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

# Seleção de Portfólio no Contexto de Grades P2P Multi-serviço

Álvaro Vinícius de Souza Coêlho

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 Ciências, domínio da Ciência da Computação.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Redes de Computadores e Sistemas Distribuídos

Francisco Vilar Brasileiro, PhD
(Orientador)

Campina Grande, Paraíba, Brasil

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

C672s Coêlho, Álvaro Vinícius de Souza.
Seleção de portfólio no contexto de grades P2P multi-serviço/Álvaro Vinícius de Souza Coêlho. □ Campina Grande, 2010.
118 f.: il.

Tese (Doutorado em Ciência da Computação) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática.
Orientador: Prof. Ph.D. Francisco Vilar Brasileiro.
Referências.

1. Sistemas Distribuídos. 2. Grades Computacionais. 3. Sistemas Entre-pares. I. Título.

CDU 004.75(043)

"SELEÇÃO DE PORTFÓLIO NO CONTEXTO DE GRADES P2P MULTI-SERVIÇO"

ALVARO VINÍCIUS DE SOUZA COÊLHO

DISSERTAÇÃO APROVADA EM 27.08.2010

FRANCISCO VILAR BRASILEIRO, Ph.D
Orientador(a)

MARCO AURELIO SPOHN, Ph.D
Examinador(a)

JORGE CESAR ABRANTES DE FIGUEIREDO, D.Sc
Examinador(a)

FERNANDO ANTONIO DE CARVALHO GOMES, Dr.
Examinador(a)

JUSSARA MARQUES DE ALMEIDA, Ph.D
Examinador(a)

CAMPINA GRANDE - PB

# Resumo

A melhoria e a popularização da conectividade entre os computadores propiciou um ambiente em que os recursos que uma organização usa para obter a computação que necessita não precisam estar instalados localmente. Pode-se adquirir computação de terceiros, como um serviço. Estes serviços podem ser comprados, mas esta não é a única forma de se obter serviços computacionais externos. Uma alternativa interessante para certos tipos de aplicação, notadamente aquelas que podem ser executadas segundo a estratégia de melhor esforço, é o uso de sistemas que se baseiam na reciprocidade, onde nós oferecem serviços a outros nós esperando receber destes os serviços que necessitarão. Ocorre que, em sistemas assim, os custos e as receitas dos serviços são valorados diferentemente por cada nó, de acordo com seus interesses e com suas características operacionais. Sendo agentes racionais, o interesse dos nós será encontrar mecanismos que maximizem a quantidade de benesses que recebem do sistema, e mesmo seu interesse em permanecer nele dependerá de haver mais vantagens do que desvantagens em efetuar a troca de serviços. Os nós precisam operar obtendo *lucro*. Como o sistema é baseado em reciprocidade, o mecanismo natural para os nós maximizarem seu lucro é selecionando os serviços que lhe provejam a melhor relação possível entre os custos para serem oferecidos e as receitas recebidas em reciprocidade por eles. Infelizmente, não é praticável a implementação de um método que encontre a seleção ótima de serviços, dada a complexidade e o indeterminismo do sistema. Os nós precisam executar algoritmos heurísticos. Neste trabalho, mostramos que a seleção dos serviços impacta fortemente na lucratividade dos nós. Apresentamos inicialmente algumas heurísticas de seleção de serviços que geram bons resultados, embora não em todas as condições ambientais, já que não tratam com o devido cuidado da relação custo/benefício. Implementamos heurísticas que usam uma abordagem de subida de colina para resolver isso. Mostramos que, neste problema, não é possível se avaliar as heurísticas consistentemente e, por isso, definimos uma metodologia que permite avaliá-las comparando-as com um algoritmo de referência. Nossos resultados demonstram que é possível os nós terem lucratividade nesse ambiente, condição necessária para que permaneçam interessados em permanecer nele, embora as seleções feitas pelas heurísticas aqui propostas ainda estejam muito distantes da melhor possível.

# Abstract

The improvement and popularity of connectivity between computers have created an environment where resources used by organizations, in order to achieve the computing they want, don't needed be in their local infrastructure. They can acquire computing from third parties, as a service. These services can be bought, but there is not only one way to obtain external computing services. For some applications, specially that ones that can be performed using a best effort strategy, an interesting alternative is to use reciprocation-based systems, where peers donate services to others by expecting to receive services from them in the future. It occurs that, in such systems, costs and utilities of services are valued differently by each peer, according to his interests and operative characteristics. Being rational agents, peers aim to find mechanisms to maximize the amount of advantage they receive from the system. Moreover, their interest in remaining in the system depends on whether they to achieve more utility than costs when exchanging services. Peers need to operate with *profit*. Considering that the system is based on reciprocation, the natural way to maximize profit is by selecting services that give the best possible relation between their costs and the utility achieved by reciprocity. Unfortunately, it is not feasible to implement an algorithm to find the best possible selection of services, due to the system complexity and indeterminism. Peers need to perform heuristic-based methods. In this work we show that the services selection has a strong impact on peers' profitability. We show services selection heuristics that achieve good results, although this does not occur in all environment conditions, because they do not take care of the cost/utility management. We implemented hill-climbing approached heuristics in order to solve this problem. We showed that, in the service selection problem, it is not possible to consistently evaluate heuristics and, due to this, we defined a methodology that allows us to evaluate them by performing a comparison with a referential algorithm. Our results show that it is possible for peers to have profitability in this environment, that is necessary in order to keep them interested in remaining in the system, although the selections made by the proposed heuristics are still far from the best possible selection.

Este trabalho é inteiramente dedicado à minha **Vitória**. Sem seu Sorriso e sem sua Infância eu não teria conseguido.

# Agradecimentos

Deus, esse que propôs tantos desafios adicionais nesta jornada, forjando-me um guerreiro cada vez melhor.

Para Ayla. Sei que não estou sendo nada original, já que deve ser a pessoa que mais recebe agradecimentos. Alma cuidadosamente esculpida por Papai do Céu. Grato para sempre porque desde o início de minha caminhada eu sei que usei e abusei de seus fantásticos ombro, coração e cérebro.

Aos colegas de sala nestes anos todos, por ouvir as mesmas piadas, responder às mesmas perguntas, resolver os mesmos problemas. Exemplos vivos de compartilhamento de recursos.

E os LSD-anos todos, os de ora e os de outrora, pelo companheirismo, sempre renovando as energias para prosseguir a caminhada.

Paulo Ditarso, o grande baixinho. Mesmo temendo por mim em *New York*. Sempre ali, o cara de sorriso fácil, mente aguda e amizade incondicional.

Sidney Doria, o ex-gordo. Que deixou de ser humano gordo, mas ainda é macaco gordo: quebra qualquer galho.

Fubica, meu chefe. Paciente, humano, crítico, sincero. Por vezes eminentemente dolorosas, sempre decisivas, suas intervenções mantiveram-me no rumo certo.

Vânia, querida ex-esposa, eterna mãe de minha maior Fortuna. Tudo passa, mas o que é realmente importante se imprime indelevelmente em nossa alma.

Meus irmãos Lero e Paulo Sérgio, mais meu Pai, velho querido. Um tanto pelo que sempre foram ao longo de minha vida. Mais especialmente pelo que foram quando precisei muito. Não fazem idéia de o quanto foram fundamentais.

Keka e Cleide, pelos inúmeros favores, pelas conversas e por viabilizar o ambiente LSD.

Aninha e Vera, fiéis amigas infiltradas na COPIN, facilitando toda esta aventura.

Os vizinhos do Flamingo, em especial Kennedy, meu colega de apartamento. Não esquecerei dos momentos alegres, dos churrascos, das festas.

Dinah, Pacele, Vanessa, Rondinele, Valéria. E Adryelle. E meu querido afilhado Arthur. Mesmo não tendo saído tudo como planejado, devo grande parte desta conquista a vocês.

Ah! Não esqueçer: grato também a todas as fábricas de Cerveja do planeta. Bem como às de Café.

E todos os demais que eu porventura tenha omitido. Não subestime sua ajuda, foi muito preciosa. Tampouco minha gratidão. O problema é com a minha memória.

Finalmente, o agradecimento mais especial vai para **Rosa**. Porque eu sempre soube da delicadeza e da sensibilidade das flores. Mas na **minha Flor** encontrei muito mais. É abrigo e apoio. Não há como retribuir tudo o que recebi dela.

# Conteúdo

# Lista de Figuras

# Lista de Tabelas

# Capítulo 1

# Introdução

Nos últimos anos as redes de computadores e a Internet tornaram-se bastante populares, simplificando e promovendo a inteconectividade entre computadores, permitindo um grande compartilhamento de dados e recursos.

Do ponto de vista das organizações, as possibilidades de exploração desse potencial são bastantes variadas, pois as empresas conseguem disponibilizar e coletar informações em seus diversos níveis administrativos. Por outro lado, é possível estabelecer estratégias de cooperação, tornando cada vez mais atraente a visão de software como um serviço que provavelmente será adquirido junto a terceiros. Assim, uma escolha precisa ser considerada na gerência de Tecnologia da Informação (TI): quais recursos devem ser instalados localmente e quais devem ser adquiridos externamente na forma de serviços.

Dessa forma, verdadeiros paradigmas computacionais se estabelecem, com o potencial de transformar a própria indústria de TI, modificando a forma com que os equipamentos de hardware são comprados e projetados [Armbrust et al. 2009].

Este trabalho se insere neste contexto, mais particularmente no caso de compartilhamento de recursos em sistemas entre-pares, doravante referidos como sistemas *peer-to-peer* (P2P).

Sistemas P2P podem ser definidos como redes de computadores em que os participantes (chamados nós) atuam tanto no papel de cliente quanto no de servidor. Em sistemas dessa natureza, os nós estão sempre oferecendo e consumindo recursos uns dos outros. Por esse motivo, um aspecto importante dos sistemas P2P é definir de que forma os nós devem escalonar o uso de seus recursos de forma que a alocação seja o mais justa e eficiente possível.

Este trabalho se insere no contexto de grades P2P, apresentando como contribuições:

a) a definição de ambientes P2P com múltiplos serviços, estudando aspectos concernentes aos custos a que os nós se sujeitam para escalonar serviços, bem como às receitas que são originadas da própria oferta desses recursos; b) um estudo sobre a relação entre os serviços oferecidos por cada nó e a lucratividade por eles conferida, já que consideramos um sistema baseado em reciprocidade; c) a apresentação de heurísticas básicas de seleção de serviços, seguidas de heurísticas mais sofisticadas que usam técnicas de subida de colina para melhorar os resultados; d) definição de um ambiente com resultados experimentais e analíticos que permite avaliar as opções, os limites e os impactos que a seleção de serviços pode ter tanto sobre os nós quanto sobre a própria comunidade P2P; e) definição de uma metodologia que, mapeando o problema da seleção de serviços em grades P2P em um problema análogo, permite a avaliação de métodos de seleção de serviço pela comparação de seus resultados com soluções ótimas, cujo custo computacional é proibitivo em ambientes P2P reais.

O restante desse capítulo delimita o estudo que é apresentado nesse trabalho e o contextualiza. Em seguida apresenta as contribuições que alcançamos e, finalmente, mostra a organização do restante desse manuscrito.

## 1.1 Justificativa

A produção e o consumo de bens são modelados pelas teorias de mercado e, por esse motivo, essas teorias costumam ser propostas como uma abordagem possível para que, num sistema P2P, os nós aloquem seus recursos [Buyya and Vazhkudai 2000]. Todavia, além dos mercados, a produção econômica pode ser organizada em várias formas de compartilhamento comunitário. Particularmente, os sistemas baseados em reciprocidade vem sendo apresentados como uma alternativa viável para o escalonamento de recursos em sistemas P2P [Andrade et al. 2007] [Grothoff 2003] [Cohen 2003]. Ao contrário dos mercados, em que normalmente se pressupõe a existência de mecanismos eficientes para negociação de contratos, auditoria, contabilidade, cobrança e registro de informações bancárias, os sistemas de compartilhamento em geral, e os sistemas baseados em reciprocidade em particular, normalmente prescindem da existência de tais mecanismos para serem estabelecidos e, dessa forma, possuem custos marginais transacionais muito menores [Benkler 2004].

Andrade et al. propuseram um mecanismo de escalonamento de recursos baseado em

reciprocidade muito interessante para grades computacionais P2P, chamado de Rede de Favores (*Network of Favors* - NoF) [Andrade et al. 2007]. A idéia é que em uma grade computacional os nós ofereçam sua capacidade ociosa de processamento aos demais que os solicitem. Em troca, um nó que tenha doado ciclos ociosos no passado espera ser selecionado para utilizar o excedente computacional dos outros nós quando sua própria demanda ultrapassar sua capacidade total de processamento. Nesse caso, diz-se que os nós estão trocando "favores". Usando o esquema da NoF, cada nó mantém, para cada um dos demais nós na comunidade, um saldo - também chamado de *balance* - das interações diretas que ele teve no passado com eles. Sempre que recursos ociosos são disputados por mais de um nó, o nó que gerencia os recursos os aloca proporcionalmente ao débito que ele tem com os requisitantes. Esse mecanismo simples mostrou-se apto a marginalizar os nós "caronas" (*free-riders*) [1] nos sistemas, ao mesmo tempo em que cria um ambiente em que os nós são incentivados a doar o máximo de recursos possível à comunidade.

A NoF foi proposta inicialmente para sistemas em que apenas um serviço (ciclos de CPU) é trocado. Quando os nós estão trocando múltiplos serviços, algumas considerações adicionais precisam ser feitas. Em casos assim, os nós podem trocar um tipo de serviço por outro. Além disso, os nós podem valorar os serviços diferentemente uns dos outros [Mowbray et al. 2006]. Dessa forma, as ligações que se formam entre os nós não são baseadas apenas no comportamento, mas também na lucratividade de suas interações. Mowbray et al. estudaram também o uso de uma versão ligeiramente modificada da NoF em grades computacionais P2P onde os nós trocam múltiplos serviços. Os resultados mostraram que o mecanismo de incentivo da NoF leva a escalonamentos justos e eficientes mesmo em cenários assim. Com efeito, usando a NoF os nós ficam aptos a identificar autonomamente aqueles com os quais é possível manter interações mutuamente lucrativas. Obviamente, já que interações com *free riders* sempre dão prejuízos, estes são marginalizados do sistema, assim como acontece no caso com um único serviço.

Os serviços em grades computacionais podem ser trocados no nível de recurso (CPU, disco, tráfego de dados, etc.), mas também em um nível mais alto, como a execução de um software particular ou a disponibilização de um certo conjunto de dados. Ocorre que os

[1] *free-riders* são nós que, num sistema P2P, operam com reciprocidade abaixo do esperado, recebendo mais recursos do que oferecem

custos a que os nós se submetem para oferecer cada serviço são muito específicos, de forma que nós diferentes provavelmente arcam com custos diferentes. Da mesma forma, a receita que cada nó recebe em função dos serviços que consome também é específica.

Como resultado, cada nó precisa escolher um conjunto de serviços para oferecer, dentre todos os possíveis serviços que são disponíveis, considerando os aspectos de custo e receita. Posto que cada subconjunto de serviços possui um custo específico e retorna uma receita diferente, a seleção desse *portfólio de serviços* tem um impacto direto na lucratividade que um nó pode extrair da grade [Coêlho et al. 2008]. Com disso, as estratégias que ele emprega para definir seu portfólio de serviços precisam ser cuidadosamente escolhidas [Coêlho et al. 2009].

Este trabalho apresenta o problema da seleção de portfólio de serviços considerando suas características sob a ótica da lucratividade dos nós. A relevância deste trabalho está suportada num tripé: primeiro, permite uma compreensão do problema da seleção de portfólio de serviços numa grade P2P com múltiplos serviços, em que os nós os trocam usando mecanismos de reciprocidade. De acordo com nossas pesquisas, este é o primeiro trabalho que se debruça sobre esse tema tendo foco no resultado econômico das operações dos nós. Em segundo lugar, ao entendermos a troca de serviços no contexto que consideramos, podemos estudar mecanismos que permitam os nós executar seleções eficientes de serviços para serem oferecidos à comunidade, de forma a maximizar seu lucro. O terceiro elemento relevante que se encontra aqui é uma metodologia para avaliação da qualidade dos métodos de seleção de serviço, a fim de permitir um estudo a respeito da relação entre as estratégias de definição de portfólio e os ambientes em que os nós estão convivendo.

## 1.2 Contribuições

A contribuição principal desse trabalho é validar a hipótese de que, **em grades computacionais P2P multi-serviço onde a troca de recursos entre os nós baseia-se na reciprocidade, é possível os nós melhorarem sua lucratividade quando constroem portfólios mais adequados às condições em que se encontram**.

Os portfólios mais adequados são aqueles que permitem que uma grande quantidade de recursos seja consumida, na forma de serviços, pelos demais nós da comunidade, mas sem

perder de vista duas variáveis: que a relação entre os nós precisa ser lucrativa para todos, e para isso os nós que consomem recursos precisam ser recíprocos; além disso, os nós podem assumir custos com a oferta de serviço apenas até um certo limite, para que sua própria lucratividade seja maximizada.

Mais especificamente, esse trabalho apresenta as seguintes contribuições.

1. **Estudo dos aspectos de definição dos custos.** É importante observar que este trabalho possui relevância apenas caso os nós de fato valorem os serviços de maneira específica, de forma que um mesmo serviço tenha valores diferentes para nós diferentes. Neste sentido, uma das contribuições é o estudo dos aspectos intrínsecos aos nós a fim de demonstrar a premissa de que os serviços são valorados diferentemente uns dos outros.

2. **Impacto da Seleção de Portfólio na Lucratividade.** Avaliamos a relevância da seleção de portfólio de serviços com relação à lucratividade, medindo o quanto um nó pode variar seu lucro apenas em função dos portfólios selecionados [Coêlho et al. 2008]. Além disso, avaliamos de que forma as características do ambiente impactam nesses resultados.

3. **Relação com o Problema da Mochila.** Mostramos a relação do problema da seleção de Portfólio com problemas clássicos em computação. É importante observar que a seleção de portfólio de serviços se aproxima do *Problema da Mochila*, o que descortina toda uma série de abordagens metodológicas que podem ser aproveitadas. Esse resultado é importante pois as futuras pesquisas nesta área também se depararão com as características e as limitações que apontamos aqui. Todavia, isso impõe a necessidade de estudar de que forma os resultados de trabalhos anteriores sobre este problema podem ser aproveitados em nosso estudo.

4. **Relação com a Seleção de Carteira de Ativos.** Mostramos também a relação do problema da seleção de Portfólio com o problema da *Seleção de Carteira de Ativos*. O problema proposto por Marcowitz em 1952 [Markowitz 1952] é bem definido do ponto de vista formal, ao tempo em que mantém uma relação estreita com o problema da seleção de portfólio de serviços. Por esse motivo, também precisamos visitar os modelos que já foram definidos no contexto da Seleção de Carteira de Ativos a fim de estudar

as soluções já encontradas naquele domínio de conhecimento a fim de que possamos avançar no estudo do problema da seleção de portfólio de serviços. Da mesma forma, deixamos este caminho da relação entre a Seleção de Serviços e o Problema da Seleção de Carteiras antecipadamente pavimentado, de forma que possa ser reaproveitado em trabalhos futuros

5. **Heurísticas básicas de Seleção de Serviços.** Definimos e implementamos estratégias básicas de seleção de portfólio de serviços, avaliando-as sob diferentes condições ambientais e mostrando que forma os nós podem usar os resultados de nossa pesquisa para melhorar seu próprio desempenho [Coêlho et al. 2009].

6. **Heurísticas Avançadas.** As heurísticas básicas que definimos acima sofrem com uma gerência inefiiente do custo/benefício na seleção de serviços. Então, propusemos e avaliamos estratégias mais avançadas para seleção de serviço, com o uso de algoritmos baseados em subida de colina, que conseguem encontrar uma melhor relação de custo/benefício dos portfólios de serviços selecionados [Coêlho and Brasileiro 2010].

7. **Metodologia de Avaliação de Heurísticas.** Definimos uma metodologia para que as estratégias de seleção de serviços possam ser consistentemente avaliadas, comparando seus resultados com a performance de um hipotético algoritmo ideal de seleção de serviços [Coêlho and Brasileiro 2009].

8. **Avaliação da metodologia.** Avaliamos a acurácia da metodologia proposta comparando os resultados que as heurísticas obtiveram quando estudadas segundo a metodologia com os que foram obtidos em ambientes onde os nós efetivamente operam trocando serviços.

Finalmente, considerando que os nós precisam encontrar vantagens para permanecer no sistema, definimos que o nosso critério de sucesso seria a identificação de métodos de definição de portfólios de serviços com os quais os nós mantenham um nível de lucratividade positiva e, desta forma, tenham interesse de permanecer no sistema. Isso mostrará que é possível os nós usarem a seleção de portfólio a fim de melhorar sua lucratividade, tornando o sistema atraente para eles e, assim, validando a hipótese deste trabalho.

## 1.3 Estrutura

O restante desse documento está organizado da seguinte maneira. No Capítulo 2 mostramos os resultados dos nosso estudos em problemas correlatos. A primeira parte do Capítulo dedica-se a tabalhos que lidam com problemas similares ao que estamos tratando aqui: estudos a respeito de múltiplos serviços e a respeito da economia de troca de recursos em grades computacionais. A segunda parte do Capítulo refere-se a problemas que, embora situados em domínios diferentes, apresentam importantes analogias com o problema da seleção de serviço, conforme considerado no contexto deste trabalho: o problema da mochila e o problema da seleção de carteira de ativos.

No Capítulo 3, apresentamos as evidências que justificam uma das premissas básicas de nosso trabalho: os custos dos serviços são variados entre os nós. Nesse capítulo nós apresentamos os componentes de custo para prover serviços em grades computacionais, considerando grades P2P, e mostramos que a natureza heterogênea dos nós numa grade P2P determinam que os custos sejam variados entre eles.

Em seguida, apresentamos a definição do problema que delineia o objeto de nosso estudo, bem como as condições em que isso se dá, seguido de um modelo matemático que é descrito detalhadamente no Capítulo 4.

No Capítulo 5 nós mostramos que, independentemente das diversas condições de operação, bem como de quaisquer aspectos ambientais variados, a lucratividade dos nós é fortemente influenciada pela seleção de serviços que eles fazem. As evidências que obtivemos foram construídas através de experimentos, que são discutidos no decorrer do capítulo.

Se o Capítulo 5 justifica a necessidade de se desenvolver heurísticas para selecionar o portfólio de serviços, o Capítulo 6 mostra os nossos estudos a respeito destas heurísticas, apresentando um conjunto de heurísticas básicas, que apresentam resultados razoáveis mas pecam na gerência da relação custo/benefício dos portfólios, seguidas de heurísticas avançadas, que restringem o orçamento usando uma abordagem de subida de colina, a fim de encontrar melhores condições de operação.

Uma metodologia formal para avaliação de heurísticas de seleção de portfolio de serviços é discutida no Capítulo 7. Naquele capítulo, mostramos que problemas de otimização, como o problema da mochila, usam algoritmos de referência (que buscam uma solução ótima)

para avaliar o desempenho de métodos heurísticos. Em seguida mostramos que não é viável implementar um algoritmo de referência para o sistema, tal como o concebemos, já que o grau de complexidade e o indeterminismo dos sistemas P2P tornam isso proibitivo. Então, definimos um problema híbrido dos problemas da mochila e da seleção de carteira de ativos que permite o uso de algoritmos de referência para avaliar heurísticas de seleção de serviços, imergindo os agentes que as implementam num ambiente tal que eles operem sob as mesmas condições encontradas no ambiente P2P multi-serviços que estamos considerando.

Finalmente, no Capítulo 8, apresentamos as conclusões de nosso trabalho, discutindo as devidas contribuições, e mostramos as atividades futuras que se descortinam como decorrência deste.

# Capítulo 2

# Revisão Bibliográfica

Neste capítulo faremos um levantamento de trabalhos correlatos que nos permitiram definir com mais clareza o problema que estamos estudando, bem como vislumbrar alternativas para lidar com ele. Este levantamento se concentra em dois aspectos distintos. As seções 2.1 e 2.2 são dedicadas aos trabalhos que apresentam relação com o objeto de estudo que estamos tratando aqui: na Seção 2.1 fazemos um estudo dos trabalhos relacionados a ambientes com múltiplos serviços, enquanto na Seção 2.2, os modelos existentes na literatura de economia e troca de recursos em grades computacionais são discutidos e comparados com o que baseia este trabalho. As seções 2.3 e 2.4 são voltadas ao estudo de problemas que foram modelados em domínios distintos, mas que mostram analogia com o problema da seleção de portfólio de serviços que tratamos aqui. Nesse sentido, os aspectos que aproximam o problema da seleção de serviços do problema da mochila, com suas múltiplas variantes, são discutidos na Seção 2.3, enquanto a Seção 2.4 discute os resultados obtidos por estudos na área de economia considerando a seleção de ativos financeiros para compor carteiras de investimento.

## 2.1 Trabalhos correlatos em múltiplos serviços

A seleção de serviços tem sido largamente estudada no contexto de composições de serviços para *Web Services* [Dustdar and Schreiner 2005] [Wang and Vassileva 2007]. Alguns esforços também foram feitos com foco em plataformas de grade [Cheung et al. 2004b] [Cheung et al. 2004a] [Gu et al. 2004] mas ainda considerando a composição de serviços para *Web Services*. Os estudos se debruçam sobre o problema de encontrar um fluxo de processamento

que permita múltiplos nós trabalharem juntos, bem como balancear a carga nas ligações de rede. É bem diferente do problema de seleção de serviços que tratamos nesse trabalho, onde os nós precisam tomar decisões de gerência para oferecer serviços à comunidade.

Não obstante, há muitos trabalhos que tratam de sistemas em que múltiplos serviços são trocados de maneira recíproca. De modo geral, entendemos que há duas abordagens mais comuns para o problema, de acordo com os objetivos dos trabalhos. Por um lado há trabalhos que propõem mecanismos e *frameworks* para incentivar a reciprocidade necessária em ambientes P2P com múltiplos serviços. Outros trabalhos propõem arquiteturas de ambientes que permitem que os serviços sejam trocados.

### 2.1.1 Mecanismos e *Frameworks* de incentivo à reciprocidade

A *Network of Favors* (NoF) [Andrade et al. 2007] [Andrade et al. 2004] é um esquema de reciprocidade que não necessita de confiança ou negociação prévia entre nós. Também dispensa serviços centralizados de contabilidade, cobrança, etc. entre os nós, que são comumente referidos como *serviços bancários*.

Na NoF, os nós mantém um *saldo*, para cada outro nó com quem interagiu no passado, que representa uma agregação do valor dessas interações. Ambos os nós envolvidos numa interação atualizam seus respectivos saldos; se um nó $A$ doa serviços para um nó $B$, então $B$ incrementa seu saldo armazenado para $A$ em algumas unidades que são chamadas *favores*. De sua parte, $A$ decrementa o saldo associado a $B$ por um certo número de favores, possivelmente diferente daquele anotado por $B$. Essa possível diferença reflete um aspecto importante dos ambientes de múltiplos serviços: $A$ e $B$ podem valorar os serviços diferentemente.

Os nós novatos, que interagem pela primeira vez com outro nó, posuem um saldo inicial igual a zero. Além disso, se o valor computado por um nó $A$ em relação a um nó $B$ for negativo, $A$ anota esse valor como sendo igual a zero. Assim, um nó jamais conseguirá obter créditos usando o expediente de deixar e voltar ao sistema com uma nova identificação, numa estratégia comumente referida como "white-wash" [Andrade et al. 2007].

Quando seus recursos ociosos são solicitados por outros, um nó os aloca numa proporção direta com o débito que ele possui com os solicitantes. Se o provedor não possui débito com nenhum dos solicitantes, os recursos são alocados uniformemente entre todos eles. É

importante observar que com o uso do mecanismo da NoF, uma vez que o saldo dos *free-riders* é sempre igual a zero e este é o pior valor de saldo que um nó colaborador pode possuir, um *free rider* jamais terá prioridade sobre um colaborador. Logo, esse processo encoraja a doação de recursos ociosos porque a chance de $A$ ser selecionado por $B$ para receber seus recursos é diretamente proporcional ao número de favores que $A$ possui de saldo junto a $B$.

Em trabalhos mais recentes os autores propuseram uma versão avançada da NoF, chamada *Extended Network of Favors* (ExtNoF) que permite usar os conceitos estabelecidos na NoF para lidar com múltiplos serviços [Mowbray et al. 2006]. A ExtNoF é um mecanismo que incentiva a doação em grades P2P mas que, ao contrário da NoF, que considerava um único serviço, compartilha explicitamente múltiplos serviços como processamento, transferência de dados e armazenamento.

Efetivamente o mecanismo da NoF ataca o problema de desencorajar *free-riding* e habilita os nós a encontrar, na sua comunidade, outros nós com quem eles possam ter trocas de serviços mutuamente lucrativas. No caso da versão da NoF para múltiplos serviços, isso é feito considerando-se que os nós valoram serviços diferentemente uns dos outros. Todavia, a abordagem considera que qualquer nó sempre oferece todos os serviços disponíveis. Isso é pouco realista, porque se considerados num nível mais sofisticado como a execução de *softwares* e compartilhamento de arquivos, a diversidade de serviços é muito grande e, como todos esses serviços têm um custo associado, nem sempre é vantajoso se oferecer todos os serviços. Além disso, devido à limitação de recursos ou mesmo de capacidade financeira para arcar com os custos, provavelmente nenhum nó estará apto a oferecer todos os serviços.

Satsiou e Tassiulas [Satsiou and Tassiulas 2007] propõem um *framework* que permite a gerência das trocas de recursos em ambientes P2P. Com esse *framework* um nó constrói, para cada um dos outros, um *vetor de reputações*, em que cada valor é a reputação de um dos demais nós da comunidade. Uma reputação é um conjunto de informações sobre os serviços providos por um nó (o servidor) a outro (o cliente), e permite saber o quanto ele é confiável como servidor, aos olhos de cada cliente.

A reputação de um nó é definida em função de cada serviço específico da seguinte forma: calcula-se, para cada um dos serviços oferecidos, a razão entre a quantidade de unidades de cada serviço provida pelo nó e a quantidade total de unidades de serviço que foi requisitada a ele. Isso permite que, usando o Vetor de Reputações, um nó possa inferir sobre a confia-

bilidade dos demais e assim tomar uma decisão a respeito da quantidade de recursos que ele deverá doar para cada um dos outros. O *framework* ainda considera o Vetor Local de Reputação de um nó $i$ com relação a um nó $k$ como a soma de todas as reputações de serviços solicitados por $k$ a $i$.

O *framework* proposto efetivamente promove um ambiente P2P de mútua confiança em que se troca múltiplos serviços, e faz isso considerando aspectos de capacidade e demanda dos nós.

Entretanto, ao contrário do trabalho que apresentamos nesta tese, não considera nenhum mecanismo que permita os nós selecionar portfólios de serviços que melhorem sua lucratividade [Coêlho et al. 2009], nem leva em conta cenários em que os serviços tenham valores diferentes e os nós tenham sua lucratividade comprometida caso não considerem cuidadosamente o Portfólio de Serviços a oferecer.

### 2.1.2 Arquiteturas para negociação de recursos em grades

O processo de troca de serviços numa grade P2P é subdividido em três fases distintas [Chun et al. 2003].

1. **Descoberta do recurso** - quais nós possuem um certo recurso para trocar, e a que preço;

2. **Segurança** - o estabelecimento das condições adequadas para ambas as partes envolvidas realizarem a transação;

3. **troca**, com as respectivas contabilidades para doadores e receptores.

É nesta última etapa que situamos o foco de nosso estudo, estudando de que forma a seleção de serviços permitirá que os nós efetivamente promovam trocas de recursos, ao mesmo tempo em que mantem um nível de lucratividade alto o suficiente para mantê-los interessados em permanecer no sistema.

Para permitir que o processo de troca seja levado a cabo, algumas arquiteturas são propostas como os *Virtual Clusters* (VC) [Chase et al. 2003], em que um *pool* de recursos é compartilhado por diversos *sites*, de forma que cada um "possui" um VC que é, na verdade,

um conjunto de máquinas localizadas em *sites* diferentes, mas apresentadas num ambiente integrado como se pertencessem a um único *cluster*.

Além disso, o nó que vai doar pode configurar as máquinas com múltiplos serviços de acordo com as necessidades do nó requisitante. Para isso, é necessário haver uma política de escalonamento prévio pelos nós de forma que ele saiba previamente as requisições que vai atender e assim possa configurar suas máquinas de acordo.

Uma abordagem diferente é o SHARP (Secure Highly Available Resource Peering) [Fu et al. 2003], um ambiente para descoberta e compartilhamento de recursos entre nós que emprega *tickets* como elementos criptografados de delegação de recursos. Um nó que emite um *ticket* dá a outro nó o direito de usar uma certa parte de seus recursos por uma certa quantidade de tempo. A partir de um *ticket* pode-se criar *subtickets* realocando parte dos recursos recebidos. A implementação do SHARP garante que, após emitido, um *ticket* não pode ser perdido ou renegado, bem como pode ser verificado por qualquer nó da comunidade.

