter algum conhecimento sobre o comportamento da grade, assim como faz a *Omniscient*. Entretanto, o oráculo utilizado aqui não é perfeito. Modelamos esta imperfeição associando um erro diferente de zero, ξ_i, na predição realizada, ou seja, $gc'_i = gc_i \cdot (1 \pm \xi_i)$. Visto que esta heurística pode necessitar do estabelecimento de um segundo contrato para completar a execução da aplicação, o instante de tempo para utilização dos ciclos do primeiro contrato deve ser menor do que td_i, caso contrário pode não ser sempre possível estabelecer um segundo contrato (lembrando que $ts^i_j < tb^i_j$).

Em síntese, as heurísticas *Omniscient* e *Oblivious* produzem "benchmarks" para o lucro que podem ser obtidos a partir da execução da aplicação na infraestrutura híbrida. A heurística *Averse* revela o valor que a grade adiciona para a infraestrutura. Finalmente, a heurística *Predictive* permite avaliarmos o impacto que a qualidade na estimativa da quantidade de recursos recebidos da grade acarreta no lucro que pode ser obtido. O Algoritmo 1 pode ser visto como o "pseudo-código" do quadro de planejamento descrito para cada aplicação $\mathcal{A}_i$.

Algorithm 1: Algoritmo do Agente Planejador

begin

no tempo $time = tp_i$ **faça**

begin

estimar gc'_i de acordo com $Heuristic$

encontrar tb_1^i tal que Equação 5.9 seja maximizada, com

$ts_1^i = tp_i$ e $\mathcal{K}_1^i = d_i - gc'_i - \int_{tr_i}^{tb_1^i} i(t)dt$

estabelecer contrato $\mathcal{K}_i = \langle \mathcal{K}_1^i, ts_1^i, tb_1^i, \not\$ \rangle$

end

no tempo $time = tb_1^i$ **faça**

begin

seja d'_i o número de ciclos que ainda precisam ser processados

se $(d'_i > 0)$

begin

encontrar tb_2^i tal que Equação 5.9 seja maximizada, com

$ts_2^i = tb_1^i$ e $\mathcal{K}_2^i = d'_i - \int_{tb_1^{i+}}^{tb_2^i} i(t)dt$

estabelecer novo contrato $\mathcal{K}_i = \langle \langle \mathcal{K}_1^i, ts_1^i, tb_1^i, \not\$ \rangle, \langle \mathcal{K}\ {}_2^i, ts_2^i, tb_2^i, \not\$ \rangle \rangle$

end

end

end

5.4 Avaliação das Heurísticas

Nesta seção apresentamos uma avaliação de desempenho das heurísticas propostas, em relação ao lucro obtido pelas mesmas. Primeiramente, definimos uma métrica e descrevemos os cenários de avaliação. Em seguida, analisamos os resultados obtidos a partir de um modelo analítico do sistema definido na Seção 5.2. Por fim, com o intuito de avaliar o impacto das simplificações do modelo analítico, apresentamos resultados obtidos a partir de um modelo simulado.

5.4.1 Métrica de Avaliação e Descrição dos Cenários Avaliados

Para mensurar a eficiência de uma determinada heurística, comparamos o lucro obtido pela mesma com o produzido pela *Omniscient* e pela *Oblivious*. Definimos a *eficiência* alcançada por uma heurística H quando executa uma carga de trabalho $\mathcal{W}$ na infraestrutura híbrida como segue:

$$\mathcal{E}_H = 1 - \frac{\wp_{Omniscient}(\mathcal{W}, \mathcal{K}, \mathcal{S}) - \wp_H(\mathcal{W}, \mathcal{K}, \mathcal{S})}{\wp_{Omniscient}(\mathcal{W}, \mathcal{K}, \mathcal{S}) - \wp_{Oblivious}(\mathcal{W}, \mathcal{K}, \mathcal{S})}, \tag{5.12}$$

onde $\wp_{Omniscient}(\mathcal{W}, \mathcal{K}, \mathcal{S})$, $\wp_{Oblivious}(\mathcal{W}, \mathcal{K}, \mathcal{S})$ e $\wp_H(\mathcal{W}, \mathcal{K}, \mathcal{S})$ são, respectivamente, os lucros obtidos pelas heurísticas *Omniscient*, *Oblivious* e H, quando escalonam a carga de trabalho $\mathcal{W}$ de acordo com a utilização $\mathcal{S}$ sob a mesma infraestrutura híbrida e estabelecendo o conjunto de contratos $\mathcal{K}$.

É importante notar que a eficiência não é definida se o lucro da *Omniscient* é igual ao da *Oblivious*. Além disso, o limite superior para a eficiência é 1, mas não existe limite inferior visto que a eficiência pode ser negativa. Isto acontece quando a escolha dos contratos realizados pela heurística é tão ruim que o lucro resultante é ainda menor do que o obtido pela heurística *Oblivious*.

Para avaliar as heurísticas utilizando o modelo apresentado na Seção 5.2 precisamos definir cenários atribuindo valores para as constantes e instanciando as funções que compõem o modelo. Primeiramente, discutiremos as constantes e funções relacionadas à aplicação, e depois aquelas relacionadas às infraestrutura.

Com relação às aplicações, precisamos definir as demandas computacionais (d_i), as funções de utilidade e o tempo de vida (tr_i e td_i). Baseando-se na experiência de aplicações *e-Science* utilizadas na Comunidade OurGrid, definimos uma carga de trabalho para uma aplicação que demanda um *cluster* com 20 máquinas por aproximadamente 12 horas. Assim, temos $d_i = 864.000$ ciclos e $td_i - tr_i = 43.200$ segundos (12 horas), para qualquer $\mathcal{A}_i \in \mathcal{W}$. Assumimos $tp_1 = 0$, $tp_i = td_{i-1}$ e que Δt_i é um intervalo de 24 horas. Portanto, $tr_1 = 43.200$, $tr_i = tr_{i-1} + 86.400$, $td_1 = 86.400$ e $td_i = td_{i-1} + 86.400$ segundos. Para as funções de utilidade das aplicações AoN e $LDuD$, o valor b é a utilidade *máxima* que pode ser alcançada com a execução da aplicação. Em outras palavras, se a execução da aplicação

$\mathcal{A}_i$ finalizar exatamente em tr_i, o rendimento obtido é b unidades de utilidade. Consideramos que a utilidade obtida pela execução completa da aplicação é diretamente proporcional à sua demanda de processamento. Portanto, temos $b = \mu \cdot d_i$ (com $\mu > 1$), onde μ é um fator de lucratividade máxima para a aplicação. Por exemplo, $\mu = 2$ significa que 2 unidades de utilidade serão obtidas como resultado da execução da aplicação para cada unidade de carga de trabalho que a aplicação possui. Para a aplicação $GDuD$ temos $a = -1.00017$ e $b = 1.00054 \cdot \mu \cdot d_i$, o que provê uma utilidade que decresce lentamente até cerca de 8 horas após tr_i, e tem uma forte queda depois disso.

Consideramos que o custo da infraestrutura local durante o intervalo de tempo é igual ao número de ciclos que estão disponíveis a partir das máquinas locais neste intervalo, ou seja, temos $pr_2 = 1$. Definimos pr_3, o custo de reservar um ciclo do provedor na nuvem no tempo tp_i e consumí-lo até o tempo td_i, como função de pr_2. Nos cenários que avaliamos $pr_3 = 0.5 \cdot pr_2$, assumimos que o provedor na nuvem é capaz de operar sua infraestrutura duas vezes mais eficiente do que a infraestrutura local é operada.

Como indicado anteriormente, consideramos que o provedor na nuvem estabelece um valor βe permite apenas contratos estabelecidos para os quais $\beta \frac{i}{j} = \beta$ Visando avaliar diferentes perfis de provedores, consideramos três valores distintos para $\beta\, 1/2, 2/3$, e $3/4$.

A função $\varphi_j^i(\cdot)$, que reflete como as taxas de reserva variam com diferentes tempos de estabelecimento de contratos e tempos de consumo, foi definida como segue. Temos $\varphi_j^i(t_1, t_2)$, $\forall i \in \{1, m\}$ e $j \in \{1, n_i\}$ como uma função hiperbólica decrescente de $(t_2 - t_1)$ dada por:

$$\varphi_j^i(t_1, t_2) = \frac{18,000}{t_2 - t_1 + 5,600} + 0.8, \tag{5.13}$$

o que acarreta em um valor para os contratos de aproximadamente 4 para os mais urgentes $(\varphi_j^i(t, t^+))$ e aproximadamente $2,5$ para os contratos estabelecidos com pelo menos 12 horas de antecedência. Com isso investigamos o comportamento das heurísticas em cenários onde o custo dos contratos muda substancialmente.

