CAPÍTULO 3

SIMULADOR ORIENTADO A EVENTOS DISCRETOS PARA ENCAMINHAMENTO DE MENSAGENS EM REDES OPORTUNISTAS HETEROGÊNEAS

Bernardo Sobral Werneck
Gabriel Borges da Conceição

RESUMO
Com o avanço das diversas formas de comunicação e sua necessidade nos mais diversos cenários, tem crescido o estudo no campo das redes oportunistas, as quais se baseiam em aproveitar os eventuais encontros de dispositivos comunicantes (nós), num ambiente de mobilidade complexa, como oportunidade para encaminhamento de informações ponto a ponto objetivando alcançar determinado destino. Com isso, diversos pesquisadores têm se empenhado em desenvolver técnicas ou modelos matemáticos capazes de prever o comportamento desse tipo de rede, analisando aspectos como sua eficiência e atraso na entrega de mensagens. Concomitantemente, é fundamental a existência de ferramentas de auxílio a esses estudiosos, de forma a simular diversos cenários de comportamento das redes, apresentando indicadores confiáveis para que se possam medir ou comparar as previsões para essa rede. Sendo assim, este trabalho objetivou desenvolver um simulador de redes oportunistas baseado em eventos discretos. O desenvolvimento do simulador foi concluído mediante o previsto para este projeto de fim de curso e em sua versão atual oferece, como principais métricas, cálculo da porcentagem das mensagens entregues, bem como seus atrasos médios numa rede configurada pelo usuário, com opções de política de encaminhamento (p,q)-Epidemic, Single Copy Epidemic, Two-Hop, Direct Delivery, Spray And Wait e Binary Spray and Wait, com implementação de três diferentes formas de geração de mensagens pelos nós. A geração do trace de encontros ocorre segundo distribuição de probabilidades exponencial, usando como parâmetro as taxas de encontro. Neste caso, diz-se que a rede é homogênea quando todos os pares de nós possuem mesma taxa e heterogênea quando elas não forem todas iguais, caso em que ainda é possível admitir taxa de encontro variável para cada par. Além das duas principais métricas mencionadas, o resultado de uma simulação ainda provê alguns relatórios como o trace de encontros gerado, relatório de eventos de mensagens - uma descrição temporal dos eventos de geração, encaminhamento e entrega de mensagens - e o relatório de entrega de mensagem - especificando, para cada mensagem, informação sobre o instante de geração dessa mensagem com a identificação dos nós de origem e destino, bem como seu status de entrega, dizendo se a mensagem foi entregue ou não durante todo o tempo de execução da simulação e indicando

o instante caso a entrega tenha ocorrido. Por fim, o simulador dispõe de duas interfaces de usuário: uma aplicação web e uma interface de linha de comando, permitindo tanto a familiarização do usuário com a ferramenta de forma simples e clara, bem como seu uso mais direto e customizado.

Palavras-Chave: simulação. eventos discretos. redes oportunistas. trace de encontros. atraso de entrega. encaminhamento de mensagens. distribuição de probabilidade.

INTRODUÇÃO

Cada vez há maior interesse no estudo de redes sem fio móveis e sua aplicabilidade para desafios da sociedade atual, no que tange à mobilidade humana e à necessidade de troca de mensagens entre seus dispositivos, os quais podem ser tratados como nós. No entanto, a dificuldade de prever os diferentes comportamentos de sua movimentação pode trazer empecilhos nessa direção. Daí surgem as redes oportunistas, as quais visam aproveitar os encontros dentro dessa movimentação complexa como uma oportunidade para troca de mensagens.

Em linhas gerais, o encaminhamento oportunista de mensagens é capaz de permitir comunicação em ambientes cuja aplicação de protocolos de redes tradicionais não promovem bom funcionamento, como é o caso de cenários de redes ad hoc móveis. Pelo fato de não serem protocolos tradicionais, a modelagem de protocolos oportunistas é um problema em aberto. Na maioria dos casos, para avaliar o comportamento desse tipo de rede, é necessária a utilização de ferramentas de simulação para realizar a análise de desempenho desse tipo de comunicação. Por isso, existem alguns simuladores focados especificamente nesse contexto, como forma de prover ferramentas de análise confiáveis sobre cenários desafiadores e ainda não muito bem controlados.

Além disso, um encontro pode ser definido como uma circunstância em que dois nós estejam aptos a trocar dados entre si, de acordo com alguma política de encaminhamento. Vale ressaltar que, numa rede oportunista, não são necessários nós centrais ou mesmo rotas para ocorrer encaminhamento de dados; cada nó transmite suas informações diretamente a outro nó quando for considerado que a situação se configura como uma boa oportunidade de prover a entrega da mensagem ao nó destino, de acordo com o protocolo aplicado ao contexto.

Na prática, ainda não há tecnologia disponível para aplicação de protocolos de comunicação do tipo oportunista em dispositivos mais comuns como celulares e com- putadores, por exemplo. No entanto, deseja-se que isso seja possível ao longo dos anos com o avanço das pesquisas em prol desse propósito. Para isso, há diversos pesquisadores empenhados neste campo, visando desenvolver modelos matemáticos capazes de prever e estimar o comportamento dessas redes. Com isso, ergue-se a necessidade

Editora Epitaya | Rio de Janeiro-RJ | ISBN 978-85-94431-65-3 | 2024

também da concepção de ferramentas computacionais que deem suporte a tais estudos, promovendo simulações de diversos cenários de forma a disponibilizar parâmetros confiáveis que sirvam de análise para o comportamento de determinada rede oportunista.

Este artigo objetivou desenvolver um simulador orientado a eventos discretos para encaminhamento de mensagens em redes oportunistas heterogêneas, o qual é capaz de medir, entre outras métricas, o atraso médio na entrega das mensagens, bem como sua taxa de entrega e desvio padrão, juntamente com diversos outros relatórios da simulação disponibilizados ao usuário. Dentre suas características, o simulador é eficiente em seus cálculos e permite simples configuração de parâmetros. Para sua utilização, devem ser fornecidos como entradas, dentre outros parâmetros, as taxas de encontros dos pares de nós, o protocolo de encaminhamento oportunista das mensagens, assim como o tipo de geração de mensagens na rede. Além disso, foi alcançado também o objetivo de viabilizar maneiras simples de instalação, utilização e posterior contribuição ao simulador pela comunidade científica.