O primeiro desses trabalhos foca na disponibilização de um ambiente de compartilhamento de múltiplos serviços, enquanto o segundo trata de garantir o uso dos recursos como um *direito adquirido*, contabilizando-os como uma mercadoria que pode efetivamente ser negociada entre os nós.

Ambos os trabalhos, entretanto, não levam em conta a variação da receita recebida do grid quando diferentes conjuntos de serviços são oferecidos pelos nós, nem tampouco a limitação prática que um orçamento impõe à seleção de serviços.

## 2.2 Economia de troca de recursos em grades

Entre trabalhos correlatos a respeito de modelos para troca de recursos em ambientes P2P, encontramos na literatura abordagens que seguem duas linhas ortogonais entre si: **modelos econômicos** e **modelos de negociação**.

Os modelos econômicos discutem os fundamentos, baseados em teorias econômicas, que sustentam a viabilidade de todos os participantes do ambiente permanecerem interessados em realizar as trocas de recursos. É importante ressaltar que a seleção de portfólio de serviços precisa ser aderente a algum modelo econômico, catalizando um ambiente em que os nós se mantenham interesados em continuar convivendo.

Os modelos de negociação versam sobre o diálogo entre fornecedores e consumidores a fim de negociar as condições em que um recurso será trocado. Dessa forma, um ambiente que permite a troca baseada em reciprocidade é uma alternativa, entre outras, pela qual os nós podem negociar os recursos que são trocados. É nesse ambiente que situamos o contexto de nosso trabalho.

### 2.2.1 Modelos Econômicos

Os modelos teóricos da Economia, particularmente o caso dos mercados, servem para definir ou explicar os aspectos concernentes à troca ou venda de serviços em grades computacionais, ainda que em alguns casos seja necessário implementar características muito específicas [Shneidman et al. 2005]. Entre os modelos econômicos mais encontrados em ambientes P2P e de grades computacionais, podemos agrupar os seguites.

- Mercado de *Commodities*. Este modelo é um mapeamento da bolsa de mercadorias, mantendo-se um local, normalmente referido como *páginas amarelas*, onde os preços são publicados. Os fornecedores de recursos utilizam esse espaço para publicar as ofertas, considerando aspectos de validade e descontos. Amir et al. [Amir et al. 2000] e Brooke et al. [Brooke et al. 2000] definiram sistemas que usam essa aborgagem.

- Preço Estabelecido. Este modelo pode ser visto como uma variação do modelo de Mercado de *Comodities*, sendo que neste caso permite-se uma variação de preços, já que os fornecedores podem fazer ofertas especiais, fixando preços mais (ou menos) atraentes a fim de atrair consumidores para aquecer o mercado ou aumentar o uso de recursos em períodos de pouca procura.

- Barganha. Os modelos anteriores forçam os consumidores a aceitar os preços e as ofertas dos fornecedores. Este modelo permite uma negociação entre as partes, a fim de estabelecer preços e durações da prestação de serviços. Este modelo é explorado por Li e Yahyapour [Li and Yahyapour 2006b] num ambiente de negociação por preço.

- Rede de Contratos. Quando os consumidores/fornecedores podem negociar com vários fornecedores/consumidores ao mesmo tempo, forma-se uma rede de contratos

de prestação de serviços. Basicamente um consumidor anuncia o que deseja, convidando os fornecedores a fazer ofertas a fim de selecionar uma. Do ponto de vista dos fornecedores, há a chegada de uma solicitação de serviços, que é seguida de uma análise a fim de se determinar uma oferta que é enviada de volta. Este é um dos sistemas mais utilizados para negociação de serviços num ambiente distribuído [Smith 1980] *apud* [Buyya et al. 2002]. Exemplos de sistemas que empregam esse modelo são discutidos por Buyya et al. [Buyya et al. 2000].

- Leilão. Neste modelo, que pode ser visto como uma generalização do modelo de barganha, há uma disputa entre consumidores a fim de oferecer o maior preço possível na tentativa de atrair fornecedores ou, no modelo que é referido como *leilão reverso*, entre os fornecedores para oferecer o menor preço para convencer os consumidores. Isso permite a ação de forças de mercado para determinar a flutuação de preços de recursos. Um leilão num ambiente de grade normalmente começa com uma oferta de serviço, e uma sucessão de ofertas se repete até que alguma seja satisfatória (ou que um limite de tempo seja atingido) e o serviço é fornecido ao vencedor [Jennings 2000] [Li and Yahyapour 2006a].

- Compartilhamento proporcional de recursos. Cada consumidor faz uma oferta de créditos junto ao fornecedor a fim de obter os recursos que deseja e a alocação é proporcional ao número de créditos. Isso permite que um consumidor faça ofertas maiores para tarefas mais críticas. O *OurGrid* [Cirne et al. 2006] é um exemplo de grade que usa uma estratégia parecida, pois um fornecedor particiona os recursos entre os consumidores dividindo os recursos proporcionalmente à quantidade de recursos que recebeu destes no passado, que seriam os créditos. Outro exemplo é o *Tycoon* [Lai et al. 2004], um sistema de alocação distribuída de recursos. O *Tycoon* estabelece que os usuários recebam uma certa quantidade de *créditos* e os usam para executar aplicações distribuídas. Quando um recurso está disponível, os usuários o disputam em forma de leilão. Com isso os usuários podem definir suas próprias prioridades de aplicações.

- Comunidade/Coalizão. Os modelos acima, normalmente referidos como *modelos de mercado*, são garantidos em função da existência de mecanismos para negociação

de contratos, auditoria, contabilidade, cobrança, serviços bancários. Uma alternativa são os modelos de *comunidade*, que presumem que uma comunidade de indivíduos compartilha seus recursos para criar um ambiente computacional cooperativo. Todos os indivíduos que contribuem podem usufruir dos recursos do ambiente; por isso, é necessário um modelo que permita decidir quanto do ambiente é devido a cada um dos colaboradores. Chase et al. [Chase et al. 2003] mostram uma arquitetura que propicia a implementação deste modelo. Cirne et al. [Cirne et al. 2006] apresentam um modelo em que o conjunto de recursos ociosos de vários laboratórios somados pode ser tratado como um único ambiente cooperativo de proporções bem maiores. Os ambientes de comunidade permitem que os nós sejam reconhecidos pela sua reputação social, dispensando os aspectos de contabilidade formal, o que diminui os custos transacionais [Benkler 2004].

Além dos modelos considerados, há mercados em que os serviços são oferecidos em contratos de moeda corrente, regidos por alguma legislação de proteção de direitos de fornecedores e consumidores, e tratados como uma prestação de serviço ou uma venda de bem de consumo. Nesses casos os modelos precisam considerar taxas de impostos, inflação, índices de preços ao consumidor (IPC) e quaisquer outras peculiaridades específicas do país/região onde o serviço é prestado [Li and Yahyapour 2006b].

### 2.2.2 Negociação de Serviços

Enquanto os modelos econômicos estabelecem os mecanismos que regulam as relações entre agentes interessados em fornecer e/ou consumir serviços, uma análise ortogonal pode ser feita a respeito do processo de negociação em si, que é o espaço onde fornecedores e consumidores podem atuar, segundo seus próprios interesses, na tentativa de encontrar a condição mais favorável para realizar a troca ou a compra de recursos.

De uma maneira geral os modelos de negociação são de duas naturezas: negociações *por preço*, em que se executa um processo de determinação de preço, possivelmente decorrente de uma seqüência de ofertas entre fornecedores e consumidores, e negociações *tit for tat* [Axelrod 1984], em que os recursos são entregues pelos fornecedores aos consumidores baseando-se na reputação construída pelo histórico de suas interações passadas, que deter-

mina uma expectativa de reciprocidade futura.

### Estratégias de negociação por preço

Numa grade cuja estratégia de negociação é pelo preço, consumidores e fornecedores - muitas vezes referidos como *oponentes* - trocam uma seqüência de ofertas entre si, até que algum preço aceitável para os dois seja alcançado.

Durante o processo de negociação é necessário conciliar os conflitos de políticas e objetivos dos oponentes, caso em que algum preço é encontrado, ou, se não for possível essa conciliação, a negociação é considerada impossível. Ocorre que a tarefa de negociar preços numa grade é muito freqüente, de forma que esta precisa ser feita de maneira automática e transparente para não impactar negativamente na performance [Foster et al. 2004].

Com isso, as estratégias de negociação automática, normalmente baseadas em agentes inteligentes, são exaustivamente exploradas na literatura da área [Li and Yahyapour 2006b] [Andrieux et al. 2004] [Li and Yahyapour 2006a]. Isso pode ser feito em um processo de oferta e contra-oferta (que possivelmente precisará ser repetido), ou num processo multi-*round* em que várias ofertas são trocadas entre os oponentes.

Essas estratégias precisam refletir os diferentes perfis de cada oponente, e os agentes que as implementam precisam ser capazes de manter parâmetros de configuração que os façam atuar de acordo com um perfil variável, de acordo com o lado da negociação - fornecedor ou consumidor - e com características específicas das políticas internas de negociação de cada oponente [Li and Yahyapour 2006b].

Como exemplo do modelo de oferta e contra-oferta, há o *WS-Agreement* [Andrieux et al. 2004], um sistema em que os oponentes lançam propostas entre si e, se ambos aceitarem, o preço fica estabelecido. Em caso contrário todo o processo pode ser repetido no futuro.

Uma variação desse modelo é o caso de múltiplos turnos, onde os oponentes lançam propostas que são analisadas e, em resposta, novas propostas ou uma concordância mútua é gerada. O processo se repete até que algum preço - resultante da concordância mútua - seja encontrado. Nestes modelos, a fim de que o processo de negociação não cause impacto na performance do sistema, considera-se um *deadline* da negociação, a partir do qual assume-se que não há possibilidade de conciliação. Já existem sistemas de negociação em que agentes fazem ofertas e contra-ofertas em busca de uma conciliação, sendo que esses agentes são

flexíveis, capazes de adaptar-se às diferentes estratégias que podem ser empregadas tanto por fornecedores quanto por consumidores [Li and Yahyapour 2006a].

Tipicamente, essas estratégias de preço são usadas em associação com modelos econômicos de mercado, e pressupõem o uso de entidades centrais para cuidar de tarefas como pagamentos, contabilidade, etc. que aumentam os custos de cada transação [Benkler 2004]. Quando se considera os modelos de comunidade, esses aspectos podem ser dispensados em favor de estratégias que usem a reputação construída por cada nó ao longo de sua história.

#### Estratégias de negociação *tit-for-tat*

Um agente que implementa a estratégia *tit-for-tat*, também referida como "retaliação equivalente", inicialmente opta por cooperar com os demais. Em seguida passa a responder de acordo com a ação anterior do parceiro; se houve cooperação, o agente é cooperativo, caso contrário o agente não coopera.

Em sistemas P2P, vários problemas potenciais impactam o nível de confiança alcançado pelo ambiente [Singh 2003] [Selcuk et al. 2004]. Uma alternativa comumente aplicada é um sistema de reciprocidade no estilo *tit-for-tat*, de forma que os nós que colaboram receberão mais do sistema do que os que não colaboram. Este modelo é aplicável a grades desde que os participantes tenham a oportunidade de oferecer recursos em alguns momentos e a necessidade de demandar recursos em outros.

Neste sentido, a *Network of Favors* (NoF) [Andrade et al. 2004] que apresentamos anteriormente também pode ser considerada uma estratégia *tit-for-tat*. A NoF funciona num sistema de grade oportunista, que é concebido num modelo P2P, de forma que cada participante é um nó que, em alguns instantes, possui recursos ociosos para doar e, em outros, tem necessidade de consumir recursos da grade [Andrade et al. 2007]. O aspecto *tit-for-tat* da NoF se dá porque, quando do escalonamento de recursos, os nós priorizam os consumidores dos quais receberam mais recursos no passado.

## 2.3 O Problema da Mochila

O Problema da Mochila (*Knapsack Problem*) tem sido intensivamente estudado desde o trabalho de Dantzig [Dantzig 1957] *apud* [Kellerer et al. 2003], em parte porque teria apli-

cações imediatas mas, principalmente, por motivos teóricos: toda a família de Problemas da Mochila pertence à classe de complexidade $NP$ [Kellerer et al. 2003].

O Problema da Mochila possui múltiplas variações. Em sua forma clássica, é referido como *Problema da Mochila Geral* ou *Problema da Mochila Clássico* [Karp 1975] *apud* [Kellerer et al. 2003]. Muitas vezes as referências tratam o caso em que os itens apenas podem estar ou não na mochila, o chamado *Problema da Mochila* $0/1$, indistintamente como Problema da Mochila, porque a quantidade de problemas relevantes modelados como *Problema da Mochila* $0/1$ é mais representativa [Smolinski 2000] [Kleinberg 2005] [Zhou et al. 2008] [Zhou and Naroditskiy 2008] [Ezziane 2002] [Simões and Costa 2001] [Martello et al. 1999]. Não obstante, trata-se, a rigor, de uma variação - ainda que mais útil - da versão clássica.

O problema foi descrito da seguinte forma [Dantzig 1957] *apud* [Kellerer et al. 2003]: os mascates costumam usar mochilas para carregar os itens que desejam negociar. Ocorre que um certo Mascate possui apenas uma mochila (há variações em que várias mochilas podem ser consideradas), e esta suporta uma capacidade máxima de peso.

O Mascate possui uma série de itens para vender, e cada item tem um valor e um peso associados. Há duas formas de se ver o problema: classicamente, o Mascate possui uma quantidade ilimitada de exemplares de cada item, podendo colocar quantos desejar dentro da mochila, no limite de sua capacidade. A versão $0/1$ do Problema da Mochila propõe que o Mascate possua apenas um exemplar de cada item, podendo apenas levá-lo ou não na mochila ($0/1$ refere-se à quantidade de exemplares de cada item que haverá na mochila).

Logo, a solução do problema é uma otimização combinatória: dado o conjunto de itens, com seus respectivos valores e pesos, e dada uma mochila que não suporta mais que uma capacidade máxima, que combinação de itens o Mascate deve colocar dentro da mochila de forma que o valor total que ela carrega seja máximo? Abaixo segue uma descrição formal do problema.

Dados $n$ tipos de itens, $i_1$ até $i_n$. Cada item $i_j$ tem um valor $p_j$ e um peso $w_j$. A capacidade máxima da mochila é $c$. São dados três vetores: $W =< w_1, w_2, ..., w_n >$, o vetor de pesos; $P =< p_1, p_2, ..., p_n >$, o vetor de valores; e $X =< x_1, x_2, ..., x_n >$ que indica a quantidade de unidades de cada item que vai ser posto dentro da mochila.

O Problema da Mochila pode, então, ser formulado como:

$$\text{Maximize} \sum_{j=1}^{n} p_j \cdot x_j,$$

sujeito a

$$\sum_{j=1}^{n} w_j \cdot x_j \leq c.$$

O *Problema da Mochila* $0/1$, é idêntico ao problema clássico, apenas acrescentando-se a restrição $x_j = 0$ ou $1$.

### 2.3.1 Soluções para o Problema da Mochila

Como se trata de um problema da classe *NP-Difícil*, para o Problema da Mochila não se conhece solução com algoritmo determinístico em tempo polinomial. Na literatura existem três abordagens que são mais comuns para se resolver o problema: o método guloso [Diubin and Korbut 1999] [Magazine et al. 1975], a programação dinâmica [Martello et al. 1999] e os algoritmos genéticos [Simões and Costa 2001] [Ezziane 2002].

O método guloso para o Problema da Mochila foi discutido inicialmente em 1975 [Magazine et al. 1975], e desde então não apresenta variações significativas [Diubin and Korbut 1999]. A estratégia básica é definir uma *densidade* de valor para cada item - o valor dividido pelo peso - e tomá-los ordenados decrescentemente. Considerando o problema da Mochila $0/1$, a abordagem se torna menos eficiente, embora existam soluções com resultados razoáveis [Sarkar et al. 1992] [Calvin and Leung 2003].

Naturalmente, o algoritmo guloso não garante encontrar o melhor conjunto de itens, mas tem alta probabilidade de encontrar boas soluções, e em certos casos pode encontrar soluções muito próximas do ótimo [Diubin and Korbut 1999], com a vantagem de executar em tempo polinomial.

A estratégia com programação dinâmica parte do princípio de que uma mochila de capacidade $c = \gamma$ é formada ou pelo acréscimo de itens a uma mochila ótima de capacidade $c = \gamma'$ ou então por um conjunto de itens que não está em nenhuma mochila de capacidade $c = \gamma'$ onde $\gamma > \gamma'$. Assim, encontrada a solução para uma mochila muito pequena, vai-se tomando mochilas progressivamente maiores, e encontrando a melhor solução para cada

uma delas. No final, quando a mochila alcançar o tamanho adequado, o problema estará resolvido.

A abordagem por programação dinâmica propõe a busca pela solução ótima do problema. Todavia, a complexidade temporal ou espacial ainda é exponencial ($O(2^n)$), de forma que esses algoritmos possuem dificuldade para escalar quando a quantidade de itens aumenta. Estudos recentes conseguiram grandes avanços na performance desses algoritmos [Martello et al. 1999], particularmente quando processados em paralelo [Goldman and Trystram 2004] [Li et al. 2004].

Uma terceira abordagem tradicional para resolver o Problema da Mochila é baseada em algoritmos genéticos [Simões and Costa 2001]. Os algoritmos genéticos formam um paradigma de busca que aplica idéias da teoria da evolução (cruzamento, mutação e seleção natural) para resolver problemas cujo espaço de busca é intratável [Holland 1992]. O ciclo básico de iteração de um algoritmo genético é formado de três etapas: parte de uma população de *cromossomos*, que são formados por *genes*. Cada cromossomo representa um ponto no espaço de soluções potenciais de um problema de otimização [Goldberg 1989], e "evolui" com os processos de cruzamento e seleção a fim de encontrar cromossomos mais aperfeiçoados, e excluir os cromossomos menos capazes.

Para resolver o Problema da Mochila com o uso de algoritmos genéticos é necessário se fazer um mapeamento de seleções de itens para cromossomos. Tomando por exemplo o Problema da Mochila $0/1$, as possíveis soluções são os vetores $X = \langle x_1, x_2, ..., x_n \rangle$ onde $x_i \in \{0, 1\}$ e $0 \leq i \leq n$. Esses serão os cromossomos, e cada $x_i$ é um *gene* do cromossomo $X$ [Ezziane 2002].

A partir daí, um certo número de cromossomos aleatórios é gerado como população inicial, e os processos de mutação e cruzamento são executados, seguidos da seleção dos cromossomos que apresentam melhores desempenhos, até que algum deles consiga um resultado melhor do que um limiar pré-estabelecido, ou então um limite máximo de iterações seja alcançado.

Existem outras abordagens que são empregadas de maneira isolada ou em composição com as alternativas clássicas para o Problema da Mochila, como a *Busca Tabu* [Niar and Freville 1997] e a *Decomposição de Lagrange* [Pirkwieser et al. 2007].

### 2.3.2 Variações do Problema da Mochila

Os diversos aspectos intrínsecos do Problema da Mochila foram estudados separadamente ao longo dos anos, dando origem a diversos problemas que são, na verdade, variações do problema original. As principais variações são discutidas a seguir.

Caso exista uma relação entre o valor e o peso dos itens, particularmente no caso do problema que se tem quando o valor e o peso dos itens são idênticos, então temos o *Subset sum Knapsack problem* [Kellerer et al. 2003].

A capacidade da mochila, se tomada não como uma constante mas como uma função variável, dá origem a problemas como o *Collapsing Knapsack problem* [Wu et al. 2001], em que capacidade total da mochila é determinada por uma função não-crescente do número de itens que precisam ser armazenados.

Se os itens da mochila são categorizados em classes, o problema em que a seleção de itens impõe que pelo menos um item de cada classe seja selecionado é o *Multiple-Choice Knapsack problem*. Este problema foi estudado por Zhou e Naroditskiy no contexto de *keyword bidding* [Zhou and Naroditskiy 2008].

Considerando os valores e os pesos dos itens que podem ser incluídos na mochila, é importante ressaltar os problemas que surgem quando não há prévio conhecimento dos valores dos itens. Nesse sentido é interessante observar o *Online Knapsack problem*, que foi estudado por Zhou e Naroditskiy [Zhou and Naroditskiy 2008], além de Noga e Sarbua [Noga and Sarbua 2005] com diferentes focos de aplicação. O *Online Knapsack problem* estabelece que o valor ou o peso do item só será conhecido no exato instante em que a decisão a respeito de incluí-lo ou não na mochila precisa ser tomada. Tipicamente esse problema é usado para modelar estratégias em leilões, onde o custo (peso) de cada item só é conhecido no instante em que o agente precisa decidir a conveniência de dar um lance. Um caso especial do *Online Knapsack problem* em que os itens possuem custo unitário é referido como *Online Multiple Secretary problem* [Kleinberg 2005].

Os métodos para solucionar o *Online Knapsack problem* acabam tendo desempenhos pouco satisfatórios [Marchetti-Spaccamela and Vercellis 1995], embora alguns resultados interessantes podem ser alcançados quando se estabelece um limiar (que pode ser um valor fixo ou uma função) a partir do qual se decide se um item é selecionado ou não. Os algoritmos podem ir encontrando os melhores limiares à medida em que selecionam os itens, e

assim alcançando resultados cada vez mais competitivos [Zhou and Naroditskiy 2008].

Há uma semelhança considerável entre aspectos do problema da seleção de serviços e o *Online Knapsack Problem* no sentido em que na seleção de serviços os nós também não possuem conhecimento prévio a respeito da receita resultante de cada um dos serviços. Na verdade, conforme discutimos na Seção 7.1.1, não há como se decidir o quanto de uma receita é devida a cada serviço que um nó doou pois a doação pode se dar em função da ociosidade do recurso, bem como por uma fração do saldo que o nó se considera devedor, sendo que esta informação não está disponível para os demais nós na comunidade.

Um outro aspecto que pode ser considerado ao se variar o Problema da Mochila é a quantidade de mochilas. Define-se o *Multiple Knapsack problem* [Chekuri and Khanna 2000] como uma variação do Problema da Mochila em que múltiplas mochilas são consideradas, e o objetivo é distribuir itens nas mochilas tentando maximizar o valor total, mas respeitando a capacidade de cada uma delas. Uma variação deste problema, chamado *Multiple Constrained Knapsack problem*, considera que os valores e os pesos dos itens são iguais em todas as mochilas [Kellerer et al. 2003].

Para solucionar o *Multiple Knapsack problem*, tipicamente se adapta soluções para o Problema da Mochila [Kellerer et al. 2003]. Entretanto, alguns esforços alternativos podem ser encontrados com o uso de algoritmos que fazem buscas probabilísticas no espaço de soluções [Smolinski 2000].

Dessa forma, o *Multiple Knapsack problem* se aproxima do problema da Seleção de Serviços, já que há múltiplos nós, similares às múltiplas mochilas. Todavia, há um aspecto relevante que precisa ser considerado: enquanto as mochilas podem ser usadas passivamente de forma a permitir encontrar o melhor resultado global, no Problema da Seleção de Portfólio de Serviços os nós são entidades autônomas que decidem de acordo com os próprios interesses a forma de preencher seus respectivos portfólios de serviços.

### 2.3.3 Avaliação das soluções para as variações do Problema da Mochila

Existem muitas estratégias diferentes para solucionar o Problema da Mochila e suas variações. Algumas são aplicáveis a todas, ou quase todas as variantes, outras se limitam a aspectos específicos.

Todas as estratégias fazem uma escolha entre desempenho e acurácia, já que o problema

é *NP-Difícil*. Entre as abordagens clássicas, por exemplo, o método guloso e os algoritmos genéticos conseguem executar em tempo polinomial (o método guloso consome menos recursos, embora os algoritmos genéticos tenham mais acurácia [Kellerer et al. 2003]) enquanto a programação dinâmica executa em tempo exponencial ou pseudo polinomial[1].

Entre as abordagens alternativas ocorre o mesmo; utilizam-se de estratégias que abrem mão da melhor solução em troca de uma solução considerada boa, mas executada em tempo polinomial ([Niar and Freville 1997] [Smolinski 2000], [Chekuri and Khanna 2000], [Kleinberg 2005], [Zhou et al. 2008], [Pirkwieser et al. 2007], [Zhou and Naroditskiy 2008]).

Entretanto, a busca pela solução ótima, ainda que a custo exponencial, ainda é necessária por dois motivos. Primeiro porque para alguns problemas é absolutamente imprescindível se obter a melhor solução. Esses casos são contemplados, por exemplo, por abordagens com algoritmos paralelos para manter o custo do processamento em tempos razoáveis, ainda que a complexidade computacional permaneça alta [Li et al. 2004] [Goldman and Trystram 2004]. O segundo motivo é a validação metodológica de soluções heurísticas: os algoritmos que executam em tempo polinomial encontram soluções não-ótimas para o problema usando heurísticas. Faz-se necessário, então, uma avaliação a respeito da qualidade da solução encontrada, ou seja, o quanto essa solução se aproxima da solução ótima. Nesse sentido, as metodologias usam os métodos de busca exaustiva como algoritmos de referência.

Em função das similaridades entre o problema da seleção de serviços e o problema da mochila, e devido ao fato de este último apresentar peculiaridades interessantes e úteis ao nosso estudo, no aspecto da complexidade computacional, voltaremos a discutir o problema da mochila no Capítulo 7, onde definiremos o problema da Seleção de Serviços como uma variação do problema da mochila.

## 2.4 Seleção de Carteira de Investimentos

A seleção de carteira de investimentos é um problema que foi apresentado originalmente pelo economista americano Harry Max Markowitz [Markowitz 1952] e desde então vem gan-

[1]Há implementações que criam uma tabela com $n$ linhas, $n$ o número de itens, e $c$ colunas, $c$ o tamanho da mochila. Esse algoritmo executa em $O(n \cdot c)$. Entretanto o valor de $c$ pode gerar uma alta complexidade espacial.

hando importância e recebendo a atenção de estudiosos em economia, matemática, pesquisa operacional e computação.

A partir da versão inicial, o problema foi investigado e adaptado a diferentes condições, de forma que vários modelos e várias extensões foram propostas, estudando aspectos como custos transacionais, taxas, decurso de tempo, informação sobre os ativos, etc. Abaixo mostramos uma descrição detalhada do problema.

Um investidor tem acesso a um conjunto de ativos de investimento, e cada um deles proporciona uma rentabilidade, embora também esteja associado a um risco. Uma carteira é a decisão do investidor de quanto de seu orçamento ele vai gastar com cada um dos ativos. O problema é conciliar o risco e o retorno do investimento.

A rentabilidade futura é um valor desconhecido, mas possui uma distribuição histórica que apresenta uma média e algum desvio-padrão. Para efeitos da análise de Markowitz, a média histórica é tomada como o valor esperado de rentabilidade do ativo, mas com um *risco*, que é determinado pelo desvio-padrão. De posse dos ativos de investimento com seus valores históricos que retornaram, qual carteira maximiza o retorno esperado, minimizando os riscos?

O método mostrado por Markovitz, chamado de variança média, calcula a variança de uma carteira como a soma das varianças individuais de cada ativo, o que diminui o risco médio do investimento. Por outro lado, ele também considera as covariânças entre pares de ativos da carteira de acordo com o peso de cada ação na carteira [Markowitz 1952], a fim de estimar o risco total do investimento.

Formalmente o problema da seleção de carteira, tal como abordado por Marcovitz, pode ser descrito como [Xia et al. 2000]

Maximize

$$(1-w)\cdot\sum_{i=1}^{n} R_i\cdot x_i - w\cdot\sum_{i=1}^{n}\sum_{j=1}^{n}(\sigma_{ij}\cdot x_i\cdot x_j)$$

Sujeito a

$$\sum_{i=1}^{n} x_i = 1$$

$$x_i \geq 0, i = 1...n.$$

Onde $n$ é o número de ativos, $x_i$ é a proporção do orçamento que foi investida no ativo $i$,

$R_i$ é o retorno esperado do ativo $i$, $\sigma_{ij}$ é a covariança do retorno esperado dos ativos $i$ e $j$, e finalmente $w$ é o fator de aversão ao risco do investidor, tal que $0 \leq w \leq 1$.

Note que quanto menor for a covariança entre os ativos $i$ e $j$, menor será o risco para um fornecedor que inclua os dois ativos na sua carteira pois as variações sofridas por um deles não afetarão o outro e vice-versa. Por isso, o fator de aversão ao risco ($w$) modela o perfil do investidor: se um investidor for muito arrojado ($w \rightarrow 0$) ele vai desprezar as possíveis influências de um ativo sobre outro. Por outro lado, investidores extremamente conservadores ($w \rightarrow 1$) darão atenção apenas ao risco proporcionado pelos ativos.

O retorno e o risco de uma carteira são, respectivamente, determinados por;

$$R_p = \sum_{i=1}^{n} R_i \cdot x_i$$

e

$$\sigma_p^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} (\sigma_{ij} \cdot x_i \cdot x_j).$$

### 2.4.1 Soluções para o problema da Seleção de Carteira

Existe uma infinidade de abordagens diferentes para solucionar o problema da seleção de carteira em suas variadas formas, e sob diferentes circunstâncias. Quando se considera a busca pela melhor solução, trata-se de um problema de otimização que pode ser resolvido com as abordagens já conhecidas na Ciência da Computação. Entre as abordagens mais empregadas nas pesquisas mais recentes destacam-se o uso dos algoritmos genéticos [Zhang et al. 2006] e da otimização por enxame de partículas [Xu et al. 2007].

Os algoritmos genéticos foram mostrados na Seção 2.3.1, aplicados ao problema da mochila. No caso da seleção de carteiras, cada possível carteira é um cromossomo, e o percentual do orçamento que fica investido em cada ativo são os genes.

O processamento segue como mostrado na Seção 2.3.1, com uma adaptação para as operações de cruzamento. Note que cada cromossomo é uma seqüência de números reais $x_1, x_2, ...x_n$ onde $\sum_{i=1}^{n} x_i = 1$. Ocorre que o processo tradicional de cruzamento apenas gera cromosomos que herdam os genes de um dos cromossomos que "cruzaram". Dessa forma os cromosomos-filho teriam os mesmos genes (e portanto os mesmos valores para os ativos) de um dos pais, restringindo o espaço de busca do algoritmo. Para evitar esse efeito, para

cada par de genes $x'_i$ e $x''_i$ que vão cruzar gera-se um número $\alpha$ ($0 \leq \alpha \leq 1$). O gene $x_i$ é calculado como $\alpha \cdot x'_i + (1 - \alpha) \cdot x''_i$.

O enxame de partículas [Eberhart and Kennedy 1995] é inspirado no comportamento de grupos de animais, como enxames de abelhas, cardumes de peixes ou revoadas de pássaros, que exploram uma área em busca de comida. A idéia básica é fazer uma busca usando um conjunto de soluções geradas aleatoriamente, que correspondem a indivíduos. Cada solução potencial, chamada de *partícula*, ocupa uma posição no espaço mas se move com uma velocidade variada. Além disso, cada partícula conhece o melhor ponto em que já esteve, definindo-se "melhor ponto" como aquele em que a maior quantidade de alimento foi encontrada, bem como o melhor ponto das partículas vizinhas. O conceito é que a cada instante as partículas vão mudar sua velocidade (vetorial) de acordo com os resultados que teve. Cada partícula tende a aproximar-se um pouco da melhor solução encontrada pelas vizinhas, mas ao mesmo tempo não se afasta muito da melhor solução encontrada por ela mesma. Essa ponderação entre a melhor solução própria e da vizinhança é ajustada por alguns parâmetros como a *inércia* da partícula (o quanto ela vai variar sua velocidade a cada iteração) e dois parâmetros chamados *cognitivo* e *social*, que definem o quanto ela confia na própria solução e na solução encontrada pelo restante das partículas.

O processo é a busca das partículas por soluções cada vez melhores até que alguma delas consiga encontrar uma solução que supere um limiar pré-estabelecido ou então até que um certo número de iterações seja alcançado. Aplicado ao problema da seleção de carteiras, estabelece-se que, a cada ativo, corresponderá uma dimensão do espaço, e o processo segue como mostrado acima [Xu et al. 2007].

### 2.4.2 Variações do problema da Seleção de Carteira

A seleção de carteira proposta por Markowitz faz uma série de simplificações, entre as quais uma a respeito do comportamento do operador financeiro. Em casos reais há um custo associado à mudança de carteira (venda ou compra de ativos) que Markowitz considerou desprezível na formulação original do problema. Nesse contexto, a *Seleção de Carteira com Custos Transacionais* [Almgren and Chriss 2001] é a reprodução do problema original, com a consideração dos custos de transação que acabam impactando na rentabilidade final obtida pela carteira considerada.

Ao se considerar os custos transacionais, acrescenta-se uma dimensão a mais ao problema da seleção de carteiras porque as transações se sucedem no tempo. Nesse sentido, mais variações do problema da seleção de carteiras decorrem do fato de que a solução a ser encontrada não fica restrita a uma única seleção, mas a seleções periódicas [Samuelson 1969] ou contínuas [Campbell and Viceira 1996] de carteiras. Nesses casos o objetivo do investidor é maximizar a rentabilidade ao longo de algum intervalo de tempo, ou mesmo por tempo indeterminado, o que define o problema da *Seleção Perpétua de Carteiras* [Samuelson 1969].

