Análise de Desempenho - Trabalho Prático 2

Nome: Brandon Afonso - RA: 2021.1.08.045

Nome: João Pedro dos Santos Melo - RA:2019.1.08.044

brandon.souza@sou.unifal-mg.edu.br

joao.melo@sou.unifal-mg.edu.br

Professor(a)

Flávio Barbieri Gonzaga

Departamento de Ciências da Computação – Universidade Federal de Alfenas
Avenida Jovino Fernandes de Sales 2600 – CEP 37133840 - Alfenas – MG - Brasil
10/12/2024

RESUMO

Este trabalho apresenta o desenvolvimento e a análise de um simulador baseado em eventos discretos, projetado para avaliar o desempenho de um sistema de comunicação sob diferentes níveis de ocupação de um link. O simulador considera a transmissão de pacotes web, com tamanhos e taxas variáveis, e pacotes de chamadas ao vivo com uma taxa constante, modelando cenários realistas de redes de comunicação.

A simulação é conduzida com parâmetros de ocupação de 60%, 80%, 95% e 99%, e utiliza a Lei de Little como métrica de validação. Durante o processo, são calculadas métricas como ocupação média, tempo médio de espera e erro em relação à fórmula de Little, permitindo avaliar a precisão do simulador e o impacto da saturação no desempenho do sistema. Os resultados são apresentados e discutidos para cada cenário testado, destacando os comportamentos observados e possíveis limitações do modelo.

SUMÁRIO

1. Etapa 1

Desenvolvemos o simulador na linguagem C. Utilizamos Bytes como medidas para o link e os pacotes, e segundos para a medida do tempo dos pacotes passarem pelo link, para os tempos de eventos e para a duração da simulação.

O algoritmo tem dois parâmetros que o usuário da entrada no momento da execução do código, sendo a ocupação que deseja que o simulador se aproxime e a duração da simulação.

Neste trabalho fizemos quatro simulações com a mesma seed e mesmo duração e quatro ocupações diferentes (ocupação de 60%, ocupação 80%, ocupação de 95% e ocupação de 99%).

Parâmetros dos usuários usados nos teste:

	60%	80%	95%	99%
Duração	100000	100000	100000	100000
Ocupação	60	80	95	99

Para gerar tempos de forma aleatórias para teste diferentes usamos a função de srand() para gerar números aleatórios. Nos teste que capturamos os dados usamos a **seed “1”** em todos os teste. Para realizar a mesma simulação precisará colocar a seed no srand() na linha 74 no código do simulador.

Parâmetros fixo da simulação:

A cada segundo chega em média 100 pacotes, sendo que os pacotes podem ter 3 tamanhos diferentes:

50% = 550 Bytes
40% = 40 Bytes
10% = 1500 Bytes

Então fazendo a conta:

50 * 550 + 40 * 40 + 10*1500 = 44100 Bytes/seg

Então já temos a média de Bytes por segundo que o simulador vai ter que transmitir.

Parâmetros que calculamos:

Para conseguirmos o tamanho do link precisamos da ocupação desejada pelo usuário e a média de Bytes por segundo que o link vai tratar, usando regra de 3 conseguimos essa conta:

Média de Bytes Ocupação
Tamanho do Link 100

Para ocupação 60%:

44100 60
Tamanho do Link 100

Tamanho do Link * 60 = 4410000
Tamanho do Link = 4410000 / 60
Tamanho do Link = 73500

outra forma de fazer a conta(como fizemos a conta no código):

Tamanho do link = 44100 / (60 / 100)
Tamanho do link = 44100 / 0,6
Tamanho do Link = 73500

Para ocupação 80%:

Tamanho do link = 44100 / (80 / 100)
Tamanho do link = 44100 / 0,8
Tamanho do Link = 55125

Para ocupação 95%:

Tamanho do link = 44100 / (95 / 100)
Tamanho do link = 44100 / 0,95
Tamanho do Link ≈ 46.421,0526315

Para ocupação 99%:

Tamanho do link = 44100 / (99 / 100)
Tamanho do link = 44100 / 0,99
Tamanho do Link ≈ 44.545,4545454

Funcionamento do Simulador:

O simulador funciona passando o tempo para o próximo evento, sendo que tem 3 eventos na primeira etapa deste trabalho. No início antes de começar o rodar o tempo da simulação já definimos o tempo de dois eventos, a chegada do primeiro pacote e a primeira coleta de dados. O primeiro pacote a chegar recebe um tempo aleatório baseado no parâmetro de tempo de chegada dos pacotes (100). A coleta de dados recebe o tempo de 100 segundos.

No início do tempo da simulação depois de estabelecer todos os parâmetros iniciais, a primeira coisa a que o simulador faz é encontrar o menor tempo dos próximos eventos registrados, no início vamos ter um tempo de pacote para chegar(evento de chegada) e uma evento de coleta(Evento de coleta de dados).

Como os eventos Funcionam:

Evento de Chegada:

Este evento é quando um pacote chega. Aqui já calculamos o tempo de chegada do próximo pacote. Se não tiver nenhum pacote na fila esperando para sair, já sorteamos o tamanho do pacote e calculamos o tempo de saída desse pacote que chegou e calculamos a ocupação do link. Caso já tenha pacotes na fila, então só adicionamos ele à fila e calculamos o E[N], que representa o número médio de pacotes no sistema, e E[W], que indica o tempo médio que um pacote permanece no sistema, desde sua chegada até sua saída.

Evento de Chegada:

Este evento é quando chega o tempo de saída de um pacote. Primeiro removemos um pacote da fila. Depois, se ainda houver pacotes na fila, sorteamos o tamanho do pacote e calculamos o tempo de saída dele. Se não houver fila, colocamos o tempo de saída como o máximo da variável para que não entre no evento sem necessidade. Depois calculamos o E[N] e E[W].

Evento de Coleta:

Este evento ocorre a cada 100 segundos, nele calculamos o E[N], E[W], lambda, erro little e ocupação até esse momento e depois salvamos em um arquivo. Por último atualizamos o tempo do evento de coleta para o tempo atual mais 100 segundos.

No final, depois que o tempo da simulação acabou, calculamos os E[N], E[W], lambda, ocupação e erro little e salvamos o arquivo dos resultados.

1.2. Gráfico

Erro de Little:

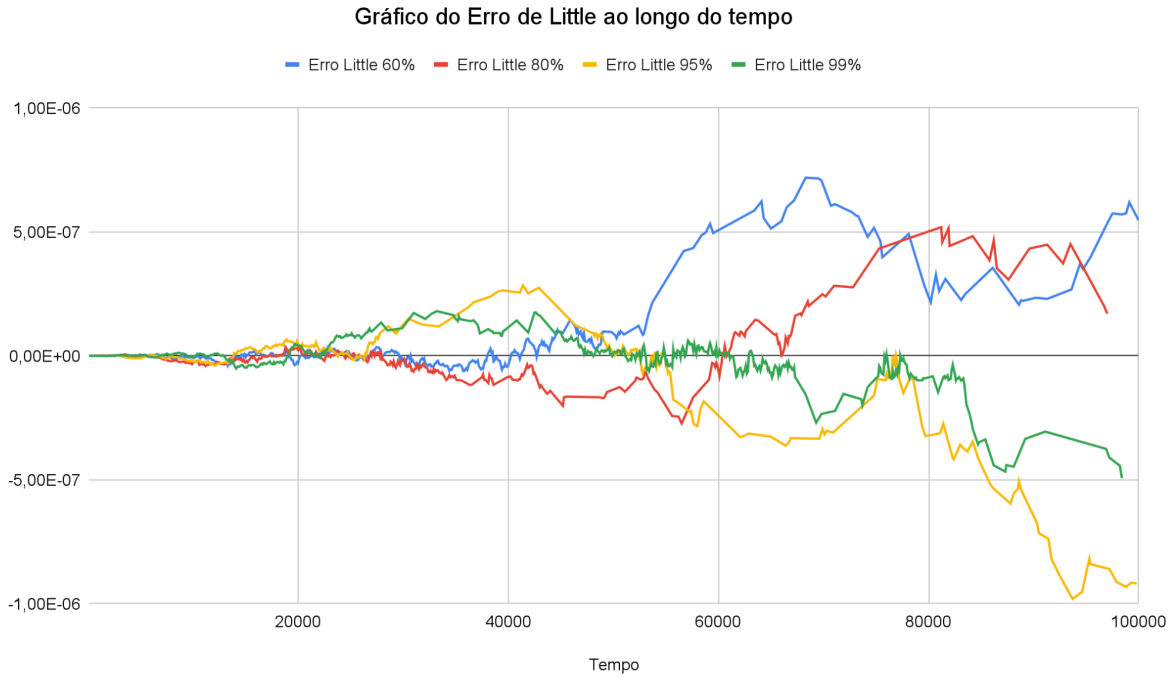

Tabela dos valor finais do Erro de Little:

60%	80%	95%	99%
5,45E-07	2,71523316275E-07	-9,54866305136E-07	-3,95935771280E-07

Ocupação:

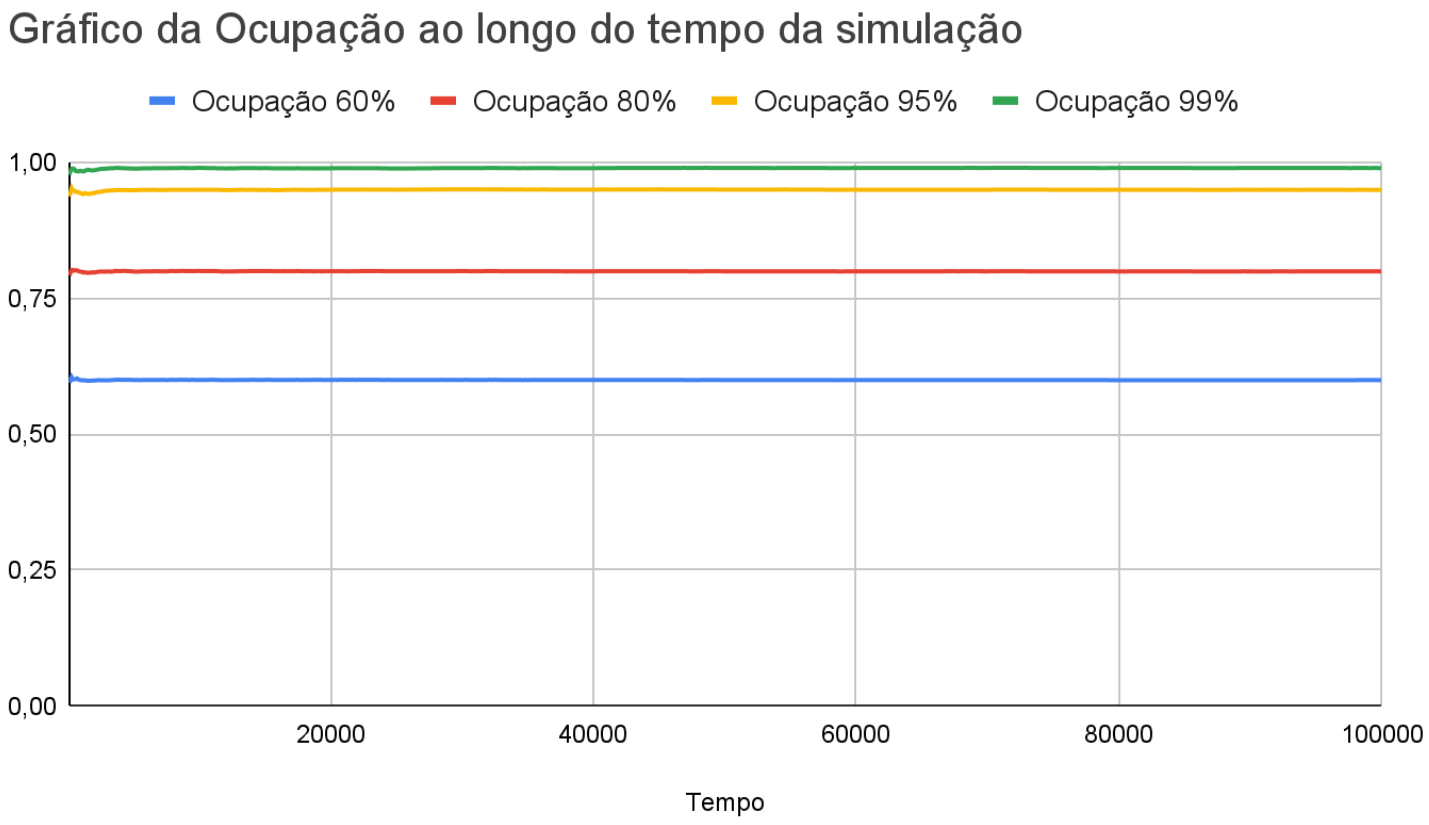

Tabela dos valores finais da ocupação:

60%	80%	95%	99%
0,599719	0,799871	0,949924	0,990112

E[N]:

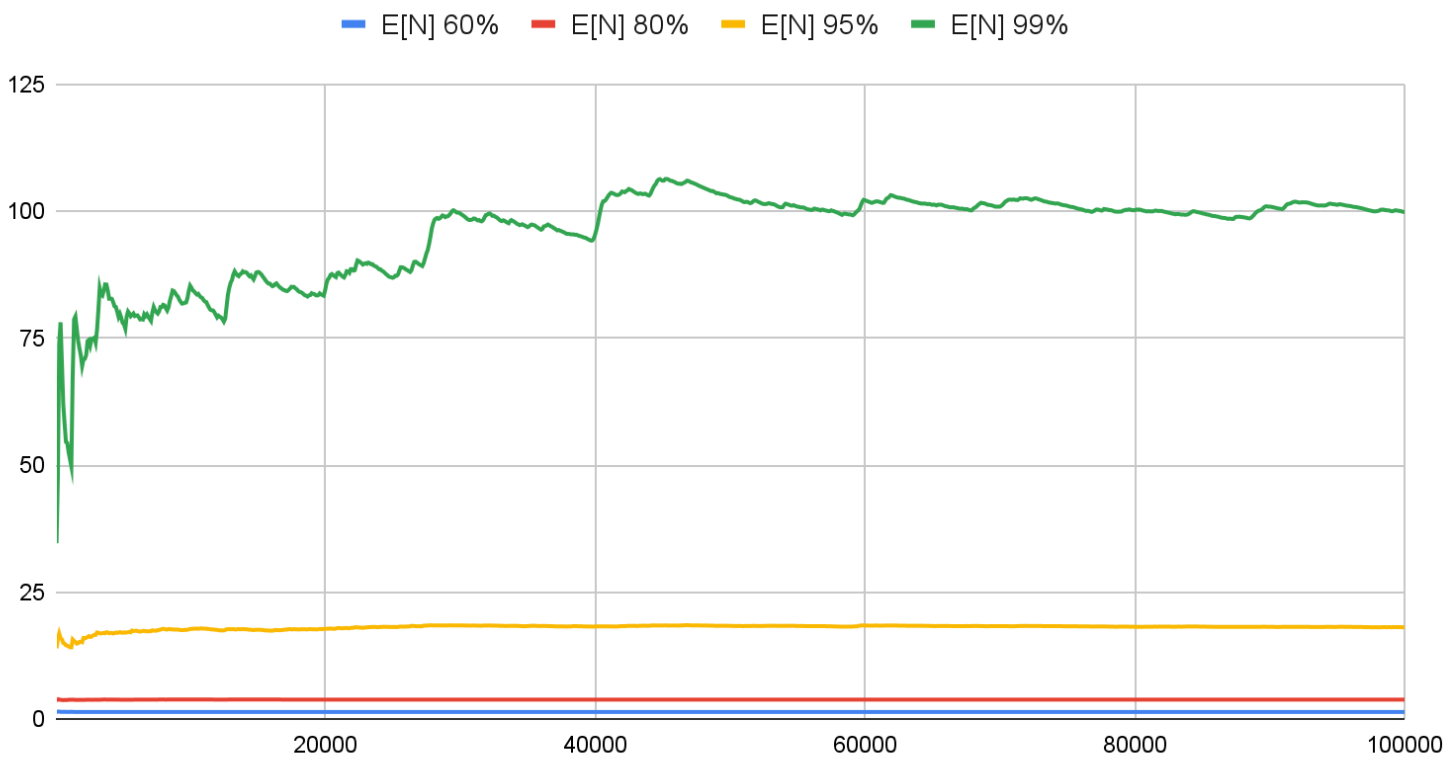

Tabela dos valores finais do E[N]:

60%	80%	95%	99%
1,471974	3,888592	18,152817	99,866141

E[W]:

Gráfico do E[W] ao longo do tempo da simulação

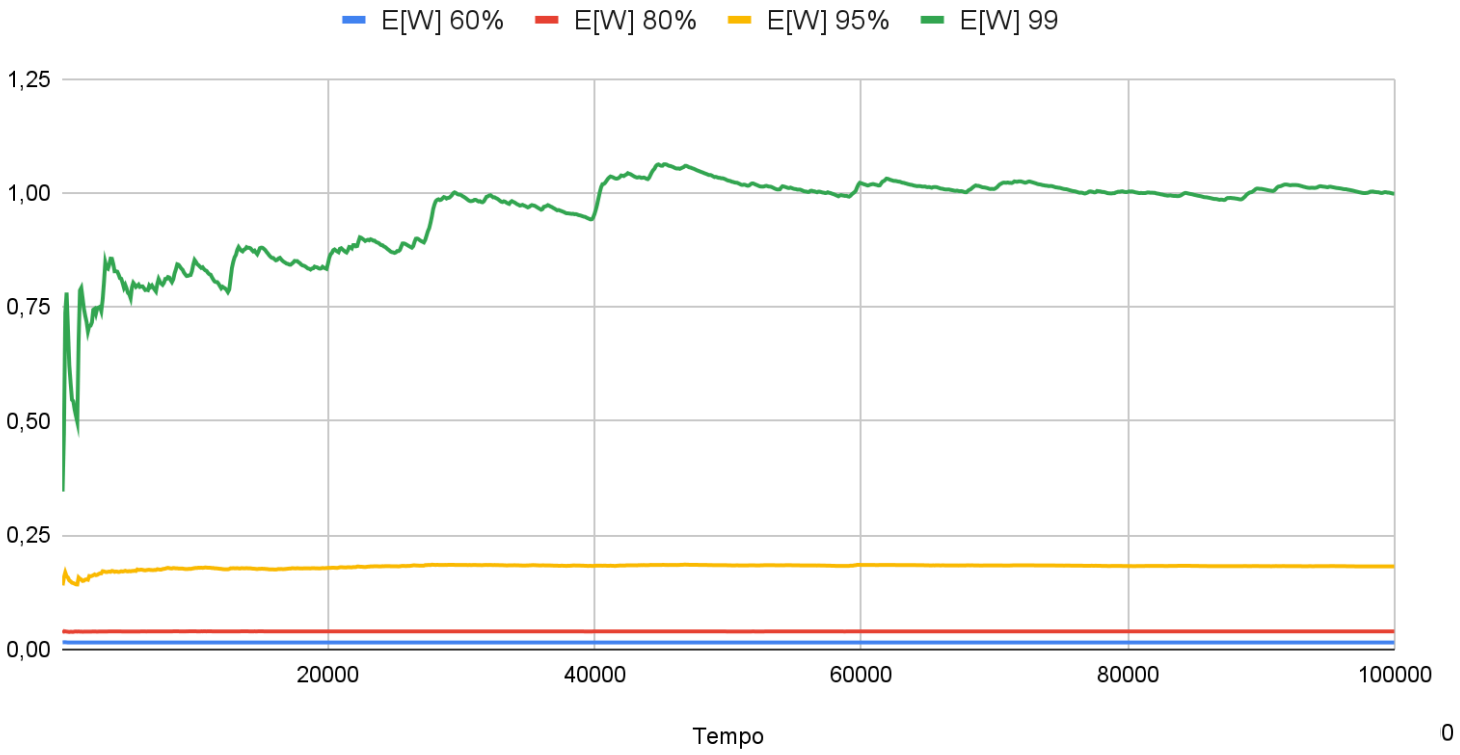

Tabela dos valores finais do E[W]:

60%	80%	95%	99%
0,014724	0,038887	0,181509	0,998674

2. Etapa 2

Desenvolvemos o simulador na linguagem C. Utilizamos Bytes como medidas para o link e os pacotes, e segundos para a medida do tempo dos pacotes passarem pelo link, para os tempos de eventos e para a duração da simulação. Nesta etapa além dos pacotes web que já transmitimos no primeiro, também vamos ter que manter chamadas ao vivo que gera uma quantidade de pacotes constantes enquanto elas estão ativas.

O algoritmo tem dois parâmetros que o usuário da entrada no momento da execução do código, sendo a ocupação que deseja que o simulador se aproxime e a duração da simulação.

Neste trabalho fizemos quatro simulações com a mesma seed e mesmo duração e quatro ocupações diferentes (ocupação de 60%, ocupação 80%, ocupação de 95% e ocupação de 99%).

Parâmetros dos usuários usados nos teste:

	60%	80%	95%	99%
Duração	100000	100000	100000	100000
Ocupação	60	80	95	99

Para gerar tempos de forma aleatórias para teste diferentes usamos a função de srand() para gerar números aleatórios. Nos teste que capturamos os dados usamos a **seed "1"** em todos os teste. Para realizar a mesma simulação precisará colocar a seed no srand() na linha 182 no código do simulador.

