Universidade
Estadual de Londrina

ANDERSON HIROSHI HAMAMOTO

APLICAÇÃO DE ALGORITMOS GENÉTICOS PARA AUXILIAR NA GERÊNCIA DE REDES

LONDRINA–PR

2014

ANDERSON HIROSHI HAMAMOTO

APLICAÇÃO DE ALGORITMOS GENÉTICOS PARA AUXILIAR NA GERÊNCIA DE REDES

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Mario Lemes Proença Jr.

LONDRINA–PR

2014

ANDERSON HIROSHI HAMAMOTO

APLICAÇÃO DE ALGORITMOS GENÉTICOS PARA AUXILIAR NA GERÊNCIA DE REDES

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA

Prof. Dr. Mario Lemes Proença Jr.
Universidade Estadual de Londrina
Orientador

Prof. Dr. Bruno Bogaz Zarpelão
Universidade Estadual de Londrina

Prof. Dr. Sylvio Barbon Junior
Universidade Estadual de Londrina

Londrina–PR, 24 de novembro de 2014

Este trabalho é dedicado à todos que assim
como eu, estão em pé sobre ombros de gigantes.

AGRADECIMENTOS

Agradeço aos meus pais pelo suporte e carinho, aos meus amigos por deixarem a graduação menos estressante e ao meu orientador pelo tempo e dedicação.

"The best programs are written so that computing machines can perform them quickly and so that human beings can understand them clearly. A programmer is ideally an essayist who works with traditional aesthetic and literary forms as well as mathematical concepts, to communicate the way that an algorithm works and to convince a reader that the results will be correct.
(Donald Ervin Knuth, Selected Papers on Computer Science)

HAMAMOTO, A. H.. **Aplicação de Algoritmos Genéticos para auxiliar na Gerência de Redes**. 65 p. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Universidade Estadual de Londrina, Londrina–PR, 2014.

RESUMO

As redes de computadores se tornaram um meio importante para a comunicação, e manter a sua integridade e segurança é crítico e compexo. Recentemente, modelos computacionais baseados na biologia, como os Algoritmos Genéticos, têm sido aplicados na detecção de anomalias de redes de computadores. Algoritmos Genéticos são baseados na teoria da evolução e são um método de busca heurística, geralmente utilizados para solucionar de problemas de otimização. Neste trabalho, serão explorados as aplicações e os resultados do Algoritmo Genético na gerência de redes.

Palavras-chave: Redes de Computadores, Gerência de Redes, Algoritmos Genéticos, Detecção de Anomalias

HAMAMOTO, A. H.. **Applying Genetic Algorithms to improve Network Management**. 65 p. Final Project (Bachelor of Science in Computer Science) – State University of Londrina, Londrina–PR, 2014.

ABSTRACT

Computer Networks has became an important asset to communications, and maintaining its integrity and security is critical and complex. Recently, biological base computation models, such as Genetic Algorithms, have being explored to detect anomaly in computer networks. Genetic Algorithm is a heuristic search method widely used for optimization problems, based on the evolution theory. In this work, the application and results of Genetic Algorithm in network management will be explored.

Keywords: Computer Networks, Network Management, Genetic Algorithms, Anomaly Detection

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

GA	Genetic Algorithms
DSNSF	Digital Signature of Network Segment using Flow Analysis
VPN	Virtual Private Network
DoS	Denial of Service
TCP	Transport Control Protocol
IP	Internet Protocol
BP	Back-Propagation
UDP	User Datagram Protocol
MIB	Management Information Base
PCA	Principal Component Analysis
ACO	Ant Colony Optimization
HW	Holt-Winters
SDI	Sistema de Detecção de Intrusos
SNMP	Simple Network Management Protocol
IETF	Internet Engineering Task Force
IPFIX	IP Flow Information Export
RMLP	Recurred Multilayered Perceptron
NSOM	Network Self-Organizing Maps
FIRE	Fuzzy Intrusion Recognition Engine

SUMÁRIO

1 INTRODUÇÃO

A Internet revolucionou a comunicação da sociedade humana. Ela possibilita transferir informações entre longas distâncias instantaneamente e possibilitou a criação de serviços como o email e lojas virtuais. Muitas das empresas e instituições usam serviços que são acessíveis somente por meio da Internet, tornando-as dependentes da mesma. Devido a sua importância, assegurar a sua qualidade e performance se tornou um trabalho necessário.

No início, a Internet era uma rede dedicada à pesquisa criada pelo Departamento Defesa do Estados Unidos e se expandiu, conhecida como ARPANet. O seu rápido desenvolvimento reflete a sua importância para a sociedade atual. A quantidade de volume que pode ser transmitido pela rede de computadores aumenta cada vez mais com as novas tecnologias desenvolvidas. Em paralelo, os meios de transmissão também se desenvolvem, possibilitando o uso da Internet sem fio e o seu acesso por dispositivos móveis como celulares e tablets, que contribuem muito para o volume da dados transmitidos pela rede. O acesso sem fio à Internet em quase todos os lugares possibilita as pessoas se conectarem, usando os mais variados serviços disponíveis nos dias atuais. O maior exemplo são as redes sociais, que facilitam a comunicação entre as pessoas.

Ao longo da história da Internet, muitos tipos de ataques foram criados, um dos mais conhecidos é o DDoS (*Distributed Denial of Service*), que tem o objetivo de sobrecarregar um servidor indisponibilizando o serviço que este oferece. Um DoS bem sucedido pode abrir brechas para facilitar o acesso a informações que não seriam disponíveis sem autenticação. Esse ataque consiste em bombardear o servidor alvo com pacotes, mantendo-o ocupado e eventualmente deixando de fornecer serviços como deveria.

O maior ataque DDoS occorrido até 2014 foi contra o Projeto *Spamhaus*, que rastreia atividades relacionadas a *spams*, chegando a um pico de aproximadamente 400 gigabits por segundo atigindo milhões de usuários [1]. O ataque usa o NTP (*Network Time Protocol*) para fazer falsas requisições de sincronização para servidores NTP, inundando o alvo com respostas.

Além de problemas com o volume do tráfego da rede, há problemas que são cusados por softwares maliciosos. Para tanto, softwares para prevenção e remoção de falhas e programas maliciosos são usados a nível de aplicação para prevenir futuros problemas. Entre estes softwares os mais comuns são o anti-vírus e o *firewall*. O anti-vírus é um software mais completo que executa uma variedade de tarefas, sendo a principal encontrar e eliminar softwares maliciosos em um sistema computacional [2]. O *firewall* possui uma natureza mais preventiva e a sua é de impedir que software que possam causar problemas ao

usuário não entrem no sistema, servido como uma barreira entre duas redes diferentes [3].

Todos os dias novas falhas e métodos de ataques são descobertos, então há a necessidade de observar o tráfego da rede para detectar problemas sem muita demora, pois pode comprometer a rede trazendo muitos prejuízos a uma entidade. Porém monitorar o tráfego de uma rede a todo instante é uma tarefa exaustiva. Portanto softwares para auxiliar o administrador da rede nesse monitoramento são estudados e implementados. Ainda não foi criado um software adequado para realizar tal tarefa. Muitos métodos foram propostos e ainda continuam sendo estudados a fim de encontrar um que consiga monitorar o tráfego de rede e alertar o administrador quando houver algum problema eficientemente, deixando-o livre para realizar outras tarefas.

Um problema é saber se há um erro na rede de computadores e outro é identificar esse erro. Alguns exemplos de erros podem ser:

- Falha técnica: malfuncionamento de algum dispositivo na rede;
- Ataque: algum agente com intenções maliciosas (roubar informações, indisponibilizar a rede);
- Algum programa consumindo muitos recursos da rede: a execução de um software que transmita muitos dados como os serviços *peer to peer*.

Um grande problema de aplicar algoritmos de reconhecimento de padrões que são executados em um grande volume dados são ruídos. Ruídos são dados incosistentes com o que acontece de verdade, assim a análise não é fiel ao que realmente ocorre.

Este trabalho tem como objetivo aplicar o conceito de Algoritmos Genéticos para a geração de uma caracterização do tráfego da rede baseado em um histórico. Com a caracterização gerada é possível comparar com o tráfego real, auxiliando o administrador da rede a detectar comportamentos anômalos na rede.

2 TRABALHOS RELACIONADOS

O método de Algoritmos Genéticos podem ser utilizados na gerência de redes por várias abordagens. Owais et al. [4] destacam os trabalhos e as abordagens mais importantes da aplicação de Algoritmos Genéticos para detecção de intrusões. Os autore ainda concluem que a sua aplicação para o desenvolvimento de diferentes tipos de Sistemas de Detecção de Intrusões foram bem sucedidas, retornando bons resultados.

Shon et al. [5] usam um Algoritmo Genético para classificar anomalias em pacotes TCP/IP. Os autores codificam os cromossomos usando uma cadeia de 24 caracteres binários, 13 para o cabeçalho do protocolo TCP e 11 para o do IP, em que '0' significa a ausência daquele campo e '1' a sua presença. Para cada campo, duas pontuações são utilizadas: pontos de anomalia, que cresce em proporção com a frequência do seu uso para ataques, e pontos de comunicação, que representa a sua relevância durante a comunicação. Com base nessa pontuação, o algoritmo genético encontra uma função polinomial para determinar se um pacote é anômalo ou não. Ao comparar os índices de falsos positivos e correção do método proposto com outras técnicas, os autores chegam à conclusão de que o algoritmo genético obteve uma ótima performance.

Tian et al. [6] utiliza o Algoritmo Genético em conjunto com Redes Neurais para a detecção de intrusos. O artigo descreve que o uso do Back-Propagation (BP), que é um método de treinamenamento para Redes Neurais, possui problemas como a convergência para o mínimo local e lentidão no aprendizado, e Algoritmos Genéticos são rápidos e não possuem o problema do mínimo local, porém encontram uma solução que se aproxima do ótimo. Portanto, os autores propõem a combinação das duas técnicas, produzindo um método rápido e preciso para a detecção de intrusos. Os autores fazem um comparação do método proposto com o BP, chegando a conclusão de que a combinação de Algoritmos Genéticos com Redes Neurais produzem um melhor resultado e promissor para a segurança de redes.

No artigo de Jongsuebsuk et al. [7] é descrito o uso de Algoritmo Genético com a Lógica Nebulosa para detectar ataques desconhecidos. A solução proposta usa um algoritmo genético para evoluir as regras da Lógica Fuzzy e aplica essa regra para classificar os dados na fase de teste. Os dados são coletados usando um *sniffer* de pacotes, formando uma entrada a cada dois segundos extraindo informações dos cabeçalhos de IP, TCP, UDP e ICMP. Cada regra do algoritmos possui 12 características para a classificação de ataques, tais como número de pacotes TCP, número de pacotes UDP, entre outros. O último campo de cada regra é a classe de ataque a qual aquele conjunto de regras representa. Os autores obtiveram um ótimo resultado, com alta precisão e baixo índice falsos positivos, menor que 1%, sendo capaz de eficientemente detectar tanto ataques conhecidos como

desconhecidos.

Em [8] os autores aplicam o método de *K-Harmonic Means* em conjunto com o *Firefly Algorithm* para classificar comportamentos do tráfego da rede em anômalo ou não usando como dados objetos MIB (*Management Information Base*). No trabalho os autores citam que o objetivo de usar estas técnicas em conjunto é para minimizar o problema da inicialização observado no *K-Means* e diminuir o problema da solução ótima local, resultando em um algoritmo mais eficiente para agrupar dados do tráfego da rede. O método aplicado obteve um resultado com 80% de verdadeiro positivo e 20% de falso negativo, concluindo que para os testes feitos o intervalo de tempo de 300 segundos obteve os melhores resultados.

No trabalho de Carvalho et al. [9] é feito um comparativo de três métodos usados para a criação de ums DSNSF (*Digital Signature of Network Segment using Flow Analysis*) para detecção de anomalias no tráfego de rede. As técnicas avaliadas foram: PCA (*Principal Component Analysis*), ACO (*Ant Colony Optimization*) e e HW (*Holt-Winters*). Os resultados mostram que o ACO e HW geraram melhores resultados do que o PCA. Com relação à complexidade, o ACO faz um número mais elevado de computações enquanto o HW e o PCA são mais eficazes neste parâmetro. Portanto, das três técnicas analisadas, o HW produziu resultados melhores, com a menor complexidade. Os autores concluem que o ACO e o HW conseguem descrever o comportamento da rede de uma forma bem precisa, enquanto o PCA, apesar de produzir bons resultados, para o ambiente em que foram feitos os testes obteve uma menor performance.

Alguns dos trabalhos revisados que aplicam os Algoritmos Genéticos para o desenvolvimento de um sistema de detecção de intrusões, usam algum outro método para melhorar os resultados, como Redes Neurais e Lógica Nebulosa. A possibilidade de aplicar os Algoritmos Genéticos com alguma outra técnica, e a sua flexibilidade, faz com que seja uma área ampla a ser explorada para a detecção de anomalias.

3 DETECÇÃO DE ANOMALIAS

Com a importância da redes de computadores, medidas que possibilitam identificar comportamentos anômalos no seu tráfego são estudadas e analisadas, afim de encontrar um método que se ja capaz de trazer resultados confiáveis rapidamente. Essas medidas são categorizadas como Detecção de Intrusos.

Um Sistema de Detecção de Intrusões (SDI) pode ser classificado em quatro categorias [10]:

- **Baseado em *Host***: monitora o comportamento de um único *host* e procura por comportamentos suspeitos dentro deste *host*;
- **Baseado em Rede**: monitora o tráfego de segmentos da rede ou dispositivos para identificar atividades suspeitas;
- **Híbrido**: os dois tipos de SDI podem ser aplicados;
- **Análise de Comportamento de Rede**: identifica ameaças que geram um tráfego de dados anômalos, como *DoS*.

O objetivo de um SDI é verificar se há uma intrusão na rede, geralmente notificando o administrador, raramente agindo. Um SDI realiza essa tarefa analisando os pacotes que trafegam na rede, avaliando o seu comportamento. Existem duas abordagens para a aplicação de detecção de intrusos [10]: reconhecimento de assinatura e detecção de anomalias. A diferença entre as duas frentes é a abordagem para detectar possíveis intrusões.