Do ponto de vista da Seleção de Portfólio de Serviços, a Seleção Perpétua de Carteiras apresenta uma semelhança imediata, porque muitas estratégias que um nó usa para selecionar serviços melhoram suas performances de lucratividade exatamente porque fazem seleções variadas ao longo do tempo [Coêlho et al. 2009] [Coêlho and Brasileiro 2010], e as seleções se sucedem indefinidamente. Além disso, ao longo do tempo é esperado também que a rentabilidade dos ativos varie [Campbell and Viceira 1996], o que também ocorre com a Seleção de Serviços.

A relação entre o investidor e a rentabilidade dos ativos também é alvo de estudos específicos. O modelo de Markovitz considera que a rentabilidade dos ativos segue uma distribuição normal. Para modelar o desempenho de ativos em situações heterodoxas (como turbulências, inflação, crises de crédito, etc.) a variação da rentabilidade deve ser modelada com distribuições específicas [Tanaka et al. 2000].

Apesar de a receita futura dos ativos ser previamente desconhecida, pode-se considerar o problema da seleção de carteiras com alguma informação preditiva quando se assume que o investidor conhece a ordem (crescente ou decrescente) em que os ativos são classificados de acordo com a rentabilidade esperada. Esta variação é conhecida como *Carteira com Ordem de Rentabilidade Esperada* [Xia et al. 2000].

Numa grade P2P como a que estamos considerando, os nós também podem fazer uma estimativa a respeito da classificação dos serviços de acordo com a receita esperada. Isso é possível porque quanto mais recíprocos forem os nós a quem se doa serviços, maiores serão as receitas esperadas. Dessa forma, os nós podem computar de alguma forma a reciprocidade que eles obtêm dos demais a fim de determinar aqueles com quem as interações foram mais lucrativas e, assim, selecionar os serviços solicitados considerando uma ordem de receita

esperada.

O problema da seleção de carteiras será revisitado no Capítulo 7, onde consideraremos os modelos usados para descrevê-lo como inspiração para metodologia que permitirá aferir a performance dos métodos usados para se resolver a Seleção de Serviços.

## 2.5 Considerações gerais

Os modelos econômicos que determinam os mecanismos pelos quais os serviços podem ser trocados num ambiente de grade computacional são classificáveis de acordo com certos paradigmas. O problema da seleção de serviços que nós estudamos aqui é pertinente a modelos de *Comunidade*, em que os nós compartilham os serviços, trocando-os segundo mecanismos de regulação social. Mais especificamente, dentre as possíveis estratégias de negociação, o foco de nosso trabalho está em negociações *tit-for-tat*, já que os nós recebem recursos dos demais na forma de serviços e pagam na mesma moeda.

Naturalmente é possível fazer considerações semelhantes e situar o problema da Seleção de Serviços num contexto de negociação por preço, e ele permaneceria com muitos aspectos inalterados, embora fosse afetado mais fortemente na sua relação com o problema da Mochila e com a Seleção de Portfólios.

Dentre os múltiplos trabalhos que consideram mecanismos e *frameworks* para se negociar múltiplos serviços, o foco difere do que propomos ter aqui. Enquanto naqueles se propõem métodos pelos quais os serviços podem ser trocados, aqui o objetivo é encontrar estratégias que permitam maximizar o ganho com as trocas. Essas estratégias não são dependentes dos métodos, e podem ser aplicadas a eles indistintamente, com o objetivo de maximizar a lucratividade dos nós.

Em computação há um problema clássico que toca o problema da Seleção de Serviços: o problema da Mochila. Em sua versão clássica ele apresenta similaridades evidentes mas pouco úteis. Não obstante, duas variações apresentam similaridades particulares com o problema da Seleção de Serviços: o *Multiple knapsack problem*, que considera múltiplas mochilas, assim como existem vários nós, e o *Online knapsack problem*, em que os valores dos itens não são previamente conhecidos, assim como os nós não têm como saber qual a receita resultante de cada serviço.

Os estudiosos em economia também lidam com um problema que guarda particulares semelhanças com a Seleção de Serviços: a seleção de carteiras de ativos, conforme mostrado por Marcowitz. Mais especificamente, duas variações do problema de Marcowitz se aproximam do problema da seleção de serviços: a Seleção Perpétua de Carteiras, já que os nós convivem por tempo indeterminado e fazem contínuas seleções de serviços, e a Seleção de Carteira com Ordem de Rentabilidade Esperada, em que o investidor conhece a ordem dos ativos em função da rentabilidade esperada, assim como os nós podem fazer, considerando os serviços que são solicitados por parceiros mais ou menos recíprocos.

# Capítulo 3

# Dos custos e receitas

Neste capítulo discutiremos as características que definem um ambiente de grade P2P com múltiplos serviços, mostrando os fatores que determinam a variação de custos e receitas entre os nós. Mostraremos como o custo de um mesmo serviço pode variar entre os nós, definindo o cenário que motiva nosso trabalho: um ambiente em que os nós trocam recursos na forma de serviços, sendo que cada serviço tem um valor específico para cada nó.

## 3.1 Grades P2P com serviços

Uma grade computacional pode ser definida como "Uma infraestrutura de hardware e software que provê acesso confiável, consistente, pervasivo e barato a recursos computacionais" [Foster 2002], permitindo que tais recursos sejam compartilhados e que problemas sejam resolvidos em colaboração multi-institucional [Nabrzyski et al. 2004]. Apesar de não haver consenso entre diferentes autores a respeito da definição de grade computacional, uma característica importante na literatura é a recorrência ao termo "Recurso Computacional".

Observando o uso desta expressão em diversas fontes, podemos afirmar que ela se refere a um elemento de hardware ou software que está disponível numa grade. Concede-se o uso de recursos aos diversos membros da comunidade na forma de serviço: quando se aloca um recurso computacional para algum cliente, efetivamente está-se usando esse recurso para prestar um serviço. Este serviço pode ser elementar, como a concessão de algum tempo de uso de CPU, de bytes de alocação de disco ou conectividade para transferência de dados, mas também pode ser considerado numa ordem mais sofisticada como a execução de softwares

ou a disponibilização de uma base de dados. Por esse motivo, podemos definir um serviço como a execução de algum trabalho útil a partir de um recurso computacional instalado. Numa grade computacional, esses serviços são fornecidos e consumidos pelos diferentes participantes. Como exemplo de serviços que podem ser trocados em grades computacionais podemos citar a execução de algoritmos em paralelo (simulação, previsão, mineração de dados, etc.), o armazenamento e a distribuição de arquivos, o gerenciamento de bancos de dados, entre outros.

Compartilhando recursos em um ambiente baseado em reciprocidade, o conjunto de serviços doados por um nó impacta na quantidade de receita que ele vai receber dos demais, já que há uma correlação entre recursos doados e recebidos [Coêlho et al. 2008]. Isto posto, torna-se necessário escolher os serviços que sejam efetivamente consumidos pelos demais, e ainda selecionar consumidores com os quais a parceria seja lucrativa. Ocorre que há custos decorrentes da própria oferta de serviços, e receitas resultantes de recursos doados em reciprocidade pelos nós que os consomem.

Conforme a maneira como estejam sendo utilizados os recursos podemos ver que, quando tomados num nível mais alto de sofisticação, a variedade de fatores que compõem os custos dos serviços tende a aumentar.

## 3.2 Fatores de custo para provedores de serviço

Segundo Opitz [Opitz et al. 2008] os custos para se prover serviços em grades computacionais podem ser subcategorizados em quatro grupos.

1. Custos de equipamento
2. Custos de negócio
3. Sistema e Pessoal
4. Comunicação de dados

A essa lista nós ainda podemos acrecentar os custos com segurança [Farahmand et al. 2005].

Neste trabalho levantaremos os custos decorrentes da concessão de serviços em grades computacionais, usando dados referentes aos custos praticados na comunidade européia, e tomando por base as grades EGEE (*Enabling Grids for E-Science*) [Duarte et al. 2007], bem como a grade Novartis Grid (uma grade de propriedade da fabricante homônima de medicamentos e que aloca máquinas ociosas para sua operação) [McKee 2005]. Todavia, faremos sempre uma análise ponderada, ao contrário da maioria das fontes de onde extraímos os dados, porque estamos assumindo uma grade oportunista, em que os recursos são usados pela comunidade apenas quando estão ociosos. Dessa forma, os valores que apresentamos são sempre considerados proporcionalmente ao tempo em que os recursos ficam disponíveis para a grade.

### 3.2.1 Custos de equipamento

Os custos de equipamento são os custos decorrentes da aquisição e substituição dos mesmos. Como neste trabalho estamos tratando com o caso em que somente recursos ociosos são oferecidos à grade, consideramos que nenhum recurso será adquirido para este fim e então os custos com aquisição de equipamentos serão desconsiderados.

Para calcular os custos com substituição de equipamentos é necessário considerar o **Tempo Médio entre Falhas** (MTTF - *Mean Time to Failure*). Alguns equipamentos, como discos, possuem uma alta sensibilidade às condições de usabilidade. Um experimento considerando discos submetidos a diferentes cargas de trabalho (I/O), mostrou que a quantidade média anual de falha em discos varia muito em função das condições de usabilidade [Pinheiro et al. 2007]. Considerando *muita carga* os discos com maior carga de I/O (os $25\%$ com mais carga) e *pouca carga* os com menor carga de I/O (os $25\%$ com menos carga) encontrou-se uma média aproximada de $4\%$ (*muita carga)* a $1\%$ (*pouca carga)* de falhas ao ano para discos considerados *novos* (com até $6$ meses de uso) e números entre $5\%$ e $1\%$ para discos com cinco anos de uso. Além da carga de uso, fatores como temperatura (a quantidade de falhas pode variar entre aproximadamente $3\%$ e $9\%$ para discos novos, chegando a faixas entre $4\%$ e $15\%$ para discos antigos [Pinheiro et al. 2007]) bem como a própria idade dos discos também influenciam na quantidade de falhas, porque as maiores taxas de falhas em disco ocorrem com discos no primeiro - provavelmente por defeitos de fabricação - ou depois do quinto ano de operação - em função da própria idade do equipamento [Schroeder and Gib-

son 2007]. Outros fatores como softwares (os *drivers* de disco) em diferentes plataformas também geram taxas diferentes de falhas [Chou et al. 2001].

Para cada tipo de falha decorre um custo específico. Esse custo pode ser em horas de trabalho (pessoal) ou mesmo a substituição do equipamento, para o caso de falhas extremas. Além disso, há que se computar o custo com a execução de procedimentos para recuperação da integridade de dados (rotinas de *scan*, restauração de backup, etc.) bem como o custo decorrente do tempo de indisponibilidade do recurso ou do sistema.

Considerando os custos decorrente de falhas, se tomamos um serviço hipotético que demanda uma certa quantidade típica de operações em disco quando está sendo executado, teremos que este serviço gerará um certo custo se estiver executando num disco novo, e um custo diferente se estiver executando num disco antigo. Da mesma forma há que se considerar se ele está coexistindo com aplicações que demandam diferentes cargas de trabalho, ou sujeito a fatores ambientais como temperaturas diferentes, ou versões variadas de sistemas operacionais e de *drivers*. Todos são fatores que também modificam o custo de se prover o serviço. Note que esses fatores podem se superpor, aumentando ainda mais a variação de possíveis valores de custo para um mesmo serviço. Assim, como os nós operam seus equipamentos sob diferentes condições ambientais e de carga, o *custo de equipamento* esperado para oferecer um mesmo serviço deverá ser diferente entre eles.

Em uma análise de custos associados a CPUs, sistemas de arrefecimento, fontes e placas-mãe as considerações que podem ser feitas são semelhantes. Valores específicos das possíveis variações de custo desses recursos considerando as necessidades de substituição são mostrados na Tabela 3.1[1] [Opitz et al. 2008].

### 3.2.2 Custos de negócio

Os custos de negócio são os custos para se manter em operação os equipamentos que provêem serviços tanto de interesse específico do nó quanto, se ociosos, para a grade. Uma parte desse investimento é a aquisição de equipamentos acessórios como refrigeradores de ar, *switches*, cabeamento estruturado, sensores de incêndio, alarmes, etc. Como estamos

[1]O limite superior extremamente alto para custo de CPU refere-se a processadores que não são usados normalmente, e os nossos estudos não os consideram, embora esses preços sejam realmente praticados para equipamentos muito sofisticados.

Tabela 3.1: Custos de substituição de diferentes equipamentos de hardware

| Componente | Custo (Euros) | *MTTF*(h) | Custos anuais de Substituição (Euros) |
|---|---|---|---|
| CPU | 120 a 7.200 | 80.000 a 360.000 | $1,2$ a $320,0$ |
| Arrefecedor de CPU | 30 a 80 | 50.000 a 150.000 | $1,8$ a $14,0$ |
| Fonte | 20 a 80 | 50.000 a 80.000 | $2,2$ a $14,0$ |
| Placa mãe | 90 a 250 | 80.000 a 170.000 | $4,5$ a $27,0$ |
| Disco | 50 a 400 | 300.000 a 600.000 | $0,5$ a $8,0$ |

considerando que os nós doarão apenas recursos ociosos, assumiremos que, assim como acontece com recursos computacionais, nenhuma infra-estrutura específica é necessária para operar na grade.

Entretanto, há um custo relacionado ao consumo de eletricidade de acordo com o tipo de serviço que é executado. Um experimento considerando o uso do `Seti@home`, uma grade computacional oportunista que usa os processadores a ela doados para a execução de programas que tentam decifrar algum tipo de comunicação extraterrestre nas ondas que são captadas no espaço, mostrou que, para diferentes sistemas, o consumo de energia variou entre 177W e 198W para diferentes CPU's executando o mesmo serviço [Opitz et al. 2008]. Além disso, a diferença média de consumo de uma CPU quando executando o serviço ou quando ociosa ficava em torno de $68\%$. Isso nos permite duas considerações: que o consumo de energia ao executar diferentes serviços é variado, já que sujeitarão a CPU a cargas diferentes, e que um mesmo serviço vai consumir diferentes quantidades de energia em diferentes nós, já que estes provavelmente utilizam CPUs diferentes. Sobre todas essas variáveis, também devemos considerar os diferentes preços praticados pelos fornecedores de energia elétrica.

### 3.2.3 Custos de sistema e de pessoal

Os custos de sistema são relacionados à própria configuração do *middleware* da grade e de aplicações que serão executadas. Assumiremos que as escolhas serão sempre por produtos *freeware*, de forma que, para prover serviços, os nós não precisem arcar com ônus de licenças ou direitos autorais. Além disso, os custos com pessoal podem sofrer um impacto com a seleção de serviços, porque alguns serviços podem demandar a execução de rotinas manuais

para configuração ou atualização de arquivos.

A quantidade de falhas também interfere nos custos de pessoal, porque há muitos tipos de falhas que demandam trabalho de pessoas para reintegrar o sistema, o que pode acarretar pagamento de horas extras ou regimes de sobreaviso, a depender da legislação trabalhista a que o nó está submetido. Os aspectos legais da relação de trabalho variam tanto em função dos diferentes valores a serem pagos pela hora de trabalho quanto porque existem sistemas legais que determinam uma grande quantidade de encargos e direitos trabalhistas enquanto em outros esses aspectos praticamente inexistem.

### 3.2.4 Custos de comunicação de dados

Os custos de comunicação de dados são ligados às configurações de rede local e de conexão à Internet. Mais uma vez, como consideramos que nenhuma infra-estrutura específica precisará ser adotada para a operação da grade, assumiremos que os custos de comunicação de dados gerados pela oferta de serviço na grade serão desprezíveis, ainda que eventualmente impliquem na inviabilidade de se oferecer algum serviço.

### 3.2.5 Custos de segurança

Os custos com segurança são decorrentes da exposição de equipamentos locais para execução de programas de origens as mais diversas. Desprezando a possibilidade da execução de códigos maliciosos, há a possibilidade de falhas de segurança inadvertidas, por descuido, imperícia ou imprevidência do proprietário do código. Essas falhas podem gerar *incidentes*, que causam algum dano à operação da organização que provia o serviço. Os danos, por sua vez, podem ser de naturezas distintas e não mutuamente excludentes, como [Opitz et al. 2008]: informação, software, hardware, pessoal e sistemas. Por exemplo, o custo por hora de um incidente que interrompa toda a operação de um sistema pode ser calculado considerando aspectos de produtividade, de negócios interrompidos ou perdidos, de performance financeira e até mesmo de reputação. Além dos custos causados pelos incidentes decorrentes de falhas na segurança, há que se considerar o custo para prover segurança a fim de minimizar o risco de ocorrerem incidentes dessa natureza, com a implementação de programas de proteção (anti-vírus), *firewall*, etc.

Para simplificar, nós consideraremos que os custos com segurança incluem apenas os custos com as medidas preventivas de segurança, como treinamento de pessoal, instalação e manutenção de softwares de defesa, etc. Os custos provocados por incidentes serão classificados como custos de manutenção de equipamentos.

## 3.3 Composição do custo de serviços numa grade

O custo de um serviço é resultante de duas análises distintas: por um lado é necessário considerar o custo de possuir e de manter os equipamentos. Esses custos são comuns a todos os serviços, e mesmo que o nó escolha não instalar nenhum serviço para oferecer, ainda assim terá que arcar com eles. Por outro lado, há os custos da própria operacionalização (instalação e execução) dos serviços, que são específicos.

Tomando por exemplo o Novartis *Grid*, podemos detalhar os fatores que compõem os custos de substituição de equipamentos considerando os valores da tabela 3.1, tomando por base uma garantia de 2 anos para os equipamentos. Esses valores foram encontrados assumindo que para cada CPU haverá um sistema de arrefecimento, uma placa-mãe, uma fonte e um disco. Os dados estão descritos na Tabela 3.2.

Tabela 3.2: Custos de substituição de equipamentos do Novartis *Grid*

| Componente | Quantidade | Custos anuais de Substituição (Euros) |
|---|---|---|
| CPU (Pentium4) | 2.700 | 8.200 a 73.000 |
| Arrefecedores de CPU | 2.700 | 10.000 a 27.000 |
| Fonte | 2.700 | 9.600 a 59.000 |
| Placa mãe | 2.700 | 5.900 a 24.000 |
| Disco | 2.700 | 1.400 a 11.000 |

O Novartis *Grid* é particularmente interessante para nosso estudo porque é uma grade que usa recursos ociosos do sistema. Assim, os dados da Tabela 3.2 consideram apenas os custos decorrentes de aproximadamente 10 horas por dia de uso, que é o tempo estimado médio em que os equipamentos estão trabalhando para a grade. Os valores totais gastos com aquisição e manutenção dos equipamentos são, naturalmente, superiores aos que mostraremos aqui.

Se a operação do Novartis se der nas melhores condições para cada equipamento, e a carga de processamento for mínima, espera-se que os custos dos equipamentos sejam próximos dos limites inferiores. A situação inversa eleva os custos para os níveis superiores. A esses custos precisamos acrescentar o consumo de energia elétrica. Para isso consideraremos a variação média de valores obtidos com CPU's quando ociosas e quando executando `Seti@home`. O consumo da CPU, supondo uma carga de processamento similar à do `Seti@home`, e os preços praticados nos países que compõem a União Européia, pode chegar a $113$ euros por ano, por CPU. Assumindo a variação média encontrada, de aproximadamente $68\%$ de consumo entre CPU's ociosas e executando, o valor mínimo esperado pode ser igual a $67,27$ euros. Considerando o número existente de $2.700$ CPU's, encontramos um custo mínimo de $181.616$ euros por ano.

Assumindo, para ilustrar, que as condições de operações são as ideais, e que todos os equipamentos permaneçam ociosos durante a operação da grade, podemos considerar que os custos com substituição de equipamentos estejam nos limites inferiores que motramos na Tabela 3.2. Esse valor, por CPU, chega a $13$ euros. Portanto, teremos valores em torno de $67,27$ (energia) mais $13$ (equipamento) euros, num custo total por CPU de $80,27$ euros por ano.

É importante lembrar que estamos desconsiderando os custos decorrentes de interrupções de operações em caso de incidentes, bem como custos adicionais com segurança. Além disso, assumimos que a carga máxima a que os processadores estarão sujeitos é similar à do processamento do `Seti@home`. Apesar dessas simplificações, podemos considerar que esse custo é o minimo a que o provedor do Novartis *Grid* estaria sujeito, caso estivesse operando nas condições de um nó numa grade P2P, para manter a sua infra-estrutura em condições de oferecer serviços à grade. Chamaremos esse custo de *Custo de Hardware*. No caso que estamos considerando, portanto, o custo de hardware seria igual a $80,27$ euros por CPU, por ano. À medida em que vai-se adicionando carga de trabalho à estrutura considerada, passa-se a incorrer em custos adicionais.

Os custos que são decorrentes da provisão dos serviços dependem de muitos fatores: a carga de CPU que cada serviço impõe aos equipamentos, a quantidade de operações a que o disco estará sujeito, as condições de operação dos componentes, mais os aspectos de segurança e os custos por incidentes. Chamaremos esses custos de *Custos de Software*.

Ainda tomando por referência os valores do Novartis *Grid*, estimaríamos um custo máximo de $113,00$ (eletricidade) mais $71,85$ (equipamento) completando um custo máximo de $184,85$ euros por CPU, por ano. Note que estamos desprezando todas as diferentes possíveis condições de operação, e estamos observando um conjunto de serviços homogêneo, já que o Novartis se destina unicamente à pesquisa em Farmacologia. Para grades mais heterogêneas esses aspectos precisariam ser considerados, e as análises considerariam os custos decorrentes da combinação das características do serviço com as condições de operação às quais os equipamentos estão submetidos.

## 3.4 Estimando custos de prover serviços numa grade

Considerando os apectos de custos de equipamento e de negócio que são relevantes para nosso estudo, podemos encontrar alguns valores de custo usando estimativas feitas com dados de um provedor de serviços do grid EGEE I [Duarte et al. 2007]. Com os dados considerados, nós obtivemos os seguintes valores: os custos totais de manutenção de todos os equipamentos da grade EGEE foram registrados na faixa entre $73.000$ a $1.600.000$ euros por ano, a depender das características dos equipamentos e do tipo de uso a que eles estão sujeitos. Como se considera tipicamente que os recursos são usados para a grade durante $1/3$ do tempo [Ian 2007], o custo de equipamento para operar a grade fica entre $24.333$ e $533.333$ euros/ano. Esses valores foram calculados considerando uma garantia padrão de $2$ anos. Da mesma forma, o custo máximo com energia fica em $1,6$ milhões de euros por ano, nas mesmas condições. Considerando a variação média de consumo entre CPU's ociosas e executando o `Seti@home`, temos um mínimo de $976.761$ euros. Somando os custos de equipamento e eletricidade temos um total que varia entre $1.373.000$ e $2.133.333$ euros.

Esse estudo foi feito considerando que esse provedor possui uma média de $11.700$ processadores. Isso dá, para o caso que estamos considerando, um custo aproximado entre $83,48$ e $182,34$ euros por ano, por processador. Fazendo as mesmas simplificações da seção 3.3, teremos que o custo de hardware desse provedor é igual a $83,48$ euros e o custo de software pode chegar a $98,85$. No trabalho onde foram apresentados esses valores não foi especificado que tipos de serviços estavam sendo oferecidos, e o estudo não leva em consideração variações decorrentes de diferentes condições de operação, nem os custos gerados

pela interrupção das operações em caso de incidentes, tampouco os custos com segurança.

O mesmo trabalho [Opitz et al. 2008] faz um estudo semelhante com a grade Novartis, que é mostrado na seção 3.3, considerando um total de $2.700$ CPUs. O estudo encontrou custos de manutenção e substituição de equipamentos na ordem de $21.000$ a $120.000$ euros/ano, e gastos com eletricidade na ordem de $211.915$ a $356.000$. Já discutimos que outros custos que foram considerados no estudo, como licensas de softwares, não se aplicam ao nosso caso, já que assumimos uma grade oportunista em que todos os softwares utilizados são gratuitos. Custos resultantes da interrupção das operações e com implementação de medidas de segurança não foram observados no estudo.

Ainda assim, os custos totais, ficariam entre $64,21$ e $147,88$ euros por ano, por processador, se considerarmos apenas os aspectos que são concernentes a uma grade oportunista e, para fazermos uma análise similar à que foi feita com a grade EGEE, um tempo médio de $8$ horas por dia de operação. Isso nos dá um custo de hardware igual a $64,21$ e um custo de software máximo de $83,67$ euros.

Evidentemente espera-se que os nós possuam, de modo geral, uma quantidade bem menor de processadores. Isso provavelmente determinará um custo anual mais alto por processador. Além disso, certamente haverão custos a considerar em decorrência de interrupções de operações em caso de incidentes, bem como custos adicionais com segurança. Esses custos são diretamente relacionados com o tipo de serviço que está sendo oferecido, postos portanto como custos de software. O mesmo se dá com custos de pessoal, já que horas-extras e sobreaviso dependem de legislação específica. Da mesma forma, com a variação de preços negociados pelos nós junto a diferentes fornecedores de energia elétrica, também os custos por $KW$ serão variados. Assim, é razoável supor que, para um nó típico, os custos, particularmente os custos de software, sejam superiores aos que nós obtivemos nesse levantamento.

## 3.5 Receita dos serviços

Assumimos que há uma receita associada a cada unidade de serviço que é consumida pelos nós, seja ela gerada localmente ou obtida da grade. Para isso, estabelecemos que os nós possuem um conjunto de serviços - seu favor típico - que são os que ele solicita da grade e

que o permitirão computar alguma tarefa de seu interesse.

Para determinar o valor que um serviço possui para um nó, faremos uma análise a partir do seu custo. É razoável supor que o custo que um nó tem para fazer sua infra-estrutura executar um processamento de seu interesse (o seu *workload*) é inferior à receita que ele recebe para instalar e manter essa infra-estrutura. Por esse motivo assumiremos que, quando executados localmente, o custo por uma unidade de qualquer dos serviços que compõem seu favor típico é inferior à receita proporcionada por esta mesma unidade de serviço. Para simplificar, definimos que os nós possuem um fator multiplicativo específico para cada nó e cada serviço ($\mu > 1$), tal que a receita resultante da execução de um serviço em um dado instante será $\mu$ vezes o seu custo.

## 3.6 Discussão

Neste trabalho, partimos da premissa de que os nós podem prover múltiplos serviços numa grade P2P, sendo que para isso eles precisam arcar com os custos relacionados. Acontece que os custos de cada serviço não são constantes, e ainda podem mudar quando considerados por nós diferentes. Isso ocorre porque a composição dos custo de prover um serviço é formada por fatores de origens diversas.

Cada um dos fatores gera custos de naturezas distintas, entre as quais podemos destacar: impacto sobre o MTTF dos equipamentos, suspensão de operação em decorrência de incidentes, variação do consumo de eletricidade, remuneração e demais aspectos da legislação trabalhista, e gastos com segurança.

Os fatores também decorrem de aspectos específicos da operação dos recursos de Tecnologia da Informação (TI) do nó. O nosso levantamento apontou os seguintes.

1. Aspectos ambientais, como a temperatura e a umidade dos locais onde os equipamentos estão instalados, impactam tanto nos custos com a manutenção dos equipamentos quanto na quantidade esperada de incidentes.

2. Aspectos tecnológicos, como o tipo e a marca dos equipamentos, as versões de sistemas operacionais e *drivers*, além da idade média dos equipamentos, também impactam nos custos de manutenção e de tratamento de incidentes, além de custos com

consumo de eletricidade.

3. Aspectos de legislação, que determinam quanto e como se gasta com pessoal, particularmente em caso de tratamento de incidentes (hora extra, sobreaviso, etc.), além de custos com taxas e impostos que incidem principalmente sobre o consumo de recursos externos como energia elétrica.

4. Segurança, porque permitir que nós remotos operem os recursos locais de TI acrescenta um risco e por isso são necessárias medida preventivas. Além disso, nesses casos a quantidade esperada de incidentes também sofre um impacto.

5. A carga de esforço necessária para as operações normais de TI a que os equipamentos são submetidos (sem contar o que será acrescentado para operar os serviços da grade), que na verdade serão refletidas nos três primeiros aspectos acima, além de impactar no consumo de insumos externos como eletricidade.

6. Os próprios serviços que são oferecidos à grade, que acrescentam uma carga extra de uso dos equipamentos, incrementando, provavelmente de maneira desigual já que dependem das características dos serviços que serão oferecidos, os custos decorrentes de cada um dos quatro primeiros aspectos listados, além de consumo de energia.

Usando dados operacionais [Opitz et al. 2008] [Schroeder and Gibson 2007] [Chou et al. 2001] e análises similares [Opitz et al. 2008], mostramos como se pode estimar os custos de prover serviços em um nó, bem como fizemos uma estimativa que considerou apenas os custos concernentes à operação numa grade P2P oportunista (que usa recursos ociosos).

Os valores que mostramos aqui são típicos de grandes instalações, que operam uma quantidade expressiva de recursos de TI (número de equipamentos na ordem de $10^3$ ou $10^4$), estando numa escala bem maior do que se espera encontrar nos nós que compartilham recursos em grades oportunistas, que provavelmente terão uma quantidade menor de recursos e, em função da escala, seus custos deverão ser superiores. Além disso, os fatores ligados à implementação de medidas de segurança, de legislação trabalhista e de tratamento de incidentes são fortemente dependentes das características de cada organização.

Entre os valores levantados, mostramos que os nós precisam arcar com uma parte deles por menor que seja a carga extra gerada pelos serviços oferecidos à grade. Definimos que

esse valor mínimo, decorrente da própria instalação de TI que o nó posssui é o *Custo de Hardware*.

Mostramos também que, à medida que os nós aceitam submeter carga de trabalho da grade à sua infra-estrutura, seus custos aumentam. Esse aumento depende da natureza dos serviços, bem como dos aspectos internos do próprio nó. Esse custo adicional gerado pela operação de serviços para a grade foi chamado de *Custo de Software*.

Os custos de Hardware e Software de cada nó só podem ser determinados usando informações específicas, e variam de um nó para outro. Não obstante, algumas considerações podem ser feitas a respeito dos valores que encontramos. Os nós, possuindo tipicamente uma infraestrutura menor do que as que foram consideradas nos trabalhos que fundamentaram nosso estudo, provavelmente terão custos unitários mais altos para mantê-la. Além disso, fatores como medidas de segurança, tratamento de incidentes (que geram interrupção de operação) e a própria legislação trabalhista são específicos de cada nó e não foram observados nesses estudos, embora também gerem impacto nos custos. Os valores de gastos com eletricidade também variam de acordo com a empresa fornecedora/concessionária e com as condições do nó, que pode obter descontos ou vantagens em função de seu perfil de consumo.

Por esse motivo é importante ressaltar que os custos de hardware de nós típicos provavelmente serão superiores aos que foram levantados aqui, já que o tamanho da plataforma de TI dos grandes provedores permite uma maior eficiência no uso dos recursos, tanto no gerenciamento dos equipamentos quanto nas condições de negociação de insumos como mão de obra e energia.

Da mesma forma, para os típicos nós que consideramos, os custos de software, que dependem das condições de operação da infra-estrutura, também superarão os que estimamos aqui, já que sobre esses custos impactam muitos dos aspectos que não pudemos observar.

Assumindo que os nós possuem um conjunto de serviços que são mais comumente solicitados, e de posse da estimativa de custos dos serviços, consideramos que o custo arcado por um nó para prover qualquer um desses serviços é menor do que a receita proporcionada a ele pelo mesmo serviço. Assim, definimos que, a qualquer instante, os nós conhecem um fator multiplicativo $\mu > 1$ que permite relacionar o custo e a receita proporcionada por uma unidade de um serviço.

Finalmente, é importante deixar claro que os valores que levantamos aqui são específicos

para disponibilizar os recursos ociosos numa grade, os custos de hardware, e decorrentes da concessão de recursos locais para a execução de serviços a terceiros, os custos de software. Para se determinar os custos totais de infra-estrutura de TI precisaríamos considerar também os custos decorrentes da operação normal dos recursos, atendendo às necessidades internas das organizações.

Com esse resultado, apesar de não conhecer os valores específicos de cada nó, já que dependem de elementos que só podem ser observados em casos específicos, podemos estabelecer uma premissa que seguirá nosso trabalho: os nós valoram os serviços diferentemente uns dos outros.

# Capítulo 4

# Definição do Problema e Modelo do Sistema

Neste capítulo nós descrevemos o problema da seleção de portfólio de serviços, de acordo com a abordagem que tomamos para lidar com ele. Em seguida, apresentamos um modelo matemático do sistema P2P com múltiplos serviços, que será usado em nossos experimentos.