Especificamos três perfis para a grade P2P a partir de sua qualidade de serviço Φ: uma grade com qualidade ruim (*bad quality grid*) com $\Phi = 0.1$, uma grade mediana (*medium quality grid*) com $\Phi = 0.5$, e uma grade boa (*good quality grid*) com $\Phi = 0.9$. Por exemplo, para a grade boa, o número de ciclos que estão disponíveis durante $[tr_i, td_i]$ é 90% dos ciclos doados para a grade durante $[tp_i, tr_i^-]$ ($\int_{tr_i}^{td_i} g(t) \cdot dt = 0.9 \cdot \int_{tp_i}^{tr_i^-} i(t) \cdot dt$). Entretanto,

é razoável supor que o número de ciclos disponíveis na grade durante um subintervalo de $[tr_i, td_i]$ é menor para um subintervalo mais cedo do que para um mais tarde (ainda que de mesmo tamanho), porque quando o intervalo mais tarde começa um número menor de favores pendentes (recursos doados) necessita ser retornado da grade para o *peer* representando a infraestrutura local dedicada. Portanto consideramos $g(t)$ como a seguinte função decrescente de t, dentro de cada período Δt_i:

$$g(t) = 2 \cdot \Phi \cdot (td_i - tr_i)^{-2} \cdot (td_i - t) \cdot \int_{tp_i}^{tr_i^-} i(t')dt'. \tag{5.14}$$

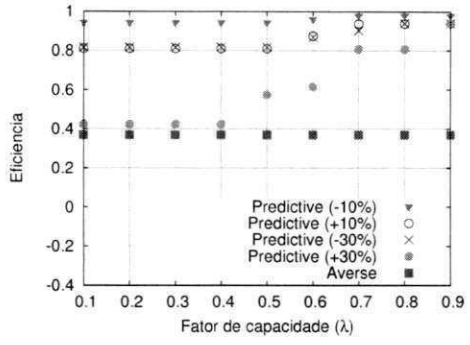
5.4.2 Resultados Analíticos

Assumimos, por simplicidade, que a capacidade da infraestrutura local não muda durante todo o intervalo de tempo, e introduzimos um *fator de capacidade* λ ($0 < \lambda < 1$) que expressa o quanto da demanda computacional das aplicações (d_i) pode ser processada localmente. Por exemplo, $\lambda = 0.5$ significa que a capacidade local pode computar 50% da carga de trabalho das aplicações. Quando $\lambda = 0$ não existem recursos locais disponíveis, e quando $\lambda = 1$ toda a carga de trabalho pode ser processada localmente. Nenhum destes dois cenários são interessantes para o nosso estudo. As figuras apresentadas nesta seção mostram gráficos de eficiência versus o fator de capacidade λ, para $\lambda \in \{0.1, 0.2, \ldots, 0.9\}$. Descrevemos primeiro os resultados gerais para todos os cenários avaliados, e depois discutimos o impacto de determinados parâmetros na eficiência obtida pelas heurísticas.

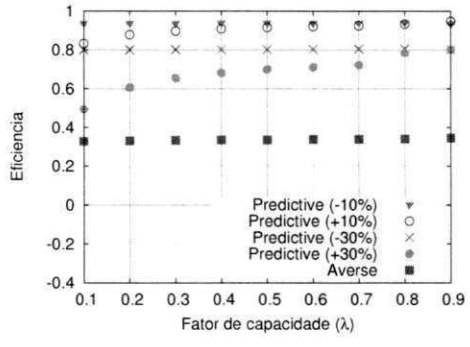
No geral, o tc_i para cada aplicação $\mathcal{A}_i$ é calculado como um compromisso entre a perda de utilidade com o seu aumento, e a redução no custo por evitar contratar ciclos no provedor na nuvem que seriam obtidos da infraestrutura local e da grade P2P durante o intervalo de tempo $[tr_i, tc_i]$. Contudo, para a aplicação $LDuD$, em todos os cenários avaliados, todas as heurísticas tomaram a decisão de contratar todos os ciclos para serem usados exatamente em tr_i. Isto acontece porque, para a aplicação $LDuD$, com o aumento de tc_i, a perda em utilidade é mais importante do que a respectiva redução no custo. Assim, em todos os cenários o lucro produzido pelas heurísticas *Omniscient* e *Oblivious* foram os mesmos e as eficiências para as outras heurísticas não são definidas. Por conseguinte, daqui em diante discutiremos apenas os valores de eficiência para as aplicações AoN e $GDuD$.

A Figura 5.1 ilustra a eficiência média obtida pelas heurísticas em todos os cenários

avaliados, a Figura 5.1(a) para a aplicação AoN, e a Figura 5.1(b) para a aplicação $GDuD$. As figuras mostram as eficiências das heurísticas *Averse* e *Predictive* para os casos onde a estimativa utilizada para o número de ciclos disponíveis na grade P2P é 10% menor, 10% maior, 30% menor ou 30% maior do que o valor real [2].



(a) AoN



(b) $GDuD$

Figura 5.1: Eficiência média versus o fator de capacidade (λ).

Como esperado, para erros relativamente baixos na estimativa dos ciclos recebidos da grade, a heurística *Predictive* oferece melhores resultados do que a heurística *Averse*, visto que a *Predictive* utiliza conhecimento parcial do comportamento da grade. Quando a *Predictive* subestima o número de ciclos disponíveis da grade, os resultados são melhores do que quando superestimados. Isto acontece porque, quando a heurística superestima o número de ciclos disponíveis na grade, ela reserva menos ciclos no seu primeiro contrato do que são

[2]Carvalho et al. [Carvalho et al. 2010] apresentam um modelo de predição para a grade computacional P2P com um erro médio de $7,2\%$. Dessa forma, consideramos que os valores utilizados neste trabalho para o erro na predição representam uma boa variação para este parâmetro.

necessários para completar a aplicação, e deve estabelecer um segundo contrato *a posteriori*. Isto resulta em altas taxas de reserva do que se todos os ciclos necessários tivessem sido contratados no primeiro contrato.

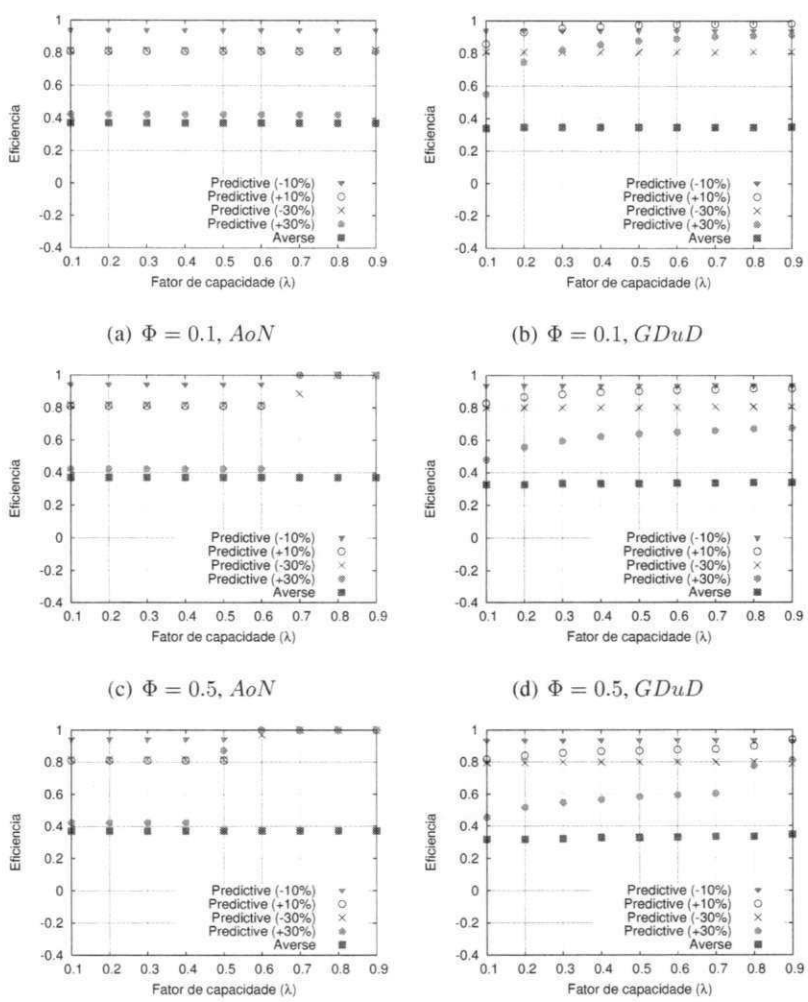
Para a aplicação AoN (vide Figura 5.1(a)), a heurística *Predictive* obtém uma eficiência maior para valores grandes de λ, porque quanto maior a capacidade local, maior será a quantidade de ciclos ociosos doados para a grade, e então mais ciclos serão provavelmente devolvidos durante o intervalo $[tr_i, tc_i]$. Para a aplicação $GDuD$ (vide Figure 5.1(b)) λ tem um impacto menor na eficiência da heurística *Predictive*, porque o tempo de conclusão para a execução do algoritmo é geralmente menor do que td_i e, assim, a heurística não toma vantagem de toda a capacidade disponível localmente e na grade antes do *deadline* td_i.