O reconhecimento de assinaturas armazena o comportamento do tráfego de ataques conhecidos [11]. Caso o comportamento da rede seja semelhante a um comportamento armazenado, é registrado que uma intrusão pode estar ocorrendo. O problema dessa abordagem é que ela reconhece somente ataques conhecidos, e novos tipos de ataques e brechas em sistemas computacionais são encontrados e explorados todos os dias.

A detecção de anomalias aborda o problema baseado somente no tráfego da rede em questão e gera um modelo a partir do seu comportamento. Dado o histórico do comportamento padrão da rede, um modelo é gerado e comportamentos futuros são classificados como anômalos ou não. Apesar de resolver o problema de não identificar comportamentos desconhecidos do modelo de reconhecimento de assinaturas, a detecção de anomalias possui um alto índice de falsos positivos [12].

Esse trabalho explora somente a detecção de anomalias, discutindo métodos usados com ênfase na aplicação de Algoritmos Genéticos para a gereção de uma caracterização do perfil do tráfego de rede.

3.1 Gerência de Fluxos

Antes de abordar os métodos usados para a detecção de anomalias, é necessário descrever como as informações do comportamento da rede são coletadas e os protocolos definidos para o mesmo.

Com a importância e a utilidade da rede de computadores, foi necessário criar um método para monitorar e garantir o seu bom funcionamento. Com isso surgiu o SNMP (*Simple Network Management Protocol*), que é um padrão definido pela IETF (*Internet Engineering Taskforce*). Esse protocolo possui três elementos principais: dispositivo gerenciado, agente e o Sitema de Gerenciamento de Redes (NMS). O dispositivo gerenciado é um nó da rede que disponibiliza informações específicas da rede, o agente é responsável por traduzir essas informações para o formato SNMP e o NMS controla e monitora os dispositivos gerenciados. SNMP não especifica quais informações coletadas serão armazenadas, isso é feito MIB (*Management Information Base*), que também armazena essas informações.

Apesar do SNMP ser muito utilizado para a gerência de redes e apresentar um método simples e poderoso de monitorar uma rede, é muito limitado, porque apresenta somente contadores do tráfego da rede. Para tanto, empresas como a Cisco[1] e a InMon[2], criaram protocolos próprios, que tratam o tráfego da rede de uma forma mais realista e completa por meio de fluxos do tráfego de rede com os protocolos NetFlow [13] e sFlow [14] respectivamente.

Em [15] os autores definem um fluxo de rede como a comunicação entre dois *sockets*, apresentando pelo menos os endereços de saída, tempo e volume de bits, pacotes e fluxos transferidos. Em contraste com o formato do SNMP que disponibiliza somente contadores o tráfego de rede, um fluxo apresenta os endereços de IP, porta, duração e quantidade de bits e pacotes transferidos.

Com o crescimento da popularidade da gerência de redes através de fluxos, a IETF decidiu conduzir um estudo sobre os protocolos baseados em fluxos existentes para definir um padrão universal [16]. O IPFIX (*IP Flow Information Export*) [17] foi definido, sendo baseado no NetFlow da Cisco, também conhecido como NetFlow versão 10.

Além das informações que o protocolo baseado em fluxos disponibiliza, outras informações podem ser derivadas destes dados, como média de pacotes por um período

[1] http://www.cisco.com
[2] http://www.inmon.com

de tempo, média dos tamanhos de pacotes e outros. Isto possibilita uma análise mais complexa e significativa, porém aumento o tempo de processamento destes dados, uma vez que em uma rede de grande porte o tráfego é muito alto, acarretando em tempo de computação maior destes dados.

3.2 Detecção de Anomalias em Redes

Uma anomalia é um item que não está de acordo de o padrão estabelecido. No caso da rede de computadores, a normalidade pode ser caracterizado por um modelo normal do comportamento do tráfego da rede, sendo anômalo, o comportamento que foge deste padrão em certo nível. Muito algoritmos e soluções usadas para o reconhecimento de padrões podem ser aplicados para a detecção de anomalias, uma vez que em essência é o mesmo domínio de problemas.

A Figura 1 tal ilustra um exemplo de um sistema de detecção de anomalias. Primeiramente, é preciso coletar os dados a serem analisados e possivelmente fazer um processamento para aplicar o método de detecção de anomalias nestes dados e armazená-los para consultas futuras. Com os dados pode-se tomar duas ações: usá-los para gerar o detector de anomalias, uma vez que há a necessidade do histórico do comportamento da rede, ou para a análise destes dados, verificando se há alguma possível anomalia. Por fim, se alguma anomalia for detectada o administrador da rede é notificado para tomar decisões.

O sistema de alarmes de um sistema de detecção de anomalias deve ser automático e funcionar em uma margem de erro, pois mesmo que o comportamento de uma rede siga um padrão, ela é dinâmica, podendo ser modificada a qualquer momento. Muitos fatores podem intervir com o comportamento da rede, como um problema técnico, um usuário fazendo um uso inadequado da rede. Portanto é necessário a análise do administrador quando o alarme for acionado, tomando decisões devidas caso realmente seja um problema ou ignorando no caso de um falso positivo.

A margem de erro usada para o sistema de alarmes é algo que tem que ser bem elaborada, uma vez que ele não pode ser definido de tal forma que deixe problemas passarem sem alertar o administrador da rede e que não possa emitir um alerta a todo comportamento fora do padrão. Pode-se utilizar por exemplo níveis de alerta de acordo com a severidade da anomalia.

Um outro problema desta abordagem é determinar a quantidade de semanas que será usado para gerar o modelo padrão da rede. Se esse parâmetro for pequeno, ele pode não conseguir criar um modelo fiel, uma vez que há a falta de informações. Se o parâmetro for grande, a rede pode ter o comportamento alterado. Além do tempo gasto para processar os dados caso o tempo a ser analisado seja muito grande.

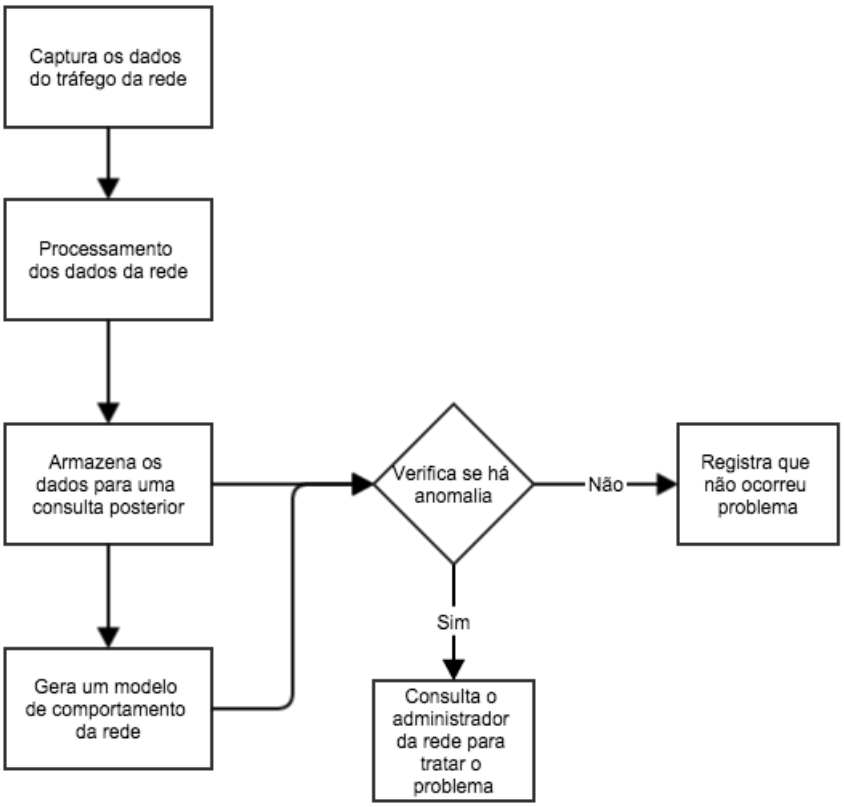

Figura 1 – Funcionamento de um Sistema de Detecção de Anomalias.

Um sistema de detecção de anomalias não consegue nos informar sobre o problema em si, ele vai conseguir simplesmente indicar que está ocorrendo um problema na rede. A identificação e o tratamento do problema ainda teria que ser feito manualmente pelo administrador da rede.

Um sistema de detecção de anomalias pode usar diferentes informações para analisar o comportamento da rede e tentar encontrar uma anomalia. Pode-se usar o comportamento de um usuário, de algum dado proveniente da rede (tráfego de bits e pacotes por exemplo) ou informações derivados de outros dados como a média de tamanho dos pacotes. Cada sistema de detecção de anomalias pode trabalhar de formas diferentes, porém muitos deles não conseguem cobrir todos os problemas que podem ocorrer dentro de uma rede.

Várias abordagens são usadas para a detecção de anomalias. Bhuyan et al. [18] define seis classes distintas de abordagens para a detecção de anomalias:

- **Estatística**: cria um modelo estatístico, testando se a nova observação pertence a esse modelo;
- **Classificações**: classifica um novo conjunto de dados em uma categoria baseado

em dados de treinamento já classificados;

- ***Clustering e Outlier***: separa os dados em grupos semelhantes de alguma forma e classifica os grupos;
- ***Soft Computing***: usa métodos que providenciam soluções inexatas mas boas e em um tempo razoável, exemplos são Algoritmos Genéticos, Redes Neurais Artificiais e Conjuntos Fuzzy;
- **Baseado em Conhecimento**: usa um conjunto de regras ou padrões prédefinidos para comparar com um novo comportamento;
- **Aprendizado com Combinações**: usa várias técnicas para criar um detector de anomalias, geralmente classificadores.

Neste trabalho, será explorada com mais detalhes a abordagem de *Soft Computing*, uma vez que o método estudado se enquadra nesta categoria. Essa categoria é bem aplicada na detecção de anomalias por se tratar de padrões que estão sujeitos a modificações em seu comportamento, fazendo com que seja complexo obter exatidão, quase sempre haverá diferenças, nem que sejam poucas.

Dentre as técnicas usadas para a detecção de anomalias pertencentes à *Soft Computing* as que mais se destacam são Redes Neurais Artificiais, Conjuntos Fuzzy e Algoritmos Evolutivos. Na literatura [18, 12], pode-se observar vários trabalhos englobando estas técnicas e algumas vezes uma mistura delas visando obter um melhor resultado.

Balajinath et al. [19] cria um sistema de detecção de intrusões baseado no comportamento de uma usuário, usando Algoritmo Genético para aprender o seu comportamento padrão. Yong et al. [20] usa um RMLP (*Recurred Multilayered Perceptron*) para para classificar dados da rede em anômalo ou normal. NSOM (*Network Self-Organizing Maps*) [21] é um sistema de detecção de anomalias que indica comportamentos que podem ser intrusões. FIRE (*Fuzzy Intrusion Recognition Engine*) [22] é um sistema de detecção de anomalias que indica se está ocorrendo alguma atividade maliciosa dentro de uma rede aplicando Lógica Nebulosa.

Bhuyan et al. [18] identificam vantagens e desvantagens do uso de *Soft Computing* para a detecção de anomalias. As vantagens dizem respeitos às características dos métodos em si, como a adaptabilidade de Redes Neurais Artificiais para este problema. As principais desvantagens são o problema de escalabilidade, disponibilidade de dados de tráfego comuns (dados possuindo anomalias vão influenciar drásticamento no treinamento) e *overfitting*, que descreve um modelo perfeitamente baseado na entrada, deixando de ser abrangente, geralmente ocorre quando há uma grande quantidade de parâmetros para uma entrada pequena.

Nos próximos capítulos serão discutidos o método de Algoritmo Genético e a sua aplicação para criação de um simples Sistema de Detecção de Anomalias.

4 ALGORITMOS GENÉTICOS

Os problemas que necessitam buscar uma solução entre a combinação da sua entrada, como o problema do caixeiro viajante, pertencem ao conjunto de problemas denominados NP (*Non-deterministic Polynomial-time*), que são extremamente difíceis de serem resolvidos, alguns não possuem uma solução computável.

Tendo em vista problemas com complexidade semelhantes ao do caixeiro viajante, soluções heurísticas são estudadas. Heurísticas são técnicas para se resolver problemas que tem o objetivo de chegar a uma ótima solução, muitas vezes não a melhor, porém uma solucão é dada ao usuário em um tempo viável.

O Algoritmo Genético é um método de busca heurístico. Essa técnica é baseada na evolução das espécies, primeiro observado por Darwin e publicado no livro "A Origem das Espécies" em 1859. Porém o modelo computacional somente começou a ser estudado nos meados de 1960 por John Holland, que publicou mais detalhadamente sobre o assunto em [23]. Esse método pertence à classe dos Algoritmos Evolutivos que são baseados na biologia.

Algoritmos Evolutivos são métodos baseados na evolução biológica e/ou o comportamento de seres vivos, tais como formigas e abelhas. Em [24], os autores comparam cinco algoritmos evolutivos aplicados a problemas de otimização, e não houve a conclusão sobre o melhor algoritmo, simplesmente alguns obtiveram um melhor desempenho em alguns casos e outros piores. Portanto heurísticas dependem muito do problema, do algoritmo e dos parâmetros aplicados no mesmo.

Um exemplo de aplicação bem sucedido de Algoritmos Genéticos é para a otimização de escalonamento de tarefas [25]. Samal et al. [26] aplica um Algoritmo Genético híbrido para implementar um escalonador de tarefas em um sistema com múltiplos processadores tolerante a falhas, os autores obteram resultados satisfatórios e soluções viáveis com o conjunto variando de 10 a 100 tarefas.

4.1 Definição do Algoritmo Genético

A implementação de um algoritmo genético depende muito do problema a ser solucionado, desde a codificação das soluções até as operações genéticas. Cada unidade de um algoritmo genético é denominado de cromossomo e geralmente é representado por uma cadeia de bits (0 e 1), em que cada caractere é chamado de gene. Porém, muitos problemas que o Algoritmo Genético pode ser aplicado possuem represetações melhores, como um número ou uma cadeia de números que representa uma ordem (permutação).

O conjunto das soluções geradas e manipuladas pelo algoritmo genético é chamado de população, e a cada iteração do algoritmo a população muda, e a população referente a uma iteração é denominada de geração.

