UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

SANDRA SATYKO GUIMARÃES WATANABE

Abordagem de Teoria dos Jogos Evolucionários para Modelagem de Aplicações de *Live Streaming* em Redes *Peer-to-Peer*

Dissertação apresentada como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Prof[a]. Dr[a]. Ingrid Jansch-Pôrto
Orientadora

Porto Alegre, fevereiro de 2010

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Watanabe, Sandra Satyko Guimarães

Abordagem de Teoria dos Jogos Evolucionários para Modelagem de Aplicações de *Live Streaming* em Redes *Peer-to-Peer* / Sandra Satyko Guimarães Watanabe. – Porto Alegre: PPGC da UFRGS, 2010.

98 p.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2010. Orientadora: Ingrid Jansch-Pôrto.

1. Redes *peer-to-peer*. 2.

Live streaming. 3. Teoria dos Jogos Evolucionários. 4. Tolerância a Falhas. I. Jansch-Pôrto, Ingrid. II. Título.

“Semeiem paz, amor e bondade; lutem por um ideal superior e nobre. O futuro está em suas mãos! E não se esqueçam nunca: como é em cima é embaixo.”

— LUCI GUIMARÃES WATANABE

AGRADECIMENTOS

Agradeço à minha irmã Scylla (lê-se “Sila”), por ter sido a primeira incentivadora deste trabalho, ao meu pai, Takaharu Watanabe, e ao meu querido noivo, Moacir, sem o qual a conclusão deste trabalho não seria possível.

Agradeço também ao Professor Roberto da Silva, pelo valioso auxílio de última hora, à Professora Ingrid, pela orientação do trabalho, e ao apoio financeiro da Empresa Brasileira de Pesquisa Agropecuária (Embrapa).

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

P2P	*Peer-to-Peer*
TJ	Teoria de Jogos
TJE	Teoria dos Jogos Evolucionários
ESS	Estratégia Evolucionariamente Estável
IP	*Internet Protocol*
ISP	*Internet Service Provider*
CDN	*Content Delivery Network*
IDE	*Integrated Development Environment*

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Existe um interesse crescente do mercado por aplicações de multimídia em *streaming* via rede. Particularmente, as aplicações de *live streaming* que utilizam a tecnologia de redes P2P para a disseminação de conteúdo têm sido alvo de grande atenção. Aplicações como PPLive e PPStream provam que as aplicações de *live streaming* em redes P2P são uma realidade com relação à tecnologia atual.

Os sistemas de *live streaming* fornecem um serviço de *multicast* no nível de aplicação para transmissões ao vivo na *Internet*. Essas aplicações de *live streaming*, quando executadas em redes P2P, têm potencial para serem altamente robustas, escaláveis e adaptativas devido à redundância e não dependência de recursos particulares dentre os nodos participantes. Porém, para fazer uso de todas as vantagens disponíveis, a aplicação deve contornar alguns desafios: i) manter a qualidade de *playback* mesmo com a inerente dinamicidade das redes P2P; ii) impedir que nodos incorretos escondam ações maliciosas atrás do anonimato que existe em P2P; iii) manter a taxa de *upload* dos nodos participantes da aplicação em um nível aceitável. A taxa de *upload* dos nodos é muito importante porque a aplicação de *live streaming* em P2P é uma aplicação cooperativa. Desta forma, espera-se que todo novo usuário ajude a aplicação retransmitindo pacotes para outros usuários, mantendo, desta forma, a capacidade global de *upload* do sistema. Infelizmente, manter a cooperação em *live streaming* não é uma tarefa trivial, visto que cada nodo enfrenta o dilema social do interesse próprio (individualmente é melhor explorar a cooperação dos outros usuários sem reciprocidade) versus a cooperação para com o grupo.

A principal contribuição deste trabalho consiste na apresentação de um modelo matemático baseado em Teoria dos Jogos Evolucionários, cujo objetivo é ajudar a compreender as aplicações de *live streaming* em redes P2P e os fatores que influenciam o seu correto funcionamento. Como contribuição secundária, este trabalho fornece uma análise estatística do comportamento do *download* e *upload* observado nestas aplicações. A análise estatística mostra que existe um decaimento da variância temporal de *download* e *upload* nas aplicações de *live streaming*, e que tal decaimento segue uma lei de potência. Os resultados evolucionários do modelo indicam que, se a queda do índice de satisfação dos usuários com a taxa de *download* for suave, e se a redução da satisfação devido ao custo de *upload* for insignificante, então existe um ambiente propício para que a cooperação entre os nodos cresça. De forma inversa, se a queda do índice de satisfação dos usuários com a taxa de *download* for abrupta, e a redução da satisfação devido ao custo de *upload* for significativa, então existe um ambiente propício para proliferação de nodos oportunistas.

A realização e descrição desta pesquisa é composta de quatro etapas principais: i) a delimitação do cenário de *live streaming* e a definição do jogo para modelagem; ii) a definição do conjunto de estratégias e da função de utilidade; iii) a criação do modelo; iv) a análise do modelo e a apresentação dos resultados de simulação. A análise do modelo

abrange três fases: i) análise estatística e comparação das características de *download* e *upload* dos dois simuladores utilizados; ii) avaliação do modelo de Teoria dos Jogos Evolucionários através de simulações; e iii) análise dos resultados evolucionários gerados pelo simulador de Teoria dos Jogos Evolucionários.

Palavras-chave: Redes *peer-to-peer*, *Live streaming*, Teoria dos Jogos Evolucionários, Tolerância a Falhas.

Evolutionary Game Theory Approach for Modeling Live Streaming Applications over Peer-to-Peer Networks

ABSTRACT

There is a growing interest in the market for networked multimedia applications. Live streaming applications that use the technology of P2P networks for distribution of live content have specially been the subject of great attention. Applications such as PPLive and PPStrem demonstrate that P2P live streaming applications are already possible with our present technology.

Live streaming systems provide a multicast service in the application level for live broadcasts to users through the Internet. These systems executing in P2P networks have the potential to be highly robust, scalable and adaptive due to the characteristics of these scenarios. However, to take advantage of these potential properties, they must overcome some challenges: i) to maintain the playback quality even with the inherent dynamics of P2P networks; ii) to prevent that incorrect peers hide malicious behavior behind their anonymity; iii) to maintain the upload contribution of peers at acceptable levels. The upload contribution of peers is highly important because live streaming applications are cooperative applications. Therefore, every new user must help the application forwarding packets to other users, thereby maintaining the global upload capacity of the system. Unfortunately, the maintenance of cooperation in live streaming system is not a trivial task, since each node faces the social dilemma of self-interest (individually is always better to explore the cooperation of other users without reciprocity) versus cooperation to the group.

The main contribution of this dissertation is the presentation of a mathematical model based on Evolutionary Game Theory, whose goal is to help understanding live streaming P2P applications and the factors that influence their correct operation. As a secondary contribution, this work provides a statistical analysis of download and upload behaviors of peers in live streaming P2P systems. The statistical analysis indicates that there is a decay in the download and upload variances, and that this decay follows a power law. The evolutionary results of the model indicate that, if the satisfaction of users with the download rate is smooth, and the reduction of satisfaction due to the upload cost is negligible, then there is a favorable environment for the growth of cooperation. Conversely, if the satisfaction of users with the download rate is abrupt, and the reduction of satisfaction due to the upload cost is significant, then there is a favorable environment to the proliferation of opportunistic nodes.

The realization and description of this research is composed of four main steps: i) the definition of the live streaming scenario and the definition of the game to model this scenario; ii) the definition of the strategy set and of the utility function; iii) the suggestion of a model; iv) the analysis of the proposed model and the presentation of obtained results. The model analysis comprehends three phases: i) the statistical analysis and the comparison of the characteristics of download and upload of the two simulators used in

this work; ii) the evaluation of the Evolutionary Game Theory model through simulation; and iii) the analysis of the results generated by the Evolutionary Game Theory simulator.

Keywords: Peer-to-peer networks, Live streaming, Evolutionary Game Theory, Fault Tolerance.

1 INTRODUÇÃO E MOTIVAÇÃO

Existe um interesse crescente do mercado por aplicações de multimídia em *streaming* via rede, tais como vídeo a partir de mídias armazenadas e *live streaming*. Dentre as técnicas existentes para a entrega do serviço destas aplicações, a mais simples é a cliente-servidor. Esta técnica consiste em alocar recursos de servidor e de rede para cada requisição de cliente. Porém, devido a fatores como grande carga no servidor, largura de banda limitada e conexões instáveis, esta arquitetura não escala bem. O incremento da quantidade de clientes participantes do sistema tem como consequência a degradação do desempenho da aplicação (YIU; JIN; CHAN, 2007; LI; YIN, 2007). Desta forma, a arquitetura cliente-servidor mostra-se ineficiente para manter a qualidade desejada de uma aplicação multimídia na *Internet*. Uma outra técnica para a entrega do serviço é utilizar IP *multicast*. O IP *multicast* estende o modelo *unicast* com transmissões eficientes de pacotes multi-ponto (LI; YIN, 2007). Porém, esta técnica tem o problema de necessitar de suporte de *hardware* especial para sua concretização. Para a rede IP *multicast* se tornar funcional, os roteadores atuais da *Internet* devem ser substituídos por roteadores *multicast-capable* de larga escala, o que envolve custos elevados de configuração de infra-estrutura e administração (WEN; LONGSHE; QIANG, 2006). Assim, a arquitetura *peer-to-peer* (P2P), já bastante popular para o compartilhamento de arquivos, vem sendo utilizada como alternativa para a distribuição de conteúdo multimídia na *Internet*.

Aplicações de *live streaming* em redes *peer-to-peer* (P2P) vêm se tornando bastante populares na *Internet* nos últimos anos. Exemplos como Coolstreaming (2005), PPLive (2007) e PPStream (2005) provam que as aplicações de *live streaming* em redes P2P não são apenas uma possibilidade, mas uma realidade com relação à tecnologia atual. Esses sistemas fornecem serviços de *multicast* no nível de aplicação para transmissões ao vivo para milhares de usuários. Ou seja, *live streaming* refere-se à distribuição sincronizada de conteúdo multimídia para vários usuários, e idealmente estes usuários querem receber a transmissão sem muito atraso em relação à transmissão original. É importante salientar que o conteúdo pode ser realmente ao vivo ou previamente gravado (PADMANABHAN et al., 2002). Um conteúdo realmente "ao vivo" refere-se, por exemplo, a uma partida de futebol que é transmitida no momento em que o jogo é realizado, e previamente gravado refere-se, por exemplo, a um episódio de seriado de televisão gravado pela emissora de TV, mas transmitido apenas em um horário específico. Mas em qualquer dos casos, os dados são gerados pouco antes do momento da transmissão pela origem dos dados, não estando disponíveis antecipadamente. Por ser uma transmissão "ao vivo", as operações disponíveis para mídias armazenadas, as funções do VCR, não estão disponíveis para *live streaming*.

Os sistemas *peer-to-peer* têm vantagens em relação a outras tecnologias. Por exemplo, como os sistemas P2P estão baseados na contribuição voluntária de recursos de cada

participante, o serviço de *live streaming* sobre esse paradigma potencialmente pode ser altamente robusto, escalável e adaptativo. Além disso, sistemas P2P podem ser preparados para absorver o impacto de uma rajada inesperada de usuários acessando o serviço. Outra vantagem de se utilizar um serviço de *multicast* em redes P2P é que, visando melhorar o desempenho da aplicação, ele ainda poderá ser combinado com o IP *multicast* quando este se tornar funcional. Com o IP *multicast*, em princípio, é possível disseminar simultaneamente os pacotes para um conjunto de destinos, atravessando cada enlace apenas uma vez, sem duplicação. Alguns estudos vêm sendo realizados no sentido de combinar serviços de *multicast* no nível de aplicação e IP *multicast* (OH-ISHI et al., 2003; JIN; CHENG; GARY CHAN, 2006; BROGLE; MILIC; BRAUN, 2008). Assim, as aplicações de *live streaming* em redes P2P são alvo de estudos por virem se configurando como a grande promessa de serviços de *multicast* baratos na *Internet*, visto que estas não precisam de uma infra-estrutura grande para sua realização, como acontece com as arquiteturas cliente-servidor e CDN.

Apesar dessas vantagens, uma aplicação de *live streaming* sobre uma rede P2P enfrenta alguns desafios. Primeiro, as redes *peer-to-peer* são inerentemente dinâmicas. Os nodos podem sofrer colapso ou desconexão involuntária a qualquer momento e também entrar e sair da rede quando quiserem. Fatores determinantes desse dinamismo são erros em aplicativos e no software de suporte, instabilidades da rede ou decisões dos usuários. Segundo, nos sistemas P2P existe o dilema social do interesse próprio versus a cooperação. Do ponto de vista do grupo, os resultados ótimos são obtidos quando todos os nodos cooperam e contribuem com os seus próprios recursos para a aplicação, mas individualmente é sempre melhor explorar a cooperação dos outros sem reciprocidade. Estes nodos egoístas tentam permanecer no sistema evitando partilhar seus recursos com os outros nodos. Terceiro, as interações entre os nodos são anônimas. Nodos incorretos podem tentar esconder ações maliciosas atrás do anonimato. Assim, devido aos requisitos de desempenho estritos dos serviços de *live streaming*, é necessário assegurar que o *throughput* dos pacotes seja confiável e estável apesar da presença de nodos incorretos. São considerados nodos corretos aqueles que executam o protocolo definido conforme fornecido originalmente pela aplicação. Nodos incorretos incluem aqueles que alteram o protocolo devido a perfis egoístas, defeituosos ou maliciosos.

Baseado nos estudos de Aiyer et al. (2005) e Haridasan, Jansch-Pôrto e van Renesse (2008), este trabalho classifica os nodos das aplicações de *live streaming* em redes P2P em três categorias: i) corretos - nodos cooperativos que executam o protocolo definido pela aplicação; ii) oportunistas - nodos que são egoístas e que se desviam do protocolo definido com o único propósito de economizar largura de banda de *upload*; iii) bizantino - nodos defeituosos ou maliciosos que se desviam arbitrariamente do protocolo. Dada a heterogeneidade de usuários na *Internet*, não é possível antecipar todas a formas de desvio de protocolo dos nodos incorretos. Existem muitos tipos diferentes de usuários por detrás dos nodos, e alguns podem ter mais habilidade e conhecimento que outros. Para entender o impacto que os nodos incorretos têm na aplicação de *live streaming* é necessário um modelo flexível o suficiente para acomodar o maior número possível de ataques e comportamentos.

A Teoria de Jogos (TJ) tem sido utilizada para modelar aplicações em redes P2P, e também para propor técnicas de incentivo à cooperação nessas aplicações. Alguns exemplos de trabalhos que combinam Teoria de Jogos e redes P2P são Feldman et al. (2006; 2004), Martin (2007), Shneidman, Parkes e Massouli (2004) e Oualha e Roudier (2008). Uma grande parte dos estudos existentes é direcionada para aplicações de compartilha-

mento de arquivos, que foram as aplicações que popularizaram as redes P2P. Além disto, os estudos realizados até o momento se preocupam grandemente em modelar somente o comportamento de nodos oportunistas, sendo que pouco trabalho tem sido realizado para modelar também o comportamento de nodos bizantinos em redes P2P. Dentre os trabalhos existentes que estudam o comportamento de nodos bizantinos, destacam-se os trabalhos de Aiyer et al. (2005) e Li et al. (2006). A Teoria dos Jogos Evolucionários (TJE) modela grandes populações de indivíduos onde cada um pode adotar uma estratégia específica. Nesta abordagem, o conceito de racionalidade [1] da Teoria de Jogos é substituído pelo conceito mais fraco de adaptabilidade ou sucesso reprodutivo: estratégias que são mais bem sucedidas em média serão utilizadas com mais frequência e irão prevalecer no final. O jogo descreve os ganhos que resultam quando estes indivíduos se encontram. A dinâmica do jogo é baseada na suposição de que cada estratégia é jogada por uma certa fração da população (a frequência da estratégia). Então, dada a distribuição das estratégias, indivíduos com o melhor ganho médio serão mais bem sucedidos que outros, e a sua proporção na população irá crescer com o tempo. Esta dinâmica, após várias rodadas do jogo, pode definir quais estratégias são melhores que outras, e em muitos casos o processo dinâmico pode evoluir para um equilíbrio (TUROCY; STENGEL, 2001). Assim, a Teoria dos Jogos Evolucionários parece ser mais apropriada que a Teoria de Jogos (TJ) para modelar aplicações de *live streaming* em redes P2P por duas razões: primeira, porque a TJE foi projetada para modelar grandes populações de indivíduos, e segunda, porque não assume que todos os jogadores são altamente sofisticados e racionais - tal suposição de alto nível de racionalidade é frequentemente irreal.

Apesar de existirem estudos e modelos para compreender aplicações de compartilhamento de arquivos e outros tipos de aplicações em redes P2P, as aplicações de *live streaming* possuem características próprias (por exemplo, requisitos estritos de tempo) que as diferenciam de outras aplicações. Desta forma, os modelos utilizados em outras aplicações não podem ser diretamente utilizados para modelar as aplicações de *live streaming*. Para manterem o serviço de maneira satisfatória, as aplicações de *live streaming* em redes P2P necessitam de uma cuidadosa elaboração durante a fase de projeto, e monitoramento constante quando estiverem implantadas, ou seja, quando as aplicações estiverem em pleno uso pelos usuários. Neste contexto, compreender como estas aplicações se comportam em determinadas situações é muito importante. Assim, neste trabalho, uma nova abordagem para modelar usuários de sistemas de *live streaming* em redes P2P é proposta como ponto de partida para o estudo destes sistemas. O objetivo do presente trabalho é investigar o cenário (bem definido) de sistemas de *live streaming* em redes P2P utilizando a Teoria dos Jogos Evolucionários, e até o momento de conclusão desta dissertação, acredita-se que este é o primeiro estudo a aplicar os conceitos de Teoria de Jogos Evolucionários a aplicações de *live streaming* em redes P2P. Assim, o presente trabalho difere-se de outros realizados até o momento por três motivos: i) este trabalho, ao invés de utilizar a Teoria de Jogos Clássica, utiliza a Teoria dos Jogos Evolucionários, que parece ser mais adequada para modelar o comportamento dos usuários no contexto de redes P2P; ii) o modelo proposto neste estudo, apesar de não modelar o comportamento de usuários bizantinos no presente momento, pode ser estendido em trabalhos futuros para acomodar o comportamento de usuários bizantinos e oportunistas em aplicações de *live streaming* em redes P2P; iii) este estudo é direcionado para a modelagem de aplicações de *live streaming* em redes P2P. Assim, a principal contribuição deste trabalho consiste

[1] A palavra "racionalidade" pode ser associado à idéia de "indivíduo estrategista". Vide seção 2.1 para mais detalhes sobre o conceito de racional/racionalidade em Teoria de Jogos.

na apresentação de um modelo matemático baseado em Teoria dos Jogos Evolucionários, cujo objetivo é ajudar a compreender as aplicações de *live streaming* em redes P2P e os fatores que influenciam o seu correto funcionamento. Através do modelo é possível predizer a evolução de uma aplicação de *live streaming* em redes P2P sob diferentes situações de mau comportamento dos nodos. Adicionalmente, este trabalho abre a possibilidade de avaliar o impacto no sistema pela ação dos nodos que se comportam incorretamente. A análise do modelo proposto é realizada através de simulações evolutivas. Como contribuição secundária, este trabalho fornece uma análise estatística do comportamento do *download* e *upload* observado em aplicações de *live streaming*.

Através da análise estatística efetuada a partir de um simulador de *live streaming* é possível observar que existe um decaimento da variância temporal de *download* e *upload* nas aplicações de *live streaming* em redes P2P com 1.000 nodos, e que tal decaimento segue uma lei de potência, tal que:

$$var(down)(t) \sim t^{-0,161(2)} \;\; e \;\; var(up)(t) \sim t^{-0,164(2)}$$

É possível observar também que o número de nodos na rede tem um efeito significativo no expoente da lei de potência que rege a variância temporal do *download* e do *upload*. De acordo com os resultados obtidos, o expoente da lei de potência da variância temporal de *upload* é máximo na maior rede analisada (1.000 nodos), e é mínimo na menor rede (125 nodos). Assim, a tendência do expoente do *upload* é crescer juntamente com o tamanho da rede. Porém, o expoente da lei de potência da variância temporal de *download* tem um comportamento inverso. O expoente é mínimo na maior rede (1.000 nodos), e é máximo em uma rede de tamanho intermediário (rede com 250 nodos). Os resultados evolucionários do modelo indicam que se a percepção dos usuários em relação à queda do índice de satisfação com a taxa de *download* for suave, e a percepção de redução da satisfação devido ao custo de *upload* for insignificante, então existe um ambiente propício para que a cooperação entre os participantes cresça. De forma inversa, se a percepção dos usuários em relação à queda do índice de satisfação com a taxa de *download* tiver uma inflexão abrupta, e a percepção de redução da satisfação devido ao custo de *upload* for significativa, então existe um ambiente propício para proliferação de nodos oportunistas. Também foi estudada a eficiência de um sistema de auditoria para conter a proliferação de nodos oportunistas. Com o sistema de auditoria atuando, alguns dos cenários que originalmente incentivam a proliferação de nodos oportunistas podem, de maneira inversa, incentivar a proliferação de nodos corretos, podendo, inclusive, convergir para um estado estacionário com somente nodos corretos na aplicação.

A realização e descrição desta pesquisa é composta de quatro etapas principais: i) a delimitação do cenário de *live streaming* e a definição do jogo para modelagem; ii) a definição do conjunto de estratégias e da função de utilidade; iii) a criação do modelo; iv) a análise do modelo e a apresentação dos resultados de simulação. A análise do modelo abrange três fases: i) análise estatística e comparação das características de *download* e *upload* dos dois simuladores utilizados; ii) validação do simulador de Teoria dos Jogos Evolucionários; e iii) análise dos resultados evolucionários gerados pelo simulador de TJE.

O restante deste trabalho está organizado de acordo com a estrutura explicada a seguir.

- O capítulo 2 faz uma introdução a: Teoria de Jogos, Teoria dos Jogos Evolucionários, redes *peer-to-peer* e *live streaming*. O conteúdo deste capítulo é des-

tinado a deixar o leitor familiarizado com os conceitos importantes para o entendimento do resto deste trabalho.

- Os trabalhos que mais se assemelham ao proposto nesta dissertação e que serviram de inspiração neste estudo são descritos no capítulo 3.

- O capítulo 4 apresenta em detalhes o cenário de *live streaming* a ser modelado. O objetivo principal deste capítulo é mostrar o "mundo real", ou seja, o cenário do qual o modelo proposto deseja criar uma abstração.

- O capítulo 5 apresenta uma análise estatística focada nas propriedades de *download* e *upload* do simulador de *live streaming* utilizado neste trabalho. O objetivo deste capítulo é obter parâmetros de comparação entre as características de *download* e *upload* de uma aplicação de *live streaming* e as características de *download* e *upload* obtidas do simulador de Teoria dos Jogos Evolucionários apresentado no capítulo 7.

- O capítulo 6 apresenta o modelo matemático baseado em Teoria dos Jogos Evolucionários. Inicialmente, delimita-se o escopo do modelo, tecendo as considerações acerca do cenário de *live streaming* que será modelado. Em seguida, é apresentado informalmente, com o auxílio de um pseudo-código, o jogo utilizado para capturar a interação entre os nodos em *live streaming*. Após a descrição do conjunto de estratégias consideradas neste trabalho, é apresentado formalmente o modelo e definidas a função de utilidade e a dinâmica da população.

- O capítulo 7 descreve o simulador de Teoria dos Jogos Evolucionários utilizado neste trabalho e apresenta os resultados obtidos a partir das simulações.

- O capítulo 8 revisa as principais conclusões obtidas a partir deste trabalho e propõe extensões que poderão ser desenvolvidas como trabalhos futuros.

2 CONCEITOS BÁSICOS

Este capítulo é dedicado a fornecer conceitos básicos de Teoria de Jogos, Teoria dos Jogos Evolucionários, redes *peer-to-peer* e *live streaming* visando facilitar o entendimento do restante deste trabalho aos leitores não familiarizados com alguma das áreas envolvidas.

2.1 Teoria de Jogos

A Teoria de Jogos (TJ) foi criada por John von Neumann e Oskar Morgenstern em 1944 em seu trabalho "The Theory of Games and Economic Behavior" (NEUMANN; MORGENSTERN, 1944). O trabalho de von Neumann e Morgenstern foi o primeiro a tentar capturar matematicamente o comportamento em situações estratégicas na qual o sucesso de um indivíduo em fazer escolhas depende das escolhas dos outros indivíduos (situações estratégicas de conflito e cooperação). As limitações na estrutura matemática inicialmente fizeram a teoria de von Neumann e Morgenstern aplicável apenas sob condições especiais e restritas. Esta situação tem mudado gradualmente ao longo dos anos, à medida que a estrutura original é aprofundada e generalizada para aumentar a sua aplicabilidade em outras ciências (ROSS, 2008). Atualmente, a TJ possui aplicações em Ciências Sociais, Biologia, Ciências Políticas, Ciência da Computação, dentre outras.

A **Teoria de Jogos** é o estudo formal da tomada de decisões onde vários agentes devem fazer escolhas que potencialmente afetam os interesses dos outros agentes. A TJ tenta capturar matematicamente o comportamento de agentes em situações estratégicas de conflito e cooperação, e os conceitos de TJ aplicam-se sempre que as ações de vários agentes são interdependentes. Os agentes podem ser indivíduos, grupos, firmas ou qualquer combinação destes. Os conceitos de TJ fornecem uma linguagem para formular, estruturar, analisar e entender cenários estratégicos (TUROCY; STENGEL, 2001).

No contexto de TJ, um **jogo** é uma descrição formal de uma situação estratégica (TUROCY; STENGEL, 2001). São as situações na qual pelo menos um agente somente pode agir para maximizar sua utilidade antecipando as respostas que serão fornecidas por um ou mais agentes em sequência a suas ações.

Um fato é considerado **senso comum** se todos os jogadores o conhecem, e têm consciência que todos os outros jogadores sabem disso. A **estrutura do jogo** é frequentemente assumida como sendo senso comum entre os jogadores.

Um **jogador** (ou agente) é uma entidade com preferências e que toma decisões em um jogo. Um jogador chamado de **racional** é um jogador que pode: i) estimar os resultados; ii) calcular os caminhos para os resultados; e iii) escolher ações que produzem os resultados preferidos desejados, dadas as ações dos outros jogadores. A Teoria de Jogos considera que a racionalidade pode, em alguns casos, ser internamente computada

pelo jogador, o que significa que não implica necessariamente em um ato deliberado ou consciente (ROSS, 2008). Frequentemente é assumido que a racionalidade de todos os jogadores é senso comum (TUROCY; STENGEL, 2001). Esta definição de racionalidade da TJ é um dos grandes pontos de crítica da Teoria de Jogos, visto que demanda um alto grau de compreensão (consciente ou inconsciente) do jogo, da sua estrutura e das preferências dos outros jogadores.

A **utilidade**, também chamada de **ganho** (em inglês, *payoff*), é um conceito abstrato que se refere à quantidade de "bem-estar" que um jogador obtém de um objeto ou evento. "Bem-estar" refere-se a um índice padronizado de valores relativos a satisfação, justificado pela referência a alguma experiência estruturada. Assim, o ganho é um número que reflete o desejo de um jogador por um resultado, e incorpora a atitude do jogador em relação ao risco. O ganho não deve ser necessariamente interpretado como um valor monetário. A **função de utilidade** é uma função que mapeia as preferências do jogador para os números do ganho. A função de utilidade é um artifício matemático para transformar as preferências ordenadas do jogador em números reais (ROSS, 2008; TUROCY; STENGEL, 2001).

Cada participante em um jogo encara uma escolha entre duas ou mais ações. Uma **estratégia** é um "programa de jogo" que diz ao jogador quais ações tomar em resposta a cada possível ação que os outros jogadores possam ter (ROSS, 2008). A estratégia determina completamente o comportamento do jogador. O conceito de estratégia é algumas vezes erroneamente confundido com um movimento. Um movimento é uma ação tomada por um jogador em algum ponto durante o jogo. Uma estratégia é um algoritmo completo para jogar o jogo, dizendo ao jogador o que fazer em cada possível situação. Assim, uma estratégia define um comportamento em particular, ou um conjunto de comportamentos, que um jogador manifesta. Uma estratégia pode ser **pura**, quando ela não é definida em termos de outras estratégias presentes no jogo, ou **mista**, quando uma estratégia é uma mistura de estratégias puras existentes (PRESTWICH, 1999). A estratégia mista é uma randomização (com dadas probabilidades) das estratégias puras disponíveis (TUROCY; STENGEL, 2001). Desta maneira, tem-se uma estratégia pura se, em uma estratégia mista, uma estratégia pura em particular é selecionada com probabilidade 1 e cada uma das outras estratégias com probabilidade 0 (zero). O **conjunto de estratégias** de um jogador é o conjunto de estratégias puras disponíveis para aquele jogador.

O nível de informação que cada usuário possui é crucial durante a análise do jogo. Um jogador pode ter **informação perfeita** e/ou **informação completa** sobre o jogo. Um jogador possui informação perfeita se todos os jogadores conhecem todos os movimentos realizados previamente por todos os outros jogadores. Os jogos de informação perfeita são os jogos mais simples em termos de perspectivas da estrutura lógica, visto que o jogador sabe tudo que aconteceu no jogo até aquele momento, e em cada ponto a estratégia do jogador diz a ele qual ação tomar (ROSS, 2008). Os jogos de tabuleiro, como o Xadrez, são tipicamente jogos de informação perfeita. A **informação completa** requer que cada jogador conheça as estratégias e ganhos de cada um dos outros jogadores, mas não as suas ações. Por exemplo, um jogador pode ter informação completa do jogo Dilema do Prisioneiro, mas ainda assim neste jogo a informação é imperfeita, já que ninguém tem conhecimento das ações dos outros jogadores que são relevantes para a tomada de suas escolhas. O Dilema do Prisioneiro é o mais famoso jogo da Teoria de Jogos. Jogos com a estrutura do Dilema do Prisioneiro foram desenvolvidos e discutidos por Merrill Flood e Melvin Dresher em 1950. O título Dilema do Prisioneiro e a versão com sentenças de prisão como ganho são devidos a Albert Tucker, que queria deixar as idéias de Flood e

Dresher mais acessíveis para uma audiência de psicólogos de Stanford (KUHN, 2009).

A diferença entre os jogos de informação perfeita e imperfeita é relacionada de maneira aproximada à distinção entre as maneiras de representar os jogos. Os jogos podem ser de **movimento sequencial** ou **movimento simultâneo**. É natural pensar em jogos de movimento sequencial como sendo aqueles cujos jogadores escolhem suas estratégias uma após a outra, e nos jogos de movimento simultâneo como aqueles cujos jogadores escolhem suas estratégias ao mesmo tempo. Isto não é exatamente correto, pois o que é de importância estratégica não é a ordem temporal dos eventos em si, mas se e quando os jogadores sabem sobre as ações dos outros jogadores em relação ao momento em que estes têm que fazer a sua própria escolha. Por exemplo, se dois competidores de negócio estão planejando campanhas de marketing, um pode executar a sua estratégia meses antes do outro, mas se nenhum sabe o que o outro executou ou irá executar quando eles tomam as suas decisões, então este é um jogo de movimento simultâneo (ROSS, 2008).

Foi afirmado anteriormente que os jogos de informação perfeita são os tipos de jogos mais simples logicamente. Isto é verdade porque, em tais jogos, contanto que o jogo termine após um número conhecido de ações, os jogadores podem usar um procedimento direto para predizer o resultado. Um jogador racional em tal jogo escolhe a sua primeira ação considerando cada série de respostas e contra-respostas que irão resultar de cada ação. Ele então se pergunta qual dos resultados finais disponíveis traz a ele o maior ganho, e escolhe a ação que inicia a cadeia de eventos que leva a este resultado. Este processo é chamado **indução de trás-para-frente** (do inglês, *backward induction*), porque o raciocínio começa a partir do resultado e se estende até o problema de decisão presente (ROSS, 2008).

Os dois tipos de objetos matemáticos utilizados para representar jogos são árvores e matrizes. A **árvore do jogo** é um conjunto de nodos conectados que tem uma direção. Pode-se desenhar as árvores de cima para baixo, ou da esquerda para a direita. No primeiro caso, os nodos no topo são interpretados como tendo precedência na sequência de ações. No caso de uma árvore desenhada da esquerda para a direita, o nodo mais à esquerda tem precedência em relação aos nodos da direita. A representação de jogos por árvores é utilizada para dar suporte ao raciocínio de indução de trás-para-frente, já que a árvore do jogo mostra a ordem na qual as ações são tomadas pelos jogadores (ROSS, 2008).

O jogos que utilizam árvores para sua representação são normalmente chamados de **jogos na forma extensiva**.

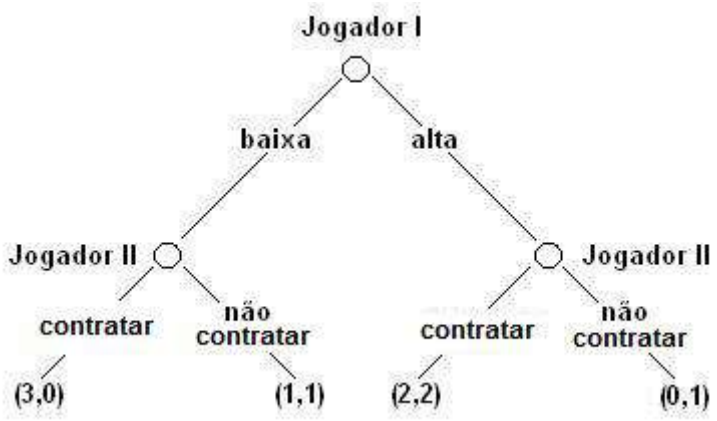

Figura 2.1: Exemplo de jogo na forma extensiva

A figura 2.1 corresponde a um jogo na forma extensiva. O exemplo da figura 2.1, retirado de Turocy e Stengel (2001), é chamado de jogo da Escolha de Qualidade, e é um jogo em árvore com informação perfeita. Cada ponto do ramo é associado a um jogador, que faz um movimento escolhendo o próximo nodo. As linhas de conexão são rotuladas com a escolha do jogador. O jogo começa com o nodo inicial, a raiz da árvore, e termina em algum dos nodos finais, que estabelecem os resultados e determinam os ganhos dos jogadores. O par ordenado ao final da árvore fornece os ganhos dos jogadores I e II, sendo que o primeiro componente do par ordenado representa o ganho do jogador I, e o segundo componente o ganho do jogador II. Na figura, a árvore começa de cima para baixo.

O jogador I é um provedor de *Internet* e o jogador II, um cliente em potencial. Eles avaliam o estabelecimento de um contrato de fornecimentos de serviços por um período de tempo. O provedor pode decidir unilateralmente entre dois níveis de qualidade de serviço, "alta" ou "baixa". Serviços de alta qualidade são mais caros para fornecer, e alguns dos custos são independentes do contrato ser assinado ou não. O serviço de alta qualidade é mais valioso para o consumidor que o serviço de baixa qualidade. De fato, é tão mais valioso que o consumidor iria preferir não contratar o serviço se soubesse que a qualidade é baixa.

O provedor de serviço, o jogador I, faz o primeiro movimento, escolhendo entre qualidade de serviço "alta" ou "baixa". Então o cliente, o jogador II, é informado daquela escolha. O jogador II pode então decidir separadamente entre "contratar" ou "não contratar" em cada caso.

O exemplo da figura 2.1 pode ser analisado por indução de trás-para-frente. Aqui, o jogador II move por último. Já que ele sabe que o jogo irá terminar após o seu movimento, ele pode seguramente selecionar a ação que é melhor para ele. Se o jogador I escolher fornecer serviço de "alta" qualidade, então o consumidor preferirá "contratar", já que o seu ganho resultante será 2, que é maior do que 1 que seria obtido quando ele escolhesse não adquirir o serviço. Sendo racional, o jogador I, que é o primeiro a fazer o movimento, antecipa as escolhas subsequentes do consumidor. Ele então percebe que a sua decisão entre "alta" e "baixa" é efetivamente entre os resultados com os ganhos (2,2) ou (1,1) para os dois jogadores, respectivamente. Evidentemente, ele prefere "alta", o qual resulta em ganho 2 para ele, a "baixa", que levaria a um resultado com ganho 1. Então, a única solução do jogo, como determinada pela indução de trás-para-frente, é que o jogador I ofereça alta qualidade de serviço, e o jogador II responda contratando o serviço.