## 4.1 Definição do Problema

Consideramos uma grade P2P em que cada nó consome e provê alguns serviços diferentes. Cada serviço possui um custo para ser oferecido, que depende de muitos fatores como recursos requeridos, instalação e manutenção de infra-estrutura, suporte técnico, etc. Assim, espera-se que os serviços possuam custos diferentes para nós diferentes. Os serviços também provêem receitas diferentes para clientes diferentes que os solicitam. Além disso, os custos e as receitas precisam ser cuidadosamente considerados de forma que a relação entre eles provejam uma lucratividade suficiente para os nós, de forma que eles mantenham interesse em continuar convivendo no sistema. Por esse motivo, os nós precisam selecionar um portfólio de serviços para prover, considerando uma relação vantajosa do ponto de vista custo/benefício, onde a receita obtida em reciprocidade à oferta desses serviços supere os custos para mantê-los disponíveis.

Considerando um intervalo de tempo suficientemente grande, cada nó possui um perfil de requisição representado pelo conjunto de serviços requeridos e as proporções entre eles,

chamado *favor típico*. Naturalmente, cada nó deve prover ao menos os serviços de seu favor típico, já que estes estão previamente disponíveis em suas instalações computacionais. Considerando que ainda há recursos disponíveis, o portfólio de serviços do nó pode ser incrementado ao se anexar mais serviços.

Nós consideramos aqui que os nós vão ser estimulados a doar serviços para outros nós através do uso de algum mecanismo de incentivo como a NoF [Andrade et al. 2004]. Assim, um grande portfólio de serviços vai dar mais espaço para que se estabeleça interações mutuamente lucrativas com outros nós. Infelizmente, um grande portfólio também acarreta um alto custo para prover os serviços.

Idealmente, deve-se escolher o portfólio de forma que o lucro que um nó consegue obter da grade seja maximizado. Naturalmente, isso só seria possível se o comportamento futuro de todos os nós fosse conhecido, o que não é o caso em grades P2P reais. Além disso, posto serem os nós agentes racionais, a mudança no portfólio de um nó pode desencadear mudanças nos portfólios dos demais nós, porque a seleção de serviços de um nó acaba tendo um impacto no lucro obtido pelos demais e, por esse motivo, provavelmente estes também mudarão sua estratégia.

Dada a impossibilidade de antecipar uma seleção ótima de portfólio, é desejável tratar o problema como uma otimização, em que os nós buscam alcançar uma seleção que se aproxime o máximo possível da melhor seleção que pode ser feita, a fim de maximizar o lucro.

Posto o problema que será objeto de nosso estudo, faz necessário apontar uma abordagem para lidar com ele. Neste trabalho tratamos deste problema a partir de uma modelagem formal, em que definimos matematicamente todos os aspectos que lhe são pertinentes.

## 4.2 Modelo do Sistema

Construímos um modelo matemático que nos permitiu lidar com os diversos aspectos do sistema que consideramos nesse estudo, permitindo uma abordagem analítica, bem como uma abordagem por simulação.

### 4.2.1 Múltiplos Serviços

Assumimos que cada serviço tem um custo para ser provido, e esse custo depende de muitas características como o tipo do serviço, a capacidade de investimento do nó, o consumo de insumos como pessoal e eletricidade, a taxa de falhas e a decorrente necessidade de substituição de equipamentos, bem como o tempo de instalação/configuração. Além disso, todo o hardware instalado por um nó também tem um custo de manutenção que precisa ser considerado. Parte desse custo é devido á disponiblização de equipamentos para execução de serviços da grade. Por esse motivo, um mesmo serviço provavelmente terá custos e receitas diferentes em nós diferentes [Mowbray et al. 2006]. Essa heterogeneidade de custos e receitas dos serviços pode fazer com que certos serviços, em certas condições, tenham uma lucratividade baixa, ou possivelmente negativa. Assim, faz-se necessário escolher um subconjunto de serviços para prover, sendo que este subconjunto precisa gerar uma lucratividade a mais alta possível [Coêlho et al. 2008].

### 4.2.2 O modelo matemático

Supomos $n$ serviços diferentes, o conjunto $S = \{s_1, s_2, ...s_n\}$ representa todos os serviços que podem ser oferecidos pelos nós. Assumimos que os nós podem prover qualquer um dos serviços, cada um com um custo associado. O tempo foi discretizado em *turnos* de forma que qualquer intervalo de tempo é composto de $T$ turnos.

Consideramos um total de $N$ nós usando algum mecanismo de reciprocidade para compartilhar múltiplos serviços. Qualquer nó $p$ precisa se limitar a um orçamento $\mathcal{B}_p$ para estabelecer e manter um conjunto de serviços a ser oferecido à grade durante um dado intervalo de tempo. Após cada turno de tempo os nós podem trocar o conjunto de serviços oferecidos. Assim, a cada turno $t$ o conjunto de serviços provido por um nó $p$ é dado por:

$$\mathcal{S}_p^t = \langle s_{p,1}^t, s_{p,2}^t, ..., s_{p,n}^t \rangle$$

onde $s_{p,j}^t \in \{0,1\} \ \forall \ p \in \{1, ..., N\}, \ \ j \in \{1, ..., n\}$ e $t \in \{1, ..., T\}$.

Em um dado instante alguns nós podem estar com recursos ociosos, disponibilizando-os para a grade. Outros podem estar consumindo os recursos da grade. Os nós que desempenham o primeiro desses papéis são ditos no estado *doando* e os que desempenham o segundo

papel são considerados no estado *recebendo*.

Para simplificar, assumimos que, quando no estado *recebendo*, o nó está com toda a sua capacidade local sendo usada para computar seus próprios serviços, e além disso está tentando receber capacidade adicional da grade.

### Custos

Como mostramos no capítulo 3, os valores dos serviços são diferentes para cada nó e, por isso, o vetor $\mathcal{C}_p = \langle c_{p,1}, c_{p,2}, ..., c_{p,n} \rangle$ representa os custos de cada serviço para o nó $p$. Assim, cada $c_{p,j}$ é o custo que o nó $p$ terá para prover o serviço $j$. Neste trabalho consideramos que o custo do serviço não mudará ao longo do tempo. Essa simplificação é razoavelmente realista, já que não se esperam alterações grandes nem rápidas dos fatores que compõem os custos dos serviços. Além disso, posto que o nó usa os custos dos serviços como insumos para a tomada de decisão a respeito da seleção de seu portfólio, eventuais mudanças nos custos acarretariam apenas mudanças na seleção dos portfólios, mas não nos mecanismos usados pelos nós para selecionar serviços. Dessa forma, o nó $p$ terá um custo de serviços, por turno $t$, definido por $\sum_{j=1}^{n} s_{p,j}^{t} \cdot c_{p,j}$

Para manutenção de sua capacidade computacional, os nós também se sujeitam a custos de hardware. O custo de hardware depende da capacidade de processamento instalada. Para o nó $p$, $q_p$ representa sua capacidade local de processamento, que será oferecida à grade quando estiver ociosa. Consideraremos essa capacidade em número de máquinas e assumiremos, por simplificação, que todas as máquinas são similares em capacidade de processamento e armazenamento de dados.

O custo por turno de hardware que será associado a cada unidade de processamento é dado por $h_p$ e assim o custo total que o nó $p$ terá para manter seu hardware disponível para a grade é dado por $q_p \cdot h_p$.

A soma do custo por turno de hardware mais o dos serviços representa o custo final do nó $p$, na forma

$$C_p^t = q_p \cdot h_p + \sum_{j=1}^{n} s_{p,j}^{t} \cdot c_{p,j} \leq \mathcal{B}_p^t,$$

O orçamento que resta ao nó $p$ no turno $t$ é dado pela diferença entre seu orçamento

total ($\mathcal{B}_p$) e o total de gastos que ele já teve até o turno $t-1$. Isso permite definir o valor do orçamento restante $\mathcal{B}_p^t$ como sendo o valor de seu orçamento total subtraído dos custos acumulados no decorrer do tempo, da seguinte forma:

$$\mathcal{B}_p^t = \mathcal{B}_p - \sum_{k=0}^{t-1} C_p^k$$

Neste trabalho consideraremos que o nó dispõe, a cada turno, de um orçamento constante. Este orçamento será igual à $T$-ésima parte do orçamento geral. Assim, a cada turno o nó $p$ dispõe de um orçamento igual a $\mathcal{B}_p^t = \mathcal{B}_p/T$

**Favor Típico**

Assumindo que serão normalmente domínios administrativos autônomos, os nós possuem objetivos e atividades de negócio específicas. Como resultado, as aplicações que lhes dão suporte também devem ser específicas. Definimos a *demanda típica* de um nó como sendo a quantidade média de unidades de computação e a proporção entre os serviços requisitados.

Por exemplo, se um nó, por um certo intervalo de tempo, solicita $40$ máquinas que tenham o serviço $s_1$ e $10$ máquinas que tenham o serviço $s_2$, a sua demanda típica é de $50$ máquinas, numa proporção $(80\%, 20\%)$ entre os serviços $s_1$ e $s_2$.

Esta proporção é o seu *Favor Típico*. Neste trabalho, assumiremos que o favor típico de cada nó é constante ao longo do tempo.

Definimos o favor típico do nó $p$ como um vetor de proporções (possivelmente iguais a zero em muitos casos) entre os serviços, de forma que a soma final dessas proporções seja igual a 1. Assim, o favor típico do nó $p$ é dado por $F_p = \langle f_{p,1}, f_{p,2}, ..., f_{p,n} \rangle$, onde $0 \leq f_{p,j} \leq 1 \ \forall \ p, j$ onde $p \in \{1, ..., N\}$, $j \in \{1, ..., n\}$, e $\sum_{j=1}^{n} f_{p,j} = 1$.

Conforme descrito na definição do problema, neste trabalho assumimos que os nós sempre oferecem seu Favor Típico. Note que, de acordo com nosso modelo, se um serviço $s$ está no favor típico do nó $p$ então $f_{p,s} \neq 0$, e assim $\lceil f_{p,s} \rceil = 1$. Dessa forma, para garantir que os serviços que compõem o favor típico sempre sejam oferecidos, há que se definir a restrição: se $\lceil f_{p,i} \rceil = 1$ então $s_{p,i}^t = 1$.

### Receita

Consideramos que, numa operação normal de organizações que utilizam TI, o custo de se manter os *data centers* instalados é necessariamente menor do que os resultados provenientes do uso destes. Por isso, é razoável supor que, no modelo, um nó $p$ tenha uma receita associada a cada um dos serviços que ele consome, e essa receita é tal que supere seus custos[1]. Para modelar isso nós consideramos um fator de lucro ($\mu_{p,s}$) que representa o quanto é lucrativo o uso do serviço $s$ pelo nó $p$. Para efeitos práticos, naturalmente, o valor de $\mu_{p,s}$ precisa ser maior do que $1$. Isso garante que o serviço é lucrativo quando provido pelo próprio nó. Esse aspecto é importante para refletir o fato de que a infra-estrutura local é planejada para prover os serviços de interesse do próprio nó, de forma que, quando disponibilizados usando a infra-estrutura local, tais serviços sempre gerarão uma receita superior aos custos.

A fim de se encontrar o total de receita em computação que o nó $p$ recebeu da grade é necessário saber a quantidade de processamento que foi recebida por cada serviço. Consideramos que, em um turno $t$, a quantidade total de computação que um nó recebe é dada por $\mathcal{R}_p^t = \langle r_{p,1}^t, r_{p,2}^t, ..., r_{p,n}^t \rangle$ onde $r_{p,s}^t$ é a quantidade de computação do serviço $s$ recebida pelo nó $p$ no instante $t$.

Mas o uso de sua própria capacidade local para processar serviços também gera receita para os nós. Por isso, é necessário também se levar em conta a quantidade local de processamento que o nó $p$ possui. Essa receita local é dada pela sua capacidade de hardware ($q_p$) e o seu favor típico, que definirá a proporção que cada serviço receberá de sua capacidade instalada. Considerando o exemplo da Seção 4.2.2, se o nó possui um total de $20$ máquinas, estas deverão ser divididas na proporção de $16$ máquinas para o serviço $s_1$ e $4$ máquinas para o serviço $s_2$. Isso é obtido porque as $20$ máquinas deverão ser multiplicadas pela proporção dos serviços que é de $0,8$ e $0,2$ respectivamente.

Para se calcular a receita total que o nó $p$ recebe no turno $t$ é necessário se multiplicar o fator de lucro pelos custos dos serviços, considerando as unidades de computação que foram recebidas para cada serviço; tanto as que foram obtidas dos recursos locais quanto as que foram fornecidas pela grade. A quantidade de computação local do serviço $s$ é dada pela multiplicação de $q_p$ por $f_{p,s}$ que é a proporção do serviço $s$ usada pelo nó $p$, enquanto a quantidade de computação do serviço $s$ que foi recebida da grade é dada por $r_{p,s}^t$. Note que,

[1] Com efeito, não teria sentido manter sua infra-estrutura se tal não ocorresse.

ao contrário da capacidade local, em que a quantidade de computação é constante, já que a infraestrutura de cada nó não varia, a computação recebida da grade é variável, podendo ser mais alta ou mais baixa a depender das condições encontradas em cada turno $t$.

Assim, a quantidade total (local e da grade) de unidades de computação configuradas com o serviço $s$ recebida pelo nó $p$ no turno $t$ é dada por $q_p \cdot f_{p,s} + r^t_{p,s}$. Quando multiplicamos esse valor pelo custo de cada unidade de procesamento ($h_p$) temos o custo a que o nó $p$ estaria sujeito caso todas as unidades fossem providas por si mesmo. É o custo do serviço *do ponto de vista do nó* $p$.

Da mesma forma, para cada serviço que ele recebe, caso esteja em seu favor típico, haveria um custo de software dado por $c_{p,s}$.

Os dois custos, de Hardware e de Software, precisam ser multiplicados pelo fator de lucro ($\mu$) a fim de termos a receita. Dessa forma, podemos definir a quantidade total de receita que o nó $p$ recebe no turno $t$ da seguinte forma:

$$U^t_p = \sum_{j=1}^{n} \mu_{p,j} \cdot [(q_p \cdot f_{p,j} + r^t_{p,j}) \cdot h_p + \lceil f_{p,s} \rceil \cdot c_{p,s}]$$

A receita, entretanto, é dependente do estado do nó. Quando o nó está em estado *doando*, e possui capacidade ociosa, ele não tem nenhuma unidade computando serviços para ele. Logo, sua receita é igual a zero e a equação acima define a receita do nó para os instantes em que ele esteja no estado *recebendo*.

### *Lucro*

No decorrer do tempo os nós vão acumulando suas receitas e seus custos. O lucro que o nó $p$ vai ter, considerando esse modelo, é definido em função da diferença entre esses dois valores. Sendo $U^t_p$ a receita total que o nó $i$ recebeu no turno $t$ e $C^t_p$ o seu custo no mesmo turno, e considerando a variação do tempo, o lucro final do nó $p$ é dado por

$$P_p = \sum_{t=1}^{T} U^t_p - \sum_{t=1}^{T} C^t_p,$$

sendo que o valor de $U^t_p$ é igual a zero quando o nó está doando serviços.

### Revisitando o Problema

Sob a luz deste modelo podemos agora definir formalmente o problema que trataremos neste trabalho. Considerando que cada nó atuará de maneira individualista, buscando maximizar a sua própria lucratividade no ambiente, temos que o objetivo será, para cada nó $p$, maximizar seu lucro $P_p$.

Todavia, esta maximização precisa atender a duas restrições: o custo total precisa ser inferior ao orçamento disponível, e os nós sempre estarão oferecendo todos os serviços de seu favor típico.

Dessa forma, o problema pode ser descrito como segue.

$$\text{Maximize } \sum_{t=1}^{T} U_p^t - \sum_{t=1}^{T} C_p^t$$

restrito a

$$\left( \sum_{t=1}^{T} C_p^t \right) \leq \mathcal{B}_p$$

bem como

$$\text{se } \lceil f_{p,i} \rceil = 1 \text{ então } s_{p,i}^t = 1,$$

onde $1 \leq t \leq T$.

# Capítulo 5

# O impacto da escolha

Neste capítulo são mostradas diversas avaliações para demonstrar o quanto as diferentes escolhas de portfólio de serviços impactam nos resultados obtidos pelos nós numa comunidade.

Inicialmente mostramos, através de experimentos estatísticos, que as variações de lucratividade obtidas quando tomamos ambientes com nós interagindo e trocando serviços são muito grandes. A segunda parte do capítulo dedica-se à avaliação de como as diferentes características do ambiente afetam o impacto da escolha. O capítulo termina com a conclusão de que, mesmo sob condições variadas, o portfólio de serviços impacta no lucro que os nós obtêm da grade.

Para fazermos as avaliações, desenvolvemos um simulador que implementa o modelo definido no capítulo 4, considerando um ambiente com uma certa quantidade de nós compartilhando serviços diferentes, utilizando a NoF como mecanismo de reciprocidade.

## 5.1 Contabilidade de Favores

De acordo com a NoF aplicada a múltiplos serviços [Mowbray et al. 2006], é necessário se definir um mecanismo de *Contabilidade de Favores* para que os nós usem informações de interações passadas a fim de decidir prioridades no escalonamento de seus recursos ociosos. O simulador implementa a NoF, de forma que cada nó mantém um saldo a respeito das interações passadas com cada um dos demais nós com quem ele tenha interagido. Este saldo indica a quantidade de favores que cada nó se considera devedor dos demais. Quando um nó $p$ doa serviços para o nó $k$, o saldo de $p$ mantido por $k$ é incrementado enquanto o saldo

análogo (de $k$ mantido por $p$) é decrementado. Os valores de incremento ou decremento dos saldos são definidos internamente pelos nós em função de seus próprios custos.

Supondo um exemplo em que $k$ esteja doando para $p$. A fim de calcular o valor de incremento do saldo de $k$ junto a si, o nó $p$ precisa considerar a quantidade de computação que ele recebeu de $k$ e multiplicar isso pelos seus próprios custos. O saldo análogo é decrementado da mesma forma, apenas considerando os custos de $k$. Como não pode haver saldo negativo, uma verificação adicional é necessária quando os nós estão doando.

Definindo $Bal_p^t(k)$ como sendo o saldo do nó $k$ mantido por $p$ no turno $t$ e $r_{p,s}^t(k)$ como sendo a quantidade de computação do serviço $s$ que foi doada entre os nós $p$ e $k$ no instante $t$, podemos definir a função de atualização do saldo como segue:

$$Bal_p^t(k) = \begin{cases} max\{0; Bal_p^{t-1}(k) - \sum_{j=1}^{n} r_{k,j}^t(i) \cdot (c_{p,j} + h_p)\}, \text{ se } p \text{ está doando para } k; \\ Bal_p^{t-1}(k) + \sum_{j=1}^{n} r_{p,j}^t(k) \cdot (c_{p,j} + h_p), \text{ se } p \text{ está recebendo de } k; \\ Bal_p^{t-1}(k), \text{ se } p \text{ não está nem doando nem está recebendo de } k; \end{cases}$$

## 5.2 Avaliação dos diferentes portfolios

Para avaliar o impacto de diferentes portfólios, foi necessário lidar com o grande número de possíveis seleções de serviços. A quantidade de combinações é dada por $\sum_{k=0}^{\nu} \binom{\nu}{k}$ onde $\nu$ é a quantidade máxima de serviços que se pode oferecer. Por esse motivo desenvolvemos um simulador que faz uma abordagem probabilística. Neste simulador, os nós fazem um grande número de seleções aleatórias, para encontrar alguma que seja muito boa e outra que seja muito ruim, embora não haja chance concreta de encontrar a melhor ou a pior possível quando simulamos uma quantidade significativa de serviços. Note que o objetivo é verificar se o impacto é grande, ao invés de encontrar a melhor ou a pior seleção.

Consideramos uma grade P2P com nós homogêneos, com média contenção (os nós operam com $50\%$ de chance de estarem doando, e demanda de processamento igual a duas vezes a própria capacidade, portanto eles solicitam à grade uma quantidade de recursos igual à sua própria capacidade de processamento). Decidimos usar cenários com custo de hardware igual a $20\%$ do orçamento, sendo o custo de cada serviço definido aleatoriamente

entre $10\%$ e $20\%$ do orçamento, além de definirmos um fator de lucro de $4$ ($\mu = 4$). Com essas configurações, todos os nós obtêm resultados positivos, o que permite uma comparação proporcional entre os resultados, embora não haja efeitos sobre o impacto das escolhas.

Depois de simulações preliminares, vimos que quando usamos qualquer valor entre $1.000$ e $10.000$ seleções aleatórias, a diferença entre a melhor e a pior seleção encontrada não aumentava significativamente. Por esse motivo, nós decidimos usar um total de $1.000$ seleções.

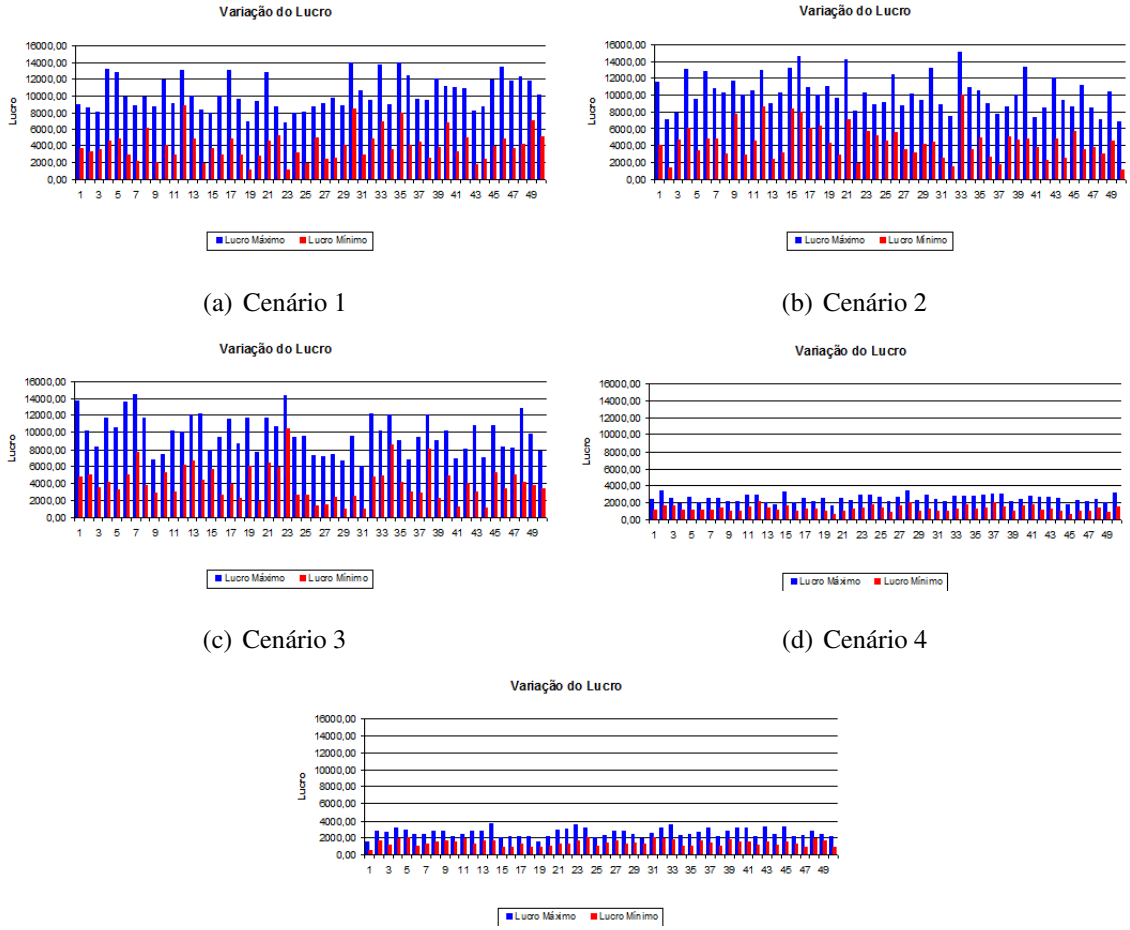
Geramos aleatoriamente $5$ cenários, usando as características descritas acima. Cada cenário é composto de um conjunto específico de: requisições dos nós (*workload*), favores típicos de cada um dos nós, custos dos serviços e orçamento por turno, também para cada nó.

Em cada cenário os nós fazem uma seleção aleatória, que é usada por um intervalo de tempo ($1.000$ turnos na nossa simulação) e registra os resultados. Depois disso, o processo é repetido $1.000$ vezes, perfazendo um total de $1.000$ seleções, cada uma delas experimentada por um tempo que dura $1.000$ turnos.

Ao fim do experimento, o maior e o menor lucro de cada nó foi registrado. A Figura 5.1 mostra os lucros máximo e mínimo obtidos pelos nós nos $5$ cenários estudados. Note que, em alguns cenários (os dois últimos), embora a variação proporcional (a proporção entre o melhor e o pior resultado) seja muito elevada, o maior lucro absoluto não passou de $4.000$, enquanto em outros cenários pode-se chegar ou ultrapassar a faixa de $14.000$ unidades. Isso é uma forte evidência de que os aspectos do ambiente também impactam na lucratividade. Estes aspectos e seus impactos serão estudados detalhadamente na Seção 5.3.

Na Figura 5.2 podemos ver a média das variações encontradas entre os nós nos diferentes cenários. Note que nenhum nó obteve, em média, uma variação de lucro inferior a $50\%$, (embora em alguns experimentos tenham aparecidos valores bem menores) e a média das variações ficou em $152\%$.

Em seguida, pudemos medir qual foi, entre todos os diferentes cenários, a maior e a menor variação de lucro obtido por cada nó. A Figura 5.3 mostra as maiores e as menores variações encontradas por cada um dos nós nos cenários simulados. Em todas as simulações, a maior variação encontrada foi de $629\%$ enquanto a menor variação encontrada foi de $29\%$.




(a) Cenário 1

(b) Cenário 2

(c) Cenário 3

(d) Cenário 4

(e) Cenário 5

Figura 5.1: Variação do lucro dos nós nos diferentes cenários estudados

### 5.2.1 Variação da lucratividade

Os resultados permitiram-nos ordenar os nós de acordo com a variação percentual da lucratividade. Na Figura 5.4, a linha entre as duas áreas cinzas representa a variação percentual média encontrada nos cenários. A figura deve ser lida como "$x\%$ dos nós obtiveram uma variação aproximada da lucratividade de, no mínimo, $y\%$". Por exemplo, nesses resultados, nós podemos notar que $90\%$ dos nós conseguiram uma variação de pelo menos $50\%$ aproximadamente, e alguns nós (menos de $2\%$ deles) obtiveram uma variação de $400\%$. As áreas cinza representam o erro máximo estatístico, considerando um intervalo de confiança de $95\%$.

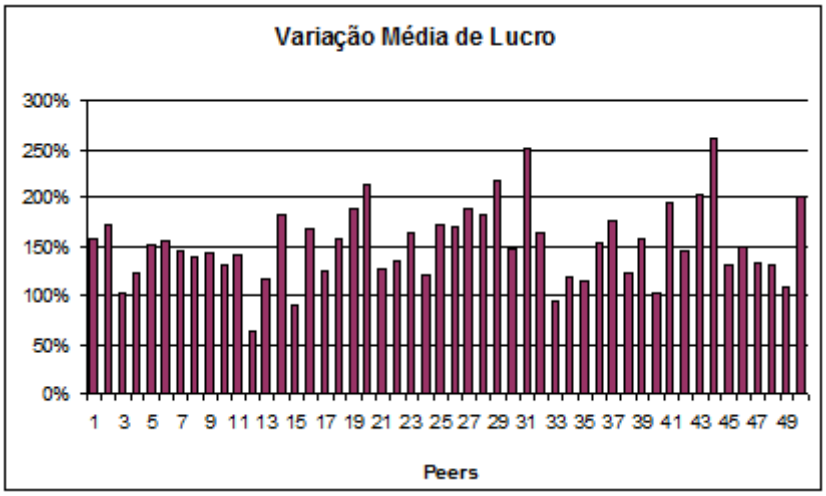



Figura 5.2: Variação média do lucro dos nós

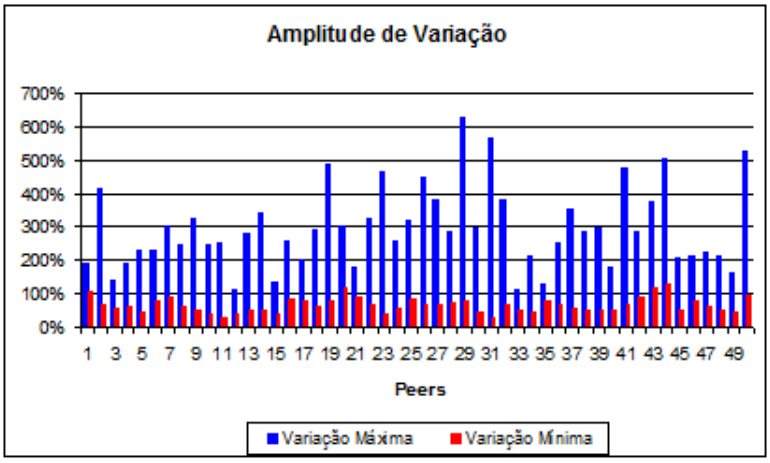



Figura 5.3: Comparação entre a maior e a menor variação de lucro dos nós

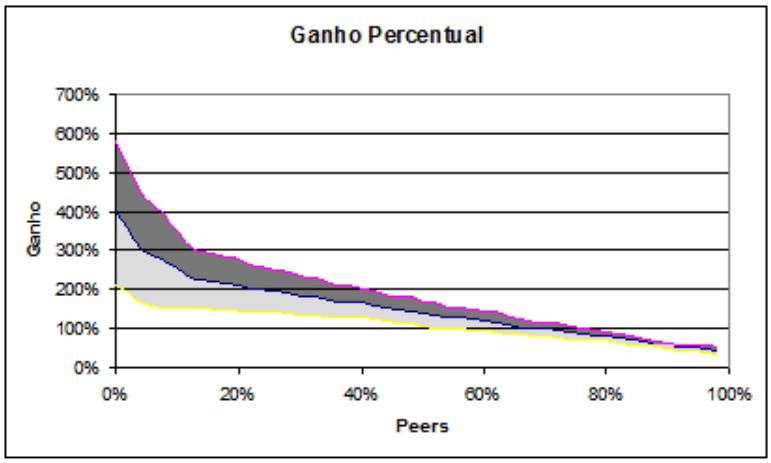



Figura 5.4: Variação do lucro X Percentual de Nós: percentual de ganho médio obtido por diferentes frações da população de nós

## 5.3 Avaliando o impacto sob diferentes condições ambientais

Os resultados da Seção 5.2 mostram que a seleção de serviços tem um forte impacto no lucro que os nós recebem da grade. Entretanto, espera-se que os demais aspectos do próprio ambiente também impactem na lucratividade dos nós, como a quantidade de serviços disponíveis para serem oferecidos, o número máximo de serviços que podem ser oferecidos pelos nós - caso de orçamentos mais ou menos generosos - e a própria contenção (relação entre a quantidade de demanda e de oferta de recursos, que determina a competitividade entre os nós) do sistema também influenciam nas possibilidades que os nós possuem de melhorar ou piorar seus resultados a partir da seleção de portfólio de serviços.

Desenvolvemos experimentos simulados em que os nós dispunham de uma quantidade máxima de serviços para oferecer, e os mantinham fixos durante uma sessão. Definimos uma sessão como sendo uma quantidade de tempo grande o suficiente para que os grupos de nós mutuamente lucrativos estivessem consolidados, segundo a ExtNoF. Selecionando diferentes conjuntos a cada sessão, e executando uma grande variedade de sessões (mas não cobrindo todas as possíveis combinações de seleções, pois a quantidade é proibitiva), medimos o melhor e o pior resultado de cada nó, considerando a relação percentual entre o que ele solicita e o que ele efetivamente recebe. Para evitar que aspectos específicos dos nós influenciassem

no resultado, estabelecemos que os serviços teriam um custo constante ($= 1$), e que a receita por unidade de serviço consumido seria igual ao seu próprio custo ($\mu = 1$). Os experimentos realizados sob essas configurações permitem medir a quantidade de computação recebida, e o quanto ela é variável em função dos diferentes aspectos considerados.

Para comparar a quantidade de computação recebida entre os nós, consideramos um resultado relativo, em confrontação com o máximo que pode ser obtido por cada nó. Esse valor é dado pela quantidade total de recursos que ele solicita. Com esses parâmetros fixados, experimentamos a variação de lucratividade proporcionada por diferentes portfólios de serviços enquanto variamos os aspectos ambientais.

### 5.3.1 Variando o tamanho do universo de serviços

A relação entre a quantidade de serviços e a quantidade de nós foi medida inicialmente. Para fazê-lo, consideramos diferentes números de serviços de forma que a relação $\frac{n}{N}$ (número de serviços / número de peers) variasse desde um cenário em que há duas vezes mais nós que serviços até uma relação de um nó para cada três serviços.

| **Parâmetros** |
|---|
| 50 nós |
| Total de serviços variou: 25, 50, 75, 100, 125 e 150 serviços |
| 5 serviços oferecidos por cada nó |
| 50% de chance de cada nó estar doando em qualquer turno |
| Favor típico com 1 serviço |

Tabela 5.1: Parâmetros usados para avaliar o impacto da variação no tamanho do universo de serviços

A Tabela 5.1 mostra a configuração de parâmetros de ambiente que consideramos nestes experimentos.