Os cromossomos são avaliados pela função de *fitness*, que tem o objetivo de quanticar o nível de adaptação de uma solução para o problema. Por exemplo, no caso do problema do Caixeiro Viajante, a sua função de *fitness* aumentaria inversamente proporcional em relação à distência total que o caixeiro percorreria se ele seguisse a solução representada pelo cromossomo.

A operação de seleção é responsável por escolher cromossomos mais adaptados levando em consideração o valor do seu *fitness*. O *crossover* tem como função mesclar dois cromossomos, que pode ou não melhorar o seu *fitness*, providenciando variações para a próxima geração. Esse método está sujeito em encontrar uma solução ótima local, deixando de procurar pela melhor solução global, e a operação de mutação pode ajudar para que o algoritmo não sofra desse problema, providenciando uma pequena chance de alterar o cromossomo.

Mitchell [27] define um algoritmo genético genérico com 5 etapas:

1. Gerar uma população;
2. Calcular o *fitness* de todos os cromossomos da população;
3. Seleção, *Crossover* e Mutação:
    a) Seleção de um par de cromossomo com a chance se ser escolhido aumentando de acordo com o seu valor de *fitness* podendo ser escolhido várias vezes;
    b) Com a probabilidade c, fazer o *crossover* de um gene aleatório, caso o *crossover* não ocorra, colocar a cópia dos elementos selecinados na nova população;
    c) Aplicar a mutação a cada gene dos dois filhos com a probabilidade m e colocá-los na nova população;
4. Trocar a população atual pela nova população;
5. Ir para o passo 2.

4.2 Operadores e Parâmetros do Algoritmo Genético

Para aplicar o algoritmo genético a um problema, é necessário escolher os métodos de seleção, *crossover* e mutação, além dos parâmetros utilizados no algoritmo. Os parâmetros necessários são: tamanho da população, número de iterações ou alguma outra condição de parada (quando não houver melhora do *fitness* dos cromossomos por exemplo) e as probabilidades de *crossover* e mutação. Dubrovin et al. [28] concluíram que a

escolha adequada dos operadores genéticos podem reduzir o espaço de busca e promover a adaptabilidade das propriedades do algoritmo, aumentando a sua eficiência. Porém, a escolha dos parâmetros, codificação dos cromossomos e da função *fitness* possui uma influência maior do que os operadores genéticos sobre a eficiência do algoritmo [29].

A operação de seleção é a responsável por levar os cromossomos mais adequados para a próxima geração, o *crossover* combina cromossomos com o objetivo de criar melhores cromossomos e a mutação de realizar pequenas modifições aleatórias nos cromossomos visando melhorar as soluções. Portanto, o *crossover* e a mutação são as operações que promovem mudanças, e a seleção a que aprova a adaptabilidade dos cromossomos.

A seleção pode ocorrer das variadas formas, e podem ser experimentadas. Dado um problema pode-se aplicar vários métodos de seleção e comparar os resultados, usando a abordagem que resultar na melhor solução. Dentre os métodos de seleção, os mais usados são: roleta, classificação, torneio e elitismo. Cada um desses métodos estão sujeito a falhas e devem ser escolhidos levando em consideração o problema em questão.

A seleção de cromossomos em uma população pode ocorrer de várias modos, os mais comuns são:

- **Roleta**: a chance cada cromossomo ser selecionado é a proporção do seu *fitness* total em relação com o *fitness* da população inteira;
- **Classificação**: ordena os cromossosomos de acordo com o seu *fitness*, usando a sua posição como a chance de seleção, mas esse método pode aumentar o tempo levado para convergir a uma solução;
- **Torneio**: forma-se um conjunto temporário selecionando aleatoriamente cromossomos da população, colocando os melhores na nova geração. A vantagem desse método é que não há a necessidade de analisar todos os indíduos da população. na população podendo aumentar a performance do algoritmo;
- **Elitismo**: seleciona os melhores indivíduos da geração atual na próxima geração, usando algum outro método de seleção para completar a população, descartando a possibilidade de perder os melhores indivíduos da população.

As operações de *crossover* e mutação são dependentes da codificação dos cromossomos e do problema a ser resolvido. Os principais métodos de *crossover* são: ponto único, vários pontos, uniforme e aritmético. Os três primeiro são aplicados a codificação e binária e de permutação, enquanto a quarta a codificação por valores. Segue a descrição de cada método de *crossover*:

- **Ponto Único**: seleção de um ponto aleatório no cromossomo, fazendo uma troca dos genes à partir daquele ponto;

C_1	1	1	1	0	0	1
C_2	1	0	1	1	1	0
C_r	1	1	1	1	1	0

Figura 2 – *Crossover* usando apenas um ponto de cruzamento.

C_1	1	1	1	0	0	1
C_2	1	0	1	1	1	0
C_r	1	1	1	0	0	1

Figura 3 – *Crossover* usando dois pontos de cruzamento.

C_1	1	1	1	0	0	1
C_2	1	0	1	1	1	0
C_r	1	0	1	0	1	1

Figura 4 – *Crossover* usando o métodos uniforme, com genes aleatórios dos pais.

- **Vários Pontos**: seleção de vários pontos aleatórios do cromossomo, fazendo uma troca entre os segmentos feitos;
- **Uniforme**: o novo cromossomo é composto por genes selecionados aleatóriamente dos dois pais;
- **Aritmético**: é feito uma operação aritmética usando os dois pais.

Os três principais métodos de mutação são:

- **Inversão de Bit**: troca o valor do bit selecionado para mutação;
- **Permutação**: seleciona dois genes e os troca de posição;
- **Valores**: adiciona um pequeno valor ao cromossomo.

C_1	1	0	1	0	0	1
C_r	1	0	1	0	1	1

Figura 5 – Método de inversão de bit da mutação.

C_1	1	3	5	4	2	6
C_r	1	6	5	4	2	3

Figura 6 – Método de permutação da mutação.

5 PROPOSTA DE GERAÇÃO DE UM DSNSF USANDO ALGORITMO GENÉTICO

Uma abordagem para um sistema de Detecção de Anomalias para a rede de computadores é a geração de uma assinatura digital de segmento de redes (DSNSF). Um DSNSF é um conjunto de informações sobre um segmento de rede ou servidor, como volume do tráfego, número de erros e tipos de protocolos [30]. Várias técnicas já foram abordadas para a sua implementação, tais como *Principal Component Analysis*, *K-means Clustering*, *ARIMA*, *Holt-Winters* e *Ant Colony Optimization* [31, 32, 33, 34].

O grupo de redes do Departamento de Computação da Universidade Estadual de Londrina está desenvolvendo trabalhos sobre detecção de anomalias baseado na avaliação de DSNSF desde 2000. Inicialmente, os trabalhos foram desenvolvidos utilizando o SNMP como o método de coleta de dados da redes, e posteriormente começou a se usar protocolos baseados em fluxos (Netflow, sFlow e IPFIX). Esta migração decorreu porque os fluxos oferecem mais informações sobre o tráfego da rede, possibilitando avaliações mais profundas. A proposta deste trabalho é a aplicação de um Algoritmo Genético para a geração de um DSNS.

5.1 Algoritmo Genético Proposto

O Algoritmo Genético proposto tem as seguintes representações e operadores:

- **Codificação**: Numérica;
- **Seleção**: Rolera e Torneio;
- ***Crossover***: Média aritmética dos valores dos dois cromossomos selecionados;
- **Mutação**: Soma uma porcentagem do valor do cromossomo;
- **Função Objetivo**: Distância Euclidiana entre o valor do cromossomo e os dados de entrada.

O algoritmo implementado descrito em 1 segue os seguintes passos:

1. Leitura dos dados de entrada;
2. Cálculo dos limites inferior e superior;
3. Geração da população inicial;

4. Para cada ponto da população:

   a) Calcular o *fitness* da população;

   b) Seleção, *Crossover* e Mutação;

   c) Voltar ao passo a) até atingir o critério de parada.

5. retornar o melhor cromossomo de cada ponto.

```
lim ← calcula os limites inferiores e superiores;
pop ← gera uma população aleatória baseada nos limites;
CalcularFitness();
i ← 0;
while i < 288 do
    j ← 0;
    while j < número máximo de iterações do
        Seleção();
        Crossover();
        Mutação();
        CalcularFitness();
        j ← j + 1;
    end
    i ← i + 1;
end
melhor ← cria o melhor perfil selecionando os melhores cromossomos de cada
ponto;
```

Algorithm 1: Algoritmo Genético usado

Os dados de entrada são lidos à partir de arquivos em formato de texto, em que cada linha contém o volume de dados (separados em bits, pacotes e fluxos) e armazenados em vetores de 288 elementos, em que cada elemento corresponde à média do tráfego de 300 segundos (5 minutos).

O limites inferior e superior, representam o menor e o maior volume trafegados em cada elemento dos vetores de entrada. A população é gerada à partir dos limites calculados, o qual o cromossomo é um valor aleatório entre os limites.

Os valores dos cromossomos da população inicial é gerada aleatoriamente, entre os limites inferior e superior, a figura 5.1 contém um gráfico com a área de valores que podem ser gerados para cromossomos. Cada ponto do DSNSF possui uma população, então a população total é composta por 288 populações, cada ponto é evoluído separadamente.

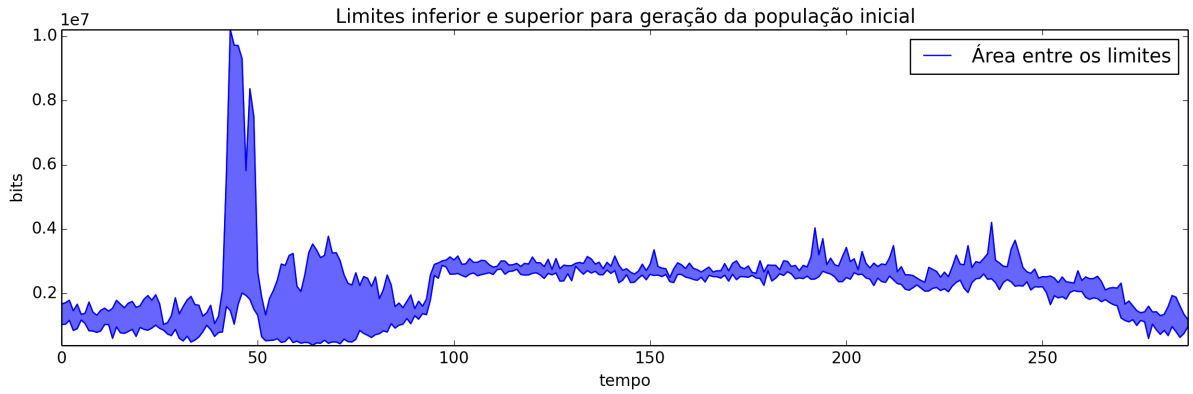

Figura 7 – Exemplo de valores possíveis para os valores de cromossomos.

O cálculo do fitness é feito usando a seguinte fórmula:

$$\sqrt{\sum_{i=1}^{n}(valor\ do\ cromossomo - entrada_i)^2}$$

O DSNSF treinado pode ser armazena em formato .txt, de forma semelhante aos dados de entrada (cada linha contém o ponto ótimo que representa a média de 300 segundos). Com isso, ele pode ser comparado com o próximo dia para a detecção de anomalias.

O *software* foi implementado com a linguagem de programação **Python**[1]. Em [35], os autores descrevem algumas vantagens de usar o **Python** para implementar um algoritmo genético, sendo a principal delas a facilidade de manipular listas e cadeias de caracteres, alta produtividade e disponibilidade de bibliotecas devido ao seu alto nível. Além disso, a facilidade de gerar gráficos para a visualização e comparação das caracterizações geradas com os dados de entrada e tráfegos futuros da rede por meio da biblioteca **Matplotlib**[2].

A implementação usa quatro dias para criar uma caracterização da rede. Caso poucos dias sejam utilizados para a geração do perfil da rede, a precisão da mesma pode ser afetada. Porém, se muitos dias forem utilizados, o comportamento do tráfego da rede pode mudar, gerando um perfil impreciso com ruídos e dados incorretos. Para tanto, 4 semanas foram usado para gerar os perfis que serão apresentados.

Como certas tarefas em uma rede podem estar escalonadas para ocorrer em um dia da semana mas não em outro, sempre é usado o mesmo dia da semana, portanto há cinco caracterizações distintas para os cinco dias úteis da semana. Então, nesse estudo de caso é usado o tráfego dos últimos 28 dias para criar a caracterização.

[1] https://www.python.org/
[2] http://matplotlib.org/

6 RESULTADOS DO MÉTODO PROPOSTO

Os parâmetros usados para gerar os perfis de tráfego da rede foram o seguintes:

- População inicial: 20 cromossomos;
- Número de iterações: 50;
- Método de seleção: Roleta e Torneio;
- Probabilidade de *crossover*: 1;
- Probabilidade de mutação: 0.1.

Foram gerados perfis de dez dias da semana, duas para cada dia útel, usando os métodos da Roleta e do Torneio. O horário usado para a análise dos resultados foi o período das 7:00 hrs às 21:00 hrs, uma vez que este é o horário mais ativo da universidade e também assim os gráficos ficam melhor de serem visualizados no trabalho. Além do perfil e do tráfego real, foram geradas também um *threshold*, que representam os limites inferior e superior para a classificação de comportamento. Foram usados como limites os valores de 20%, 30% e 40%, estes valor pode ser ajustado pelo administrador da rede de acordo com os seus critérios.

6.0.1 Análise de 08/10/2012 à 12/10/2012 em bits

Na Figura 8 é possível observar que o tráfego do dia 12/09/2012 (quarta-feira) sofreu um distúrbio no ponto 200 ao 230, e o tráfego do dia 05/10/2012 (sexta-feira) também teve um comportamento um pouco acima do normal comparado com os outros dias, o que influenciou na geração do DSNSF dos dias 10/10/2012 e 12/10/2012.

As Figuras 9, 10 e 11 apresentam os DSNSFs criados pelo Algoritmo Genético usando a seleção por Roleta, com as entradas da Figura 8 e *thresholds* de 20, 30 e 40 porcento respectivamente. As Figuras mostram 12, 13 e 14, os DSNSFs gerados pelo método do Torneio com as mesmas entradas e *thresholds*.