Algumas vezes os jogos são representados através de matrizes ao invés de árvores. A matriz é o segundo tipo de objeto matemático utilizado para representar jogos. A **matriz do jogo**, diferentemente da árvore do jogo, simplesmente mostra os resultados, representados em termos da função de utilidade, para cada possível combinação de estratégias que os jogadores possam usar. Utilizando matrizes, a ordem do jogo não é considerada importante. Os jogos representados por matrizes também são chamados de **jogos na forma normal** ou **jogos na forma estratégica**.

		Jogador II (cliente)	
		contratar	não contratar
Jogador I (provedor)	alta	(2,2)	(0,1)
	baixa	(3,0)	(1,1)

Figura 2.2: Exemplo de jogo na forma normal

A figura 2.2 mostra um exemplo de jogo na forma normal. O primeiro componente do par ordenado corresponde ao ganho do jogador I e o segundo componente corresponde ao ganho do jogador II. Assim como proposto no exemplo 2.1, o jogador I é um provedor de *Internet* e o jogador II, um cliente em potencial. O cliente prefere comprar se o jogador I fornecer serviço de alta qualidade, e não comprar em caso contrário. Este é o mesmo jogo da Escolha de Qualidade proposto como exemplo na figura 2.1, com a diferença de que o nível do serviço não pode ser colocado de maneira verificável no contrato. O fato de não ser possível colocar de maneira verificável a qualidade do serviço dá um caráter de simultaneidade ao exemplo. O jogador I (provedor), ao formular o contrato, não sabe antecipadamente se o jogador II (cliente) vai contratar ou não o serviço, e o jogador II não tem como saber se o jogador I vai fornecer um serviço de baixa ou alta qualidade. Sem levar em consideração se o cliente (jogador II) escolhe contratar ou não, o provedor (jogador I) sempre prefere fornecer o serviço de baixa qualidade, visto que o ganho é sempre o maior possível - se o jogador I escolhe "alta" e o jogador II escolhe "contratar", o jogador I recebe ganho 2 ao invés de ganho 3, que seria obtido caso tivesse escolhido "baixa"; se o jogador I escolhe "alta" e o jogador II escolhe "não contratar", o jogador I recebe ganho 0 ao invés de ganho 1, resultante caso tivesse escolhido "baixa". Então, é possível perceber que a estratégia "baixa" domina a estratégia "alta" para o jogador I, visto que é a estratégia que maximiza seus os seus ganhos. Sendo um jogador racional, o jogador I escolhe a estratégia dominante "baixa". Uma vez que o jogador II acredita que o jogador I é racional, ele percebe que o jogador I sempre prefere "baixa", e então antecipa o serviço de baixa qualidade como a escolha do provedor. Então ele prefere "não contratar" (ganho 1) a contratar (ganho 0). Então, a racionalidade dos dois jogadores leva à conclusão que o provedor vai implementar o serviço de baixa qualidade e, como resultado, o contrato não será assinado. O resultado da racionalidade individual é pior para ambos os jogadores do que um outro resultado, que é a combinação de estratégia (alta, contratar), onde o serviço de alta qualidade é fornecido e o cliente assina o contrato. No entanto, esse resultado não é digno de confiança, uma vez que o provedor seria tentado a voltar atrás e oferecer apenas o serviço de baixa qualidade.

Os jogos nas formas normal e extensiva não são equivalentes. Os jogos na forma extensiva contêm informação sobre a sequência do jogo e o nível de informação dos jogadores em relação à estrutura do jogo - os jogos na forma normal não. Em geral, um jogo na forma normal poderia representar qualquer um dos vários jogos na forma extensiva. Quando a ordem do jogo é irrelevante para o resultado do jogo, então deve-se estudar a sua forma normal. Quando a ordem do jogo é relevante, a forma extensiva deve ser especificada ou as conclusões não serão realistas (ROSS, 2008).

O **Equilíbrio de Nash** (NASH, 1950a,b, 1951), também chamado de Equilíbrio Estratégico, expressa a propriedade de que nenhum jogador pode unilateralmente alterar sua estratégia e conseguir um ganho melhor. O Equilíbrio de Nash recomenda uma estratégia para cada jogador, e, dado que os outros jogadores também seguem a recomendação, um jogador não pode melhorar sua estratégia unilateralmente. Considerando que todos os outros jogadores também sejam racionais, é razoável para cada jogador esperar que seus oponentes também sigam a recomendação. O Equilíbrio de Nash pode não ser único, existindo casos com dois ou mais equilíbrios de Nash. Porém, nos exemplos das figuras 2.1 e 2.2, existe apenas um equilíbrio de Nash, que são as soluções dos exemplos. A combinação de estratégias (alta, contratar) e (baixa, não contratar) são as soluções e equilíbrios de Nash dos exemplos das figuras 2.1 e 2.2, respectivamente. É importante salientar que uma estratégia dominada nunca pode fazer parte de um equilíbrio, uma vez

que um jogador que pretenda jogar uma estratégia dominada poderia mudar para a estratégia dominante e se sair melhor. Assim, se a eliminação de estratégias dominadas leva a uma única combinação de estratégia, então este é um equilíbrio de Nash. Jogos maiores podem ter equilíbrios únicos que não resultam de considerações sobre estratégias dominadas/dominantes (TUROCY; STENGEL, 2001).

2.2 Teoria dos Jogos Evolucionários

A interpretação padrão da Teoria de Jogos não cooperativos é que o jogo analisado é jogado exatamente uma vez por jogadores completamente racionais que conhecem todos os seus detalhes, incluindo as preferências de cada um por resultados. A Teoria dos Jogos Evolucionários, ao contrário, imagina que o jogo é jogado várias vezes por jogadores biologicamente ou sociologicamente condicionados que são aleatoriamente extraídos de uma grande população. Mais especificamente, cada jogador é "pré-programado" para algum comportamento (formalmente, uma estratégia no jogo), sendo assumido que algum processo evolutivo de seleção opera com o tempo na distribuição de comportamentos da população. A Teoria dos Jogos Evolucionários (TJE) fornece uma ferramenta de vasta aplicabilidade. Seu domínio em potencial vai da Biologia Evolucionária às Ciências Sociais em geral, e à Economia em particular (WEIBULL, 1997).

A Teoria dos Jogos Evolucionários originou-se como uma aplicação da teoria matemática dos jogos ao contexto biológico, a qual surgiu pela percepção de que quando a adaptabilidade depende da frequência (a frequência representa a fração da população que emprega uma determinada estratégia), então um aspecto estratégico é introduzido à evolução. Apesar desse contexto biológico, existe um grande interesse das Ciências Sociais na TJE. Tal interesse em uma teoria com raízes explicitamente biológicas deriva de três fatores. Primeiro, a "evolução" tratada pela TJE não precisa ser uma evolução biológica. "Evolução" pode, neste contexto, ser (frequentemente) entendida como uma evolução cultural, quando esta se refere a mudanças em crenças e normas através do tempo. Segundo, a TJE substitui o conceito de racionalidade da TJ pelo conceito mais fraco de adaptabilidade ou sucesso reprodutivo, que, em muitos casos, é mais apropriado para modelar sistemas sociais do que a suposição de racionalidade como senso comum entre os jogadores. A TJ impõe aos jogadores uma exigência muito elevada de racionalidade, que pode ser chamada de hiper-racionalidade. Estudos experimentais em economia mostram que esta suposição elevada de racionalidade não descreve o comportamento real de seres humanos. Seres humanos raramente são os jogadores hiper-racionais descritos pela Teoria de Jogos. Terceiro, a TJE, como uma teoria explicitamente dinâmica, fornece um elemento importante que falta na Teoria de Jogos (ALEXANDER, 2008).

Adaptabilidade (em inglês, ***fitness***) é um termo que vem da Teoria Darwiniana de que os animais que sobreviveram (ou seja, foram mais aptos) tiveram maior probabilidade de deixar um número maior de descendentes. Assim, em termos biológicos, a adaptabilidade é uma medida do número de cópias dos genes de um indivíduo (PRESTWICH, 1999). Em termos sociais, a adaptabilidade pode ser pensada como uma vantagem adaptativa que determina o número de cópias do comportamento de um indivíduo.

A **Estratégia Evolucionariamente Estável (ESS)** consiste em uma estagnação evolucionária com respeito aos comportamentos sendo considerados, onde não existe mudança na frequência relativa das estratégias com o tempo. Ela pode acontecer de duas maneiras: um comportamento é mais apto que todos os outros (**ESS pura**); ou existe uma combinação específica de comportamentos onde não há um comportamento que seja mais apto

que qualquer um dos demais (**ESS mista**) (PRESTWICH, 1999).

Para que uma estratégia seja **estável evolucionariamente**, ela precisa ter a propriedade de que, se quase todos os membros da população a seguirem, nenhum mutante (ou seja, nenhum indivíduo que adote uma nova estratégia) pode invadir a população com sucesso (ALEXANDER, 2008). O conceito de estratégia evolucionariamente estável refere-se ao cenário em que uma pequena população mutante, onde os indivíduos são "programados" com uma outra estratégia pura ou mista (estratégia mutante), é injetada em uma grande população inicial homogênea, onde todos os indivíduos já são "programados" para jogar uma estratégia pura ou mista. Na população, os indivíduos são repetidamente extraídos de maneira aleatória para participar de um jogo simétrico par-a-par. A estratégia em questão (a estratégia da população homogênea) é dita estável evolucionariamente se existir, para cada estratégia mutante, uma barreira positiva de invasão tal que, se a população de indivíduos jogando a estratégia mutante cair abaixo desta barreira, então a primeira estratégia obterá ganho maior que a estratégia mutante. Uma estratégia é robusta a pressões evolucionárias de uma maneira exata, e a estabilidade evolucionária é um teste de robustez contra uma única mutação por vez. Esta abordagem é focada em interações simétricas par-a-par dentro de uma única e grande população. Em particular, ela não lida com interações que acontecem entre mais de dois indivíduos por vez. Além disso, o critério de estabilidade evolucionária refere-se à relação próxima que existe entre os ganhos no jogo e a propagação da estratégia na população. A adaptabilidade é um conceito sutil, que biologicamente pode ser entendida como o número de descendentes. Supõe-se que os ganhos no jogo representam os ganhos de adaptabilidade biológica ou valor reprodutivo das interações em questão. A propriedade de estabilidade evolucionária não explica como a população chegou a tal estratégia. Ao invés disso, questiona se a estratégia é robusta a pressões evolucionárias, quando for alcançada a estabilidade (WEIBULL, 1997).

Apesar da postura biológica, a estabilidade evolucionária pode fornecer um critério de robustez relevante para o comportamento humano. Em ambientes sociais e econômicos pode ser pensada como uma convenção. A estabilidade evolucionária requer que qualquer grupo pequeno de indivíduos ao tentar uma estratégia alternativa seja menos bem sucedido que os outros indivíduos que jogam a estratégia *status quo* (WEIBULL, 1997).

2.3 Redes *peer-to-peer*

As aplicações de compartilhamento de conteúdo foram as primeiras a popularizar o uso de redes P2P. Aplicações como Napster (2003), Kazaa (2000), eMule (2002), Gnutella (2003) e mais recentemente, BitTorrent (2001) foram as responsáveis pela popularização das aplicações em redes P2P na Internet. Em uma rede P2P, um nodo não faz apenas *download* de dados da rede, mas faz também *upload* dos dados que foram previamente baixados para outros usuários da rede. A largura de banda de *upload* dos usuários deve ser utilizada de maneira a reduzir os encargos de largura de banda que, de outra maneira, seriam colocados em um servidor. Esta seção objetiva conceituar redes P2P, e os conceitos aqui apresentados são mais diretamente relacionados a aplicações de compartilhamento de conteúdo em redes P2P. Esta seção é baseada no trabalho de Androutsellis-Theotokis e Spinellis (2004).

As arquiteturas P2P foram projetadas para compartilhar recursos de computação (conteúdo, armazenamento, ciclos de CPU) pela troca direta, ao invés de requisitar a intermediação ou suporte de uma autoridade servidora central. A motivação para basear aplicações em arquiteturas P2P deriva grandemente da habilidade desta de funcionar, escalar

e se auto-organizar na presença de uma população altamente dinâmica de nodos e recursos de rede sem a necessidade de um servidor central. As arquiteturas P2P também são caracterizadas por sua habilidade de se adaptar a falhas mantendo conectividade e desempenho aceitáveis.

A operação de qualquer sistema de distribuição de conteúdo em redes P2P conta com uma rede física de computadores e conexões entre eles, sob os quais é formada uma rede lógica, referida como rede de sobreposição. A topologia de conexão e o grau de centralização da rede de sobreposição, bem como os mecanismos de roteamento e localização que ela emprega para mensagens e conteúdo, são cruciais para a operação do sistema, já que eles afetam tolerância a falhas, manutenção, adaptabilidade a falhas, desempenho, escalabilidade e segurança. Os mecanismos de roteamento e localização de mensagens e conteúdo dependem da estrutura e do grau de centralização da rede. Por isso, a rede de sobreposição pode ser classificada de acordo com duas características: o nível de centralização e a estrutura.

Em sua forma mais pura, as redes de sobreposição *peer-to-peer* deveriam ser totalmente descentralizadas. Na prática, isto nem sempre é verdade, sendo encontrados sistemas com diferentes níveis de centralização. Especialmente as seguintes três categorias são encontradas:

- Arquitetura puramente descentralizada - todos os nodos da rede desempenham exatamente as mesmas tarefas, agindo como servidor e cliente. Não existe uma coordenação central de suas atividades. A rede Gnutella é um exemplo de arquitetura puramente descentralizada. Na rede Gnutella, a comunicação entre os nodos é feita utilizando quatro tipos de mensagens: i) *ping* - mensagem de um nodo para outro específico, na qual o primeiro faz uma requisição para se anunciar; ii) *pong* - mensagem em resposta ao *ping*; iii) *query* - mensagem de solicitação de pesquisa; iv) *query hits* - mensagem de resposta à mensagem *query*. Para localizar arquivos, o Gnutella emprega um mecanismo de inundação para distribuir mensagens de *ping* e *query*, e cada nodo repassa as mensagens recebidas para todos os seus vizinhos.

- Arquitetura parcialmente centralizada - a base é a mesma dos sistemas puramente descentralizados. Alguns dos nodos, porém, assumem tarefas mais importantes, agindo como índices para os arquivos compartilhados por nodos locais. A maneira pela qual as tarefas são designadas as estes supernodos varia entre diferentes sistemas. É importante, porém, notar que esses supernodos não constituem pontos únicos de falha para uma rede P2P, já que eles são designados dinamicamente e, se eles falharem, a rede irá automaticamente tomar medidas para substituí-los por outros. Os supernodos indexam os arquivos compartilhados pelos nodos conectados a ele, e fazem o serviço de *proxy* das solicitações de pesquisa em nome destes nodos. Todas as consultas (*queries*) são, portanto, inicialmente direcionadas para os supernodos.

- Arquitetura Híbrida Descentralizada - nesses sistemas, existe um servidor central facilitando a interação entre nodos. O servidor central mantém diretórios de meta dados, descrevendo os arquivos compartilhados armazenados pelos nodos. Apesar da interação fim-a-fim e da troca de arquivos acontecerem diretamente entre dois nodos, os servidores centrais facilitam estas interações desempenhando as consultas e identificando os nodos que armazenam os arquivos.

A estrutura refere-se a como a rede de sobreposição é criada. Pode ser de maneira não

determinística ou baseada em regras específicas. As redes P2P são caracterizadas como segue, em termos de sua estrutura.

- Não estruturada - a localização do conteúdo é completamente desrelacionada com a topologia da rede de sobreposição. Em uma rede não estruturada, o conteúdo tipicamente precisa ser localizado. Os mecanismos de procura empregados nas redes não estruturadas têm implicações óbvias, particularmente com respeito a questões de disponibilidade, escalabilidade e persistência. Sistemas não estruturados são geralmente mais apropriados para acomodar populações de nodos altamente dinâmicas.

- Estruturada - apareceu originalmente como tentativa de endereçar a questão de escalabilidade que os sistemas não estruturados estavam enfrentando. Em redes estruturadas, a topologia da rede de sobreposição é fortemente controlada e os arquivos (ou ponteiros para eles) são colocados em locais precisamente especificados. Esses sistemas essencialmente fornecem um mapeamento entre conteúdo e localização, na forma de tabela de roteamento distribuída. As consultas podem ser então eficientemente roteadas para o nodo com o conteúdo desejado. Sistemas estruturados oferecem uma solução escalável para consultas de correspondência exata, ou seja, consultas onde o identificador exato do objeto de dados requisitado é conhecido (quando comparado a consultas com palavras-chave). Uma desvantagem para sistemas estruturados é que neles é difícil manter a estrutura necessária para rotear as mensagens eficientemente em face de uma população muito dinâmica de nodos, na qual os nodos podem se juntar e sair a altas taxas.

- Imprecisamente estruturada - esta categoria está entre a estruturada e a não estruturada.

2.4 Aplicações de *live streaming*

As aplicações de *live streaming* oferecem um serviço para transmissões ao vivo de voz e/ou vídeo pela *Internet*. Essas aplicações não são vinculadas a um tipo particular de arquitetura, sendo que o serviço pode ser oferecido utilizando arquiteturas cliente-servidor, IP *multicast* ou redes P2P. Neste tipo de aplicação, o conteúdo "ao vivo" pode ser realmente ao vivo ou previamente gravado (PADMANABHAN et al., 2002). Porém, em qualquer dos caso, os dados são gerados pouco antes da transmissão pela origem da transmissão de dados. Desta forma, o conteúdo *live streaming* não está disponível previamente. Por ser uma transmissão "ao vivo", as operações disponíveis para mídias armazenadas, as funções do VCR (como por exemplo, *pause* e *stop*), não estão disponíveis para *live streaming*. Um característica do conteúdo *live streaming* é que este tem validade limitada. Se o conteúdo *live streaming* for transmitido fora de certos requisitos de tempo, ele perde sua importância significativamente.

Um ponto fundamental de diferença entre *live streaming* e as mídias armazenadas está na relação entre os usuários e o objeto transmitido. Nas mídias armazenadas, o acesso ao objeto é guiado pelo usuário, que decide o que assistir e quando. Assim, o objeto é diretamente influenciado pelas preferências do usuário. Em *live streaming*, o acesso à mídia é guiado pelo objeto transmitido (por exemplo, a hora de um espetáculo, um programa de TV ou um jogo de futebol). Logo, os usuários são diretamente influenciados por aspectos relacionados à natureza do objeto. Por isso, pode-se dizer que, em *live streaming*, o objeto transmitido assume um papel ativo (porque direciona o acesso à mídia), e os

usuários, que compõem a audiência, assumem um papel passivo (porque podem apenas se juntar ou sair da audiência do objeto) (VELOSO et al., 2002). É importante salientar que, em uma aplicação de *live streaming*, quanto menor for o atraso, maior é a percepção de vivacidade (conteúdo realmente ao vivo) dos usuários. Assim, a aplicação deve sempre visar a diminuição do atraso fim-a-fim (DO; HUA; TANTAOUI, 2004).

No caso específico de aplicações de *live streaming* que utilizam redes P2P como infra-estrutura de transmissão, os nodos participantes da rede retransmitem os *pacotes* gerados pela fonte (a origem da transmissão), ajudando na difusão do conteúdo "ao vivo". Porém, o papel passivo em relação ao objeto transmitido permanece, visto que os nodos que compõem a audiência permanecem guiados pelo objeto transmitido. Uma aplicação de *live streaming* em redes *peer-to-peer* é muito mais sensível ao comportamento de nodos maliciosos, à heterogeneidade na largura de banda dos nodos e à recuperação da rede de sobreposição (reação à dinâmica da rede P2P), do que uma aplicação de compartilhamento de arquivos, por exemplo. Tal sensibilidade é maior devido aos requisitos de tempo impostos pela natureza da aplicação. No capítulo 4 são fornecidos mais detalhes sobre as aplicações de *live streaming* em redes P2P.

3 ANÁLISE DOS TRABALHOS RELACIONADOS

Desde a sua criação, a Teoria de Jogos é utilizada em uma vasta gama de trabalhos. Os trabalhos que mais se assemelham ao aqui proposto e que serviram de inspiração neste estudo são os descritos a seguir.

O modelo BAR (AIYER et al., 2005) foi a primeira proposta a capturar o comportamento de usuários em serviços cooperativos que atravessam múltiplos domínios administrativos. Os usuários são classificados como bizantinos, racionais e altruístas. Em tais serviços cooperativos, os usuários colaboram para prover um serviço que beneficia cada nodo, mas não existe um autoridade central que monitore suas ações. Utilizando a TJ, os autores descrevem o modelo BAR e propõem uma arquitetura geral em três camadas para reduzir a complexidade de construir serviços sobre o modelo proposto. Eles também descrevem o BAR-B, um serviço de *backup* cooperativo que tolera usuários bizantinos e racionais (LI et al., 2006).

No modelo BAR, os nodos altruístas são definidos como nodos corretos do sistema, que seguem fielmente o protocolo. Os nodos racionais são nodos egoístas, que procuram maximizar seus benefícios de acordo com uma função de utilidade (ou ganho) conhecida. Os nodos racionais irão se desviar do protocolo sugerido se, e apenas se, tal desvio do comportamento aumentar sua função de utilidade no sistema. Os nodos bizantinos podem desviar-se arbitrariamente do protocolo sugerido por qualquer razão. Eles podem estar mal configurados ou podem estar otimizados por uma função de utilidade desconhecida, que difere da função de utilidade empregada pelos nodos racionais.

Os protocolos BAR tolerantes permitem os comportamentos bizantino e racional, mas eles forçam os nodos altruístas e racionais a se comportarem identicamente. Os protocolos BAR tolerantes contam com uma forma de policiamento que garante que todos os nodos não bizantinos seguem o mesmo protocolo, e este método torna impossível tirar vantagem dos nodos altruístas (MARTIN, 2007). Além disso, a TJ utilizada pelo modelo BAR supõe que todos os usuários são completamente racionais e egoístas e sempre visam maximizar seus próprios ganhos. Assim, apesar de não aparecer explicitamente na descrição do modelo BAR, existe a suposição de que todos os usuários têm habilidade para tirar proveito da aplicação, ou seja, possuem uma racionalidade bastante alta. Esta suposição vem da TJ, que considera senso comum o fato de todos os usuários terem conhecimento completo do jogo e de todos os possíveis resultados (estrutura do jogo). Porém, tal suposição de hiper-racionalidade pode ser um pouco exagerada em ambientes reais de interação, como no contexto de aplicações de *live streaming*. Esperar uma racionalidade alta em um ambiente de *Internet* implica esperar que cada usuário tenha pleno conhecimento do funcionamento dos protocolos e das aplicações, bem como habilidade de programação suficiente para explorar as fontes de não determinismo existentes. De outra maneira, os usuários racionais não seriam capazes de tirar vantagem de um serviço de *Internet*.

O protocolo BAR *Gossip* (LI et al., 2006) tem como alvo a distribuição de conteúdo *live streaming* em ambientes onde há a presença de usuários bizantinos, altruístas e racionais. É a primeira aplicação de *media streaming* em rede *peer-to-peer* projetada para o modelo BAR.

A característica que define um protocolo epidêmico é que cada nodo troca dados com nodos aleatoriamente selecionados. É precisamente esta aleatoriedade que dá aos protocolos epidêmicos sua robustez. Porém, a aleatoriedade é uma fonte de não determinismo que torna o protocolo epidêmico tradicional difícil de ser implementado sob o modelo BAR. O protocolo epidêmico tradicional dá oportunidade para os usuários racionais se esconderem atrás de ações egoístas, que aparentam ser um comportamento não determinístico legítimo (LI et al., 2006). O protocolo BAR *Gossip* supõe que os usuários se inscrevem para a transmissão antes desta começar e que os usuários não bizantinos permanecem no sistema durante toda a duração da transmissão. Antes da transmissão começar, cada cliente deve gerar um par de chaves público/privada para a sessão. Desta forma todos os usuários são conhecidos e identificados. O protocolo BAR *Gossip* elimina a principal fonte de não determinismo no protocolo epidêmico tradicional, que é a aleatoriedade na seleção dos parceiros, mas ainda mantém a imprevisibilidade e rápida convergência do protocolo epidêmico tradicional. O BAR *Gossip* explora as propriedades dos geradores de números pseudo-randômicos e um esquema de assinatura única para construir um algoritmo verificável de seleção de parceiros pseudo-randômicos. Desta forma, os autores demonstram que os jogadores racionais não tem incentivo para se desviarem do protocolo; aliás, desvios estão sujeitos a punição (desconexão), o que é contraditório com a função de utilidade que os racionais tentam otimizar. Uma das limitações do protocolo é não acomodar *membership* dinâmico, visto que todos os usuários devem obrigatoriamente se juntar à transmissão antes desta começar, e que os usuários não bizantinos permanecem no sistema até o fim da transmissão. Outra limitação é de que a distribuição dos pares de chaves público/privada pode se configurar em uma restrição para a escalabilidade do sistema.

O objetivo do trabalho de Zhang, Xue e Kou (2007) é estabelecer um modelo evolucionário de jogo de compartilhamento de arquivos em redes P2P baseado na visão da TJE. Visto como um sistema evolucionário de aprendizagem, a dinâmica e os macroaspectos da rede P2P são enfatizados, fazendo o processo evolutivo do sistema o foco de pesquisa. Cada nodo na rede P2P é um consumidor e um provedor de recursos. Os nodos da rede P2P têm apenas a escolha de compartilhar ou não compartilhar através de um processo evolucionário de escolha das estratégias, no qual eles aprendem através do tempo que algumas estratégias são melhores que outras. O compartilhamento de recursos em redes P2P é como um jogo *stag hunt* (SKYRMS, 2001). O jogo *stag hunt* é uma história, descrita por Rousseau, que virou um jogo, e é um protótipo de contrato social. Rousseau dá um exemplo de caçada de um veado (*stag*). Se vários indivíduos caçarem juntos, eles podem provavelmente matar um veado e se alimentarem bem. Se eles caçarem sozinhos, cada um obtém um coelho e se alimenta, em média, menos bem. Mas, de acordo com Rousseau, a cooperação é frágil. Se um coelho cruzar o caminho de um caçador engajado em uma caçada cooperativa, ele esquece seus companheiros e vai atrás do coelho sem se preocupar com o custo aos seus companheiros caçadores imposto por sua deserção do esquema cooperativo. A partir do modelo de jogo evolucionário, os resultados balanceados de longo prazo da evolução do sistema podem ser compartilhamento total ou competição total. A conclusão do artigo é de que existe uma correlação entre a direção de evolução do sistema e a matriz de pagamento, e de que a direção de evolução do sistema é afetada

pelas condições iniciais do sistema.

Em redes P2P, os nodos tipicamente precisam rotear pacotes entre os pares. Diante da necessidade de colaboração entre os nodos, a existência de "caronistas" (do inglês, *freeriders*), que são nodos que utilizam a rede mas se recusam a rotear pacotes para outros nodos, traz problemas à aplicação em redes P2P, já que este sistema é fundamentado sobre o comportamento cooperativo dos participantes. Blanc, Liu e Vahdat (2005) utilizam o jogo *random matching* para modelar as interações entre os nodos em uma rede P2P genérica. Apesar de não estar explícito no artigo, acredita-se que o termo "rede P2P genérica" refere-se a uma rede P2P independente de aplicação. No jogo *random matching*, em cada rodada os nodos são combinados aleatoriamente, e então cada par joga uma única rodada do Dilema do Prisioneiro. A estratégia do jogo é baseada em um sistema de reputação primitivo, e é referida como estratégia da "norma social". Cada nodo tem um sistema de reputação consistindo de números entre (0,1,...,τ); 0 (zero) significa inocente, e os outros números significam culpado. Os autores assumem a existência de uma autoridade confiável, que observa as ações dos jogadores e atualiza as reputações de acordo com estas observações. A reputação assegura que um nodo que deserta será punido na próxima jogada, mesmo que ele jogue com um nodo diferente em cada jogada. A estratégia é a seguinte:

- se os dois jogadores são inocentes, ambos cooperam;
- se os dois jogadores são culpados, ambos desertam;
- se um é inocente e o outro culpado, então o jogador culpado coopera e o inocente deserta.

Qualquer desvio da estratégia descrita acima aciona uma punição que dura τ rodadas. Ou seja, o nodo que se desviou da estratégia é marcado como "culpado", fazendo com que seja punido pelos outros nodos. Após τ rodadas, o nodo torna-se inocente de novo, contanto que ele tenha seguido a estratégia acima. Se um nodo se desvia durante a fase de punição (a fase de punição dura τ rodadas), a punição é recomeçada do princípio. No artigo é afirmado que a estratégia da norma social é um equilíbrio de sub-jogo perfeito, contanto que o tamanho da punição τ e o fator de desconto δ sejam estabelecidos corretamente. O fator de desconto pode ser interpretado como uma probabilidade de que o jogador irá continuar jogando o jogo por uma nova rodada. Ele mede a "paciência" do jogador: $\delta = 1$ significa que os jogadores são infinitamente pacientes, enquanto que valores menores de δ significam que os jogadores preferem ganhos de períodos mais curtos. O equilíbrio é também estável, no sentido de que, começando de qualquer estado inicial, ele irá se re-estabelecer após τ rodadas. Desta forma, a principal conclusão do artigo é de que, sob certas hipóteses, a estratégia da norma social é um equilíbrio de sub-jogo perfeito. Os resultados de simulação mostram que o esquema proposto é robusto na presença de nodos *freeriders* (que sempre desertam do esquema cooperativo) e ruídos (ou erros), apesar de existir uma troca entre a existência de fortes incentivos e a tolerância a ruídos. Os autores também mostram que um sistema de reputação não confiável que monitora apenas uma fração dos eventos de roteamento pode ainda assim ser efetiva, com a condição de que a punição seja suficientemente severa.

Feldman et al. (2004) modelam sistemas P2P utilizando o Dilema do Prisioneiro Generalizado. No jogo proposto, um jogador é o cliente e o outro jogador é o servidor, e apenas a decisão do servidor é significativa para o resultado da transação. Um jogador

pode ser um cliente em um jogo e um servidor em outro. Os autores propuseram uma função de decisão recíproca como base para um conjunto de técnicas de incentivo. As técnicas propostas são totalmente distribuídas e incluem: seleção diferenciada de servidor, históricos compartilhados, reputação subjetiva, política adaptativa a nodos sem histórico e históricos de curta duração. Através de simulações, os autores mostraram que estas técnicas podem levar um sistema de usuários a níveis quase ótimos de cooperação. O papel passivo do cliente na versão generalizada do Dilema do Prisioneiro faz com que o jogo proposto pelos autores seja semelhante ao Jogo do Ditador (BOLTON; KATOK; ZWICK, 1998). Como o papel do cliente é totalmente passivo, então ele não tem entradas estratégicas que contribuam para o resultado do jogo. Sendo assim, assim como no jogo do Ditador, pode-se concluir que a versão generalizada do Dilema do Prisioneiro proposta no artigo não pode ser considerada formalmente um jogo, conforme o termo usado na TJ. Para ser um jogo, o resultado de cada jogador deve depender das ações de pelo menos um dos outros jogadores. Como nesse caso o resultado do jogador depende apenas de suas próprias ações, esta situação está relacionada à Teoria de Decisão e não à Teoria de Jogos. Além disto, em especial, esta versão do Dilema do Prisioneiro não captura alguns aspectos essenciais de aplicação de *live streaming* em *peer-to-peer*, tais como requisitos de tempo e topologia da rede de sobreposição.

O trabalho de Menasché, Figueiredo e Souza e Silva (2005) explora o cenário onde um número de aplicações egoístas dividem um enlace congestionado e dinamicamente adaptam suas estratégias para maximizar a qualidade da mídia entregue a seus respectivos usuários. Os autores estudam o ponto de equilíbrio de tais sistemas, e o processo dinâmico pelo qual as aplicações adaptam suas taxas. Para caracterizar este processo, os autores propõem um modelo de jogos dinâmico em duas camadas baseado em modelos evolucionários. Os modelos evolucionários determinam como as taxas de dados das aplicações evoluem com o tempo, enquanto os modelos de desempenho são usados para quantificar a qualidade instantânea da mídia. Os autores fornecem avaliações numéricas e analíticas do modelo proposto. A principal contribuição do trabalho é o *framework* do modelo, o qual combina modelos evolucionários da Teoria de Jogos com modelos de desempenho que permitem entender o comportamento de aplicações (ou usuários) egoístas.

O artigo publicado por Oualha e Roudier (2008) discute a eficiência de usar auditoria como um incentivo para reduzir *freeriders* em aplicações de armazenamento em redes P2P. Assim como em uma aplicação de compartilhamento de arquivos em P2P, incentivos para a cooperação devem ser usados a fim de evitar o comportamento egoísta dos nodos (evitar que os nodos peguem uma "carona" na aplicação armazenando seus dados em outros nodos sem contribuir para a infra-estrutura de armazenamento). Os protocolos de verificação remota de dados têm como objetivo auditar os nodos para detectar aqueles que se comportam mal e dizem armazenar algum dado que na verdade eles destruíram. Os autores afirmam que tais protocolos de auditoria formam a base para um mecanismo eficiente de incentivo à cooperação, pois a auditoria fornece informações sobre o comportamento dos nodos. Desta forma, ela também pode ser utilizada pelos nodos da aplicação para decidirem se eles devem cooperar ou não com um outro nodo. Os autores então avaliam este mecanismo usando um modelo evolucionário de jogo, descrevendo a evolução das estratégias em grandes populações como um resultado de muitas interações locais, cada uma envolvendo um pequeno número de indivíduos randomicamente selecionados. Eles utilizam este modelo para estudar sob quais condições uma estratégia baseada em auditoria vence as estratégias egoístas. Os autores demonstram que uma estratégia baseada em auditoria (chamada de "*discriminate*") pode dominar ou superar a

estratégia *freerider*. A estratégia "*discriminate*", sob algumas condições, é uma estratégia evolucionariamente estável na aplicação de armazenamento de dados em redes P2P.

O trabalho de Pai et al. (2005) apresenta o Chainsaw, um protocolo de *multicast* no nível de aplicação em redes P2P que utiliza a topologia em malha como rede de sobreposição para transmissão de conteúdo. No protocolo Chainsaw, os nodos são notificados dos novos pacotes pelos seus vizinhos e devem requisitar explicitamente cada pacote de seus vizinhos para poder recebê-los. Desta forma, a duplicação de dados pode ser praticamente eliminada. Como esta dissertação utiliza grandemente os conceitos deste protocolo, ele será visto com mais detalhes na seção 4.3.2.

O trabalho de Haridasan, Jansch-Pôrto e van Renesse (2008) descreve um sistema de auditoria projetado para encorajar a troca de dados em aplicações de *live streaming* em redes P2P. A auditoria é empregada para assegurar que os nodos corretos sejam capazes de receber pacotes mesmo na presença de nodos oportunistas - nodos que não fazem *upload* de dados de forma suficiente. É demonstrado que o sistema de auditoria escala bem quando comparado a soluções prévias do tipo "olho-por-olho" (*tit-for-tat*). A seção 4.4.1 fornece mais detalhes sobre este sistema de auditoria.

4 O CENÁRIO: APLICAÇÕES DE *LIVE STREAMING* EM REDES *PEER-TO-PEER*

As aplicações de *live streaming* em redes *peer-to-peer* precisam de muito desempenho em relação a largura de banda - em geral, na ordem de centenas de *kilobits* por segundo. Além disto, têm requisitos de tempo estritos - os dados devem ser disseminados levando em consideração limites máximos de latência. Quanto menor a latência, maior a sensação de vivacidade percebida pela audiência. Os requisitos de tempo requerem entrega pontual dos pacotes para assegurar uma boa qualidade de *playback*. Consequentemente, as aplicações de *live streaming* são muito mais sensíveis a nodos maliciosos do que as aplicações de compartilhamento de arquivos, por exemplo. Assim, uma aplicação de *live streaming* em P2P precisa maximizar a eficiência de comunicação de dados e ser resiliente a ataques que comprometam a entrega do serviço. A aplicação também deve ser capaz de lidar com alta rotatividade em uma grande população de nodos (LIU et al., 2007).

As aplicações de *live streaming* em redes P2P, dependendo de suas características (por exemplo, topologia da rede de sobreposição e protocolo de disseminação de dados), oferecem serviços com qualidades diferentes. O objetivo principal deste capítulo é mostrar o "mundo real", ou seja, mostrar as características de diferentes aplicações de *live streaming* em redes P2P. O modelo proposto no capítulo 6 é uma abstração do cenário apresentado neste capítulo.