A heurística *Averse* ignora a grade e reserva todos os ciclos necessários (e que não serão providos pela infraestrutura local) junto ao provedor de computação na nuvem. Isto resulta em uma eficiência muito baixa para todos os cenários investigados. Esta eficiência não é afetada pelos valores de λ. Quando λ aumenta, o número de ciclos provavelmente disponíveis na grade também aumenta, reduzindo o custo provável de executar a aplicação. Porém, esta redução no custo é também obtida pelas outras heurísticas. Olhando detalhadamente para nossos resultados, descobrimos que quando λ aumenta, os lucros obtidos pelas heurísticas *Omniscient*, *Oblivious* e *Averse* aumentam na mesma proporção, tornando a eficiência da heurística *Averse* inalterável.

Avaliamos o impacto que a qualidade de serviço da grade Φ acarreta na eficiência, configurando $\Phi \in \{0.1; 0.5; 0.9\}$. A Figura 5.2 apresenta a eficiência média obtida para cada valor de Φ, para ambas as aplicações AoN e $GDuD$. Não surpreendentemente, a eficiência da heurística *Averse* não é afetada por Φ. Por outro lado, a heurística *Predictive* obtém melhores resultados com a melhora na qualidade de serviço da grade P2P, e alcança eficiência 1 para alguns cenários onde Φ e λ são grandes.

A Figura 5.3 apresenta a eficiência média obtida para diferentes valores de β Podemos notar a partir desta figura que, à medida que βaumenta, a eficiência das heurísticas diminui, e que βtem um grande impacto na heurística *Averse*, que reserva mais ciclos do provedor na nuvem do que as outras heurísticas.

Em síntese, nossos resultados mostram que é importante utilizar ao menos algum conhecimento sobre o comportamento da grade na hora de decidir quais contratos estabelecer

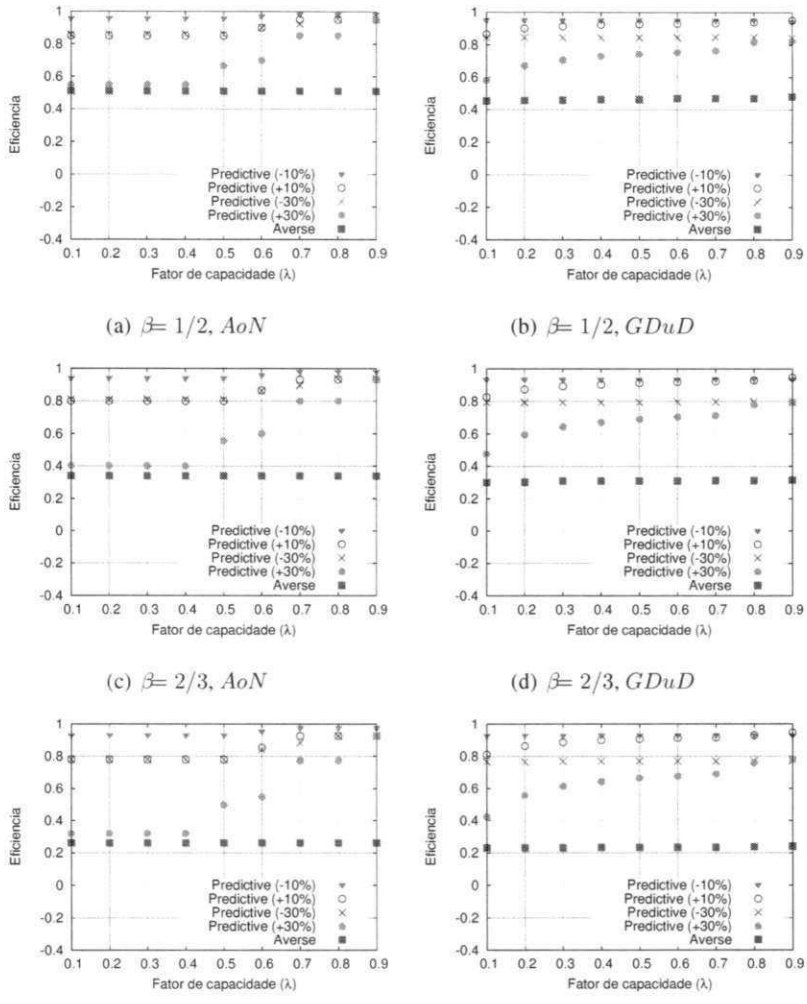


(a) $\Phi = 0.1$, AoN (b) $\Phi = 0.1$, $GDuD$

(c) $\Phi = 0.5$, AoN (d) $\Phi = 0.5$, $GDuD$

(e) $\Phi = 0.9$, AoN (f) $\Phi = 0.9$, $GDuD$

Figura 5.2: Eficiência versus o fator de capacidade (λ) para grades com diferentes fatores de qualidade de serviço (Φ).



(a) $\beta = 1/2, AoN$ (b) $\beta = 1/2, GDuD$

(c) $\beta = 2/3, AoN$ (d) $\beta = 2/3, GDuD$

(e) $\beta = 3/4, AoN$ (f) $\beta = 3/4, GDuD$

Figura 5.3: Eficiência versus o fator de capacidade (λ) para diferentes valores de β

com o provedor na nuvem. Assim, construir estimativas para o comportamento da grade é essencial para fechar contratos que levam a uma alta eficiência no uso da infraestrutura híbrida.

5.4.3 Resultados Simulados

Com o modelo analítico utilizado na subseção anterior apresentamos fortes indícios de que a infraestrutura híbrida de TI pode tomar vantagem do baixo custo de uma grade P2P para melhorar a eficiência alcançada ao computar aplicações científicas. Contudo, fazemos uma forte simplificação no modelo quando assumimos que a carga de trabalho é suficientemente paralela, de modo que quaisquer dois ciclos de quaisquer duas fontes podem ser utilizados alternadamente e em paralelo. Para avaliar o impacto desta suposição nos resultados apresentados anteriormente, implementamos um modelo de simulação que utiliza uma modelagem de carga de trabalho mais realística. Este modelo também considera a volatilidade dos recursos que são providos pela grade P2P e pelo provedor de computação na nuvem. Os primeiros se tornam indisponíveis a qualquer momento (caso o usuário da máquina doada à grade passe a utilizá-la) e os últimos estão disponíveis apenas durante o período de duração do contrato. O modelo de simulação implementa as heurísticas *Predictive*, *Averse*, e *Oblivious*. Implementar a heurística *Omniscient* acabou por se tornar uma tarefa muito complexa.

Modelo Simulado

Assim como no modelo analítico, na simulação os ciclos computacionais são providos por máquinas que estão disponíveis na infraestrutura local, recebidas como doações da grade P2P e possivelmente adquiridas do provedor de computação na nuvem. Por simplicidade, consideramos que todas as máquinas no sistema possuem o mesmo poder de processamento. Por outro lado, cada aplicação que compõem a carga de trabalho é caracterizada por uma tupla $\mathcal{A}_i = \langle q_i, \overline{w}_i, wmax_i, tr_i, u_i(\Delta t)\rangle$, onde q_i é o número de tarefas paralelas independentes que compõem a aplicação, $\overline{w}_i$ é a média do tempo de processamento destas tarefas, e $wmax_i$ é o tempo de processamento da tarefa mais longa a ser processada.

A granularidade do tempo na simulação é em milisegundos e um ciclo é o processamento

executado em um milisegundo por qualquer máquina no sistema. Logo, se uma tarefa leva 20 minutos para ser processada em uma máquina, levará este mesmo tempo em qualquer outra máquina no sistema. Entretanto, consideramos que nem a aplicação, nem a infraestrutura implementa soluções como *checkpointing* ou migração de tarefas. Uma tarefa escalonada para rodar em uma máquina na grade ou no provedor da nuvem, que se torna indisponível antes de completar sua execução, necessitará ser re-escalonada para começar sua execução novamente em outra máquina.

A infraestrutura híbrida de TI é composta de máquinas locais $i = \lfloor(\lambda \cdot q_i \cdot \overline{w}_i)/(t_d - t_r)\rfloor$, para qualquer intervalo $[tp_i, td_i]$. Estas máquinas são doadas para a grade durante o período $[tp_i, tr_i)$. Devido à natureza da grade P2P, não é possível saber o número exato de máquinas que estarão disponíveis na grade durante um dado intervalo de tempo. Contudo, usando a modelagem da QoS descrita anteriormente é possível calcular a quantidade total de ciclos que serão recebidos da grade P2P no intervalo $[tp_i, tr_i]$, que é dada por $\int_{tr_i}^{td_i} g(t)dt = \lfloor\Phi \cdot i \cdot (tr_i - tp_i)\rfloor$. A quantidade de máquinas disponíveis na nuvem dependerá dos contratos estabelecidos. Estes contratos são similares aos utilizados no modelo analítico, com a diferença que agora referem-se à quantidade de máquinas (ao invés de ciclos) que serão utilizadas durante um intervalo de tempo (ao invés de um instante de tempo). No modelo simulado, representamos o conjunto de contratos estabelecidos para aplicação $\mathcal{A}_i$ pela tupla $\mathcal{K}_i = \langle\langle m_1^i, ts_1^i, tb_1^i, te_1^i, \beta_1^i, ..., m_j^i, ts_j^i, tb_j^i, te_j^i, \beta_j^i\rangle\rangle$, onde tb_j^i representa o início do período de reserva, enquanto o te_j^i representa o final; m_j^i é o número reservado de máquinas, onde ts_j^i e β_j^i são definidos da mesma maneira que no modelo analítico. A antecedência de um contrato é dada por $tb_j^i - ts_j^i$.

