Universidade Federal do Rio de Janeiro
Departamento de Ciência da Computação

Trabalho de Simulação

Autores:
Marco Vinícius Lima Reina de Barros
Pedro Henrique Pereira de Jesus
Ronald Andreu Kaiser

Todos os membros do grupo participaram da implementação do código e da documentação do relatório.

26 de Novembro de 2010

Conteúdo

Capítulo 1

Introdução

1.1 Funcionamento geral

O simulador possui uma lista de eventos que é processada continuamente, até alcançar um número máximo de clientes que desejamos atender por rodada.

São executadas tantas rodadas quanto forem necessárias até todos os intervalos de confiança dos valores que estão sendo estimados forem válidos, ou seja, <= 10% da média do estimador.

Inicialmente, calculamos o tempo de chegada do primeiro cliente que representa um evento de chegada no sistema. A passsagem de um cliente pelo sistema possibilita a criação dos seguintes eventos:

- <tempo, tipo: chegada no sistema>
- <tempo, tipo: entrada no servidor pela primeira vez>
- <tempo, tipo: saída do servidor>
- <tempo, tipo: entrada no servidor pela segunda vez>

Quando um evento de chegada ocorre, outro evento de chegada é criado com o tempo definido com o tempo de chegada baseado em uma distribuição exponencial, que representa o tempo de chegada do próximo cliente. Deste modo, os clientes vão chegando no sistema e a lista de eventos é processada.

Quando um evento é processado, ele é removido da lista de eventos e os novos eventos gerados a partir deste são criados e adicionados na lista, ordenada pelos tempos em que cada evento ocorre.

Todos os parâmetros, descritos na seção 1.7 são passados para o simulador em sua inicialização.

1.2 Estruturas internas utilizadas

Para viabilizar a implementação da ideia geral apresentada acima, dividimos o simulador em alguns módulos, abaixo estão explicitados os mais importantes:

Módulos utilitários:

- Estimator: Módulo que possui métodos para retornar os estimadores de média, variância e calcula intervalos de confiança.
- Dist: Módulo com o método que retorna os tempos aleatórios de chegada de uma distribuição exponencial.

Classes:

- Client: classe que representa um cliente que entra no sistema. Possui seus tempos de entrada e saída da fila, tempo no servidor e cor.
- EventHeap: classe que representa a lista de eventos que é processada durante uma rodada de simulação.
- Simulator: classe que implementa a lógica principal do simulador, processa as rodadas tratando os eventos e as chegadas dos clientes. E calcula as estimativas das variáveis aleatórias.
- Analytic: classe que serve para calcular os resultados de forma analítica.

1.3 Linguagem de Programação

Para a codificação do simulador foi utilizada a linguagem de programação Python, versão 2.5.5.

1.4 Geração de variáveis aleatórias

A linguagem Python utiliza o gerador de números aleatórios "Mersenne Twister", um dos métodos mais extensivamente testados existentes.

O método garante que a sequência de números gerados pela chamada random() só se repetirá em um período de $2^{19937} - 1$. Como o período é bem extenso, não precisamos nos preocupar com redefinir seeds que gerassem sequências sobrepostas.

A semente inicial utilizada pelo gerador, por default, é o timestamp corrente no momento do import do módulo random.

1.5 Métodos utilizados

Foi utilizado o método replicativo para a simulação.

1.6 Implementação do conceito de cores

O conceito de cores foi implementando adicionando o atributo "color"no objeto Client, que possui 2 valores: TRANSIENT ou EQUILIBRIUM. O número de clientes que representam a fase transiente são associados à cor TRANSIENT e os outros clientes são associados à cor EQUILIBRIUM.

Ao final da rodada de simulação os clientes que possuem a cor TRANSIENT são descartados do cálculo dos estimadores.

1.7 Escolha dos parâmetros

Ao iniciar o simulador, são executados em sequência todas as simulações necessárias para obtermos todos os dados requeridos para ambas as políticas de atendimento com os parâmetros:

- $\rho = 0.2$ - # de clientes na frase transiente = 30000
- $\rho = 0.4$ - # de clientes na frase transiente = 40000
- $\rho = 0.6$ - # de clientes na frase transiente = 80000
- $\rho = 0.8$ - # de clientes na frase transiente = 400000
- $\rho = 0.9$ - # de clientes na frase transiente = 500000

A escolha do número de clientes da fase transiente para cada utilização foi estimada de acordo com o que é exposto no capítulo 3.

O número de clientes processados a cada rodada é um parâmetro de entrada para o simulador. Para o cálculo dos resultados foram utilizados apenas os dados de 100000 clientes, sem contar os presentes na fase transiente.