4.1 Funcionamento geral das aplicações de *live streming* em P2P

O objetivo das aplicações de *live streaming* em redes P2P é fornecer um serviço de *multicast* no nível de aplicação. Estas aplicações tipicamente possuem um único nodo de origem de disseminação de dados, chamado de **fonte**. A fonte gera pacotes com números sequenciais monotonicamente crescentes, os quais os usuários usam para reorganizar corretamente os pacotes para reproduzir o *playback*. A fonte gera os pacotes a uma taxa constante, chamada de **taxa de *stream***. A disseminação dos pacotes gerados pela fonte (o *stream* de dados) é feita por uma rede P2P, onde cada nodo da rede ajuda retransmitindo os pacotes para outros nodos, não sobrecarregando, desta maneira, a fonte. O endereço da fonte é previamente conhecido, servindo de ponto de entrada para novos usuários.

4.2 Topologia das redes de sobreposição

De maneira geral, a rede de sobreposição das aplicações de *live streaming* pode ter duas topologias diferentes: topologia baseada em árvore e topologia em malha (YIU; JIN; CHAN, 2007; LIU; GUO; LIANG, 2008) ou derivadas destas. Estas topologias são

descritas a seguir.

- Topologia baseada em árvore - esta topologia tem uma estrutura bem organizada (com relação pai-filho entre os nodos), na qual os nodos são organizados de forma a montarem uma árvore, onde a fonte da transmissão de *live streaming* se encontra na raiz da árvore. Esta topologia pode ser implementada de duas maneiras, que são descritas abaixo.

    1. Topologia de árvore única - a topologia de árvore única para *multicast* tem a vantagem de ser simples de implementar, porém possui problemas de tolerância a falhas de enlace (existe um único caminho entre a fonte e o grupo de *multicast*), de sobrecarga dos nodos internos à árvore em relação aos nodos folha (os nodos internos redistribuem conteúdo para os nodos folha, que apenas recebem conteúdo) e vulnerabilidade a *peer churn* (a saída de nodos pode interromper temporariamente a entrega de pacotes para todos os nodos que fazem parte da sub-árvore na qual o nodo que saiu era raiz).
    2. Topologia de árvores múltiplas - a topologia de árvores múltiplas foi proposta para resolver o problema dos nodos folha não contribuírem com largura de banda de *upload*. A topologia de árvores múltiplas mantém mais de um caminho de distribuição de conteúdo entre a fonte da transmissão e o grupo de *multicast*, ou seja, ao invés de uma única árvore de distribuição, múltiplas sub-árvores são construídas. Cada nodo se junta a todas as sub-árvores para receber dados, e possui posições diferentes em sub-árvores diferentes. Ele pode ser um nodo interno em uma sub-árvore e ser um nodo folha em outra sub-árvore. A largura de banda de um nodo é utilizada para fazer *upload* de dados sempre que ele for colocado como um nodo interno em alguma sub-árvore. Para alcançar uma utilização alta da largura de banda dos nodos, o número de sub-árvores na qual um nodo é colocado como nodo interno deve ser ajustado de forma proporcional a sua largura de banda de *upload*. Os problemas de tolerância a falhas e balanceamento de carga da estrutura em árvore única são solucionados, porém é uma tarefa difícil manter as várias árvores de *multicast*.

- Topologia em malha - não se baseia em nenhuma estrutura regular para distribuição de conteúdo para um grupo de usuários. Na topologia em malha, os nodos não estão confinados a uma topologia estática de distribuição de conteúdo - os nodos estabelecem e terminam as parcerias com outros nodos dinamicamente. Em um dado momento, um nodo mantém relações de parceria com múltiplos nodos vizinhos. Um nodo pode fazer *download*/*upload* de dados de/para múltiplos vizinhos simultaneamente. Se um nodo vizinho sai, o nodo ainda pode fazer *download* de dados dos vizinhos restantes. Ao mesmo tempo, o nodo irá encontrar novos vizinhos para manter o nível desejado de conectividade. Como múltiplos vizinhos são mantidos pelos nodos, essa topologia é bastante robusta a *peer churn*.

4.3 Protocolos para disseminação de conteúdo ao vivo em redes P2P

Os protocolos para disseminação de conteúdo multimídia estão intrinsecamente ligados à rede de sobreposição adotada pela aplicação de *live streaming*. Os principais protocolos são descritos abaixo.

4.3.1 Protocolos para topologia em árvore

Tipicamente, os protocolos de disseminação de conteúdo ao vivo da topologia de árvore única apenas exigem que um nodo, ao receber dados de seu nodo pai, envie diretamente os pacotes recebidos para os seus nodos filhos. Ou seja, o nodo recebe dados de seu nodo pai, no nível acima, e passa adiante os dados para os seus filhos, no nível abaixo. Na topologia de árvores múltiplas, o *stream* de dados é dividido em *substreams*, e cada *substream* é transmitido por uma sub-árvore diferente. Assim, ao invés de uma única árvore de distribuição, múltiplas sub-árvores são construídas, uma para cada *substream*. Dentro de cada sub-árvore, o *substream* correspondente é transmitido nível a nível na sub-árvore, da origem da transmissão até os nodos folha. Apesar da divisão do *stream* de dados, o protocolo é o mesmo das topologias de árvore única, em que um nodo recebe dados de um nodo pai, no nível acima, e re-envia os dados recebidos para os seus nodos filhos, no nível abaixo (LIU; GUO; LIANG, 2008).

4.3.2 Protocolos para topologia em malha

Os protocolos epidêmicos são utilizados em redes de sobreposição com topologia em malha. Nos protocolos epidêmicos tradicionais, um nodo escolhe aleatoriamente um subconjunto de nodos alvo e envia os dados recém recebidos a eles. Ao mesmo tempo, o nodo recebe dados de outros nodos. A escolha aleatória de alvos para repassar dados fornece resiliência a falhas. Porém, o protocolo epidêmico tradicional (também chamado *push-based gossip*) não é adequado por causa da duplicação excessiva de dados no processo de aleatório de disseminação. Um nodo pode transmitir dados que o destinatário já possui, assim como é possível que dois nodos enviem dados idênticos para o mesmo destinatário. Ao invés do protocolo epidêmico do tipo *push-based gossip*, um protocolo epidêmico do tipo *pull-based* (como Chainsaw, proposto por Pai et al. (2005), o qual será detalhado a seguir) é mais apropriado para ser utilizado em aplicações de *media streaming* em P2P porque naturalmente evita a redundância na transmissão de dados. Os protocolos *pull-based* são adotados porque reduzem grandemente a redundância de transmissão de dados dos protocolos epidêmicos tradicionais. O funcionamento geral dos protocolos do tipo *pull-based* é baseado na troca periódica de disponibilidade de informação entre vizinhos (um subconjunto de nodos escolhidos para a troca de dados). Ao receber a informação de disponibilidade de dados de um nodo x, um nodo y escolhe um dos segmentos de dados que não possui e envia uma requisição do segmento de dados para x. Então, logo que for possível, x entrega o segmento de dados solicitado por y.

Como ocorre nos protocolos epidêmicos, no Chainsaw, cada nodo (inclusive a fonte) interage somente com um subconjunto de nodos, que são caracterizados como seus vizinhos. É importante salientar que a fonte não transmite todos os pacotes gerados a cada um de seus vizinhos. Isto evita que os nodos vizinhos da fonte não queiram repassar pacotes por terem acesso imediato a todo o conteúdo. Toda vez que um novo pacote é gerado, a fonte transmite uma mensagem de notificação com o objetivo de informar seus vizinhos da existência do novo pacote. De maneira análoga ao comportamento da fonte, a cada novo pacote recebido, os nodos enviam uma mensagem de notificação para seus vizinhos, informando-os que possuem novos pacotes de dados. Desta maneira, cada nodo sabe quais os pacotes que seus vizinhos possuem. Os nodos solicitam os pacotes de seus vizinhos à medida que estes são necessários. Levando em conta as disponibilidades, cada nodo escolhe randomicamente a qual vizinho irá solicitar um pacote. O nodo deve retornar o pacote solicitado dentro de um certo limite de tempo, após o qual o pacote

solicitado tornar-se-á inútil para o receptor. Cada nodo mantém controle de quais pacotes ele requisitou de cada vizinho e evita duplicar requisições do mesmo pacote a múltiplos vizinhos para evitar o recebimento de pacotes duplicados. Os nodos mantêm controle das requisições pendentes de seus vizinhos e enviam os pacotes correspondentes assim que a largura de banda permita. Os nodos também limitam superiormente o número de requisições pendentes de cada vizinho. Isto assegura que o atendimento das requisições seja distribuído de forma aproximadamente homogênea entre todos os vizinhos. Cada nodo mantém uma janela de interesse de tamanho constante, que se movimenta para frente (uma "janela deslizante") a uma taxa igual à taxa de *stream* da fonte. Se um pacote não tiver sido recebido no tempo em que ele ainda está dentro da janela, então o nodo irá considerar aquele pacote perdido e não irá mais tentar obtê-lo de seus vizinhos. O fluxograma da figura 4.1 fornece uma visão simplificada do funcionamento do protocolo Chainsaw.

4.4 Mecanismo de controle de comportamento

Em qualquer ambiente colaborativo, é esperado que as trocas entre os participantes sejam justas. Contudo, frequentemente tal cooperação não acontece espontaneamente. Assim, são adotados mecanismos de controle com o objetivo de estimular que os participantes do ambiente se comportem de maneira adequada. O princípio básico destes mecanismos visa conter o desejo humano de desertar enquanto todos os outros membros cooperam (para evitar a "tragédia dos comuns" [1]). Por ser um ambiente colaborativo, espera-se que um nodo que se junte a aplicação de *live streaming* disponibilize uma parte de sua largura de banda para fazer *upload* de dados para outros nodos. A "tragédia dos comuns" em uma aplicação de *live streaming* pode acontecer quando os nodos participantes fazem muito mais *download* de pacotes do que *upload* (os usuários agem de maneira egoísta). Em *live streaming*, o objetivo dos mecanismos de controle é diminuir a qualidade de *playback* dos nodos egoístas, incentivando-os, desta forma, a seguirem o protocolo. Também é necessário que sejam agregados mecanismos adicionais para prevenir que nodos defeituosos ou maliciosos atrapalhem a aplicação.

4.4.1 Sistema de auditoria

O objetivo de um sistema de auditoria é encorajar a troca de dados entre os nodos da aplicação. Como as aplicações de *live streaming* populares na Internet possuem código fechado, não foram localizadas, até o momento de conclusão deste trabalho, informações disponíveis sobre a existência e, se incorporado, como seria a atuação do sistema de auditoria destas aplicações. Porém, devido a grande quantidade de nodos na aplicação, é razoável pensar que tal sistema de auditoria deveria utilizar uma abordagem probabilística, como a proposta por Haridasan, Jansch-Pôrto e van Renesse (2008). É importante salientar também que, ao se incorporar um sistema de auditoria à aplicação, os procedimentos adotados deveriam levar em consideração a possível ocorrência de falsos positivos e falsos negativos, para evitar ações punitivas a nodos inocentes e manter um nível adequado de identificação de oportunistas.

[1] A "tragédia dos comuns" teve origem no estudo da terra por pastores na Idade Média. No cenário descrito, existem terrenos baldios onde os criadores de gado podem levar seus rebanhos para pastar livremente. Porém, existe um limite para colocar um rebanho em um terreno. A partir de um certo número de cabeças, o pasto se torna escasso, e o gado passa a não engordar, podendo até morrer. Para cada pastor, individualmente, sempre é vantajoso colocar mais uma cabeça de gado no terreno. Do ponto de vista individual, apenas uma cabeça de gado a mais não fará diferença, mas se todos os pastores agirem assim, ao final o pasto ficará inutilizável para todos, e todo o grupo sairá prejudicado. Toda política de proteção ao meio ambiente tem um mecanismo "anti-tragédia dos comuns" embutida (MARINHO, 2005).

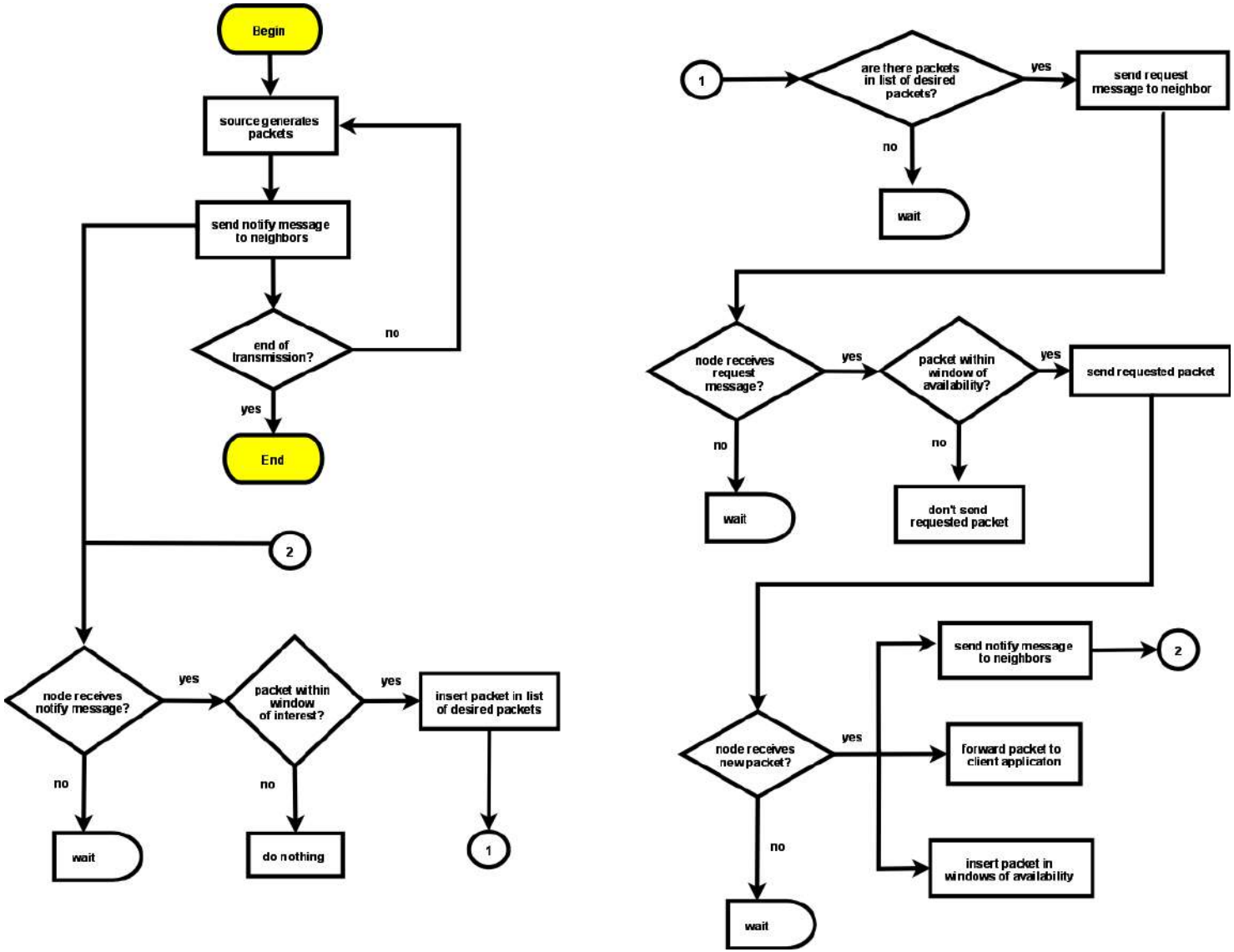

Figura 4.1: Fluxograma do funcionamento do protocolo Chainsaw

O sistema de auditoria proposto por Haridasan, Jansch-Pôrto e van Renesse (2008) determina um limiar variável de *upload* de pacotes com a qual um nodo deve contribuir para permanecer na aplicação. Os nodos são forçados a fornecer informações com respeito à quantidade de pacotes enviados para os vizinhos e recebidos destes, e o sistema de auditoria é responsável por detectar e remover os nodos que não cumprem as especificações do sistema em termos de contribuição e fornecimento de informações. Este sistema de auditoria utiliza dois componentes: auditores locais e globais. Os auditores locais são executados nos nodos participantes do sistema, e portanto não são confiáveis, visto que se um nodo é malicioso, ele pode relatar dados falsos. Os auditores globais são componentes confiáveis que são executados em nodos externos dedicados. Podem existir poucos ou apenas um auditor global. Os auditores locais têm as funções de publicar o histórico de troca de dados do nodo hospedeiro, e realizar auditoria nos históricos dos nodos vizinhos ao nodo hospedeiro. Se a quantidade de dados enviada pelos nodos vizinhos não satisfizer o limiar de *upload*, ou se o conjunto de pacotes que os nodos vizinhos afirmam ter enviado e recebido não corresponderem especificamente ao conjunto de dados que o nodo hospedeiro registra ter recebido e enviado para os vizinhos, então o auditor local faz uma acusação contra o nodo vizinho ao auditor global. Os auditores globais têm as funções de definir o limiar de *upload* de pacotes com a qual um nodo deve contribuir para permanecer na aplicação, e desconectar do sistema os nodos que se comportam mal. Os valores definidos não são divulgados para que não sejam usados como referência por nodos egoístas. Assim, através das observações das interações, os auditores globais do sistema de auditoria definem o melhor valor para o limiar de *upload* dos nodos em qualquer tempo da sessão, e tomam a decisão final com respeito à punição dos nodos acusados como incorretos pelos auditores locais. O "melhor valor" tem por objetivo definir um parâmetro que permita operação com qualidade adequada aos usuários e evitar os falsos positivos. O sistema de auditoria referenciado emprega três modos de operação para determinar o limiar de *upload* dos nodos. Estes modos de operação são: Fixo, Gradual e Baseado em Percentual. O modo Fixo utiliza um limiar não variável (por exemplo, $limiar$ = 0,6). O modo Gradual começa com um valor inicial, por exemplo $limiar$ = 0,5, aumentando o limiar de *upload* apenas se o sistema está comprometido. Os auditores globais tiram amostras do sistema para identificar a taxa média de *download* dos nodos, e se este valor é considerado baixo, eles aumentam o limiar de *upload*. Uma vez que a taxa média de *download* alcance novamente um nível satisfatório, o limiar pode ser reduzido ao seu valor inicial. O modo Baseado em Percentual também utiliza a taxa média de *download* dos nodos para detectar se o limiar deve ser alterado ou não. O valor inicial do limiar é nulo ($limiar$ = 0) e o novo limiar é escolhido das taxas de *upload* das amostras. Após cada coleta de amostras, se o sistema parecer estar comprometido, as taxas de *upload* coletadas são ordenadas e o valor dividindo os 10% menores é usado como o novo limiar de *upload*.

4.4.2 Outros mecanismos de controle de comportamento

Um ataque Sybil acontece quando um nodo tenta obter mais de uma identidade na rede (DOUCEUR, 2002). Para diminuir a ameaça dos ataques Sybil em aplicações em redes P2P, inclusive *live streaming*, é preciso dificultar para um participante a obtenção de múltiplas identidades. Existem algumas técnicas para tentar combater este tipo de ataque (YU et al., 2006; ROWSTRON; DRUSCHEL, 2001), e a mais simples (mas não a mais eficiente para controlar este tipo de ataque) é limitar uma identidade a um endereço IP.

Em geral, as aplicações em redes P2P não mantêm identidades permanentes, per-

mitindo que usuários entrem na aplicação de maneira rápida e fácil. Assim, na maioria dos sistemas P2P, as identidades têm baixo custo (ou mesmo custo zero) para estimular o crescimento da rede, uma vez que incentiva a entrada de participantes no sistema. Especificamente em aplicações de *live streaming*, deve-se permitir que os nodos entrem em qualquer ponto da transmissão. Porém, identidades de baixo custo deixa a aplicação vulnerável a ataques do tipo *whitewashing* (FELDMAN et al., 2006), onde um nodo sistematicamente entra na aplicação após ter sido desconectado por mau comportamento. Nodos *whitewashers* podem levar o sistema ao colapso se não forem devidamente punidos. Infelizmente, não é trivial diferenciar um nodo recém-chegado legítimo de um *whitewasher* em aplicações em redes P2P. Existem duas maneiras de se combater ataques *whitewashing* em redes P2P. A primeira é exigir a utilização de identidades com custo zero, mas insubstituíveis, por exemplo, através da atribuição de identidades por uma autoridade central de confiança. Na ausência de uma autoridade central, pode-se impor um custo inicial de entrada a todos os recém-chegados, os legítimos e os *whitewashers*. O custo de uma nova identidade deve ser customizado de forma a não desincentivar novos usuários a entrarem na aplicação, devido ao longo tempo de espera, e ao mesmo tempo desestimular os usuários que pretendam se utilizar de ataques do tipo *whitewashing* (FELDMAN et al., 2006, 2004). Além disso, é necessário considerar a heterogeneidade de dispositivos com capacidades computacionais muito diversas.

4.5 Outras características

A maior parte das aplicações de *live streaming* em produção é de propriedade de alguma empresa de desenvolvimento (*proprietary*) e de código fechado. Em termos de configuração de rede, a maioria delas disponibiliza poucas opções de parametrização. A aplicação PPLive (2007) permite algumas configurações de rede, como escolher a largura de banda do nodo dentre uma lista de opções disponíveis, o número máximo de conexões por canal e o número máximo de conexões concorrentes. Outras aplicações como Coolstreaming (2005), PPMate, SopCast (2007) e TVUplayer (2005) não permitem qualquer configuração de rede por parte dos usuários.

Assim como acontece com outras aplicações em redes P2P, é razoável esperar que nem todos os nodos da aplicação se comportem de maneira correta. Os usuários, por diversas razões, podem procurar maneiras de alterar/configurar a aplicação cliente para alcançarem seus objetivos. Com variações quanto à nomenclatura e forma de agrupamentos, em geral, os usuários das aplicações de *live streaming* são classificados em três classes (HARIDASAN; JANSCH-PORTO; RENESSE, 2008), que são descritas abaixo:

- Corretos - os nodos desta classe são considerados plenamente cooperativos e seguem o protocolo como dado originalmente.

- Oportunistas - os nodos desta classe são considerados egoístas porque procuram economizar largura de banda fazendo menos *upload* de dados do que eles teriam que fazer caso seguissem o protocolo da aplicação. Um nodo *freerider* é um nodo oportunista extremo que sempre faz *upload* igual a zero.

- Bizantinos - os nodos desta classe podem expressar qualquer tipo de comportamento. Eles podem ser maliciosos ou apenas defeituosos ou mal configurados. É amplamente aceito que não é possível antecipar todas as formas de desvio do protocolo pelos nodos bizantinos. O trabalho de Haridasan e van Renesse (2006) uti-

liza um modelo simplificado que fornece quatro possibilidades de comportamento bizantino:

i) o nodo age como se tivesse falhado por colapso (CRISTIAN, 1991), não requisitando e nem repassando pacotes;

ii) o nodo requisita pacotes, mas não repassa pacotes para os vizinhos, ou seja, faz *upload* igual a 0 (zero) - este comportamento é similar ao do nodo *freerider*, mas as motivações do nodo bizantino são completamente diferentes;

iii) o nodo solicita muitos pacotes aos seus vizinhos, sobrecarregando-os - sob este ataque, os nodos sobrecarregados têm menos recursos para responder a requisições válidas de outros nodos;

iv) uma combinação dos comportamentos ii) e iii) - o nodo não repassa pacotes e sobrecarrega os vizinhos.

Algumas considerações devem ser feitas acerca da classificação dos nodos em *live streaming*. É razoável esperar que os nodos classificados como corretos possuam uma versão não alterada da aplicação de *live streaming* cliente. Os nodos classificados como oportunistas possuem versões da aplicação cliente alteradas com o propósito de economizar largura de banda de *upload* (ou pelo menos customizadas de maneira egoísta, quando a aplicação permite customizações de rede) da aplicação de *live streaming*. Os nodos classificados como bizantinos possuem versões da aplicação cliente alteradas com o propósito de atrapalhar a aplicação de algum modo (um nodo malicioso), ou pode ser apenas uma aplicação cliente defeituosa. Assim, um nodo bizantino atrapalha o funcionamento da aplicação de maneira imprevisível.

Os nodos das aplicações de *live streaming* em P2P não têm obrigação de se manter interessados e, portanto, vinculados à aplicação. Deve-se prever uma participação dinâmica destes por muitas razões diferentes, visto que existem vários fatores que influenciam a entrada e saída de nodos da aplicação, tais como a popularidade do conteúdo transmitido, a duração dos programas e o horário de transmissão. A própria qualidade de *playback* pode influenciar a saída de nodos, quando esta for insatisfatória. Existem alguns estudos que indicam que cerca 90% dos nodos conectados a transmissões de programas populares e não populares permanecem, em média, menos de 90 minutos conectados ao sistema (HEI et al., 2007; SILVERSTON; FOURMAUX, 2006; VU et al., 2007). Porém, é razoável pensar que os usuários não bizantinos que participam de uma aplicação de *live streaming* tenham interesse em manter uma qualidade de *playback* satisfatória enquanto estiverem no sistema.

5 O SIMULADOR DE *LIVE STREAMING*: ANÁLISE ESTATÍSTICA DE *DOWNLOAD* E *UPLOAD*

Este capítulo é destinado a apresentar uma análise estatística do *download* e *upload* do simulador de *live streaming* utilizado neste trabalho, o qual foi apresentado inicialmente por Haridasan e van Renesse (2006) e Haridasan, Jansch-Pôrto e van Renesse (2008). O objetivo deste capítulo é obter parâmetros para a comparação das características de *download* e *upload* de uma aplicação de *live streaming* com as características de *download* e *upload* obtidas do simulador de Teoria dos Jogos Evolucionários, o qual será apresentado no capítulo 7.

5.1 Descrição do simulador

O simulador de *live streaming* é escrito em Java e utiliza o protocolo Chainsaw, conforme descrito na subseção 4.3.2, como o protocolo para disseminação de conteúdo ao vivo entre os usuários da rede P2P. Desta forma, no restante desta dissertação, este simulador será referenciado como simulador do Chainsaw, ou apenas Chainsaw, quando o contexto estiver claro que se refere ao simulador e não ao protocolo.

O simulador do Chainsaw organiza os nodos em uma topologia em malha, a qual é criada ao estilo do Fireflies (JOHANSEN; ALLAVENA; RENESSE, 2006). O Fireflies é um protocolo para dar suporte a uma rede de sobreposição tolerante a intrusão, onde cada um dos participantes tem um visão completa dos nodos ativos na rede. Uma desvantagem óbvia de fornecer a todos os nodos da rede uma visão completa dos nodos participantes da aplicação é uma diminuição da escalabilidade. Porém, os autores acreditam que o Fireflies pode escalar facilmente para milhares de nodos e que isso é suficiente para o funcionamento de muitas aplicações em redes P2P. Assim como proposto no Fireflies, o conjunto de nodos vizinhos a um determinado nodo no simulador do Chainsaw é definido utilizando-se anéis, sendo que o número médio de vizinhos de cada nodo no Chainsaw é igual a duas vezes o número de anéis. Os anéis (o número de anéis é um parâmetro do Chainsaw) são estruturas circulares de endereçamento nos quais todos os nodos são posicionados. Os anéis são utilizados para criar a rede de sobreposição em malha, e o posicionamento dos nodos nos anéis é feito de forma que, em cada anel, a organização dos mesmos seja diferente (com alta probabilidade). Um nodo ordena os membros em cada anel no sentido horário, com ele mesmo na posição 0 (zero), o seu primeiro sucessor na posição 1 e assim por diante. A idéia básica do protocolo é que um nodo monitore, em cada anel, seu sucessor ativo de mais baixa posição. A rede de sobreposição em malha simulada pelo Chainsaw é estática, ou seja, uma vez estipulados os vizinhos de cada nodo, a relação de vizinhança é mantida constante em toda a sessão de *live streaming*,

independentemente do desempenho (em relação à quantidade de *upload* de pacotes) dos nodos vizinhos.

A versão original do simulador do Chainsaw foi alterada para simular as três classes de usuários comumente encontradas em aplicações de *live streaming*, como descritas na seção 4.5: i) corretos; ii) oportunistas; e iii) bizantinos (os quatro tipos de ataques). Porém, para esta análise estatística, não serão simulados ataques à rede, já que o objetivo deste capítulo é fazer uma análise da aplicação de *live streaming* em condições normais de funcionamento. Assim, em todas as simulações deste capítulo, as redes são formadas inteiramente por nodos corretos.

O simulador do Chainsaw foi executado utilizando a IDE (*Integrated Development Environment*) Eclipse em um computador PC de arquitetura Intel, processador Intel Core 2 Duo de 1.83GHz, 2GB de memória RAM e sistema operacional Windows Vista Home Premium. Este simulador utiliza vários parâmetros para configurar uma sessão de *live streaming*. Os principais são descritos na tabela 5.1. Para este trabalho, alguns parâmetros foram pré-estabelecidos e utilizados sem variações em todas as simulações deste capítulo. Os valores dos parâmetros fixos também são mostrados na tabela 5.1.

Tabela 5.1: Principais parâmetros do simulador do Chainsaw

Parâmetro	**Descrição**	**Valor**
N	Número de nodos	-
PACKETSIZE	Tamanho do pacote de dados (em kilobits)	10 kb
MCASTRATE	Taxa de *stream*	500 kbps
NONSENDERATTENDRATIO	Capacidade máxima de *upload* dos nodos (exceto a fonte)	1,1 . MCASTRATE = 550 kbps
SENDERATTENDRATIO	Capacidade máxima de *upload* da fonte	4,0 . MCASTRATE = 2.000 kbps
SEEDNEIGHBORS	Número de vizinhos da fonte	20
kMax	Número médio de vizinhos dos nodos (exceto a fonte)	6
MCASTWINDOWSIZE	Janela de disponibilidade (em segundos)	10s
MCASTINTERESTWINDOWSIZE	Janela de interesse (em segundos)	8s
LAN_LATENCY	Latência entre nodos (em milisegundos)	50ms
MCASTTRANSMISSIONTIME	Duração da sessão (em segundos)	600s

5.2 Análise estatística

A análise estatística realizada nesta seção utiliza redes com 125, 250, 500 e 1.000 nodos. Para cada tamanho de rede são realizadas 30 execuções do Chainsaw. Como os processos de criação da rede de sobreposição e de disseminação de pacotes pelo Chainsaw são estocásticos, são realizadas 30 execuções de simulação para se obter confiança nos resultados obtidos.

A sessão de *live streaming* foi fixada em 600 segundos, mas os dados colhidos para análise abrangem os segundos de 31 a 600 porque os primeiros 30 segundos são necessários para preencher completamente os *buffers* das janelas de disponibilidade e de interesse.

A figura 5.1 apresenta os histogramas da distribuição das taxas de *upload* dos nodos em redes com 125, 250, 500 e 1.000 nodos. A construção de histogramas tem caráter preliminar em muitos estudos estatísticos e é um importante indicador de como é a distribuição dos dados analisados. O eixo X corresponde às taxas de *upload* dos nodos, que são agrupadas em classes para comporem os retângulos do histograma, e o eixo Y corresponde à respectiva frequência da classe da taxa de *upload*, ou seja, corresponde à quantidade de nodos inseridos em determinada classe da taxa de *upload*. Os dados dos histogramas correspondem à taxa média de *upload* de cada nodo participante da sessão de *live streaming*, ou seja, corresponde à taxa média de *upload* de cada nodo da rede nos 570 segundos da sessão (600 segundos menos 30 segundos iniciais). Além disto, os dados dos histogramas são obtidos de 30 execuções do Chainsaw, ou seja, cada histograma contém dados de *upload* de 30 vezes o número de nodos na rede (por exemplo, 30 vezes 125 = 3.750 nodos). Assim como observado por Haridasan, Jansch-Pôrto e van Renesse (2008), detectou-se que existem discrepâncias entre as quantidades de *upload* efetivamente doadas pelos nodos da aplicação. Idealmente, para manter a justiça no consumo de banda dos nodos, seria desejável que todos os nodos fizessem *upload* de dados o mais próximo possível da taxa de *stream*. Porém, como pode ser observado na figura 5.1, na realidade alguns nodos fazem *upload* abaixo da taxa de *stream*, enquanto outros fazem *upload* à taxa máxima definida pela aplicação. A posição física dos nodos no sistema é uma fator que influencia esta discrepância. Devido à posição física dos nodos na rede, alguns nodos terminam participando mais ativamente no processo de disseminação, enquanto outros terminam contribuindo com menos. Esta discrepância é observada apesar de todos os nodos estarem seguindo o protocolo Chainsaw como dado originalmente. Como pode ser observado nos gráficos, a taxa de *upload* em torno de 75% da taxa de *stream* é a taxa mínima de *upload* que começa a ser significativa em termos de número de nodos. Este valor mínimo observado da taxa de *upload* (75% da taxa de *stream*) é importante porque será utilizado como parâmetro de entrada para o simulador de Teoria dos Jogos Evolucionários a ser apresentado no capítulo 7.

Neste trabalho, as variâncias temporais do *download* e *upload* são utilizadas como pontos de comparação entre o simulador de TJE e o Chainsaw. Um aspecto importante no estudo descritivo de um conjunto de dados é o da determinação da dispersão dos dados em relação à media. A dispersão é importante porque dois conjuntos de dados diferentes podem ter a mesma média, mas dispersão de dados bastante diferentes. Desta forma, o restante desta seção é destinada a caracterizar as variâncias de *download* e *upload* do Chainsaw.

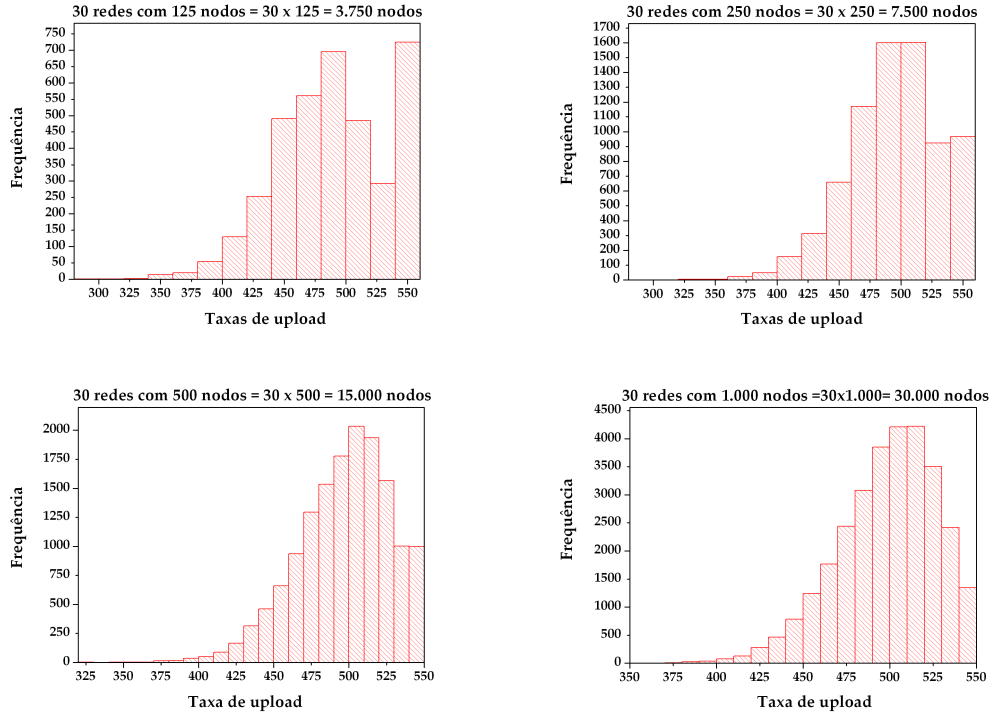

Figura 5.1: Histogramas das taxas de *upload*

Sejam n_p o número de nodos da rede P2P, $(j) \in n_p$ um nodo da rede, e n_s o número de execuções coletadas para análise. Então, a variância do *download* da execução i no tempo t (variância temporal de *download* da execução i), e a variância do *upload* da execução i no tempo t (variância temporal de *upload* da execução i), respectivamente $var(down)_i(t)$ e $var(up)_i(t)$, são definidas como segue.

$$var(down)_i(t) = \frac{1}{(n_p - 1)} \sum_{j=1}^{n_p} (down_i^{(j)}(t) - \langle down_i(t) \rangle)^2 \tag{5.1}$$

e

$$var(up)_i(t) = \frac{1}{(n_p - 1)} \sum_{j=1}^{n_p} (up_i^{(j)}(t) - \langle up_i(t) \rangle)^2 \tag{5.2}$$

Onde $down_i^{(j)}(t)$ e $up_i^{(j)}(t)$ correspondem, respectivamente, à taxa de *download* e *upload* do nodo j, na execução i e no tempo t, e $\langle down_i(t) \rangle$ e $\langle up_i(t) \rangle$ correspondem às médias do *download* e *upload* da execução i no tempo t, as quais são calculadas da seguinte forma:

$$\langle down_i(t) \rangle = \frac{1}{n_p} \sum_{j=1}^{n_p} (down_i^{(j)}(t)) \tag{5.3}$$

e

$$\langle up_i(t) \rangle = \frac{1}{n_p} \sum_{j=1}^{n_p} (up_i^{(j)}(t)) \tag{5.4}$$

A partir das equações 5.1 e 5.2, a média da variância temporal do *download* e *upload* de todas as execuções (30, neste caso), respectivamente $var(down)(t)$ e $var(up)(t)$, são definidas como:

$$var(down)(t) = \frac{1}{n_s(n_p - 1)} \sum_{i=1}^{n_s} \sum_{j=1}^{n_p} (down_i^{(j)}(t) - \langle down_i(t) \rangle)^2 \tag{5.5}$$

e

$$var(up)(t) = \frac{1}{n_s(n_p - 1)} \sum_{i=1}^{n_s} \sum_{j=1}^{n_p} (up_i^{(j)}(t) - \langle up_i(t) \rangle)^2 \tag{5.6}$$

Os gráficos 5.2 e 5.3 mostram a média da variância temporal do *download* e *upload*, com as respectivas barras de erro, de 30 execuções do Chainsaw com redes de 1.000 nodos. O erro é calculado através da fórmula do erro padrão da média (se), e é computado da seguinte forma:

$$se(t) = \frac{sd(t)}{\sqrt{n}} \tag{5.7}$$

onde $sd(t)$ é o desvio padrão da amostra e n é o tamanho da amostra (número de execuções, neste caso, 30 execuções). Foi observado que a partir do tempo $t > t_{min}$, onde $t_{min} \sim 30s$, as variâncias seguem uma lei de potência tal que:

$$var(down)(t) \sim t^{\theta_{down}} \tag{5.8}$$

e

$$var(up)(t) \sim t^{\theta_{up}} \tag{5.9}$$

onde, θ_{down} = -0,161(2) e θ_{up} = -0,164(2), e (2) indica o erro, ou seja, os valores de θ_{down} e θ_{up} têm erro 2 na terceira casa decimal. A lei de potência é um tipo especial de relação matemática entre duas quantidades, na qual uma variável é proporcional à uma potência da outra. Leis de potência são encontradas em fenômenos da natureza e em fenômenos artificiais, tais como a distribuição de terremotos, extinção de espécies e *crashes* de bolsas de valores (GLERIA; MATSUSHITA; SILVA, 2004).