É importante realçar que o simulador é baseado em eventos discretos. Assim, dado o fim de um evento (encontro de nós ou geração de mensagem, por exemplo), o relógio do sistema avança diretamente para o instante do próximo, contrastando com a modelagem contínua, na qual cada fatia do tempo deve ser executada. Em linhas gerais, a simulação discreta possibilita uma análise mais simples de um processo complexo, observando o processo em instantes discretos do tempo, além de possibilitar análise do comportamento de um sistema dinâmico por meio de uma abstração na qual o sistema é representado por uma sequência de ocorrências instantâneas (eventos).

Já existem alguns simuladores que podem ser aplicados em cenários de redes oportunistas como o ns-3 1 e o The ONE 2 (The Opportunistic Network Environment). Todavia, o simulador desenvolvido no presente projeto foca em permitir ao usuário definir de forma direta as taxas de encontro dos pares de nós, além da possibilidade de elas serem variáveis ao longo do tempo. Os resultados de uma simulação são sensíveis à mudanças nos valores dessas taxas, por isso, é importante que o analista tenha controle sobre a definição delas e a possibilidade de analisar como as mesmas afetam os resultados.

Mais especificamente, essas taxas estão relacionadas às características de mobilidade dos nós e servem para determinar os instantes de encontro para cada par (gerando o chamado trace de encontros), segundo alguma distribuição de probabilidades escolhida, no caso desse simulador, a distribuição exponencial. O trace de encontros representa a descrição temporal dos instantes em que ocorrem encontros de cada par de nós e é um importante parâmetro para a simulação.

É importante destacar que não há nenhum simulador conhecido que possua a funcionalidade de permitir configuração dos encontros dos pares

diretamente a partir de taxas de encontro, principalmente nas situações de comportamento heterogêneo e variável ao longo do tempo; sendo o mais comum que os encontros sejam definidos a partir de modelos de mobilidade dos nós. E foi justamente esta lacuna o principal foco do simulador desenvolvido no presente projeto.

METODOLOGIA

Primeiramente, foi realizado um estudo sobre redes oportunistas e os diversos conceitos relacionados, como encontro de nós, políticas de encaminhamento de mensagens e medidas de atraso na entrega de mensagens numa rede.

Em segundo lugar, foi também necessário ambientar-se com os princípios da simulação orientada a eventos discretos, na qual o simulador foi baseado. Juntamente a isso, tomou-se conhecimento do estado da arte desse tipo de simulação aplicada ao campo das redes oportunistas, na qual estão contidos os simuladores já mencionados ns-3 e The ONE.

Após estudos de viabilidade, foi decidida a utilização da linguagem de programação Java, para o controle da simulação e da interface de linha de comando, e da linguagem de programação ECMAScript com a biblioteca ReactJS para interface gráfica da aplicação web. Todo o processo de simulação foi planejado a ser totalmente contido no módulo feito em Java. No entanto, para facilitar a interação com o usuário e tornar o simulador mais claro em seu uso, esse módulo é colocado num servidor (backend) em contato com uma interface web (frontend), sendo conectados através de uma API REST. E, além desta utilização, ainda é possível que o usuário interaja mais diretamente com o simulador através de sua interface de linha de comando, a qual permite configuração da simulação a partir de arquivos escritos pelo usuário, mas com sintaxe e parâmetros pré estabelecida pelo simulador.

A primeira etapa de desenvolvimento consistiu em criar a estrutura de um caminho de simulação completo, ainda que com poucas opções para os parâmetros da rede. Esta primeira fase foi capaz de gerar o trace de encontros a partir da taxa de encontros dos pares segundo a distribuição exponencial e executar a simulação segundo o protocolo epidêmico, resultando no cálculo do atraso de entrega médio das mensagens da rede. A validação da geração do trace de encontros resumiu-se à comparação gráfica entre o método para geração de variáveis aleatórias utilizado e o gráfico real da respectiva função distribuição acumulada, segundo a Seção 5.1.

As fases subsequentes contiveram a validação do cálculo do atraso médio, taxa de entrega e desvio padrão para as mensagens da rede; estudo e implementação de diversas outras políticas de encaminhamento para a transmissão das mensagens, além da criação de três diferentes maneiras de geração de mensagens, as quais foram capazes de englobar todas as configurações dos cenários de teste e validação utilizados quando no

Editora Epitaya | Rio de Janeiro-RJ | ISBN 978-85-94431-65-3 | 2024

desenvolvimento do simulador, e que devem abranger também a maior parte dos cenários desejados pelo usuário.

TECNOLOGIAS UTILIZADAS

Para o desenvolvimento do simulador, foi necessária a escolha das ferramentas e linguagens de programação que melhor se adequassem ao projeto.

Para o pacote de simulação, principal produto do presente trabalho, era necessária uma linguagem que pudesse ser facilmente organizada e separada em módulos, para facilitar o entendimento do código e a abstração das diferentes partes do sistema. Uma linguagem orientada a objetos seria ideal, permitindo a criação de interfaces e classes abstratas, muito adequadas ao presente projeto. Além disso, seria interessante experiência prévia da equipe na linguagem. Por esses motivos, a linguagem escolhida foi Java1.

Para a interface gráfica do projeto, foi escolhida a implementação de uma aplicação web, por ser de mais rápida implementação e ser uma forma amplamente difundida de interfaceamento com o usuário, sendo portanto mais facilmente assimilada pelo mesmo. Para isso, procurou-se um framework de desenvolvimento web, de modo a agilizar a criação da interface e se apropriar de sua estruturação. Por esses fatores, foi escolhido o framework React2 para desenvolvimento da interface.

Java

Java é uma linguagem de programação orientada a objetos extremamente difundida, sendo uma das mais usadas no mundo. A divisão do código em classes, com possibilidades de extensão por herança e aplicação de interfaces, são características extremamente importantes para o presente projeto. Além disso, o caráter de ferramente livre com código aberto faz com que a difusão da linguagem seja extremamente importante, pois permite que outras pessoas contribuam no projeto fazendo alterações necessárias, tanto para usufruto próprio em casos específicos, quanto para extensão do mesmo quando em contribuições pertinentes para a comunidade.