1.8 Máquina utilizada

Para a simulação utilizamos uma máquina com as seguintes configurações:

- Processador: Intel Core Duo 2 GHz
- Memória: 2GiB DDR 2 667

- Sistema Operacional: MAC OS X 10.5.8

Incluir durações(fator mínimo):

Capítulo 2

Teste de Correção

Nesta seção você descreverá os testes de correção que foram efetuados para garantir o pleno funcionamento do simulador. Você deve demonstrar que o seu programa está simulando exatamente e com correção o esquema proposto. As fórmulas analíticas não podem ser utilizadas para garantir a correção. Servem apenas de orientação, pois na maioria das vezes partimos para a simulação exatamente por não termos os resultados analíticos. Procure rodar o simulador com cenários determinísticos com estatística conhecida, demonstrando que o programa está correto. Você deverá anexar comentários sobre a boa qualidade dos intervalos de confiança obtidos e como os valores exatos se encaixam nestes intervalos, para os diversos valores de r.

Capítulo 3

Estimativa da fase transiente

Nesta seção você descreverá como a fase transiente foi estimada para os diversos valores de r (obviamente existe um caso mais crítico). A fase transiente deve sempre implicar num certo número de eventos de partida que são desprezados, esperando que o sistema entre em equilíbrio. Este número de partidas em cada cenário e para cada valor de utilização deve ser documentado, qualquer que seja o método escolhido para determinar o fim da fase transiente. A determinação da fase transiente é obrigatória, pois é um exercício para determinar a entrada em equilíbrio do sistema. Você terá que justificar suas escolhas. Este é um processo empírico. Apresente resultados quantitativos que justificam sua escolha. Se você usou o método batch, além da estimação da fase transiente, mostre como as estatísticas entre as rodadas foram coletadas. Procure demonstrar a influência da escolha da fase transiente na qualidade das medidas. É preciso indicar com clareza se a estimativa utilizada é a mesma para os diferentes cenários e diferentes valores da utilização. A determinação da fase transiente deve ser independente da semente inicial. Comprove isso.

Capítulo 4

Resultados

Comente os resultados obtidos. Procure analisar a evolução dos valores e o porquê de sua obtenção. Garanta que todos os resultados analíticos estão dentro do intervalo de confiança. Isso é essencial! Apresente os resultados analíticos conhecidos junto com os valores medidos. Para cada utilização indique precisamente o número de rodadas, o tamanho das rodadas (explique como foi determinado), e o tamanho da fase transiente.

4.1 Tabelas

4.2 Comentários

Capítulo 5

Otimização

Para cada valor de utilização e para cada cenário, obtenha os resultados otimizados, isto é, considerando sempre o número de eventos de partida computados, procure determinar FATOR MÍNIMO (disciplina) = número de rodadas x tamanho de rodada (número de partidas) + fases transientes (número de partidas desprezadas), que satisfaz no seu simulador aos requisitos do tamanho do intervalo de simulação, independente do valor de utilização. Este fator mínimo deve ser independente da semente e isso tem que ser demonstrado. Obviamente, este FATOR será obtido para o caos mais crítico, entre parâmetro e valor de utilização. Com seu simulador operando corretamente, a busca da combinação ótima dará um bônus extra ao grupo que conseguir os menores valores, dentro de uma margem de 20MÍNIMO e outro tenha conseguido 1.200.000, eles estarão tecnicamente empatados e ambos ganharão bônus (20

Capítulo 6

Conclusões

Coloque aqui seus comentários finais. Descreva dificuldades encontradas, as otimizações feitas, e outras conclusões que você tirou do trabalho. Comente o que poderia ser melhorado, como, por exemplo, o tempo de execução do seu programa. Adicione quaisquer comentários que você julgar relevante. Cada uma das seções terá sua avaliação. Portanto, não deixe de colocar nenhuma seção no seu relatório. Se você não incluir uma seção, deixe-a em branco, mas não altere a numeração. Recomendamos fortemente que isso não ocorra. Não deixe de ler o capítulo de simulação na apostila.

Capítulo 7

Implementação

Anexo - Listagem documentada do programa A documentação do programa fonte deverá ser feita com rigor, explicando cada sub-rotina ou passo da programação. Código fonte sem comentários não é aceitável. Mande a listagem do código fonte como um anexo ao relatório. Uma versão eletrônica do documento impresso deve ser disponibilizada. O Grupo deve apresentar um executável funcionando em PC Windows ou Linux Ubuntu. O relatório completo deve ser entregue impresso (e não em mídia eletrônica). O programa executável deverá ser enviado para aguiar@nce.ufrj.br. Se o Grupo usar alguma linguagem específica, ele deve compilar o ambiente e apresentar um executável que rode sem necessidade de instalação especial, em Windows 7 ou Ubuntu. O programa será testado com alguns valores particulares para averiguar sua correção e integridade.