Com o objetivo de estudar o efeito de escala (quantidade de nodos na rede) na variância do *download* e *upload*, também foram calculadas as variâncias para redes com quantidade variada de nodos. Foi observado que a lei de potência se mantém, mas os valores de θ_{down} e θ_{up} não são os mesmos. O tamanho da rede tem um efeito significativo no expoente da lei de potência que rege a variância do *download* e do *upload* nas aplicações de *live streaming* em redes P2P. Como pode ser observado na figura 5.4, θ_{up} é máximo (θ_{up} = -0,164(2)) na maior rede analisada (1.000 nodos), e é mínimo (θ_{up} = -0,230(2)) na menor rede (125 nodos). Assim, a tendência de θ_{up} é crescer juntamente com o tamanho da rede. Porém, θ_{down} tem um comportamento bastante diferente. θ_{down} é mínimo (θ_{down} = -0,161(2)) na maior rede (1.000 nodos), e é máximo (θ_{down} = -0,072(2)) em uma rede de tamanho intermediário (rede com 250 nodos).

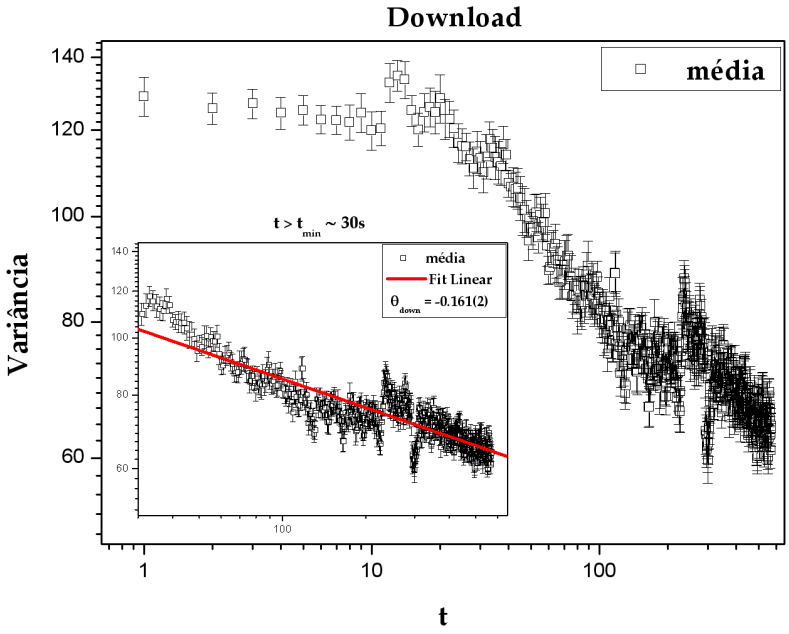

Figura 5.2: Gráfico externo: variância do *download* de 30 execuções (rede com 1.000 nodos); gráfico interno: *fit* linear da variância, com o respectivo valor θ_{down} da lei de potência

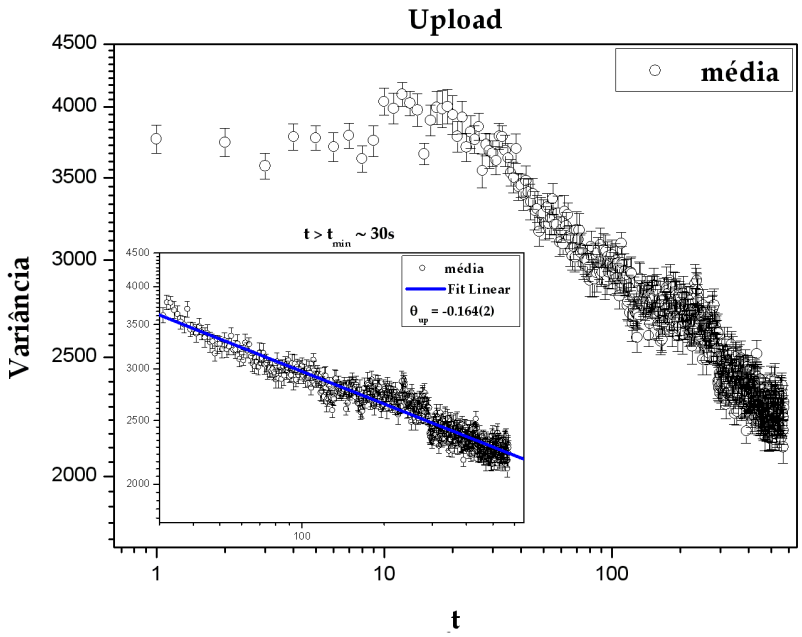

Figura 5.3: Gráfico externo: variância do *upload* de 30 execuções (rede com 1.000 nodos); gráfico interno: *fit* linear da variância, com o respectivo valor θ_{up} da lei de potência

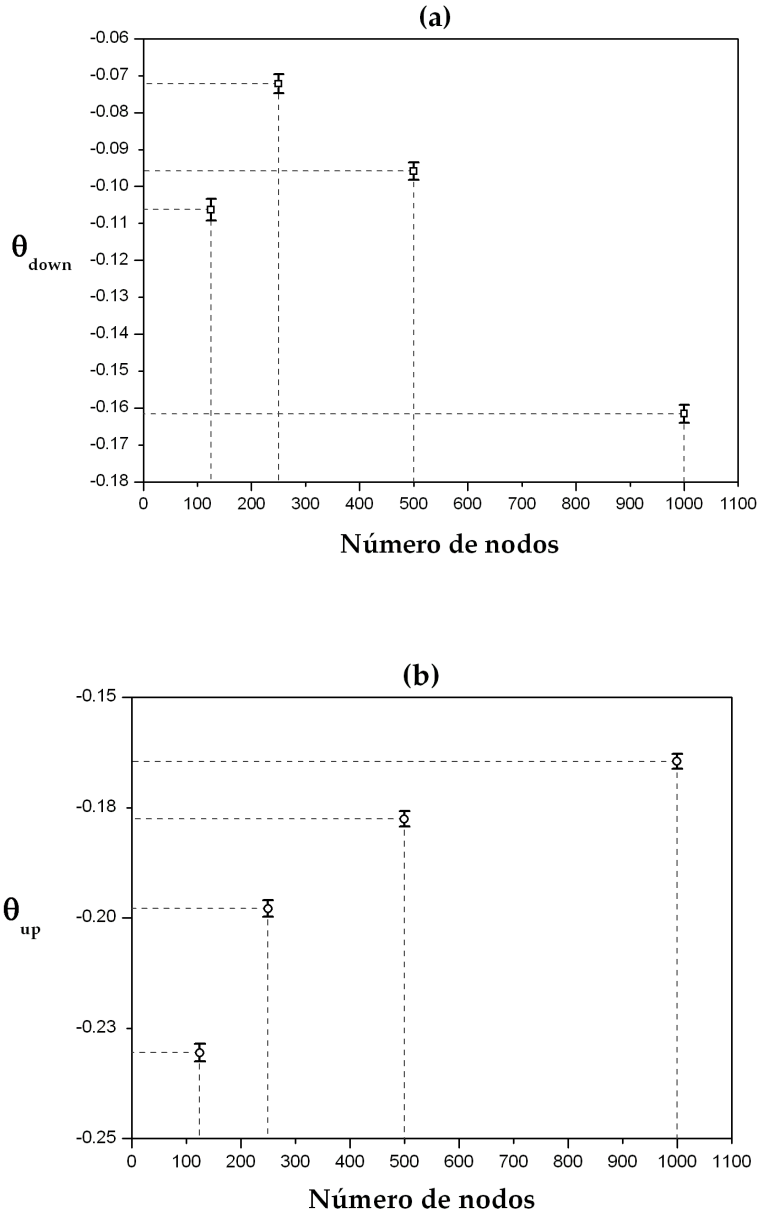

Figura 5.4: Valores de θ_{down} (figura (a)) e θ_{up} (figura (b)) para redes com 125, 250, 500 e 1.000 nodos

6 O MODELO

Neste capítulo é proposto um modelo baseado em Teoria dos Jogos Evolucionários com o objetivo de investigar o cenário de aplicações de *live streaming* em redes P2P.

6.1 Delimitação do cenário

O cenário em questão é o de uma aplicação de *live streaming* em redes P2P, com protocolo epidêmico do tipo *pull-based* e topologia em malha, a qual vários usuários se juntam com o objetivo de assistir a uma determinada transmissão. A aplicação é cooperativa, de forma que um novo usuário deve ajudar a aplicação retransmitindo pacotes para outros usuários, visando manter, desta forma, a capacidade global de *upload* da aplicação. A aplicação cliente não permite customização de rede, e espera que os usuários tenham largura de banda suficiente para fazer *download* e *upload* de pacotes a uma taxa determinada pela aplicação. Assume-se que os usuários das aplicações de *live streaming* são usuários que contratam um serviço de Internet de um ISP (*Internet Service Provider*) para seus pontos de conexão. Em geral, os ISPs fornecem conexões assimétricas, sendo que a largura de banda de *download* é normalmente muito maior que a largura de banda de *upload*. Neste trabalho, por premissa, todos os usuários são homogêneos quanto ao tipo de conexão e respectiva capacidade de *download*/*upload*, e a largura de banda de *download* é suficiente para assistir à transmissão. Cada usuário possui uma função de utilidade bem definida, a qual relaciona a quantidade de *upload* e *download* da aplicação e fornece um índice de satisfação do usuário com a aplicação utilizada. O objetivo da função de utilidade é modelar o desejo dos usuários de *Internet* em assistir à transmissão de *live streaming* e ainda economizar largura de banda de *upload* para ser utilizada em outras aplicações que consomem largura de banda, como aplicações de voz sobre IP (VoIP) e de compartilhamento de arquivos. O processo dinâmico acontece quando os usuários procuram alternativas para melhorar a sua satisfação (maximizar a sua função de utilidade). Uma alternativa surge quando uma aplicação cliente diferente, disponibilizada por terceiros, é colocada à disposição dos usuários. Supõe-se que os usuários têm uma visão limitada do sistema e das ações dos outros jogadores, considerando-se o cenário de informação incompleta e imperfeita. Assume-se também que os usuários não são capazes de fazer considerações de longo prazo sobre o impacto de determinada aplicação cliente no desempenho da aplicação (usuários "míopes"). Portanto, utilizam um processo de tentativa e erro para a escolha de uma aplicação cliente.

6.2 O jogo para modelagem do cenário de *live streaming*: descrição informal

O jogo proposto nesta seção visa capturar a interação que existe em aplicações de *live streaming* que utilizam o protocolo do tipo Chainsaw, conforme descrito na subseção 4.3.2, e que agrupam os nodos em uma topologia em malha para interagirem em grupo. Este jogo pode ser visto como uma abstração de alto nível do protocolo Chainsaw, e as interações realizadas através do jogo são descritas pelo algoritmo 1.

Com o objetivo de explicar o jogo, considera-se inicialmente que todos os nodos são "corretos". Mais detalhes sobre o comportamento de nodos não corretos no contexto do jogo são fornecidos na seção 6.3. Um nodo correto ajuda seus vizinhos fazendo *upload* dos dados que foram requisitados por eles. Em contrapartida, o nodo espera que seus vizinhos ajam da mesma maneira, fazendo *upload* de dados quando ele requisitar. Um nodo correto só pode ajudar seus vizinhos se for classificado como "contribuinte". A classificação de um nodo como contribuinte pode envolver outros parâmetros da aplicação, mas para um estudo inicial será considerada apenas a taxa de *upload* dos nodos. Assim, se a taxa de *upload* do nodo estiver dentro dos limites máximo e mínimo estabelecidos pela aplicação, então ele é classificado como contribuinte.

Cada nodo possui taxas de *download* e *upload*, as quais são alteradas dinamicamente à medida que o nodo interage com os seus vizinhos. Deve-se lembrar que o *download* de um nodo é composto pelo *upload* que seus vizinhos fazem para ele. Assim, nesta seção é proposto um jogo de vizinhança cooperativa de indivíduos. O jogo é modelado em rodadas de interação, e todo nodo i da aplicação interage com os seus vizinhos em cada interação it. Neste contexto, um nodo i solicita ajuda a seus vizinhos quando sua taxa de *download* está abaixo da taxa de *stream* da aplicação, representada por T_s. A ajuda é recebida quando os nodos vizinhos de i aumentam o *upload*, representando que transmitem dados para o nodo i, o qual, em consequência, tem a sua taxa de *download* aumentada. A quantidade de *upload* que será requisitada pelo nodo i é representada por $g = g_{max} * x * c(it) + g_{min}$. Os parâmetros g_{max} e g_{min} são utilizados para definir a quantidade máxima de *upload* que poderá ser requisitada, e com o objetivo de dar um caráter aleatório a esta quantidade, a variável g_{max} é multiplicada por $x = U[0, 1]$, que é uma variável uniforme pseudo-aleatória. A função $c(it)$ corresponde a uma função de "resfriamento" na troca de dados. Percebe-se, pela observação do simulador de *live streaming*, que existe um decaimento da variância temporal de *upload* e *download*, indicando que a dispersão dos valores em relação ao *download* e *upload* médios cai com o tempo. Desta forma, a função $c(it)$ é uma função *ad hoc* utilizada para tentar capturar essa característica do simulador do Chainsaw. No jogo proposto, um nodo i solicita a ajuda g a um número aleatório de vizinhos n, e divide a quantidade de dados requisitada entre os vizinhos, de forma que a ajuda solicitada a cada vizinho escolhido é $f = g/n$. No protocolo Chainsaw, os pacotes chegam de maneira aleatória a cada nodo vizinho de i, os quais os disponibilizam para i. O nodo i então checa o que cada vizinho pode fornecer e tenta balancear as requisições entre eles. A função f tem como objetivo capturar a característica aleatória que existe na obtenção de pacotes. De maneira análoga ao que acontece quando o *download* do nodo i está abaixo da taxa de *stream*, quando a taxa de *download* do nodo i está acima da taxa de *stream* T_s, o nodo dispensa a ajuda extra e diminui o *upload* de seus vizinhos, representando que os vizinhos foram dispensados de transmitir alguns pacotes para o nodo i, que, consequentemente, tem sua taxa de *download* diminuída. No jogo, $a_{\max}$ é um parâmetro que corresponde ao parâmetro NONSENDERATTENDRA-

TIO do Chainsaw (vide tabela 5.1), o qual estabelece a capacidade máxima de *upload* dos nodos corretos em uma sessão de *live streaming*. O valor do parâmetro NONSENDER-ATTENDRATIO não é determinado pelo Chainsaw, mas a partir de valores estimados ou vinculados à realidade tecnológica atual com relação à largura de banda de *upload*. $a_{\min}$ corresponde a um valor aproximado mínimo de *upload* exigível de um nodo correto, e é um valor observado nas simulações do Chainsaw (vide seção 5.2). Um nodo vizinho de i só pode auxiliar i se sua taxa de *upload* estiver entre $a_{\max}T_s$ e $a_{\min}T_s$ (se o nodo vizinho for um nodo "contribuinte").

Algoritmo 1: O jogo do *live streaming*

```
1.  for it = 1 to number_of_interactions do
2.      for i = 0 to (number_of_nodes − 1) do
3.          (∗ Create a random vector of the neighborhood ∗)
4.          neighborhood = random vector of the neighborhood of node i
5.          (∗ If download rate is below the stream rate, node i asks for help ∗)
6.          if node(i).down < T_s then
7.              (∗ Randomly chooses a number of neighbors to ask for help ∗)
8.              n = U[1, number_of_neighbors]
9.              (∗ c(it) = "cooling" function of the system at round it ∗)
10.             (∗ g_max, g_min = maximum upload quantity ∗)
11.             x = U[0, 1]
12.             g = g_max ∗ x ∗ c(it) + g_min
13.             f = g/n
14.             for j = 1 to n do
15.                 (∗ If the upload of node j is below the maximum upload rate, then node
                    j is a contributing node ∗)
16.                 if neighborhood[j].upload < a_max ∗ T_s then
17.                     (∗ Increase the upload of node j and the download of node i ∗)
18.                     neighborhood[j].upload = neighborhood[j].upload + f
19.                     node(i).down = node(i).down + f
20.         else
21.             (∗ If download rate is above the stream rate, then node i dimisses help ∗)
22.             n = U[1, number_of_neighbors]
23.             x = U[0, 1]
24.             g = g_max ∗ x ∗ c(it) + g_min
25.             f = g/n
26.             for j = 1 to n do
27.                 (∗ If the upload of node j is above the minimum upload rate, then node j
                    is a contributing node ∗)
28.                 if neighborhood[j].upload > a_min ∗ T_s then
29.                     (∗ Decrease the upload of node j and the download of node i ∗)
30.                     neighborhood[j].upload = neighborhood[j].upload − f
31.                     node(i).down = node(i).down − f
```

A descrição formal do jogo apresentada no algoritmo 1 é fornecida na seção 6.4.

6.3 O conjunto de estratégias

Neste modelo, serão consideradas duas das três classes de comportamento mostradas na seção 4.5. Desta forma, neste trabalho os nodos são agrupados em duas classes, as quais caracterizam as duas estratégias puras consideradas. A classificação dos nodos é baseada em Haridasan, Jansch-Pôrto e van Renesse (2008). Cada estratégia corresponde a uma aplicação cliente colocada à disposição dos usuários de *live streaming*, e neste

trabalho são caracterizadas em termos de $a_{\max}$ e $a_{\min}$. É importante salientar que as taxas de *upload* $a_{\max}$ e $a_{\min}$ não são valores conhecidos *a priori* pelos usuários.

- Correto - utilizando esta estratégia, o nodo segue o protocolo de disseminação como fornecido originalmente. Com esta estratégia, o nodo sempre coopera, ou seja, sempre faz *upload* para os seus vizinhos conforme determinado pela aplicação. Este nodo faz *upload* de dados com valores entre $a_{\min}$ e $a_{\max}$, os quais foram estabelecidos pela aplicação de *live streaming*.

- Oportunista - com esta estratégia, o nodo não coopera de maneira correta com os seus vizinhos, fornecendo menos dados para *upload* a seus vizinhos do que o determinado pela aplicação. Este nodo tenta economizar largura de banda fornecendo dados para *upload* a uma taxa fixa ω. Seja k um nodo oportunista, e $a_{\max\_k}$ e $a_{\min\_k}$ as taxas máxima e mínima de *upload* do nodo oportunista k. Então tem-se que:

$$a_{\max\_k} = \omega \;\; e \;\; \begin{cases} a_{\min\_k} = a_{\min} & \text{, se } \omega > a_{\min} \\ a_{\min\_k} = \omega & \text{, se } \omega < a_{\min} \end{cases}$$

  Como exemplo numérico, sejam $a_{\max}$ = 1,1, $a_{\min}$ = 0,75 e ω = 0,9. Então $a_{\max\_k}$ = 0,9 e $a_{\min\_k}$ = 0,75. Se ω = 0,6, então $a_{\max\_k} = a_{\min\_k}$ = 0,6. Um *freerider* é um nodo oportunista extremo que sempre faz *upload* ω = 0 (zero), de forma que $a_{\max\_k} = a_{\min\_k}$ = 0.

Apesar de não ser modelado no conjunto de estratégias deste trabalho, algumas considerações podem ser feitas sobre a inserção do comportamento bizantino neste modelo. Como ponto de partida, pode-se pensar em um nodo bizantino em *live streaming* como um "agente anti-social", conforme descrito por Brandt e Weiß (2002), os quais utilizam o contexto do leilão Vickrey para introduzir este tipo de jogador. O agente anti-social é um jogador que aceita pequenas perdas se puder infligir perdas pesadas aos outros jogadores. Do ponto de vista do agente anti-social, seu próprio lucro (do inglês, *profit*) e as perdas de outros jogadores têm igual importância. Levando isto em consideração, o agente anti-social é formalizado através de um agente que tenta maximizar a diferença ponderada entre seu próprio lucro e o lucro dos outros competidores. O agente anti-social pretende maximizar a sua utilidade (ou ganho, da palavra inglesa *payoff*), que é dada pela seguinte equação:

$$payoff_i = (1 - d_i) profit_i - d_i \sum_{j \neq i} profit_j \;^{1}$$

$d_i \in [0, 1]$ é chamada taxa de derrogação. A taxa de derrogação é útil porque captura formalmente o grau de anti-sociabilidade do jogador. Ela fornece ao ganho mais flexibilidade ao analisar o agente anti-social. A equação cobre jogadores "normais" (jogadores que não são anti-sociais) quando d = 0, indicando que o ganho é igual ao lucro. Se d < 0,5, o jogador tem "anti-sociabilidade moderada" e coloca um pouco mais de ênfase em seu lucro do que nas perdas dos outros jogadores. Se d > 0,5, prejudicar outros jogadores é mais importante que o próprio lucro. Se d = 1, o jogador é considerado puramente destrutivo e possui "anti-sociabilidade agressiva". Nesse caso, o objetivo do agente é prejudicar

[1] A palavra *profit* é utilizada como dada na equação original.

os outros jogadores a qualquer custo. Se d = 0,5, o agente tem "anti-sociabilidade balanceada", ou seja, seu próprio lucro e o lucro dos outros jogadores têm igual importância.

É importante notar que a atitude anti-social descrita, que é plausível em todos os mercados competitivos, é racional à medida que o agente pretende se beneficiar das perdas dos rivais no futuro (por exemplo, para colocar o rival fora do mercado). O estudo sobre a inserção do nodo bizantino como agente anti-social será deixado para trabalhos futuros devido à complexidade que sua inserção traz aos casos estudados e à falta de bons modelos para descrever o seu comportamento.

6.4 O modelo conceitual

Seja $I = \{1, 2, ..., N\}$ o conjunto de usuários da aplicação de *live streaming* em P2P, onde N é um inteiro positivo e $|I|$ corresponde ao número de elementos de I. O jogo descrito na seção 6.2 é modelado em interações it, e cada usuário $i \in I$ da aplicação corresponde a um jogador. Seja $C = \{0, 1\}$ o conjunto de estratégias disponíveis (cada aplicação cliente corresponde a uma estratégia pura do jogo) para cada usuário $i \in I$, de tal forma que 0 corresponde à estratégia correta, e 1 à estratégia oportunista, como descritas na seção 6.3. Assume-se que todos os usuários possuem a mesma função de utilidade (u), a ser definida na subseção 6.4.1. Os nodos são agrupados em uma topologia em malha para interagirem em grupo, de maneira que todos os nodos têm o mesmo número médio de vizinhos. O nodo i interage apenas com um subgrupo $j \in \xi_i = \{j_1, ..., j_{|\xi_i|}\}$, onde $\xi_i \in I$ representa a vizinhança do nodo i, e $|\xi_i|$ corresponde ao número de vizinhos em ξ_i. Como resultado da interação no jogo, cada nodo obtém taxas de *download* e *upload* na interação it, representadas por $down_i(it)$ e $up_i(it)$, respectivamente. Seja $c(it)$ uma função "ad hoc" de "resfriamento" na troca de pacotes, e sejam $g_i(it)$, $x_i(it)$, $n_i(it)$ e $f_j(it)$ funções auxiliares, as quais correspondem às variáveis g, x, n e f descritas no algoritmo 1. E sejam $S_i^{(d)}(it)$ e $S_j^{(u)}(it)$ funções auxiliares de sinal para $down_i(it)$ e para $up_i(it)$, ou seja, funções que serão utilizadas nas equações 6.1 e 6.2 para que se tenha uma soma ou uma subtração, dependendo do caso. Desta forma, sejam as funções auxiliares definidas como segue.

$$S_i^{(d)}(it) = \begin{cases} 1 & \text{, se } down_i(it) < T_s \\ -1 & \text{, se } down_i(it) \geq T_s \end{cases}$$

$$S_j^{(u)}(it) = \begin{cases} 1 & \text{, se } up_j(it) < a_{\max}.T_s \\ -1 & \text{, se } up_j(it) > a_{\min}.T_s \end{cases}$$

$$g_i(it) = g_{max}.x_i(it).c(it) + g_{min}$$

$$x_i(it) = U[0, 1]$$

$$n_i(it) = U[1, |\xi_i|]$$

$$f_j(it) = \frac{g_i(it)}{|n_i(it)|}$$

Então, $down_i(it)$ e $up_j(it)$ são definidos como:

$$down_i(it+1) = \begin{cases} down_i(it) + S_i^{(d)}(it) \sum_{j=1}^{n_i} S_j^{(u)}(it) f_j(it), \\ \quad \text{se } \exists\ j \in n_i \text{ tal que } a_{\min}.T_s < up_j(it) < a_{\max}.T_s \\ down_i(it), \text{ caso contrário} \end{cases} \tag{6.1}$$

e

$$up_j(it+1) = \begin{cases} up_j(it) + S_j^{(u)}(it) f_j(it), \\ \quad \text{se } \exists\ j \in n_i \text{ tal que } a_{\min}.T_s < up_j(it) < a_{\max}.T_s \\ up_j(it), \text{ caso contrário} \end{cases} \tag{6.2}$$

6.4.1 A função de utilidade

Em Economia, a utilidade pode ser entendida como uma medida relativa de satisfação no consumo de vários tipos de bens ou serviços. Dada esta medida, é possível falar de forma significativa em aumentar e diminuir a utilidade, e através disso explicar o comportamento em termos de tentativa de aumentar a utilidade de um indivíduo.

Nesta seção, uma função de utilidade simples é proposta para capturar a utilidade (também chamada de ganho ou *payoff*) recebida pelos usuários que participam de aplicações de *live streaming* em redes P2P. Os nodos interagem no jogo descrito na subseção 6.2 e, como resultado, cada nodo obtém taxas de *download* e *upload*. Assume-se neste trabalho que os usuários estão interessados, em primeiro lugar, em manter uma boa qualidade de *playback* (manter um bom índice de continuidade) enquanto assistem à transmissão, e, em segundo lugar, em economizar largura de banda de *upload*. O índice de continuidade está diretamente relacionado à qualidade do *playback*, e é definido como o número de pacotes que chegam antes do *deadline* de *playback*, sobre o número total de pacotes transmitidos pela fonte por intervalo de tempo. Tal função de utilidade valoriza o "entretenimento" oferecido pela aplicação, e o objetivo é capturar o interesse dos usuários de *Internet* em assistir à transmissão de *live streaming*, e ainda assim economizar largura de banda (sempre que possível) para utilizar em outras aplicações que consomem largura de banda de *upload*, tais como aplicações de VoIP e de compartilhamento de arquivos. Assume-se, então, que os usuários com altas taxas de *download* e baixas taxas de *upload* estão mais satisfeitos, e consequentemente obtêm uma utilidade maior. Desta forma, a função de utilidade deste modelo será definida levando em consideração as duas grandezas envolvidas no jogo: o *download* e o *upload* realizados pelos nodos em cada interação. Apenas para facilitar o entendimento da função de utilidade, a qual será definida na equação 6.5, esta será dividida em duas subfunções, chamadas (apenas no contexto desta dissertação) de "utilidades parciais": i) a primeira subfunção, mostrada na equação 6.3, estabelece a "utilidade parcial" do jogador com relação ao *download* na interação it; ii) a segunda subfunção, mostrada na equação 6.4, estabelece a "utilidade parcial" do jogador com relação ao *upload* na interação it. Assim, as funções de utilidade parciais com

relação ao *download* ($u\_down_i(it)$) e *upload* ($u\_up_j(it)$), respectivamente, são definidas como segue.

$$u\_down_i(it) = \frac{1}{1 + \exp(s(\varphi - down_i(it)))} \tag{6.3}$$

e

$$u\_up_i(it) = \exp(-r(\frac{up_i(it)}{a_{\max}.T_s})) \tag{6.4}$$

Como pode ser observado na equação 6.3, foi escolhida uma função logística para modelar a utilidade parcial com relação ao *download*. A função logística é utilizada para modelar a satisfação do usuário em relação ao *download* por dois motivos: i) ela naturalmente retorna valores entre [0,1], o que é interessante no momento de estabelecer a dinâmica evolucionária com base na utilidade (ou satisfação) do usuário; ii) ela satura a partir de um determinado ponto, característica que é interessante porque a taxa de *download* naturalmente tem um limite superior em *live streaming* - os nodos não têm interesse em fazer *download* de mais pacotes do que os transmitidos pela fonte, mesmo que de forma pontual (em determinada interação it) o nodo, por questões como perda de pacotes, faça temporariamente *download* abaixo ou acima da taxa de *stream*. A variável φ corresponde à taxa de *download* mínima para que o usuário consiga assistir de maneira satisfatória à transmissão. Assim, φ é um parâmetro de satisfação com relação à transmissão. Desta maneira, a equação 6.3 (a utilidade parcial do usuário em relação ao *download* na interação it) pode ser vista como um indicativo de quão satisfeito o usuário está com a taxa de *download*. O parâmetro s é utilizado para controlar quão fortemente ocorre o decréscimo na inflexão da função logística, ou seja, s é um parâmetro utilizado para realizar hipóteses sobre o crescimento/decrescimento da satisfação do usuário à medida que a taxa de *download* cresce/decresce. Como exemplo numérico para explicar o efeito de s na na equação 6.3, seja φ = 475 kbps. A figura 6.1 apresenta diferentes curvas da função logística para diferentes valores de s, onde, no eixo Y, tem-se os valores retornados de u_down para diferentes *downloads* ($down$), no eixo X.

A equação 6.4 representa a utilidade parcial do usuário com relação ao *upload* na interação it. Na função de utilidade, a ser apresentada na equação 6.5, a utilidade parcial do usuário com relação ao *upload* será utilizada como um "redutor" da utilidade parcial com relação ao *download*. Como pode ser observado na equação 6.4, foi escolhida uma função com decaimento exponencial para modelar a utilidade parcial com relação ao *upload*. A função exponencial é utilizada porque é uma função bastante simples com a qual é possível obter valores entre $[0, 1]$ utilizando-se taxas normalizadas de *upload* ($up_i(it)/a_{\max}.T_s$). Com a equação 6.4, tem-se que: i) sempre que a taxa de *upload* normalizada for igual a 0 (zero), a equação retorna 1 (um), indicando que a utilidade final do usuário não será reduzida; ii) para taxas de *upload* normalizadas maiores que zero, a equação retorna valores < 1, indicando que a satisfação final do usuário será reduzida de algum valor. De forma resumida, quanto maior a taxa de *upload* do nodo (up_i), maior será o fator de redução da utilidade parcial do *upload*. O parâmetro r é utilizado para controlar a intensidade do decaimento da função exponencial, e, consequentemente, a intensidade do fator de redução da equação 6.4, permitindo realizar hipóteses sobre o decrescimento da satisfação final do usuário à medida que a taxa de *upload* cresce. Como exemplo numérico para explicar o efeito de r na equação 6.4, sejam T_s = 500 kbps e $a_{\max}$ = 1,1. A

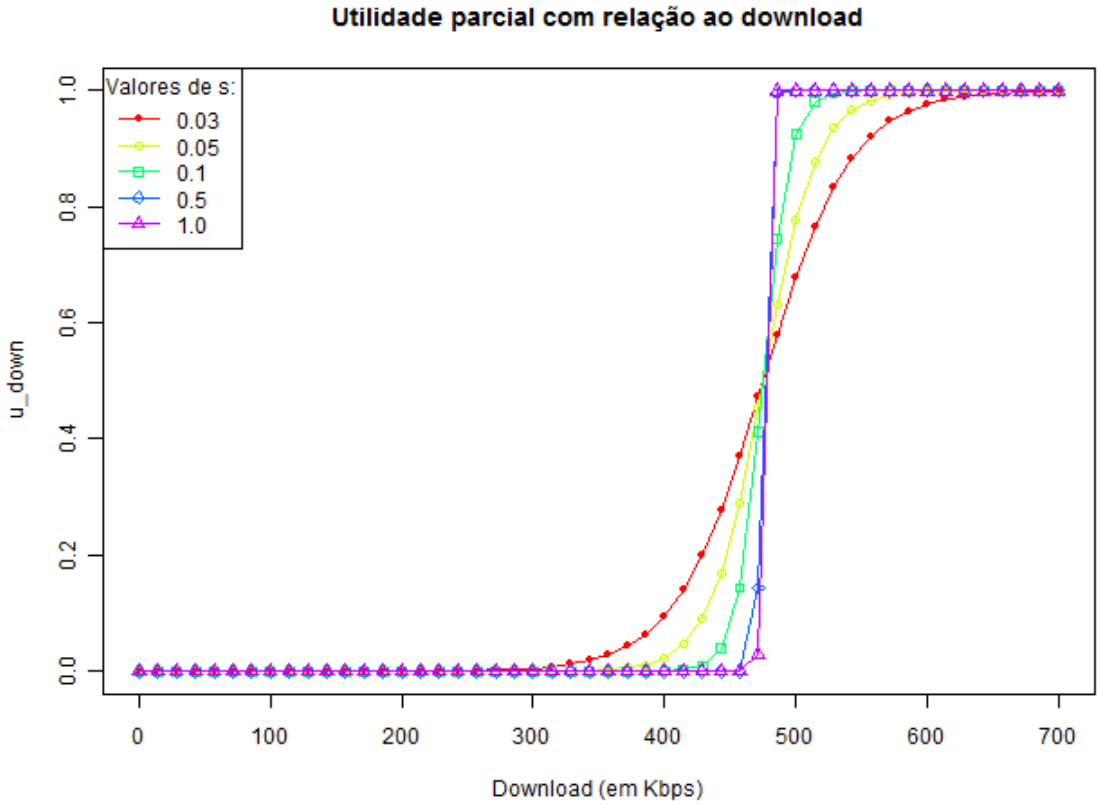

Figura 6.1: Efeito de s na utilidade parcial com o *download* (equação 6.3) - diferentes valores de s produzem curvas logísticas com diferentes graus de inflexão

figura 6.2 apresenta diferentes curvas da função exponencial para diferentes valores de r, onde, no eixo Y, tem-se os valores retornados de u_up para diferentes *uploads* (up), no eixo X.

Tendo como ponto de partida as equações 6.3 (a utilidade parcial com o *download*) e 6.4 (a utilidade parcial com o *upload*), a utilidade (final) recebida por cada jogador é definida em termos de $u\_down_i(it)$ (satisfação com relação ao *download*) e $u\_up_i(it)$ (redução da satisfação com o aumento do *upload*) da seguinte forma:

$$u_i(it) = u\_down_i(it) \; . \; u\_up_i(it) \tag{6.5}$$

Assim como as equações das utilidades parciais, a função de utilidade, conforme definida na equação 6.5, retorna valores entre [0,1], onde $u_i(it)$ = 1 é a utilidade (satisfação) máxima e $u_i(t)$ = 0 é a utilidade (satisfação) mínima. Como exemplo numérico da função de utilidade, sejam φ = 475 kbps, T_s = 500 kbps, $a_{\max}$ = 1,1, r = 0,5 e s = 0,05. Então, os valores de retorno da função de utilidade apresentada na equação 6.5, para diferentes valores de taxas de *download* e *upload*, são mostrados na figura 6.3.