React

A biblioteca de ECMAScript React é um framework para facilitar desenvolvimento de aplicações web. Utilizando componentes baseados em estados, essa biblioteca permite que o desenvolvedor abstraia das complexidades da renderização de HTML baseado em variáveis do sistema.

DESENVOLVIMENTO DO PROJETO

O presente capítulo explicita as etapas do desenvolvimento do simulador, com todas as suas funcionalidades, maneiras de utilização e resultados do produto. O código fonte de desenvolvimento do projeto encontra-se em um repositório no *github*[1].

Determinação dos módulos

Foi determinado que a ferramenta seria dividida em 2 principais módulos: simulador e aplicações.

O simulador é o principal módulo do projeto, sendo o responsável pela execução das simulações nos parâmetros explicitados. Ele deve ser um módulo à parte tanto para organizar o desenvolvimento, quanto para possibilitar que usuários o utilizem sem depender da aplicação. Quanto à aplicação, pode haver diferentes frentes, tendo sido planejadas as aplicações Web e de linha de comando (Command Line Interface). O módulo do simulador recebe objetos de configuração bem definidos, cabendo à aplicação utilizada realizar a tradução das informações cedidas pelo usuário para a configuração correspondente.

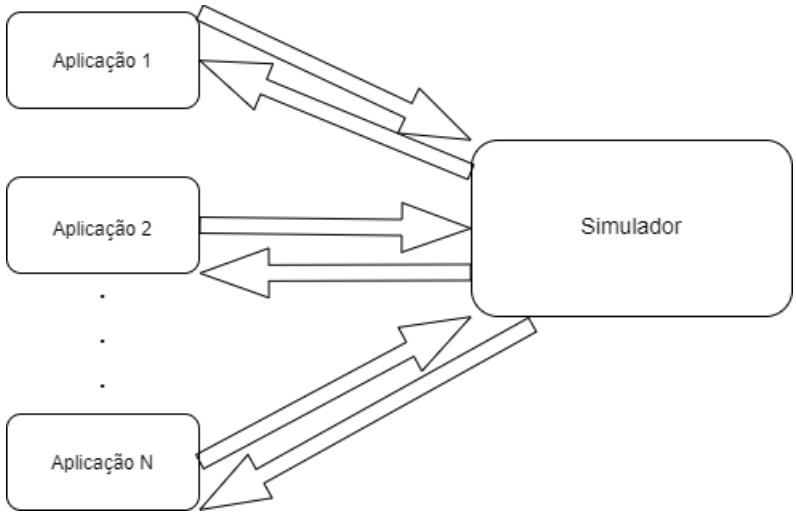

Figura 2 – Estrutura dos módulos do projeto.

A aplicação de interface Web possui como principal objetivo familiarização do usuário com a ferramenta, sendo sua configuração mais clara e de mais alto nível. Neste caso, existe uma API REST

[1] Disponível em <https://github.com/orgs/Opportunistic-Network-Simulator/dashboard>

Editora Epitaya | Rio de Janeiro-RJ | ISBN 978-85-94431-65-3 | 2024

(Representational State Transfer Application Programming Interface) entre a interface e o simulador de forma a receber as requisições HTTP do usuário sob a forma de endpoints, realizar o parse em termos de configuração, executar a simulação e expor as responses ao usuário.

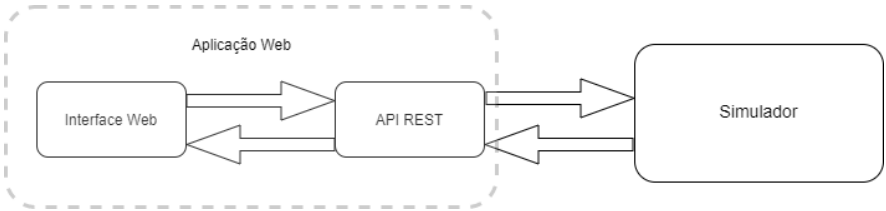

Figura 3 – Estrutura do módulo da aplicação Web do projeto.

Já a aplicação de linha de comando visa ao usuário utilização mais direta do simulador, sendo configurada sob a forma de arquivos com sintaxe pré definida e com resultados sendo gerados em arquivos/relatórios, promovendo maior gerenciamento das simulações por parte do usuário. Detalhes maiores acerca dos parâmetros de utilização do simulador com suas aplicações estão presentes na Seção 4.3.

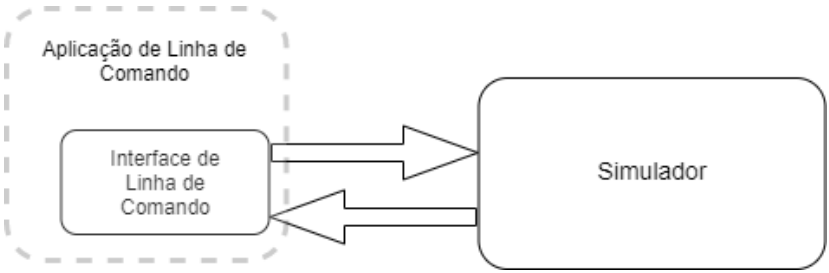

Figura 4 – Estrutura do módulo da aplicação de linha de comando do projeto.

Estrutura do simulador

Dentro do módulo do simulador, ainda existem 3 submódulos responsáveis por diferentes tarefas: geração de trace de encontros de nós, geração de eventos de novas mensagens e a execução da simulação. Esses módulos são independentes entre si, sendo possível, a nível de arquitetura, a utilização de um sem o outro. Isso seria interessante para, por exemplo, gerar um trace de encontros ou rodar uma simulação com um trace já fornecido.

Geração de Trace

O módulo opcional de geração de trace serve para, caso o usuário opte por fazer a simulação baseada em uma tabela de taxas de encontros entre nós, o trace usado pela simulação seria gerado por ele.