Há um pico no tráfego da rede do dia 11/10/2012 próximo ao ponto 225, que pode ter sido causando por um usuário fazendo o uso indevido da rede. O dia 12/10/2012 teve o tráfego muito abaixo do DSNSF gerado, o que pode ser caracterizado como um dia anômalo, pode ser que havia poucas pessoas na UEL devido a um feriado, causando uma queda no volume de dados da rede.

As tabelas 1 e 2 aprensentam a quantidade de anomalias detectadas no tráfego em bits com os métodos de seleção por Roleta e Torneio respectivamente.

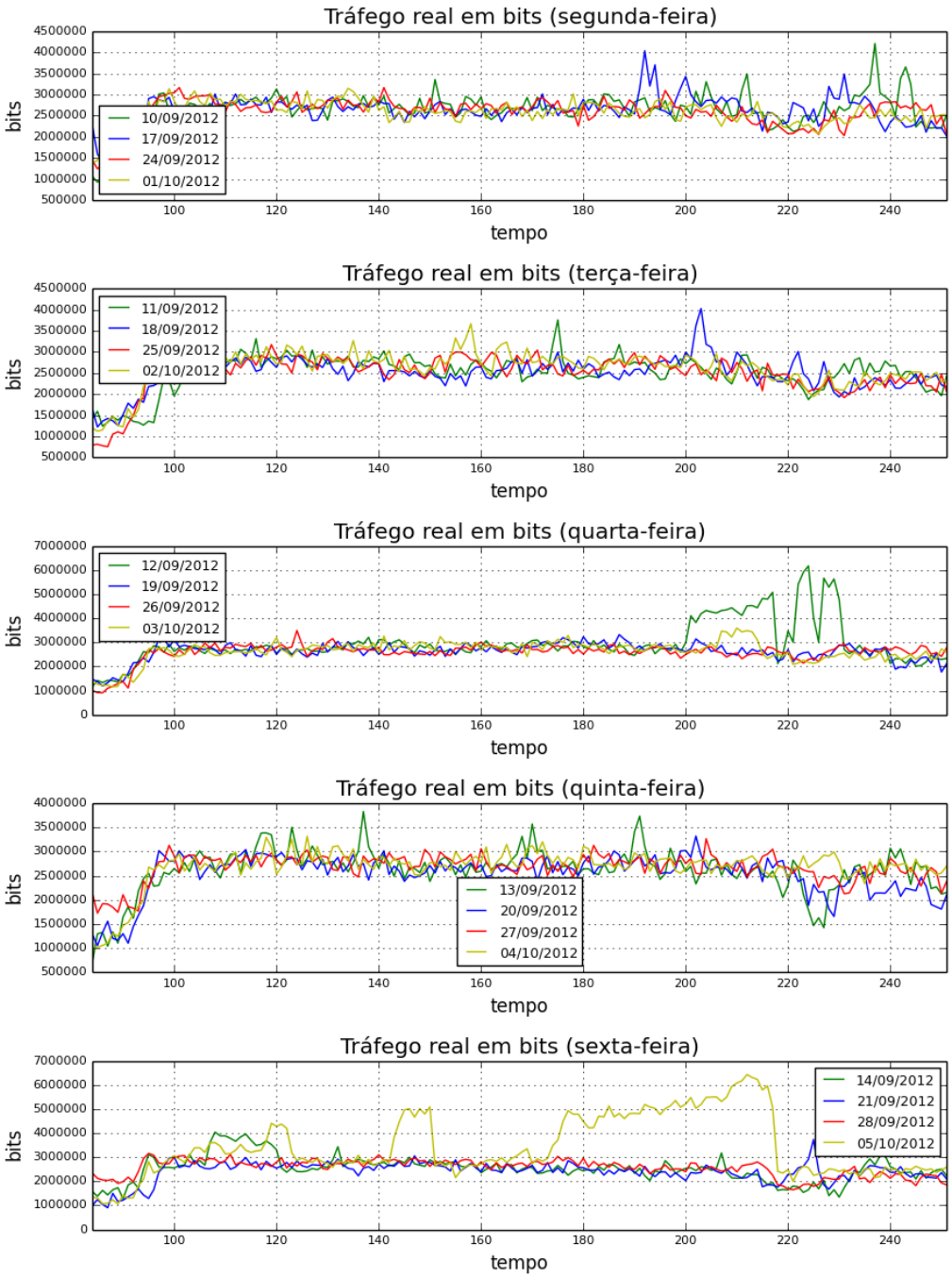

Figura 8 – Tráfego real da rede em bits dos dias 10/09/2012 até o dia 05/10/2012 (somente de segunda-fera à sexta-feira).

6.0.1.1 Análise de Bits (seleção por Roleta)

Tabela 1 – Quantidade de pontos classificados como normais e anômalos de acordo com o DSNSF gerado em bits dos dias 08/10/2012 à 12/10/2012.

Dia	***Threshold***	**Pontos Normais**	**Pontos Anômalos**
08/10/2012	20%	163	5
	30%	168	0
	40%	168	0
09/10/2012	20%	152	16
	30%	165	3
	40%	166	2
10/10/2012	20%	152	16
	30%	163	5
	40%	167	1
11/10/2012	20%	114	54
	30%	142	26
	40%	161	7
12/10/2012	20%	6	162
	30%	29	139
	40%	55	113

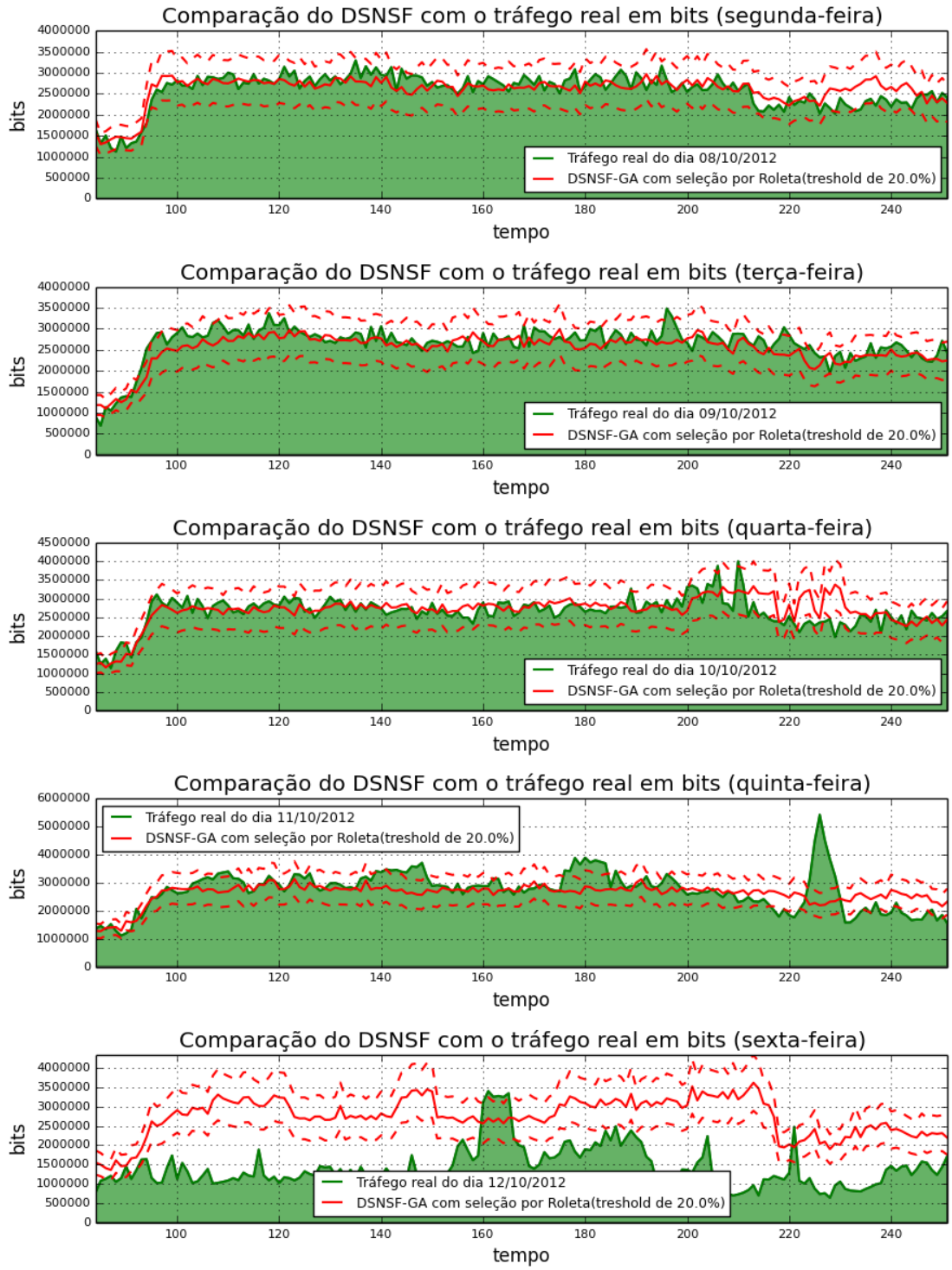

Figura 9 – Comparação dos perfis gerados em bits usando o método da Roleta com o tráfego real dos dias 08/10/2012 à 12/10/2012 com threshold de 20%.

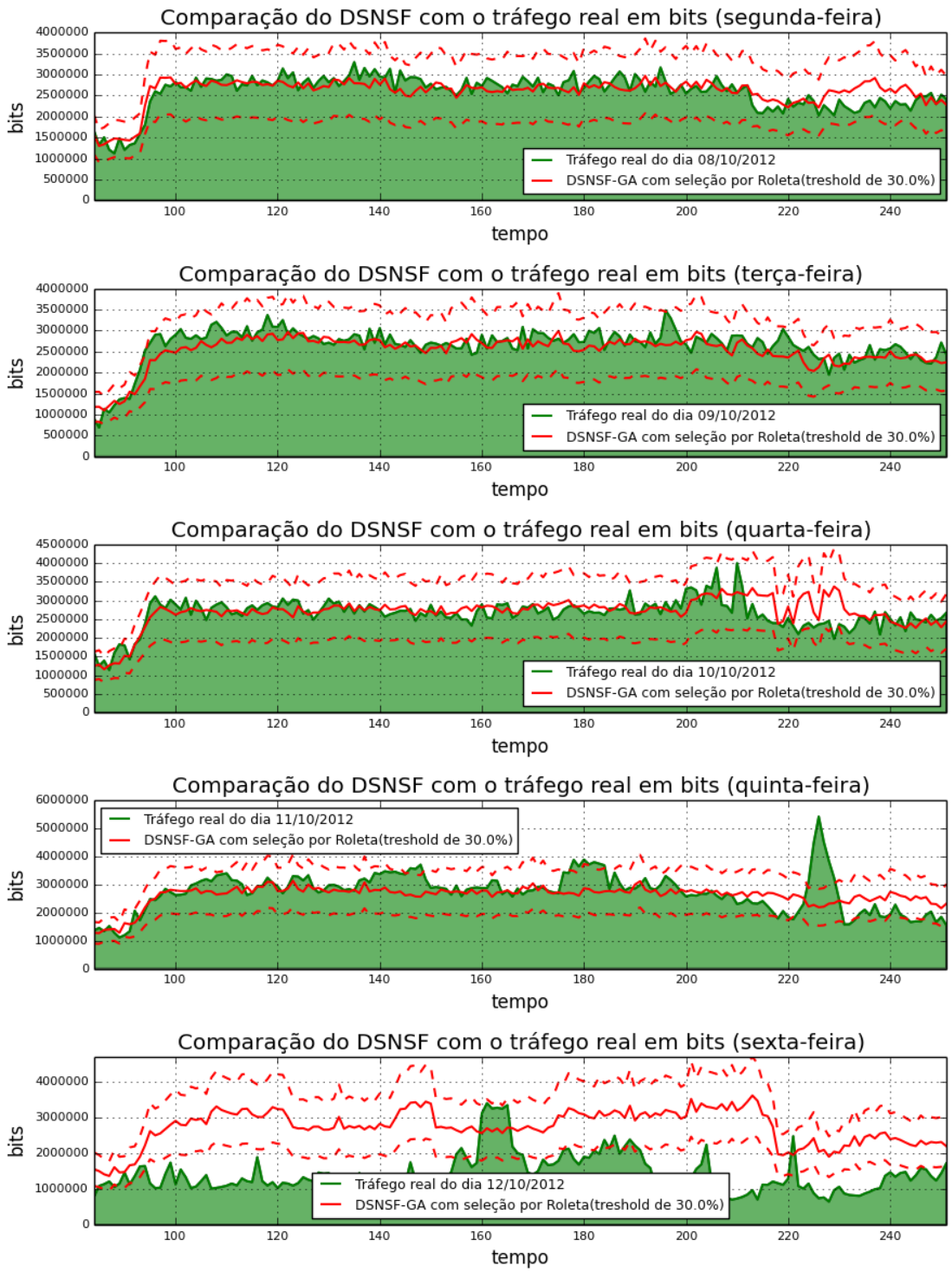

Figura 10 – Comparação dos perfis gerados em bits usando o método da Roleta com o tráfego real dos dias 08/10/2012 à 12/10/2012 com threshold de 30%.

