Replicação de Processos

Marlon Henry Schweigert[1]**,Jonathan de Oliveira Cardoso**[1]

[1]Centro de Ciências Tecnológicas – Universidade do Estado de Santa Catarina (UDESC)
Joinville – Santa Catarina – Brasil

{marlon.henry,jonathan.cardoso}@edu.udesc.br

Resumo. *Este artigo descreve a construção do serviço para resolução de empacotamento de pesos utilizando tolerância a falhas entre os processos, com mecanismos de lockstep, recovery e health.*

1. Replicação

A replicação de processos é um ponto crucial para o funcionamento de sistemas de alta disponibilidade. Grandes empresas atuais utilizam dessa abordagem para tratar grandes problemas de escalabilidade utilizando replicação de dados e processos.

Ter um sistema altamente escalável, e consecutivamente replicável, é um ponto chave de sistemas como NetFlix, Google, Facebook e afins.

O objetivo desde relatório é resumir as técnicas de replicação do serviço criado, a qual seu objetivo e retornar a resposta do problema de otimização para empacotamento de cargas implementado em java[Bloch 2001].

2. Serviço

O serviço criado está dividido em 4 camadas:

1. FrontEnd: Responsável por ler requisições do terminal, tratar problemas da sintaxe da requisição e descartar requisições inválidas. É executado em uma thread.
2. BackEnd: Responsável por ler as requisições de pacotes tratados pelo FrontEnd e executar a 2 resoluções, varificando se a integridade da solução está correta (Lockstep). É executado em uma thread.
3. Solver: Responsável por executar os algoritmos de solução. É executado em 2 threads.
4. Health: Responsável por verificar a execução de todos os seriços da aplicação e restaura-los caso necessário. É executado em 1 thread.

3. Plano de Testes

Foi criado um script para automatizar os testes, da seguinte maneira: Foi executado uma bateria de testes com 10 requisões cada uma com 7 entradas(bin-packing). Essa bateria de testes foi repitida por 10 vezes. Ao final destas execução, foi gerado um arquivo com o tempo de recuperação do frontend, backend, tempo de solução, tempo completo de execução.

4. Resultados

Figura 1. Recuperação do Frontend

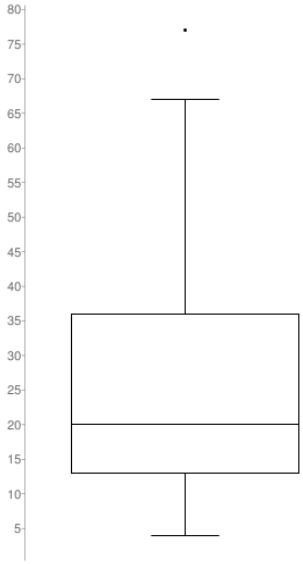

Figura 2. Recuperação do Backend

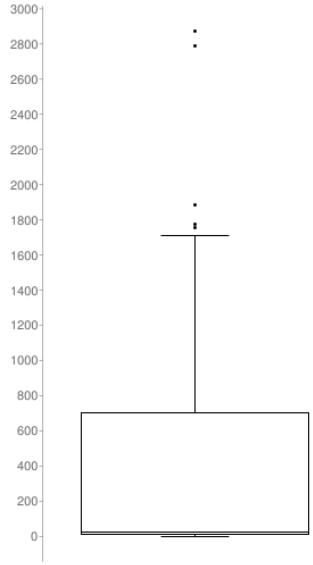

Tabela 1. Recuperação do Frontend

CATEGORIA	VALOR
Média	26,87ms
Desvio Padrão	17,42ms
População	45 unidades

Estes dados são referentes a recuperação do serviço de leitura de requisições.

Tabela 2. Recuperação do Backend

CATEGORIA	VALOR
Média	389,95ms
Desvio Padrão	653,00ms
População	88 unidades

Estes dados são referentes a recuperação do serviço resolução de requisições.

Figura 3. Solução Sem Erros

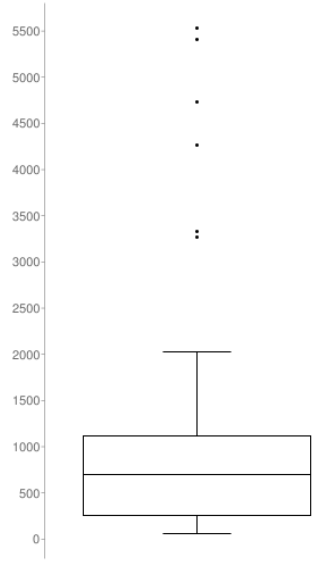

Figura 4. Solução Final

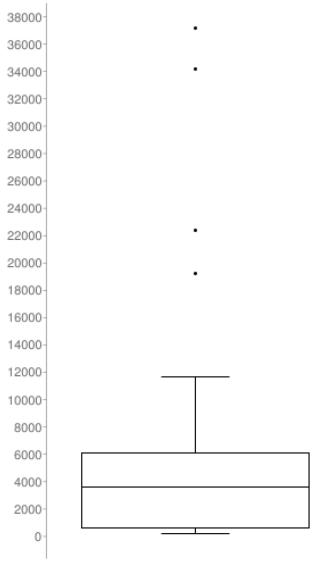

Tabela 3. Tempo de resposta da solução

CATEGORIA	VALOR
Média	893,49ms
Desvio Padrão	906,98ms
População	148 unidades

Estes dados são referentes ao tempo de resposta de uma solução a qual não ocorre erros.

Tabela 4. Tempo de resposta da solução ao usuário

CATEGORIA	VALOR
Média	5791,01ms
Desvio Padrão	8426,14ms
População	41 unidades

Estes dados são referentes ao tempo de resposta a qual ocorrem erros no BackEnd.

5. Conclusão

Podemos observar que o tempo de recuperação do backend é muito mais crítico que o frontend, pois ao ocorrer um erro no backend é necessário recuperar toda a solução.O tempo de solução com os possíveis erros foi de até 10x o tempo de execução sem erros.

Levantamos a hipótese de que se existisse uma gravação temporária da possível solução diminuiria significativamente o tempo de recuperação do backend.

Referências

Bloch, J. (2001). *Effective Java*.