Parâmetros fixo da simulação:

Continuamos com os pacotes web da primeira etapa, então, a cada segundo chega em média 100 pacotes, sendo que os pacotes podem ter 3 tamanhos diferentes:

50% = 550 Bytes
40% = 40 Bytes
10% = 1500 Bytes

Então fazendo a conta:

50 * 550 + 40 * 40 + 10*1500 = 44100 Bytes/seg

Esse são os parâmetros das chamada:

- Intervalo entre o início de novas chamadas (exponencial com média ≤ 30 segundos)
- Tempo de duração das chamadas (exponencial com média ≥ 60 segundos)
- Taxa de geração de pacotes: 64 Kbps
- Intervalo entre pacotes: 20 ms CBR (Constant Bit Rate)

Então como em média as chamadas começam a cada 30 segundos e elas duram em média 60 segundos, a quantidade média de chamadas que vão estar ativas ao mesmo tempo é de duas, 60/30 = 2.

20ms = 0,02s(segundos)

64 Kbps = 64000 Bps (quantidade de Bytes por segundo por chamada)

1s/0,02s = 50 (quantidade de pacotes por segundo por chamada)

64000/50 = 1280 (tamanho do pacote da chamada em Bytes)

Média de pacotes Bytes/seg = 64000 * 2 + 44100 = 172100

Parâmetros calculados:

Assim como na primeira etapa 1, calculamos o tamanho do link usando a mesma regra de 3:

Para ocupação 60%:

Tamanho do link = 172100 / (60 / 100)
Tamanho do link = 172100 / 0,6
Tamanho do Link ≈ 286.833,3333

Para ocupação 80%:

Tamanho do link = 172100 / (80 / 100)
Tamanho do link = 172100 / 0,8
Tamanho do Link = 215.125

Para ocupação 95%:

Tamanho do link = 172100 / (95 / 100)
Tamanho do link = 172100 / 0,95
Tamanho do Link ≈ 181.157,8947

Para ocupação 99%:

Tamanho do link = 172100 / (99 / 100)
Tamanho do link = 172100 / 0,99
Tamanho do Link ≈ 173.838,3838

Funcionamento do Simulador:

O simulador funciona passando o tempo para o próximo evento, sendo que tem 3 eventos da primeira etapa e mais 2 eventos da parte das chamadas. No início antes de começar o rodar o tempo da simulação já definimos o tempo de três eventos, a chegada do primeiro pacote, a primeira coleta de dados e a chegada da primeira chamada. O primeiro pacote a chegar recebe um tempo aleatório baseado no parâmetro de tempo de chegada dos pacotes (100). A coleta de dados recebe o tempo de 100 segundos. A primeira chamada recebe um valor aleatório baseado no parâmetro de chegada de chamada (1/30).

No início do tempo da simulação depois de estabelecer todos os parâmetros iniciais, a primeira coisa que o simulador faz é encontrar o menor tempo dos próximos eventos registrados.

Como os eventos Funcionam:

Evento de Chegada:

O evento de chegada de pacotes agora está dividido em dois, por que precisamos calcular o E[N] e E[W] para todos os pacotes e separado só para os pacotes das chamadas. O primeiro evento de chegada é só para os pacotes web, nele é sorteado o tamanho do pacote é calculado o tempo de saída dele se não houver fila, se houver fila, o pacote vai para o final da fila sem tempo de saída. Nesse evento é calculado o E[N] e E[W] de todos os pacotes.

No evento de chegada de pacote das chamadas o tamanho do pacote é fixo(1280 Bytes), funciona como os pacotes web nas filas. Neste evento calculamos o E[N] e E[W] para todos os pacotes e separadamente para os pacotes das chamadas.

Evento de Saída:

Este evento é quando chega o tempo de saída de um pacote. Antes de removermos o pacote da fila, verificamos seu tamanho, para saber se é um pacote de chamada, se for, calculamos o E[N] e E[W] das chamadas. Depois removemos o primeiro pacote da fila. Depois, se ainda houver pacotes na fila, calculamos o tempo de saída dele. Se não houver fila, colocamos o tempo de saída como o máximo da

variável para que não entre no evento sem necessidade. Depois calculamos o E[N] e E[W] de todos os pacotes.

Evento de Coleta:

Este evento ocorre a cada 100 segundos, nele calculamos o E[N], E[W], lambda, erro little e ocupação até esse momento e depois salvamos em um arquivo. Por último atualizamos o tempo do evento de coleta para o tempo atual mais 100 segundos.

Evento de Chegada de Chamada:

Neste evento criamos a chama definindo o seu tempo de chegada, saída e a chegada de seu próximo pacote. Também calculamos o tempo de chegada da próxima chamada. Adicionamos a chamada no final da fila de chamadas, por que assim sempre temos a ordem de chegada dos pacotes das chamadas. Por último, pegamos qual é a próxima chamada a sair da fila para já ter seu tempo salvo.

Evento de Saída de Chamada:

Por último dos eventos, a saída da chamada, já temos ela salva, então é removê-la da lista de chamadas.

No final, depois que o tempo da simulação acabou, calculamos os E[N], E[W], lambda, ocupação e erro little e salvamos o arquivo dos resultados.

2.3 Gráficos

Gráficos de todos os pacotes:

Ocupação:

Grafico da Ocupação ao longo do tempo

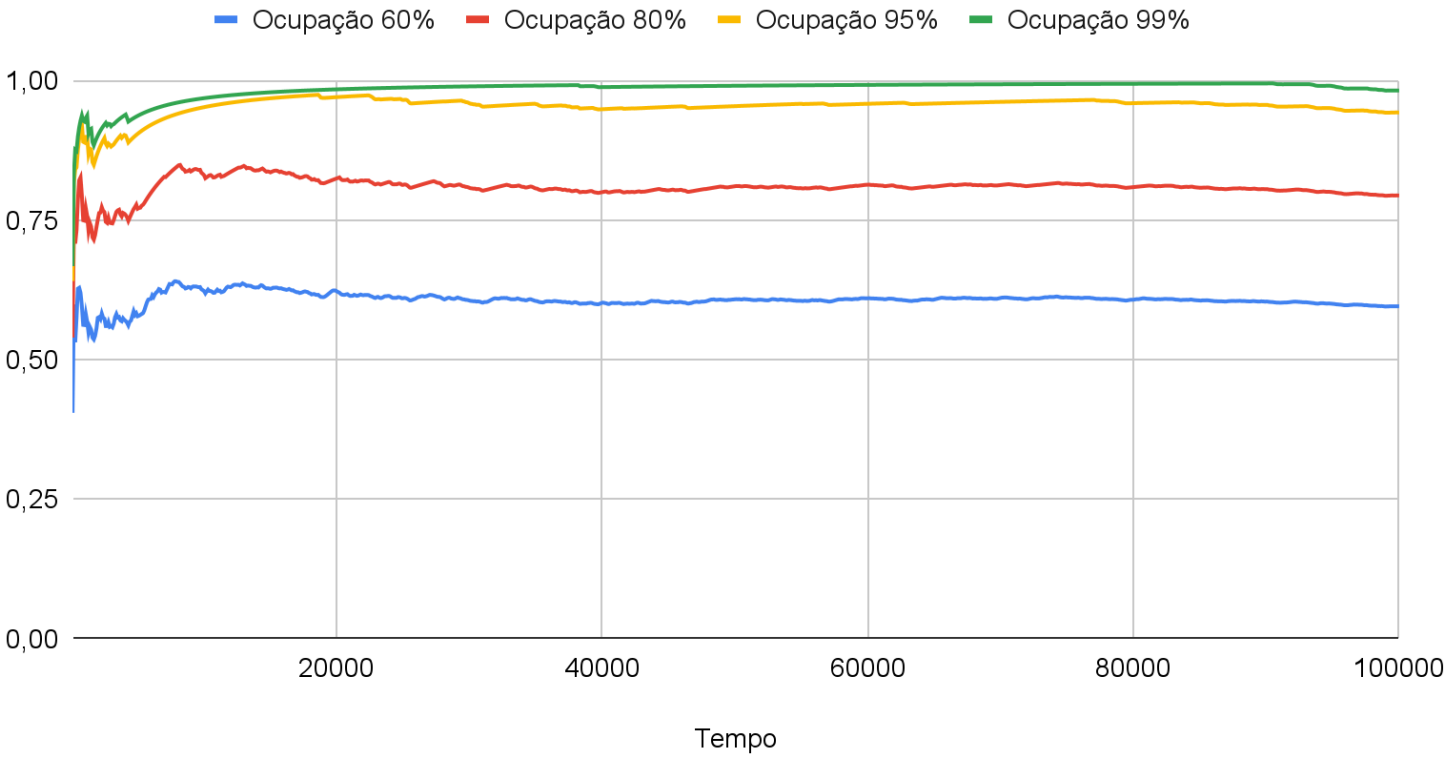

Tabela dos valores finais da ocupação:

60%	80%	95%	99%
0,595475	0,793966	0,942835	0,982108

E[N]:

Gráfico do E[N] ao longo do tempo

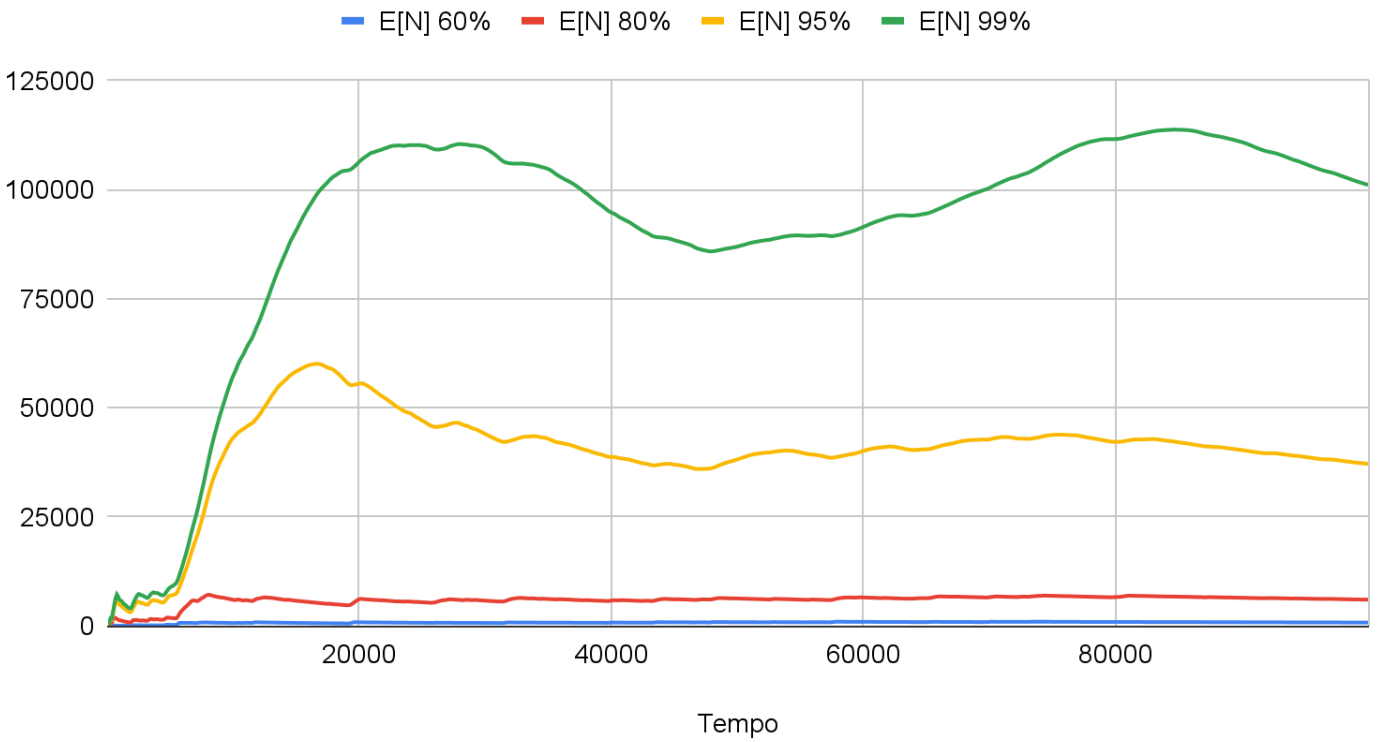

Tabela dos valores finais do E[N]:

60%	80%	95%	99%
743,304514	5986,987344	37115,95803	101077,7703

E[W]:

Gráfico do E[W] ao longo do tempo

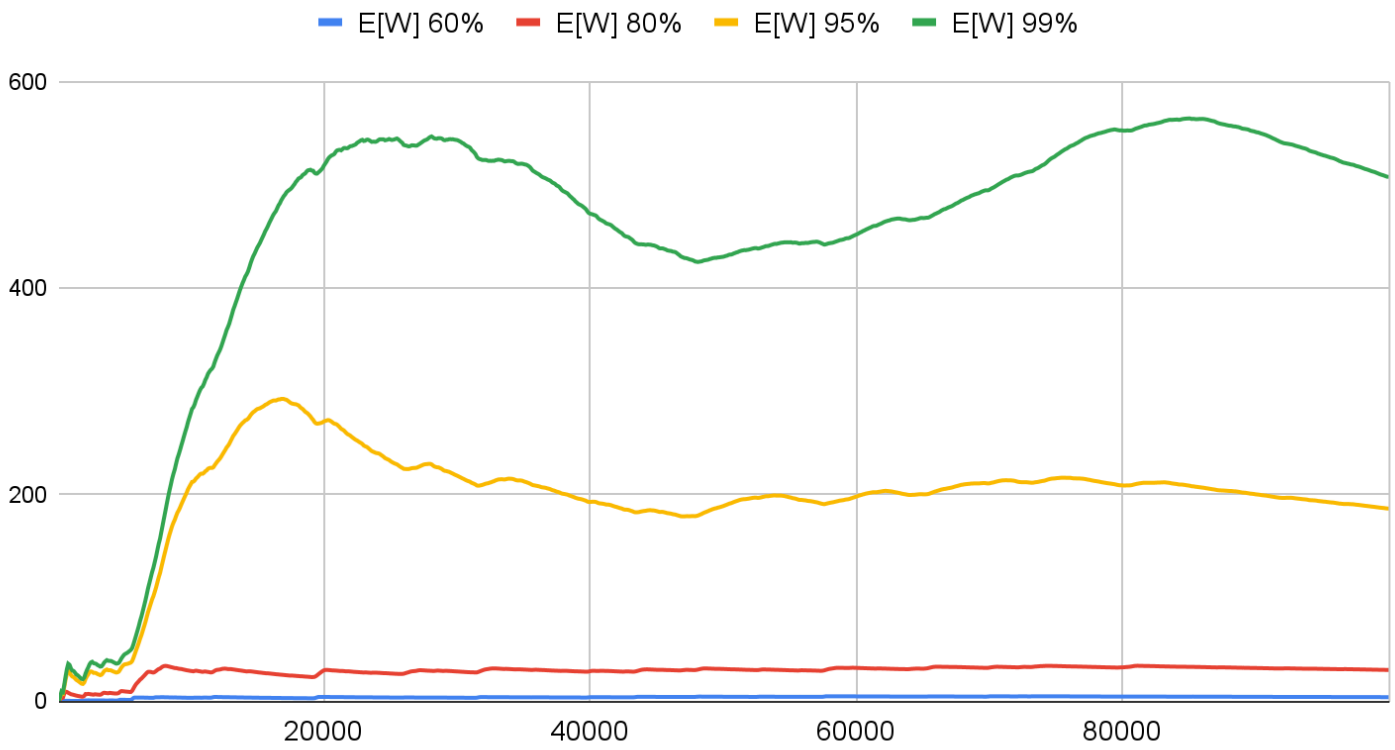

Tabela dos valores finais do E[W]:

60%	80%	95%	99%
3,734785	30,08203	186,491689	507,872221

Erro Little:

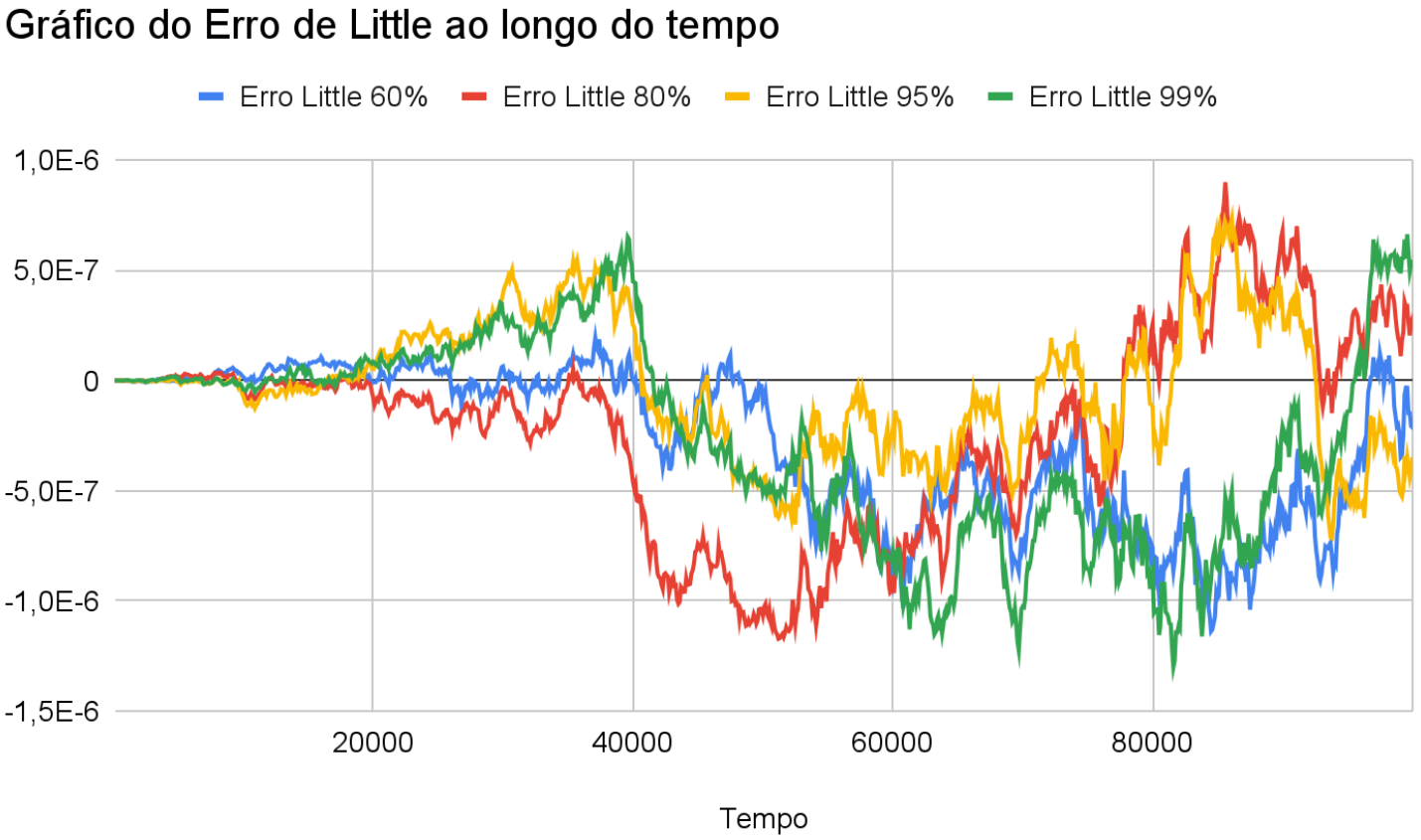

Tabela dos valores finais do Erro Little:

60%	80%	95%	99%
-0,0000002128423857	0,0000002818802711	-0,0000003483526534	0,0000005476848486

Gráficos dos pacotes das chamadas:

E[N]:

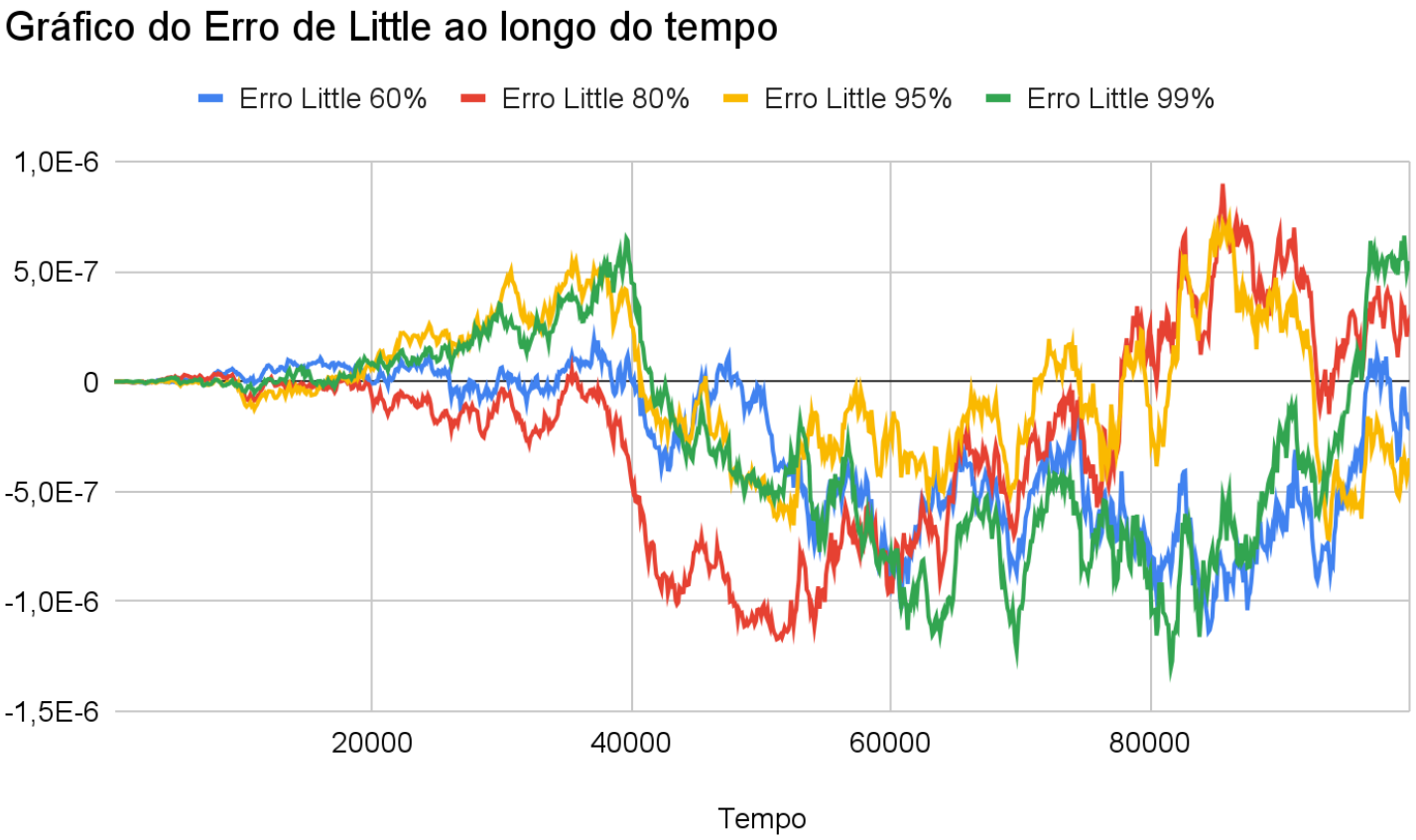

Tabela dos valores finais do E[N] das chamadas:

60%	80%	95%	99%
486,632289	3423,098438	19189,36065	50871,54657

E[W]:

Gráfico do E[N] das chamadas os longo do tempo

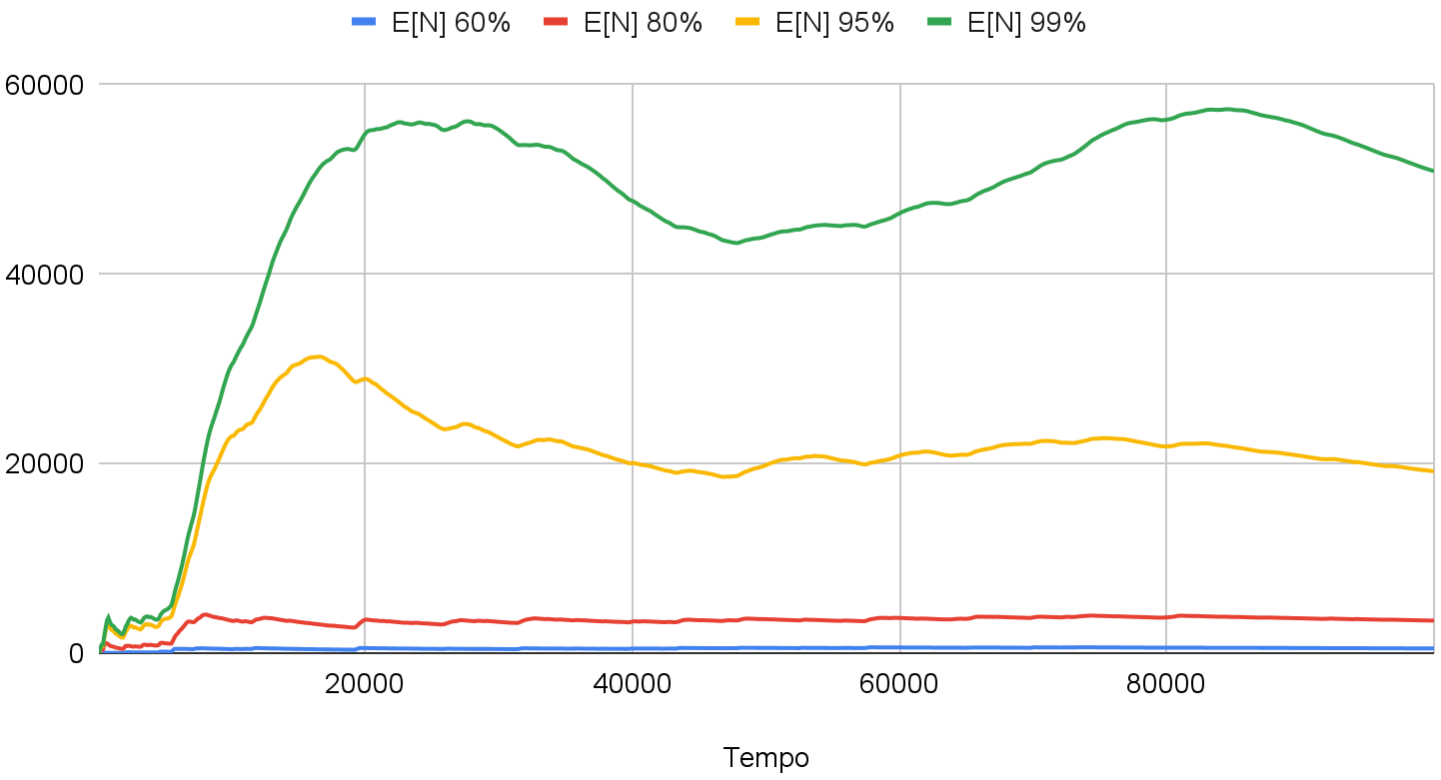

Tabela dos valores finais do E[W] das chamadas:

60%	80%	95%	99%
4,915844	34,57933	193,846377	513,892316

Erro Little:

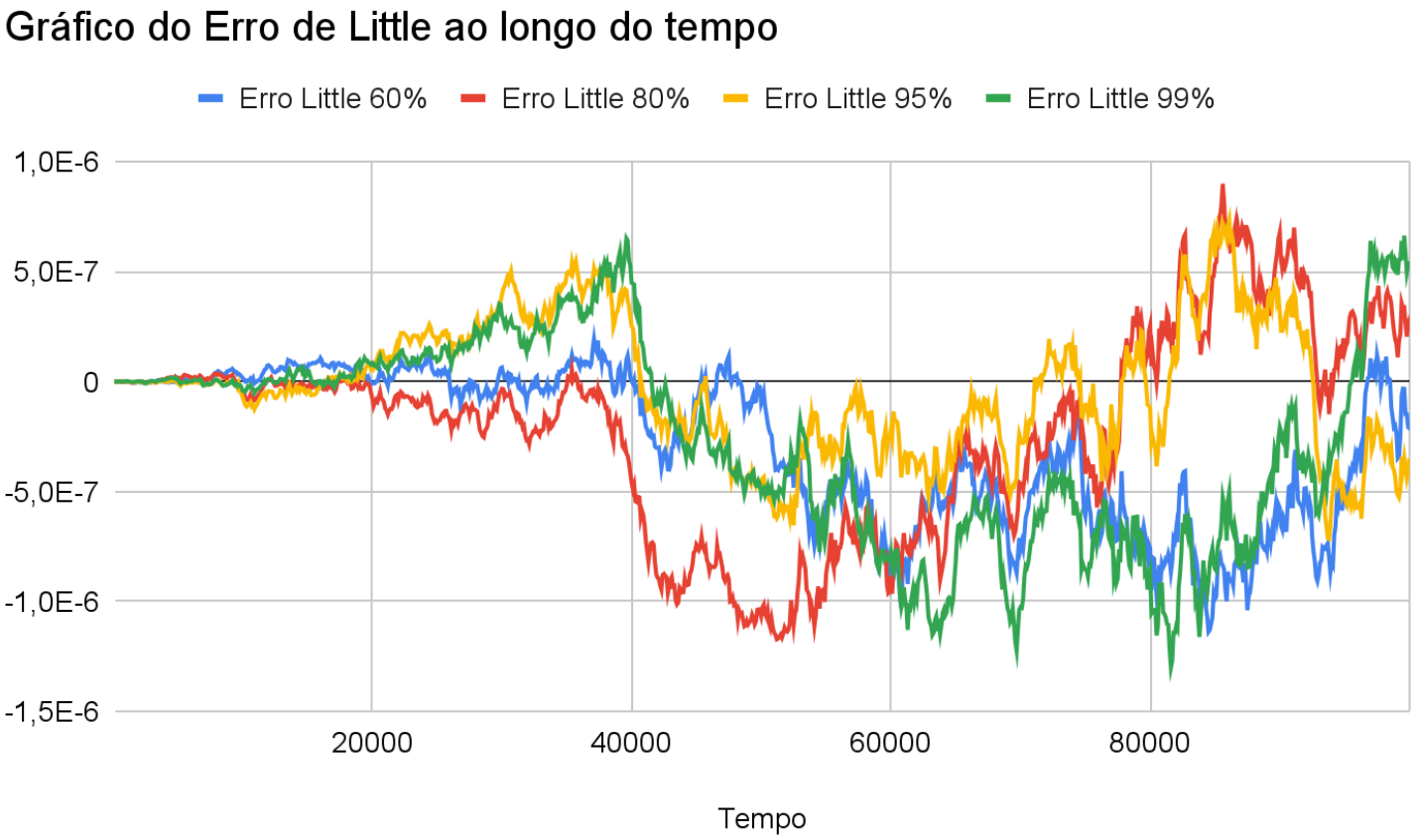

Tabela dos valores finais do Erro Little das chamadas:

60%	80%	95%	99%
0,0000006630339797	0,000000754235165	0,00000009808917767	0,0000007086597904

Análise dos Gráficos:

Os gráficos de todos os pacotes e só dos pacotes das chamadas ficaram bem similares em questão de curvatura, mostrando um comportamento parecido das duas partes. O E[N] e E[W] ficaram maiores que o esperado, por causa do sistema de FIFO que a fila dos pacotes utilizam, provavelmente uma fila com prioridade seria melhor para esse caso, assim os pacotes ficariam na fila por muito tempo.

3 Cálculo Do Erro De Little

O cálculo do erro de Little nas duas etapas da simulação (tráfego de navegação web e tráfego de chamadas em tempo real) segue uma abordagem semelhante, mas é importante considerar cada tipo de tráfego separadamente, pois eles têm características distintas que influenciam o desempenho do sistema. A seguir, explico como o erro de Little é calculado e monitorado em cada uma das etapas:

3.1 Métrica do Erro de Little

Etapa 1: Simulação de Navegação Web

- **Objetivo**: Verificar a conformidade com a Lei de Little sob um tráfego de pacotes de navegação web.
- **Características**:
    1. A chegada dos pacotes segue uma distribuição exponencial.
    2. Pacotes de diferentes tamanhos (550 Bytes, 40 Bytes, 1500 Bytes).
- **Cálculos**:
    1. **Taxa média de chegada de pacotes (λ)**: Calculada dividindo o número total de pacotes pelo tempo total da simulação.
    2. **Número médio de pacotes no sistema (E[N])**: O número de pacotes no sistema é acumulado durante a simulação. Isso inclui pacotes em fila e pacotes em transmissão.
    3. **Tempo médio de espera dos pacotes (E[W])**: Calculado considerando o tempo que cada pacote permanece no sistema (desde sua chegada até a saída).
    4. **Erro de Little**: O erro de Little é dado pela diferença entre o valor observado de E[N] e o valor esperado (λ * E[W]). Esse erro é monitorado ao longo da simulação, periodicamente a cada 100 segundos.

Etapa 2: Inclusão de Chamadas em Tempo Real

- **Objetivo**: Analisar o impacto do tráfego de chamadas em tempo real sobre a simulação e o erro de Little.
- **Características**:
    1. Pacotes de chamadas em tempo real são gerados com uma taxa fixa (64 Kbps) e um intervalo constante de pacotes (20 ms).
    2. As chamadas em tempo real têm requisitos mais rigorosos de tempo de espera.
- **Cálculos**:

1. **Taxa média de chegada de pacotes (λ)**: Para os pacotes de chamadas em tempo real, a taxa é calculada da mesma forma que na etapa anterior, mas separadamente para os dois tipos de tráfego.
2. **Número médio de pacotes no sistema (E[N])**: Agora, o número médio de pacotes é calculado separadamente para o tráfego de navegação web e para as chamadas em tempo real. Isso ajuda a entender o impacto das chamadas em tempo real no número de pacotes no sistema.
3. **Tempo médio de espera dos pacotes (E[W])**: Assim como o cálculo de E[N], o tempo médio de espera é calculado separadamente para os pacotes de navegação web e para os pacotes de chamadas em tempo real.
4. **Erro de Little**: O erro de Little é calculado para cada tipo de tráfego separadamente (tráfego web e chamadas em tempo real), bem como um erro global para o sistema completo. O erro global é a média ponderada dos erros de Little de ambos os tipos de tráfego. Esse erro é monitorado ao longo da simulação e pode revelar distorções causadas pela interação entre os diferentes tipos de tráfego.

Análise do Erro de Little:

- **Objetivo**: Verificar a conformidade com a Lei de Little e observar o comportamento do sistema sob diferentes cenários de tráfego.
- **Monitoramento**: O erro de Little é calculado periodicamente e monitorado a cada 100 segundos da simulação para verificar se o sistema segue a relação teórica proposta pela Lei de Little. O comportamento esperado é que, em sistemas estáveis, o erro seja pequeno, especialmente quando o tráfego é moderado e não há congestionamento excessivo.

4. Conclusão

Neste trabalho foi desenvolvido e simulamos um sistema de comunicação baseado em eventos discretos, considerando tanto a transmissão de pacotes web quanto a geração constante de pacotes provenientes de chamadas ao vivo. A simulação foi realizada com diferentes níveis de ocupação do link, variando de 60% a 99%, para avaliar o impacto da saturação no desempenho do sistema de comunicação.

Através dos resultados obtidos, foi possível observar que, conforme a ocupação do link aumenta, as métricas de desempenho, como o tempo médio de espera e a ocupação do link, se deterioram, especialmente em cenários de alta ocupação. A implementação da Lei de Little foi válida, com o erro de Little sendo calculado e comparado ao comportamento esperado do sistema, o que validou a precisão da simulação para os cenários testados.

Além disso, o trabalho forneceu informações sobre o comportamento do sistema sob diferentes condições, o que é crucial para otimizar o uso de redes de comunicação em contextos com tráfego misto, como pacotes web e chamadas ao vivo.

Por fim, a simulação mostrou ser uma ferramenta valiosa para entender o comportamento do sistema em cenários de alta carga e para estudar as consequências da saturação da rede, fornecendo uma base sólida para futuras otimizações e ajustes de rede.