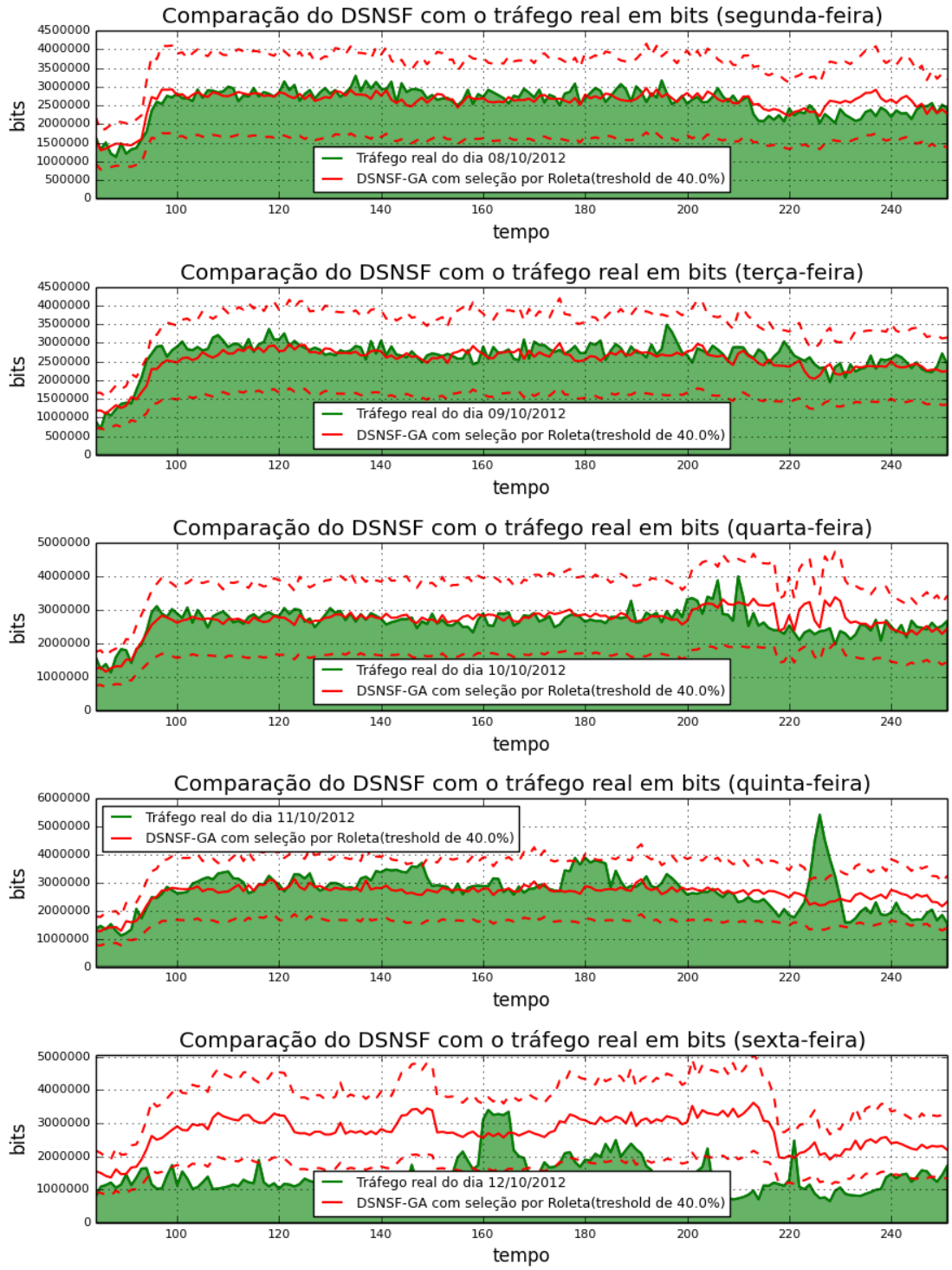

Figura 11 – Comparação dos perfis gerados em bits usando o método da Roleta com o tráfego real dos dias 08/10/2012 à 12/10/2012 com threshold de 40%.

6.0.1.2 Análise de Bits (seleção por Torneio)

Tabela 2 – Quantidade de pontos classificados como normais e anômalos de acordo com o DSNSF gerado em bits dos dias 08/10/2012 à 12/10/2012.

Dia	***Threshold***	**Pontos Normais**	**Pontos Anômalos**
08/10/2012	20%	163	5
	30%	168	0
	40%	168	0
09/10/2012	20%	154	14
	30%	165	3
	40%	166	2
10/10/2012	20%	149	19
	30%	165	3
	40%	167	1
11/10/2012	20%	114	54
	30%	143	25
	40%	161	7
12/10/2012	20%	7	161
	30%	28	140
	40%	54	114

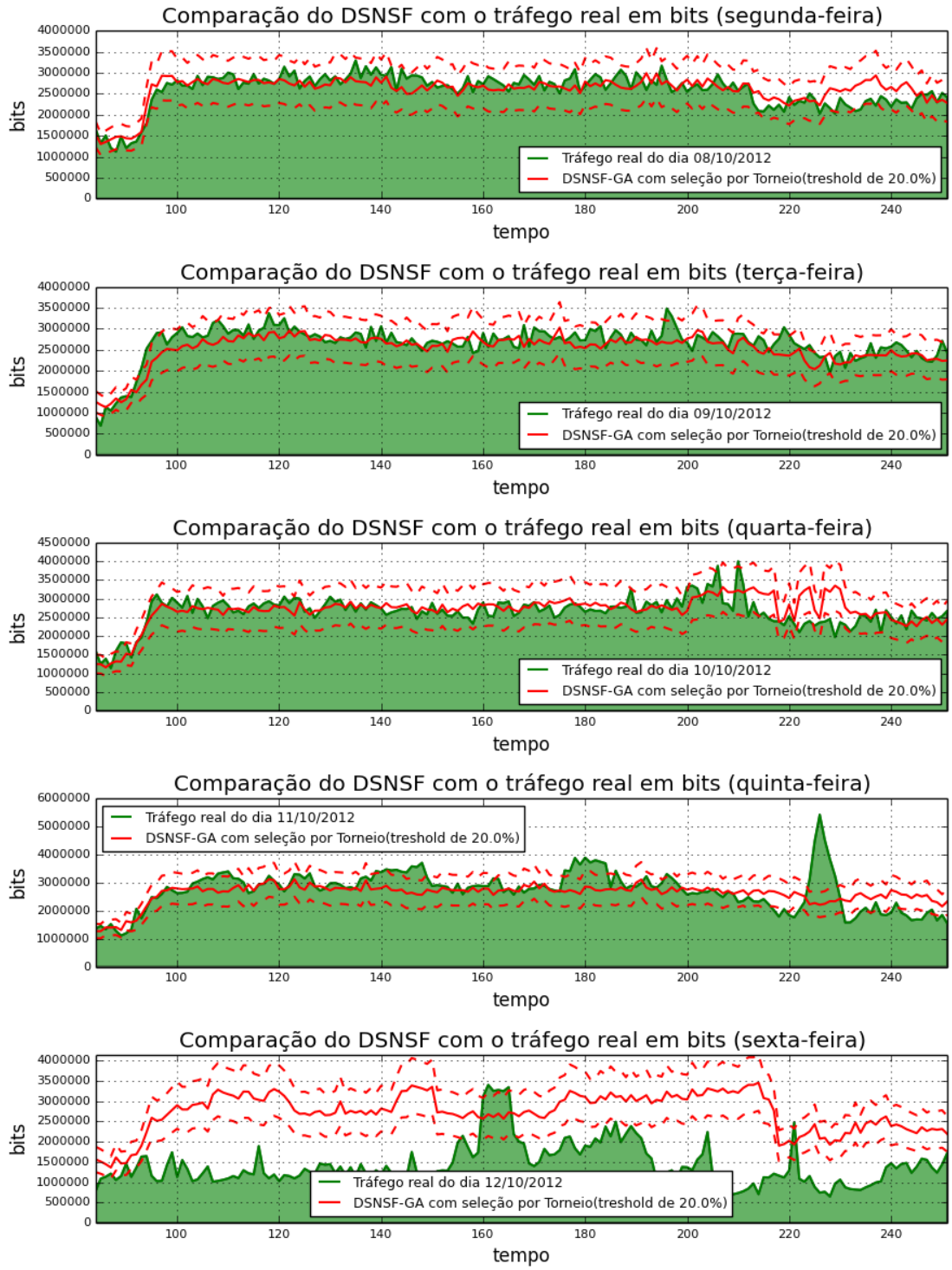

Figura 12 – Comparação dos perfis gerados em bits usando o método do Torneio com o tráfego real dos dias 08/10/2012 à 12/10/2012 com threshold de 20%.

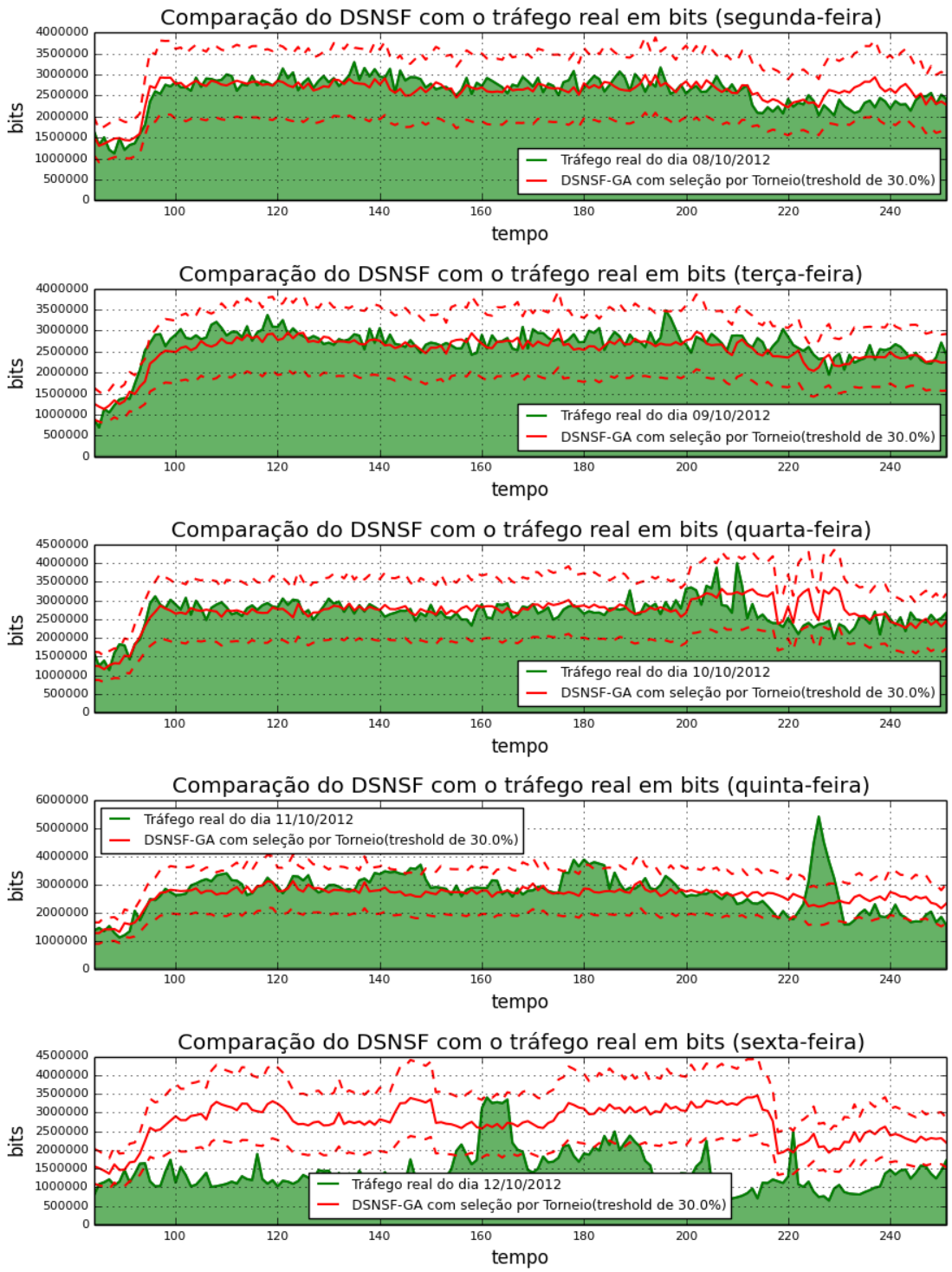

Figura 13 – Comparação dos perfis gerados em bits usando o método do Torneio com o tráfego real dos dias 08/10/2012 à 12/10/2012 com threshold de 30%.

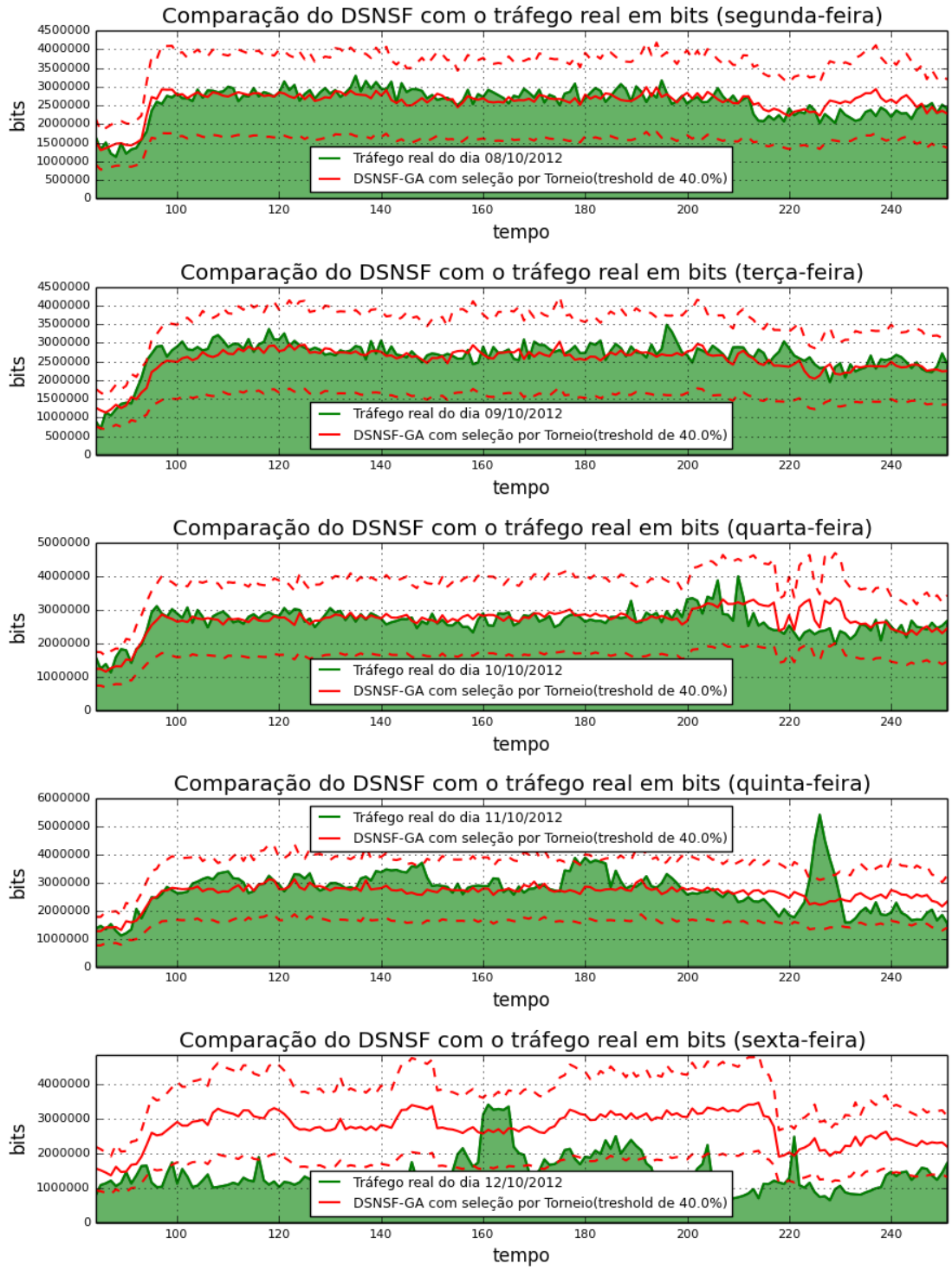

Figura 14 – Comparação dos perfis gerados em bits usando o método do Torneio com o tráfego real dos dias 08/10/2012 à 12/10/2012 com threshold de 40%.