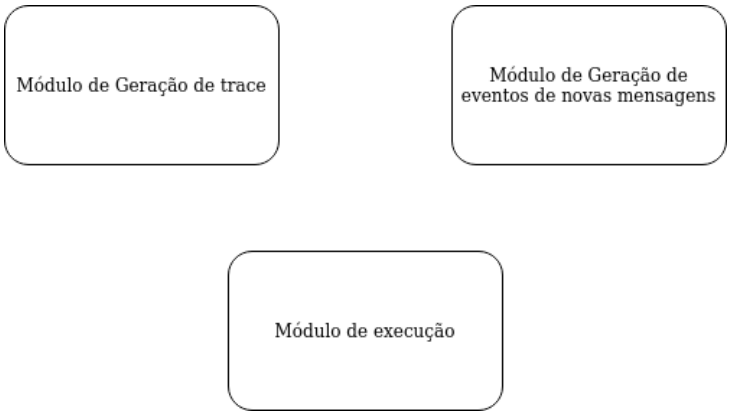

Figura 5 – Estrutura dos submódulos do simulador.

Geração de Eventos de Novas Mensagens

O submódulo de geração de eventos de novas mensagens é responsável por, a partir de restrições determinadas pelo usuário, como quantidade de mensagens geradas ou possibilidades de nó de origem e destino de mensagem, determinar em que momentos serão geradas novas mensagens, e a partir disso gerar os eventos para a fila a ser processada na simulação.

Execução

O módulo de execução é responsável por efetivamente executar a simulação, com base em uma fila de eventos a serem processados. Os possíveis eventos são a geração de uma mensagem, um encontro entre nós e um evento que indica o fim da simulação. Por fim, são gerados relatórios com os resultados.

Elaboração da estrutura de classes do simulador

A classe central do sistema é a Simulation, ela é responsável pela agregação de todas as entidades da simulação, além de sua execução. As classes NodeGroup e MessageGroup são agregados de nós e mensagens, respectivamente. O protocolo de encaminhamento de mensagens é representado pela classe abstrata MessageTransmitionProtocol, que possui um único método abstrato handleMeet. As implementações dessa classe geram os diferentes protocolos que podem ser usados na simulação, como

Editora Epitaya | Rio de Janeiro-RJ | ISBN 978-85-94431-65-3 | 2024

exemplificado pelos EpidemicProtocol e SingleCopyProtocol. A fila de eventos é processada pela classe EventQueue, que processa objetos da classe abstrata Event, cujas classes concretas são MeetEvent, evento em que há o encontro de dois nós, e MessageGenerationEvent, evento responsável pela geração de uma mensagem.

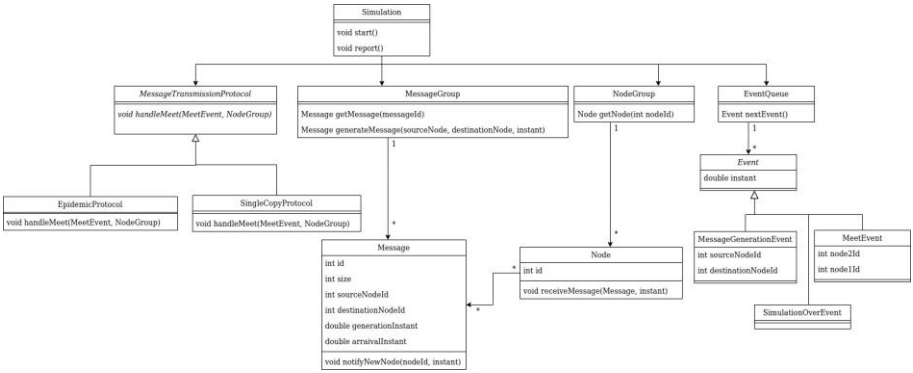

Figura 6 – Diagrama de classes do simulador.

Implementação

A estrutura completa de simulação foi construída e o retorno de sua execução é composto por diversos relatórios contendo o valor médio para o atraso e a taxa de entrega de mensagens, sendo esse o principal resultado, além do trace de encontros gerado e utilizado em cada rodada de simulação, e da descrição temporal dos eventos de encaminhamento e entrega de mensagens também para cada rodada da simulação.

A simulação ocorre para um determinado cenário da rede, o qual deve ser definido pelo usuário a partir dos seguintes parâmetros:

Número de rodadas: Parâmetro que determina quantas vezes será executada a simulação a fim de obter, como resposta final, o valor médio dos resultados de todas as execuções. Isso se dá pelo fato de que o output da simulação não é bem definido, ou seja, o atraso médio de entrega de todas as mensagens do cenário e a porcentagem das mensagens entregues podem variar de uma simulação para outra. Isso ocorre devido à aleatoriedade na geração do trace de encontros. Portanto, é comum que sejam executadas diversas rodadas de forma a obter a convergência dos resultados ao resultado médio. Como as rodadas são independentes, o simulador cria uma thread para cada uma, de forma a paralelizar as execuções, tornando o processo mais rápido.

Estratégia de geração de mensagens: Há 3 opções, sendo elas:

a) **FIXED_NODES:** Um nó especificado pelo usuário gera uma mensagem cujo destino é outro nó também especificado pelo usuário.

b) **RANDOM_NODES:** Um nó aleatório gera uma mensagem cujo destino é outro nó aleatório.
c) **ALL_PAIRS:** Cada nó gera N − 1 mensagens, sendo o destino cada um de todos os outros nós, onde N é o número total de nós.

A estratégia de geração das mensagens, juntamente com o trace de encontros, será utilizada para construir a fila de eventos do sistema a ser utilizado no módulo de execução.

Protocolo de encaminhamento de mensagens: Há opção de utilização dos seguintes protocolos:

a) (p, q) - Epidemic, o qual abrange mais três subcasos:
i. Epidemic
ii. Direct Delivery
iii. Two-Hop

b) Single Copy Epidemic

c) Spray and Wait, além de seu derivado:
i. Binary Spray and Wait

Lista de Pares: Uma lista contendo os ids (indentificadores) dos nós que formam cada par, além de sua taxa de encontro. A taxa de encontro de cada par é definida em s−1 e pode ser fixa ou variável, segundo um grau de variabilidade a ser também informado. A utilização desse grau de variabilidade para fins do andamento da simulação está descrita na subseção 2.3.1. Essa lista de pares com suas taxas de encontro é usada para gerar o trace de encontros da simulação.

Tempo total de simulação: Variável definida em segundos para indicar até qual instante da simulação podem ser gerados encontros entre os pares.