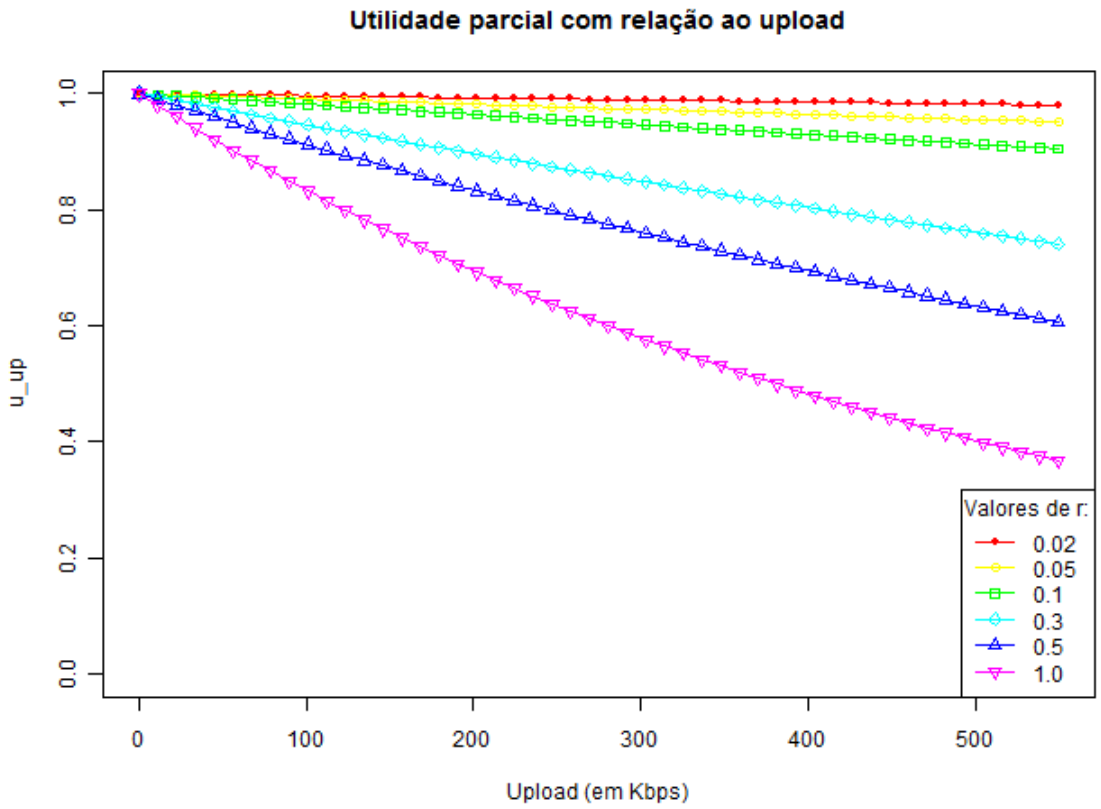

Figura 6.2: Efeito de r na utilidade parcial com o *upload* (equação 6.4) - diferentes valores de r produzem curvas exponenciais com diferentes intensidades de decaimento

6.4.2 A dinâmica da população

De acordo com Weibull (1997), o processo evolutivo combina um mecanismo de mutação, que provê variedade de estratégias, e um mecanismo de seleção, que favorece algumas estratégias em relação a outras. A estabilidade evolutiva destaca o papel da mutação, e a dinâmica de replicação destaca o papel da seleção. A estabilidade evolutiva, em ambientes sociais como uma aplicação de *live streaming*, pode ser pensada como uma convenção. Em uma aplicação de *live streaming*, a estabilidade evolutiva requer que qualquer grupo pequeno de usuários que tente uma estratégia alternativa seja menos bem sucedido que os outros usuários que jogam a estratégia *status quo*. Consequentemente, os usuários que jogam a estratégia *status quo* não têm incentivos para mudar de estratégia, já que eles são mais bem sucedidos que os usuários que tentam a estratégia alternativa. E os usuários que tentam a estratégia alternativa têm um incentivo para retornar à estratégia *status quo*. Para aplicações de *live streaming*, a replicação por meio de reprodução biológica (número de descendentes que herdam a estratégia do pai) pode não ser adequada para explicar como as estratégias se espalham na população, visto que a aplicação de *live streaming* é um ambiente social de interação entre indivíduos.

Apesar de a aplicação cliente de um usuário trocar dados somente com um subconjunto de nodos da rede, para o usuário esta estrutura é irrelevante, visto que o usuário não tem conhecimento de quem são os usuários por detrás dos nodos vizinhos. Desta forma, a estrutura da rede não será levada em consideração no processo dinâmico de ajuste das estratégias. Será considerado que potencialmente os usuários têm acesso a qualquer outro

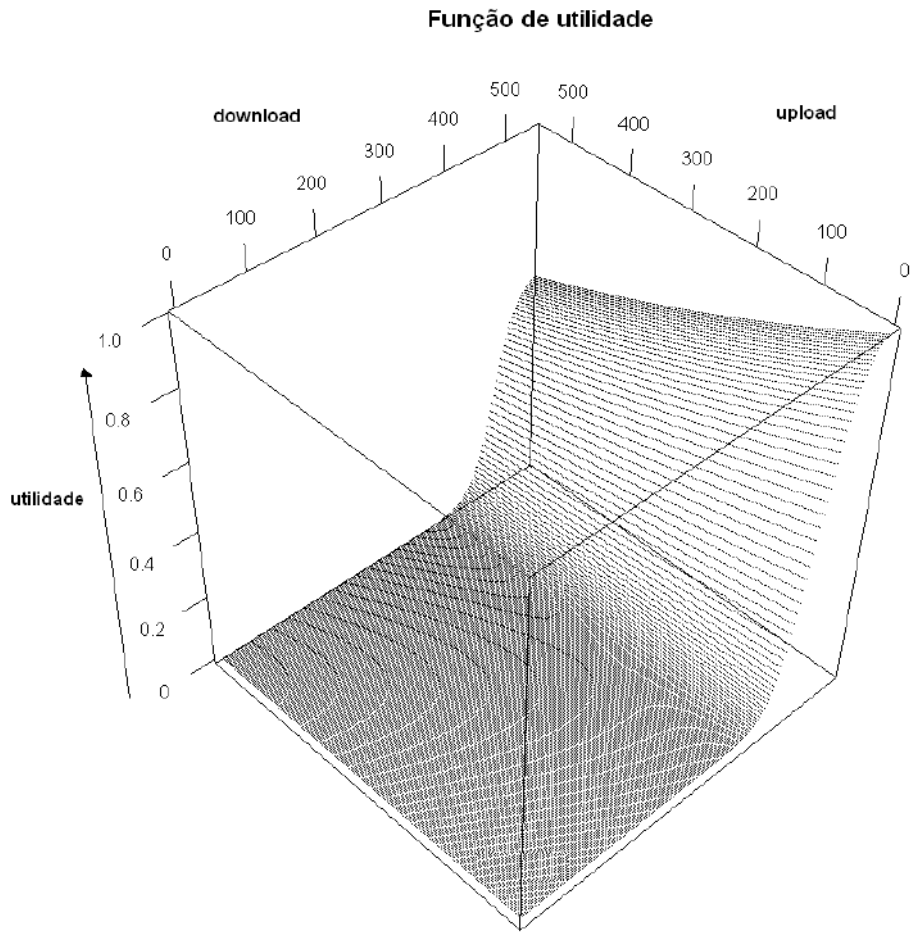

Figura 6.3: Função de utilidade (equação 6.5)

usuário da aplicação, já que, além da interação pessoal entre usuários, parte do conhecimento é adquirido através de fóruns e grupos sociais na *Internet*.

O processo dinâmico de ajuste das estratégias deste modelo é baseado nos trabalhos de Szabó e Hauert (2002), Szabó e Vukov (2004) e Huang et. al (2007), no qual os jogadores, com o objetivo de maximizar suas utilidades, reavaliam e alteram suas estratégias. A utilidade dos jogadores representa a adaptabilidade em um contexto social (como descrita na seção 2.2), ou seja, a probabilidade de que um jogador copie a estratégia de outro. Desta maneira, dadas as utilidades $u_i(it)$ e $u_j(it)$, um dado jogador i irá copiar a estratégia de um outro jogador j (aleatoriamente escolhido da população) com probabilidade w, tal que:

$$w = \frac{1}{1 + \exp[(u_i(it) - u_j(it))/k]} \tag{6.6}$$

onde k introduz algum ruído que ocasionalmente leva a decisões irracionais (por exemplo, a imitação de uma estratégia com utilidade pior). Para k = 0, a estratégia do jogador j é adotada deterministicamente pelo jogador i, contanto que $u_j(it) > u_i(it)$. Para $k > 0$, estratégias com desempenho pior podem ser adotadas com certa probabilidade.

7 SIMULAÇÃO E RESULTADOS

Este capítulo é destinado a apresentar o simulador de Teoria dos Jogos Evolucionários, desenvolvido a partir do modelo do capítulo 6, e também mostrar a análise dos resultados obtidos deste simulador. O desenvolvimento de aplicações de *live streaming* pode se tornar uma tarefa bastante desafiadora. Para que a aplicação consiga manter o serviço de maneira satisfatória, a aplicação de *live streaming* em redes P2P necessita de uma cuidadosa elaboração durante a fase de projeto, e monitoramento constante quando estiver implantada, ou seja, quando a aplicação estiver em pleno uso pelos usuários. Assim, o estudo de aplicações de *live streaming* pode ser facilitado utilizando-se simuladores. A vantagem de se usar simuladores é que eles permitem obter resultados satisfatórios sem despender recursos na criação do ambiente real do modelo a ser analisado. Além disto, podem ser realizados testes em larga escala que podem ser controlados e reproduzidos, característica de grande importância em *live streaming*, visto que a aplicação pode chegar a ter, simultaneamente, milhares de usuários. Tem-se ainda um maior controle sobre o tempo, permitindo que fenômenos possam ser estudados de forma acelerada. Desta forma, utilizar simulação é uma forma apropriada e de baixo custo de se determinar, com resultados muito próximos da realidade, o comportamento dos usuários da aplicação de *live streaming* em redes P2P.

De forma análoga à análise do Chainsaw, todos os gráficos deste capítulo são resultado de 30 execuções do simulador de TJE. As barras de erro possuem valores muito pequenos, e por simplicidade foram omitidas dos gráficos deste capítulo. Mais detalhes sobre o cômputo dos valores apresentados nos gráficos são dados nas seções correspondentes.

7.1 Descrição do simulador de TJE

Este simulador utiliza a linguagem Fortran e foi desenvolvido especialmente para esta dissertação. É um simulador do modelo de Teoria dos Jogos Evolucionários descrito no capítulo 6, e é utilizado neste capítulo para avaliar o modelo proposto. O simulador de TJE foi executado em um computador PC de arquitetura Intel, processador Intel Core 2 Duo de 1.83GHz, 2GB de memória RAM e sistema operacional Windows Vista Home Premium. A partir dos dados de entrada obtidos do simulador do Chainsaw, o simulador de TJE funciona em interações que correspondem a rodadas de simulação it, onde cada rodada de simulação representa um intervalo de tempo de transmissão do Chainsaw, e em três estágios, descritos a seguir.

- Estágio de interação - este estágio é executado em cada rodada it. Os nodos interagem conforme mostrado na subseção 6.2. Um nodo i solicita ajuda a seus vizinhos quando sua taxa de *download* está abaixo da taxa de *stream* da aplicação. A ajuda é recebida quando os nodos vizinhos de i aumentam o *upload*, representando que transmitem dados para o nodo i, o qual, em consequência, tem a sua taxa de *download* aumentada. De maneira análoga ao que acontece quando o *download*

está abaixo da taxa de *stream*, quando a taxa de *download* do nodo i está acima da taxa de *stream*, o nodo dispensa a ajuda extra; como resultado, ocorre redução no *upload* de seus vizinhos, representando que os vizinhos foram dispensados de transmitir alguns pacotes para o nodo i, que, consequentemente, tem sua taxa de *download* diminuída. Este estágio é executado em todas as rodadas, e, como resultado da interação, cada nodo obtém taxas de *download* e de *upload* na rodada it.

- Estágio de replicação - este estágio dá o caráter dinâmico da aplicação. Os jogadores alteram suas estratégias conforme descrito na subseção 6.4.2. Os jogadores, com o objetivo de maximizar a utilidade obtida por eles, reavaliam e alteram suas estratégias. A utilidade dos jogadores representa a adaptabilidade, ou seja, a probabilidade de que um jogador copie a estratégia de outro. Como consequência, a frequência das estratégias (percentual de jogadores utilizando uma determinada estratégia) evolui com o tempo, ou seja, a frequência das estratégias se altera com o tempo, indicando que a população evolui. Quando ativo, o estágio de replicação é executado em cada rodada it. Seria possível que a replicação acontecesse de maneira assíncrona aos outros estágios, porém, com o objetivo de não inserir novos aspectos dinâmicos que dificultariam a análise dos resultados, a análise de um estágio de replicação assíncrono não é abordada nesse trabalho.
- Estágio de ação do sistema de auditoria - o sistema de auditoria fica monitorando o estado da aplicação, interferindo quando o desempenho da aplicação é comprometido. Quando ativo, o sistema de auditoria é executado em cada rodada it. Mais detalhes sobre a ação do sistema de auditoria na aplicação são fornecidos na subseção 7.3.3.

Através de estudos preliminares, percebeu-se que diferentes topologias e estados iniciais de *download* e *upload* utilizados para carregar o simulador de TJE produzem pouco efeito nos resultados finais do simulador. As características da variância de *download* e *upload* do simulador de TJE, carregados com o mesmo conjunto de parâmetros iniciais do Chainsaw, não se alteram significativamente com a mudança da topologia ou da condição inicial. Isto se deve ao fato das topologias criadas pelo Chainsaw (com mesmo número de nodos e mesmo número de vizinhos) serem semelhantes, o que produz estados iniciais de *download* e *upload* também bastante semelhantes. Desta forma, a topologia da rede e o estado inicial de *download* e *upload* do simulador de TJE não serão variados durante as simulações. A partir da topologia e estado inicial carregados, o simulador de TJE é executado 30 vezes. Cada uma das 30 execuções utiliza uma semente diferente para a geração de números pseudo-aleatórios, o que produz 30 execuções diferentes.

Alguns parâmetros são pré-estabelecidos porque são baseados no simulador do Chainsaw, por isso não são variados nas simulações deste capítulo. Estes parâmetros são descritos na tabela 7.1. Outros parâmetros não são baseados no simulador do Chainsaw (são privativos do simulador de TJE) e são descritos na tabela 7.2.

7.2 Análise estatística do *download* e *upload* do simulador de TJE

Nesta seção, é apresentada uma análise estatística do simulador de TJE. Assim como a análise estatística realizada no simulador Chainsaw, esta análise tem enfoque nas propriedades de *upload* e *download*. Para obter resultados comparáveis aos do Chainsaw, somente o estágio de interação está ativo no simulador de TJE. Os efeitos dos outros estágios serão apresentados nas próximas seções. Os gráficos desta seção apresentam a média da variância de *download* e *upload*. O cômputo da média das variâncias e do erro das 30

Tabela 7.1: Parâmetros do simulador de TJE obtidos do Chainsaw

Parâmetro do Chainsaw	**Parâmetro no sim. de TJE**	**Valor no sim. de TJE**
N	-	-
PACKETSIZE	packet	10 kb
MCASTRATE	T_s	500 kbps
NONSENDERATTENDRATIO	a_{max}	1,1 . T_s = 550 kbps
SENDERATTENDRATIO	s_ratio	4,0 . T_s = 2.000 kbps
SEEDNEIGHBORS	s_neighbor	20
kMax	max_neighbor	6
MCASTTRANSMISSIONTIME	ntime	570 interações
capacidade mínima de upload	a_{min}	0,75 . T_s = 375 kbps

Tabela 7.2: Parâmetros privativos do simulador de TJE

Parâmetro	**Descrição**
φ	Taxa mínima de *download* para que o usuário assista à transmissão de maneira satisfatória
s	Controla a inflexão da equação 6.3 da utilidade parcial com o *download*
r	Controla o decaimento da equação 6.4 da utilidade parcial com o *upload*
k	Controla a quantidade de ruído na dinâmica da população
β	Controla o decaimento da função de "resfriamento"
g_{max} e g_{min}	Quantidade máxima de dados trocada entre os nodos
τ	Número de rodadas de punição por mau comportamento
λ	Limiar do sistema de auditoria

execuções do simulador de TJE é realizado de forma igual ao do simulador do Chainsaw (vide equações 5.5, 5.6 e 5.7). Como mencionado no início deste capítulo, as barras de erro são omitidas dos gráficos porque os valores obtidos são muito pequenos.

Como observado na análise estatística do Chainsaw feita na seção 5.2, existe um decaimento da variância de *upload* e *download* dos nodos da aplicação. A função $c(it)$ corresponde a uma função "ad hoc" de "resfriamento" na troca de dados utilizada para tentar capturar esta característica do Chainsaw. Como o decaimento da variância no Chainsaw segue uma lei de potência, conforme mostrado na seção 5.2, será analisada uma função $c(it)$ de lei de potência, tal que:

$$c(it) = it^{\beta} \tag{7.1}$$

onde β é um parâmetro utilizado para controlar o decaimento da função.

7.2.1 Função $c(it)$ com lei de potência

Nesta subseção serão analisados os efeitos de β, g_{max} e g_{min} (parâmetros da função $c(it)$) nas características de *download* e *upload* do modelo. Os valores estudados para cada parâmetro são apresentados na tabela 7.3, e foram determinados a partir de observações preliminares do efeito da função de resfriamento no *download* e *upload*.

Inicialmente, será estudado o efeito de β, e para isso g_{max} e g_{min} são pré-fixados com

valores iguais a 20 e 10, respectivamente. O número de nodos também será mantido constante, com $N = 1.000$. O efeito de escala nas características de *download* e *upload* será estudado na subseção 7.2.2.

Tabela 7.3: Intervalo de valores dos parâmetros da função $c(it)$

Parâmetro	**Valores estudados**
β	-0,10; -0,15; -0,20; -0,25 e -0,30
g_{max}	10; 20; 30 e 40
g_{min}	0; 5; 10; 15 e 20

A figura 7.1 apresenta a média da variância de *download* e os respectivos valores de θ_{down} para os diferentes valores de β estudados. Assim como no capítulo 5, θ_{down} representa o expoente da lei de potência da variância do *download*.

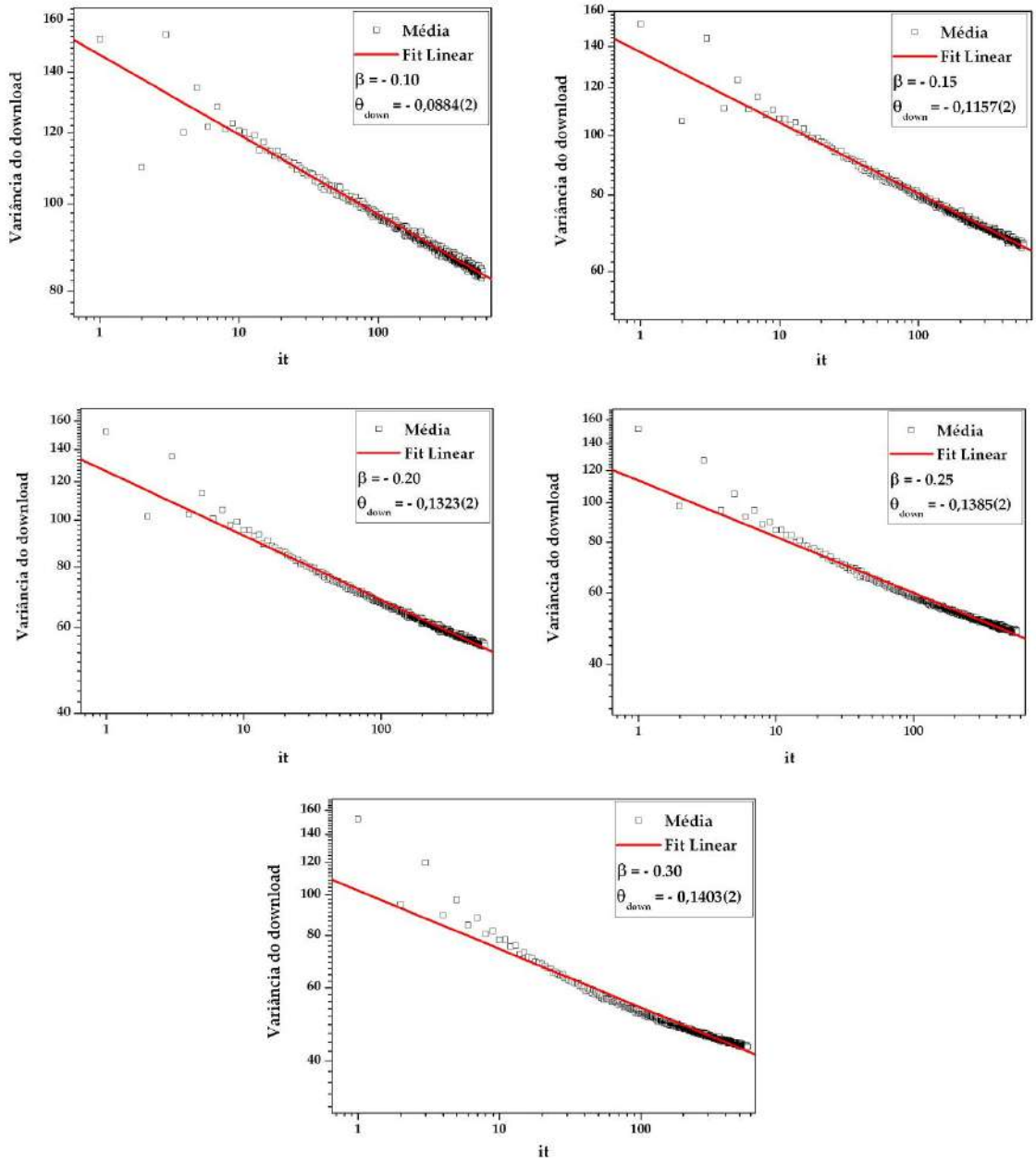

Figura 7.1: Valores de θ_{down} para diferentes valores de β, com g_{max} = 20 e g_{min} = 10 pré-fixados

Como pode ser observado na figura 7.1, o decaimento da variância de *download* apresenta uma forma bastante regular de lei de potência. Além disto, o decaimento da variância de *download* parece ser direcionado por β. θ_{down} é máximo para o maior valor estudado de β (θ_{down} = -0,0884(2) quando β = -0,10), e θ_{down} é mínimo para o menor valor estudado de β (θ_{down} = -0,1403(2) quando β = -0,30).

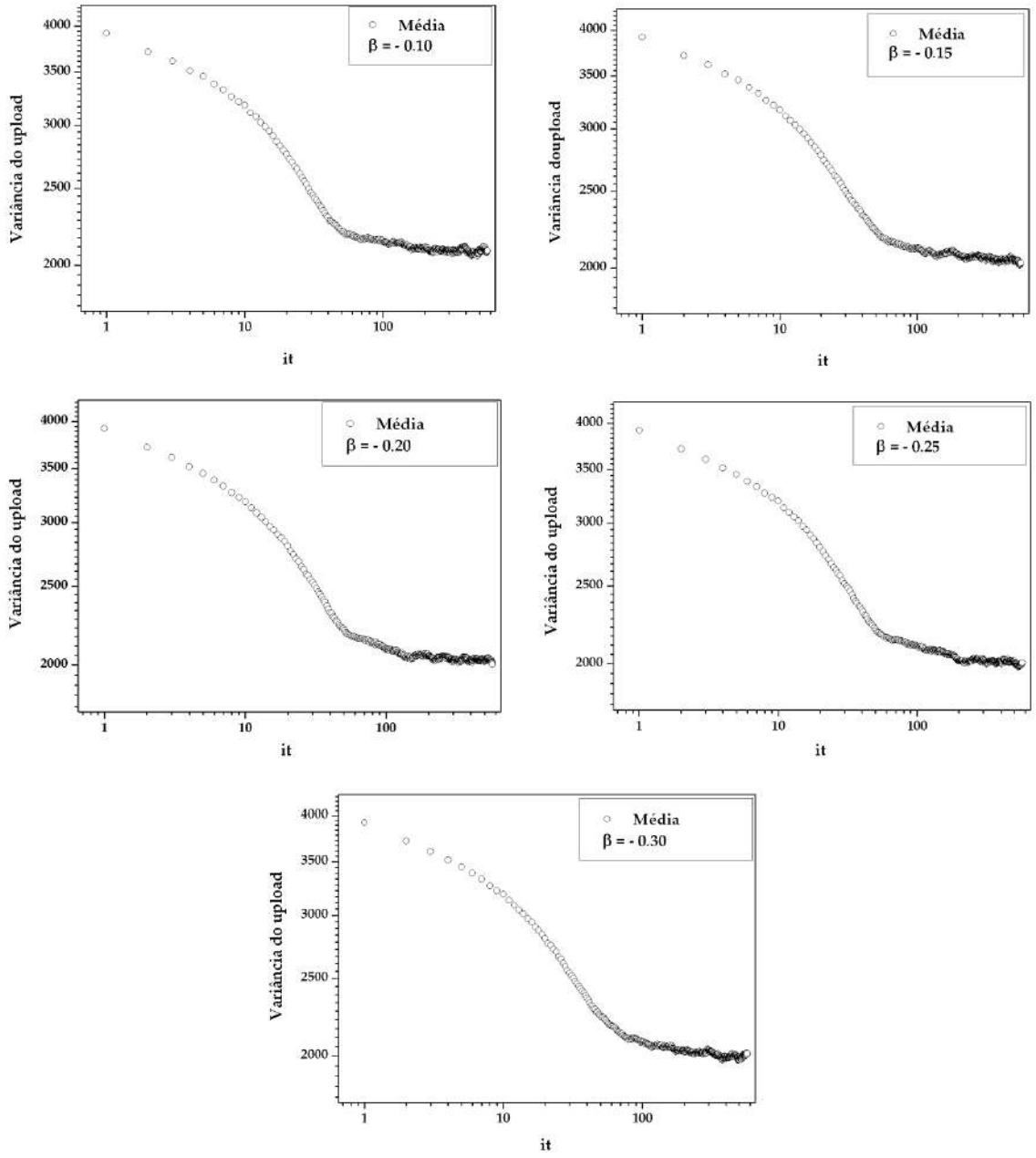

Figura 7.2: Variância do *upload* para diferentes valores de β, com g_{max} = 20 e g_{min} = 10 pré-fixados

Diferentemente da variância do *download*, que segue uma lei de potência, pode ser observado, através da figura 7.2, que o decaimento da variância de *upload* não segue uma lei de potência, independentemente do valor de β. Desta forma, não é possível realizar o *fit* linear para calcular o valor de θ_{up} nos gráficos da figura 7.2.

As variações de g_{max} e g_{min} não alteram as características das variâncias de *download*

e *upload*, por isso serão apresentados apenas os valores de θ_{down}, já que o *upload* não segue uma lei de potência. Os gráficos completos que mostram a média da variância de *downlaod* e *upload*, para os diferentes valores de g_{max} e g_{min}, são apresentados no apêndice desta dissertação.

Para analisar o efeito de g_{max} e g_{min} em θ_{down}, o valor de β é pré-fixado em β = -0,20. Uma vez pré-fixado β, g_{min} é mantido constante e com valor igual a g_{min} = 10 para se analisar apenas do efeito de g_{max} em θ_{down}, e g_{max} é mantido constante em g_{max} = 20 para se analisar apenas o efeito de g_{min} em θ_{down}.

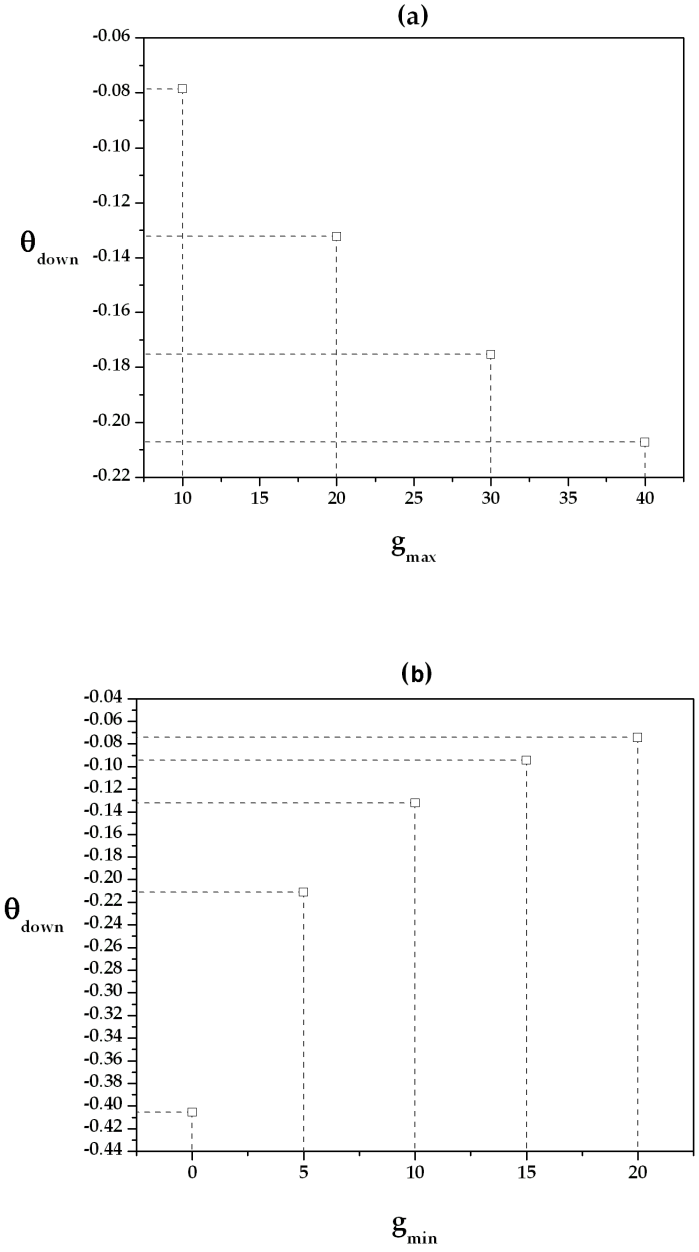

Figura 7.3: Efeito de g_{max} (figura (a)) e g_{min} (figura (b)) em θ_{down}

A figura 7.3.(a) apresenta os valores de θ_{down} para diferentes valores de g_{max}. Os resultados obtidos parecem indicar que g_{max} e θ_{down} são inversamente proporcionais. Quanto

maior é o valor de g_{max}, menor é o valor de θ_{down}. θ_{down} é máximo (θ_{down} = -0,0786) quando g_{max} é mínimo (g_{max} = 10), e θ_{down} é mínimo (θ_{down} = -0,2072) quando g_{max} é máximo (g_{max} = 40). A figura 7.3.(b) apresenta os valores de θ_{down} para diferentes valores de g_{min}. Os resultados indicam que g_{min} e θ_{down} são diretamente proporcionais. Quanto maior o valor de g_{min}, maior é o valor de θ_{down}. θ_{down} é mínimo (θ_{down} = -0,4054) quando g_{min} é mínimo (g_{min} = 0), e θ_{down} é máximo (θ_{down} = -0,0743) quando g_{min} é máximo (g_{min} = 20).

7.2.2 Efeito de escala

Com o objetivo de analisar o efeito de escala no decaimento da variância, todos os parâmetros da função de resfriamento $c(it)$ são mantidos constantes nesta seção. Como a variância do *upload* não segue uma lei de potência, os valores dos parâmetros foram escolhidos para formar uma combinação que propicie o melhor valor aproximado de θ_{down} do simulador de TJE com o valor observado de θ_{down} no Chainsaw. A combinação β = -0,20, g_{max} = 30 e g_{min} = 10 fornece um valor de θ_{down} = -0,1752(3) no simulador de TJE, o qual é o valor que mais se aproxima do valor de θ_{down} observado no Chainsaw (θ_{down} = -0,161(2)).

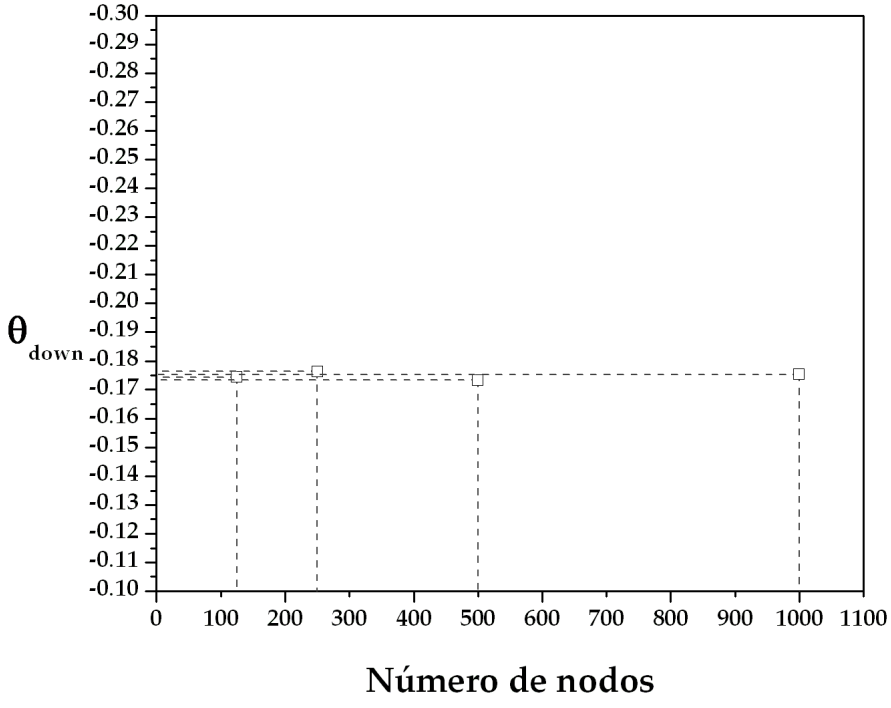

Figura 7.4: Efeito de escala: valor de θ_{down} em redes com 125, 250, 500 e 1.000 nodos

A figura 7.4 apresenta os valores de θ_{down} para diferentes tamanhos de rede. É possível perceber que o tamanho da rede provoca efeito negligenciável no valor de θ_{down}, que se mantém praticamente invariável (os valores de θ_{down} diferem-se apenas a partir da terceira casa decimal).

7.3 Resultados evolucionários do modelo

Nesta seção serão apresentados os resultados evolucionários do modelo. Como foi afirmado anteriormente, assume-se que todos os nodos permanecem por um tempo infinito conectados à aplicação. Alguns parâmetros são pré-estabelecidos e mantidos fixos em todas as simulações desta seção. Estes parâmetros são: número de nodos da rede (N = 1.000), os parâmetros da função $c(it)$ (β = -0,20, g_{max} = 30 e g_{min} = 10), e a taxa mínima de *download* para que o usuário assista à transmissão de maneira satisfatória (φ = 0,95.T_s - 95% da taxa de *stream*). O parâmetro φ é pré-estabelecido em 95% da taxa de *stream* porque alguns estudos parecem indicar que esta é uma taxa aceitável para que os usuários assistam à transmissão de forma satisfatória (XIE et al., 2007; ZHANG; LIU; LI, 2005).

7.3.1 Análise do comportamento pró-social na aplicação

O objetivo desta subseção é analisar o efeito que o comportamento pró-social tem na utilidade recebida pelos usuários que participam da aplicação. Desta forma, todos os usuários são corretos, ou seja, seguem o protocolo como dado originalmente. Para realizar esta análise, e como a população é homogênea, apenas o estágio de interação está ativo. O efeito dos nodos oportunistas na aplicação é apresentado na próxima seção.

Como exemplo numérico para esta subseção, os parâmetros s e r são mantidos fixos em s = 0,05 e r = 0,05. O efeito de s e r na aplicação será mostrado na próxima seção. Com s = 0,05 e r = 0,05, para um nodo qualquer i, a utilidade mínima aceitável, em uma interação it, é igual a $u_i(it)$ = 0,4756 (vide equação 6.5), que corresponde a utilidade recebida quando o *download* é igual a φ ($down_i(it) = \varphi.T_s$ = 0,95 . 500 = 475 kbps), e o *upload* é máximo ($up_i(it) = a_{max}.T_s$ = 1,1 . 500 = 550 kbps).