A simulação procede de maneira muito similar ao algoritmo do planejador apresentado na Seção 5.3, com a diferença de que, no modelo simulado, consideramos: (i) máquinas na nuvem reservadas por um intervalo de tempo, e (ii) carga de trabalho dividida em tarefas, onde cada tarefa deverá ser executada completamente em uma única máquina. Estas considerações trazem os seguintes impactos: (i) a maneira como o otimizador é executado para encontrar os melhores contratos a serem estabelecidos (caso existam), e (ii) a maneira como é calculado o número de máquinas necessárias na nuvem em cada contrato.

Das várias alternativas possíveis, fizemos as seguintes escolhas. O primeiro contrato é sempre estabelecido em tp_i, começando em tr_i e com uma duração que é igual a $tc_i - tr_i$,

onde tc_i é o melhor tempo para concluir a aplicação, calculado pelo otimizador. Podemos notar que, como a execução de pelo menos uma tarefa dura $wmax_i$, então a duração do primeiro contrato nunca é menor do que este tempo ($tc_i \geqslant tr_i + wmax_i$). Seguimos um raciocínio similar para definir $te_1^i - wmax_i$ como o instante de tempo onde a heurística *Predictive* decide se um segundo contrato será necessário. Seja $\overline{n_c}$ o número médio de tarefas que podem ser executadas completamente em uma máquina durante o intervalo de tempo $[tr_i, tc_i]$, definido como $\overline{n_c} = \lfloor (tc_i - tr_i)/\overline{w}_i \rfloor$; e $\overline{g}$ a duração média uma máquina disponível na grade. Para a heurística *Predictive*, a quantidade de máquinas necessária no primeiro contrato que permite a execução ser finalizada em tc_i é calculada usando a seguinte expressão conservativa:

$$m_1^i = \left\lceil \frac{q_i - i \cdot \overline{n_c} - \left\lfloor \frac{\int_{tr_i}^{tc_i} g(t)dt}{\overline{g}} \right\rfloor \cdot \left\lfloor \frac{\overline{g}}{\overline{w}_i} \right\rfloor}{\overline{n_c}} \right\rceil. \tag{5.15}$$

Para as heurísticas *Averse* e *Oblivious*, o número de máquinas reservado no primeiro contrato é simplesmente dado por $\lfloor q_i/\overline{n_c} - i \rfloor$, uma vez que a grande não é levada em consideração. Usamos um raciocínio similar para calcular o segundo contrato para a heurística *Predictive*, caso necessário, considerando qualquer recurso ainda disponível do contrato anterior e apenas as tarefas que não foram iniciadas no instante de tempo em que o segundo contrato é planejado.

O simulador utiliza um otimizador com uma busca exaustiva para escolher o melhor tc_i. Para reduzir o tempo de execução de cada simulação, consideramos uma granularidade maior para o tempo de simulação de dois minutos. Fornecemos dois registros como entrada para o simulador, um que provê a disponibilidade das máquinas na grade; e outro que provê os erros na predição da grade para diferentes tamanhos de janelas de predição. Estes dois registros, em conjunto, são utilizados para calcular $g(t)$.

Descrição dos Cenários Avaliados

Executamos as simulações usando um subconjunto dos cenários descritos na Subseção 5.4.1. Definimos: $\beta = 2/3$, $d_i = q_i \cdot \overline{w}_i = 504,000$ e $\lambda \in 0.2, 0.5, 0.8$. Consideramos uma única aplicação que contém 300 tarefas, cujas demandas computacionais são distribuídas uniformemente no intervalo $[1080, 2280]$, o que representa tarefas com um tempo médio de

execução de 28 minutos; que é o tamanho típico de tarefas que executam em grades computacionais *desktop* [Carvalho et al. 2010]. Para cada cenário, executamos simulações suficiente para prover valores médios com um error menor do que 5%, com um nível de confiança de 95%. As simulações diferem umas das outros a partir dos registros de disponibilidade da grade e erros de predição.

Os registros de erros na predição para a quantidade de recursos recebidos da grade foram gerados a partir de um trabalho anterior [Carvalho et al. 2010]. A qualidade da predição depende da antecedência com a qual ela é realizada. Assim, coletamos erros na predição para diferentes intervalos, variando de estimativas para janelas de apenas 2 minutos a frente, até intervalos de 1 dia. O valor absoluto para o erro na predição teve uma média e mediana de, respectivamente, 6.9% e 3.6%; e 99% dos erros absolutos foram menores do que 25.7%.

Finalmente, geramos os registros para a disponibilidade das máquinas na grade a partir de dados reais coletados da comunidade OurGrid (http://status.ourgrid.org/). Para cada par de valores λ e Φ, um número suficiente de registros foram gerados tal que $\int_{tr_i}^{td_i} g(t)dt = \lfloor \Phi \cdot \lfloor (\lambda \cdot q_i \cdot \overline{w}_i)/(td_i - tr_i) \rfloor \cdot (tr_i - tp_i)$.

Avaliação dos Resultados do Modelo Analítico

A partir dos resultados obtidos com o modelo analítico, utilizar uma heurística que tenta predizer a QoS da grade P2P é a melhor escolha para escalonar aplicações *bag-of-tasks* na infraestrutura híbrida. A partir dos experimentos com o modelo de simulação buscamos avaliar se a heurística *Predictive* alcança realmente uma maior lucratividade do que as heurísticas *Averse* e *Oblivious*, como previsto pelo modelo analítico. Para isto, executamos ambos os modelos com as mesmas entradas e comparamos seus resultados.

Para cada cenário executamos 400 simulações dos dois modelos. Cada simulação de um cenário em particular tem uma respectiva resolução do modelo analítico, que utiliza a mesma configuração para os parâmetros do sistema e os mesmos registros para a disponibilidade da grade e erros na predição. É importante ressaltarmos que nestes experimentos os valores para a função $g(t)$ utilizados nos dois modelos são gerados a partir deste registros.

Para decidir que tipo de teste estatístico deveríamos utilizar para comparar os resultados, avaliamos se os valores dos lucros obtidos para cada experimento seguiam uma distribuição normal, para cada cenário avaliado. Para isto, executamos o teste *Shapiro-Wilk* [Shapiro e

Wilk 1965]. Descobrimos que os dados não são distribuídos segundo uma normal e, por conseguinte, não podemos utilizar o conhecido teste *t-test* [Jain 2008]. Decidimos utilizar dois testes não paramétricos utilizados para comparar duas alternativa: *Wilcoxon Rank Sum* e *Mann-Whitney* [Hollander e Wolfe 1999]. Para o teste *Wilcoxon Rank Sum* definimos as seguintes hipóteses comparando as heurísticas *Predictive* e *Averse* (respectivamente, *Oblivious*):

Hipótese Nula A heurística *Predictive* produz um lucro que não é maior do que o da heurística *Averse* (respectivamente, *Oblivious*);

Hipótese Alternativa A heurística *Predictive* produz um lucro que é maior do que o da heurística *Averse* (respectivamente, *Oblivious*).

Executamos o teste *Wilcoxon Rank Sum* considerando cada cenário onde λ e Φ variam. Para todos os cenários, exceto um que será explicado mais a frente, a hipótese alternativa foi verdadeira para ambos os modelos (analítico e simulado), com um nível de significância de 0.05. Com o intuito de reforçar os resultados, executamos testes *Mann-Whitney* nos quais a hipótese alternativa indica que os resultados de lucro obtidos pela heurística *Predictive* são maiores do que os resultados obtidos pelas heurísticas *Averse* e *Oblivious*. Novamente, em ambos os casos, nossa hipótese alternativa foi confirmada verdadeira para um nível de significância de 0.05, exceto para o mesmo cenário mencionado anteriormente.

A exceção ocorre para a heurística *Averse* no modelo simulado, quando a função de utilidade é AoN, λ é 0.2 e Φ é 0.1, que é o cenário no qual a contribuição da grade é menor. No modelo de simulação os resultados alcançados pela *Predictive* são iguais aos resultados obtidos pela *Averse*; enquanto que no modelo analítico, os resultados alcançados pela *Predictive* são melhores. No modelo simulado, para estes cenários, ambas as heurísticas tomam as mesmas decisões, por diferentes razões. Devido uma estimativa conservativa dos recursos disponíveis (Equação 5.15), a heurística *Predictive* estima que nenhum recurso estará disponível na grade e decide adquirir todos os recursos extra necessários do provedor de computação na nuvem. A heurística *Averse* sempre decide sem levar em consideração a grade P2P e, portanto, também adquiri todos os recursos extra a partir do provedor na nuvem. Como resultado, os resultados obtidos por ambas as heurísticas são os mesmos. Para este cenário específico, executamos novos teste estatísticos cuja hipótese nula confirma que os

resultados são os mesmos; esta hipótese nula é verdadeira com um nível de significância de 0.05. Quando verificamos os resultados analíticos do mesmo cenário, percebemos que qualquer ciclo proveniente da grade pode ser proveitosamente utilizado. Logo, tentar estimar a quantidade de recursos será recebida da grade é sempre melhor do que ignorá-la.