Relatórios gerados

Para a análise dos resultados da simulação, são gerados relatórios para o usuário manusear e interpretar os dados. Quatro tipos diferentes de relatórios são gerados: um de resumo da simulação, o meeting trace utilizado, eventos relacionados à transmissão das mensagens e dados sobre entrega de cada mensagem gerada na simulação.

Como uma simulação pode executar diversas rodadas, os relatórios são gerados para cada uma delas individualmente, além de um relatório geral, como explicado a seguir.

Relatório de resumo da simulação

O relatório de resumo da simulação é um arquivo em formato de texto que apresenta os principais dados da simulação, sendo eles: porcentagem de mensagens entregues e média, variância e desvio padrão do atraso na entrega de mensagens. Esse relatório é gerado para cada rodada, assim

Editora Epitaya | Rio de Janeiro-RJ | ISBN 978-85-94431-65-3 | 2024

como um geral contendo o valor médio das métricas considerando todas as rodadas.

Trace de Encontros

Para cada rodada da simulação executada, é gerado um arquivo em formato CSV contendo o trace de encontros utilizado. Cada linha contém as seguintes informações: instante em que ocorre o encontro e id dos nós que se encontraram.

Relatório de eventos de mensagens

O relatório de eventos de mensagens é um arquivo em formato CSV que contém dados sobre os eventos pertinentes às mensagens que ocorreram durante a simulação. Os eventos são dos tipos: geração de mensagem, encaminhamento de mensagem e chegada de mensagem em nó destino. Cada linha do arquivo contém o id da mensagem, instante em que ocorre o evento e, quando pertinente, os nós envolvidos no evento.

Relatório de entrega de mensagem

O relatório de entrega de mensagens é um arquivo em formato CSV que contém dados referentes à entrega de cada mensagem gerada em uma rodada da simulação. Cada linha do arquivo contém o id da mensagem, o instante de geração, o instante de chegada no nó destino (caso tenha sido entregue), o status (entregue ou não), o id do nó origem e o id do nó destino.

VALIDAÇÃO

Este capítulo objetiva expor a validação de tudo o que foi desenvolvido ao longo de todo o projeto.

Validação da Geração do Trace de Encontros com Distribuição de Probabilidades Exponencial

Com o embasamento teórico presente na Seção 2.3.1 e a utilização dos respectivos conceitos, como mencionado no Capítulo 4, a primeira versão da geração do trace de encontros dos pares, isto é, a definição dos exatos instantes em que ocorrem seus encontros, foi implementada com auxílio da técnica da transformada inversa sobre a distribuição de probabilidades exponencial, cuja variável aleatória é o intervalo de tempo entre encontros e cujo parâmetro λ é a taxa de encontros.

A metodologia aplicada para validação foi gerar diversas variáveis, anotar os valores encontrados para posteriormente gerar o gráfico da função distribuição acumulada e comparar com a original, de forma a garantir que foi uma aproximação válida. Os passos foram especificamente os seguintes:

1. Sortear pseudo randomicamente de maneira uniforme um valor 0 ≤ u ≤ 1 (proba- bilidade). Para isso, foi utilizada a função random() da classe Math do Java.
2. Utilizar o valor u encontrado no passo 1 para resolver a equação

2.5, cuja solução representa o intervalo de tempo dt que transcorrerá até o próximo encontro de determinado par.
3. Armazenar o par (dt, u), o qual representa um ponto da função distribuição acumulada gerado.
4. Repetir os passos de 1 a 3 inclusive até obter 1000 pares ordenados (dt, u).
5. Ordenar os pares em ordem crescente de dt.
6. Salvar os pares ordenados num arquivo de texto. Vale ressaltar que toda a rotina descrita até este passo foi realizada de forma automatizada com a linguagem de programação Java.
7. Capturar os pares ordenados salvos no arquivo de texto e gerar dois gráficos: o resultante dos pares capturados e o da própria função distribuição acumulada exponencial. Este passo foi realizado também de forma automatizada, mas com auxílio das bibliotecas numpy e matplotlib da linguagem de programação Python, devido a sua maior praticidade para esse fim.

O resultado pode ser visto na Figura 7. O eixo das abscissas representa os valores de dt calculados, enquanto o das ordenadas, as probabilidades u geradas. Os gráficos ficaram praticamente sobrepostos.

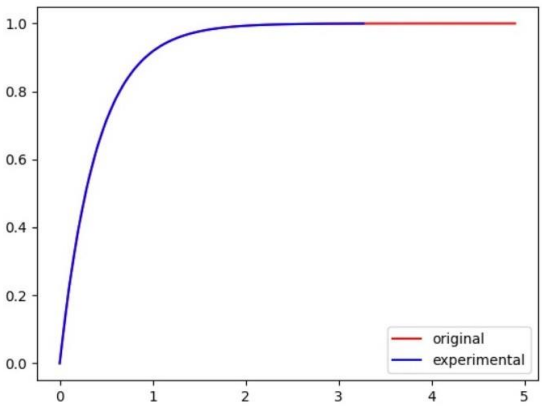

Figura 7 – Gráficos das funções distribuição acumulada experimental (azul) e original (ver- melho) para validação da geração de variáveis aleatórias segundo a distribuição exponencial com parâmetro 2, 5 s−1.

Ampliando a região central do gráfico, como exposto na Figura 8, é possível perceber pequenas divergências entre os gráficos, o que é aceitável e resultante da discretização no processo de geração das variáveis aleatórias dt segundo a técnica da transformada inversa.

Editora Epitaya | Rio de Janeiro-RJ | ISBN 978-85-94431-65-3 | 2024

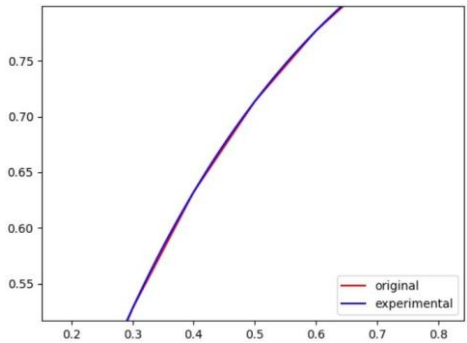

Figura 8 – Gráficos ampliados das funções distribuição acumulada experimental (azul) e original (vermelho) para validação da geração de variáveis aleatórias segundo a distribuição exponencial com parâmetro 2, 5 s−1.