Sejam n_p o número de usuários da aplicação, $(j) \in n_p$ um usuário da aplicação, e n_s o número de execuções coletadas para análise (30 execuções). Então, a utilidade média ($u_{avg}(it)$) de todas as execuções é definida como segue.

$$u_{avg}(it) = \frac{1}{n_s.n_p} \sum_{i=1}^{n_s} \sum_{j=1}^{n_p} (u_i^{(j)}(it)) \tag{7.2}$$

Como pode ser observado na figura 7.5, o resultado do comportamento pró-social na aplicação é que a utilidade média permanece acima do nível aceitável, indicando que os usuários, em média, assistem à transmissão de forma satisfatória. A utilidade mínima aceitável varia dependendo dos valores de s e r, mas em uma população homogênea de cooperadores, a utilidade média sempre fica acima da utilidade mínima aceitável.

7.3.2 Efeito de nodos oportunistas na aplicação

Nesta subseção, será analisado o efeito de nodos oportunistas na aplicação de *live streaming*. Para realizar esta análise, serão realizadas hipóteses sobre os valores de s e r (os parâmetros que controlam as características da função de utilidade), com o objetivo de compreender como esses parâmetros influenciam a proliferação de nodos oportunistas na aplicação. Além disto, será analisado o efeito que os nodos oportunistas provocam no desempenho da aplicação, medindo-se o desempenho através da utilidade. Nesta subseção os estágios de interação e de replicação estão ativos. Com o estágio de replicação ativo, a frequência (percentual da população) de nodos corretos e oportunistas se altera com o tempo, indicando que existe uma evolução da população com o tempo. Com o objetivo de estudar o efeito dos nodos oportunistas na aplicação em um prazo mais longo, para esta

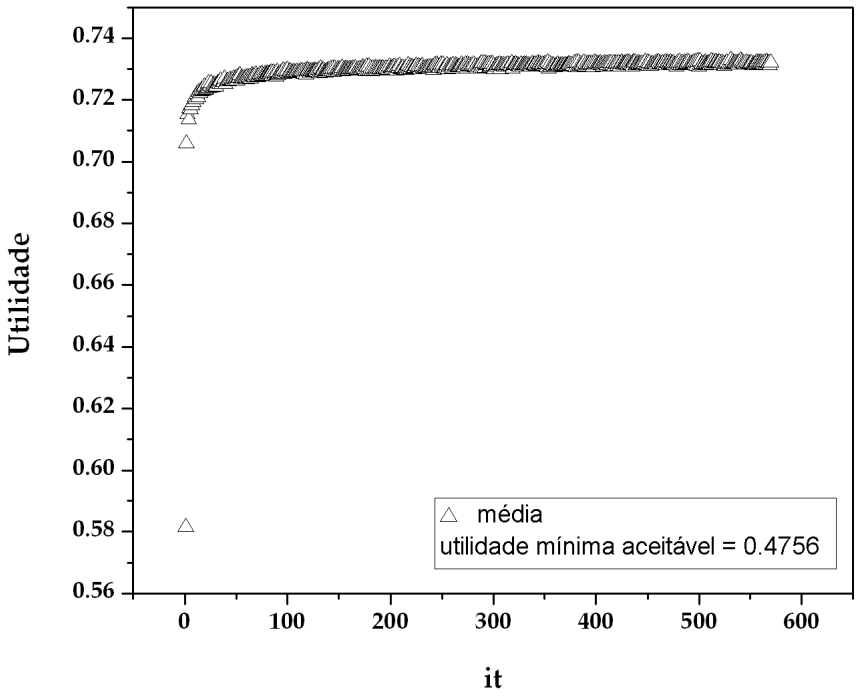

Figura 7.5: Comportamento pró-social: utilidade média

subseção o número de interações do simulador é aumentado para $ntime$ = 2.000. Neste trabalho será considerado apenas o efeito de nodos *freeriders*, ou seja, nodos oportunistas extremos que contribuem sempre a uma taxa $\omega = 0$. Seria possível estudar o efeito de nodos oportunistas menos radicais que os *freeriders* na aplicação. Porém, com o objetivo de não inserir novos aspectos que dificultem a análise dos resultados, o efeito de nodos oportunistas menos radicais na aplicação não é abordado neste trabalho. A utilidade média computada corresponde à utilidade calculada entre todos os usuários da aplicação (corretos e oportunistas).

Os parâmetros s e r controlam o comportamento da função de utilidade (como mostrado nas figuras 6.1, 6.2 e 6.3), de forma que s controla quão forte é a inflexão da função logística da utilidade parcial em relação ao *download*, e r controla o decaimento da função exponencial da utilidade parcial em relação ao *upload* (a qual funciona como redutor da utilidade à medida que o *upload* aumenta). O parâmetro k controla a quantidade de ruído que ocasionalmente leva a decisões irracionais no processo dinâmico de ajuste das estratégias. Seguindo o trabalho de Szabó e Hauert (2002), nesta dissertação o parâmetro k será pré-estabelecido em $k = 0,1$ e utilizado sem variação em todas as simulações evolutivas. O estudo do efeito de k na evolução da população será deixado para trabalhos futuros.

Para estudar o efeito dos nodos oportunistas na aplicação de *live streaming*, serão estudadas nove combinações de s e r, com três percentuais diferentes de nodos oportunistas na população inicial - 1%, 5% e 10%. Os percentuais escolhidos são relativamente baixos porque o objetivo é entender se mesmo uma população pequena de nodos oportunistas é capaz de causar danos à aplicação. Os valores estudados de s são: 0,05; 0,1 e 0,5 - estes valores foram escolhidos de forma a representarem curvas logísticas de inflexão suave,

intermediária e íngreme. Os valores de r são: 0,02; 0,05 e 0,3 - estes valores foram escolhidos de forma a representarem reduções de satisfação mínima, intermediária e um valor mais significativo.

A figura 7.6 apresenta os resultados do percentual médio de nodos oportunistas (gráficos da esquerda) e da utilidade média (gráficos da direita) de 30 execuções do simulador de TJE com os parâmetros s = 0,05 e r = 0,02. Para s = 0,05 e r = 0,02, a utilidade mínima aceitável é igual a u = 0,4901 (vide equação 6.5 para o cômputo da utilidade), que é a utilidade recebida quando o *download* é igual a φ e o *upload* é máximo.

Nos gráficos da figura 7.6, os círculos representam as execuções que convergiram para o estado estacionário em que todos os nodos são corretos. Pelos resultados mostrados, é possível observar que todas as execuções, mesmo com diferentes percentuais de nodos oportunistas na população inicial, convergiram para o estado em que os nodos são todos corretos. Algumas execuções convergiram mais rapidamente que outras, o que é uma consequência do processo estocástico no jogo e na dinâmica da população, mas todas convergiram em menos de 2.000 interações para o estado estacionário. Todas as execuções apresentam um baixo percentual médio de nodos oportunistas (menos de 6,2%) e uma utilidade média alta (maior que 0,57), acima da utilidade aceitável u = 0,4901. Quanto mais rapidamente a execução converge para o estado estacionário de nodos corretos, mais baixo é o percentual médio de nodos oportunistas e mais alta é a utilidade média da execução. O que se conclui desta combinação de s e r é que ela favorece a proliferação de nodos corretos na aplicação. Através da realização de testes adicionais, verificou-se que inclusive com 50% de nodos oportunistas na população inicial, todas as execuções convergem para o estado estacionário com somente nodos corretos.

A figura 7.7 apresenta os resultados do percentual médio de nodos oportunistas (gráficos da esquerda) e da utilidade média (gráficos da direita) das execuções do simulador de TJE com os parâmetros s = 0,05 e r = 0,05. Para s = 0,05 e r = 0,05, a utilidade mínima aceitável é igual a u = 0,4756 (vide equação 6.5 para o cômputo da utilidade), que é a utilidade recebida quando o *download* é igual a φ e o *upload* é máximo.

Nos gráficos da figura 7.7, os círculos representam as execuções que convergiram para o estado estacionário em que todos os nodos são corretos, os triângulos representam as execuções que convergiram para o estado estacionário em que todos os nodos são oportunistas, e os quadrados representam as execuções que não convergiram para um estado estacionário. Com 1% de nodos oportunistas na população inicial, sete execuções convergem para o estado estacionário de nodos corretos, uma converge para o estado estacionário somente com nodos oportunistas, e 22 execuções não convergem para um estado estacionário. Nas execuções que convergem para o estado estacionário de nodos corretos, a utilidade média fica acima da utilidade média aceitável e o percentual de nodos oportunistas é bastante próximo de zero, indicando que a convergência aconteceu rapidamente nestas execuções. A execução que alcança o estado estacionário em que todos os nodos são oportunistas apresenta a utilidade média mais baixa (0,142) e o maior percentual médio de nodos oportunistas (quase 60%). Nas execuções que não alcançam um equilíbrio (22 execuções), o percentual médio de nodos oportunistas fica em torno de 18%, e a utilidade média fica em torno de 0,32, abaixo da utilidade média aceitável, o que indica que as execuções não são satisfatórias para os usuários participantes. Com 5% e 10% de nodos oportunistas na população inicial, nenhuma das execuções alcança o estado estacionário em que os nodos são corretos, e a grande maioria das execuções (28 em ambos os casos) não alcança um estado estacionário. Com 5% de nodos oportunistas na população inicial, o percentual médio de nodos oportunistas fica em torno de 21%, e a utilidade média

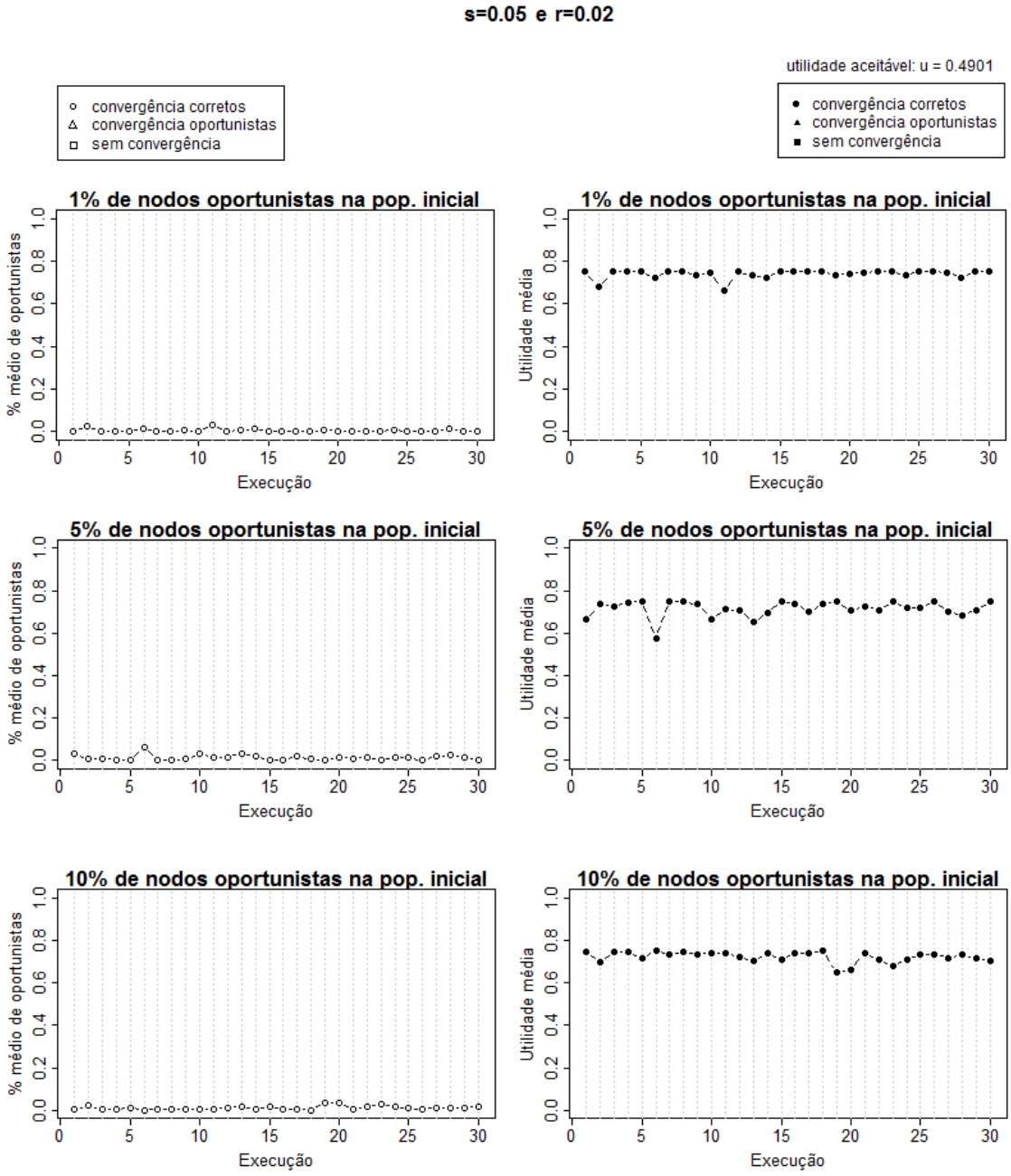

Figura 7.6: Percentual médio de nodos oportunistas (gráficos da esquerda) e utilidade média (gráficos da direita) de cada execução do simulador de TJE com os parâmetros s = 0,05 e r = 0,02

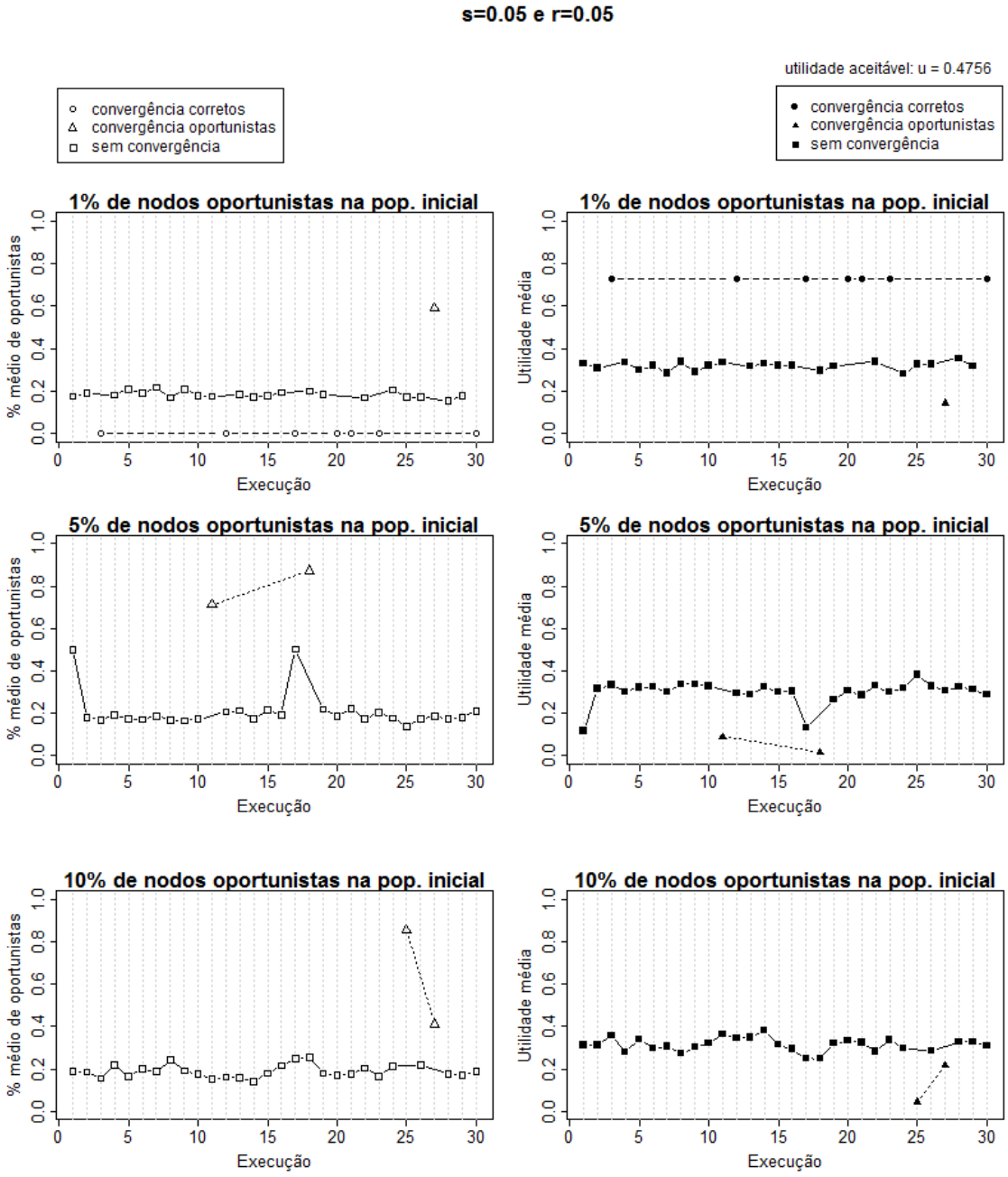

Figura 7.7: Percentual médio de nodos oportunistas (gráficos da esquerda) e utilidade média (gráficos da direita) de cada execução do simulador de TJE com os parâmetros s = 0,05 e r = 0,05

fica em torno de 0,30, abaixo da utilidade média aceitável, o que indica que as execuções não apresentam níveis satisfatórios para os usuários. Com 10% de nodos oportunistas na população inicial, o percentual médio de nodos oportunistas fica em torno de 19%, e a utilidade média fica em torno de 0,31, também abaixo da utilidade média aceitável. As execuções que alcançaram o estado estacionário em que todos os nodos são oportunistas também apresentam utilidade média abaixo da utilidade aceitável. O que se conclui desta combinação de s e r é que ela não parece favorecer a cooperação. Apesar de algumas execuções terem alcançado o estado estacionário em que todos os nodos são corretos (os círculos nos gráficos), e outras poucas execuções terem alcançado o estado estacionário em que todos os nodos são oportunistas (os triângulos nos gráficos), com esta combinação a grande maioria das execuções encontra-se em estados de não convergência, porém com percentual médio de nodos oportunistas suficiente para que as utilidades médias das execuções fiquem abaixo da utilidade aceitável.

A figura 7.8 apresenta os resultados do percentual médio de nodos oportunistas (gráficos da esquerda) e da utilidade média (gráficos da direita) das execuções do simulador de TJE com os parâmetros s = 0,05 e r = 0,3. Para s = 0,05 e r = 0,3, a utilidade mínima aceitável é igual a u = 0,3704 (vide equação 6.5 para o cômputo da utilidade), que é a utilidade recebida quando o *download* é igual a φ e o *upload* é máximo.

Nos gráficos da figura 7.8, os triângulos representam as execuções que convergiram para o estado estacionário em que todos os nodos são oportunistas, e os quadrados representam as execuções que não convergiram para um estado estacionário. Pelos resultados mostrados, é possível observar que praticamente todas as execuções, mesmo com diferentes percentuais de nodos oportunistas na população inicial, convergiram para o estado em que os nodos são todos oportunistas. Algumas execuções convergiram mais rapidamente que outras, o que é uma consequência do processo estocástico no jogo e na dinâmica da população. Quanto mais rapidamente a execução converge para o estado estacionário de nodos oportunistas, mais alto é o percentual médio de nodos oportunistas e mais baixa é a utilidade média da execução. As poucas execuções que não convergiram para um estado estacionário apresentam utilidade muito abaixo do aceitável e alto percentual médio de nodos oportunistas na execução (acima de 70%). Percebe-se também que o percentual médio de nodos oportunistas nas execuções está bastante disperso, não se concentrando, como observado em outras configurações, ao redor de um determinado valor. O que se conclui desta combinação de s e r é que ela favorece a proliferação de nodos oportunistas na aplicação.

A figura 7.9 apresenta os resultados do percentual médio de nodos oportunistas (gráficos da esquerda) e da utilidade média (gráficos da direita) das execuções do simulador de TJE com os parâmetros s = 0,1 e r = 0,02. Para s = 0,1 e r = 0,02, a utilidade mínima aceitável é igual a u = 0,4901 (vide equação 6.5 para o cômputo da utilidade), que é a utilidade recebida quando o *download* é igual a φ e o *upload* é máximo.

Nos gráficos da figura 7.9, os círculos representam as execuções que convergiram para o estado estacionário em que todos os nodos são corretos, e os quadrados representam as execuções que não convergiram para um estado estacionário. Pelos resultados mostrados, é possível observar que praticamente todas as execuções, mesmo com diferentes percentuais de nodos oportunistas na população inicial, convergiram para o estado em que os nodos são todos corretos. Com 1% de nodos oportunistas, 29 execuções convergem para o estado estacionário de nodos corretos, com 5% de nodos oportunistas, 27 convergem para o estado de nodos corretos, e com 10% de nodos oportunistas, todas as 30 execuções convergem para o estado estacionário de nodos corretos. Os números diferen-

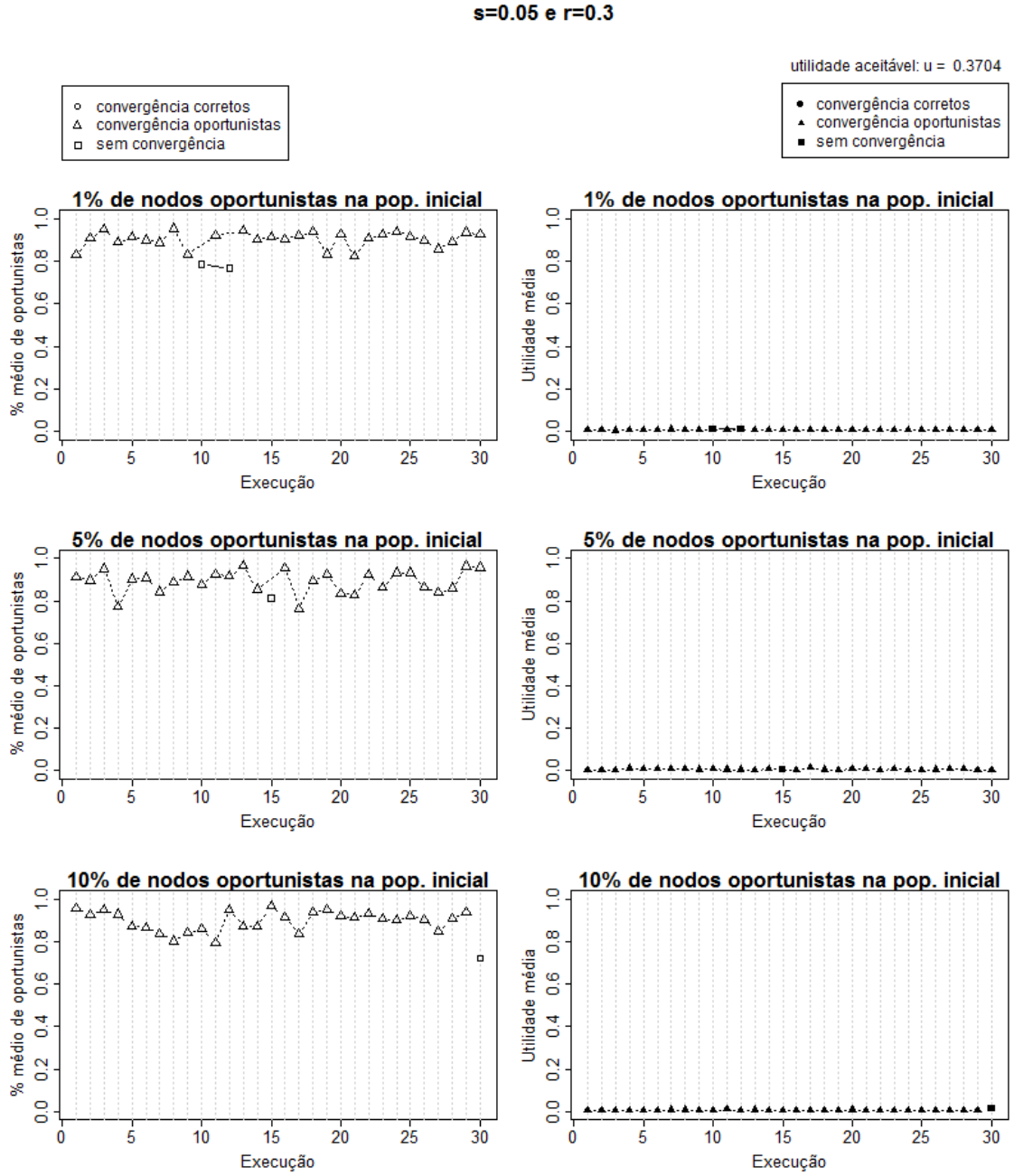

Figura 7.8: Percentual médio de nodos oportunistas (gráficos da esquerda) e utilidade média (gráficos da direita) de cada execução do simulador de TJE com os parâmetros s = 0,05 e r = 0,3

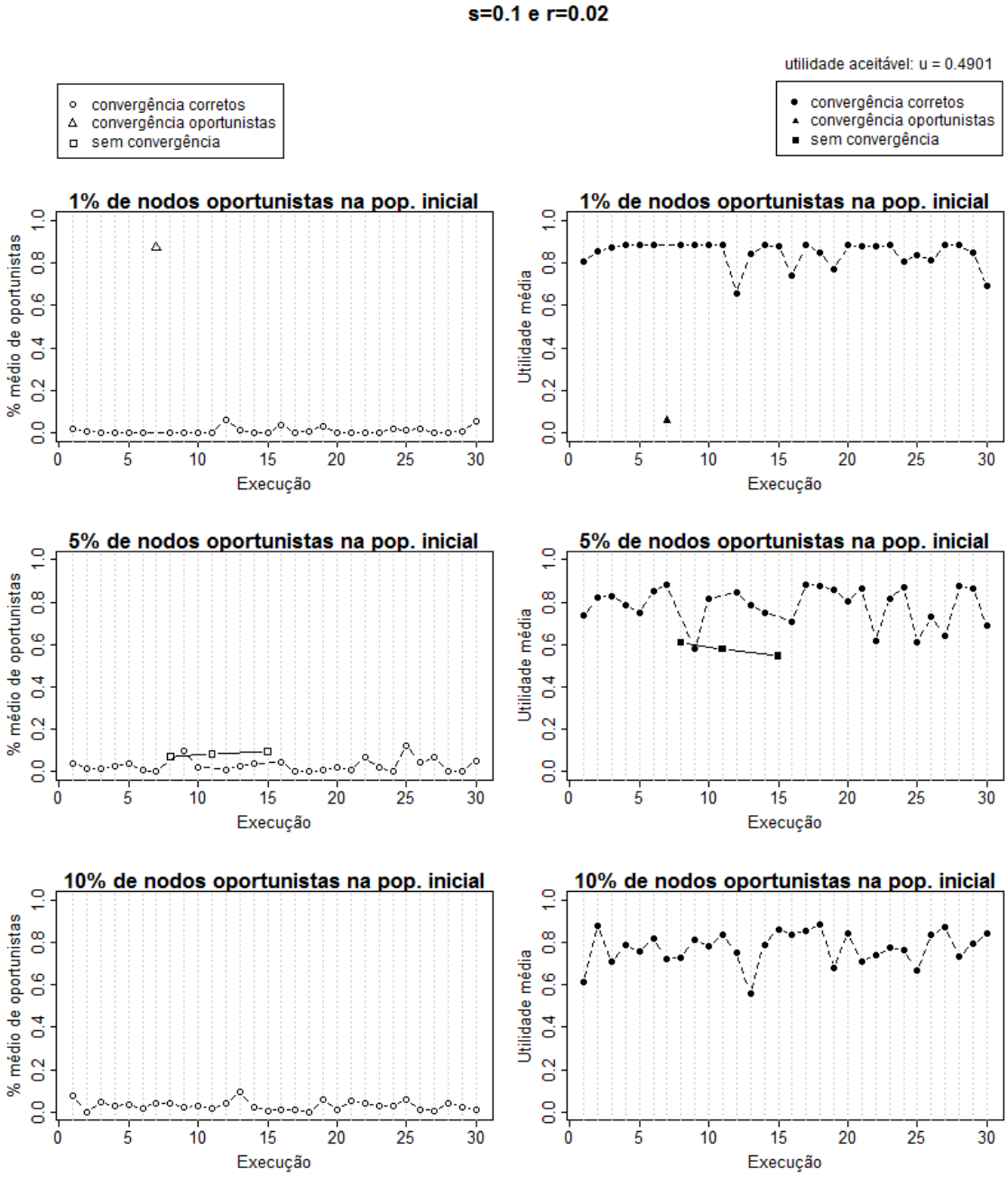

Figura 7.9: Percentual médio de nodos oportunistas (gráficos da esquerda) e utilidade média (gráficos da direita) de cada execução do simulador de TJE com os parâmetros s = 0,1 e r = 0,02

tes de convergência com cada percentual, bem como os diferentes pontos de convergência (algumas execuções convergiram mais rapidamente que outras), é consequência do processo estocástico intrínseco do jogo e da dinâmica da população. Todas as execuções que convergiram para o estado estacionário com nodos corretos apresentam baixo percentual médio de nodos oportunistas e utilidade média acima da utilidade aceitável, e mesmo as execuções que não convergiram apresentam baixo percentual médio de nodos oportunistas e utilidade média acima da utilidade aceitável. Assim, pode-se concluir que esta configuração de s e r também favorece a proliferação de nodos corretos na aplicação.

A figura 7.10 apresenta os resultados do percentual médio de nodos oportunistas (gráficos da esquerda) e da utilidade média (gráficos da direita) das execuções do simulador de TJE com os parâmetros s = 0,1 e r = 0,05. Para s = 0,1 e r = 0,05, a utilidade mínima aceitável é igual a u = 0,4756 (vide equação 6.5 para o cômputo da utilidade), que é a utilidade recebida quando o *download* é igual a φ e o *upload* é máximo.

Nos gráficos da figura 7.10, os círculos representam as execuções que convergiram para o estado estacionário em que todos os nodos são corretos, os triângulos representam as execuções que convergiram para o estado estacionário em que todos os nodos são oportunistas, e quadrados representam as execuções que não convergiram para um estado estacionário. Com 1% de nodos oportunistas na população inicial, sete execuções convergem para o estado estacionário de nodos corretos, seis convergem para o estado estacionário somente com nodos oportunistas, e 17 execuções não convergem para um estado estacionário. Nas execuções que convergem para o estado estacionário de nodos corretos, a utilidade média fica acima da utilidade média aceitável e o percentual de nodos oportunistas é bastante próximo de zero, indicando que a convergência aconteceu rapidamente nestas execuções. As execuções que alcançam o estado estacionário em que todos os nodos são oportunistas apresentam utilidades médias abaixo do aceitável e percentuais médios de nodos oportunistas variando entre 38% e 83%, indicando que o momento da convergência dos nodos oportunistas deu-se de forma dispersa, sem se concentrar próximo a algum valor. Nas execuções que não alcançam um equilíbrio (17 execuções), o percentual médio de nodos oportunistas fica em torno de 27%, e a utilidade média fica em torno de 0,23, abaixo da utilidade média aceitável, o que indica que as execuções não são satisfatórias para os usuários participantes. Com 5% de nodos oportunistas na população inicial, nenhuma das execuções alcança o estado estacionário em que os nodos são corretos, e a grande maioria das execuções não alcança um estado estacionário. Além disto, o percentual médio de nodos oportunistas fica em torno de 28%, e a utilidade média fica em torno de 0,22, abaixo da utilidade média aceitável, o que indica que as execuções não apresentam níveis satisfatórios para os usuários. Com 10% de nodos oportunistas na população inicial, nenhuma das execuções alcança o estado estacionário em que os nodos são corretos, e a grande maioria das execuções não alcança um estado estacionário. Além disto, o percentual médio de nodos oportunistas fica em torno de 31%, e a utilidade média fica em torno de 0,21, também abaixo da utilidade média aceitável. As execuções que alcançaram o estado estacionário em que todos os nodos são oportunistas também apresentam utilidade média abaixo da utilidade aceitável. O que se conclui desta combinação de s e r é que ela não parece favorecer a cooperação, pois apesar de algumas execuções terem alcançado o estado estacionário em que todos os nodos são corretos, outras alcançaram o estado estacionário em que todos os nodos são oportunistas, resultando em utilidade abaixo do aceitável. O restante das execuções não convergiram e apresentaram percentual médio de nodos oportunistas suficiente para que as utilidades médias das execuções ficassem abaixo da utilidade aceitável.

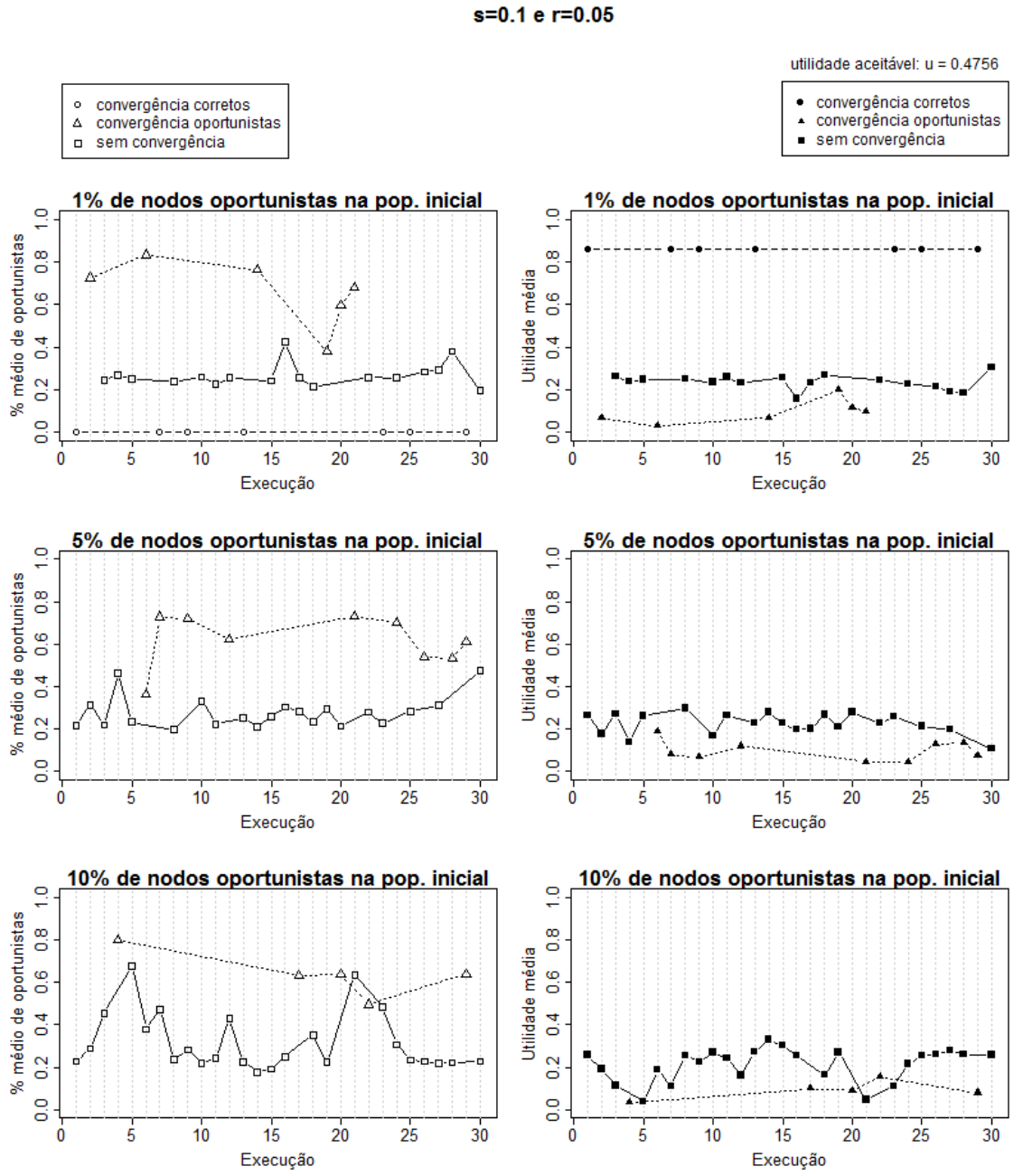

Figura 7.10: Percentual médio de nodos oportunistas (gráficos da esquerda) e utilidade média (gráficos da direita) de cada execução do simulador de TJE com os parâmetros s = 0,1 e r = 0,05

A figura 7.11 apresenta os resultados do percentual médio de nodos oportunistas (gráficos da esquerda) e da utilidade média (gráficos da direita) das execuções do simulador de TJE com os parâmetros s = 0,1 e r = 0,3. Para s = 0,1 e r = 0,3, a utilidade mínima aceitável é igual a u = 0,3704 (vide equação 6.5 para o cômputo da utilidade), que é a utilidade recebida quando o *download* é igual a φ e o *upload* é máximo.