Os testes realizados apresentam um forte indício que, apesar das simplificações do modelo analítico, este é preciso o suficiente para mostrar, assim como no modelo simulado, que a heurística *Predictive* é sempre melhor do que a *Oblivious*. A heurística *Predictive* é também melhor do que a *Averse* na maioria dos cenários, e tão boa quanto quando a contribuição da grade para infraestrutura híbrida é mínima.

Resultados de Desempenho do Modelo Simulado

Apresentamos agora os resultados alcançados com as simulações. Uma vez que a heurística *Omniscient* não foi implementada, ao invés da relativa eficiência como métrica, avaliamos a *melhora relativa* em termos de lucro alcançada pela heurística *Predictive* em comparação com as outras duas implementadas (*Averse* e *Oblivious*). A Tabela 5.1 apresenta os resultados médios obtidos para diferentes valores de λ.

Tabela 5.1: A média da melhora relativa em relação ao lucro para $\lambda \in 0.2, 0.5, 0.8$.

Heurística relativa ao qual o desempenho é calculado	$\lambda = 0.2(\%)$	$\lambda = 0.5(\%)$	$\lambda = 0.8(\%)$
AoN			
Averse	0.4	1.6	2.4
Oblivious	0.7	2.1	2.7
GDuD			
Averse	0.1	0.6	1.0
Oblivious	0.2	1.0	2.1

Novamente, os melhores resultados são alcançados para os maiores valores de λ. Todavia, pelos resultados obtidos podemos perceber que a melhora no desempenho é modesta. Nas simulações, o desempenho da heurística *Predictive* não somente pela precisão nas predições realizadas para a disponibilidade de recursos, como também pela volatilidade das destas máquinas recebidas da grade. Como mostram os resultados do modelo analítico, me-

lhores modelos de predição beneficiariam o desempenho de heurísticas do tipo *Predictive*. Além disso, destacamos também que a implementação proposta da *Predictive* é apenas uma dentre várias outras possibilidades. A utilização de um escalonador mais sofisticado que diferenciasse recursos com atributos de QoS distintos, por exemplo, poderia prover melhores resultados.

Acreditamos que a pequena melhora nos resultados obtidos pela heurística *Predictive*, em comparação com a *Averse* e *Oblivious*, deve-se ao fato de que os benefícios de usar a grade P2P foram minimizados nos cenários de simulação utilizados. Podemos perceber este fato através de uma comparação entre as heurísticas *Averse* e *Oblivious*. Ambas as heurísticas utilizam a mesma abordagem para estabelecer os contratos e a diferença entre as duas é que a *Averse* utiliza a grade para executar a aplicação, enquanto a *Oblivious* a ignora completamente.

O benefício de usar a grade P2P é minimizado porque executamos as simulações utilizando cenários de pior caso, ou seja, cenários da grade em que existe uma alta contenção pelos recursos. Isto afeta não apenas a disponibilidade destes recursos, mas também nos erros retornados pelo modelo de predição da grade. Não consideramos cenários de baixa contenção onde, em uma janela de tempo particular, a quantidade de recursos recebida da grade pode ser inclusive muito maior do que a quantidade doada. Esta é uma situação possível de acontecer em ambientes reais. Para termos uma noção, executamos simulações utilizando um fator de qualidade da grade no qual a quantidade de recursos retornada é duas vezes a quantidade doada ($\Phi = 2$), e os resultados obtidos pela heurística *Predictive* foram de até 7% e 8% melhores em comparação aos da, respectivamente, *Averse* e *Oblivious*, considerando valores intermediários para λ. Além disso, conjecturamos que um ambiente real no qual mecanismos de tolerância à falhas, tais como replicação ou *checkpointing*, podem levar a resultados ainda melhores com um aumento na previsibilidade do sistema. Finalmente, é importante ressaltarmos que os resultados são referentes a uma melhoria no lucro obtido, que mesmo quando pequenas, estas melhorias são sempre significantes, uma vez que isto impacta diretamente o modelo de negócio do cliente.

5.5 Considerações a Respeito do Gerenciamento a Curto Prazo

Neste capítulo definimos uma infraestrutura híbrida de TI a qual consideramos que será uma realidade em um futuro próximo. O poder de processamento desta infraestrutura é formado por: recursos locais dedicados, poder computacional adquirido sob-demanda de um provedor de computação na nuvem e recursos ociosos negociados em uma grade P2P. Apresentamos os resultados para o gerenciamento desta infraestrutura híbrida dirigido a negócios que considera contratos de curta duração com provedores de computação na nuvem. Acreditamos que este trabalho pode assistir proprietários/gerentes de infraestruturas de TI, provendo heurísticas para decidir quando e quanto reservar recursos junto aos provedores na nuvem.

Descrevemos heurísticas de planejamento de contratos que podem ser utilizadas por um agente planejador autônomo, visando reduzir o custo de computar aplicações com restrições de tempo no ambiente proposto, ao mesmo tempo que garantias de execução até os respectivos *deadlines* são alcançadas. Também derivamos um modelo de tarifação para provedores de computação na nuvem que considera contratos de curta duração. Embora este trabalho tenha antecipado a utilização de um esquema de reserva que é utilizado pela maioria dos provedores na nuvem atualmente, praticamente todos eles empregam uma abordagem a longo prazo para a reserva dos recursos. Advogamos aqui que modelos mais flexíveis, assim como o apresentado neste capítulo, serão considerados por provedores quando a concorrência no mercado de computação na nuvem se tornar uma realidade. Por fim, avaliamos a utilidade total obtida pela execução de uma aplicação na infraestrutura híbrida baseados em funções de utilidade propostas para aplicações com restrições de tempo.

Pelos resultados obtidos, concluímos que o uso desta infraestrutura híbrida de TI pode tomar vantagem do baixo custo de uma grade P2P para melhorar a eficiência alcançada ao computar as aplicações, principalmente em cenários com baixa contenção por recursos, onde o benefício de usar a grade é maior. Entretanto, não é uma boa abordagem confiar completamente na grade, exceto em situações próximas de um provisionamento superestimado da infraestrutura local dedicada. Ademais, a vantagem da utilização de um sistema *best-effort* pode ser maximizada pelo uso parcial de informações sobre o comportamento da grade. Em particular, mostramos que obter uma estimativa precisa para o comportamento da grade é es-

sencial para realização de contratos que levam a uma alta eficiência no uso da infraestrutura híbrida.

Capítulo 6

Considerações Finais e Trabalhos Futuros

6.1 Considerações Finais

Neste trabalho definimos uma infraestrutura híbrida de TI para execução de aplicações *bag-of-tasks*, cujo poder de processamento é formado por: recursos locais dedicados, poder computacional adquirido sob demanda de um provedor de computação na nuvem e recursos ociosos negociados em uma grade *peer-to-peer* (P2P). Apresentamos os resultados para o gerenciamento desta infraestrutura híbrida dirigido a negócios que considera contratos de curta e longa duração com provedores de computação na nuvem. Acreditamos que esta infraestrutura híbrida será uma realidade em um futuro próximo e que este trabalho poderá assistir proprietários/gerentes de infraestruturas de TI, provendo estratégias e heurísticas para decidir como melhor planejar e operar tais ambientes.

O objetivo principal desse trabalho é averiguar a hipótese de que infraestruturas híbridas de TI podem tomar proveito do baixo custo da utilização de grades computacionais P2P para maximizar a lucratividade através da execução de aplicações com restrições de tempo. Como mencionamos anteriormente, buscamos neste trabalho investigar estratégias que podem ser utilizadas por um gerente de uma infraestrutura de TI na hora de *planejar*, *reservar*, *alocar* e *investir* (ou *empregar*) os recursos provenientes do mercado de computação na nuvem.

A partir dos resultados obtidos, concluímos que o uso desta infraestrutura híbrida de TI pode tomar vantagem do baixo custo de uma grade P2P para melhorar a eficiência alcançada

ao computar aplicações científicas do tipo *bag-of-tasks*. Entretanto, não é uma boa abordagem confiar completamente na grade, exceto em situações próximas de um provisionamento superestimado da infraestrutura local dedicada. Ademais, a vantagem da utilização de um sistema *best-effort* pode ser maximizada pelo uso parcial de informações sobre o comportamento da grade. Em particular, mostramos que obter uma estimativa para o comportamento da grade é essencial para realização de contratos que levam a uma alta eficiência no uso da infraestrutura híbrida, principalmente se estes contratos forem de curta duração.

Podemos então destacar as seguintes contribuições abaixo sumarizadas:

- Modelamos o ambiente no qual a infraestrutura híbrida está inserida, em termo de seus componentes: a carga de trabalho submetida à infraestrutura híbrida e as funções de utilidade de suas aplicações; os recursos dedicados oferecidos pela infraestrutura local; o mercado de computação na nuvem através de seus provedores e modelos de tarifação; e a grade computacional P2P e sua respectiva qualidade de serviço.

