Uma implementação *MapReduce* sobre Akka - Caso LRIT.

Jonas Ferreira da Silva Medeiros De La Cerda

Universidade Federal Fluminense

4 de dezembro de 2015

LRIT - O que é?

- Sistema estabelecido pela IMO (*International Maritime Organization*) visando à **segurança** (*security*) e **salvaguarda** (*safety*) da vida humana ao mar;
- Obrigatório aos países signatários da convenção SOLAS (em torno de 200 nações) para as **embarcações mercantes** acima de uma determinada tonelagem.

Instituto de Computação

LRIT - O que faz?

Permite o intercâmbio de informações de posições dos **navios mercantes** entre os governos contratantes.

- Monitorar áreas;
- Monitorar embarcações;
- Incidentes SAR (*Search and Rescue*);
- Interrogar embarcações.

Instituto de Computação

LRIT - Overview

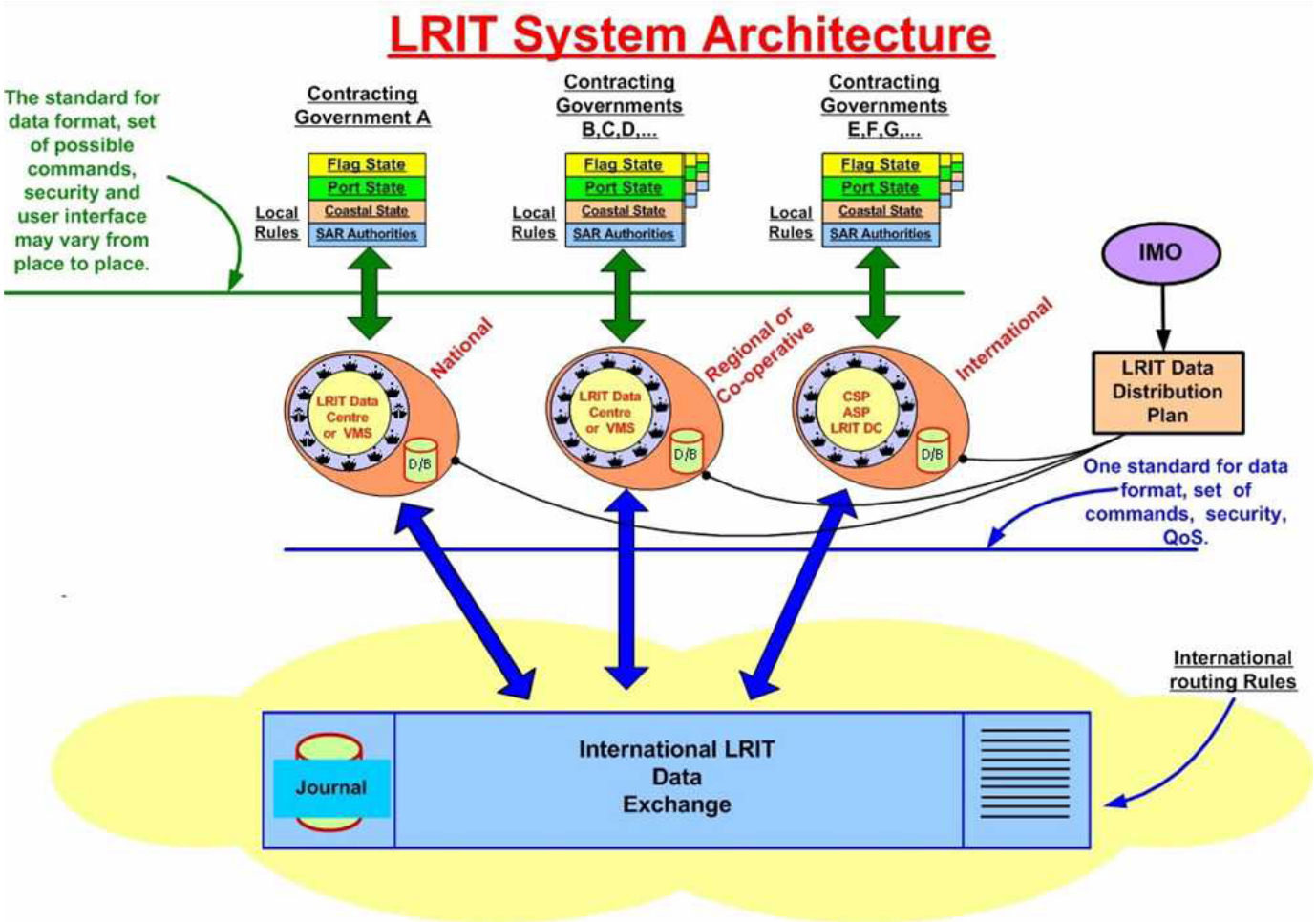


Figura: Visão geral do sistema LRIT

LRIT - Características

- Troca de mensagens assíncrona;
- Funciona sobre internet utilizando SOAP/XML e HTTPS;