A Figura 5.5 mostra os melhores e os piores resultados médios relativos obtidos pelos nós nas diferentes circunstâncias. O melhor resultado dos nós ficou acima dos $90\%$, indicando que em média os nós receberam $9$ de cada $10$ serviços solicitados, independente da quantidade de serviços. Já o pior resultado mostrou-se mais sensível à relação entre as quantidades de serviços e de nós no ambiente. Quando há poucos serviços e muitos nós, mesmo

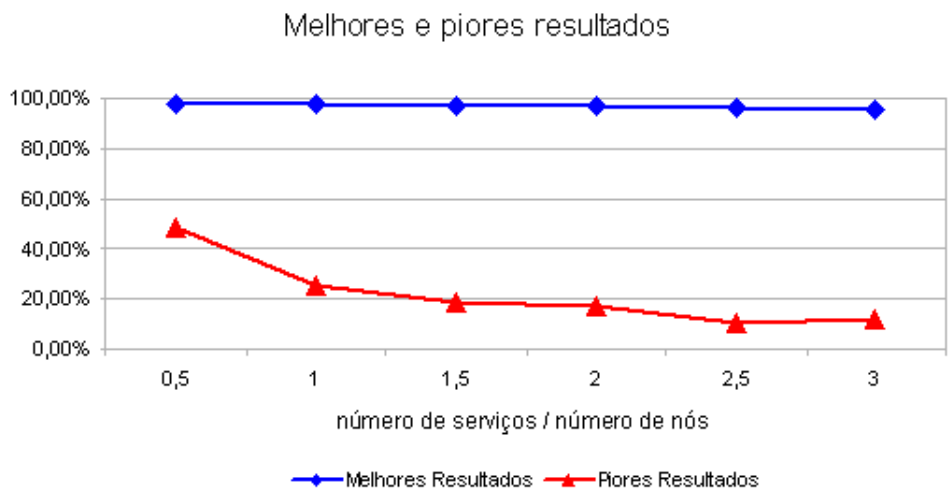



Figura 5.5: Melhores e piores resultados em função da relação entre número de nós e de serviços

as piores seleções conseguem uma lucratividade razoável, indo de $49\%$ quando há $2$ nós para cada serviço, e chegando a $25\%$ quando essa relação é de $1-1$, caindo em seguida à medida em que há mais serviços disponíveis.

Esse fenômeno é explicado porque quando há muitos nós e poucos serviços, a probabilidade de um nó selecionar aleatoriamente algum serviço que acaba sendo consumido por outro que seja recíproco no futuro é mais alta. À medida em que a quantidade de serviços vai crescendo, essa chance diminui e, por conta da reciprocidade do ambiente, os nós recebem menos recursos.

### 5.3.2 Variando a quantidade de serviços oferecidos

Em seguida estudamos de que forma a quantidade de serviços que cada nó pode oferecer (que é função da generosidade do orçamento e da quantidade de recursos disponíveis) impacta na variação de lucratividade obtida com diferentes portfólios de serviços.

Para realizar esses experimentos, nós fixamos em $1$ a relação entre o tamanho do universo de serviços e o total de nós da comunidade ($\frac{n}{N} = 1$), estabelecendo uma grade com $50$ nós e $50$ serviços. Variamos a quantidade de serviços oferecidos por cada nó com valores de $1$, $2$, $3$, $5$, $10$, $15$, $20$ e $25$, enquanto os demais parâmetros foram mantidos constantes (grade com contenção média, sessões de $1000$ turnos, favor típico composto de $1$ serviço e $1000$ seleções

diferentes).

| **Parâmetros** |
|---|
| 50 nós |
| 300 serviços no total |
| Total serviços oferecidos por cada nó variou: 1, 2, 3, 5, 10, 15, 20 e 25 serviços |
| 50% de chance de cada nó estar doando em qualquer turno |
| Favor típico com 1 serviço |

Tabela 5.2: Parâmetros usados para avaliar o impacto da variação na quantidade de serviços oferecidos

A tabela 5.2 mostra os parãmetros que usamos nos expeimentos que mediram o impacto da escolha quando variamos a quantidade de serviços oferecidos por cada nó.

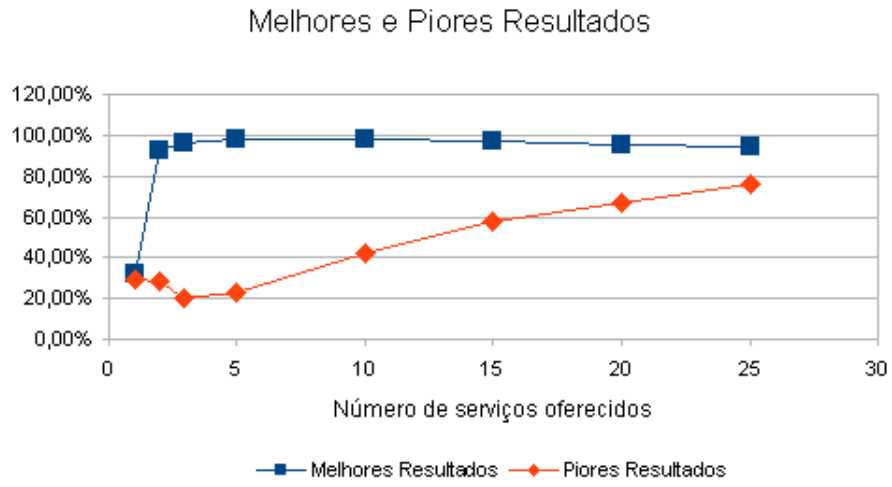



Figura 5.6: Melhores e piores resultados em função da quantidade de serviços oferecidos

A Figura 5.6 mostra os melhores e os piores resultados médios obtidos pelos nós enquanto a quantidade de serviços varia. É importante observar que quando os nós oferecem apenas o seu favor típico (que nesses experimentos é formado por apenas 1 serviço), praticamente não há variação porque apenas os nós que eventualmente consumam o mesmo serviço poderiam promover trocas entre si, sendo que as variações de portfólio são impossíveis, já que nesses casos a única variação possível é em função de recursos obtidos por estarem ociosos. A figura mostra que o melhor resultado dos nós se aproximou de 100% quando a

quantidade de serviços oferecidos chega a $5$ (ou $10\%$ do total de serviços) e, assim, torna-se possível variar o portfólio. Todavia, o pior resultado ficou em pouco mais de $20\%$ nesses casos. Esses "piores resultados" decaem cada vez mais quando há poucos serviços, ($n \leq 3$) porque as possíveis combinações de seleções são muito limitadas, mas vão melhorando à medida em que a quantidade de serviços oferecidos aumenta, chegando a praticamente inexistir diferença quando a quantidade de serviços oferecidos chega a $25$ (ou $50\%$ do total).

Da mesma forma, a figura mostra que a variação é menor com uma quantidade pequena de serviços oferecidos e cresce à medida em que essa quantidade aumenta. Quando a quantidade de serviços se torna muito alta essa variação volta a diminuir.

Isso ocorre porque com poucos serviços oferecidos a quantidade de possíveis seleções diferentes é muito pequena (ou apenas uma, caso o nó possa oferecer somente o serviço de seu favor típico), de forma que não há como o nó variar seus resultados significativamente. Por outro lado, quando a quantidade de serviços oferecidos se torna muito grande (aproximando-se de uma fração significativa total de serviços existentes no universo), os nós encontram uma situação análoga: a quantidade de serviços "não selecionados" diminui, fazendo com que a quantidade de possíveis diferentes seleções seja menor e, por esse motivo, não seja possível variar mais significativamente seus resultados.

Com efeito, uma vez que que a quantidade de possíveis variações é dada por $\frac{n!}{(n-x)! \cdot x!}$, sendo $x$ o número de serviços oferecidos, quando $x$ se aproxima de $0$, o valor se aproxima de $1$. Por outro lado, quando $x$ se aproxima de $n$, o valor também se aproxima de $1$. Assim, quando se oferecem muitos ou poucos serviços, a quantidade de variações se torna muito pequena.

Um outro aspecto importante que se pode ver na Figura 5.6 é que há um ligeiro decremento dos melhores resultados obtidos pelos nós quando a quantidade de serviços oferecidos cresce muito. Isso ocorre porque, quando a quantidade de serviços oferecidos pelos nós aumenta, eles passam a concorrer entre si já que a quantidade de nós que oferecem um mesmo serviço aumenta. Essa concorrência acaba tendo um impacto no quanto cada um deles consegue receber individualmente da grade.

### 5.3.3 Variando a contenção

Finalmente, para entender como a escassez ou a fartura de recursos afeta o sistema, executamos um experimento similar em que variamos a contenção do sistema como um todo. Para variarmos a contenção, usamos a probabilidade de os nós estarem doando em cada instante. Quando a probabilidade é muito baixa, o sistema está operando em alta contenção porque há um número muito alto de solicitações contra poucas ofertas. À medida que a probabilidade de estar doando aumenta a contenção diminui. Estabelecemos que uma contenção é considerada média quando os nós atuam com $50\%$ de chance de estar doando.

| **Parâmetros** |
|---|
| 50 nós |
| 300 serviços no total |
| 5 serviços oferecidos por cada nó |
| Chance de cada nó estar doando em qualquer turno variou: 10%, 30%, 50%, 70% e 90% |
| Favor típico com 1 serviço |

Tabela 5.3: Parâmetros usados para avaliar o impacto da contenção do sistema

A Tabela 5.3 mostra os parâmetros que usamos para realizar os experimentos que mediram o impacto da contenção na variação de lucratividade obtida pelos nós.

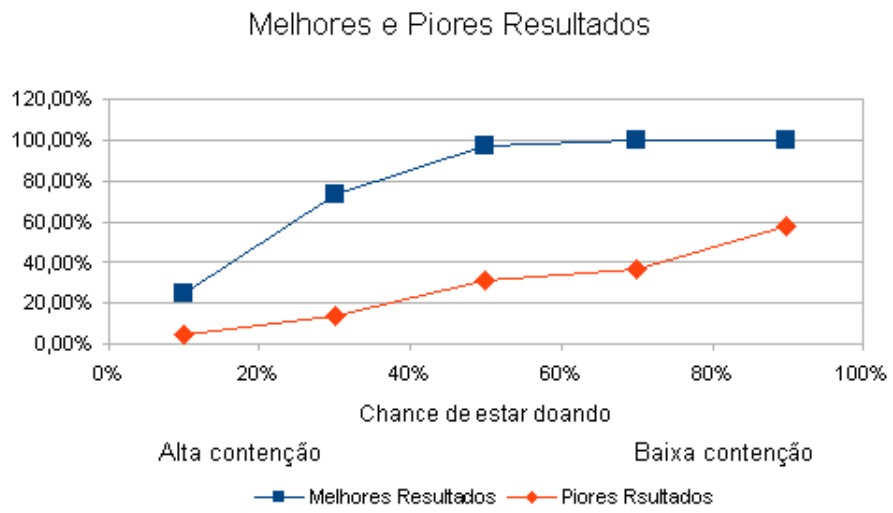



Figura 5.7: Melhores e piores resultados em função da contenção da grade

A Figura 5.7 mostra como a contenção afeta a diferença entre os melhores e os piores resultados encontrados pelos nós em função dos diferentes portfólios selecionados. Podemos ver que quando o sistema opera em alta contenção (o número médio de doadores fica em torno de $10\%$ a $20\%$) a amplitude da variação de resultados é consideravelmente menor do que quando a contenção é média ($40\%$ a $60\%$ de doadores). Além disso, o ambiente com alta contenção diminui a lucratividade, pois mesmo os melhores resultados são muito baixos (inferiores a $60\%$). Isso ocorre porque, quando há alta contenção, a quantidade de recursos ofertados é pequena para o número de consumidores. Dessa forma, a quantidade máxima de recursos que os nós podem receber também é pequena em relação à quantidade de recursos solicitados. Esse fenômeno limita a própria amplitude de variação, já que não há como os nós alcançarem resultados mais expressivos em função da própria exigüidade de recursos.

Quando a contenção se torna baixa os melhores resultados se aproximam do ótimo ($100\%$), embora se perceba uma ligeira diminuição na diferença entre os melhores e os piores resultados. Esses fenômenos se explicam porque, quando há baixa contenção - caso em que há muitos recursos oferecidos e pouca demanda por eles - há pouca competição por recursos e os nós acabam recebendo recursos ociosos de outros mesmo com seleções ruins de portfólio. Note que, de acordo com a NoF, um nó só define prioridades quando há competição pelos recursos. Dessa forma, a reciprocidade do sistema se torna mais relevante quando o escalonador de recursos escolhe, entre os vários nós que estão competindo, premiar os que foram mais generosos no passado. Assim, havendo pouca competição por recursos, há também pouca oportunidade de usufruir da reciprocidade do sistema, diminuindo a importância de seleções que maximizem parcerias lucrativas.

A Figura 5.8 ilustra esse aspecto sob a ótica da relação entre a quantidade total de recursos solicitados e recebidos. Tomando a média dos melhores resultados obtidos, vemos que somente quando a contenção é muito alta é que se torna impossível receber a totalidade dos recursos demandados. Em seguida, à medida em que a contenção diminui, a quantidade de recursos solicitados também diminui, permitindo obtê-los com mais facilidade.

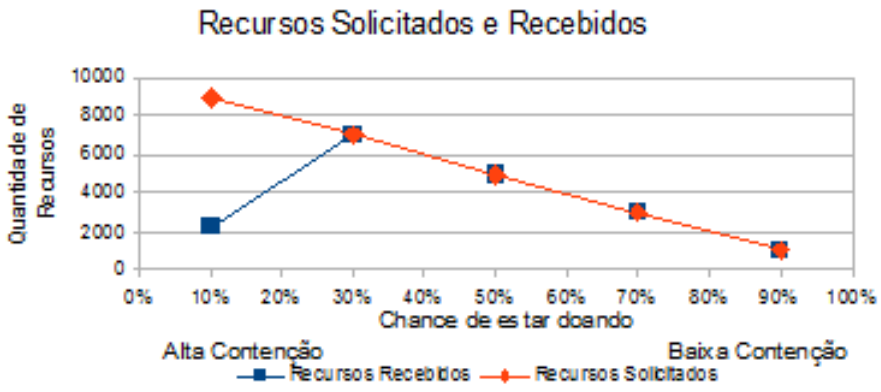



Figura 5.8: Quantidade de recursos solicitados e recebidos em função da contenção da grade

## 5.4 Discussão

Os experimentos mostraram que o lucro recebido pelos nós pode variar significativamente em função do portfólio de serviços que eles selecionam para oferecer. Em um ambiente com muitos serviços disponíveis essa variação pode alcançar valores muito diferentes, em função do conjunto de serviços que é selecionado pelos nós.

Nas simulações da Seção 5.2 considerou-se um orçamento igual para todos os nós. Além disso, os nós sempre solicitavam uma quantidade de máquinas exatamente igual à sua capacidade instalada. Caso sejam outros os valores desses parâmetros os resultados variarão, podendo haver casos de nós com lucro negativo ou lucros mais altos, mas sempre havendo uma variação do lucro em função do conjunto de serviços selecionados.

Foram gerados $5$ cenários diferentes. Para cada um dos cenários houve $1.000$ seleções diferentes de serviços a fim de se medir a variação do lucro em função do portfólio de serviços. Experimentos com até $10.000$ seleções não resultaram em mudanças significativas dos resultados (salvo em casos isolados). Isso se deve ao fato de que, com o orçamento fixo, a quantidade de possíveis seleções é restrita a uma fração da quantidade total de combinações de serviços. Para orçamentos maiores provavelmente serão necessárias mais seleções.

Observamos também que os cenários trouxeram variações entre si, com alguns cenários gerando maior variação nos lucros dos nós. Encontramos médias de variações que vão de $95\%$ a $198\%$ entre os cenários considerados. Esse fenômeno é esperado, já que em tese podem haver cenários em que absolutamente não há variação de lucro, como no caso em que todos os nós são idênticos, com custos iguais para todos os serviços, e que selecionam

exatamente os mesmos serviços para o seu favor típico. Tais cenários, porém, além de irreais são extremamente improváveis de serem gerados aleatoriamente nesses experimentos simulados.

Além disso, mostramos que aspectos como a contenção, o tamanho do universo de serviços, a quantidade de nós e a capacidade orçamentária (que determina a quantidade de serviços oferecidos) também influenciam na variação de lucro que os nós podem obter em função do portfólio de serviços observados. De fato, ambientes com altíssima contenção, ou comunidades cujos nós ofereçam todos - ou quase todos - os serviços disponíveis, ou ainda casos em que o universo de serviços seja mínimo, realmente não permitem que os nós explorem diferentes portfólios de serviços oferecidos a fim de maximizar a lucratividade.

Em todas as demais condições estudadas o conjunto de serviços selecionados impacta na lucratividade recebida, outorgando aos nós a possibilidade de usar estratégias para selecionar serviços que sejam mais interessantes do ponto de vista da lucratividade.

# Capítulo 6

# Heurísticas de Seleção de Serviço em comunidades P2P

Nos sistemas P2P que consideramos neste trabalho, todos os nós têm como objetivo encontrar uma seleção de portfólio que maximize seu lucro. Infelizmente, a cada seleção feita por um nó $p$ se segue um cenário específico, que é composto pela demanda de outros nós, mais seus respectivos portfólios. O número de possíveis combinações de portfólios feitos por cada um dos $N$ nós no sistema é dada por $\sum_{k=0}^{x} \binom{x}{k}$, sendo $x$ o número de serviços que podem ser selecionados. Isto posto, a combinação total de possíveis estados após cada seleção feita por cada um dos nós é dada por $\left[\sum_{k=0}^{x} \binom{x}{k}\right]^{N}$.

Além disso, devido ao caráter estocástico do sistema, é impossível saber a ordem em que cada solicitação vai ser atendida pelos nós provedores. Há também o fato de que os serviços selecionados, e possívelmente doados, pelo nó $p$ acabam gerando mudanças nas prioridades dos outros nós, de acordo com o esquema de reciprocidade. Considerando que toda solicitação é, em algum instante, recebida por cada um dos nós, e sendo $N \cdot (1 - p^{don})$ o número de nós que estão consumindo, então temos um total de $N \cdot (1 - p^{don})!$ possíveis filas de solicitações que chegam a cada nó que está provendo serviços.

Sabendo que os nós precisam usar critérios para selecionar os serviços a fim de maximizar sua lucratividade, e tendo em vista o número exponencial de possíveis estados do sistema depois de cada seleção, mais o número fatorial de possíveis combinações de solicitações atendidas em cada instante, mais o fato de que o comportamento futuro dos outros

nós é desconhecido, podemos afirmar que a seleção do melhor conjunto possível de serviços é uma busca vã, dada a complexidade computacional do processo.

Isto posto, resta aos nós implementar métodos de seleção que se baseiem em heurísticas. Estas soluções abrem mão da melhor escolha possível em troca de escolhas que sejam satisfatoriamente boas ao tempo em que podem ser executadas com complexidade polinomial.

## 6.1 Avaliação das heurísticas

Definimos dois conjuntos de métodos algorítmicos (heurísticas) que fazem a seleção dos serviços para os nós, e medimos a lucratividade alcançada e a receita relativa destas heurísticas sob diferentes condições. O primeiro conjunto são as heurísticas baseadas no *modelo do sistema*, e o segundo são heurísticas que se baseiam em informações do ambiente.

No primeiro grupo estão heurísticas que se utilizam de informações a respeito do sistema que consideramos: o fato de ser um sistema baseado em reciprocidade e os aspectos intrínsecos dos nós, como o custo dos serviços e o orçamento disponível. No segundo grupo temos heurísticas que trabalham colhendo informações do ambiente, a fim de usá-las para tomar decisões que permitam melhorar a lucratividade dos nós.

Mostramos os resultados encontrados para cada heurística, e no final fazemos considerações sobre o desempenho de cada uma delas.

A avaliação das heurísticas foi feita usando uma simulação semelhante à que usamos para aferir o impacto das escolhas, com a diferença de que neste caso o seletor de serviços é implementado com um algoritmo específico.

Consideramos um ambiente com $50$ nós ($N = 50$), e $300$ serviços ($n = 300$). Definimos que a capacidade local de cada nó é igual a $10$ unidades de computação $q_p = 10, \forall p$ onde $1 \leq p \leq N$, e o custo de manutenção de cada unidade é igual a $0,1$. Considerando que o custo é composto de múltiplos diferentes fatores que se superpõem, e de acordo com o teorema do limite central [Dudley 1978], temos que o custo de software deverá seguir uma distribuição normal; assim, estabelecemos que esta distribuição seria dada por $c_{p,s} = X \sim N(0,75; 0,25)$. O fator de lucro seria igual a $4,0$ ($\mu = 4$). É importante ressaltar que resultados experimentais nos permitem afirmar que a variação do fator de lucro não altera o comportamento das heurísticas, apenas tornam os resultados mais ou menos lu-

crativos. Finalmente, estabelecemos que o favor típico seria composto de dois serviços, um deles com uma proporção que variava no intervalo $[0,1; 0,2; ...; 0,9]$ e o outro na proporção complementar.

A simulação se desenvolveu em turnos, conforme descrito no modelo matemático, sendo que em cada turno os nós podem estar no estado *doando* ou *recebendo*. Ficou estabelecido que o sistema teria uma contenção média: a quantidade esperada de nós doadores e consumidores deveria ser aproximadamente igual. Para isso, a probabilidade de cada nó estar no estado *doando* é igual a $50\%$ ($p^{don} = 0,5$).

O orçamento disponível por turno para cada nó ($\mathcal{B}_p^t$) variou entre $3$ e $25$. Para cada valor de orçamento nós geramos $100$ cenários diferentes. Um cenário é definido como sendo o perfil de todos os nós no sistema: os seus favores típicos, os custos de provisão para todos os serviços que podem ser oferecidos, além do *workload* geral, com informações sobre quais os nós estão consumindo ou doando serviços ao longo do tempo.

Executamos simulações de cada cenário durante um total de $1.000$ turnos ($T = 1.000$). Essas simulações com $100$ cenários e $1.000$ turnos foram suficientes para gerar resultados com um erro relativo sempre inferior a $2\%$ com um nível de confiança igual a $95\%$ em todos os nossos experimentos. A Tabela 6.1 mostra os valores que usamos para os parâmetros nos vários cenários simulados.

Tabela 6.1: Sumário de parâmetros do sistema nos cenários simulados

| **Parâmetro** | **Valor** |
|---|---|
| $N$ | $50$ |
| $n$ | $300$ |
| $q_p, \forall p, 1 \leq p \leq N$ | $10$ |
| $h_p, \forall p, 1 \leq p \leq N$ | $0,1$ |
| $c_{p,s}, \forall p, s, 1 \leq p \leq N, 1 \leq s \leq n$ | $N(0,75; 0,25)$ |
| $\mu_{p,s}, \forall p, s, 1 \leq p \leq N, 1 \leq s \leq n$ | $4$ |
| $T$ | $1.000$ |
| $p^{don}$ | $0,5$ |
| $\mathcal{B}_p^t, \forall p, t, 1 \leq p \leq N, 1 \leq t \leq T$ | $\{3, 4, 5, 10, 15, 20, 25\}$ |

Definimos uma heurística papalva, chamada de *Random*, que será usada como patamar

mínimo de validação das demais. Esta heurística implementa um nó que não considera nenhum aspecto na seleção de serviços para serem oferecidos, e o faz de forma absolutamente aleatória, limitado ao orçamento disponível.

Nesta avaliação nós consideramos que *Random* é, em média, a pior seleção possível já que esta heurística superaria apenas aquelas que intencionalmente busquem seleções ruins que, em casos práticos, jamais seriam implementadas por um nó.

## 6.2 Heurísticas baseadas no Modelo

Neste trabalho nós definimos duas heurísticas "básicas", baseadas no modelo do sistema, que foram inspiradas nas soluções de dois problemas diferentes: o Problema da Mochila e o Problema da Seleção de Carteira de Investimento. Estas duas heurísticas usam, como insumo para a tomada de decisão a respeito de quais serviços devem ser selecionados ou não, informações disponíveis para os nós, e que são definidas no próprio modelo do sistema: os custos dos serviços, o orçamento disponível, o histórico de interações passadas de um nó com os demais.

### 6.2.1 Heurística *Cost-based*

Considerando que um serviço pode ser oferecido ou não, o problema da Mochila $0/1$ é bastante similar ao problema da seleção de portfólio, já que naquele problema apenas pode-se incluir $0$ ou $1$ exemplar de cada item na mochila. Dessa forma, podemos mapear o peso dos itens no custo dos serviços, o valor dos itens na receita dos serviços e a capacidade da mochila no orçamento do nó. Assim, inspirada em abordagens pelo método guloso para o problema da Mochila [Diubin and Korbut 1999] bem como para o problema da mochila $0/1$ [Sarkar et al. 1992] [Calvin and Leung 2003], desenvolvemos a heurística *Cost-based*[1].

O método guloso toma os itens a serem incluídos de acordo com a *densidade*, que é definida como sendo o quociente entre o valor e o peso do item. Note que o valor do serviço é uma informação que não está definida, de forma que a heurística *Cost* é uma implementação adaptada do método guloso para o problema da mochila.

[1]Uma discussão mais aprofundada sobre o problema da Mochila pode ser encontrada na Seção 2.3.

Efetivamente a heurística *Cost-based* busca maximizar a quantidade de serviços no portfólio. Para isso, considerando a limitação de orçamento, busca-se incluir os serviços segundo uma ordem crescente de custos. O objetivo é tirar proveito do fato de que, quanto maior for a quantidade de serviços oferecidos, maior será a probabilidade de encontrar algum consumidor.

Geramos um conjunto aleatório de cenários para cada valor de orçamento disponível para os nós no ambiente, num totasl de $100$ cenários por valor. Em cada cenário os nós implementavam a heurística *Cost-based* e conviveram durante $1.000$ turnos, após os quais medimos seu desempenho. Com os mesmos cenários, substituída apenas o método de seleção de serviços, medimos também o desempenho da heurística *Random*. Os resultados finais de lucro podem ser vistos na Figura 6.1.

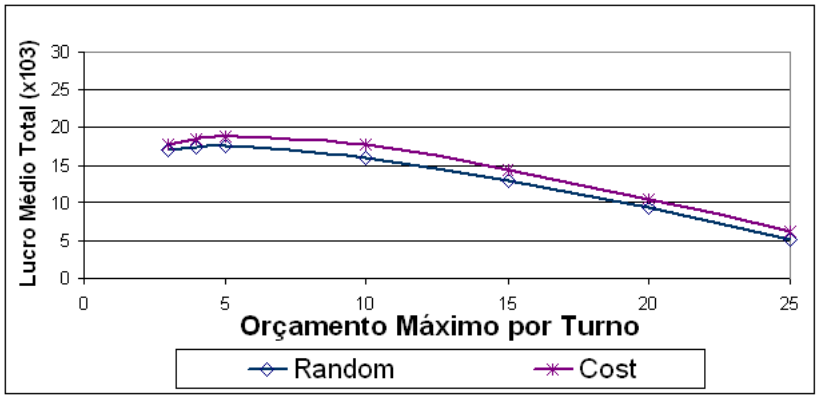



Figura 6.1: Cost-Based e Random: lucro (erro $\leq 2\%$ com I.C. de $95\%$)

Note que o desempenho médio da heurística *Random* ficou muito pouco aquém do que foi alcançado pela heurística *Cost* em praticamente todos os cenários. Como o custo total a que as heurísticas se oneraram foi aproximadamente o mesmo (veja Figura 6.2), então a pequena diferença se explica na receita que foi recebida, como podemos ver na Figura 6.3.

Ocorre que o universo de serviços é relativamente grande, com $300$ possíveis serviços diferentes. Com isso, a vantagem pretendida pela heurística *Cost-based* se revela um tanto menor, já que a quantidade de serviços que ela oferece é efetivamente superior à quantidade que é oferecida pela heurística *Random*, mas essa diferença acaba sendo pouco significativa em relação ao total de serviços disponíveis. Naturalmente, espera-se que a heurística *Cost* tenha um desempenho mais significativo quando o universo de serviços for menor.

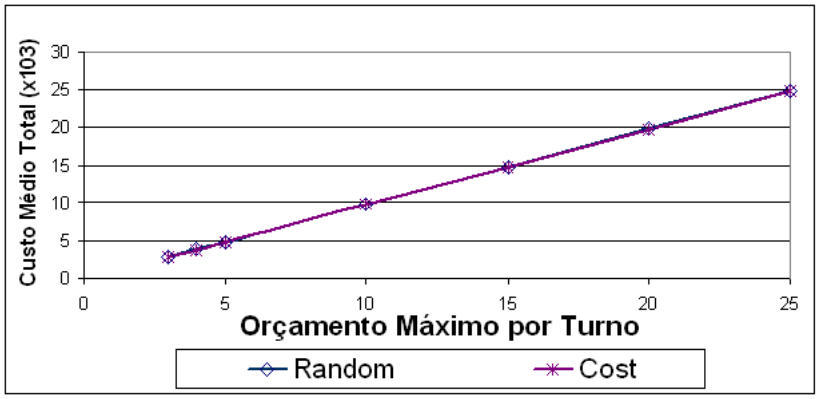



Figura 6.2: Cost-Based e Random: custo (erro $\leq 2\%$ com I.C. de $95\%$)

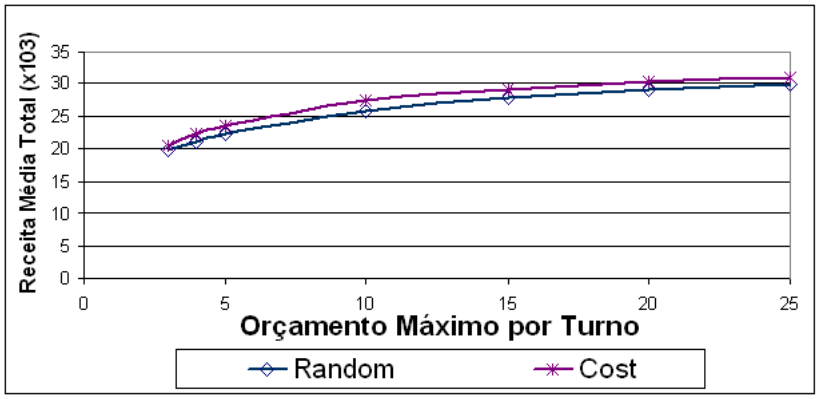



Figura 6.3: Cost-Based e Random: receita (erro $\leq 2\%$ com I.C. de $95\%$)

### 6.2.2 Heurística *Reciprocation-based*

Outro problema que também guarda importantes similaridades com a Seleção de Portfólio de Serviços é o Problema da seleção de Carteira de Investimento, onde o agente precisa decidir que ativos comporão uma carteira de investimentos sem possuir conhecimento sobre a lucratividade que eles proporcionarão no futuro, assim como os nós não conhecem a lucratividade futura de nenhum serviço.

Uma variação deste problema, conhecida como *Seleção de Carteira com Ordem Esperada de Lucratividade* [Xia et al. 2000] é particularmente similar ao problema da seleção de serviços, já que nesse problema o agente conhece a ordem esperada de lucratividade dos ativos[2].

Num ambiente P2P baseado em reciprocidade é possível inferir uma ordem esperada de

[2]O problema da Seleção de Carteira de Investimentos, é contemplado na Seção 2.4.

receita gerada pelo consumo de serviços ao se considerar informações passadas sobre os nós que os consumiram. Os serviços solicitados pelos nós que foram mais recíprocos são, provavelmente, os que geram mais receita. Assim, a heurística *Reciprocation-based* é inspirada na seleção de carteiras com ordem de rentabilidade esperada, ao considerar que, apesar de desconhecida, há uma ordem esperada da receita dos serviços. Dessa forma, seleciona-se os serviços tentando maximizar as interações lucrativas, ofertando prioritariamente os serviços que são consumidos pelos nós que se mostram mais recíprocos ao longo do tempo.

Como a heur´sitica *Cost-based* superou a heurística *Random*, mostramos uma comparação entre o desempenho das heurísticas *Cost-based* e *Reciprocation-based* pode ser vista na Figura 6.4. A heurística *Reciprocation-based* tem resultados consistentemente melhores, já que promove mais interações lucrativas, o que aumenta sua receita (veja na Figura 6.6, a despeito de os custos das duas heurísticas evoluírem de maneira similar, como se vê na Figura 6.5.