- Fornecemos fortes indícios que um planejamento de capacidade eficiente pode mitigar incertezas durante a operação da infraestrutura híbrida, buscando reservar recursos futuros a preços mais acessíveis. Neste sentido, demonstramos que o planejamento de capacidade a longo prazo deve ser robusto a estas incertezas e indicar um cenário de reserva subótimo junto aos provedores na nuvem.

- Investigamos como uma estratégia de alocação de recursos pode melhorar o lucro esperado a partir da fase de planejamento, através de um escalonamento eficiente dos recursos disponíveis em tempo de execução. Neste caso, podemos utilizar informações mais acuradas a respeito do sistema na hora de alocar efetivamente os recursos previamente reservados, buscando "corrigir" o efeito das incertezas na fase de planejamento.

- Validamos a ideia de que a infraestrutura híbrida pode tomar proveito da grade P2P no intuito de reduzir os custo incorridos na execução de aplicações com restrições de tempo. Para isto, propusemos heurísticas de planejamento de contratos com diferentes níveis de "confiança" na grade P2P *best-effort* no momento de estabelecer compromissos de curta duração com provedores de computação na nuvem.

É importante destacarmos que os resultados obtidos durante este trabalho partem de avaliações que consideram diversas premissas nos cenários avaliados. Essas premissas buscam simplificar alguns detalhes do sistema híbrido proposto. Essas simplificações podem ou não trazer implicações no efeito das duas abordagens de gerenciamento modeladas (longo e curto prazo), para o caso de uma avaliação empírica utilizando sistemas reais. Contudo, devido à complexidade da infraestrutura híbrida em relação aos vários parâmetros modelados, esses simplificações foram essenciais para que pudéssemos obter indícios das vantagens alcançadas com as estratégias de gerenciamento propostas. Apesar disso, conjecturamos que as simplificações adotadas não influencia a principal ideia das nossas avaliações, que é a vantagem trazida pelo uso da grade P2P no contexto do ambiente proposto.

6.2 Trabalhos Futuros

Como verificamos no Capítulo 4, temos indícios de que um planejamento a longo prazo é essencial em termos de melhorar a lucratividade esperada. Dessa forma, uma melhor caracterização da carga de trabalho submetida à infraestrutura híbrida é altamente desejável. Com esta caracterização, poderemos avaliar diferentes cenários de demanda, a partir da geração sintética da carga de trabalho. Além disso, precisamos refinar os parâmetros instanciados no modelo do sistema, principalmente com relação aos parâmetros da função de utilidade das aplicações e das relações entre os preços cobrados por diferentes provedores na nuvem.

Ainda dentro da análise do gerenciamento a longo prazo, precisamos elaborar melhor os cenários de avaliação. Dessa forma, precisamos definir novas métricas e novas formas de comparação entre diferentes cenários, visando destacar melhor a importância de cada fase, bem como cada componente, no planejamento da infraestrutura híbrida. A partir destes novos cenários de avaliação, pretendemos demonstrar que o planejamento a longo prazo é importante. Em outras palavras, precisamos demonstrar, através de cenários práticos de avaliação, que a falta de um planejamento prévio pode levar a uma má utilização dos recursos disponíveis em tempo de execução da carga de trabalho. Pretendemos também ratificar o valor da grade computacional P2P através de evidências numéricas do ganho obtido com os recursos retornados da grade.

Dentro do contexto de um gerenciamento a curto prazo, é essencial a criação de meca-

nismos mais acurados capazes de estimar dinamicamente a quantidade de ciclos que serão recebidos da grade durante o estabelecimento de contratos, possibilitando a construção de heurísticas robustas a algum nível de erro nesta estimativa. Além disso, pretendemos investigar outras formas de tarifação a curto prazo que podem ser empregadas no mercado de computação na nuvem.

Por fim, pretendemos investigar estratégias de "investimento em um mercado futuro" a partir dos recursos previamente reservados. Em outras palavras, pretendemos elaborar heurísticas operacionais para infraestrutura híbrida considerando uma análise de retorno futuro dos recursos provenientes da grade P2P. Buscaremos responder perguntas tais como: será que compensa "doar" recursos alocados nos provedores na nuvem (e que não estão sendo utilizados) para a grade P2P, a fim de obter um maior saldo futuro? Esta análise também pretende avaliar qual é o melhor momento para utilização do saldo mantido na grade P2P, a partir de uma estimativa do valor futuro do recurso retornado. Será que não é mais vantajoso deixar de executar uma determinada aplicação agora para priorizar outra com maior utilidade num futuro próximo? Para tanto, uma análise de risco também deve ser elaborada com o intuito de mitigar as incertezas impostas pelo ambiente, tais como: demanda computacional, disponibilidade de recurso, estimativas no retorno e no valor futuro da grade, etc.

Referências Bibliográficas

[Amazon AWS 2010]Amazon AWS 2010. *Amazon EC2 pricing models [Online]. Disponível em: http://aws.amazon.com/ec2/#pricing. Acessado em: Novembro de 2010.*

[Amazon AWS Case Studies]AMAZON AWS Case Studies. Disponível em: <http://aws.amazon.com/solutions/case-studies/>.

[Andrade et al. 2007]ANDRADE, N. et al. Automatic grid assembly by promoting collaboration in peer-to-peer grids. *J. Parallel Distrib. Comput.*, Academic Press, Inc., Orlando, FL, USA, v. 67, n. 8, p. 957–966, 2007. ISSN 0743-7315.

[Araujo et al. 2005]ARAUJO, E. C. et al. The SegHidro Experience: Using the Grid to Empower a Hydro-Meteorological Scientific Network. In: *Proc. of the 1st IEEE International Conference on e-Science and Grid Computing (e-Science 2005)*. Melbourne, Australia: [s.n.], 2005.

[Assuncao, Costanzo e Buyya 2009]ASSUNCAO, M. D. de; COSTANZO, A. di; BUYYA, R. Evaluating the cost-benefit of using cloud computing to extend the capacity of clusters. In: *HPDC '09: Proceedings of the 18th ACM international symposium on High performance distributed computing*. New York, NY, USA: ACM, 2009. p. 141–150. ISBN 978-1-60558-587-1.

[Assunção, Costanzo e Buyya 2009]ASSUNÇÃO, M. de; COSTANZO, A. di; BUYYA, R. Evaluating the cost-benefit of using cloud computing to extend the capacity of clusters. In: ACM NEW YORK, NY, USA. *Proceedings of the 18th ACM International Symposium on High Performance Distributed Computing*. [S.l.], 2009. p. 141–150.

[Brasileiro et al. 2007]BRASILEIRO, F. et al. Bridging the High Performance Computing Gap: the OurGrid Experience. In: *Proc. 1st Latin-American Grid Workshop (co-located with CCGrid'2007)*. Rio de Janeiro, Brazil: [s.n.], 2007.

[Buyya et al. 2002]BUYYA, R. et al. Economic models for resource management and scheduling in Grid computing. *Concurrency and Computation: Practice and Experience*, v. 14, n. 13-15, p. 1507–1542, 2002.

[Camorlinga e Schofield 2006]CAMORLINGA, S.; SCHOFIELD, B. Modeling of workflow-engaged networks on radiology transfers across a metro network. *IEEE Transactions on Information Technology in Biomedicine*, v. 10, n. 2, p. 275–281, 2006.

[Carr 2008]CARR, N. G. *The big switch : rewiring the world, from Edison to Google*. 1st ed.. ed. [S.l.]: W. W. Norton & Co., New York:, 2008. vii, 278 p. ISBN 9780393062281 0393062287.

[Carvalho et al. 2010]CARVALHO, M. et al. Predicting the quality of service of a peer-to-peer desktop grid. In: *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*. Washington, DC, USA: IEEE Computer Society, 2010. (CCGRID '10), p. 649–654. ISBN 978-0-7695-4039-9. Disponível em: <http://dx.doi.org/10.1109/CCGRID.2010.50>.

[Carvalho 2011]CARVALHO, M. W. A. de. *Predição da Qualidade de Serviço em Grades Computacionais P2P*. Dissertação (Mestrado) — Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande, Campina Grande - PB, Brasil, 2011.

[CERN-LHCC-02-26]CERN-LHCC-02-26. *Control and monitor the high energy physics experiments*. Disponível em: <http://cmsdoc.cern.ch/cms/TRIDAS/Temp/CMS_DAQ_TDR.pdf>.

[Chun e Culler 2002]CHUN, B. N.; CULLER, D. E. User-centric performance analysis of market-based cluster batch schedulers. In: *Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'02)*. Washington, DC, USA: IEEE Computer Society, 2002. p. 30. ISBN 0-7695-1582-7.

[Cirne et al. 2006]CIRNE, W. et al. Labs of the World, Unite!!! *Journal of Grid Computing*, v. 4, n. 3, p. 225–246, 2006. Disponível em: <http://dx.doi.org/10.1007/s10723-006-9040-x>.