Nos gráficos da figura 7.11, os triângulos representam as execuções que convergiram para o estado estacionário em que todos os nodos são oportunistas, e os quadrados representam as execuções que não convergiram para um estado estacionário. Pelos resultados mostrados, é possível observar que o resultado desta configuração é muito semelhante do resultado da configuração s = 0,05 e r = 0,3. Com os três diferentes percentuais de nodos oportunistas, nenhuma das execuções convergiu para o estado estacionário de nodos corretos, algumas poucas não convergiram, e praticamente todas as execuções convergiram para o estado em que os nodos são todos oportunistas. O percentual médio de nodos oportunistas nas execuções (que convergiram para nodos oportunistas e que não convergiram) está bastante disperso, não se concentrando ao redor de um determinado valor. As poucas execuções que não convergiram para um estado estacionário apresentam utilidade média abaixo do aceitável e alto percentual médio de nodos oportunistas na execução. Desta forma, conclui-se que esta combinação de s e r favorece a proliferação de nodos oportunistas na aplicação.

A figura 7.12 apresenta os resultados do percentual médio de nodos oportunistas (gráficos da esquerda) e da utilidade média (gráficos da direita) das execuções do simulador de TJE com os parâmetros s = 0,5 e r = 0,02. Para s = 0,5 e r = 0,02, a utilidade mínima aceitável é igual a u = 0,4901 (vide equação 6.5 para o cômputo da utilidade), que é a utilidade recebida quando o *download* é igual a φ e o *upload* é máximo.

Nos gráficos da figura 7.12, os círculos representam as execuções que convergiram para o estado estacionário em que todos os nodos são corretos, e os quadrados representam as execuções que não convergiram para um estado estacionário. Pelos resultados mostrados, é possível observar que, diferentemente do resultado obtido com a combinação de r = 0,02 com os outros valores menores de s, com s = 0,5 o número de execuções que convergem para o estado estacionário em que todos os nodos são corretos cai para aproximadamente a metade, mesmo com diferentes percentuais de nodos oportunistas na população inicial. As execuções que convergem para o estado estacionário com nodos corretos levam a utilidades médias acima do aceitável. Porém, os casos em que a convergência não ocorre, os valores da utilidade média e do percentual médio de nodos oportunistas varia um pouco dependendo do percentual inicial de oportunistas. Com 1% de nodos oportunistas, a utilidade média das execuções que não convergem fica em torno de 0,52 e o percentual médio de nodos oportunistas fica em torno de 12%. Isto indica que as execuções que não convergem ficam com utilidade pouco acima da utilidade aceitável. Com 5% de nodos oportunistas, a utilidade média das execuções que não convergem fica em torno de 0,50 e o percentual médio de nodos oportunistas fica em torno de 13%, indicando também que as execuções que não convergem ficam com utilidade acima da utilidade aceitável. Com 10% de nodos oportunistas, a utilidade média das execuções que não convergem fica em torno de 0,47 e o percentual médio de nodos oportunistas fica em torno de 16%, indicando que as execuções que não convergem ficam com utilidade pouco abaixo da utilidade aceitável. Os resultados indicam que, com esta configuração, a proliferação de nodos corretos ainda é favorecida, porém com menos força que as outras configurações de r = 0,02 com valores menores de s.

A figura 7.13 apresenta os resultados do percentual médio de nodos oportunistas (grá-

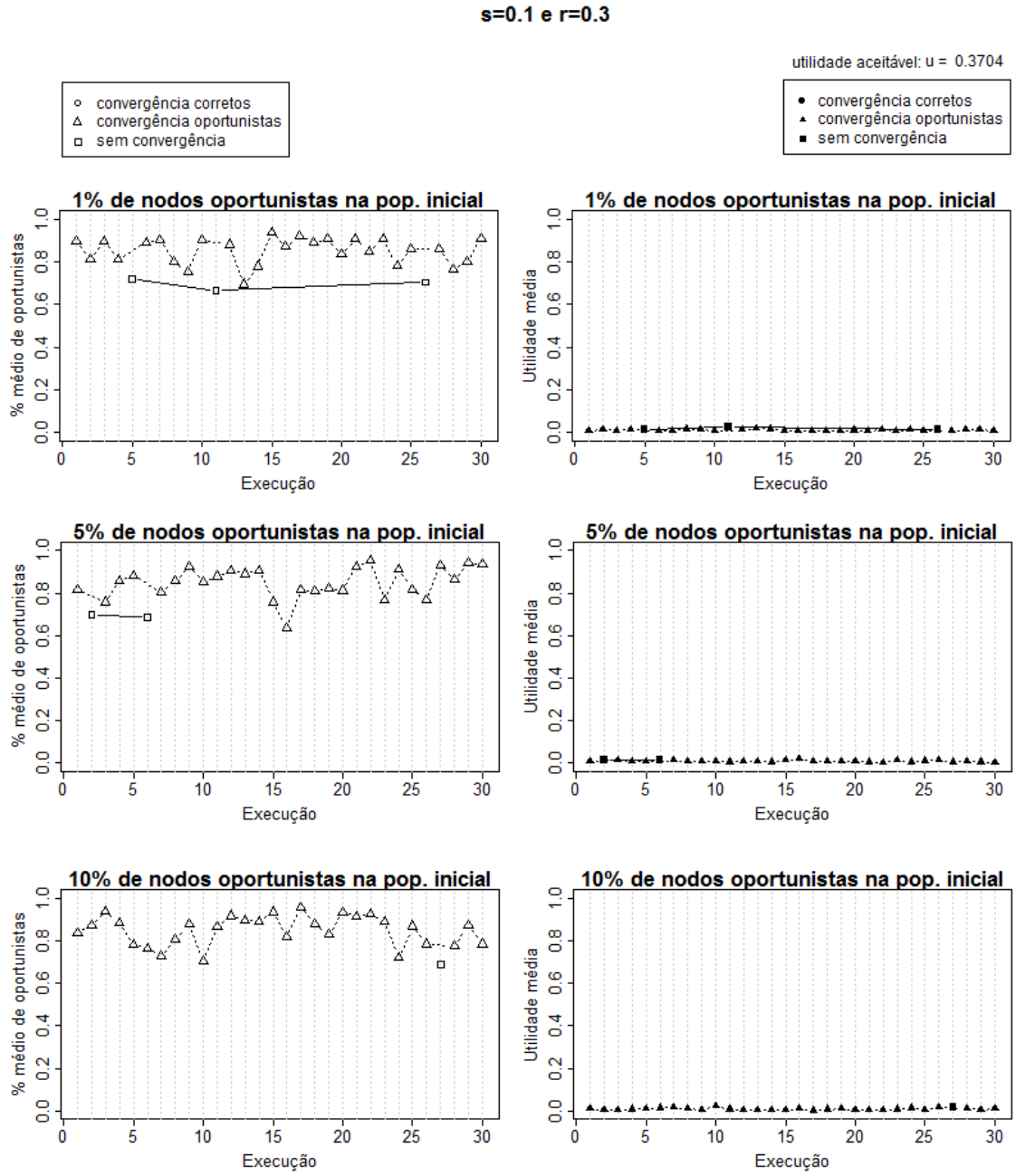

Figura 7.11: Percentual médio de nodos oportunistas (gráficos da esquerda) e utilidade média (gráficos da direita) de cada execução do simulador de TJE com os parâmetros s = 0,1 e r = 0,3

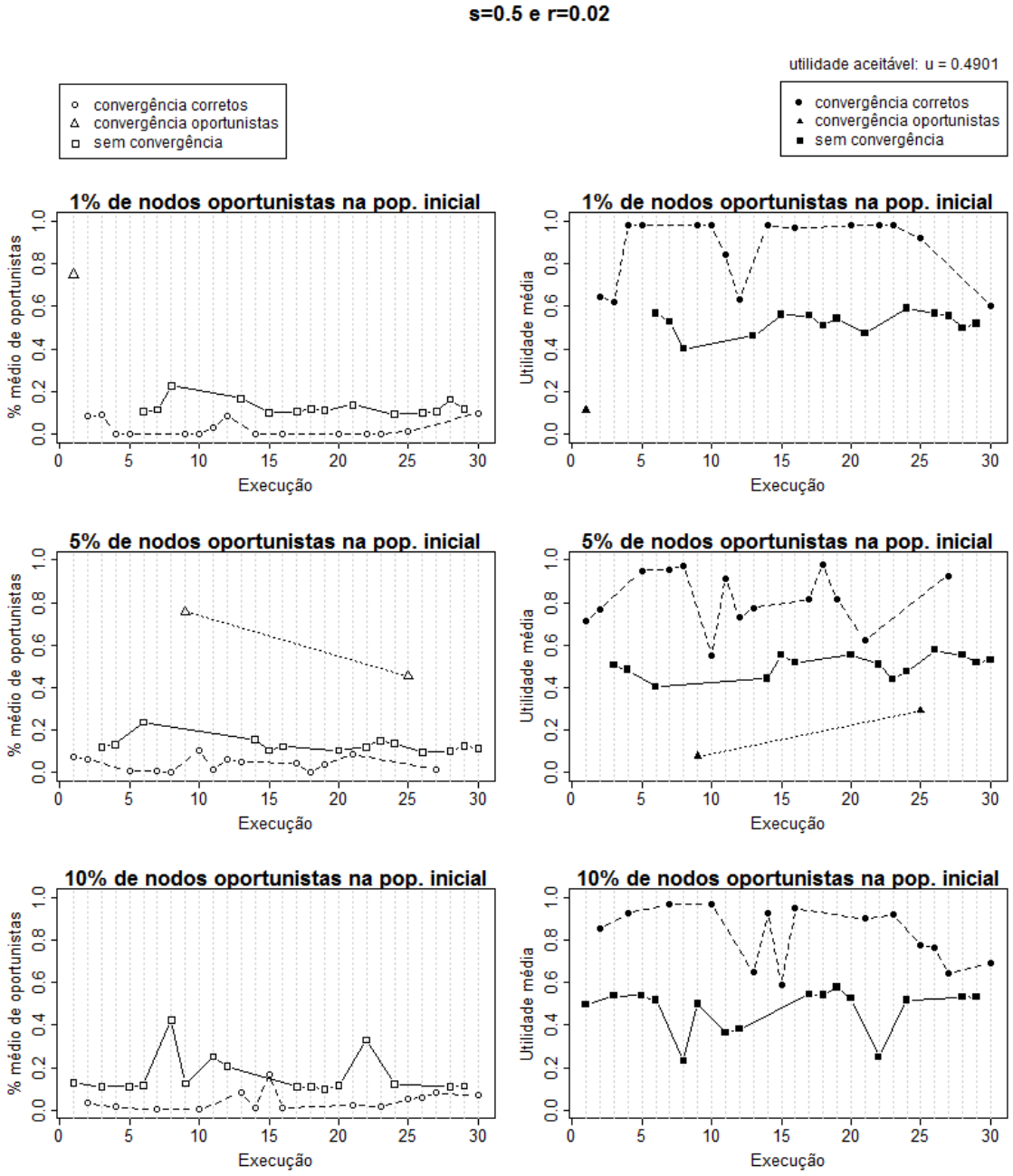

Figura 7.12: Percentual médio de nodos oportunistas (gráficos da esquerda) e utilidade média (gráficos da direita) de cada execução do simulador de TJE com os parâmetros s = 0,5 e r = 0,02

ficos da esquerda) e da utilidade média (gráficos da direita) das execuções do simulador de TJE com os parâmetros s = 0,5 e r = 0,05. Para s = 0,5 e r = 0,05, a utilidade mínima aceitável é igual a u = 0,4756 (vide equação 6.5 para o cômputo da utilidade), que é a utilidade recebida quando o *download* é igual a φ e o *upload* é máximo.

Nos gráficos da figura 7.13, com 1% de nodos oportunistas na população inicial, uma execução converge para o estado estacionário de nodos corretos, onze convergem para o estado estacionário somente com nodos oportunistas, e 18 execuções não convergem para um estado estacionário. As execuções que alcançam o estado estacionário em que todos os nodos são oportunistas apresentam utilidades médias abaixo do aceitável e percentuais médios de nodos oportunistas variando entre 70% e 88%. Nas execuções que não alcançam um equilíbrio (18 execuções), o percentual médio de nodos oportunistas varia entre 30% e 60%, e a utilidade média fica em torno de 0,10, abaixo da utilidade aceitável, o que indica que as execuções não são satisfatórias para os usuários participantes. Com 5% e 10% de nodos oportunistas na população inicial, nenhuma das execuções alcança o estado estacionário em que os nodos são corretos. Percebe-se também que o percentual médio de nodos oportunistas nas execuções (que convergiram para nodos oportunistas e que não convergiram) está bastante disperso, não se concentrando, como em casos anteriores, ao redor de um valor. Em relação à utilidade média, em todas as execuções ela permanece abaixo da utilidade aceitável. Desta forma, percebe-se que esta configuração favorece a proliferação de nodos oportunistas.

A figura 7.14 apresenta os resultados do percentual médio de nodos oportunistas (gráficos da esquerda) e da utilidade média (gráficos da direita) das execuções do simulador de TJE com os parâmetros s = 0,5 e r = 0,3. Para s = 0,5 e r = 0,3, a utilidade mínima aceitável é igual a u = 0,3704 (vide equação 6.5 para o cômputo da utilidade), que é a utilidade recebida quando o *download* é igual a φ e o *upload* é máximo.

Pelos resultados mostrados na figura 7.14, é possível observar que o resultado desta configuração é muito semelhante ao resultado da configuração r = 0,3 com os valores s = 0,05 e s = 0,1 (figuras 7.8 e 7.11, respectivamente). Com os três diferentes percentuais de nodos oportunistas, nenhuma das execuções convergiu para o estado estacionário de nodos corretos, algumas não convergiram, e as demais execuções convergiram para o estado em que os nodos são todos oportunistas. As execuções que não convergiram para um estado estacionário apresentam utilidade média abaixo do aceitável e alto percentual médio de nodos oportunistas na execução. Percebe-se também que o percentual médio de nodos oportunistas nas execuções (que convergiram para nodos oportunistas e que não convergiram) está bastante disperso, não se concentrando, como observado em outras configurações, ao redor de um determinado valor. Com esta combinação de s e r concluise que ela favorece a proliferação de nodos oportunistas na aplicação.

A tabela 7.4 resume os resultados evolucionários obtidos nesta subseção. Percebese, pelos resultados mostrados na tabela, que uma inflexão mais suave da curva logística (parâmetro s) associado a valores pequenos de redução da satisfação (parâmetro r) levam a um ambiente propício para a proliferação de nodos corretos na aplicação. Ou seja, se a percepção dos usuários em relação ao crescimento da satisfação com a taxa de *download* for suave ($s \leq$ 0,1), e a percepção de redução da satisfação devido ao custo de *upload* for negligenciável ($r \leq$ 0,02), então existe um ambiente propício para que a cooperação entre os participantes cresça. Se a percepção dos usuários em relação ao crescimento da satisfação com a taxa de *download* for abrupta ($s >$ 0,1), e a percepção de redução da satisfação devido ao custo de *upload* for negligenciável ($r \leq$ 0,02), então existe um ambiente que favorece "fracamente" a cooperação. De forma inversa ao ambiente que fa-

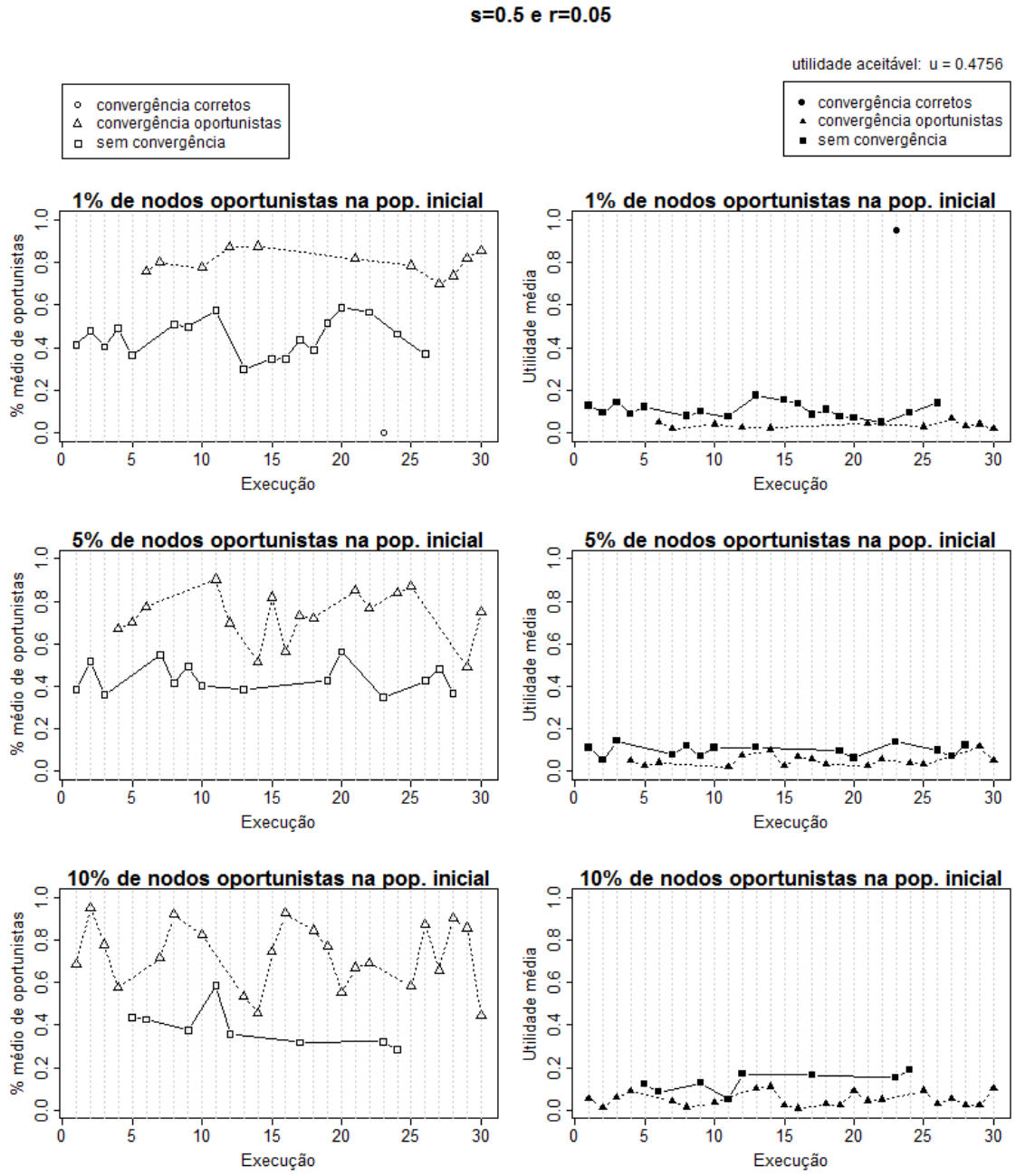

Figura 7.13: Percentual médio de nodos oportunistas (gráficos da esquerda) e utilidade média (gráficos da direita) de cada execução do simulador de TJE com os parâmetros s = 0,5 e r = 0,05

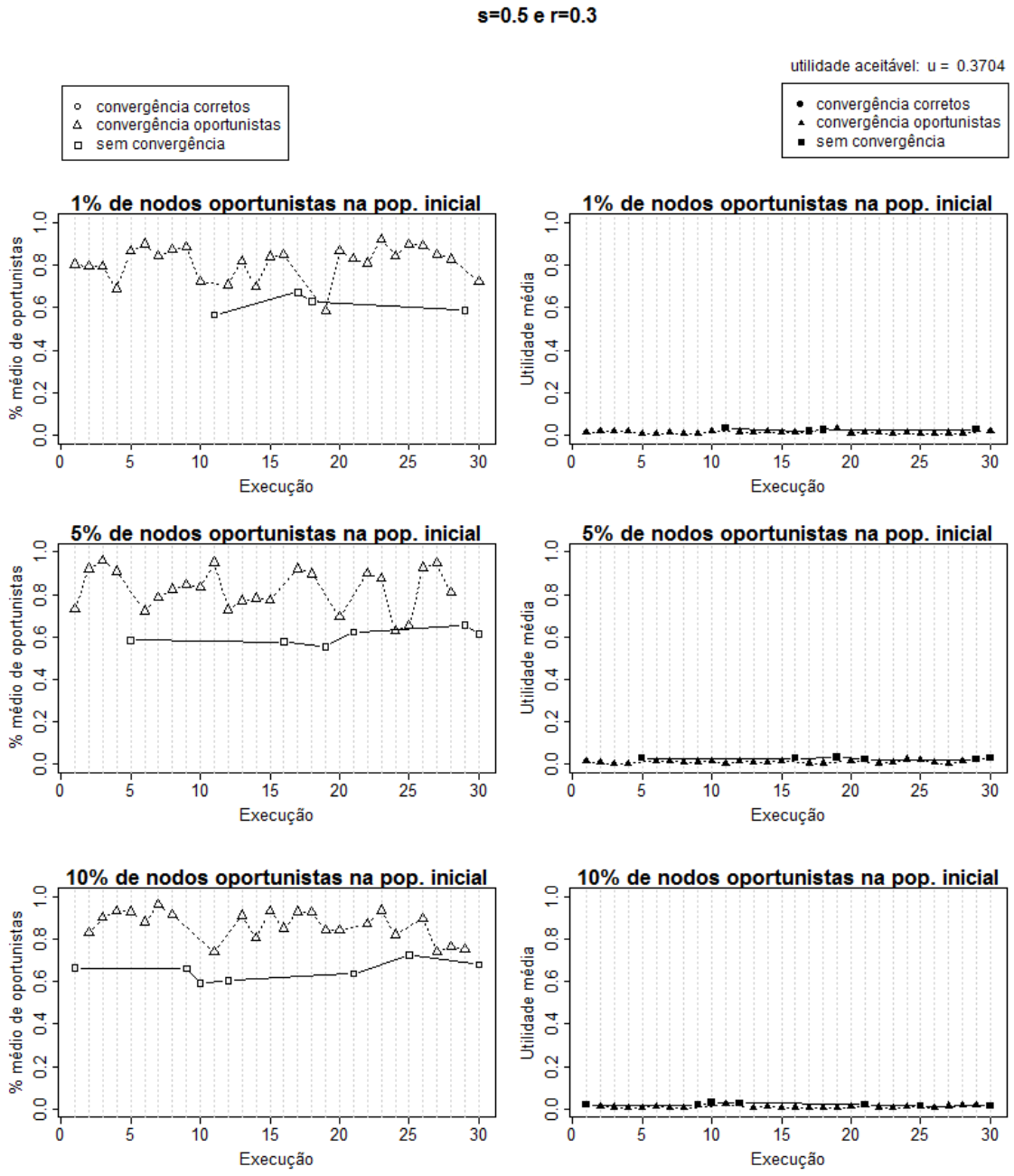

Figura 7.14: Percentual médio de nodos oportunistas (gráficos da esquerda) e utilidade média (gráficos da direita) de cada execução do simulador de TJE com os parâmetros s = 0,5 e r = 0,3

vorece a cooperação, se a percepção dos usuários em relação ao crescimento da satisfação com a taxa de *download* for mais abrupta ($s > 0{,}1$), e especialmente se a percepção de redução da satisfação devido ao custo de *upload* for significativa ($r > 0{,}02$), então existe um ambiente propício para proliferação de nodos oportunistas.

Tabela 7.4: Resumo do efeito de s e r na evolução de 30 execuções de simulação, com percentual variado de nodos oportunistas na população inicial

s	r									% de oportunistas na pop. inicial
	0.02			0.05			0.3			
	convergência corretos	sem convergência	convergência oportunistas	convergência corretos	sem convergência	convergência oportunistas	convergência corretos	sem convergência	convergência oportunistas	
0.05	30	0	0	7	22	1	0	2	28	1%
	30	0	0	0	28	2	0	2	28	5%
	30	0	0	0	28	2	0	1	29	10%
0.1	29	0	1	7	17	6	0	3	27	1%
	27	3	0	0	21	9	0	2	28	5%
	30	0	0	0	25	5	0	1	29	10%
0.5	15	14	1	1	18	11	0	4	26	1%
	14	14	2	0	16	14	0	6	24	5%
	14	16	0	0	8	22	0	7	23	10%

7.3.3 Efeito do sistema de auditoria na dinâmica da população

Nesta seção será analisado o efeito de um sistema de auditoria no processo dinâmico de ajuste das estratégias. Para esta análise, os três estágios do simulador (interação, replicação e sistema de auditoria) estão ativos, e será utilizado um sistema de auditoria simplificado, baseado no trabalho de Haridasan, Jansch-Pôrto e van Renesse (2008). O sistema de auditoria estabelece (mas não divulga) um limiar de *upload* λ com a qual os nodos devem contribuir para permanecer na aplicação. Existem três modos de operação no sistema de auditoria proposto por Haridasan, Jansch-Pôrto e van Renesse (2008): **Fixo**, **Gradual** e **Baseado em Percentual**. O modo Fixo utiliza um limiar não variável (por exemplo, λ = 0,6). O modo Gradual utiliza um valor base, por exemplo, λ = 0,6, e se o *download* médio da amostra (a amostra é calculada a partir de valores médios de *download* medidos e computados periodicamente para avaliar os níveis de interação do sistema) estiver abaixo de um valor satisfatório, então λ é aumentado em uma fração, por exemplo, 0,1. Quando o *download* médio é satisfatório de novo, o valor de λ retorna ao valor base. O modo adaptativo Baseado em Percentual utiliza λ = 0,0, e se o *download* médio estiver abaixo de uma valor satisfatório, λ é escolhido baseado no nível de cooperação da amostra (os níveis de cooperação coletados são ordenados e o valor dividindo os 10% menores é usado como o novo λ). É responsabilidade do Sistema de Auditoria minimizar a incidência de falsos positivos e falsos negativos, mas neste estudo o sistema de auditoria é considerado perfeito quanto a incidência de falsos positivos e negativos. Mais detalhes sobre o sistema de auditoria foram objeto da seção 4.4.1.

Como é assumido que os nodos permanecem por um tempo infinito conectados à aplicação, para simular a desconexão, que é a punição efetivada pelo sistema de auditoria, um nodo, ao ser detectado como incorreto, sofrerá uma punição na qual o seu *download* será igual a zero por τ rodadas. Neste trabalho será utilizado apenas o sistema de auditoria com modo de operação Fixo, e com λ = 0,6.

O sistema de auditoria proposto por Haridasan, Jansch-Pôrto e van Renesse (2008) utiliza amostras de tamanho fixo, que independem do tamanho da rede. O tamanho da amostra é fixo em 300 nodos, e este valor de amostra será utilizado na versão simplificada de auditoria utilizada no simulador de TJE.

A figura 7.15 apresenta os resultados de 30 execuções do simulador de TJE com s = 0,5, r = 0,05, τ = 3 e 50% de nodos oportunistas na população inicial.

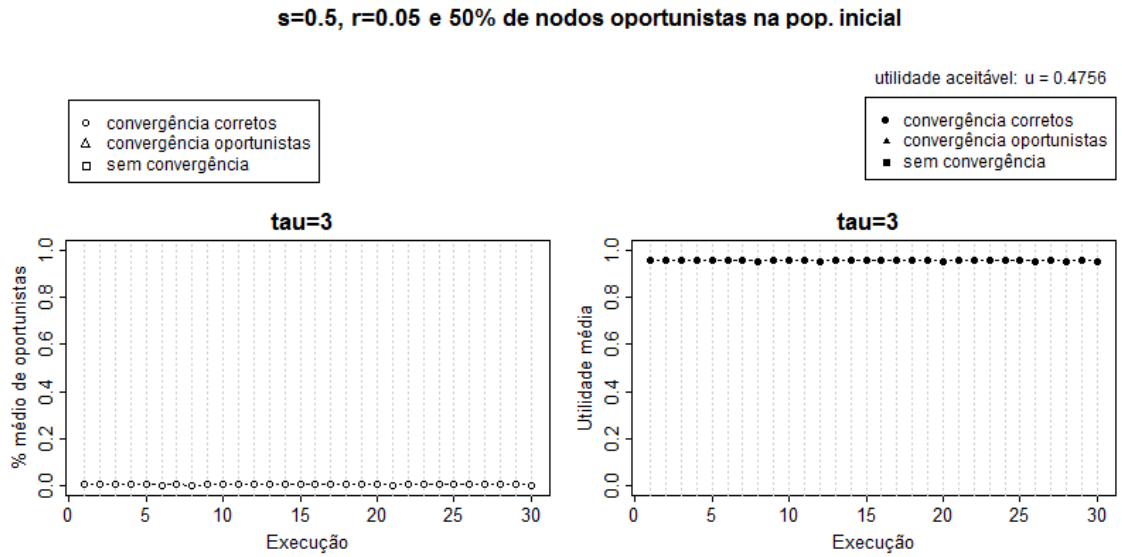

Figura 7.15: Atuação do sistema de auditoria com τ = 3; percentual médio de nodos oportunistas (gráfico da esquerda) e utilidade média (gráfico da direita) de cada execução do simulador de TJE

Comparando os resultados apresentados na figura 7.15 com os resultados apresentados na figura 7.13 da subseção anterior, percebe-se que com a mesma configuração de s e r, e inclusive com um percentual maior de nodos oportunistas na população inicial, existe uma convergência rápida para o estado evolucionário somente com nodos corretos quando o sistema de auditoria está ativo. Com o sistema de auditoria inativo, mesmo um percentual pequeno de nodos oportunistas na população inicial é capaz de levar a aplicação a um estado de estacionário em que todos os nodos são oportunistas, ou então a um estado em que a quantidade de nodos oportunistas é tão alta que a utilidade média da execução fica abaixo do aceitável. Com o sistema de auditoria ativo e configurado para punir por apenas 3 rodadas, é possível direcionar a aplicação a um estado de convergência em que todos os nodos são corretos, como mostrado da figura 7.15. O baixo percentual de nodos oportunistas nas 30 execuções indica que a convergência para o estado estacionário somente com nodos corretos deu-se de forma rápida com o sistema de auditoria ativo. Como consequência, a utilidade média em todas as execuções fica acima da utilidade aceitável. Com o sistema de auditoria ativo é possível alterar um cenário que antes favorecia a proliferação de nodos oportunistas, para um cenário em que a cooperação entre os nodos (proliferação de nodos corretos) é favorecida, mesmo com um percentual alto de nodos oportunistas na população inicial.

A figura 7.16 apresenta os resultados de 30 execuções do simulador de TJE com s = 0,5, r = 0,3, τ = 5; 10; 20 e 10% de nodos oportunistas na população inicial.

Comparando os resultados apresentados na figura 7.16 com os resultados apresentados na figura 7.14 da subseção anterior, percebe-se que, com a mesma configuração de s e r,

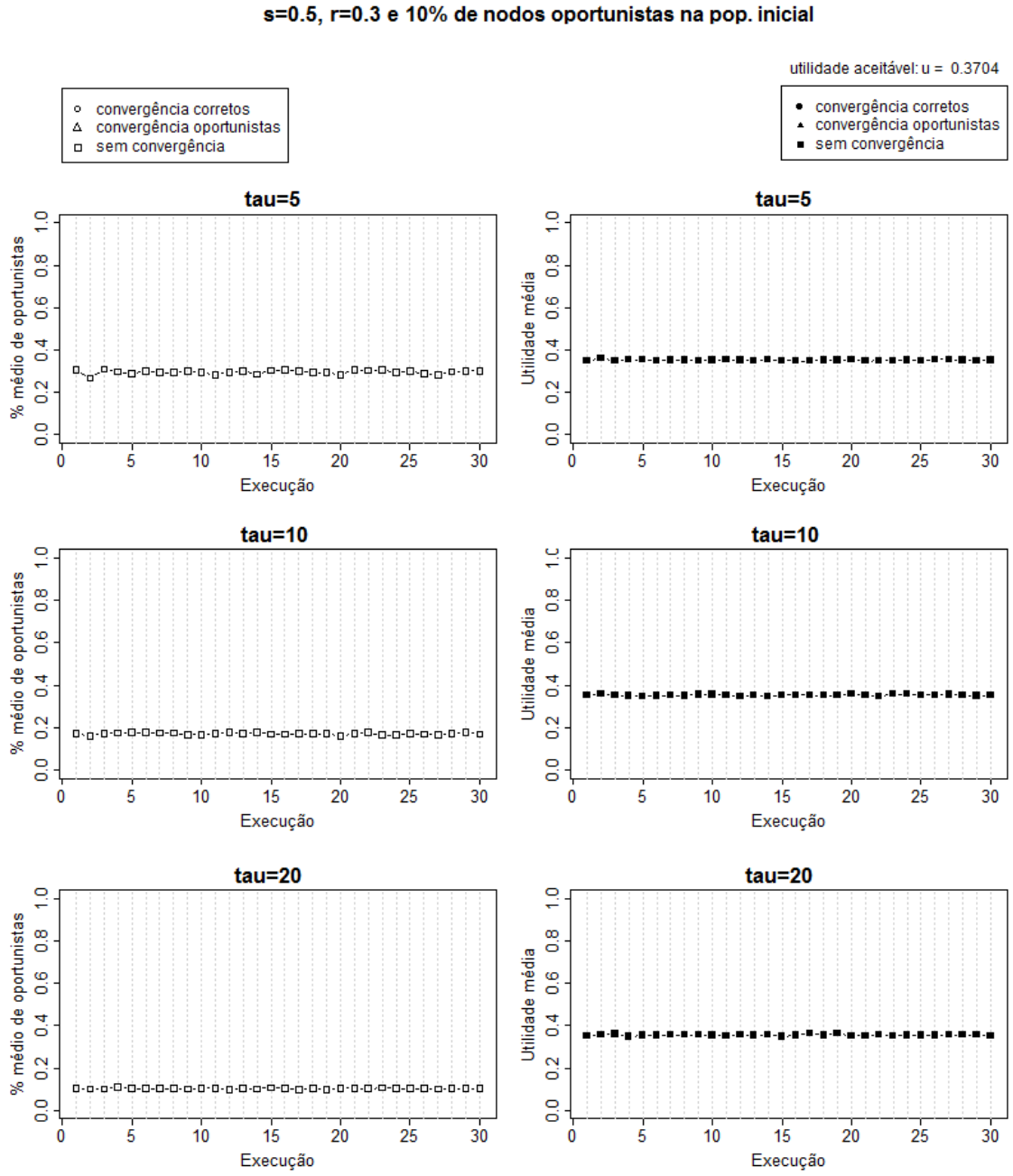

Figura 7.16: Atuação do sistema com τ = 5; 10 e 20; percentual médio de nodos oportunistas (gráficos da esquerda) e utilidade média (gráficos da direita) de cada execução do simulador de TJE

a convergência para o estado de equilíbrio somente com nodos oportunistas é desfavorecida. Com o sistema de auditoria inativo, praticamente todas as execuções do simulador de TJE, com a configuração s = 0,5 e r = 0,3, independentemente do percentual de nodos oportunistas na população inicial, levam a um estado estacionário somente com nodos oportunistas. Com o sistema de auditoria ativo e configurado para punir por 5 rodadas, a aplicação é direcionada a um estado sem convergência, mas que parece favorecer a cooperação, visto que a utilidade média é consideravelmente aumentada, quando comparada à utilidade média observada na figura 7.14, em que o sistema de auditoria está inativo. Apesar de conter a proliferação de nodos oportunistas, o sistema de auditoria configurado para efetuar punição por τ = 5 rodadas não é suficiente para manter a utilidade média das execuções acima da utilidade aceitável. Mesmo aumentando o número de rodadas de punição para τ = 10 ou τ = 20, a utilidade média não é significativamente aumentada, continuando abaixo da utilidade satisfatória. Isto indica que, para esta configuração de s e r, o sistema de auditoria não tem resultados satisfatórios, apesar de favorecer a proliferação de nodos corretos.

8 CONCLUSÃO

Um dos grandes pontos fortes do serviço de *multicast* via *Internet* em relação às transmissões tradicionais de rádio e TV é poder romper as barreiras geográficas impostas pela tecnologia de transmissão tradicional. Através da *Internet*, uma transmissão ao vivo pode alcançar milhares de usuários, mesmo que geograficamente distantes da origem da transmissão. Dentro deste contexto, as aplicações de *live streaming* em redes P2P têm obtido muita atenção nos últimos anos, já que um serviço de transmissão de conteúdo ao vivo sob essa tecnologia possui um custo bastante reduzido de infra-estrutura - todo novo nodo que se junta à aplicação deve contribuir disponibilizando uma parte de sua largura de banda para fazer *upload* de dados para outros nodos. Outras vantagens de se utilizar serviços de *live streaming* em redes P2P são alto potencial de robustez, escalabilidade e adaptabilidade devido à redundância e não dependência de recursos particulares dentre os nodos participantes. Apesar das vantagens para sua utilização, e devido aos seus requisitos funcionais, as aplicações de *live streaming* são muito sensíveis a nodos oportunistas e bizantinos.