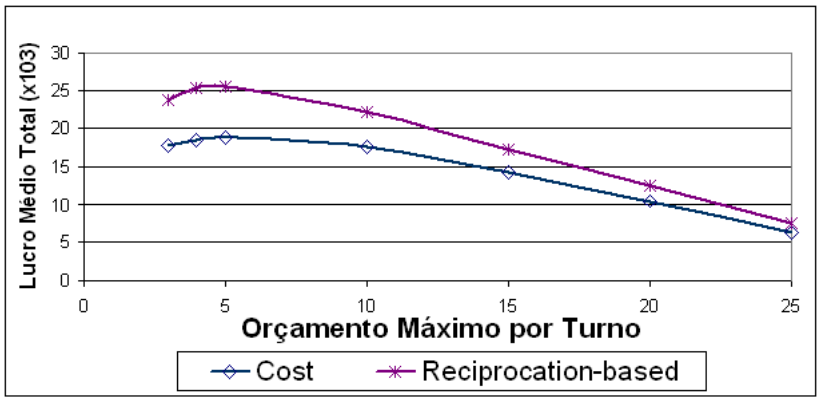



Figura 6.4: Reciprocation-Based e Cost-Based: lucro (erro $\leq 2\%$ com I.C. de $95\%$)

É importante observar que a receita cresce de maneira logarítmica, já que quando se aumenta o orçamento disponível para os nós do sistema também se aumenta a quantidade de serviços oferecido por cada nó e, por conseguinte, espera-se que mais nós ofereçam um mesmo serviço ao mesmo tempo. Isso gera competição, forçando a receita a ser rateada entre os nós competidores, impedindo que esta cresça indefinidamente.

Contudo, tanto a heurística *Cost-based* quanto a heurística *Reciprocation-based* usam o orçamento disponível até a exaustão. Com isso, há um ponto (nas nossas simulações quando o orçamento é aproximadamente igual a $10$) a partir do qual o crescimento do orçamento, e

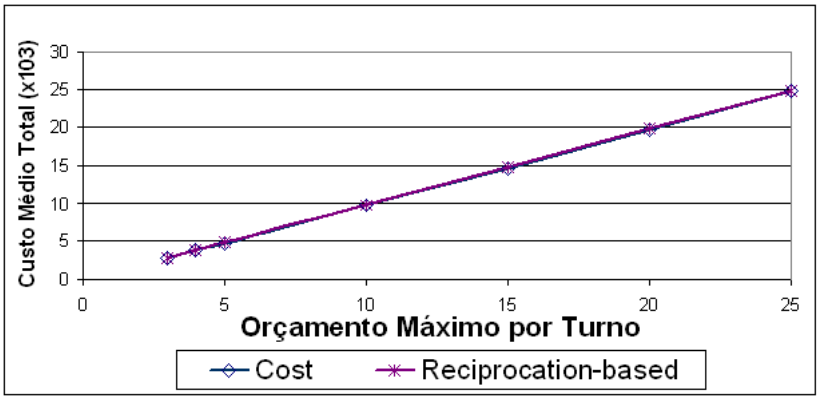



Figura 6.5: Reciprocation-Based e Cost-Based: custo (erro $\leq 2\%$ com I.C. de $95\%$)

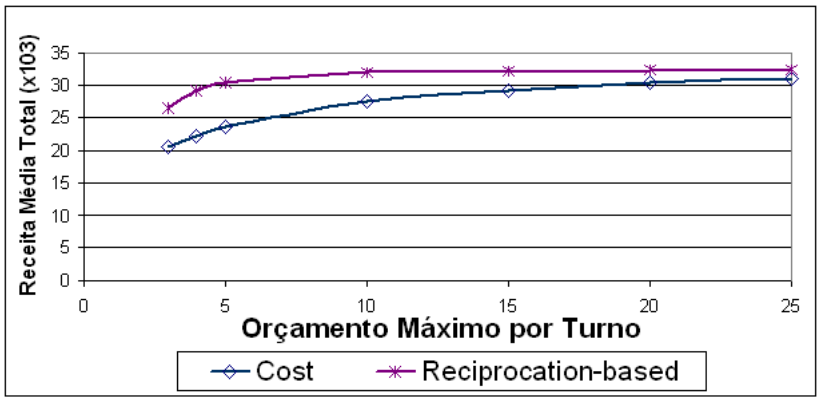



Figura 6.6: Reciprocation-Based e Cost-Based: receita (erro $\leq 2\%$ com I.C. de $95\%$)

consequentemente do custo, não produz um aumento de igual tamanho na receita, levando o lucro para níveis cada vez menores.

Torna-se evidente, portanto, que ambas as heurísticas não promovem um uso muito eficiente da relação custo/benefício em todos os cenários. Com efeito, nos casos em que a receita não cresce na mesma proporção que o custo, a lucratividade está diminuindo e o lucro se torna progressivamente menor até que eventualmente seja negativo. Isso leva à proposição de heurísticas que não utilizem necessariamente todo o orçamento disponível.

## 6.3 Heurísticas de restrição orçamentária

Transcendendo os limites do modelo do sistema, ao debruçarmo-nos sobre o problema de gerenciamento de custo/benefício, desenvolvemos duas heurísticas que tentam restringir a

quantidade de orçamento que os nós efetivamente usam em cada turno, usando uma abordagem por subida de colina [Coêlho and Brasileiro 2010].

### 6.3.1 Heurística *RBR*

Inicialmente desenvolvemos a heurística *Restricted Budget Reciprocation* ($RBR$), que opera da mesma forma que a heurística *Reciprocation-based*, com a diferença de que não usa necessariamente todo o orçamento disponível para prover serviços. A implementação da heurística $RBR$ define que, a cada $k$ turnos, o nó encontra um *ponto de decisão* que pode disparar uma mudança no limite de orçamento utilizado. A heurística escolhe entre aumentar ou diminuir o limite de orçamento utilizado, dependendo de sua ação anterior: se a lucratividade é crescente, a última decisão é considerada acertada e, por esta razão, mantida. Caso contrário, ela é invertida.

É importante observar que a lucratividade é a variação do lucro com o tempo, que indica se a quantidade de lucro acumulada por unidade de tempo está progressivamente maior ou menor. Desta forma, neste trabalho nós a consideramos a lucratividade como sendo a derivada da função de lucro, tomada no ponto de decisão.

Inicialmente, o limite é estabelecido como sendo o orçamento máximo por turno (i.e. $B_p^t$) e a estratégia inicial é aumentar o limite. A cada ponto de decisão subsequente, o nó testa se a lucratividade está crescendo ou diminuindo. Se está crescendo, a estratégia é mantida e, se possível, o limite é aumentado ou diminuído, de acordo com a estratégia que está sendo empregada. Quando a lucratividade diminui, o nó inverte sua estratégia; todavia, caso a diminuição do limite gere uma situação em que não haverá orçamento disponível, então o limite não é mudado. Da mesma forma, o limite não pode superar o orçamento máximo definido por turno.

Para se decidir o novo limite de orçamento que será utilizado, após a decisão de aumentá-lo ou diminuí-lo, um valor, chamado *passo de variação* ($\sigma$) é somado ou subtraído do limite até então utilizado. Note que no caso de a subtração do orçamento utilizado gerar um valor inferior a $0$ nenhuma alteração será feita. Com isso, quando o orçamento disponível for menor do que o passo de variação ($B_p^t \leq \sigma$), a heurística $RBR$ opera exatamente como a heurística *Reciprocation-based*.

Comparamos o desempenho das heurísticas $RBR$ e *Reciprocation-based*, sendo que,

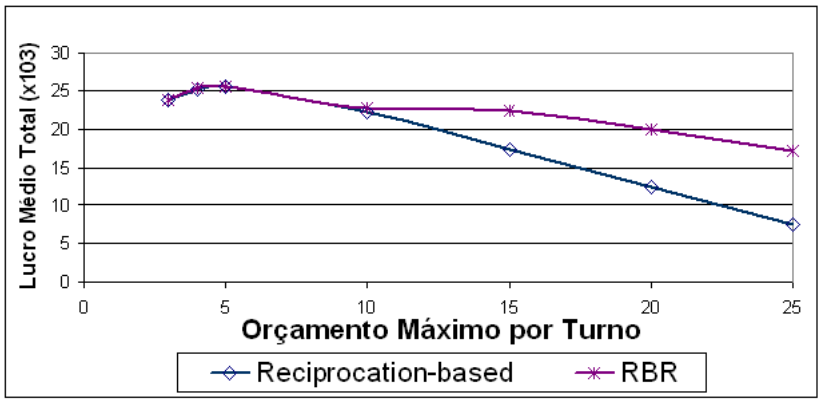



Figura 6.7: $RBR$ e Reciprocation-Based: lucro (erro $\leq 2\%$ com I.C. de $95\%$)

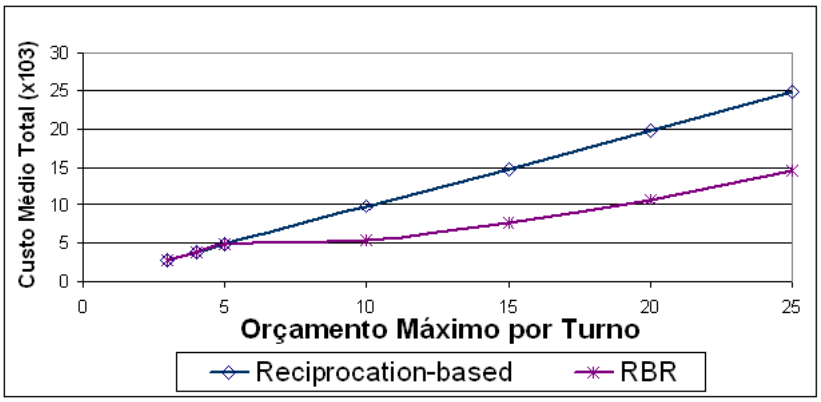



Figura 6.8: $RBR$ e Reciprocation-Based: custo (erro $\leq 2\%$ com I.C. de $95\%$)

depois de alguns experimentos preliminares testando valores diferentes, decidimos usar o parâmetro $\sigma = 5$ em todas as simulações, posto que este foi o valor com o qual a heurística obteve os melhores resultados.

Conforme se vê na Figura 6.7, a heurística $RBR$ consegue obter lucro superior à heurística *Reciprocation-based* quando os orçamentos são mais altos ($B_p^t \geq \sigma = 5$). É importante observar que, do ponto de vista da receita, a heurística *Reciprocation-based* supera a heurística $RBR$, conforme se vê na Figura 6.9, já que, não havendo restrição orçamentária, também não há restrição na quantidade de parcerias promovidas no caso da heurística *Reciprocation-based*.

Ainda assim a performance da heurística $RBR$ acaba sendo superior, em função do gerenciamento do custo, conforme se vê na Figura 6.8. Com efeito, a quantidade de receita que a heurística *Reciprocation-based* recebe a mais acaba sendo inferior à quantidade

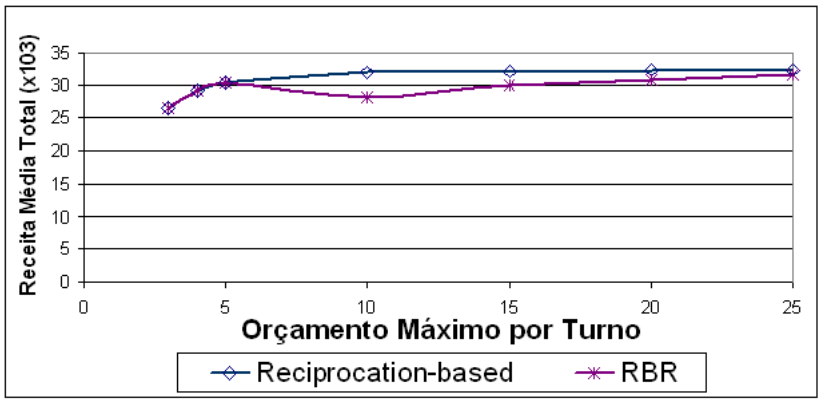



Figura 6.9: $RBR$ e Reciprocation-Based: receita (erro $\leq 2\%$ com I.C. de $95\%$)

de custo a que ela se submete, o que gera uma lucratividade inferior.

### 6.3.2 Máximos locais e Reinício aleatório: heurística *RBR*$^3$

Sabe-se que esse tipo de abordagem de subida de colina sujeita o agente ao problema de circunscrever seu espaço de busca a áreas que contenham valores locais ótimos. Nessas situações, o agente encontra um ponto que, embora não seja a melhor solução geral para o problema, possui a característica de que, a partir dele, qualquer que seja a sua decisão, esta o levará a pontos relativamente próximos do atual, sendo que em qualquer deles sua função objetivo acaba decrescendo. Isso o mantém preso àquela solução, referida como *ponto de ótimo* (ou *máximo*, ou *mínimo*) *local*.

Classicamente, a abordagem para escapar desse fenômeno de ótimo local em algoritmos de subida de colina é chamada de *Subida de Colina com Reinício Aleatório* [Russell and Norvig 2003]. Esta abordagem determina que, com uma probabilidade $\epsilon$, o nó vai parar de seguir a estratégia de subida de colina e vai testar um outro ponto aleatório no espaço de soluções. Se a função objetivo calculada neste novo ponto superar a que era calculada até então, o algoritmo continua normalmente sua execução a partir dali. Caso contrário, o processo é revertido[3].

A heurística *Restricted Budget with Random Restart Reciprocation-based* ($RBR^3$) é a implementação da heurística $RBR$ com o reinício aleatório. Os nós que implementam a

[3]Há variações dessa implementação, com a geração de múltiplos pontos aleatórios ou algoritmos que testam não só um ponto como sua vizinhança.

heurística $RBR^3$ executam, a cada ponto de decisão, com uma probabilidade $\epsilon$, a geração de um novo orçamento aleatório, uniformemente distribuído dentro do intervalo $[0, B_p^t]$. Este orçamento é verificado após um intervalo de $k$ turnos e sua lucratividade é aferida. Caso o nó tenha conseguido melhorar sua lucratividade neste intervalo o orçamento será mantido. Em caso contrário o processo é desfeito e o orçamento utilizado volta a ser o mesmo que era antes do reinício aleatório.

Assim como a heurística $RBR$, a heurística $RBR^3$ atua da mesma maneira que a heurística *Reciprocation-based* quando $B_p^t \leq \sigma$.

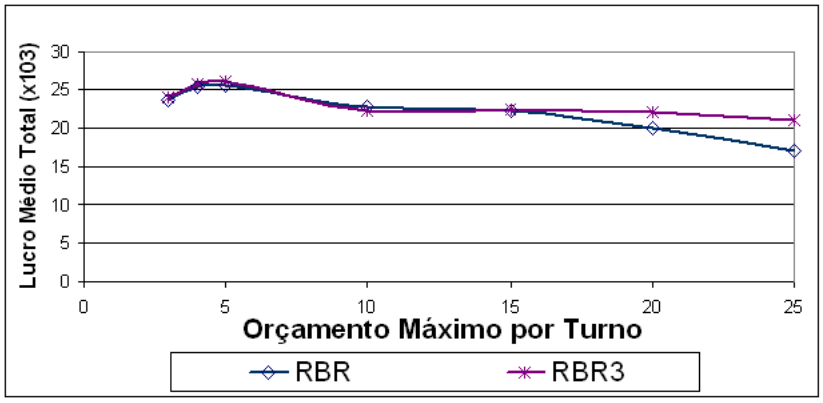



Figura 6.10: $RBR^3$ e $RBR$: lucro (erro $\leq 2\%$ com I.C. de $95\%$)

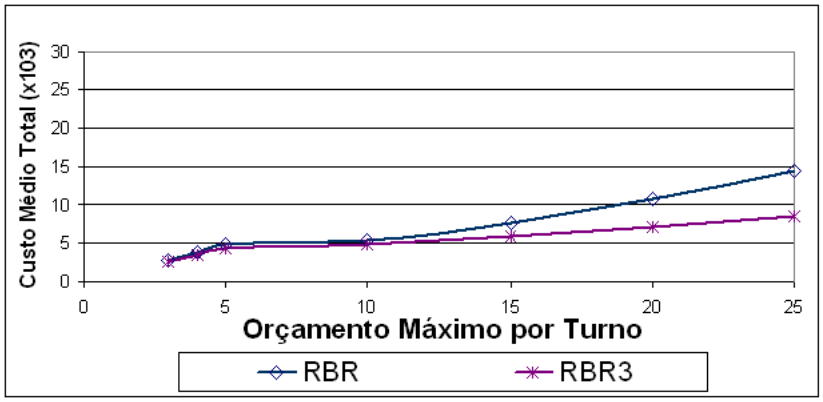



Figura 6.11: $RBR^3$ e $RBR$: custo (erro $\leq 2\%$ com I.C. de $95\%$)

Definimos, também após experimentos preliminares, usar o parâmetro de reinício aleatório como sendo $\epsilon = 10\%$, que foi o que levou a heurística $RBR^3$ aos melhores resultados. Com esse parâmetro, comparamos o desempenho das heurísticas $RBR$ e $RBR^3$.

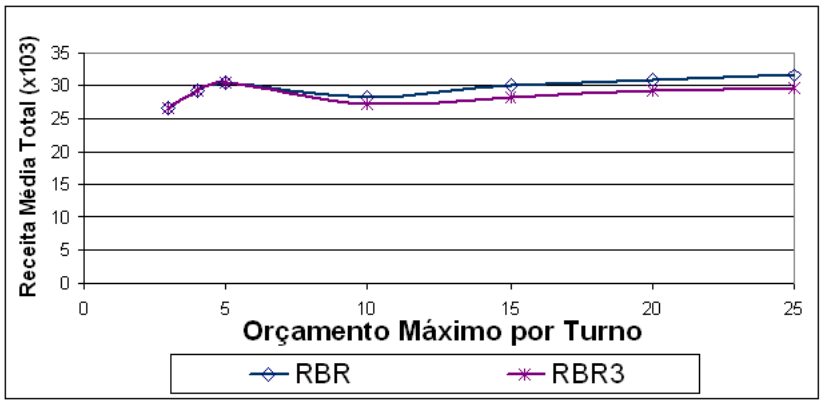


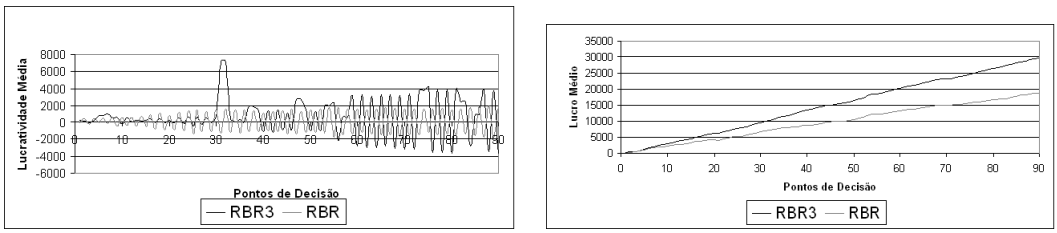

Figura 6.12: $RBR^3$ e $RBR$: receita (erro $\leq 2\%$ com I.C. de $95\%$)

A Figura 6.10 mostra que o lucro final da heurística $RBR^3$ acaba superando cada vez mais de longe o que foi obtido pela heurística $RBR$ quando o orçamento se torna maior do que $15$.

É importante observar que, de maneira análoga ao que acontece na comparação das heurísticas $RBR$ e *Reciprocation-based*, ocorre que a heurística $RBR^3$ consegue obter uma economia mais significativa de custos, ainda que comprometendo parte de sua receita, de forma que sua lucratividade final se torna mais alta.

Podemos ver na Figura 6.12 que a receita alcançada pela heurística $RBR$ é superior à receita recebida pela heurística $RBR^3$ para orçamentos maiores ($B_p^t > \sigma = 5$), e igual quando o orçamento é mais baixo ($B_p^t \leq \sigma = 5$), já que nesses casos ambas as heurísticas atuam da mesma maneira que a heurística *Reciprocation-based*. Ao mesmo tempo, na Figura 6.11 vemos que a heurística $RBR^3$ é mais parcimoniosa com os custos, conseguindo uma economia que supera a perda de receita.

Uma análise cuidadosa dos dados dá algumas evidências de que este comportamento pode ser explicado em função do fenômeno de ótimo local discutido acima. A Figura 6.13 é uma amostra aleatória da execução, num mesmo cenário também selecionado aleatoriamente, das duas heurísticas restritivas de orçamento. Os resultados apresentados nas duas partes da figura são de um mesmo nó, aleatoriamente selecionado. A Figura 6.13(a) mostra a lucratividade média ao longo dos diferentes pontos de decisão, considerando os valores de lucratividade em cada turno até o ponto de decisão, enquanto a Figura 6.13(b) mostra o lucro acumulado nesses pontos.




(a) Evolução da profitabilidade média de um nó escolhido aleatoriamente ao longo do tempo

(b) Evolução do lucro acumulado de um nó escolhido aleatoriamente ao longo do tempo

Figura 6.13: Evoluções da lucratividade média e lucro acumulado de um nó selecionado aleatoriamente, executando as heurísticas $RBR$ e $RBR^3$

Podemos observar na Figura 6.13(a) que a heurística $RBR$ apresenta frequentes oscilações, alternando lucratividades positivas e negativas. Estas oscilações são resultado de situações em que o nó inverte sua estratégia, escolhendo aumentar (ou diminuir) o orçamento utilizado mas poucos turnos depois verifica que a lucratividade voltou a ser negativa, o que o força a novamente mudar sua decisão, e assim sucessivamente. Como o aumento sucessivo da quantidade de parcerias mutuamente lucrativas, o ganho ou a perda decorrentes de cada decisão acabam sendo progressivamente maiores, o que explica as oscilações de amplitude maior que se verifica ao longo do tempo.

Os momentos de oscilação são exatamente os pontos de ótimo local. Podemos observar que, nos mesmos pontos, a heurística $RBR^3$ possui períodos mais longos em que a lucratividade permanece positiva, pois está menos sujeita a esse fenômeno.

O resultado disso pode ser visto na Figura 6.10, onde o lucro acumulado da heurística $RBR^3$ acaba crescendo mais rapidamente que o da heurística $RBR$.

## 6.4 Discussão

As estratégias de seleção de serviços em ambientes P2P baseados em reciprocidade não podem se basear na busca por uma solução ótima, posto que os custos computacionais, a complexidade e o indeterminismo do sistema tornam este método inimplementável em termos práticos. Aos nós, portanto, resta a alternativa de buscar métodos baseados em heurísticas, que possam fazer seleções satisfatórias, ainda que não disponham de todas as informações sobre o ambiente.

As estratégias de seleção de serviço podem elevar os níveis de receita dos nós para níveis bastante altos. Todavia, isso pode comprometer o lucro dos nós porque eles terão que arcar com altos custos para conseguir oferecer uma grande quantidade de serviços. Assim, quando o orçamento é muito limitado, os nós precisam selecionar os serviços mais lucrativos ao passo que, se o orçamento se torna mais alto, um cuidado adicional deve ser tomado a fim de se comprometer somente com a quantidade necessária de custo para manter um nível interessante de lucratividade.

Neste capítulo mostramos como é possível usar a reciprocidade do ambiente para conciliar isso, selecionando serviços de acordo com as interações passadas, a fim de maximizar as relações mutuamente lucrativas, mas ao mesmo tempo ilustrando que é interessante manter à vista o gerenciamento da relação custo/benefício, pois a reciprocidade e as parcerias podem não serem suficientes para garantir que os nós operarão com lucro.

Cabe obsevar que as diferenças de desempenho que medimos nestes experimentos estão associadas às mudanças no orçamento médio dos nós no sistema, que foi o aspecto ambiental que medimos. Evidentemente existem outros aspectos do ambiente que podem interferir nisso, favorecendo mais ou menos quaisquer das heurísticas.

Os resultados apresentados neste capítulo foram obtidos em um simulador que, em função das características do sistema representado, apresenta um custo computacional excessivamente alto. Além disso, neste ambiente é impossível comparar os resultados das heurísticas contra um selecionador ótimo de serviços, já que a implementação deste método é impraticável, conforme explicado acima.

A fim de se implementar um selecionador ótimo para servir de análise metodológica das heurísticas, teremos que definir um problema que mantenha as características deste problema da seleção de serviços mas, adicionalmente, esteja num nível de complexidade computacional mais baixo, a fim de permitir a implementação de algoritmos que encontrem soluções ótimas e que possam ser utilizados como referência.

# Capítulo 7

# Avaliando Algoritmos de seleção de serviços

Conhecendo algumas heurísticas de seleção de portfólio de serviços e sabendo de certas condições que permitem que algumas tenham desempenho melhor do que outras, deparamo-nos com uma questão de ordem metodológica: é preciso determinar o quanto essas heurísticas estão longe da performance de um algoritmo ideal, que faça sempre a melhor seleção possível.

Todavia, ocorre que não há como se implementar este seletor ideal, devido às características de complexidade e indeterminismo do sistema. A solução, então, é buscar a definição de um problema que, ao tempo em que preserve, do ponto de vista das heurísticas, as mesmas condições que elas encontram quando operam em ambientes reais, também permita a definição de um algoritmo perfeito, que execute a seleção ótima de portfólio. Neste sentido os problemas da Mochila e da Seleção de Carteira de Investimentos guardam importantes peculiaridades das quais tiraremos proveito aqui.

## 7.1 O Problema da Mochila e a Seleção de Portfólio de Serviços

Pode-se perceber a semelhança entre o problema da seleção de portfólio de serviços e o Problema da Mochila[1]. A analogia entre os pesos dos itens e os custos dos serviços, entre o valor dos itens e o lucro dos serviços, e entre a capacidade da mochila e o orçamento a que o nó está limitado permite-nos não só assemelhar os dois problemas como também demonstrar que o problema da seleção de serviços possui complexidade computacional no mínimo igual à do Problema da Mochila.

Não há como definirmos a receita que um nó receberá a partir da oferta de um serviço: isso depende de fatores como quantas vezes o serviço será consumido, qual a reciprocidade que os nós que o consumirem concederão de volta, etc. Mas sabemos que efetivamente haverá alguma *receita futura*, (possivelmente igual a zero) para cada serviço que é oferecido.

Da mesma forma, para cada serviço $s$ os nós podem determinar seu custo total, considerando os aspectos de hardware e software.

Assim, sendo ($c^t_{p,s}$) o custo total que o nó $p$ assume ao oferecer o serviço $s$ no instante $t$, e sendo ($\lambda_{p,s,t}$) a receita futura que advirá para o nó $p$ em função da oferta do serviço $s$ no instante $t$, podemos definir o *lucro* do nó $p$ ao oferecer um serviço $s$ no instante $t$ ($\pi_{p,s,t}$) como sendo a diferença entre a receita futura que será obtida quando esse serviço é selecionado para doação no instante $t$ e o custo a que o nó $p$ incorre para oferecê-lo no mesmo instante.

Formalmente o lucro que um nó $p$ vai receber ao oferecer o serviço $s$ no instante $t$ é dado por

$$\pi_{p,s,t} = \lambda_{p,s,t} - c^t_{p,s}.$$

É importante observar que, ao contrário do Problema da Mochila, em que os valores dos itens são conhecidos, na Seleção de Serviços há uma restrição importante: a receita futura ($\lambda_{p,s,t}$) recebida em função da oferta do serviço $s$ depende de fatores, como o comportamento dos demais nós na oferta/demanda dos serviços, que fogem do controle e não são conhecidos pelo nó que está selecionando os serviços. Além disso, há também o fato de que a receita

[1] O problema da Mochila foi apresentado na seção 2.3.

resultante da reciprocidade dos demais nós que consumiram serviços só será realizada num instante posterior a $t$.

Se desprezarmos por um momento a complexidade de se determinar a receita futura de um serviço, podemos mapear os serviços em itens da mochila da seguinte maneira: para cada item $s$, o seu valor ($p_s$) é dado pelo lucro $\pi_{p,s,t}$ do $s-$ésimo serviço; o peso do item $s$ ($w_s$) é dado pelo custo $c^t_{p,s}$ do $s-$ésimo serviço; e o valor $x_s$, que é $1$ se o item está na mochila ou $0$ caso contrário, é definido como igual a $s^t_{p,s}$ que, por definição, é $1$ caso o serviço seja ofertado ou $0$ caso contrário.

Isso permite determinar que o Problema da Mochila $0/1$ é uma versão simplificada do problema da Seleção de Serviços, num caso em que as receitas futuras são previamente conhecidas. Assim, é possível determinar um algoritmo em tempo polinomial que reduz o Problema da Mochila para o da Seleção de Serviços, embora não possamos afirmar ser a recíproca verdadeira. Por conseguinte, podemos afirmar que o problema da seleção de serviços é no mínimo tão difícil computacionalmente quanto o Problema da Mochila.

Como o Problema da Mochila está na classe *NP-Difícil* [Kellerer et al. 2003], segue que o problema da seleção de serviços também o é.

Apesar de apresentarem uma grande semelhança - o que classifica a Seleção de Serviços como um caso particular do Problema da Mochila - o problema da Seleção de Serviços também guarda importantes diferenças com relação ao Problema da Mochila clássico. Existem três aspectos em que o problema da Seleção de Serviços difere do Problema da Mochila.

1. O valor é desconhecido. Conforme será discutido na Seção 7.3, não há um algoritmo que determine a receita resultante de cada serviço. Ocorre que o valor dessa receita não é facilmente previsível, já que depende do comportamento dos demais nós da comunidade.

2. Múltiplas mochilas. Ao contrário do Problema da Mochila clássico, a Seleção de Serviços apresenta um conjunto de nós, e cada um deles possui um orçamento limitado e seleciona um conjunto de serviços para oferecer. Nesse sentido o problema da seleção de serviços se assemelha ao *Multiple Knapsack problem*, embora ainda guarde diferenças significativas conforme veremos na Seção 7.1.1.

3. Iteratividade. Essa é a principal diferença entre o Problema da Mochila e o problema

da Seleção de Serviços. Enquanto naquele se busca uma solução única e definitiva para o problema, neste a busca é repetida iterativamente, com possíveis mudanças de valores, e o objetivo não é encontrar uma solução pronta, mas sim maximizar a lucratividade ao longo do tempo.

### 7.1.1 A Seleção de Serviços e as Variações do Problema da Mochila

Conforme discutido, uma característica importante que separa o problema da seleção de serviços do Problema da Mochila clássico é o prévio desconhecimento dos valores dos itens; os nós não têm como saber qual a receita que eles receberão em função da oferta de um serviço. Nesse sentido é interessante considerar as semelhanças entre a Seleção de Serviços e o *Online Knapsack problem*, em que os valores dos itens só são conhecidos no instante em que o agente precisa decidir por incluí-los ou não na mochila.

Também na Seção 7.1, mostramos um segundo aspecto que diferencia a seleção de serviços do Problema da Mochila Clássico que é o fato de que não há um único nó selecionando serviços, mas vários deles. Isso tem uma forte analogia com o *Multiple Knapsack problem*.

Todavia, as soluções propostas para o *Multiple Knapsack problem* pressupõem que as múltiplas mochilas existentes podem ser usadas passivamente de forma a permitir encontrar o melhor resultado global. No caso da seleção de serviços, essas considerações não são aceitáveis porque os nós são entidades autônomas e, por esse motivo, não estão obrigados a aceitar seleções pouco lucrativas a fim de aumentar a lucratividade média do sistema. Sendo agentes autônomos e racionais, é forçoso assumir que eles buscarão maximizar seu próprio lucro, mesmo que isso imponha uma perda de desempenho ao sistema como um todo.

Além disso, os sistemas P2P com troca de serviços podem evoluir para uma configuração de *equilíbrio de Nash* [Gupta and Somani 2005]. Se a situação de algum nó for francamente desfavorável e ele não tiver, conforme previsto no modelo de Nash, a possibilidade de melhorar seu desempenho com uma mudança de estratégia, o esperado é que ele se desinteresse e deixe o sistema. Caso uma grande quantidade de nós se encontre em situação desfavorável e decida se retirar da comunidade, o sistema pode entrar em colapso. Isso se contrapõe ao *Multiple Knapsack problem* em que as mochilas permanecem à disposição independente de como estão sendo usadas.

Por esse motivo, ao contrário da concepção das soluções para o *Multiple Knapsack problem*, em que um algoritmo central é executado e encontra soluções para cada uma das mochilas, o problema da seleção de serviço precisa ser resolvido de maneira distribuída, com cada nó tentando encontrar a melhor estratégia para si.

## 7.2 Selecionando ativos ou serviços

Há uma clara relação entre o problema da seleção de carteira[2] e a seleção de portfólio de serviços; em ambos os problemas os agentes precisam definir de que forma, entre muitas alternativas, deve-se dividir um orçamento que está disponível. A rentabilidade dos investimentos é, num e noutro problema, previamente desconhecida. O objetivo dos agentes nos dois casos também é o mesmo: maximizar o lucro.

As semelhanças entre as seleções de serviço e de portfólio são evidentes. Todavia, há que se ressaltar os pontos em que os dois problemas apresentam diferenças inconciliáveis. Nos nossos levantamentos bibliográficos, encontramos quatro aspectos relevantes que diferenciam os dois problemas.

1. Investimento mínimo em ativos. Um investidor tem a possibilidade de dividir seu orçamento entre os diferentes ativos, direcionando uma fração a cada um deles. Essa fração pode ter o tamanho que o investidor julgar conveniente. Os nós não podem arbitrar uma fração do orçamento a ser direcionada a cada um dos serviços, a fim de melhorar sua performance, posto que estes possuem um custo fixo.