[Coêlho, Brasileiro e Maciel Jr. 2009]COêLHO, A.; BRASILEIRO, F.; MACIEL JR., P. D. Using heuristics to improve service portfolio selection in p2p grids. In: *Proceedings of the 11th IFIP/IEEE international conference on Symposium on Integrated Network Management*. Piscataway, NJ, USA: IEEE Press, 2009. (IM'09), p. 438–444. ISBN 978-1-4244-3486-2. Disponível em: <http://dl.acm.org/citation.cfm?id=1688933.1689000>.

[Coêlho et al. 2008]COêLHO, A. et al. On the impact of choice in multi-service p2p grids. In: BARTOLINI, C.; SAHAI, A.; SAUVé, J. P. (Ed.). *Proceedings of BDIM 2008, 3rd IEEE/IFIP International Workshop on Business-Driven IT Management, April 7, 2008, Salvador, Brazil*. [S.l.]: IEEE, 2008. p. 98–101. ISBN 978-1-4244-2191-6.

[Duarte et al. 2006]DUARTE, A. N. et al. GridUnit: Software Testing on the Grid. In: *Proc. 28th ACM/IEEE International Conference on Software Engineering (ICSE'06)*. Shanghai, China: [s.n.], 2006.

[Farooq, Majumdar e Parsons 2006]FAROOQ, U.; MAJUMDAR, S.; PARSONS, E. W. A framework to achieve guaranteed QoS for applications and high system performance in multi-institutional grid computing. In: *ICPP '06: Proceedings of the 2006 International Conference on Parallel Processing*. Washington, DC, USA: IEEE Computer Society, 2006. p. 373–380. ISBN 0-7695-2636-5.

[Farooq, Majumdar e Parsons 2007]FAROOQ, U.; MAJUMDAR, S.; PARSONS, E. W. Engineering grid applications and middleware for high performance. In: *WOSP '07: Proceedings of the 6th international workshop on Software and performance*. New York, NY, USA: ACM, 2007. p. 141–152. ISBN 1-59593-297-6.

[Foster e Roy 2000]FOSTER, I.; ROY, A. A quality of service architecture that combines resource reservation and application adaptation. In: *8th Int. Workshop on Quality of Service*. [S.l.: s.n.], 2000. p. 181–188.

[Funk, Goossens e Baruah 2001]FUNK, S.; GOOSSENS, J.; BARUAH, S. On-line scheduling on uniform multiprocessors. In: *RTSS '01: Proceedings of the 22nd IEEE Real-Time Systems Symposium (RTSS'01)*. Washington, DC, USA: IEEE Computer Society, 2001. p. 183. ISBN 0-7695-1420-0.

[Galante e Bona 2012]GALANTE, G.; BONA, L. C. E. d. A survey on cloud computing elasticity. In: *Proceedings of the 2012 IEEE/ACM Fifth International Conference on Utility and Cloud Computing*. Washington, DC, USA: IEEE Computer Society, 2012. (UCC '12), p. 263–270. ISBN 978-0-7695-4862-3. Disponível em: <http://dx.doi.org/10.1109/UCC.2012.30>.

[Gartner Research 2007]Gartner Research. *Critical Actions for the Transition of IT Organizations: 2007 to 2008 (ID Number: G00148987)*. June 2007.

[Hollander e Wolfe 1999]HOLLANDER, M.; WOLFE, D. *Nonparametric Statistical Methods*. Wiley, 1999. (A Wiley-Interscience publication). ISBN 9780471190455. Disponível em: <http://books.google.com.br/books?id=RJAQAQAAIAAJ>.

[Hwang, Dongarra e Fox 2011]HWANG, K.; DONGARRA, J.; FOX, G. C. *Distributed and Cloud Computing: From Parallel Processing to the Internet of Things*. 1st. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011. ISBN 0123858801, 9780123858801.

[InfoWorld 2010]InfoWorld. *The case for the hybrid cloud. [Online]. Available: http://www.infoworld.com/d/cloud-computing/the-case-the-hybrid-cloud-196.* March 2010.

[Iosup et al. 2008]IOSUP, A. et al. The Grid Workloads Archive. *Future Gener. Comput. Syst.*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 24, n. 7, p. 672–686, 2008. ISSN 0167-739X.

[Irwin, Grit e Chase 2004]IRWIN, D. E.; GRIT, L. E.; CHASE, J. S. Balancing risk and reward in a market-based task service. In: *Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing (HPDC'04)*. Washington, DC, USA: IEEE Computer Society, 2004. p. 160–169. ISBN 0-7803-2175-4.

[Jain 2008]JAIN, R. *The Art Of Computer Systems Performance Analysis:*. Wiley India Pvt. Limited, 2008. ISBN 9788126519057. Disponível em: <http://books.google.com.br/books?id=eOR0kJjgMqkC>.

[Kim et al. 2009]KIM, H. et al. Investigating the Use of Autonomic Cloudbursts for High-Throughput Medical Image Registration. *10th IEEE / ACM International Conference on Grid Computing (Grid 2009)*, 2009.

[Kuntschke et al. 2006]KUNTSCHKE, R. et al. Grid-Based Data Stream Processing in e-Science. In: *E-SCIENCE '06: Proceedings of the Second IEEE International Conference on e-Science and Grid Computing*. Washington, DC, USA: IEEE Computer Society, 2006. p. 30. ISBN 0-7695-2734-5.

[Lee e Snavely 2007]LEE, C. B.; SNAVELY, A. E. Precise and realistic utility functions for user-centric performance analysis of schedulers. In: *Proceedings of the 16th international symposium on High performance distributed computing*. New York, NY, USA: ACM, 2007. (HPDC '07), p. 107–116. ISBN 978-1-59593-673-8. Disponível em: <http://doi.acm.org/10.1145/1272366.1272381>.

[Lopes, Brasileiro e Maciel Jr. 2010]LOPES, R. V.; BRASILEIRO, F. V.; MACIEL JR., P. D. Business-driven capacity planning of a cloud-based it infrastructure for the execution of web applications. In: *IPDPS Workshops*. IEEE, 2010. p. 1–8. Disponível em: <http://dblp.uni-trier.de/db/conf/ipps/ipdps2010w.html#LopesBM10>.

[Maciel Jr. et al. 2012]MACIEL JR., P. D. et al. Business-driven short-term management of a hybrid it infrastructure. *Journal of Parallel and Distributed Computing*, Academic Press, Inc., Orlando, FL, USA, v. 72, n. 2, p. 106–119, fev. 2012. ISSN 0743-7315. Disponível em: <http://dx.doi.org/10.1016/j.jpdc.2011.11.001>.

[Maciel Jr. et al. 2011]MACIEL JR., P. D. et al. Evaluating the impact of planning long-term contracts on the management of a hybrid it infrastructure. In: AGOULMINE, N. et al. (Ed.). *Integrated Network Management*. IEEE, 2011. p. 89–96. ISBN 978-1-4244-9221-3. Disponível em: <http://dblp.uni-trier.de/db/conf/im/im2011.html#MacielBLCM11>.

[Maciel Jr. et al. 2008]MACIEL JR., P. D. et al. On the planning of a hybrid it infrastructure. In: *IEEE/IFIP Network Operations and Management Symposium: Pervasive Management for Ubioquitous Networks and Services, NOMS 2008, 7-11 April 2008, Salvador, Bahia, Brazil*. IEEE, 2008. p. 496–503. Disponível em: <http://dblp.uni-trier.de/db/conf/noms/noms2008.html#MacielFMBC08>.

[Marshall, Keahey e Freeman 2010]MARSHALL, P.; KEAHEY, K.; FREEMAN, T. Elastic Site: Using Clouds to Elastically Extend Site Resources. *Cluster Computing and the Grid, IEEE International Symposium on*, IEEE Computer Society, Los Alamitos, CA, USA, v. 0, p. 43–52, 2010.

[Marshall, Keahey e Freeman 2010]MARSHALL, P.; KEAHEY, K.; FREEMAN, T. Elastic Site Using Clouds to Elastically Extend Site Resources. *10th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2010)*, 2010.

[McGough et al. 2007]MCGOUGH, A. S. et al. GRIDCC: real-time workflow system. In: *WORKS '07: Proceedings of the 2nd workshop on Workflows in support of large-scale science*. New York, NY, USA: ACM, 2007. p. 3–12. ISBN 978-1-59593-715-5.

[Miettinen 1999]MIETTINEN, K. *Nonlinear Multiobjective Optimization*. [S.l.]: Springer, 1999. ISBN 978-0-7923-8278-2.

[Mills e Dabrowski 2008]MILLS, K. L.; DABROWSKI, C. Can Economics-based Resource Allocation Prove Effective in a Computation Marketplace? *J. Grid Comput.*, v. 6, n. 3, p. 291–311, 2008.

[Osthoff et al. 2003]OSTHOFF, C. et al. A Grid Computing Testbed for EM Algorithm Financial Market Applications. In: *Third IFIP Conference on E-Commerce, E-Business and E-Goverment*. Guarujá, SP: [s.n.], 2003. p. 583–590.

[Pandey et al. 2010]PANDEY, S. et al. A Particle Swarm Optimization-based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments. *International Conference on Advanced Information Networking and Applications (AINA 2010)*, 2010.