- Requisitos de desempenho e disponibilidade;
- Tecnologicamente agnóstico.

Instituto de Computação

LRIT - Brasil

- JavaEE;
- JBoss;
- EJBs, MDBs, apache httpd (mod_jk);
- PostgreSQL e PostGIS.

Instituto de Computação

LRIT - Reúso

- A mesma arquitetura e componentes podem ser reinstanciados para outros tipos de veículos.
- Volume de dados a ser processado (tanto em tempo real quanto em lote) pode crescer.
- Adequações ao processamento devem ser feitas a fim de suportar demandas maiores.

Instituto de Computação

MapReduce[Dean and Ghemawat, 2004]

- Modelo de programação e implementação para processar grandes conjuntos de dados;
- Baseado em construtos da programação funcional: *Map* e *Reduce*.
- Recebe um conjunto de chaves e valores, produz um conjunto de chaves e valores.

Instituto de Computação

MapReduce - Overview

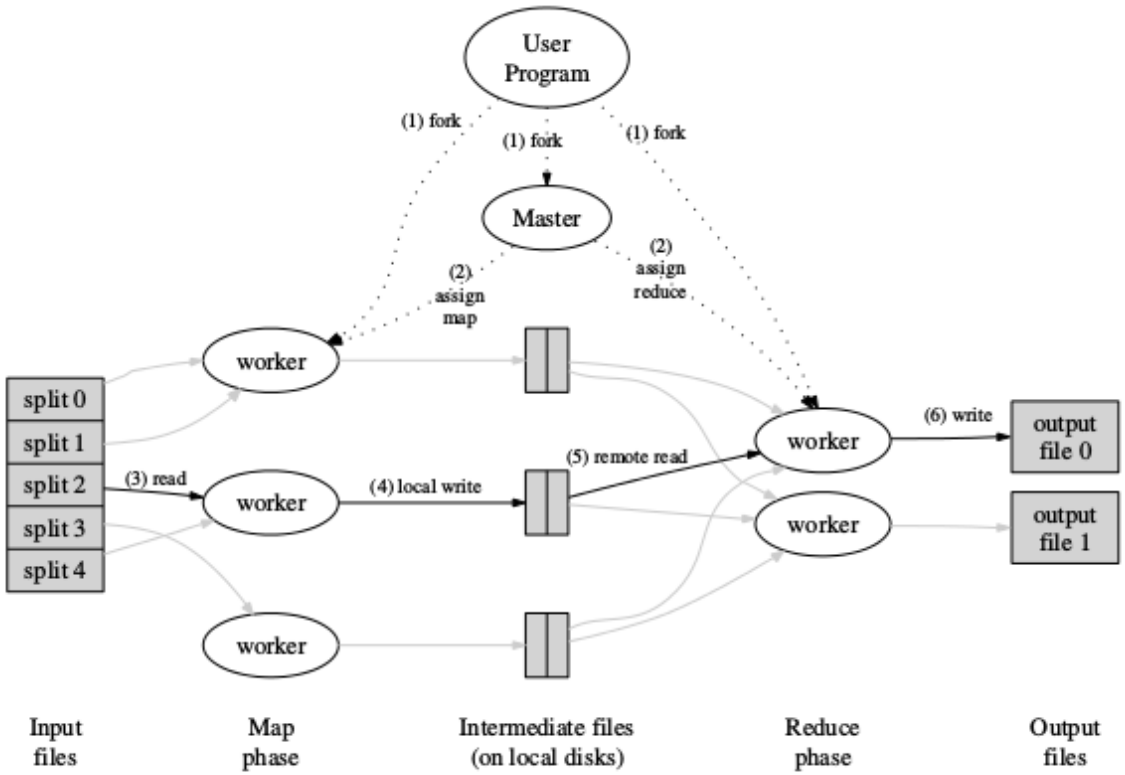


Figura: MapReduce overview[Dean and Ghemawat, 2004]