De fato, não havia expectativa de se encontrar valores de dt divergentes para o mesmo valor de u, já que o resultado é provindo de uma simples equação 2.5. O que poderia ocorrer era não serem gerados valores de u bem distribuídos (caso existisse alguma inconsistência em sua geração pseudo randômica), promovendo consequente concentração dos valores de dt num determinado subintervalo, o que não foi observado.

Validação do Protocolo Epidêmico Single-Copy

Para validar o simulador utilizando o protocolo epidêmico, foi feita uma análise baseada em um caso da referência (8). O contexto do caso se trata de 15 nós, divididos igualmente em 3 comunidades. Os pares formados pelos nós pertencentes a uma mesma comunidade possuem taxa de encontro de 2, 5 s−1, enquanto os pares formados pelos nós de diferentes comunidades possuem taxa de encontro de 0. Há duas exceções, que são dois nós da primeira comunidade, considerados “nós viajantes”, os quais possuem uma taxa de encontro de 1, 25 s−1 com os todos os outros nós (inclusive os de sua comunidade). Este cenário de teste pode ser observado na Figura 9.

Na referência (8), foi analisado esse caso utilizando um protocolo epidêmico, com um modelo single copy, em que uma mensagem pode estar presente em apenas um nó. Os resultados obtidos pelo método analítico em (9) foram de 3, 351948 s para o atraso de entrega médio, bem como desvio padrão de 1, 731 s para o atraso entre os pares. Foi então executada a simulação para este cenário no simulador do projeto, sendo obtidos os valores 3.359931 s (erro de 0, 2%) e 1, 736 s, respectivamente. Com isso, pôde-se observar que os resultados do simulador do projeto reproduzem os valores obtidos pelo modelo analítico presente em (8).

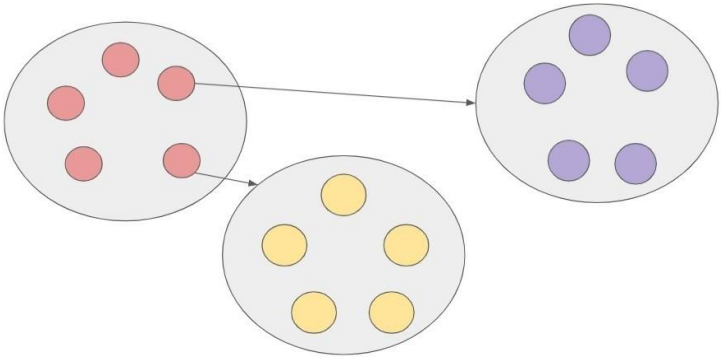

Figura 9 – Cenário de teste das comunidades.

Validação dos Protocolos Direct Delivery, Spray and Wait e Binary Spray and Wait

Para validação dos Protocolos Direct Delivery (subseção 2.2.1.1), Spray and Wait (subseção 2.2.3) e Binary Spray and Wait (subseção 2.2.3.1), foi feita uma análise baseada em cenários descritos pela referência (2).

Cada cenário é composto por 50 nós, segundo o modelo de mobilidade random waypoint (2). Os nós possuem um alcance de transmissão de 50 m, e estão dispostos espacialmente numa área de 2000m x 2000m. Suas velocidades são iguais e fixas, sendo esse valor fixo o que diferencia cada cenário.

Além disso, a geração das mensagens ocorre a cada 3 s de forma aleatória em termos de quem será o nó origem e o nó destino. Cada mensagem possui um tempo limite (TTL) de 1800s, após o qual ela é descartada e considerada como não entregue O estudo também considera que as mensagens têm tamanho podendo variar de 5 a 15kB.

Um importante fator é que esse estudo de caso é baseado em modelo de mobilidade, enquanto o simulador próprio é orientado a taxas de encontro, sendo necessário realizar tal conversão. Para isso, havia algumas possibilidades. Uma delas era utilizar software de terceiros com o objetivo de transformar o modelo de mobilidade descrito em trace de encontros e importá-lo ao simulador. Porém, a forma adotada foi baseada no estudo presente na referência (10). Ele apresenta uma fórmula empírica para transformação dos parâmetros do cenário, para o modelo de mobilidade do tipo random waypoint para taxa de encontro.

No caso de todos os nós apresentarem a mesma velocidade constante, a fórmula se apresenta como:

Editora Epitaya | Rio de Janeiro-RJ | ISBN 978-85-94431-65-3 | 2024

$$\lambda \approx \frac{2\omega r v}{\pi L^2}, \qquad (5.1)$$

onde ω ≈ 1.3683 é uma constante adimensional específica para o modelo de mobilidade random waypoint, r representa o alcance dos nós para a transmissão de uma mensagem, em metros, L é o lado de uma área quadrada na qual os nós estão inseridos espacialmente, em metros, e v representa a velocidade constante dos nós, em m/s.

Para o cenário descrito anteriormente, temos que:

$$\lambda(v) \approx 1.0889.10{-}3v \qquad (5.2)$$

Com isso em mãos, para cada cenário, pode ser calculada a taxa de encontro aproximada de todos os pares, definida pela equação 5.2, a partir da qual o simulador realizará a geração do trace de encontros.

Outra aproximação adotada pela simulação de teste foi na geração das mensagens. A forma como as mensagens são construídas no estudo de caso é bastante específica e isso não é um dos focos do simulador (o qual prioriza ao usuário a possibilidade de configuração dos pares e suas taxas de encontro). Como citado no item 2 da seção 4.3, há 3 formas disponíveis para geração de mensagens. A configuração que mais se aproximou do cenário em questão foi a apresentada no subitem 2c da mesma seção. Além disso, foi configurado o valor de 1800s para o tempo total de simulação, já que as mensagens serão todas geradas no instante t = 0 e, no cenário apresentado, possuem justamente o tempo limite de 1800s. Além disso, nenhuma consideração foi feita quanto ao tamanho das mensagens, já que o estado de desenvolvimento atual do projeto não considera limitação de buffer dos nós.

Nos resultados, o estudo da referência (2) citado analisa diversas métricas. No en- tanto, apenas duas dizem respeito ao escopo do simulador do presente projeto: porcentagem das mensagens entregues e atraso de entrega médio.