2. Limite de ativos contemplados. O problema da seleção de carteira impõe aos investidores um orçamento qualquer, que precisa ser dividido entre todos os ativos. O investidor pode escolher investir $0\%$ do orçamento em algum ativo, assim como os nós podem não oferecer um dado serviço. Todavia, um investidor pode definir sua carteira de forma que todos os ativos sejam contemplados, enquanto na seleção de serviços dificilmente um nó terá a opção de oferecer todos os serviços.

3. Rentabilidade desconhecida. Os nós não têm como saber quanto recebem, ou receberam no passado, em função de cada serviço que é oferecido. Por sua vez, um in-

[2]Uma descrição mais detalhada a respeito desse problema pode ser encontrada na Seção 2.4.

vestidor não só sabe exatamente o quanto recebeu de cada ativo como também pode usar resultados passados para especular quanto poderia ganhar com carteiras diferentes. Isso é útil porque permite determinar conjuntos de treinamento para o emprego de algoritmos de aprendizagem como redes neurais [Fernandez and Gomez 2007], enxame de partículas [Xu et al. 2007] e algoritmos genéticos [Zhang et al. 2006].

4. Variação de rentabilidade entre os investidores. Uma mesma carteira de investimento vai gerar exatamente a mesma rentabilidade para qualquer investidor. Isso não se verifica na relação entre os nós e os serviços porque eles os valoram diferentemente entre si [Mowbray et al. 2006].

### 7.2.1 A Seleção de Serviços e as Variações do Problema da Seleção de Carteira

Apesar de a seleção de serviços e a seleção de carteira de ativos possuírem as discrepâncias que mostramos, também é verdade que, quando consideramos variações do problema da seleção de carteira de ativos, encontramos semelhanças que merecem destaque.

O primeiro problema de particular interesse é a *seleção perpétua de carteiras*, em que o problema da seleção ótima não se restringe a uma solução estabelecida, mas a uma sucessão de soluções ao longo do tempo [Samuelson 1969]. O objetivo é obter a melhor rentabilidade durante toda a "existência" do investidor, com o menor risco possível. Esse aspecto também é pertinente à seleção de serviços. Os nós precisam fazer seleções perpetuamente, uma após a outra, buscando maximizar a lucratividade acumulada ao longo do tempo.

Uma versão mais sofisticada do problema da seleção perpétua de carteiras é mostrada por Campbell et. al. [Campbell and Viceira 1996], onde o investidor precisa fazer múltiplas seleções para maximizar seu ganho ao longo do tempo e ainda precisa lidar com o fato de que as rentabilidade dos ativos muda continuamente. Isso é mais próximo do caso da seleção de serviços, porque o ambiente P2P possui uma natureza dinâmica, e isso faz com que a lucratividade de cada serviço (que varia em função de quais nós o consome, e em que quantidade) também seja variada ao longo do tempo.

Finalmente, Xia et al. [Xia et al. 2000] apresenta uma versão particularmente interessante para o estudo que fazemos a respeito da seleção de serviços. A *seleção de carteira com*

*ordem de rentabilidade esperada* propõe que o investidor possua uma informação a mais a respeito dos ativos em que pode investir: ele consegue classificar os ativos de acordo com a ordem de rentabilidade esperada de cada um deles. Com essa consideração, a formulação do problema da seleção de carteiras é parecida com a que mostramos na seção 2.4 com as seguintes restrições:

$$R_i \geq R_{i+1}, i = 1...n-1$$

Havendo um certo intervalo $[a_i, b_i]$ de variação da rentabilidade de cada ativo $x_i$, de forma que:

$$a_i \leq R_i \leq b_i, i = 1...n$$

Essa restrição adicional determina que haverá uma variação do valor $R_i$ do ativo, e essa variação vai ficar dentro do intervalo $[a_i, b_i]$. Note que apesar de haver uma ordem esperada, é possível que o ativo $i$ acabe tendo um valor menor do que o ativo $i+1$ caso os intervalos se superponham ($b_i \geq a_{i+1}$).

Usando essa formulação do problema da seleção de carteira (os autores propõem uma estratégia para definir a ordem de rentabilidade esperada dos ativos), é possível encontrar seleções de ativos com rentabilidade elevada, e os autores ilustram isso com uma implementação usando algoritmos genéticos.

Este problema é particularmente próximo do problema da seleção de serviços porque um nó pode estabelecer uma ordem de rentabilidade esperada dos serviços, se ele usar suas informações a respeito da reciprocidade dos demais nós. Note que os nós de quem ele mais recebeu recursos são os mais recíprocos. Também é importante ter em mente que esses nós mais "generosos" estão, na verdade, retribuindo serviços que eles consumiram antes. Logo, os serviços consumidos por esses nós são, também, os serviços que, ao serem doados, geraram mais lucro. Dessa forma, tomados na ordem decrescente de saldo dos nós que os solicitam, os serviços estarão sendo tomados numa ordem *esperada* de lucratividade.

É importante ressaltar que não há garantias de que a lucratividade obedeça à ordem estabelecida. A ordem final de rentabilidade dos serviços pode diferir da ordem de reciprocidade dos nós devido a dois tipos de circunstâncias. A primeira é o caso de o serviço que é oferecido não ser consumido, e então o retorno será igual a zero, ou sê-lo por outros nós menos

recíprocos, e então o valor retornado será inferior. A segunda circunstância é o caso de um serviço ser solicitado por mais de um nó, o que torna inexata a comparação baseada na reciprocidade. Por exemplo, supondo que, tomados em ordem decrescente de saldos, o nó $n_1$ seja mais recíproco que o nó $n_2$, e este seja mais recíproco que o nó $n_3$. Se um serviço $s_1$ é solicitado pelo nó $n_1$ e um serviço $s_2$ é solicitado pelos nós $n_2$ e $n_3$, então pode ser que o retorno somado dos nós $n_2$ e $n_3$ tornem o serviço $s_2$ o mais lucrativo. Dessa forma, ao usarmos a ordem de reciprocidade dos nós para decidir a ordem de rentabilidade dos serviços, deparamo-nos com a *possibilidade* de isso refletir o ordenamento geral dos serviços pela rentabilidade, mas não há nenhuma garantia a respeito disso.

## 7.3 Algoritmo de referência para a Seleção de Serviços

Apesar das discrepâncias em relação ao problema da Seleção de Serviços, permanece válido que a abordagem metodológica que é tipicamente usada para se medir a performance de métodos heurísticos para solucionar o Problema da Mochila é bastante robusta, já que emprega a implementação de algoritmos de referência, de forma que as heurísticas podem ter seus resultados comparados com uma solução ótima.

Um possível algoritmo de referência que poderia ser usado na avaliação metodológica do problema da seleção de serviços funcionaria da seguinte forma: conhecendo as receitas futuras que o nó vai receber em função de cada serviço, este algoritmo varreria todo o espaço (busca exaustiva) e encontraria a melhor seleção de serviços para cada instante. Ao avaliarmos os algoritmos de seleção de serviço, eles seriam comparados com o algoritmo de referência para verificar a qualidade da solução encontrada. O algoritmo de referência seria um expediente usado apenas para este fim devido à sua complexidade computacional.

A implementação deste algoritmo, porém, não é praticável em função da complexidade do sistema, conforme discutido abaixo.

Podemos afirmar que isso ocorre porque os nós consomem serviços específicos. Desta forma, o dilema "Selecionar ou não Selecionar" um serviço é, na verdade, uma decisão a respeito de quais parcerias ele vai promover já que, ao selecionar um portfólio de serviços, o nó está selecionando um conjunto de nós que potencialmente vão consumí-los, a despeito dos demais que não se interessam por nenhum dos serviços do portfólio. Como implemen-

tam um escalonamento baseado em reciprocidade, os nós que consomem os serviços do nó $p$ o priorizarão no futuro. Assim, as escolhas que o nó $p$ faz num dado instante influenciam as decisões futuras de escalonamento dos demais, e estas também influenciarão no escalonamento do próprio nó $p$, e assim sucessivamente.

A maior parte da receita recebida da grade é função da reciprocidade das parcerias, ou seja, da doação feita por outros nós que consumiram serviços no passado. Isso significa que não é conveniente desenhar uma função "Oráculo" que saiba antecipadamente as receitas futuras de cada serviço oferecido.

Essa inconveniência se sustenta em dois motivos. O primeiro motivo é que, como descrito acima, a receita futura é influenciada pela escolha atual. Isso significa que não é praticável desenhar uma função que saiba antecipadamente quais as receitas futuras, já que os próprios resultados futuros dependem das escolhas do nó. O segundo motivo é que um nó recebe serviços computacionais em reciprocidade a serviços doados no passado, mas é impossível determinar quanto da receita é devido a cada unidade de servido anteriormente doado, bem como em que instante[3].

Para se construir um oráculo "onisciente", que conheça as receitas futuras em função das seleções feitas por um nó, ele precisará conhecer os custos dos serviços para cada um dos nós, bem como suas demandas e seus algoritmos de escalonamento. Além disso ele precisará conhecer a ordem em que as requisições serão feitas, bem como a ordem em que serão atendidas, e por quem.

Ainda assim, a alternativa de implementar o oráculo que mapeia o espaço de todas as possíveis seleções e todos os possíveis cenários decorrentes (cada seleção descortina um cenário específico) se torna pouco atraente devido à complexidade computacional envolvida. Por esta razão, o oráculo não poderia lidar senão com ambientes muito modestos, com pequena quantidade de nós e serviços.

Isso impõe uma barreira: não sendo possível implementar um algoritmo de referência no modelo que temos estudado até aqui, precisamos contornar este problema. Para isso, teremos que definir um outro problema, que guarde certas características especiais.

[3] Supondo que tenha havido consumo de mais de um serviço, e em mais de um instante no passado.

## 7.4 Formalização de um problema híbrido

Apesar de não ser animador estudar a seleção de serviços como um problema de múltiplas mochilas num ambiente distribuído, podemos apresentar uma simplificação que, sem perder de vista o problema, o torna tratável metodologicamente [Coêlho and Brasileiro 2009].

Como os nós são agentes autônomos e buscam seu benefício individual, podemos tratar o problema de como cada nó pode fazer para maximizar seu próprio ganho, sendo que ele não tem controle sobre o que ocorre no restante do ambiente.

Vamos definir o problema como uma variação do Problema da Mochila, além de considerar aspectos do Problema da Seleção de Carteira de Ativos, determinando as seguintes características:

- O nó conhece os custos locais de todos os serviços;
- O nó não conhece a receita futura de nenhum serviço;
- A receita futura de alguns serviços pode fazer seu lucro ser negativo;
- As receitas podem variar ao longo do tempo;
- Há uma função $B$ que ordena os serviços em função da receita esperada;
- Há muitos nós atuando ao mesmo tempo, e não há como controlar o comportamento dos demais;
- É um problema iterado, ou seja, não basta fazer uma única seleção. São necessárias constantes seleções ao longo do tempo.

A função $B$ é definida da seguinte maneira: ordena-se os nós pela ordem de reciprocidade (por exemplo, se o esquema de reciprocidade for a NoF [Andrade et al. 2007], podemos usar o saldo) e toma-se os serviços que serão selecionados pelos nós nessa mesma ordem. Os serviços demandados pelos nós que são mais recíprocos são os que se espera serem os mais lucrativos, embora não se saiba o quanto, e os demais se seguem em seqüência decrescente.

Note que estas restrições definem o problema da seleção de serviços do ponto de vista de um nó, que é o que desejamos resolver. Agora podemos definir uma variação do Problema da Mochila que possui particularidades específicas, como segue.

Seja um conjunto de $n$ itens, e um vetor de pesos $W = w_1, w_2...w_n$. Cada item possui um valor desconhecido, possivelmente negativo e possivelmente variável ao longo do tempo. O vetor $P^t = \langle p^t(1), p^t(2)...p^t(n)\rangle$ são os valores de cada item no instante $t$. Os itens deverão ser postos ou não em uma mochila que suporta uma carga máxima igual a $c$. O vetor $X^t = \langle x_1^t, x_2^t...x_n^t\rangle$, onde $x_i^t \in \{0, 1\}$ indica quais itens estarão na mochila no instante $t$.

O agente não conhece os valores $p^t(1), p^t(2)...p^t(n)$ dos itens, mas dispõe da função $B(P^t)$ que retorna a lista de itens em ordem decrescente de valor.

O problema, então, é encontrar sucessivas seleções de itens para colocar na mochila de forma que o valor da mochila seja maximizado.

Formalmente,

$$\text{Maximize } \int_{t_0}^{\infty} \left( \sum_{j=1}^{n} p^t(j) \cdot x_j^t \right) dt$$

sujeito a

$$\sum_{j=1}^{n} w_j \cdot x_j^t \leq c \text{ considerando } x_j^t \in \{0, 1\}$$

Sendo que $t_0$ é o instante em que o agente passa a atuar na seleção de serviços, $\forall p^t(j), 1 \leq j \leq n$, $p^t(j)$ é desconhecido. Mas a função $B(P^t) \rightarrow \langle l_1, l_2, ..., l_n\rangle$ onde $0 \leq l_i \leq n$ e $\langle l_1, l_2, ..., l_n\rangle$ é tal que $\forall l_i, l_k \in \langle l_1, l_2, ..., l_n\rangle$ se $i < k$ então espera-se ter $p^t(l_i) \geq p^t(l_k)$.

É importante ressaltar que em casos reais o tempo é contínuo, já que os portfólios de serviços estarão sendo constantemente modificados, e os nós poderão estar doando ou recebendo serviços uns dos outros. Assim, para lidar com o problema, faremos ainda duas simplificações: o tempo será discretizado, e haverá um intervalo $[t_o, t_f]$ de tempo que o agente terá para maximizar sua lucratividade. Com isso podemos definir a função objetivo como sendo

$$\text{Maximize } \sum_{t=t_o}^{t_f} \sum_{j=1}^{n} p^t(j) \cdot x_j^t$$

## 7.5 A Metodologia

Dado que a receita futura que um nó recebe é dada pela doação feita por diferentes nós, e considerando o teorema do Limite Central [Dudley 1978], podemos estimar a receita futura de acordo com uma distribuição normal. É importante observar, entretanto, que existe uma chance de que algum serviço oferecido em um turno $t$ não seja consumido por outros nós. Isso acontecerá para um serviço $s$ se nenhum dos nós que contém $s$ em seu favor típico esteja consumindo este serviço no turno $t$. A probabilidade disso acontecer depende de fatores como a quantidade de nós na comunidade, o número de serviços diferentes solicitados pelos nós (doravante chamaremos este valor de *tamanho* do favor típico - $|\mathcal{F}_p|$), e da quantidade de serviços que estão sendo oferecidos (isso depende do orçamento dos nós e dos custos dos serviços).

Assim, assumindo que a demanda por qualquer dos serviços é similar, a probabilidade de um serviço $s$ oferecido por um nó $p$ num instante $t$ não ser consumido por um nó $k$ é dada por $(1 - \frac{|\mathcal{F}_k|}{n})$. Considerando um universo de $N$ nós, a probabilidade de um serviço não ser consumido por nenhum nó é dada pela expressão $\Omega$ abaixo.

$$\Omega = \prod_{i=1}^{N} (1 - \frac{\overline{|\mathcal{F}_i|}}{n})$$

Seja $\lambda_{p,s,t}$ a receita futura que o serviço $s$ retorna para $p$ em função de que $p$ ofereceu $s$ no turno $t$. Considerando a probabilidade de um serviço não ser consumido, este valor é dado por:

$$\lambda_{p,s,t} = \begin{cases} 0 \text{ com probabilidade } \Omega \\ \text{ou} \\ X \sim N(\alpha, \sigma) \text{ com probabilidade } 1 - \Omega \end{cases}$$

onde $\alpha$ é a média e $\sigma$ é o desvio padrão da receita futura.

Seja $\overline{c_p}$ o custo médio, por serviço, para o nó $p$, de todos os serviços, e $B_t$ o orçamento médio disponível por turno no turno $t$. Então, a quantidade média de serviços diferentes oferecidos pelo nó $p$ é dada por $(\frac{B_t - h_p \cdot q_p}{\overline{c_p}})$. Além disso, sendo $\overline{c}$ o custo médio de um serviço em todo o sistema, e sendo $p^{don}$ a probabilidade de um nó estar no estado *doando*, a quantidade esperada de nós que estão doando é dada por $p^{don} \cdot (N-1)$, enquanto a quantidade

de nós no estado *recebendo* é dada por $(1 - p^{don}) \cdot (N - 1)$.

Dessa forma, considerando que $p$ esteja oferecendo serviços no turno $t$, a quantidade esperada de nós que estão junto disputando pela doação do serviço $s$, contando com $p$, é dada por:

$$\chi = (p^{don} \cdot (N-1)) \cdot \frac{\frac{B_t - \overline{h} \cdot \overline{q}}{\bar{c}}}{n} + 1$$

onde $\overline{q}$ é o custo médio de hardware e $\overline{h}$ é a capacidade média de hardware no sistema.

Considerando um ambiente de competição, a receita gerada por cada serviço precisa ser dividida entre os nós que estão disputando (os nós que estão oferecendo o mesmo serviço). Para simplificar, nós definimos $C_{p,s}$ como o custo total para prover cada serviço, considerando tanto o custo de software ($c_{p,s}$) quanto o custo de manutenção da estrutura local ($h_p \cdot q_p$). Na Seção 4.2 há uma explicação mais detalhada destas variáveis.

Assim, o lucro médio gerado pelo serviço $s$ ao ser oferecido pelo nó $p$ no turno $t$ é dado por

$$\pi_{p,s,t} = \frac{\lambda_{p,s,t}}{\chi} - C_{p,s}.$$

## 7.6 Avaliando Heurísticas

Nesta Seção usamos a metodologia proposta para avaliar os algoritmos heurísticos que nós propusemos no Capítulo 6. Para isto definimos dois limites de comparação metodológica: um limite inferior, que é a seleção feita por um nó que não faz nenhuma consideração ao escolher os serviços oferecidos, e um limite superior que é determinado pelo algoritmo de referência.

### 7.6.1 Desempenho de um método que faz seleções aleatórias

Desenvolvemos um modelo simples para medir a performance de um algoritmo trivial de seleção de serviços que escolhe aleatoriamente o portfólio. Para fazer isso, é necessário calcular a receita futura que $p$ obtém quando oferece o serviço $s$ num turno $t$.

Considerando a probabilidade de alguns dos serviços oferecidos por um nó seja consumido por algum outro, podemos definir uma expressão para determinar a receita futura

recebida da grade por um nó que seleciona serviços aleatoriamente. Sendo $\overline{\lambda_p}$ a receita futura média recebida pelo nó $p$ para cada serviço provido, e sendo $\overline{C_p}$ o custo médio a que $p$ está sujeito para oferecer um serviço, a receita futura do nó $p$ é dada por

$$(1-\Omega)\cdot\left(\frac{\overline{\mathcal{B}^t}}{\overline{C_p}}\right)\cdot\overline{\lambda_p}$$

onde $\Omega$ é a probabilidade de um serviço não ser consumido e $\overline{B^t}$ é o orçamento médio dos nós no turno $t$.

Considerando um ambiente de competição, e um total esperado de $\chi$ nós que oferecem cada serviço, a receita esperada recebida em reciprocidade de cada serviço oferecido é dada por

$$\Psi=\frac{(1-\Omega)\cdot\left(\frac{\overline{\mathcal{B}^t}}{\overline{C_p}}\right)\cdot\overline{\lambda_p}}{\chi}$$

.

### 7.6.2 Medindo o Desempenho

Executamos simulações criando mochilas parametrizadas conforme proposto na descrição da metodologia. Cada mochila corresponde a um turno em que o agente deverá fazer uma seleção de serviços e colher um resultado (que ele desconhece). Desta forma, criamos um ambiente metodológico que representa uma comunidade de $50$ nós com custo médio de hardware por unidade computacional $h_p = 0.1$ e quantidade média de unidades computacionais dos nós $q = 10$. Os custos de software dos serviços foram escolhidos aleatoriamente a partir de uma distribuição normal $X \sim N(0.75, 0.25)$ (considerando que os custos são compostos de muitos diferentes fatores, e também de acordo com o Teorema do Limite Central), e o favor típico de todos os nós sendo composto por dois serviços. A receita gerada por cada unidade de serviço foi calculada usando a média e o desvio padrão dos valores obtidos em experimentos reais usando características similares e é normalmente distribuída como $X \sim N(10.34, 2.5)$. Finalmente, a probabilidade de um nó estar doando é $p^{don} = 50\%$.

Implementamos um Algoritmo de Referência que executa a melhor seleção de Mochila, bem como as heurísticas *Cost-based*, *Reciprocation-based*, $RBR$ e $RBR^3$ para serem testados usando esta metodologia. Apesar de essas heurísticas terem sido comparadas no capítulo

6, é importante ter em vista que outras variações ambientais poderão ser feitas, e os resultados podem favorecer mais ou menos diferentes heurísticas.

Os lucros médios finais por turno, incluindo os limites metodológicos (Analítico e Referência) podem ser vistos na Figura 7.1.

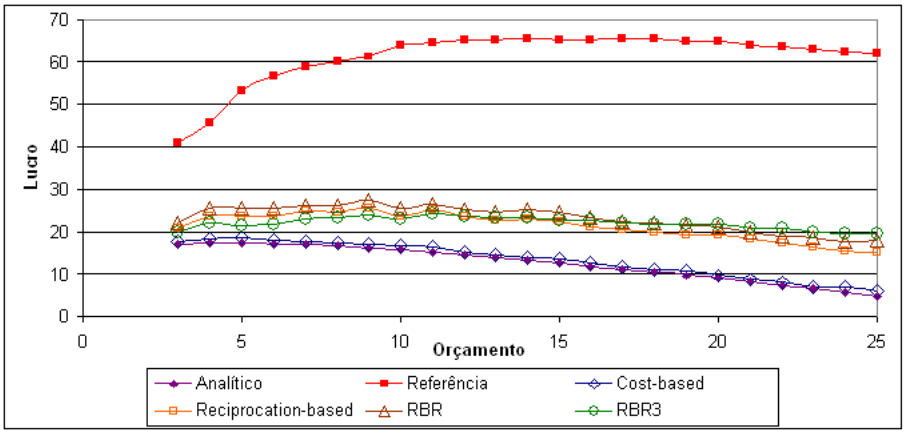


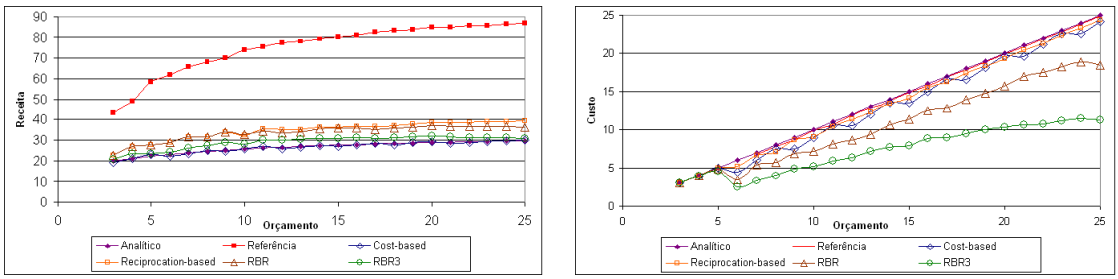

Figura 7.1: Lucro das heurísticas. Resultados obtidos usando a metodologia proposta (erro $\leq 2\%$ com I.C. de $95\%$)

A performance de todas as heurísticas são muito ruins quando comparadas com o algorítimo de referência. Esta diferença pode ser explicada por uma razão simples: o algoritmo de referência conhece os serviços que não serão consumidos em cada instante, e jamais os oferece. Considerando os parâmetros que usamos, a probabilidade de cada serviço não ser consumido gira em torno de $72.05\%$.

Note que quando o orçamento é muito grande (próximo a $25$), o lucro de todas as heurísticas, incluindo o algoritmo de referência, começa a cair. Isto é explicado pelo fenômeno da "mão invisível", conforme predito por Adam Smith em ambientes em que há competição. Como podemos ver na figura 7.2(a), mesmo o algoritmo de referência encontra uma receita máxima que não é superável quando o orçamento é muito alto. A explicação deste fenômeno é que, nesses casos, existem muitos nós oferecendo o mesmo serviço e isso diminui a receita de cada serviço (que é dada por $\frac{\lambda_{p,s,t}}{\chi}$, porque o valor de $\chi$ tem uma proporção direta com o orçamento médio do sistema.

Assim, baseando-nos nesta metodologia, no limite, se todos os nós estão usando orçamentos muito altos, a melhor escolha é não oferecer nenhum serviço porque a receita vai se tornar muito baixa ($\lim_{\chi \to \infty} \frac{\lambda_{p,s,t}}{\chi} = 0$). Isto é verdade em sistemas reais também, porque

havendo muitos nós oferecendo todos os serviços, o número de doações que se conseguirá fazer para cada serviço oferecido se torna excessivamente baixo, diminuindo o lucro final gerado por eles. Neste caso hipotético a melhor escolha seria não oferecer nenhum serviço.




(a) Receita obtida usando a metodologia (erro $\leq 2\%$ com I.C. de $95\%$)

(b) Custo obtido usando a metodologia (erro $\leq 2\%$ com I.C. de $95\%$)

Figura 7.2: Custo e Receita obtidos usando a metodologia

A Figura 7.2(b) mostra o custo a que as heurísticas se sujeitam. Os resultados são similares aos que se vê no Capítulo 6: as heurísticas $RBR$ e $RBR^3$ conseguem economizar uma quantidade expressiva do orçamento, embora percam um pouco da receita e assim conseguem alcançar lucros maiores. Estes resultados obtidos pelas heurísticas na metodologia podem ser vistos mais claramente (um "*zoom*") na Seção 7.7.2.

Efetivamente, pode-se usar esta metodologia a fim de se inferir muitas coisas a respeito das heurísticas de seleção de serviço. Todavia, é importante observar o quanto os resultados alcançados pelas heurísticas neste ambiente estão próximos dos resultados obtidos por elas quando operam em ambientes P2P que efetivamente permitem trocas de serviços.

## 7.7 Validando a metodologia

Esta metodologia funciona a partir da concepção de um ambiente específico, em que as heurísticas são imersas e operam da mesma forma que o fariam se estivessem efetivamente trocando serviços. Não obstante, ainda é necessária uma investigação adicional a respeito da acurácia dos resultados obtidos pelas heurísticas quando são avaliadas segundo esta metodologia em comparação com os que se obtém em ambientes com múltiplos nós compartilhando múltiplos serviços.

Para fazer isso, configuramos dois ambientes: o primeiro, chamado de ambiente

*metodológico* é uma implementação do problema simplificado descrito na Seção 7.5. O segundo, chamado de ambiente *real* é uma implementação do modelo de um ambiente em que os nós efetivamente trocam serviços usando a NoF como esquema de reciprocidade, conforme fizemos nos experimentos dos capítulos 5 e 6.

Consideramos um ambiente metodológico tão próximo quanto possível do ambiente real a fim de obter em um os resultados o mais próximos possíveis dos do outro. Ambos os ambientes foram implementados usando parâmetros similares: o número de nós e serviços, os custos e as receitas médias dos serviços, a probabilidade de os nós estarem doando, o orçamento dos nós e o número de serviços no favor típico dos nós. Todos esses valores são os mesmos descritos no Capítulo 6.

### 7.7.1 Avaliação Estatística

Para garantir que os resultados do ambiente real não sejam significativamente diferentes dos resultados do ambiente metodológico, é necessário garantir dois aspectos: a correlação entre os dados metodológicos e reais precisa ser alta e os valores médios não podem ser muito diferentes. Para validar esta similaridade entre os dados decidimos executar duas análises estatísticas diferentes: o valor P do *Teste T*, a fim de verificar se os resultados possuem médias próximas, e *Pearson*, a fim de verificar se a correlação linear entre os dados reais e metodológicos é positiva.

Tabela 7.1: Resultados do teste de Pearson para Custos, Receitas e Lucros dos ambientes real e metodológico

| | Custo | Receita | Lucro |
|---|---|---|---|
| Cost-Based | 0.998 | 0.993 | 0.998 |
| Reciprocation-Based | 0.999 | 0.925 | 0.825 |
| RBR | 0.985 | 0.755 | 0.601 |
| $RBR^3$ | 0.975 | 0.584 | 0.438 |

Como podemos ver na Tabela 7.1, todos os resultados de teste de Pearson são positivos, mostrando que o comportamento das heurísticas nos ambientes real e metodológico são similares em todos os cenários. Os resultados do Teste T mostrados na Tabela 7.2 permite-nos

ver que a probabilidade de as médias dos dados metodológicos e reais serem diferentes é baixa em quase todos os casos e, mesmo nos piores exemplos, é sempre inferior a $45\%$.

Tabela 7.2: resultados do Teste T para Custos, Receitas e Lucros dos ambientes real e metodológico

| | Custo | Receita | Lucro |
|---|---|---|---|
| Cost-Based | 0.133 | 0.0005 | 0.0003 |
| Reciprocation-Based | 0.087 | 0.205 | 0.188 |
| RBR | 0.016 | 0.138 | 0.431 |
| $RBR^3$ | 0.030 | 0.449 | 0.0465 |

Os resultados mostram que a metodologia pode ser usada para fazer uma boa predição sobre o comportamento das heurísticas, a despeito do fato de que a sua acurácia é menor quando avaliando heurísticas que usam decisões "inteligentes", como a $RBR$ e a $RBR^3$.

Isto acontece porque esses resultados foram obtidos usando uma receita fixa para cada serviço, que foi a receita média que encontramos nas simulações descritas no Capítulo 6. De acordo com aqueles resultados, estabelecemos que a receita seria normalmente distribuída como $X \sim N(10.34, 2.5)$. Ocorre que este valor fixo não é constante em todos os experimentos. Com efeito, há uma receita média para cada heurística, em cada condição ambiental.

Entretanto, decidimos usar um valor fixo para a receita dos serviços porque em casos reais será impossível antecipar a média e o desvio padrão das receitas dos serviços sob diferentes condições ambientais. Todavia, posto que para cada combinação cenário/heurística existe uma receita média diferente, repetimos a experiência usando um parâmetro artificial, que foi a receita média obtida usando cada um dos diferentes orçamentos. Os resultados são mostrados abaixo

A Tabela 7.3 mostra que a correlação se torna mais positiva quando usando médias próximas das que foram encontradas nas condições específicas. Da mesma forma, os resultados do Teste T na Tabela 7.4 indicam uma probabilidade ainda menor de que os dados reais e metodológicos tenham médias diferentes (sempre $\leq 18\%$). Note que, infelizmente, é impossível conhecer previamente estes valores de receita para antecipar os resultados corretos das heurísticas. Contudo, a metodologia permite predizer o comportamento delas, bem como

Tabela 7.3: Resultados do teste de Pearson para Custos, Receitas e Lucros dos ambientes real e metodológico usando informações artificiais

| | Custo | Receita | Lucro |
|---|---|---|---|
| Cost Based | 0.999 | 0.993 | 0.997 |
| Reciprocation Based | 0.999 | 0.780 | 0.975 |
| RBR | 0.996 | 0.823 | 0.930 |
| $RBR^3$ | 0.948 | 0.798 | 0.918 |

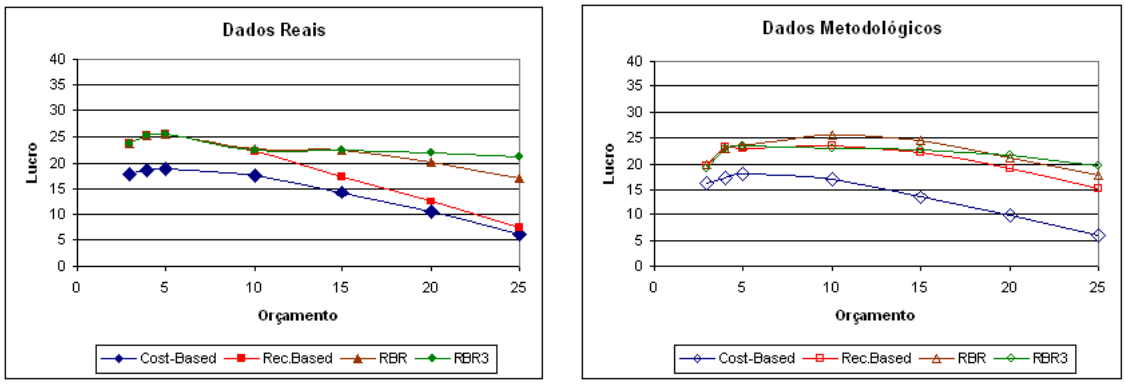
Tabela 7.4: Resultados do Teste T para Custos, Receitas e Lucros dos ambientes real e metodológico usando informações artificiais

| | Custo | Receita | Lucro |
|---|---|---|---|
| Cost Based | 0.004 | 0.0004 | 0.002 |
| Reciprocation Based | 0.087 | 0.094 | 0.180 |
| RBR | 0.118 | 0.121 | 0.048 |
| $RBR^3$ | 0.062 | 0.009 | 0.0004 |

fazer uma comparação entre as mesmas, além de confrontá-las contra um hipotético seletor ótimo de serviços.