[Plale et al. 2005]PLALE, B. et al. Towards Dynamically Adaptive Weather Analysis and Forecasting in LEAD. In: *Computational Science - ICCS 2005, 5th International Con-*

ference, Atlanta, GA, USA, May 22-25, 2005, Proceedings, Part II. [S.l.: s.n.], 2005. p. 624–631.

[Plaza et al. 2006]PLAZA, A. et al. Commodity cluster-based parallel processing of hyperspectral imagery. *J. Parallel Distrib. Comput.*, Academic Press, Inc., Orlando, FL, USA, v. 66, n. 3, p. 345–358, 2006. ISSN 0743-7315.

[Popovici e Wilkes 2005]POPOVICI, F. I.; WILKES, J. Profitable services in an uncertain world. In: *Proceedings of the 2005 ACM/IEEE Conference on Supercomputing (SC'05)*. Washington, DC, USA: IEEE Computer Society, 2005. p. 36. ISBN 1-59593-061-2.

[Rappa 2004]RAPPA, M. A. The utility business model and the future of computing services. *IBM Systems Journal*, v. 43, n. 1, 2004.

[Rogers e Cliff 2012]ROGERS, O.; CLIFF, D. A financial brokerage model for cloud computing. *Journal of Cloud Computing: Advances, Systems and Applications*, v. 1, n. 1, p. 2, 2012. Disponível em: <http://dx.doi.org/10.1186/2192-113X-1-2>.

[Sawchuk et al. 2003]SAWCHUK, A. A. et al. From Remote Media Immersion to Distributed Immersive Performance. In: *ETP '03: Proceedings of the 2003 ACM SIGMM workshop on Experiential telepresence*. New York, NY, USA: ACM, 2003. p. 110–120. ISBN 1-58113-775-3.

[Shapiro e Wilk 1965]SHAPIRO, S. S.; WILK, M. B. An analysis of variance test for normality (complete samples). *Biometrika*, v. 3, n. 52, 1965.

[Silva, Veiga e Ferreira 2008]SILVA, J. N.; VEIGA, L.; FERREIRA, P. Heuristic for resources allocation on utility computing infrastructures. *6th International Workshop on Middleware for Grid Computing - MGC*, 2008.

[Thain, Tannenbaum e Livny 2005]THAIN, D.; TANNENBAUM, T.; LIVNY, M. Distributed computing in practice: the Condor experience. *Concurrency - Practice and Experience*, v. 17, n. 2-4, p. 323–356, 2005.

[Vaquero et al. 2008]VAQUERO, L. M. et al. A break in the clouds: towards a cloud definition. *ACM SIGCOMM Computer Communication Review*, ACM, New York,

NY, USA, v. 39, n. 1, p. 50–55, dez. 2008. ISSN 0146-4833. Disponível em: <http://doi.acm.org/10.1145/1496091.1496100>.

[Vogels 2009]VOGELS, W. *Introducing Amazon EC2 Reserved Instances - A way to further reduce IT costs*. March 2009. Disponível em: <http://www.allthingsdistributed.com/2009/03/amazon_ec2_reserved_instances.html>.

[Voorsluys et al. 2005]VOORSLUYS, W. et al. Fostering Collaboration to Better Manage Water Resources. *Proceedings of GCE 2005: Workshop on Grid Computing Portals*, November 2005.

[Weigang, Dib e Cardoso 2004]WEIGANG, L.; DIB, M. V. P.; CARDOSO, D. A. Grid Service Agents for Real Time Traffic Synchronization. In: *WI '04: Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*. Washington, DC, USA: IEEE Computer Society, 2004. p. 619–623. ISBN 0-7695-2100-2.

[Wilter et al. 2005]WILTER, A. et al. The BioPAUÃ Project: A Portal for Molecular Dynamics Using Grid Environment. *Brazilian Symposium on Bioinformatics*, 2005.

[Yochai 2004]YOCHAI, B. Sharing Nicely: On Shareable Goods and the Emergence of Sharing as a Modality of Economic Production. *The Yale Law Journal*, v. 114, p. 273–358, 2004. Disponível em: <http://www.yalelawjournal.org/archive_abstract.asp?id=94>.

[Yu, Buyya e Tham 2005]YU, J.; BUYYA, R.; THAM, C. Cost-based Scheduling of Scientific Workflow Applications on Utility Grids. *Proceedings of the 1st International Conference on e-Science and Grid Computing (e-Science 2005)*, p. 140–147, 2005.

[Zhang e Sivasubramaniam 2001]ZHANG, Y.; SIVASUBRAMANIAM, A. Scheduling best-effort and real-time pipelined applications on time-shared clusters. In: *SPAA '01: Proceedings of the thirteenth annual ACM symposium on Parallel algorithms and architectures*. New York, NY, USA: ACM, 2001. p. 209–219. ISBN 1-58113-409-6.

[Zhu e Agrawal 2009]ZHU, Q.; AGRAWAL, G. A resource allocation approach for supporting time-critical applications in grid environments. In: *Proceedings of the 2009 IEEE International Symposium on Parallel&Distributed Processing*. Washington, DC, USA: IEEE

Computer Society, 2009. (IPDPS '09), p. 1–12. ISBN 978-1-4244-3751-1. Disponível em: <http://dx.doi.org/10.1109/IPDPS.2009.5161023>.

[Zhu e Agrawal 2012]ZHU, Q.; AGRAWAL, G. Resource provisioning with budget constraints for adaptive applications in cloud environments. *IEEE Transactions on Services Computing*, IEEE Computer Society, Los Alamitos, CA, USA, v. 5, n. 4, p. 497–511, 2012. ISSN 1939-1374.

Apêndice A

Sumário da Notação

Símbolo	Significado
$\mathcal{W}$	Carga de trabalho a ser executada;
$\mathcal{P}$	Conjunto de provedores IaaS;
$\mathcal{K}$	Conjunto de contratos estabelecidos com os provedores;
$\mathcal{S}$	Escalonamento da carga de trabalho na infraestrutura disponível;
$\mathcal{A}_i$	Aplicação que compõe a carga de trabalho;
d_i	Demanda computacional da aplicação $\mathcal{A}_i$ no modelo analítico;
tr_i	Instante de tempo em que a aplicação $\mathcal{A}_i$ está pronta para execução;
$u_i(\Delta t)$	Função de utilidade da aplicação $\mathcal{A}_i$;
td_i	*Deadline* da aplicação $\mathcal{A}_i$;
tc_i	Instante de tempo em que a aplicação $\mathcal{A}_i$ é finalizada;
pr_j	Preço de reserva do provedor $\mathcal{P}_j$;
$pu_j(t)$	Preço de uso do provedor $\mathcal{P}_j$ no tempo t;
$\mathcal{K}_j(t)$	Quantidade de unidades de recursos reservada a partir do provedor $\mathcal{P}_j$;
tb_j	Instante de tempo onde inicia o contrato com o provedor $\mathcal{P}_j$;
te_j	Instante de tempo onde finaliza o contrato com o provedor $\mathcal{P}_j$;
$uc_j^i(t)$	Ciclos utilizados no provedor $\mathcal{P}_j$ para execução da aplicação $\mathcal{A}_i$ no tempo t;
$\mathcal{R}(\mathcal{K})$	Custo de reserva do conjunto de contratos $\mathcal{K}$;
$\mathcal{C}(\mathcal{W},\mathcal{S})$	Custo incorrido pela execução da carga de trabalho $\mathcal{W}$ a partir de $\mathcal{S}$;
$\mathcal{U}(\mathcal{W},\mathcal{S})$	Utilidade obtida pela execução da carga de trabalho $\mathcal{W}$ a partir de $\mathcal{S}$;
$\wp(\mathcal{W},\mathcal{K},\mathcal{S})$	Lucro a partir de um planejamento $\mathcal{K}$ para processar $\mathcal{W}$ a partir de $\mathcal{S}$;

$\mathcal{E}(\mathcal{K})$	Eficiência de um contrato $\mathcal{K}$;
$i(t)$	número de ciclos disponíveis em t a partir da infraestrutura local;
$g(t)$	número de ciclos disponíveis em t a partir da grade P2P;
$\mathcal{K}_j^i$	Quantidade de ciclos reservada em um contrato de curta duração;
ts_j^i	Instante de tempo no qual um contrato de curta duração é estabelecido;
β_j^i	Variável que reflete a fração relativa no custo para a reserva e para o uso dos ciclos;
$\varphi_j^i(\Delta t)$	Função que reflete como o preço de reserva por ciclo varia com a urgência do contrato (Δt);
$\gamma_{\mathcal{L}}(\Delta t)$	Custo de manter a infraestrutura local dedicada;
$\gamma(\mathcal{W}, \mathcal{K}, \mathcal{S})$	Custo de reservar e utilizar os ciclos junto ao provedor de computação na nuvem;
$ec_i(tc_i)$	Número estra de ciclos necessários para finalizar a aplicação $\mathcal{A}_i$ no instante de tempo tc_i;
$\mathcal{E}_H$	Eficiência de uma heurística H;

Tabela A.1: Sumário da notação dos principais termos.