6.0.2 Análise de 29/10/2012 à 02/11/2012 em pacotes

A Figura 15 representa o tráfego de pacotes do dia 29/10/2012 até o dia 01/11/2012. Nesta figura é possível observar que os tráfegos de quinta e sexta-feira houveram oscilações entre os dias, chegando a dobrar o volume de pacotes em comparação com outros dias.

As Figuras 16, 17 e 18 mostram os DSNSFs criados usando o Algoritmo Genético proposto com a seleção por Roleta e *thresholds* de 20, 30 e 40 porcento respectivamente com as entradas da Figura 15. As Figuras 19, 20 e 21 apresentam os DSNSFs com as mesmas entradas e *thresholds* e a seleção por Torneio.

o dia 20/10/2012 do ponto 125 ao 135 houve algum problema que aumentou signicamente o tráfego, e depois no mesmo dia à partir do ponto 230. No dia 31/10/2012 houve dois momentos que o volume de pacotes caiu drásticamente, sendo que em uma dessas instâncias o volume chegou a ser não existente, podendo ter sido causados por um problema técnico, como falha na energia elétrica. O dia 02/11/2012 teve o volume de pacotes menor que o DSNSF gerado, que pode ser sido gerado por um número menor de usuários na rede em comparação com os dias anteriores.

As tabelas 3 e 4 aprensentam a quantidade de anomalias detectadas no tráfego de pacotes com os métodos de seleção por Roleta e Torneio respectivamente. À partir delas é possível observar a semelhança na precisão da detecção de anomalias dos dois métodos.

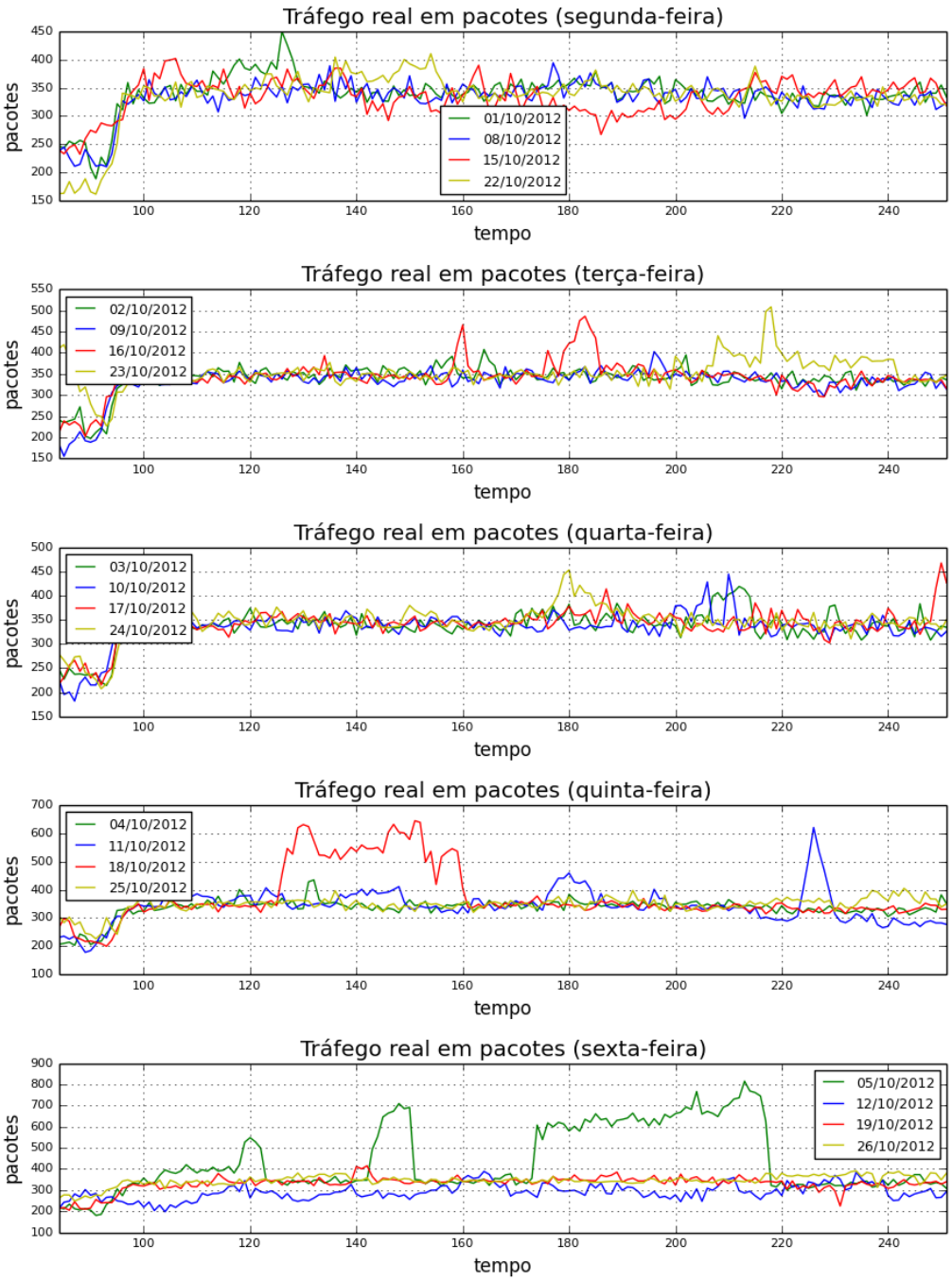

Figura 15 – Tráfego real da rede em pacotes dos dias 01/10/2012 até o dia 26/10/2012 (somente de segunda-fera à sexta-feira).

6.0.2.1 Análise de Pacotes (seleção por Roleta)

Tabela 3 – Quantidade de pontos classificados como normais e anômalos de acordo com o DSNSF gerado em pacotes dos dias 29/10/2012 à 02/11/2012 (seleção por Roleta).

Dia	***Threshold***	**Pontos Normais**	**Pontos Anômalos**
29/10/2012	20%	150	18
	30%	159	9
	40%	162	6
30/10/2012	20%	138	30
	30%	144	24
	40%	145	23
31/10/2012	20%	159	9
	30%	163	5
	40%	164	4
01/11/2012	20%	160	8
	30%	168	0
	40%	168	0
02/11/2012	20%	85	83
	30%	128	40
	40%	159	9

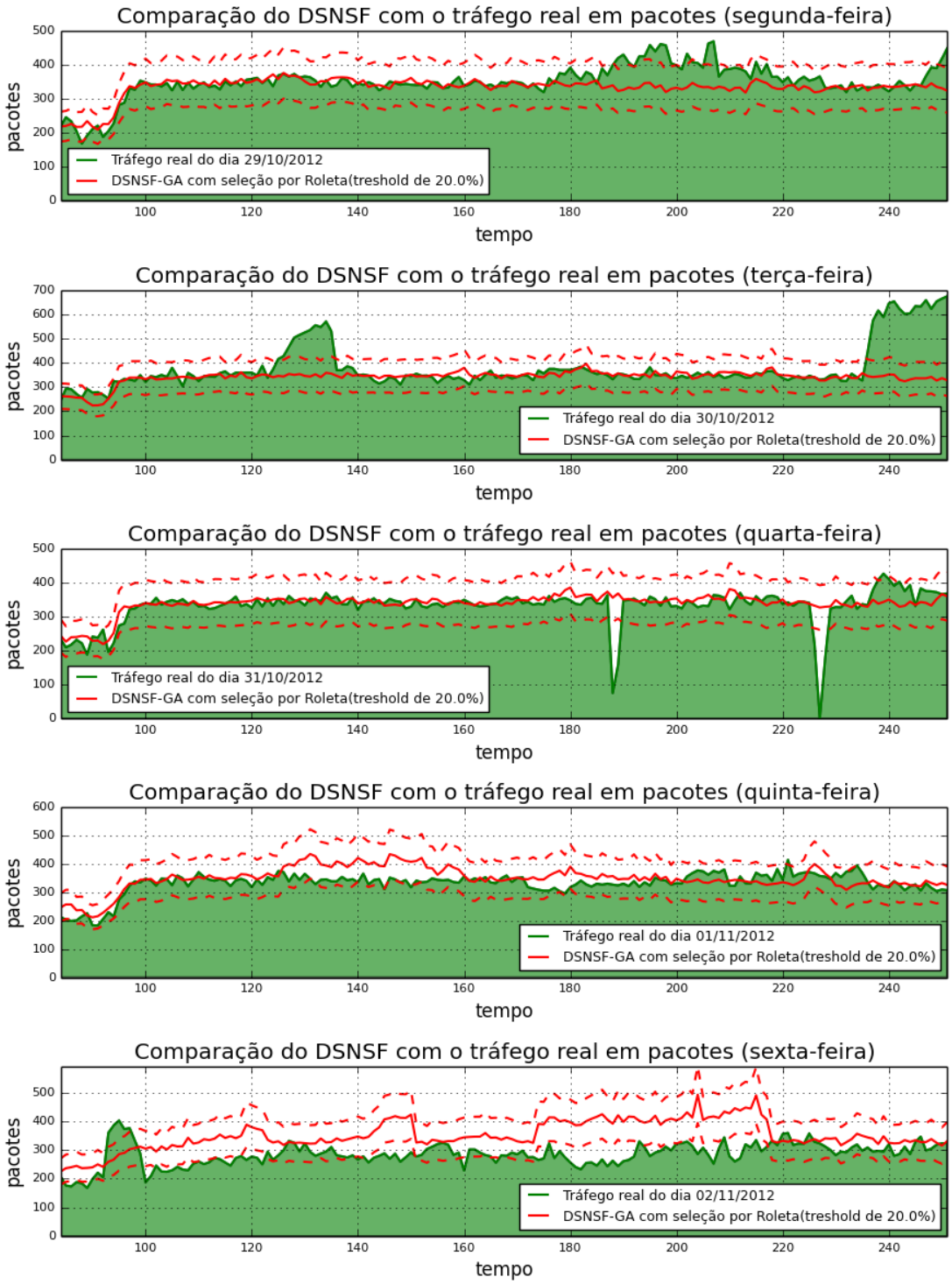

Figura 16 – Comparação dos perfis gerados em pacotes usando o método da Roleta com o tráfego real dos dias 29/10/2012 à 02/11/2012 com threshold de 20%.

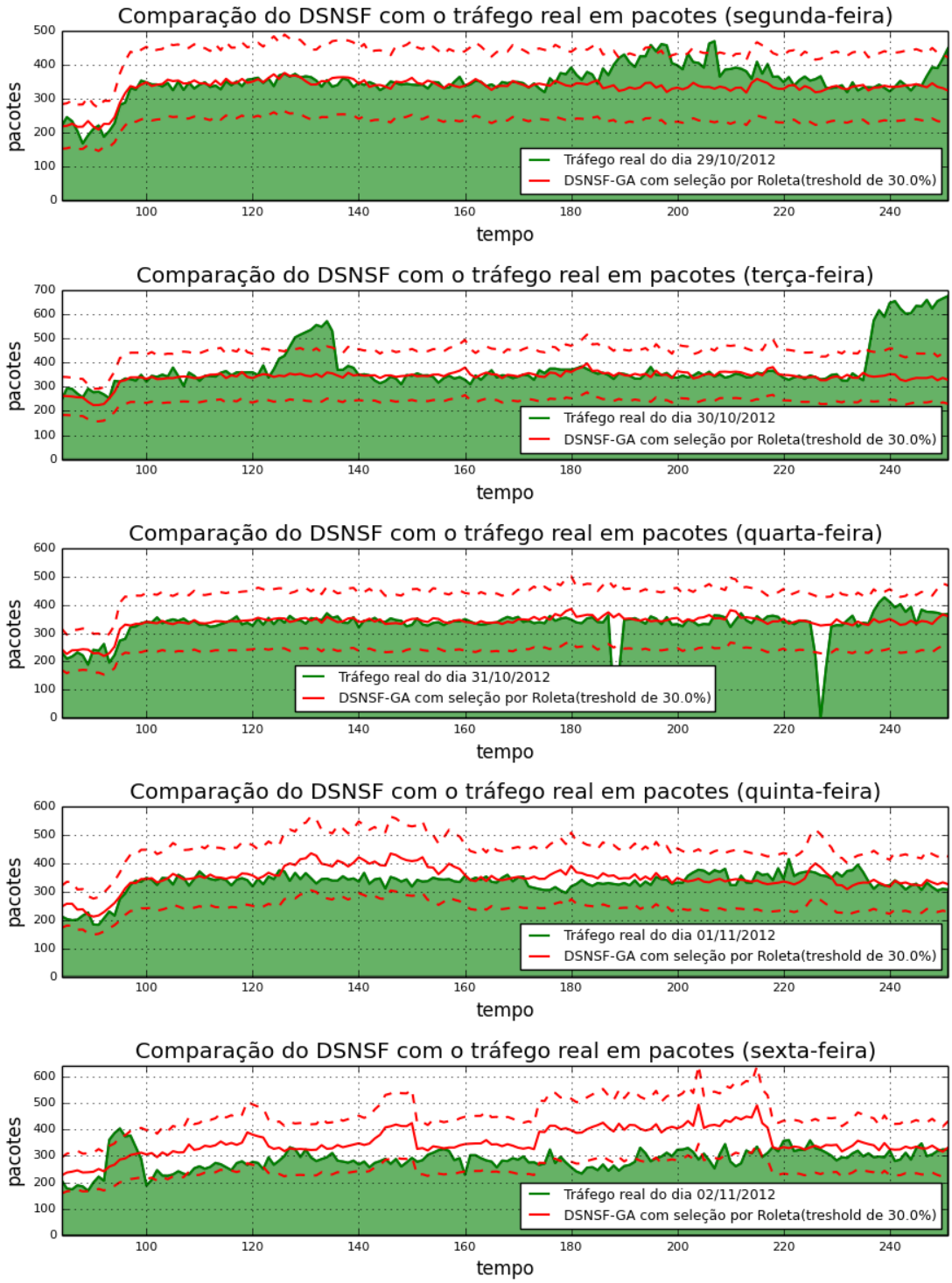

Figura 17 – Comparação dos perfis gerados em pacotes usando o método da Roleta com o tráfego real dos dias 29/10/2012 à 02/11/2012 com threshold de 30%.