Foram simulados os cenários com velocidades de 2m/s, 8m/s, 12m/s e 18m/s, sendo encontrados os seguintes resultados pelo simulador do projeto:

Quadro 1 – Tabela dos resultados da simulação própria para o protocolo Direct Delivery.

Velocidade (m/s)	Taxa de entrega	Atraso médio (s)
2	0.1339	879.9574
8	0.4751	795.8980
12	0.5673	734.9856
18	0.7469	709.0422

No estudo de referência, havia apenas os gráficos com os resultados, não estando disponíveis os valores precisos. Portanto, apresentamos na Figura 10 o gráfico que apresenta os resultados obtidos pelo simulador, de forma a realizar a comparação com o artigo.

Quadro 2 – Tabela dos resultados da simulação própria para o protocolo Spray and Wait.

Velocidade (m/s)	Taxa de entrega	Atraso médio (s)
2	0.3731	971.9955
8	0.9469	627.9281
12	0.9910	403.3884
18	0.9976	319.2654

Quadro 3 – Tabela dos resultados da simulação própria para o protocolo Binary Spray and Wait.

Velocidade (m/s)	Taxa de entrega	Atraso médio (s)
2	0.6085	983.1266
8	0.9693	486.3556
12	0.9763	318.6891
18	0.9902	263.9624

Figura 10 – Taxa de entrega em função da velocidade dos nós no simulador próprio.

O gráfico equivalente na referência adotada apresentou o comportamento como descrito pela Figura 11:

De antemão, as mensagens não entregues são aquelas que excederam o tempo limite de 1800s antes que houvesse encontro do nó

Editora Epitaya | Rio de Janeiro-RJ | ISBN 978-85-94431-65-3 | 2024

origem com o nó destino. Pode-se perceber que, em ambos, a porcentagem das mensagens entregues aumentou com o aumento da velocidade dos nós, o que é natural devido ao fato de uma maior velocidade provocar maior probabilidade de um encontro num mesmo intervalo de tempo.

Ambos os gráficos possuíram comportamento visualmente semelhante, tanto em termos dos valores quanto em característica das curvas apresentadas.

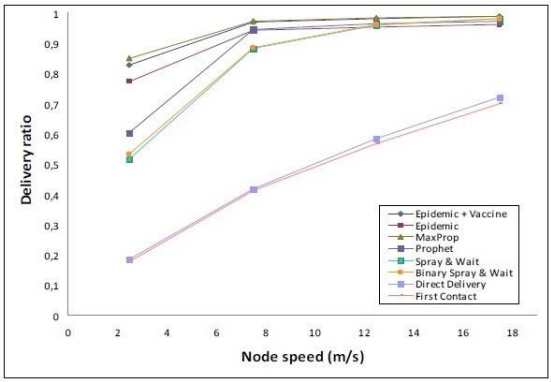

Figura 11 – Taxa de entrega em função da velocidade dos nós no estudo presente na referência (2).

Agora, no que diz respeito ao atraso médio na entrega das mensagens, o simulador próprio apresentou o comportamento como descrito na Figura 12:

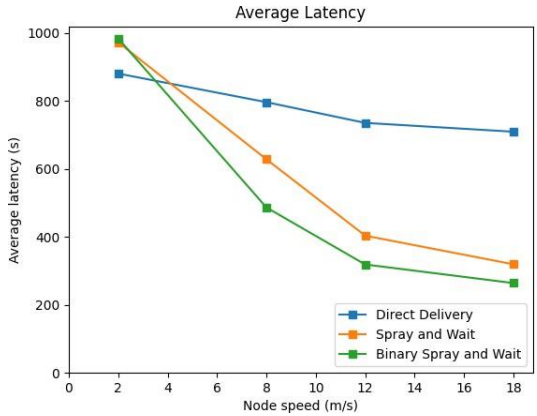

Figura 12 – Atraso médio de entrega das mensagens em função da velocidade dos nós no simulador próprio (2).

Enquanto o gráfico equivalente na referência adotada apresentou o comportamento como descrito pela Figura 13:

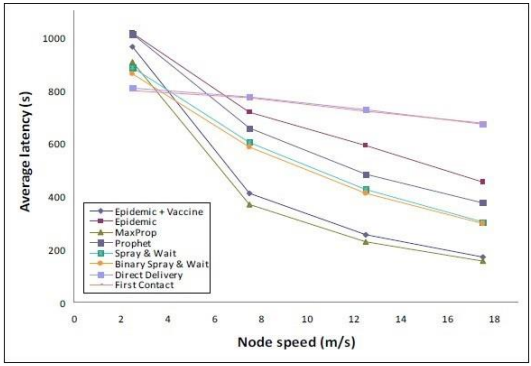

Figura 13 – Atraso médio de entrega das mensagens em função da velocidade dos nós no estudo presente na referência (2).

Ambos os gráficos possuíram comportamento visualmente semelhante, tanto em termos dos valores quanto em característica das curvas apresentadas, à exceção do cenário descrito pela velocidade v = 2m/s. No simulador próprio, esse cenário apresentou atraso de entrega cerca de 10% acima. No entanto, algumas divergências eram esperadas, muito devido às duas aproximações realizadas: o cálculo da taxa de encontro, bem como a geração das mensagens.

UTILIZAÇÃO

Como mencionado na Seção 4.1, o simulador foi planejado de forma a ser indepen- dente de aplicação, tendo sido planejadas e implementadas duas diferentes aplicações: Web e Command Line.

Ambas as aplicações são disponibilizadas ao usuário no repositório do projeto1, com as devidas instruções de utilização. O usuário será instruído a fazer download de um arquivo compactado contendo uma pasta pada cada aplicação. O único recurso necessário no computador do usuário é o Java na versão 8 ou superior.

Editora Epitaya | Rio de Janeiro-RJ | ISBN 978-85-94431-65-3 | 2024

Aplicação Web

No diretório dessa aplicação estão contidos os arquivos html compilados da interface, bem como o arquivo compilado do servidor, de forma a permitir que o usuário consiga executar suas simulações localmente, não necessitando de hospedagem do simulador num servidor remoto.

