Universidade

Federal de Alfenas

Relatório Análise de Desempenho

Guilherme Vieira R.A.: 2014.1.08.012

Vinícius Nogueira R.A.: 2013.1.08.038

Relatório

1. Fórmula

Baseado nos dados iniciais e nos códigos desenvolvidos em aulas práticas, utilizamos uma forma de colocar a ocupação desejada para o sistema todo, utilizando o intervalo de pacote e uma média de tamanho de pacotes de 441, como feito em aula. A função foi a seguinte:

```
double intervaloPacote(double link, double ocupacao, double tamMedioPacote) {
    double result = (link * ocupacao) / tamMedioPacote;
    return (1.0 / result);
}
```

//Main
intervalo = intervaloPacote(1250000, 0.8, 441); //Retorna 0.000441
intervalo = 1/intervalo;

Assim, conseguimos mandar o tamanho do link em Mbps e a ocupação por 0.4 a 0.99. Além disso, conseguimos encontrar uma fórmula que encontraria a ocupação exata dos pacotes web, dada pelo **((tamanho médio dos pacotes*intervalo de pacotes web)/link):**

```
public double ocupacao_pct_Web(double tam, double link, double intervalo) {
    return tam*intervalo / link;
}
```

na chamada ficaria:
double ocweb = ocupacao_pct_Web(441, 1250000, intervalo);
Convertendo o link para mbps e o tamanho médio de pacotes de 441 obtivemos exatamente a ocupação web desejada.
Para a ocupação Cbr calculamos da seguinte forma:

```
public double ocupacao_pct_Cbr(double tam, double link, double intervalo, double duracao) {
    return (tam*intervalo / link)*(duracao/intervalo);
}
```

na chamada ficaria:
double ocCbr = ocupacao_pct_Cbr(1200, 1250000, chegada_proximo_pct_cbr, duracao_conexao);

e a ocupação total seria **OcupaçãoTotal = ocweb + ocCbr;**
Entretanto há uma margem de erro de 0.3 da real simulação.

2. Implementação

Para o nosso problema, passamos o código para Java, onde criamos uma classe 'Pacote', com os atributos 'tempo' (tempo em que o pacote chega no roteador), e 'tamanho' (tamanho do pacote).

Foi criada também a classe 'Conexão' que possui um array de pacotes CBRs 'filaCbr', onde o intervalo deles é baseado na variável "chegada_prox_pacote_cbr", implementada no código feito em aula com o valor de 0,02. A classe possui também em sua estrutura uma variável de duração da conexão 'duracao_conexao', em que colocamos todas as conexões com mesma duração, também existe o tempo atual 'tempo_atual' que é o tempo em que a conexão começou, e por fim também dentro da classe o tamanho de cada pacote 'tam'. A filaCbr é composta por objetos do tipo Pacote.

Na classe 'ResolveRoteador', foi implementada toda a lógica contida na função 'main' do algoritmo da sala de aula como também as funções, 'minimo' (acha o valor mínimo entre dois valores), 'gera_tam_pct' (obtém o tamanho dos pacotes web), 'chegada_pct' (retorna um tempo de chegada aleatório), e 'aleatorio' (retorna uma valor de ponto flutuante aleatório entre 0, 1)

Além das funções citadas anteriormente que foram implementadas em sala de aula, e adaptadas Java, tornando-se métodos, novos métodos também foram implementados para solucionar o problema em questão. São eles:

- proxPacote: recebe uma lista de conexões sendo esta do tipo 'Conexão', em que cada conexão contém uma fila de pacotes cbr. Então é verificado o tempo em que o primeiro pacote de cada fila iniciou até que o menor tempo seja encontrado. Assim o pacote a qual pertence o menor tempo de início, é removido da fila, e se a conexão de pacotes fique com uma fila de pacotes cbr fique vazia, esta é removida da conexão;
- ocupacao_pct_Web: calcula o valor da ocupação de um pacote web (tamanho do pacote * intervalo / link)
- ocupacao_pct_Cbr: calcula o valor da ocupação de um pacote web (tam do pacote * intervalo / link) * (duracao simulação / intervalo)
- ocupacao_total: calcula a ocupação total da simualação ocupacao cbr + ocupação web * (duracao simulação / ocupacao cbr);
- intervaloPacote: calcula o intervalo de tempo entre os pacotes 1.0 / (largura de banda * ocupação total) / (tamanho médio do pacote). Este método atribui valor a variável intervalo utilizada no método Resolve;

- Resolve: possui toda a lógica implementada na ‘main’ do algoritmo feito em sala. Porém com ajustes. Dentro do loop do tempo de simulação, foi mantida a lógica de criação de pacotes. Mas adicionamos uma verificação para a chegada da próxima conexão. Ontem o tempo atual da simulação é comparado com o tempo da chegada da próxima conexão. Caso os dois tempos sejam iguais um novo pacote cbr é adicionado a uma fila de conexões. Outra mudança também notável é a quando a condição do pacote cbr é verdadeira, este pacote é obtido pelo método ‘proxPacote’, citado acima.

3. Resultados

Ocupação (%)	Ocupação Obtida	Little	En	Ew	Lambda
40	0,447963853474792	0,000000344541812	0,78153710295486	0,00066034360	1183,530500
60	0,647982578212692	-0,000000226620438	1,75010080332587	0,00099976180	1750,518000
80	0,848021742236203	0,000000105788430	5,13882715122413	0,00221788556	2316,993777
90	0,948144135719695	0,000001445345109	16,5956577744826	0,00638072160	2600,905900
95	0,998156286104185	-0,000002238927038	494,073719188896	0,18015054090	2742,560300
99	1,000085995703080	-0,000000489875674	522183,724760263	182,836753646	2856,010700