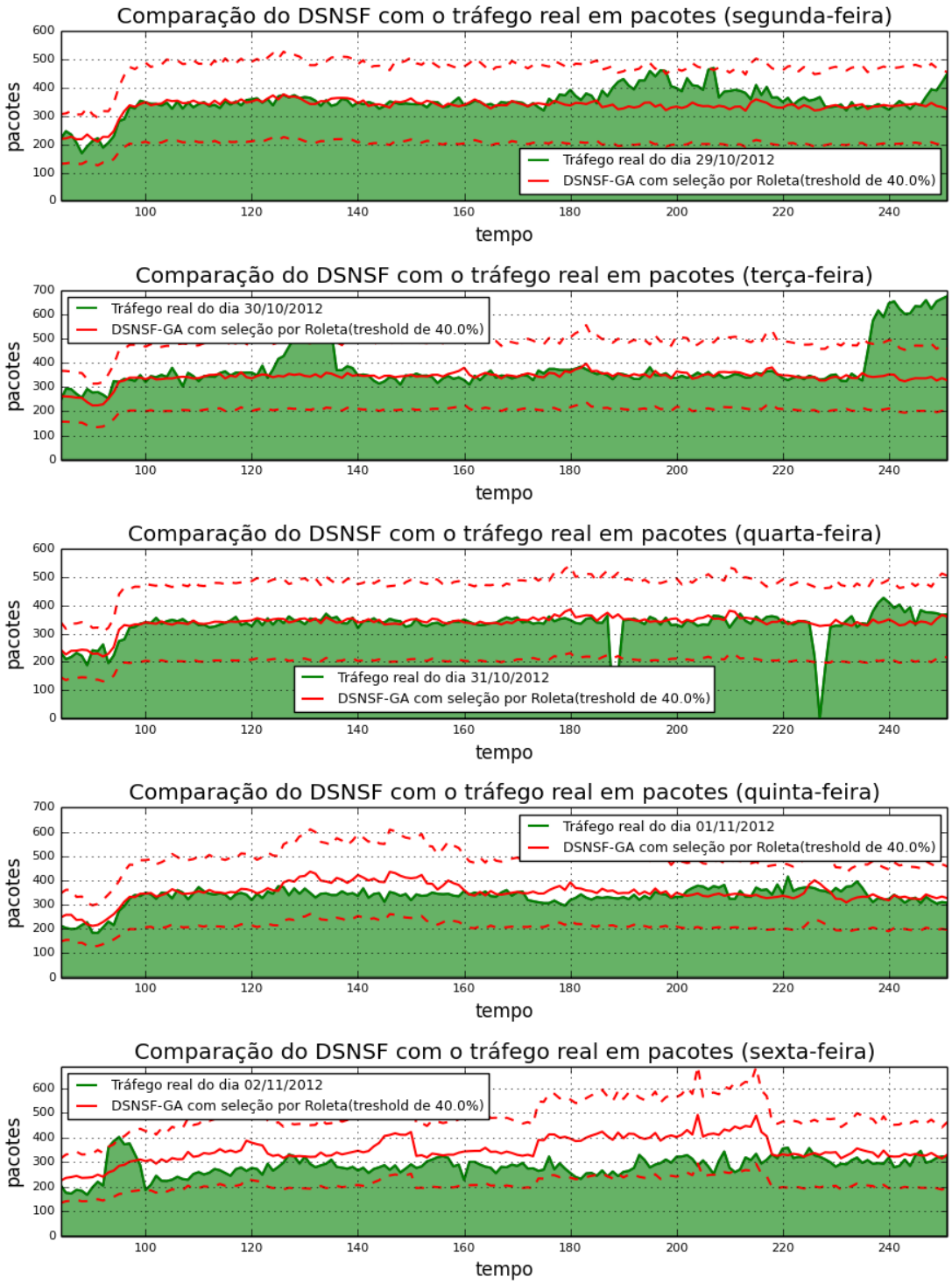

Figura 18 – Comparação dos perfis gerados em pacotes usando o método da Roleta com o tráfego real dos dias 29/10/2012 à 02/11/2012 com threshold de 40%.

6.0.2.2 Análise de Pacotes (seleção por Torneio)

Tabela 4 – Quantidade de pontos classificados como normais e anômalos de acordo com o DSNSF gerado em pacotes dos dias 29/10/2012 à 02/11/2012 (seleção por Torneio).

Dia	***Threshold***	**Pontos Normais**	**Pontos Anômalos**
29/10/2012	20%	150	18
	30%	160	8
	40%	166	2
30/10/2012	20%	139	29
	30%	144	24
	40%	145	23
31/10/2012	20%	159	9
	30%	163	5
	40%	164	4
01/11/2012	20%	161	7
	30%	168	0
	40%	168	0
02/11/2012	20%	83	85
	30%	131	37
	40%	160	8

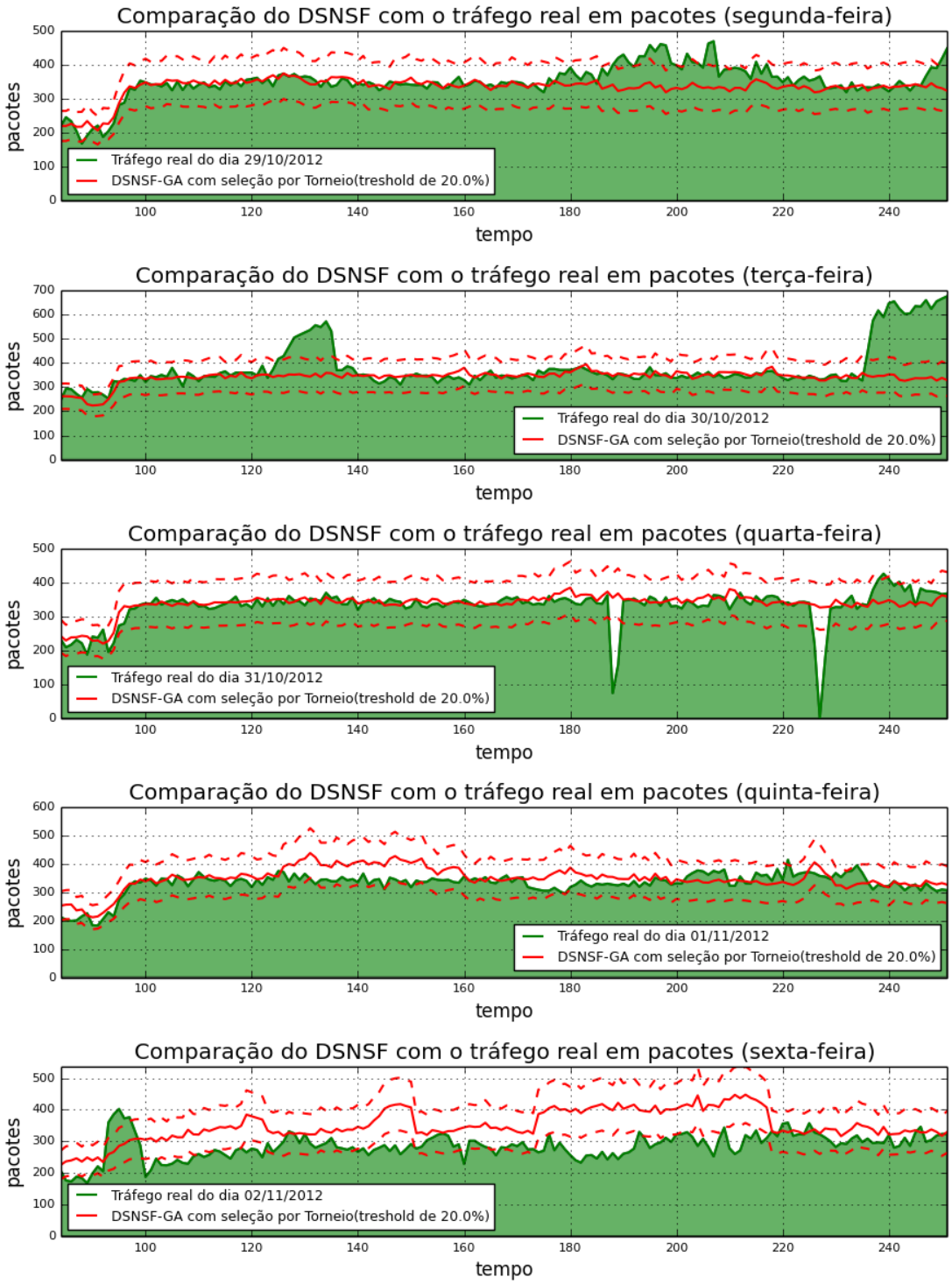

Figura 19 – Comparação dos perfis gerados em pacotes usando o método do Torneio com o tráfego real dos dias 29/10/2012 à 02/11/2012 com threshold de 20%.

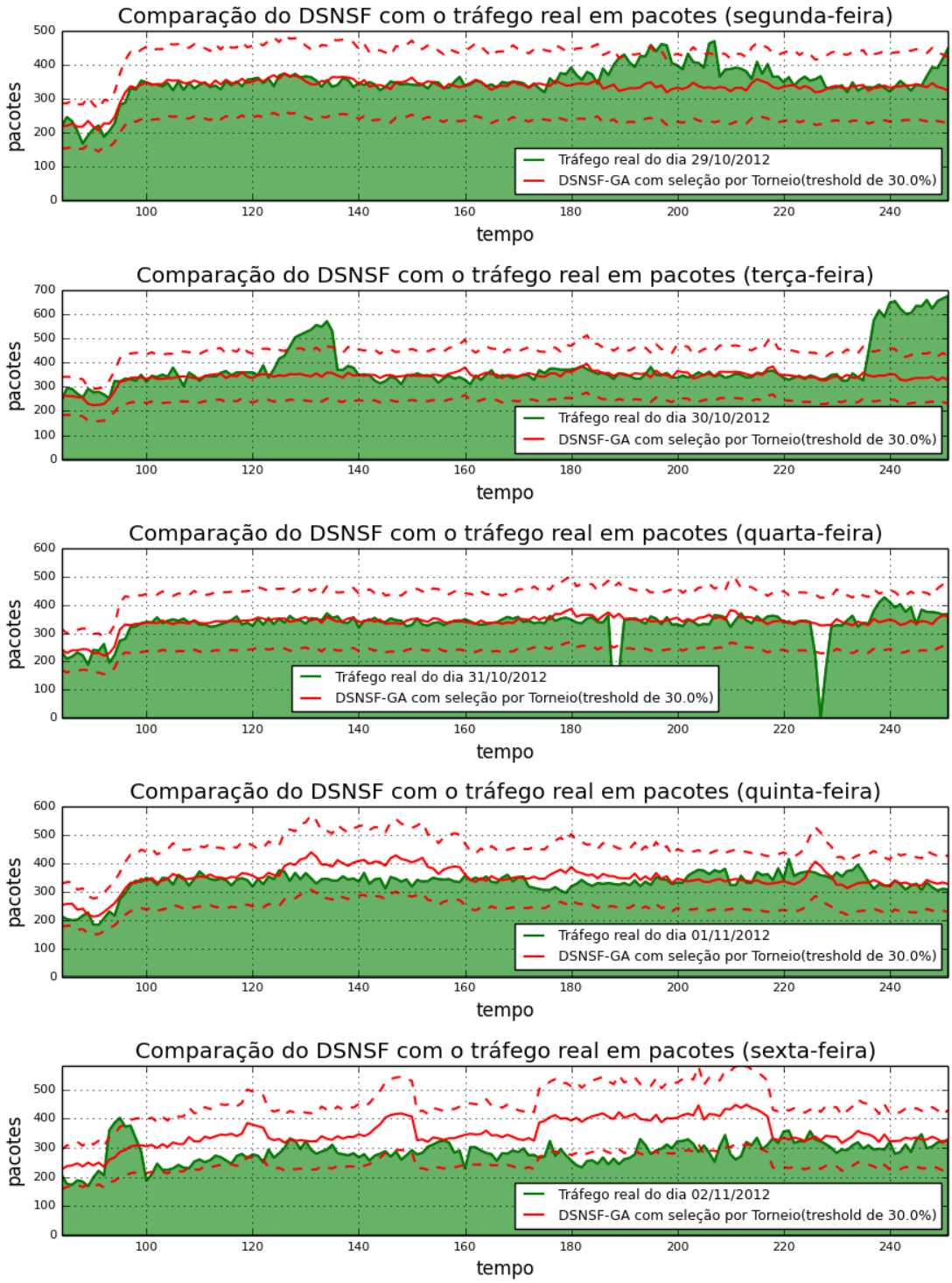

Figura 20 – Comparação dos perfis gerados em pacotes usando o método do Torneio com o tráfego real dos dias 29/10/2012 à 02/11/2012 com threshold de 30%.