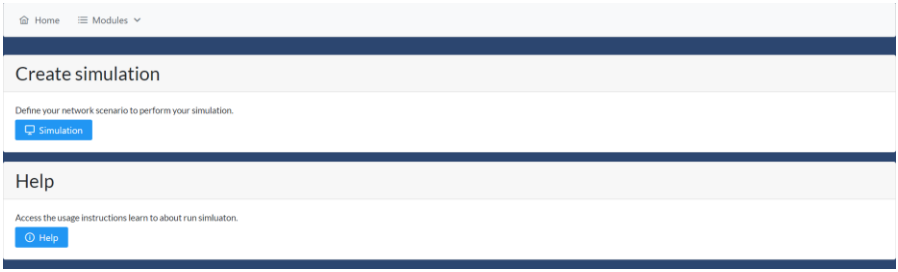

Figura 14 – Imagem da interface web do simulador.

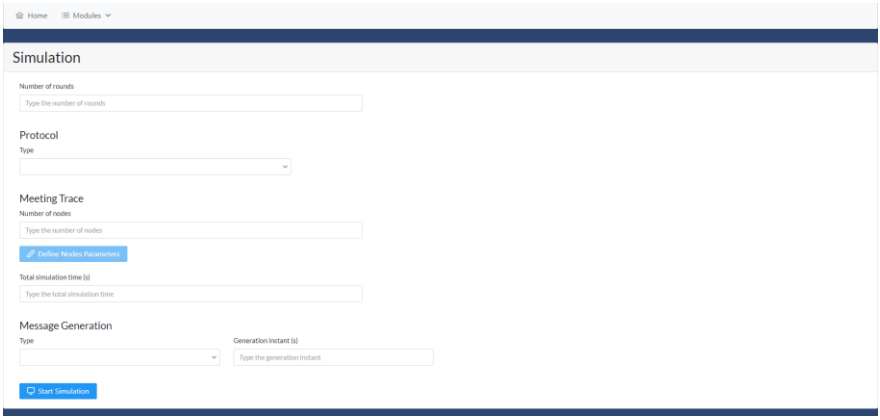

Figura 15 – Imagem da interface web do simulador.

Essa interface tem o objetivo de ser bastante clara e intuitiva ao usuário, principal- mente em sua fase de familiarização com a ferramenta, também o auxiliando quanto às restrições de valores dos parâmetros. O cenário da rede é facilmente configurável, conforme descrito nas instruções de utilização e de acordo com os parâmetros descritos na Seção 4.3.

Ao executar a simulação, o usuário pode acompanhar o progresso da simulação, bem como algumas mensagens de log e enfim obter os relatórios de resultado da simulação na página da interface.

Aplicação de Linha de Comando

Já para essa aplicação, tem-se apenas um arquivo executável de java que é acionado via linha de comando. A configuração é feita através de

um arquivo toml2 informado via argumento. O formato toml visa promover uma escrita simples e de alto nível para o usuário e que possua também facilidade de parse para execução de programas.

No arquivo de configuração devem estar descritos os parâmetros necessários, como mencionados na Seção 4.3. A lista de pares deve ser escrita num arquivo separado em formato json e o nome desse arquivo é apontado pelo arquivo de configuração base.

Essa aplicação possui o objetivo de permitir utilização mais direta e customizada, além de fazer com que o usuário armazene as configurações construídas. Assim, promove maior controle dos testes realizados na ferramenta e proporciona maior organização.

CONCLUSÃO

Concluída a etapa final do projeto, foi possível alcançar o resultado planejado satisfatoriamente: o desenvolvimento de um simulador de redes oportunistas heterogêneas baseado em taxas de encontro. Foram estudados, implementados e validados o gerador de trace de encontros sobre a distribuição de probabilidades exponencial, descrito na Seção 2.3.1, os protocolos de encaminhamento apresentados na Seção 2.2 e as três formas de geração de mensagens mencionadas na Seção 4.3.

Além disso, foram desenvolvidas as interfaces web e de linha de comando, como descritas no Capítulo 6, de forma a permitir utilização simples e clara da ferramenta por parte do usuário, bem como um aproveitamento mais direto e customizado.

Embora o desenvolvimento tenha conseguido almejar importantes funcionalidades, as aplicações possíveis para um simulador com tal finalidade são muito vastas. Com isso, pode-se vislumbrar implementações futuras como opões de geração de trace de encontros baseados em outras distribuições de probabilidade, assim como consideração de tamanho de cada mensagem e de buffer dos nós, além de mais tipos de gerações de mensagens a serem estudas dentre as diversas possibilidades existentes. Todo o código fonte do simulador encontra-se presente no repositório do github 1.

REFERÊNCIAS

1 AD HOC. 20 abr. de 2021. Disponível em: <https://tinyurl.com/3s8jbjyd>.

2 SOCIEVOLE, A.; RANGO, F. D.; COSCARELLA, C. Routing approaches and perfor-mance evaluation in delay tolerant networks. *D.E.I.S. Department, University of Calabria*,p. 3–5, 2011.

3 ANASTASI, G. Ieee 802.11b ad hoc networks: Performance measurements. 2015. 11 mai. de 2021. Disponível em: <https://tinyurl.com/p2r6jthw>.

4 FALL, K. A delay-tolerant network architecture for challenged internets.

Proceedings of 2003 ACM SIGCOMM: Conference on Applications Technologies Architectures and Protocols for Computer Communications, v. 25, p. 27–34, 2003.

5 KHABBAZ; M., J.; ASSI; C., M.; FAWAZ; W., F. *Disruption-Tolerant Networking: A Comprehensive Survey on Recent Developments and Persisting 35 Challenges*. 14. ed.[S.l.]: IEEE Communications Surveys & Tutorials, 2012. 607–640 p.

6 MATSUDA, T.; TAKINE, T. (p,q)-epidemic routing for sparsely populated mobile ad hoc networks. *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS*, v. 26, p. 783–784, 2008.

7 SPYROPOULOS, T. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, p. 3–4, 2005.

8 BOLDRINI, C.; CONTI, M.; PASSARELLA, A. Performance modelling of opportunistic forwarding under heterogeneous mobility. *Computer Communications*, v. 48, p. 56–70,2014.

9 DIAS, G. M. de S. *Modelagem Matemática do Atraso de Entrega de Mensagens em RedesOportunistas com Taxas de Encontro Heterogêneas*. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2018.

10 GROENEVELT, R.; NAIN, P.; KOOLE, G. The message delay in mobile ad hoc networks. *Performance Evaluation*, p. 220–221, 2005.