### 7.7.2 Avaliação Gráfica

Abaixo mostramos gráficos emparelhados, fazendo um "zoom" para se ver os resultados metodológicos mais de perto, permitindo uma comparação visual entre estes resultados e os resultados reais.




(a) Lucro das heurísticas: ambiente real (b) Lucro das heurísticas: ambiente metodológico

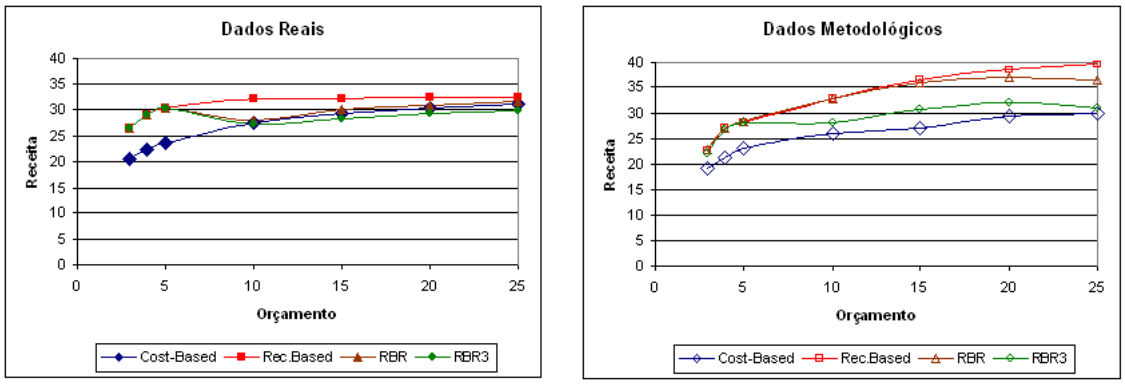
Figura 7.3: Lucro: dados dos ambientes real e metodológico

Como podemos ver na Figura 7.3, os lucros finais são bastante similares, apesar de a metodologia haver previsto um lucro mais alto para a heurística *Reciprocation-based* quando usando orçamentos maiores, e um lucro menor quando o orçamento é mais limitado. Esta diferença pode ser explicada pela diferença de valores da receita dos serviços, conforme mencionamos na Seção 7.7.1. Note que a diferença está principalmente nos resultados da receita: a receita real (Figura 7.4(a)) é ligeiramente inferior à receita prevista metodologicamente (Figura 7.4(b)) para orçamentos mais restritos e o inverso ocorre quando o orçamento se torna mais generoso.

A heurística $RBR$ também apresentou uma leve discrepância nos resultados com orçamentos intermediários (entre $5$ e $15$), e podemos ver nas Figuras 7.4(a) e 7.4(b) que esta discrepância se deveu a diferenças nas receitas obtidas.

As heurísticas $RBR$ e *Reciprocation-based* são as que trabalham tentando promover mais relações lucrativas com outros nós e por isso as doações que fazem acabam tendo mais chance de encontrar reciprocidade, o que aumenta o valor médio de receita por serviço doado.

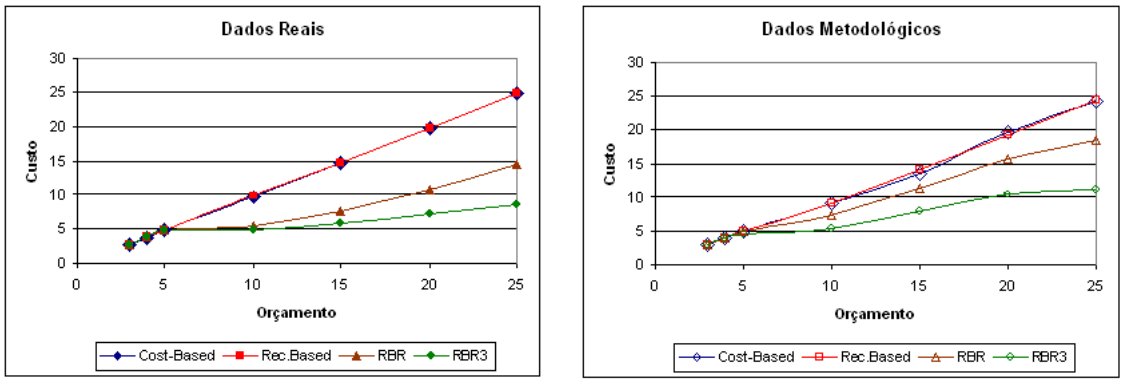
A heurística $RBR^3$ limita estas relações ao restringir o orçamento, e por isso sua receita por serviço doado varia menos.




(a) Receita das heurísticas: ambiente real (b) Receita das heurísticas: ambiente metodológico

Figura 7.4: Receita: dados dos ambientes real e metodológico

Os resultados dos custos mostrados na Figura 7.5 são mais acurados para todas as heurísticas, apesar de a previsão ser que as heurísticas de Restrição Orçamentária fossem ligeiramente mais perdulárias (Veja a Figura 7.5(b)) do que e efetivamente acabaram se revelando (Veja a Figura 7.5(a)).




(a) Custo das heurísticas: ambiente real (b) Custo das heurísticas: ambiente metodológico

Figura 7.5: Custos: dados reais e metodológicos

## 7.8 Discussão

Mostramos que o problema da seleção de serviços é uma variação de um problema clássico que é estudado nas Ciências Econômicas: a seleção de Carteira de Ativos. Apesar de entre

os dois problemas haver uma distância considerável, alguns aspectos os aproximam; particularmente a correspondência entre os pares (*investidor*,*ativos*) no primeiro domínio com (*nó*,*serviços*) no segundo. Além disso, se o problema da seleção de carteiras for tomado perpetuamente - caso em que o investidor precisa fazer múltiplas seleções ao longo do tempo tentando maximizar a lucratividade acumulada - ele se assemelha ainda mais ao caso dos nós, que convivem por tempo indeterminado numa comunidade, consumindo e oferecendo recursos, mas mantendo sua lucratividade a mais alta possível. Uma variação do problema da carteira de ativos propõe que o investidor, apesar de desconhecer o valor exato da rentabilidade futura de um investimento, tem acesso a uma ordem de rentabilidade, que classifica os ativos de acordo com a lucratividade que se espera deles. Isso é uma particularidade relevante, já que os nós podem fazer uma classificação semelhante usando o saldo dos demais para ordenar os serviços.

Sob um outro ponto de vista, mostramos que o problema da seleção de serviços também possui afinidades com variações de um problema da Ciência da Computação, que é o Problema da Mochila. A relação é simples de ser mostrada, associando a capacidade da mochila com o orçamento dos nós, os itens da mochila aos serviços que podem ser oferecidos, os valores dos itens ao lucro obtido pela doação dos serviços, e os pesos dos itens aos custos dos serviços. Dessa forma podemos abordar, metodologicamente, o problema da seleção de serviços da forma em que trabalhos correlatos fazem com o problema da mochila: usando um algoritmo de busca exaustiva para servir de referência, e as abordagens propostas são comparadas com esse algoritmo a fim de medir a qualidade de suas soluções.

A fim de usarmos uma abordagem similar no estudo do problema da seleção de serviços, definimo-lo do ponto de vista de um único nó, que não tem controle sobre os demais nem sobre os aspectos ambientais da própria comunidade em que ele convive. Dessa forma, podemos definir um algoritmo de referência que conhece as receitas futuras dos serviços, e consegue fazer a melhor escolha. Além desse algoritmo de referência, estabelecemos analiticamente a performance de um nó hipotético que seleciona os serviços aleatoriamente.

Medimos o desempenho das heurísticas e constatamos que elas ficam muito distantes da performance do algoritmo de referência devido ao fato de elas não terem como antecipar se um dado serviço será ou não consumido.

Essa metodologia que definimos aqui permite estudar o desempenho esperado de difer-

entes algoritmos de seleção de serviços, definindo um ambiente que pode ser parametrizado para representar diferentes condições reais, prestando-se ao suporte a múltiplos estudos a respeito do desempenho dos algoritmos.

Avaliamos ainda a acurácia dos resultados desse modelo quando comparados aos resultados que são obtidos por nós quando operam em ambientes reais com troca de serviço. Para observar esses aspectos, definimos experimentos em que pudemos observar vários nós convivendo de fato sob diferentes condições ambientais de ambientes P2P com múltiplos serviços e repetimo-los usando a metodologia a fim de comparar os resultados.

# Capítulo 8

# Conclusões e Trabalhos Futuros

Neste capítulo apresentaremos as conclusões de nosso trabalho, ilustrando as contribuições apresentadas. Além disso, mostramos os trabalhos que podem ser desenvolvidos a partir, e em conseqüência deste.

## 8.1 Conclusões

Num ambiente P2P há um princípio básico de que os nós atuem tanto no papel de provedores quanto no de consumidores de serviços. Isso garante a escalabilidade e a qualidade dos serviços providos pelo próprio sistema.

Entretanto, tomados como agentes autônomos que são, os nós precisam ter interesse de permanecer no sistema. Em termos econômicos, a quantidade de benesses resultantes da convivência numa comunidade precisa superar o investimento necessário para permanecer nela. Em outras palavras, o ambiente precisa ser lucrativo. Caso o ambiente não seja lucrativo, os nós se desinteressarão e isso pode inviabilizar o sistema como um todo. Considerando ambientes baseados em reciprocidade, e em que há múltiplos serviços sendo trocados, os nós precisarão encontrar sua lucratividade a partir do retorno dado pelos demais que consumiram seus serviços.

Este trabalho descortina o mundo em que os nós compartilham múltiplos serviços num ambiente baseado em reciprocidade, buscando maximizar sua lucratividade e, assim, viabilizar a existência do próprio ambiente.

Mostramos aqui que os serviços possuem custos e receitas variados entre os nós, posto

que as características operacionais e ambientais determinam condições diferentes para manter a infra-estrutura e disponibilizar serviços na comunidade.

Dessa forma, as seleções de serviços se tornam o principal meio de os nós obterem recursos da comunidade. Selecionar os serviços que sejam consumidos pelos nós com os quais as parcerias sejam mais lucrativas trará resultados mais interessantes. Conforme evidências mostradas neste trabalho, dentro da infinidade de seleções possíveis, a amplitude de lucratividade resultante também é muito grande, o que impõe o uso de critérios para seleção de serviços.

Isso nos permite afirmar que os nós efetivamente precisam se preocupar em definir métodos para a seleção do portfólio de serviços, sendo esta a única maneira pela qual eles podem melhorar sua lucratividade. Nesse sentido, definimos dois grupos de métodos: os que implementam heurísticas baseadas no modelo do sistema e os que implementam heurísticas de restrição orçamentária usando uma abordagem de subida de colina. Neste segundo grupo ainda podemos identificar uma implementação que utiliza o reinício aleatório para evitar pontos de ótimo local.

Contudo, já que os algoritmos que os nós usarão para selecionar serviços precisarão ser baseados em heurísticas, e considerando também que o problema pertence à classe *NP-Difícil*, torna-se necessário definir uma abordagem alternativa que permita uma avaliação da qualidade dos algoritmos de seleção.

Em problemas de otimização com alta complexidade computacional, tipicamente usa-se um método de busca exaustiva pela solução ótima como algoritmo de referência para avaliar as soluções heurísticas que são consideradas. Porém, no mundo da seleção de serviços em ambientes P2P essa alternativa se torna inviável devido ao alto custo computacional envolvido, mesmo quando considerados sistemas com dimensões bastante modestas.

Para lidar com isso, apresentamos a definição de um problema híbrido, forjado em parte pelo formalismo do problema da seleção de carteira de ativos, que é estudado em Economia, e usando a metodologia de avaliação típica para as abordagens do problema da mochila, que é estudado em Ciência da Computação. Encapsulando em equações probabilísticas os aspectos externos ao agente que está selecionando serviços, este novo problema possui a característica de ser tratável metodologicamente via algoritmo de referência, ao mesmo tempo em que preserva, para o agente que está sendo avaliado, as mesmas propriedades do problema da

seleção de serviços quando considerado em ambientes P2P reais.

Evidenciamos, com a aplicação desta metodologia, que apesar de garantir a lucratividade necessária para manter o interesse dos nós em permanecer no sistema, o desempenho das heurísticas ainda está muito aquém do máximo possível. Além disso, podemos afirmar que, nos cenários estudados, as heurísticas de restrição orçamentária superam quaisquer das demais heurísticas testadas, já que estas encontram uma melhor relação custo-benefício. Além disso, ao introduzirmos um mecanismo de reinício aleatório, conseguimos deiminuir as perdas decorrentes do fenômeno de máximo local.

## 8.2 Trabalhos futuros

Neste trabalho, debruçamo-nos sobre um aspecto bem específico do ambiente, que tem um forte impacto no comportamento das heurísticas: o orçamento dos nós. Com efeito, variando o orçamento médio dos nós pudemos observar como as heurísticas operam quando a concorrência pela oferta dos serviços aumenta ou diminui, e quando há muita ou pouca flexibilidade para a seleção de diferentes portfólios de serviço.

Um dos trabalhos imediatos que ocorrem como conseqüência desta pesquisa é a verificação do comportamento das heurísticas quando submetidas a outras variações de condições ambientais. Pode-se definir experimentos para avaliar como elas se comportam quando a contenção do sistema varia, desde ambientes em que existam muitos nós solicitando serviços e poucos nós oferecendo até ambientes onde o inverso se dê. Pode-se investigar também o impacto que a variação da quantidade total de nós do sistema provoca no desempenho das heurísticas, bem como a variação da quantidade de serviços. Finalmente, pode-se observar o que acontece caso a quantidade de serviços solicitado pelos nós - o tamanho do favor típico - varie.

Considerando o estágio atual desta pesquisa, verificamos que ainda é possível melhorar bastante o desempenho dos métodos heurísticos de seleção de portfolio de serviços, e isto pode ser feito ao nos basearmos em duas estratégias, não necessariamente disjuntas.

Por um lado, algoritmos baseados em aprendizagem de máquina, em que o agente evolui seu conhecimento a respeito do ambiente, encontrando maneiras de melhorar sua performance, podem ser ferramentas úteis na concepção e desenvolvimento de métodos para se-

leção de serviços. As heurísticas de restrição orçamentária que apresentamos aqui podem ser consideradas os primeiros embriões desta investigação.

Da mesma forma, considerando que estamos lidando com um ambiente colaborativo, pretendemos implementar heurísticas associativas, onde os nós trabalharão juntos para encontrar a melhor seleção de serviços de cada um deles, permitindo o uso de alguns métodos de otimização que podem ser melhor implementados em sistemas multi-agentes, como o Enxame de Partículas e os Algoritmos Genéticos.

# Bibliografia

[Almgren and Chriss 2001] Almgren, R. and Chriss, N. (2001). Optimal execution of portfolio transactions.

[Amir et al. 2000] Amir, Y., Awerbuch, B., Barak, A., Borgstrom, R., and Keren, A. (2000). An opportunity cost approach for job assignment in a scalable computing cluster. *Parallel and Distributed Systems, IEEE Transactions on*, 11(7):760–768.

[Andrade et al. 2004] Andrade, N., Brasileiro, F., Cirne, W., and Mowbray, M. (2004). Discouraging free-riding in a peer-to-peer grid. In *HPDC13, the Thirteenth IEEE International Symposium on High-Performance Distributed Computing*.

[Andrade et al. 2007] Andrade, N., Brasileiro, F., Cirne, W., and Mowbray, M. (2007). Automatic grid assembly by promoting collaboration in peer-to-peer grids. *J. Parallel Distrib. Comput.*, 67(8):957–966.

[Andrieux et al. 2004] Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Pruyne, J., Rofrano, J., Tuecke, S., and Xu, M. (2004). Web services agreement specification (ws-agreement). *Global Grid Forum GRAAP-WG, Draft, August*.

[Armbrust et al. 2009] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., Lee, G., Patterson, D. A., Rabkin, A., Stoica, I., and Zaharia, M. (2009). Above the clouds: A berkeley view of cloud computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley.

[Axelrod 1984] Axelrod, R. (1984). *The Evolution of Cooperation*. Basic Books.

[Benkler 2004] Benkler, Y. (2004). Sharing nicely: On shareable goods and the emergence of sharing as a modality of economic production. *Yale Law Journal*, 114(2):273–359.

[Brooke et al. 2000] Brooke, J., Foster, M., Pickles, S., Taylor, K., and Hewitt, W. T. (2000). Mini-grids: Effective test-beds for grid application. In *GRID '00: Proceedings of the First IEEE/ACM International Workshop on Grid Computing*, pages 158–169, London, UK. Springer-Verlag.

[Buyya et al. 2000] Buyya, R., Abramson, D., and Giddy, J. (2000). Nimrod/g: An architecture for a resource management and scheduling system in a global computational grid. *HPC Asia*, 2000.

[Buyya et al. 2002] Buyya, R., Abramson, D., Giddy, J., and Stockinger, H. (2002). Economic models for resource management and scheduling in grid computing. *Concurrency and Computation: Practice and Experience*, 14(13-15):1507–1542.

[Buyya and Vazhkudai 2000] Buyya, R. and Vazhkudai, S. (2000). Compute power market: Towards a market-oriented grid. *The First IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2001)*.

[Calvin and Leung 2003] Calvin, J. M. and Leung, J. Y.-T. (2003). Average-case analysis of a greedy algorithm for the 0/1 knapsack problem. *Oper. Res. Lett.*, 31(3):202–210.

[Campbell and Viceira 1996] Campbell, J. Y. and Viceira, L. M. (1996). Consumption and portfolio decisions when expected returns are time varying. Working Paper 5857, National Bureau of Economic Research.

[Chase et al. 2003] Chase, J., Irwin, D., Grit, L., Moore, J., and Sprenkle, S. (2003). Dynamic virtual clusters in a grid site manager. *High Performance Distributed Computing, 2003. Proceedings. 12th IEEE International Symposium on*, pages 90–100.

[Chekuri and Khanna 2000] Chekuri, C. and Khanna, S. (2000). A ptas for the multiple knapsack problem. In *SODA '00: Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 213–222, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

[Cheung et al. 2004a] Cheung, W. K., Liu, J., Tsang, K. H., and Wong, R. K. (2004a). Dynamic resource selection for service composition in the grid. In *WI '04: Proceedings of*

*the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 412–418, Washington, DC, USA. IEEE Computer Society.

[Cheung et al. 2004b] Cheung, W. K., Liu, J., Tsang, K. H., and Wong, R. K. (2004b). Towards autonomous service composition in a grid environment. In *ICWS '04: Proceedings of the IEEE International Conference on Web Services*, pages 550–557, Washington, DC, USA. IEEE Computer Society.

[Chou et al. 2001] Chou, A., Yang, J., Chelf, B., Hallem, S., and Engler, D. R. (2001). An empirical study of operating system errors. In *Symposium on Operating Systems Principles*, pages 73–88.

[Chun et al. 2003] Chun, B., Fu, Y., and Vahdat, A. (2003). Bootstrapping a distributed computational economy with peer-to-peer bartering. *Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems*.

[Cirne et al. 2006] Cirne, W., Brasileiro, F., Andrade, N., Costa, L., Andrade, A., Novaes, R., and Mowbray, M. (2006). Labs of the world, unite!!! *Journal of Grid Computing*, 4(3):225–246.

[Coêlho and Brasileiro 2009] Coêlho, A. and Brasileiro, F. (2009). On the evaluation of services selection algorithms in multi-services p2p grids. In *Fourth IEEE International Workshop on Business-driven IT Management (BDIM'09)*, pages 52–60, Piscataway, NJ, USA. IEEE Press.

[Coêlho and Brasileiro 2010] Coêlho, A. and Brasileiro, F. (2010). Smarter heuristics for business-driven services selection in multi-services p2p grids. In *The 7th IEEE 2010 International Conference on Services Computing (SCC 2010)*, pages 417–424. IEEE Press.

[Coêlho et al. 2009] Coêlho, A., Brasileiro, F., and Maciel, P. D. (2009). Using heuristics to improve service portfolio selection in p2p grids. In *IM'09: Proceedings of the 11th IFIP/IEEE international conference on Symposium on Integrated Network Management*, pages 438–444, Piscataway, NJ, USA. IEEE Press.

[Coêlho et al. 2008] Coêlho, A., Maciel Jr., P. D., Figueiredo, F. d., Candeia, D., and Brasileiro, F. (2008). On the impact of choice in multi-service p2p grids. In *Third IEEE*

*International Workshop on Business-driven IT Management (BDIM'08)*, pages 98–101, Salvador, Bahia, Brazil.

[Cohen 2003] Cohen, B. (2003). Incentives build robustness in bittorrent. *Workshop on Economics of Peer-to-Peer Systems*, 6:68–72.

[Dantzig 1957] Dantzig, G. (1957). Discrete variable extremum problems. *Operations Res*, 5:266–277.

[Diubin and Korbut 1999] Diubin, G. and Korbut, E. (1999). On the average behaviour of greedy algorithms for the knapsack problem. *Preprint 99-14, Humboldt-Universität zu Berlin, Institut für Mathematik.*

[Duarte et al. 2007] Duarte, A. N., Nyczyk, P., Retico, A., and Vicinanza, D. (2007). Global grid monitoring: the egee/wlcg case. In *GMW '07: Proceedings of the 2007 workshop on Grid monitoring*, pages 9–16, New York, NY, USA. ACM.

[Dudley 1978] Dudley, R. M. (1978). Central limit theorems for empirical measures. *The Annals of Probability*, 6(6):899–929.

[Dustdar and Schreiner 2005] Dustdar, S. and Schreiner, W. (2005). A survey on web services composition. *Int. J. Web Grid Serv.*, 1(1):1–30.

[Eberhart and Kennedy 1995] Eberhart, R. and Kennedy, J. (1995). A new optimizer using particle swarm theory. In *Micro Machine and Human Science, 1995. MHS '95., Proceedings of the Sixth International Symposium on*, pages 39–43.

[Ezziane 2002] Ezziane, Z. (2002). Solving the 0/1 knapsack problem using an adaptive genetic algorithm. *Artif. Intell. Eng. Des. Anal. Manuf.*, 16(1):23–30.

[Farahmand et al. 2005] Farahmand, F., Navathe, S. B., Sharp, G. P., and Enslow, P. H. (2005). A management perspective on risk of security threats to information systems. *Inf. Technol. and Management*, 6(2-3):203–225.

[Fernandez and Gomez 2007] Fernandez, A. and Gomez, S. (2007). Portfolio selection using neural networks. *Computers and Operations Research*, 34:1177.

[Foster 2002] Foster, I. (2002). What is the grid? - a three point checklist. *GRIDtoday*, 1(6).

[Foster et al. 2004] Foster, I., Jennings, N., and Kesselman, C. (2004). Brain meets brawn: why grid and agents need each other. *Autonomous Agents and Multiagent Systems, 2004. AAMAS 2004. Proceedings of the Third International Joint Conference on*, pages 8–15.

[Fu et al. 2003] Fu, Y., Chase, J., Chun, B., Schwab, S., and Vahdat, A. (2003). Sharp: an architecture for secure resource peering. *ACM SIGOPS Operating Systems Review*, 37(5):133–148.

[Goldberg 1989] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional.

[Goldman and Trystram 2004] Goldman, A. and Trystram, D. (2004). An efficient parallel algorithm for solving the knapsack problem on hypercubes. *J. Parallel Distrib. Comput.*, 64(11):1213–1222.

[Grothoff 2003] Grothoff, C. (2003). An excess-based economic model for resource allocation in peer-to-peer networks. *Wirtschaftsinformatik*, 45(3):285–292.

[Gu et al. 2004] Gu, X., Nahrstedt, K., and Yu, B. (2004). Spidernet: An integrated peer-to-peer service composition framework. In *HPDC '04: Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing*, pages 110–119, Washington, DC, USA. IEEE Computer Society.

[Gupta and Somani 2005] Gupta, R. and Somani, A. K. (2005). Game theory as a tool to strategize as well as predict nodes behavior in peer-to-peer networks. In *ICPADS '05: Proceedings of the 11th International Conference on Parallel and Distributed Systems (ICPADS'05)*, pages 244–249, Washington, DC, USA. IEEE Computer Society.

[Holland 1992] Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. The MIT Press.

[Ian 2007] Ian, B. (2007). The egee productions grid.

[Jennings 2000] Jennings, N. R. (2000). Automated haggling: Building artificial negotiators. In *6th Pacific Rim International Conference on Artificial Intelligence*, page 1, 'Melbourne, Australia.

[Karp 1975] Karp, R. M. (1975). On the computational complexity of combinatorial problems. *Networks*, 5:45–68.

[Kellerer et al. 2003] Kellerer, H., Pferschy, U., and Pisinger, D. (2003). *Knapsack Problems*. Springer Verlag, 1st edition edition.

[Kleinberg 2005] Kleinberg, R. (2005). A multiple-choice secretary algorithm with applications to online auctions. In *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 630–631, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

[Lai et al. 2004] Lai, K., Huberman, B., and Fine, L. (2004). Tycoon: A distributed market-based resource allocation system. *Arxiv preprint cs.DC/0404013*.

[Li and Yahyapour 2006a] Li, J. and Yahyapour, R. (2006a). Learning-based negotiation strategies for grid scheduling. *Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)-Volume 00*, pages 576–583.

[Li and Yahyapour 2006b] Li, J. and Yahyapour, R. (2006b). Negotiation strategies for grid scheduling. In *in Proceedings of the First International Conference on Grid and Pervasive Computing, ser. Lecture Notes in Computer Science (LNCS 3947*, pages 567–583. IEEE Press.

[Li et al. 2004] Li, K.-L., Li, R.-F., and Li, Q.-H. (2004). Optimal parallel algorithm for the knapsack problem without memory conflicts. *J. Comput. Sci. Technol.*, 19(6):760–768.

[Magazine et al. 1975] Magazine, M. J., Nemhauser, G. L., and Trotter, L. E., J. (1975). When the greedy solution solves a class of knapsack problems. *OPERATIONS RESEARCH*, 23(2):207–217.

[Marchetti-Spaccamela and Vercellis 1995] Marchetti-Spaccamela, A. and Vercellis, C. (1995). Stochastic on-line knapsack problems. *Math. Program.*, 68(1):73–104.

[Markowitz 1952] Markowitz, H. (1952). Portfolio selection. *The Journal of Finance*, 7(1):77–91.

[Martello et al. 1999] Martello, S., Pisinger, D., and Toth, P. (1999). Dynamic programming and strong bounds for the 0-1 knapsack problem. *Manage. Sci.*, 45(3):414–424.

[McKee 2005] McKee, P. (2005). Grid—the 'white knight' for business? *BT Technology Journal*, 23(3):45–51.

[Mowbray et al. 2006] Mowbray, M., Brasileiro, F., Andrade, N., Santana, J., and Cirne, W. (2006). A reciprocation-based economy for multiple services in peer-to-peer grids. In *P2P'06: Proceedings of the Sixth IEEE International Conference on Peer-to-Peer Computing*, pages 193–202, Washington, DC, USA.

[Nabrzyski et al. 2004] Nabrzyski, J., Schopf, J. M., and Weglarz, J., editors (2004). *Grid resource management: state of the art and future trends*. Kluwer Academic Publishers, Norwell, MA, USA.

[Niar and Freville 1997] Niar, S. and Freville, A. (1997). A parallel tabu search algorithm for the 0-1 multidimensional knapsack problem. In *IPPS '97: Proceedings of the 11th International Symposium on Parallel Processing*, pages 512–516, Washington, DC, USA. IEEE Computer Society.

[Noga and Sarbua 2005] Noga, J. and Sarbua, V. (2005). An online partially fractional knapsack problem. In *ISPAN '05: Proceedings of the 8th International Symposium on Parallel Architectures,Algorithms and Networks*, pages 108–112, Washington, DC, USA. IEEE Computer Society.

[Opitz et al. 2008] Opitz, A., König, H., and Szamlewska, S. (2008). What does grid computing cost? *Journal of Grid Computing*.

[Pinheiro et al. 2007] Pinheiro, E., Weber, W.-D., and Barroso, L. A. (2007). Failure trends in a large disk drive population. In *FAST'07: Proceedings of the 5th conference on USENIX Conference on File and Storage Technologies*, page 2, Berkeley, CA, USA. USENIX Association.

[Pirkwieser et al. 2007] Pirkwieser, S., Raidl, G. R., and Puchinger, J. (2007). Combining lagrangian decomposition with an evolutionary algorithm for the knapsack constrained maximum spanning tree problem. In Cotta, C. and van Hemert, J. I., editors, *EvoCOP*, volume 4446 of *Lecture Notes in Computer Science*, pages 176–187. Springer.

[Russell and Norvig 2003] Russell, S. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach*, chapter Informed Search and Exploration, pages 110–113. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition edition.

[Samuelson 1969] Samuelson, P. A. (1969). Lifetime portfolio selection by dynamic stochastic programming. *The Review of Economics and Statistics*, 51(3):239–246.

[Sarkar et al. 1992] Sarkar, U. K., Chakrabarti, P. P., Ghose, S., and De Sarkar, S. C. (1992). A simple 0.5-bounded greedy algorithm for the 0/1 knapsack problem. *Inf. Process. Lett.*, 42(3):173–177.

[Satsiou and Tassiulas 2007] Satsiou, A. and Tassiulas, L. (2007). A trust-based exchange framework for multiple services in p2p systems. In *P2P '07: Proceedings of the Seventh IEEE International Conference on Peer-to-Peer Computing*, pages 45–52, Washington, DC, USA. IEEE Computer Society.

[Schroeder and Gibson 2007] Schroeder, B. and Gibson, G. A. (2007). Disk failures in the real world: what does an mttf of 1,000,000 hours mean to you? In *FAST '07: Proceedings of the 5th USENIX conference on File and Storage Technologies*, Berkeley, CA, USA. USENIX Association.

[Selcuk et al. 2004] Selcuk, A. A., Uzun, E., and Pariente, M. R. (2004). A reputation-based trust management system for p2p networks. In *CCGRID '04: Proceedings of the 2004 IEEE International Symposium on Cluster Computing and the Grid*, pages 251–258, Washington, DC, USA. IEEE Computer Society.

[Shneidman et al. 2005] Shneidman, J., Ng, C., Parkes, D. C., AuYoung, A., Snoeren, A. C., Vahdat, A., and Chun, B. (2005). Why markets could (but don't currently) solve resource allocation problems in systems. In *Tenth Workshop on Hot Topics in Operating Systems*

*(HotOS X)*, Santa Fe, NM, USA. USENIX, the Advanced Computing Systems Association.

[Simões and Costa 2001] Simões, A. and Costa, E. (2001). An evolutionary approach to the zero/one knapsack problem: Testing ideas from biology. In *In Proceeding of Fifth International Conference on Aritical Neurakl Network and Genetic Algorithms*.

[Singh 2003] Singh, A. (2003). Trustme: Anonymous management of trust relationships in decentralized p2p systems. In *In IEEE Intl. Conf. on Peer-to-Peer Computing*, pages 142–149.

[Smith 1980] Smith, R. (1980). The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, 29(12):1104–1113.

[Smolinski 2000] Smolinski, B. A. (2000). Approximating the 0-1 multiple knapsack problem with agent decomposition and market negotiation. In *IEA/AIE '00: Proceedings of the 13th international conference on Industrial and engineering applications of artificial intelligence and expert systems*, pages 296–305, Secaucus, NJ, USA. Springer-Verlag New York, Inc.

[Tanaka et al. 2000] Tanaka, H., Guo, P., and Türksen, I. B. (2000). Portfolio selection based on fuzzy probabilities and possibility distributions. *Fuzzy Sets Syst.*, 111(3):387–397.

[Wang and Vassileva 2007] Wang, Y. and Vassileva, J. (2007). Toward trust and reputation based web service selection: A survey. *International Transactions on Systems Science and Applications*, pages 118–132.

[Wu et al. 2001] Wu, J., Yunfei, L., and Schröder, H. (2001). A minimal reduction approach for the collapsing knapsack problem. *Computers and Artificial Intelligence*, 20(4).

[Xia et al. 2000] Xia, Y., Liu, B., Wang, S., and Lai, K. K. (2000). A model for portfolio selection with order of expected returns. *Comput. Oper. Res.*, 27(5):409–422.

[Xu et al. 2007] Xu, F., Chen, W., and Yang, L. (2007). Improved particle swarm optimization for realistic portfolio selection. In *SNPD '07: Proceedings of the Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and*

*Parallel/Distributed Computing (SNPD 2007)*, pages 185–190, Washington, DC, USA. IEEE Computer Society.

[Zhang et al. 2006] Zhang, W.-G., CHEN, W., and Wang, Y.-L. (2006). The adaptive genetic algorithms for portfolio selection problem. *IJCSNS International Journal of Computer Science and Network Security*, 6(1):196–200.

[Zhou et al. 2008] Zhou, Y., Chakrabarty, D., and Lukose, R. (2008). Budget constrained bidding in keyword auctions and online knapsack problems. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 1243–1244, New York, NY, USA. ACM.

[Zhou and Naroditskiy 2008] Zhou, Y. and Naroditskiy, V. (2008). Algorithm for stochastic multiple-choice knapsack problem and application to keywords bidding. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 1175–1176, New York, NY, USA. ACM.