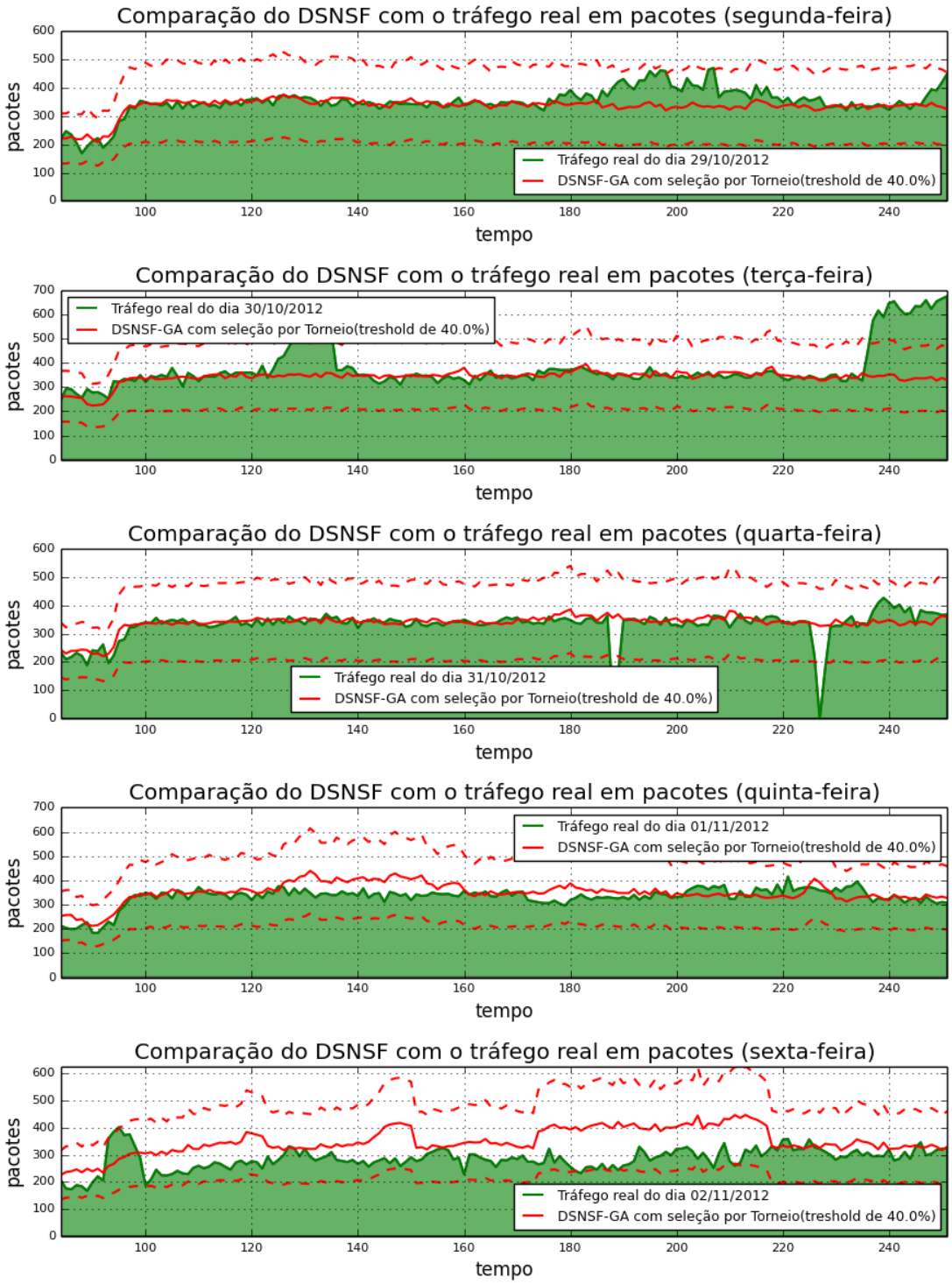

Figura 21 – Comparação dos perfis gerados em pacotes usando o método do Torneio com o tráfego real dos dias 29/10/2012 à 02/11/2012 com threshold de 40%.

7 CONCLUSÃO

O *treshold* usado para classificar um comportamento em anômalo ou não é algo a ser estudado com mais profundidade, uma vez que há uma grande diferença em usar 20% e 40%, como pode ser observado nos resultados obtidos. Se o este valor for pequeno qualquer variação no tráfego pode ser caracterizado como uma anomalias, enquanto se este valor for alto um comportamento que na verdade é anômalo não vai ser detectado pelo sistema.

É possível observar pelas Tabelas 1, 2, 3 e 4, que a quantidade de anomalias detectadas pelos métodos de seleção da Roleta e do Torneio do Algoritmo Genético para a geração de um modelo de comportamento padrão baseado nas grandezas de bits e pacotes foram semelhantes. O método de seleção por Torneio possui uma complexidade computacional de *O(1)* enquanto a seleção por Roleta possui a complexidade de *O(n)*, portanto a seleção por Torneio possui um tempo de execução menor.

As principais vantagens observadas dos Algoritmos Genéticos são:

- Tempo de convergência dos resultados;
- Facilidade de acoplar outras técnicas em conjunto com os Algoritmos Genéticos;
- Simplicidade para a criação de um modelo para solucionar um problema;
- Não estar sujeito a soluções ótimas locais (diferente de algoritmos que buscam à partir de um único ponto como *Hill Climbing*).

O trabalhos futuros são a eliminação de ruídos dos dados de entrada e a aplicação de um outro método como Redes Neurais ou Lógica Nebulosa em conjunto com o Algoritmo Genético proposto para aumentar a eficácia do DSNSF.

REFERÊNCIAS

[1] NEWS Briefs. *Computer*, v. 47, n. 4, p. 15–19, Apr 2014. ISSN 0018-9162.

[2] WANG, X. bin et al. Review on the application of artificial intelligence in antivirus detection systemi. In: *Cybernetics and Intelligent Systems, 2008 IEEE Conference on*. [S.l.: s.n.], 2008. p. 506–509.

[3] MUSTAFA, U. et al. Firewall performance optimization using data mining techniques. In: *Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International*. [S.l.: s.n.], 2013. p. 934–940.

[4] OWAIS, S. et al. Survey: Using genetic algorithm approach in intrusion detection systems techniques. In: *Computer Information Systems and Industrial Management Applications, 2008. CISIM '08. 7th*. [S.l.: s.n.], 2008. p. 300–307.

[5] SHON, T.; KOVAH, X.; MOON, J. Applying genetic algorithm for classifying anomalous tcp/ip packets. *Neurocomputing*, v. 69, p. 2429 – 2433, 2006. ISSN 0925-2312. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0925231206000907>.

[6] TIAN, J.; GAO, M. Network intrusion detection method based on high speed and precise genetic algorithm neural network. In: *Networks Security, Wireless Communications and Trusted Computing, 2009. NSWCTC '09. International Conference on*. [S.l.: s.n.], 2009. v. 2, p. 619–622.

[7] JONGSUEBSUK, P.; WATTANAPONGSAKORN, N.; CHARNSRIPINYO, C. Network intrusion detection with fuzzy genetic algorithm for unknown attacks. In: *Information Networking (ICOIN), 2013 International Conference on*. [S.l.: s.n.], 2013. p. 1–5. ISSN 1976-7684.

[8] ADANIYA, M. et al. Anomaly detection using dsns and firefly harmonic clustering algorithm. In: *Communications (ICC), 2012 IEEE International Conference on*. [S.l.: s.n.], 2012. p. 1183–1187. ISSN 1550-3607.

[9] CARVALHO, L. et al. Digital signature of network segment using pca, aco and holt-winters for network management. In: *e-Health Networking, Applications Services (Healthcom), 2013 IEEE 15th International Conference on*. [S.l.: s.n.], 2013. p. 564–568.

[10] SABAHI, F.; MOVAGHAR, A. Intrusion detection: A survey. In: *Systems and Networks Communications, 2008. ICSNC '08. 3rd International Conference on*. [S.l.: s.n.], 2008. p. 23–26.

[11] SHIRI, F.; SHANMUGAM, B.; IDRIS, N. A parallel technique for improving the performance of signature-based network intrusion detection system. In: *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*. [S.l.: s.n.], 2011. p. 692–696.

[12] ZHANG, W.; YANG, Q.; GENG, Y. A survey of anomaly detection methods in networks. In: *Computer Network and Multimedia Technology, 2009. CNMT 2009. International Symposium on*. [S.l.: s.n.], 2009. p. 1–3.

[13] CLAISE, B. *Cisco Systems NetFlow Services Export Version 9*. IETF, 2004. RFC 3954 (Informational). (Request for Comments, 3954). Disponível em: <http://www.ietf.org/rfc/rfc3954.txt>.

[14] PHAAL, P.; PANCHEN, S.; MCKEE, N. *InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks*. IETF, 2001. RFC 3176 (Informational). (Request for Comments, 3176). Disponível em: <http://www.ietf.org/rfc/rfc3176.txt>.

[15] TRAMMELL, B.; BOSCHI, E. An introduction to ip flow information export (ipfix). *Communications Magazine, IEEE*, v. 49, n. 4, p. 89–95, April 2011. ISSN 0163-6804.

[16] LEINEN, S. *Evaluation of Candidate Protocols for IP Flow Information Export (IPFIX)*. IETF, 2004. RFC 3955 (Informational). (Request for Comments, 3955). Disponível em: <http://www.ietf.org/rfc/rfc3955.txt>.

[17] QUITTEK, J. et al. *Requirements for IP Flow Information Export (IPFIX)*. IETF, 2004. RFC 3917 (Informational). (Request for Comments, 3917). Disponível em: <http://www.ietf.org/rfc/rfc3917.txt>.

[18] BHUYAN, M.; BHATTACHARYYA, D.; KALITA, J. Network anomaly detection: Methods, systems and tools. *Communications Surveys Tutorials, IEEE*, v. 16, n. 1, p. 303–336, First 2014. ISSN 1553-877X.

[19] BALAJINATH, B.; RAGHAVAN, S. Intrusion detection through learning behavior model. *Comput. Commun.*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 24, n. 12, p. 1202–1212, jul. 2001. ISSN 0140-3664. Disponível em: <http://dx.doi.org/10.1016/S0140-3664(00)00364-9>.

[20] PARLOS, A.; CHONG, K.; ATIYA, A. Application of the recurrent multilayer perceptron in modeling complex process dynamics. *Neural Networks, IEEE Transactions on*, v. 5, n. 2, p. 255–266, Mar 1994. ISSN 1045-9227.

[21] LABIB, K.; VEMURI, R. Fuzzy network profiling for intrusion detection. *Proc. 19th International Conference of the North American Fuzzy Information Processing Society, Atlanta*, v. 5, n. 2, p. 301–306, July 2000.

[22] DICKERSON, J.; DICKERSON, J. Fuzzy network profiling for intrusion detection. In: *Fuzzy Information Processing Society, 2000. NAFIPS. 19th International Conference of the North American*. [S.l.: s.n.], 2000. p. 301–306.

[23] HOLLAND, J. H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992. ISBN 0262082136.

[24] ELBELTAGI, E.; HEGAZY, T.; GRIERSON, D. Comparison among five evolutionary-based optimization algorithms. *Advanced Engineering Informatics*, v. 19, n. 1, p. 43 – 53, 2005. ISSN 1474-0346. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1474034605000091>.

[25] TEEKENG, W.; THAMMANO, A. Modified genetic algorithm for flexible job-shop scheduling problems. *Procedia Computer Science*, v. 12, n. 0, p. 122 – 128, 2012. ISSN 1877-0509. Complex Adaptive Systems 2012. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1877050912006321>.

[26] SAMAL, A. K.; MALL, R.; TRIPATHY, C. Fault tolerant scheduling of hard real-time tasks on multiprocessor system using a hybrid genetic algorithm. *Swarm and Evolutionary Computation*, v. 14, n. 0, p. 92 – 105, 2014. ISSN 2210-6502. Disponível em: <http://www.sciencedirect.com/science/article/pii/S2210650213000576>.

[27] MITCHELL, M. *An Introduction to Genetic Algorithms*. Cambridge, MA, USA: MIT Press, 1998. ISBN 0262631857.

[28] DUBROVIN, V.; FEDORCHENKO, E.; ZHYLENKO, I. Genetic algorithms operators' adjustings optimumness research. In: *Modern Problems of Radio Engineering, Telecommunications and Computer Science, 2008 Proceedings of International Conference on*. [S.l.: s.n.], 2008. p. 67–69.

[29] HOQUE, M. S. et al. An implementation of intrusion detection system using genetic algorithm. In: _____. *International Journal of Network Security Its Applications*. [s.n.], 2012. v. 4, p. 109–120. Disponível em: <http://www.airccse.org/journal/nsa/0312nsa08.pdf>.

[30] ZARPELÃO, B. B.; MENDES, L. D. S.; PROENCA JR., M. L. Anomaly detection aiming pro-active management of computer network based on digital signature of network segment. *J. Netw. Syst. Manage.*, Plenum Press, New York, NY, USA, v. 15, n. 2, p. 267–283, jun. 2007. ISSN 1064-7570. Disponível em: <http://dx.doi.org/10.1007/s10922-007-9064-y>.

[31] FERNANDES, G. et al. Digital signature to help network management using principal component analysis and k-means clustering. In: *Communications (ICC), 2013 IEEE International Conference on*. [S.l.: s.n.], 2013. p. 2519–2523. ISSN 1550-3607.

[32] PENA, E. et al. Anomaly detection using digital signature of network segment with adaptive arima model and paraconsistent logic. In: *Computers and Communication (ISCC), 2014 IEEE Symposium on*. [S.l.: s.n.], 2014. p. 1–6.

[33] ASSIS, M. de; RODRIGUES, J.; JUNIOR, M. L. P. A novel anomaly detection system based on seven-dimensional flow analysis. In: *Global Communications Conference (GLOBECOM), 2013 IEEE*. [S.l.: s.n.], 2013. p. 735–740.

[34] CARVALHO, L. et al. Ant colony optimization for creating digital signature of network segments using flow analysis. In: *Chilean Computer Science Society (SCCC), 2012 31st International Conference of the*. [S.l.: s.n.], 2012. p. 171–180. ISSN 1522-4902.

[35] LEE, W.; KIM, H.-Y. Genetic algorithm implementation in python. In: *Computer and Information Science, 2005. Fourth Annual ACIS International Conference on*. [S.l.: s.n.], 2005. p. 8–11.