Apesar de existir uma quantidade considerável de modelos sobre a aplicação de Teoria de Jogos ao contexto de aplicações de compartilhamento de arquivos, estes modelos não são adequados para serem diretamente aplicados ao contexto de aplicações de *live streaming* porque as aplicações de *live streaming* possuem particularidades que as distinguem grandemente das aplicações de compartilhamento de arquivos. As principais particularidades estão relacionadas aos aspectos temporais das transmissões “ao vivo” e ao período de duração (horas, no máximo). Neste trabalho, uma nova abordagem para modelar aplicações de *live streaming* em redes P2P foi proposta e avaliada. O objetivo do modelo proposto é predizer a evolução da aplicação sob diferentes situações de mau comportamento dos nodos. Este trabalho baseou-se na Teoria dos Jogos Evolucionários para criar um modelo dinâmico e evolutivo, ou seja, um modelo que capture da maneira realista a evolução da cooperação (medida em termos da utilidade percebida pelo usuário) de uma aplicação *live streaming* quando nodos não corretos aparecem na aplicação.

A principal contribuição deste trabalho consiste, então, na apresentação de um modelo matemático baseado em Teoria dos Jogos Evolucionários, cujo objetivo é ajudar a compreender as aplicações de *live streaming* em redes P2P e os fatores que influenciam o seu funcionamento adequado. A avaliação do modelo foi realizada através de simulações. Como contribuição secundária, este trabalho apresenta os resultados de uma análise estatística do comportamento do *download* e *upload* observado do simulador do Chainsaw.

A análise estatística do simulador do Chainsaw indicou a existência de uma lei de potência no decaimento da variância temporal de *download* e *upload*. Esta característica é interessante porque a lei de potência é uma relação matemática especial entre duas quantidades, e é objeto de estudo por ser encontrada em vários fenômenos naturais e

artificiais. Em redes com 1.000 nodos, as leis de potência para o *download* e *upload* são:

$$var(down)(t) \sim t^{-0,161(2)} \;\; e \;\; var(up)(t) \sim t^{-0,164(2)}$$

Observou-se também que o tamanho da rede P2P tem um efeito significativo no expoente da lei de potência das variâncias de *download* e *upload*. Foram analisadas redes com 4 (quatro) tamanhos diferentes: 125, 250, 500 e 1.000 nodos. Para o *upload*, o tamanho da rede e o expoente da lei de potência são diretamente proporcionais (o expoente da lei de potência da variância de *upload* é máximo (θ_{up} = -0,164(2)) na rede com 1.000 nodos, e é mínimo (θ_{up} = -0,230(2)) na rede com 125 nodos. Já para o *download*, o tamanho da rede e o expoente da lei de potência tem uma uma relação inversa - o expoente da lei de potência da variância do *download* é mínimo (θ_{down} = -0,161(2)) na rede com 1.000 nodos, e é máximo (θ_{down} = -0,072(2)) na rede com 250 nodos, uma rede de tamanho intermediário.

O objetivo da análise do modelo através das simulações evolutivas foi mostrar a evolução da aplicação (a mudança da frequência de nodos corretos e oportunistas) com o tempo, partindo de diferentes configurações de percentual de nodos oportunistas na população inicial da aplicação. Uma restrição das simulações é que o tamanho da população é mantido constante (rede estática). Através das simulações foi possível observar que uma inflexão mais suave da curva logística do *download* (parâmetro s), associado a valores negligenciáveis de redução da satisfação com o *upload* (parâmetro r) levam a um ambiente propício para a proliferação de nodos corretos na aplicação. Ou seja, se a percepção dos usuários em relação ao crescimento da satisfação com a taxa de *download* for suave ($s \leq$ 0,1), e a percepção de redução da satisfação devido ao custo de *upload* for negligenciável ($r \leq$ 0,02), então existe um ambiente propício para que a cooperação entre os participantes cresça. Porém, se a percepção dos usuários em relação ao crescimento da satisfação com a taxa de *download* for abrupta ($s >$ 0,1), e a percepção de redução da satisfação devido ao custo de *upload* for negligenciável ($r \leq$ 0,02), então existe um ambiente que favorece a cooperação de forma fraca. De maneira inversa ao ambiente que favorece a cooperação, se a percepção dos usuários em relação ao crescimento da satisfação com a taxa de *download* for abrupta ($s >$ 0,1), e a percepção de redução da satisfação devido ao custo de *upload* for significativa ($r >$ 0,02), então existe um ambiente propício para a proliferação de nodos oportunistas. As simulações evolutivas também indicam que um sistema de auditoria pode (com certas limitações) ser efetivo em manter a cooperação entre os participantes da aplicação de *live streaming*. Um sistema de auditoria ajuda a favorecer a proliferação de nodos corretos em situações que naturalmente os nodos oportunistas dominariam a população.

Considerando que o presente trabalho explora uma abordagem nova, resultante da combinação de mecanismos de Teoria dos Jogos Evolucionários que não haviam sido ainda aplicados ao contexto de aplicações de *live streaming* em redes P2P, e que as aplicações de *live streaming* ainda não foram estudadas suficientemente quanto ao comportamento dos usuários que as empregam, este trabalho traz resultados preliminares e deixa vários pontos em aberto que poderão ser desenvolvidos através de pesquisas complementares. Aborda-se, a seguir, algumas dessas possibilidades.

- Estudo de outras funções de "resfriamento" ($c(it)$), objetivando obter resultados mais próximos, em termos de expoente da lei de potência das variâncias de *download* e *upload*, aos resultados obtidos do simulador do Chainsaw.

- Inserção do comportamento de nodos bizantinos ao modelo, de forma a possibilitar o estudo combinado ou em separado do impacto de nodos oportunistas e bizantinos na aplicação.

- As aplicações de *live streaming* em redes P2P são naturalmente dinâmicas em seu tamanho, característica que não foi analisada neste trabalho, uma vez que ambos os simuladores tomam por base redes estáticas quanto à quantidade de nodos para a execução das simulações. Desta forma, a incorporação de um sistema de manutenção da rede P2P aos simuladores é desejável, possibilitando assim a adoção de premissas envolvendo redes dinâmicas quanto ao número de nodos participantes nas análises a serem conduzidas.

- Realizar um estudo sobre a percepção custo (*upload*) x beneficio (qualidade de *playback*) dos usuários de aplicações de *live streaming*, tendo como objetivo refinar, com base em dados reais, a função de utilidade.

REFERÊNCIAS

AIYER, A. S.; ALVISI, L.; CLEMENT, A.; DAHLIN, M.; MARTIN, J.-P.; PORTH, C. BAR Fault Tolerance for Cooperative Services. **SIGOPS Operating Systems Review**, New York, NY, USA, v.39, n.5, p.45–58, 2005.

ALEXANDER, M. J. **Evolutionary Game Theory**. The Stanford Encyclopedia of Philosophy (Winter 2008 Edition), Edward N. Zalta (ed.). Disponível em: <http://plato.stanford.edu/archives/win2008/entries/game-evolutionary/>. Acesso em: fevereiro 2009.

ANDROUTSELLIS-THEOTOKIS, S.; SPINELLIS, D. A survey of peer-to-peer content distribution technologies. **ACM Computer Surveys**, New York, NY, USA, v.36, n.4, p.335–371, 2004.

BITTORRENT. **Página do projeto BitTorrent**. Disponível em: <http://www.bittorrent.com/>. Acesso em: fevereiro 2009.

BLANC, A.; LIU, Y.-K.; VAHDAT., A. Designing Incentives for Peer-to-Peer Routing. In: INFOCOM: 24TH ANNUAL JOINT CONFERENCE OF THE IEEE COMPUTER AND COMMUNICATIONS SOCIETIES, 2005. **Proceedings...** [S.l.: s.n.], 2005. v.1, p.374–385.

BOLTON, G. E.; KATOK, E.; ZWICK, R. Dictator game giving: rules of fairness versus acts of kindness. **International Journal of Game Theory**, [S.l.], v.27, n.2, p.269–2999, 1998.

BRANDT, F.; WEIß, G. Antisocial Agents and Vickrey Auctions. **Lecture Notes in Computer Science**, [S.l.], v.2333, p.335–347, 2002.

BROGLE, M.; MILIC, D.; BRAUN, T. Quality of Service for Peer-to-Peer Based Networked Virtual Environments. In: ICPADS '08: 14TH IEEE INTERNATIONAL CONFERENCE ON PARALLEL AND DISTRIBUTED SYSTEMS, 2008. **Proceedings...** [S.l.: s.n.], 2008. p.847–852.

COOLSTREAMING. **Página da aplicação Coolstreaming**. Disponível em: <http://www.coolstreaming.us>. Acesso em: fevereiro 2009.

CRISTIAN, F. Understanding fault-tolerant distributed systems. **Communications of the ACM**, New York, NY, USA, v.34, n.2, p.56–78, 1991.

DO, T.; HUA, K.; TANTAOUI, M. P2VoD: providing fault tolerant video-on-demand streaming in peer-to-peer environment. In: IEEE INTERNATIONAL CONFERENCE ON COMMUNICATIONS, 2004. **Proceedings...** Springer Berlin / Heidelberg, 2004. v.3, p.1467–1472.

DOUCEUR, J. R. The Sybil Attack. In: IPTPS: FIRST INTERNATIONAL WORKSHOP ON PEER-TO-PEER SYSTEMS, 2002. **Proceedings...** Springer Berlin / Heidelberg, 2002. v.2429, p.251–260.

EMULE. **Página do projeto eMule**. Disponível em: <http://www.emule-project.net/>. Acesso em: fevereiro 2009.

FELDMAN, M.; LAI, K.; STOICA, I.; CHUANG, J. Robust Incentive Techniques for Peer-to-Peer Networks. In: EC '04: THE 5TH ACM CONFERENCE ON ELECTRONIC COMMERCE, 2004, New York, NY, USA. **Proceedings...** ACM Press, 2004. p.102–111.

FELDMAN, M.; PAPADIMITRIOU, C.; CHUANG, J.; STOICA, I. Free-riding and Whitewashing in Peer-to-Peer Systems. **IEEE Journal on Selected Areas in Communications**, [S.l.], v.24, n.5, p.1010–1019, 2006.

GLERIA, I.; MATSUSHITA, R.; SILVA, S. D. Sistemas Complexos, Criticalidade e Leis de Potência. **Revista Brasileira de Ensino de Física**, São Paulo,SP,BR, v.26, n.2, p.99–108, 2004.

GNUTELLA. **Página do projeto Gnutella**. Disponível em: <http://rfc-gnutella.sourceforge.net/>. Acesso em: fevereiro 2009.

HARIDASAN, M.; JANSCH-PORTO, I.; RENESSE, R. van. Enforcing fairness in a live-streaming system. In: MULTIMEDIA COMPUTING AND NETWORKING, 2008. **Proceedings...** SPIE, 2008. v.6818.

HARIDASAN, M.; RENESSE, R. van. Defense against Intrusion in a Live Streaming Multicast System. In: P2P '06: THE SIXTH IEEE INTERNATIONAL CONFERENCE ON PEER-TO-PEER COMPUTING, 2006, Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2006. p.185–192.

HEI, X.; LIANG, C.; LIANG, J.; LIU, Y.; ROSS, K. W. A Measurement Study of a Large-Scale P2P IPTV System. **Multimedia, IEEE Transactions on**, [S.l.], v.9, n.8, p.1672–1687, Dec 2007.

HUANG, Z.-G.; WU, Z.-X.; GUAN, J.-Y.; WU, A.-C.; WANG, Y.-H. **The public goods game on homogeneous and heterogeneous networks**: investment strategy according to the pool size. Disponível em: <http://www.citebase.org/abstract?id=oai:arXiv.org:0708.2805>. Acesso em: outubro 2009.

JIN, X.; CHENG, K.-L.; GARY CHAN, S.-H. SIM: scalable island multicast for peer-to-peer media streaming. In: IEEE INTERNATIONAL CONFERENCE ON MULTIMEDIA AND EXPO, 2006. **Proceedings...** [S.l.: s.n.], 2006. p.913–916.

JOHANSEN, H.; ALLAVENA, A.; RENESSE, R. van. Fireflies: scalable support for intrusion-tolerant network overlays. In: EUROSYS '06: PROCEEDINGS OF THE 1ST ACM SIGOPS/EUROSYS EUROPEAN CONFERENCE ON COMPUTER SYSTEMS 2006, 2006, New York, NY, USA. **Proceedings...** ACM, 2006. p.3–13.

KAZAA. **Página da aplicação Kazaa**. Disponível em: <http://www.kazaa.com/>. Acesso em: fevereiro 2009.

KUHN, S. **Prisoner's Dilemma**. The Stanford Encyclopedia of Philosophy (Spring 2009 Edition), Edward N. Zalta (ed.). Disponível em: <<http://plato.stanford.edu/archives/spr2009/entries/prisoner-dilemma/>. Acesso em: fevereiro 2009.

LI, B.; YIN, H. Peer-to-Peer Live Video Streaming on the Internet: issues, existing approaches, and challenges. **IEEE Communications Magazine**, [S.l.], v.45, n.6, p.94–99, 2007.

LI, H. C.; CLEMENT, A.; WONG, E. L.; NAPPER, J.; ROY, I.; ALVISI, L.; DAHLIN, M. Bar Gossip. In: THE 7TH USENIX SYMPOSIUM ON OPERATING SYSTEMS DESIGN AND IMPLEMENTATION, 2006. **Proceedings...** USENIX Association, 2006. p.191–204.

LIU, J.; RAO, S. G.; LI, B.; ZHANG, H. Opportunities and Challenges of Peer-to-Peer Internet Video Broadcast. In: THE IEEE SPECIAL ISSUE ON RECENT ADVANCES IN DISTRIBUTED MULTIMEDIA COMMUNICATIONS, 2007. **Proceedings...** [S.l.: s.n.], 2007. v.96, n.1, p.11–24.

LIU, Y.; GUO, Y.; LIANG, C. A survey on peer-to-peer video streaming systems. **Peer-to-Peer Networking and Applications**, [S.l.], v.1, n.1, p.18–28, 2008.

MARINHO, R. **Prática na Teoria - Aplicações da Teoria de Jogos e da Evolução aos Negócios**. [S.l.]: Editora Saraiva, 2005.

MARTIN, J.-P. **Leveraging Altruism in Cooperative Services**. Disponível em: <http://research.microsoft.com/users/Cambridge/jpmartin/papers/Martin07LeveragingTr.pdf>. Acesso em: dezembro 2007.

MENASCHé, D. S.; FIGUEIREDO, D. R.; SILVA, E. de Souza e. An evolutionary game-theoretic approach to congestion control. **Performance Evaluation**, Amsterdam, The Netherlands, The Netherlands, v.62, n.1-4, p.295 – 312, 2005.

NAPSTER. **Página da aplicação Napster**. Disponível em: <http://free.napster.com/>. Acesso em: fevereiro 2009.

NASH, J. Equilibrium Points in n-Person Games. In: PNAS, 1950. **Proceedings...** [S.l.: s.n.], 1950. v.36, p.48–49.

NASH, J. The Bargaining Problem. In: ECONOMETRICA, 1950. **Proceedings...** [S.l.: s.n.], 1950. v.18, p.155–162.

NASH, J. Non-cooperative Games. In: ANNALS OF MATHEMATICS JOURNAL, 1951. **Proceedings...** [S.l.: s.n.], 1951. v.54, p.286–295.

NEUMANN, J. von; MORGENSTERN, O. **The Theory of Games and Economic Behavior**. [S.l.]: Princeton University Press, 1944.

OH-ISHI, T.; SAKAI, K.; KIKUMA, K.; KUROKAWA, A. Study of the Relationship Between Peer-to-Peer Systems and IP Multicasting. **IEEE Communications Magazine**, [S.l.], v.41, n.1, p.80–84, 2003.

OUALHA, N.; ROUDIER, Y. Validating Peer-to-Peer Storage Audits with Evolutionary Game Theory. **Lecture Notes in Computer Science**, [S.l.], v.5343, p.47–58, 2008.

PADMANABHAN, V. N.; WANG, H. J.; CHOU, P. A.; SRIPANIDKULCHAI, K. Distributing streaming media content using cooperative networking. In: NOSSDAV '02: THE 12TH INTERNATIONAL WORKSHOP ON NETWORK AND OPERATING SYSTEMS SUPPORT FOR DIGITAL AUDIO AND VIDEO, 2002, New York, NY, USA. **Proceedings...** ACM, 2002. p.177–186.

PAI, V.; KUMAR, K.; TAMILMANI, K.; SAMBAMURTHY, V.; MOHR, A. E. Chainsaw: eliminating trees from overlay multicast. In: IPTPS: 4TH INTERNATIONAL WORKSHOP ON PEER-TO-PEER SYSTEMS, 2005, Ithaca, NY, USA. **Proceedings...** Springer Berlin / Heidelberg, 2005. v.3640, p.127–140.

PPLIVE. **Página da aplicação PPLive**. Disponível em: <http://www.pplive.com>. Acesso em: fevereiro 2009.

PPSTREAM. **Página da aplicação PPStream**. Disponível em: <http://www.ppstream.com/>. Acesso em: fevereiro 2009.

PRESTWICH, K. **Game theory**. Disponível em: <http://www.holycross.edu/departments/biology/kprestwi/behavior/ESS/pdf/games.pdf>. Acesso em: outubro 2007.

ROSS, D. **Game Theory**. The Stanford Encyclopedia of Philosophy (Winter 2008 Edition), Edward N. Zalta (ed.). Disponível em: <http://plato.stanford.edu/archives/win2008/entries/game-theory/>. Acesso em: fevereiro 2009.

ROWSTRON, A.; DRUSCHEL, P. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. **SIGOPS Operating Systems Review**, New York, NY, USA, v.35, n.5, p.188–201, 2001.

SHNEIDMAN, J.; PARKES, D. C.; MASSOULI, L. Faithfulness in internet algorithms. In: PINS '04: THE ACM SIGCOMM WORKSHOP ON PRACTICE AND THEORY OF INCENTIVES IN NETWORKED SYSTEMS, 2004, New York, NY, USA. **Proceedings...** ACM, 2004. p.220–227.

SILVERSTON, T.; FOURMAUX, O. **P2P IPTV Measurement**: a comparison study. Disponível em: <http://www.citebase.org/abstract?id=oai:arXiv.org:cs/0610133>. Acesso em: outubro 2009.

SKYRMS, B. The Stag Hunt. In: THE AMERICAN PHILOSOPHICAL ASSOCIATION, 2001. **Proceedings...** American Philosophical Association, 2001. p.31–41.

SOPCAST. **Página da aplicação SopCast**. Disponível em: <http://www.sopcast.com>. Acesso em: dezembro 2009.

SZABó, G.; HAUERT, C. Evolutionary prisoner's dilemma games with voluntary participation. **Physical Review E**, [S.l.], v.66, n.6, p.062903, Dec 2002.

SZABó, G.; VUKOV, J. Cooperation for volunteering and partially random partnerships. **Physical Review E**, [S.l.], v.69, n.3, p.036107, Mar 2004.

TUROCY, T. L.; STENGEL, B. von. **Game Theory***: draft prepared for the enciclopedia of information systems. Disponível em: <http://www.cdam.lse.ac.uk/Reports/Files/cdam-2001-09.pdf>. Acesso em: outubro 2007.

TVUPLAYER. **Página da aplicação TVUplayer**. Disponível em: <http://tvunetworks.com/>. Acesso em: dezembro 2009.

VELOSO, E.; ALMEIDA, V.; MEIRA, W.; BESTAVROS, A.; JIN, S. A Hierarchical Characterization of a Live Streaming Media Workload. In: IMW '02: THE 2ND ACM SIGCOMM WORKSHOP ON INTERNET MEASUREMENT, 2002, New York, NY, USA. **Proceedings...** ACM, 2002. p.117–130.

VU, L.; GUPTA, I.; LIANG, J.; NAHRSTEDT, K. Measurement and modeling a large-scale overlay for multimedia streaming. In: QSHINE 2007: THE 4TH INTERNATIONAL CONFERENCE ON HETEROGENEOUS NETWORKING FOR QUALITY, RELIABILITY, SECURITY AND ROBUSTNESS, 2007. **Proceedings...** ACM, 2007.

WEIBULL, J. W. **Evolutionary Game Theory**. 1.ed. [S.l.]: The MIT Press, 1997.

WEN, G.; LONGSHE, H.; QIANG, F. **Recent Advances in Peer-to-Peer Media Streaming Systems**. Disponível em: <http://www.china-cic.org.cn/english/digital%20library/200610/6.pdf>. Acesso em: dezembro 2007.

XIE, S.; LI, B.; KEUNG, G. Y.; ZHANG, X. Coolstreaming: design, theory, and practice. **Multimedia, IEEE Transactions on**, [S.l.], v.9, n.8, p.1661–1671, Dec. 2007.

YIU, W.-P. K.; JIN, X.; CHAN, S.-H. G. Challenges and Approaches in Large-Scale P2P Media Streaming. **IEEE MultiMedia**, Los Alamitos, CA, USA, v.14, n.2, p.50–59, 2007.

YU, H.; KAMINSKY, M.; GIBBONS, P. B.; FLAXMAN, A. SybilGuard: defending against sybil attacks via social networks. In: SIGCOMM '06: CONFERENCE ON APPLICATIONS, TECHNOLOGIES, ARCHITECTURES, AND PROTOCOLS FOR COMPUTER COMMUNICATIONS, 2006, New York, NY, USA. **Proceedings...** ACM, 2006. p.267–278.

ZHANG, Q.; XUE, H.-F.; KOU, X. dong. An Evolutionary Game Model of Resources-sharing Mechanism in P2P Networks. In: IITA '07: WORKSHOP ON INTELLIGENT INFORMATION TECHNOLOGY APPLICATION, 2007, Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2007. p.282–285.

ZHANG, X.; LIU, J.; LI, B. **On Large Scale Peer-to-Peer Live Video Distribution**: coolstreaming and its prelimianry experimental results. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.61.3377>. Acesso em: outubro 2009.

APÊNDICE A OUTROS GRÁFICOS

Neste apêndice são mostrados os gráficos completos sobre o efeito de g_{max} e g_{min} nas variâncias de *download* e *upload*, os quais foram analisados na seção 7.2.1. Os dados destes gráficos foram mostrados de forma resumida na figura 7.3. As figuras A.1 e A.2 apresentam as curvas da variância do *donwload* e *upload*, respectivamente, para diferentes valores de g_{max}. E as figuras A.3 e A.4 apresentam as curvas da variância do *download* e *upload*, respectivamente, para os diferentes valores de g_{min} estudados.

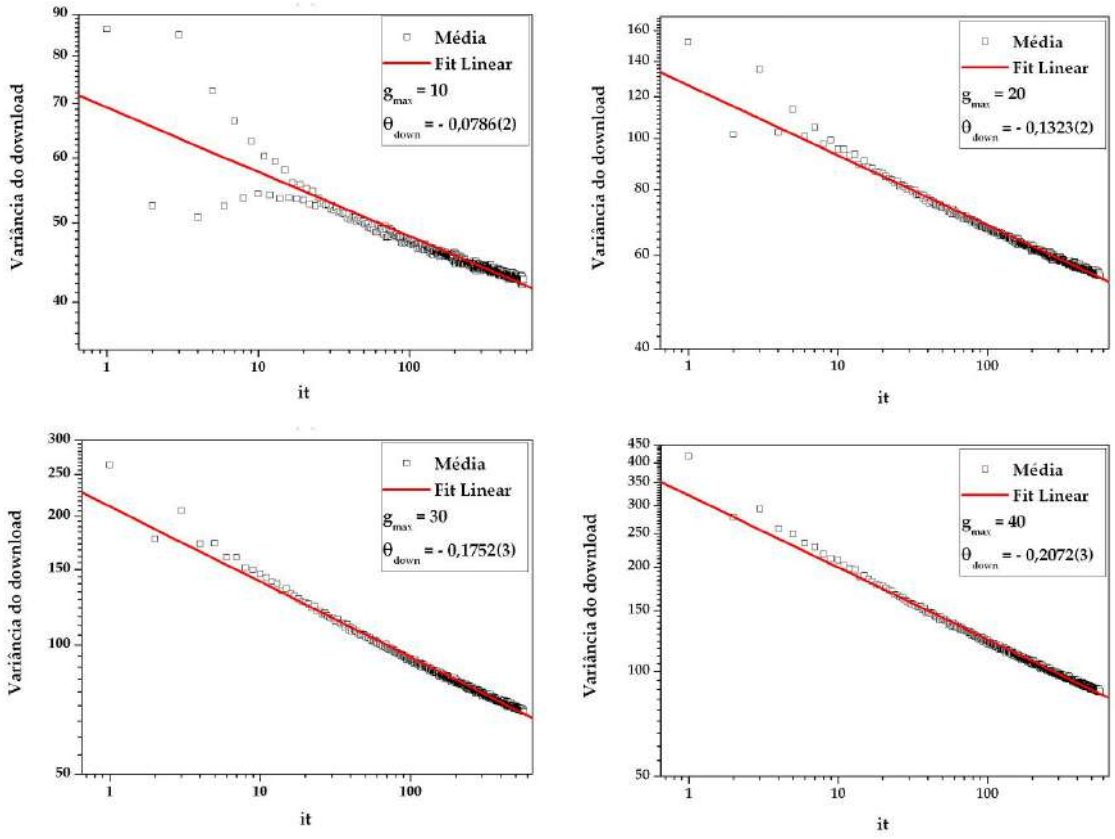

Figura A.1: Gráficos do efeito de g_{max} em θ_{down}, com $\beta = 0,20$ e $g_{min} = 10$ pré-fixados

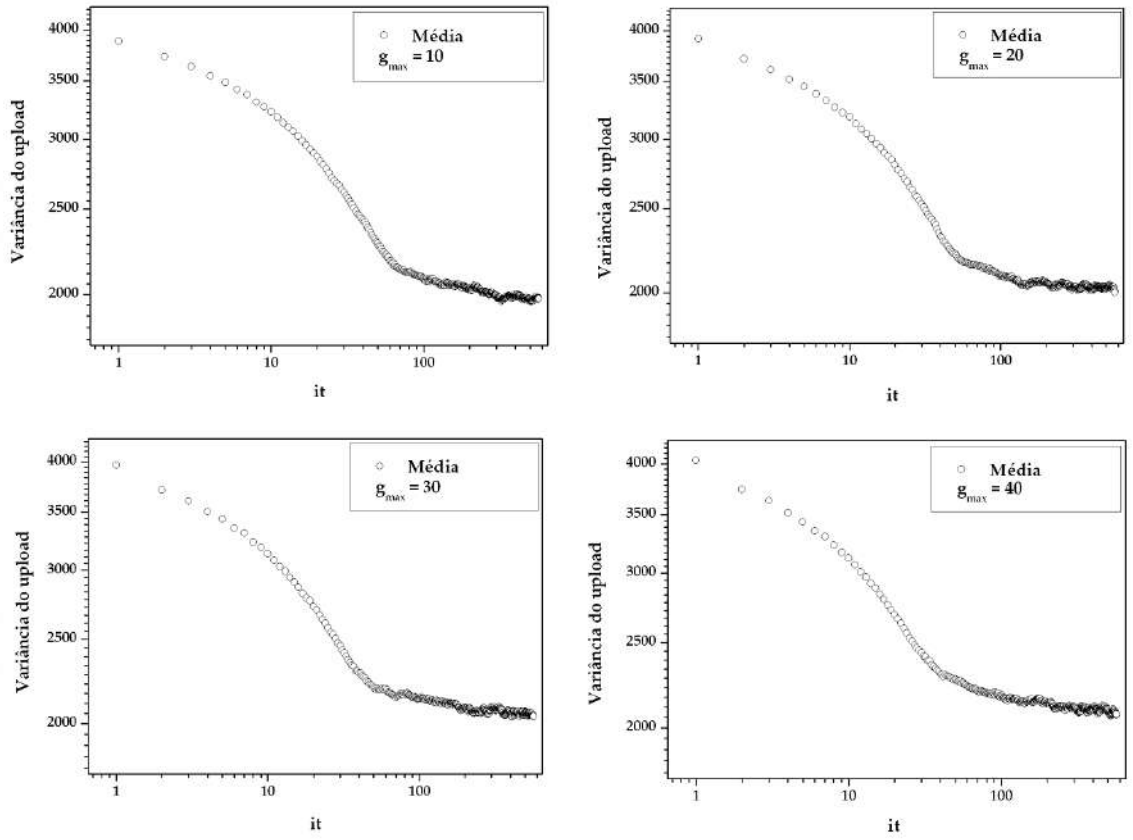

Figura A.2: Gráficos do efeito de g_{max} no decaimento da variância de *upload*, com $\beta = 0,20$ e $g_{min} = 10$ pré-fixados

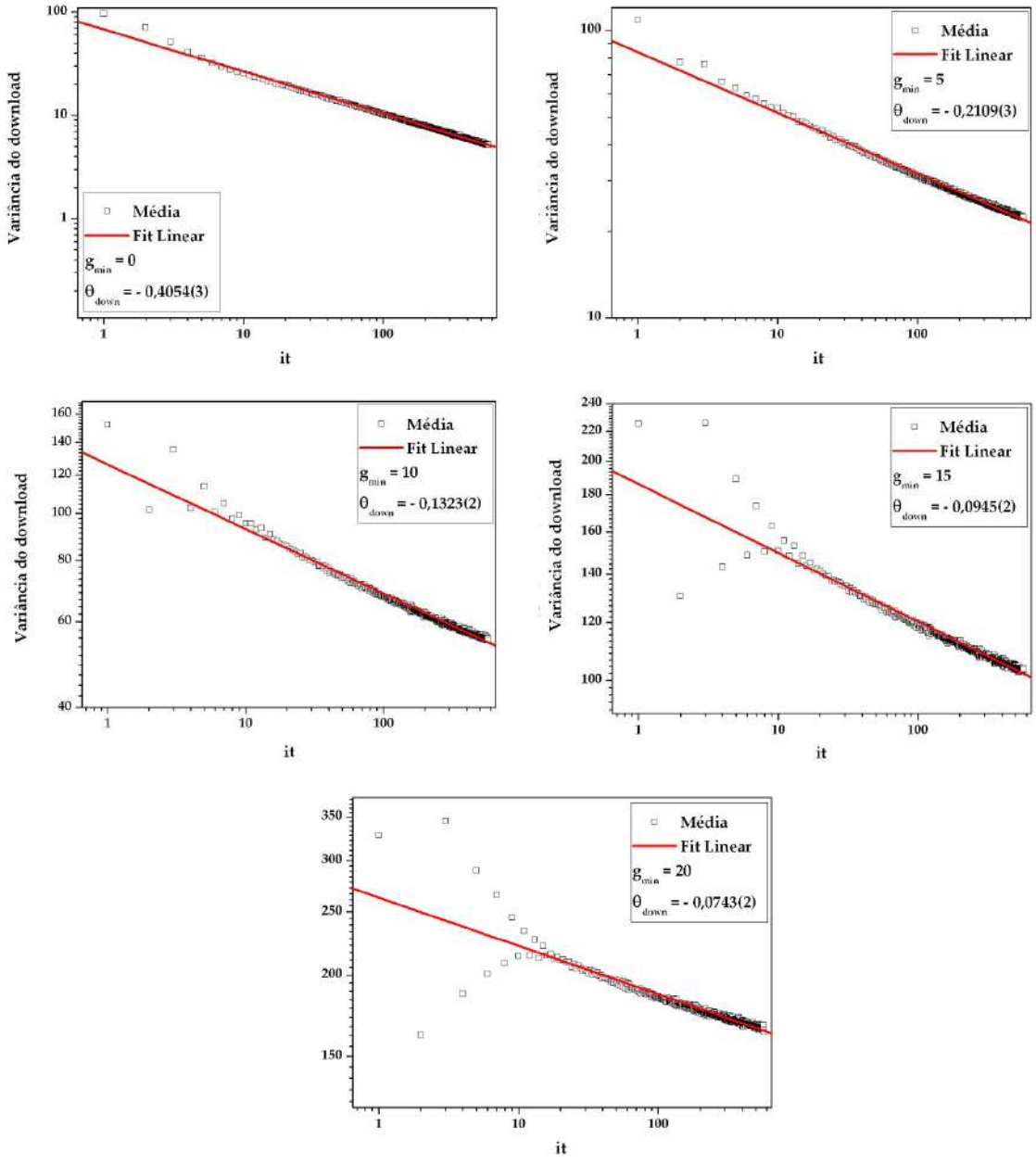

Figura A.3: Gráficos do efeito de g_{min} em θ_{down}, com $\beta = 0, 20$ e $g_{max} = 20$ pré-fixados

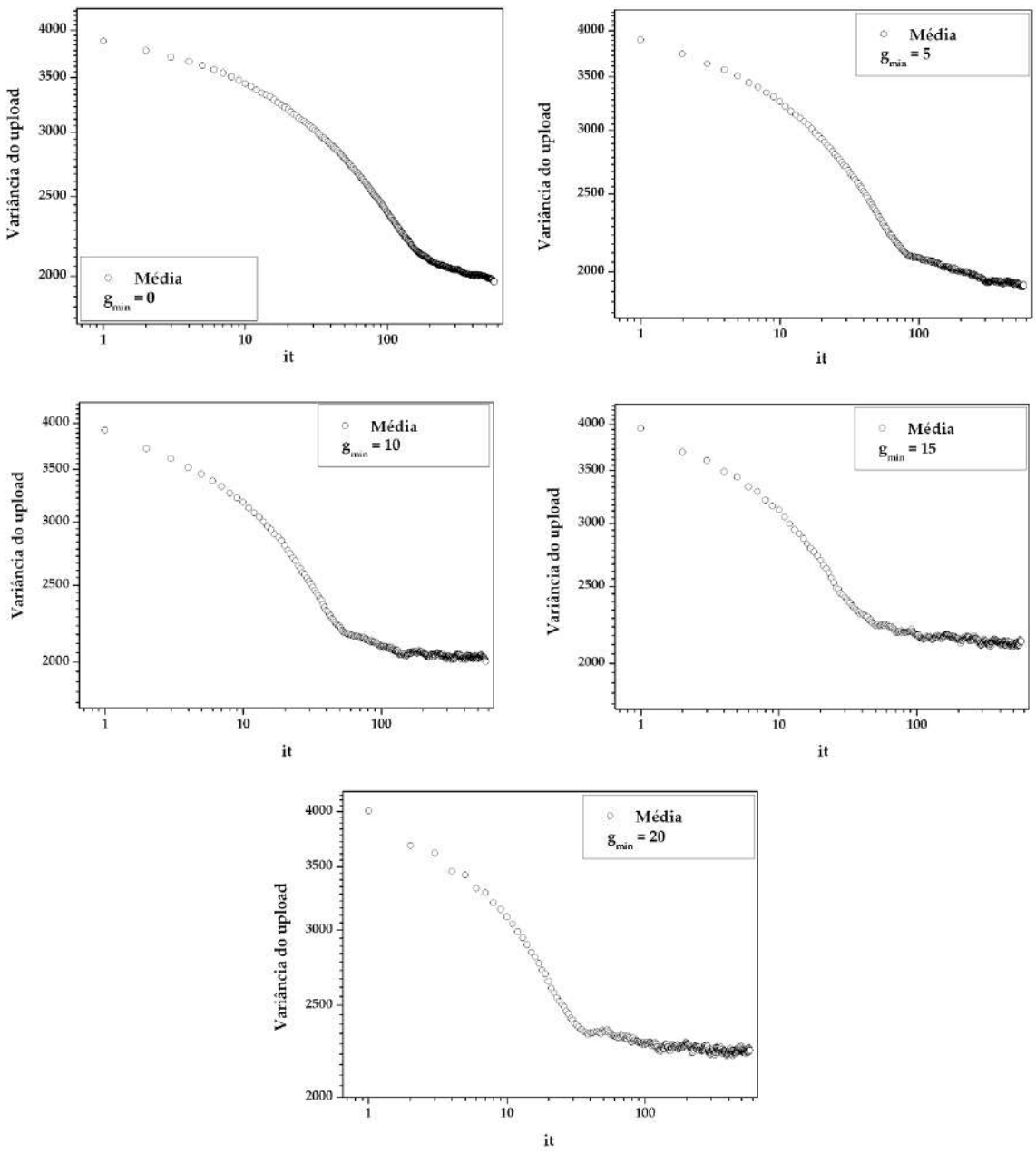

Figura A.4: Gráficos do efeito de g_{min} no decaimento da variância de *upload*, com $\beta = 0, 20$ e $g_{max} = 20$ pré-fixados