Instituto de Computação

MapReduce - Exemplo

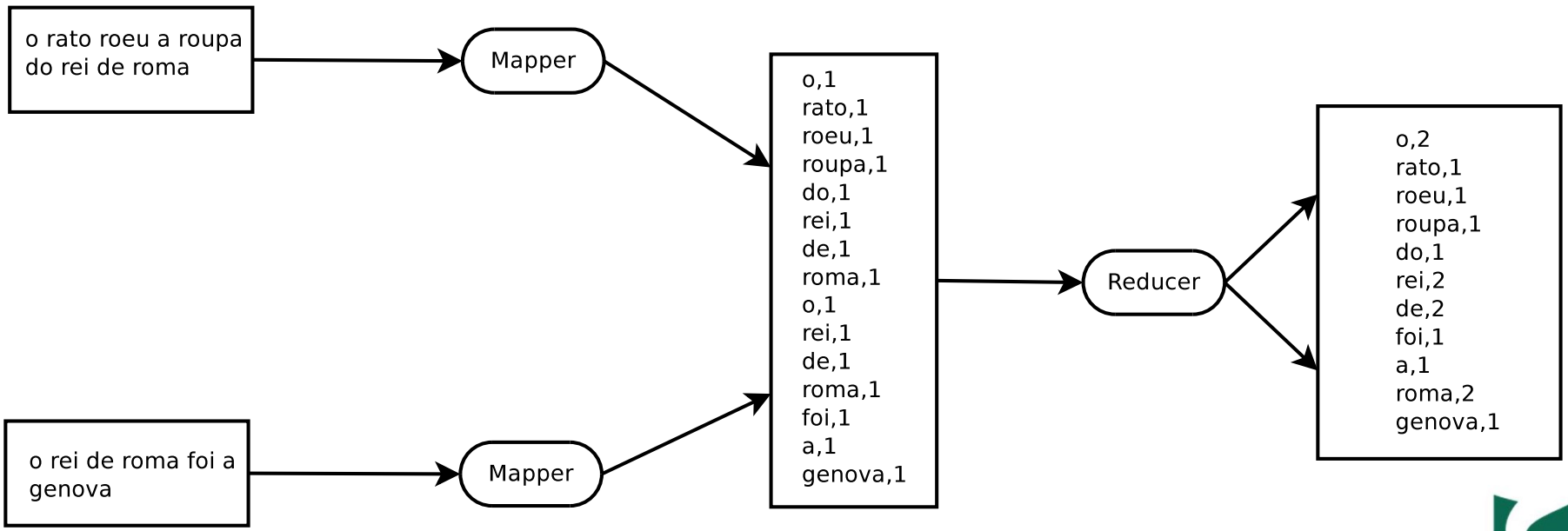


Figura: Problema de contagem de palavras.

Instituto de Computação

Problema

Solução atual não escalará com:

- Aumento de embarcações;
- Aumento da frequência de relatórios de posições (15 minutos / 15 segundos);
- Aumento do número de requisições;
- Aumento do número de áreas monitoradas.

Instituto de Computação

Solução - Objetivo

Implementar uma solução que seja mais simples de escalar quando a demanda crescer. Tais funcionalidades devem ser escaláveis:

- Busca em área: selecionar posições presentes em uma área;
- Busca de interessados: selecionar áreas que contenham uma posição.

Instituto de Computação

Solução - Prova de Conceito

- Busca de interessados;
- Assíncrono;
- Distribuído;
- MapReduce;
- Master/Slave;
- Scala[1] e Akka[2];
- Disponível em https://github.com/chandonbrut/mestrado/gridcomp

[1] https://scala-lang.org
[2] https://akka.io

Instituto de Computação

Solução - Overview

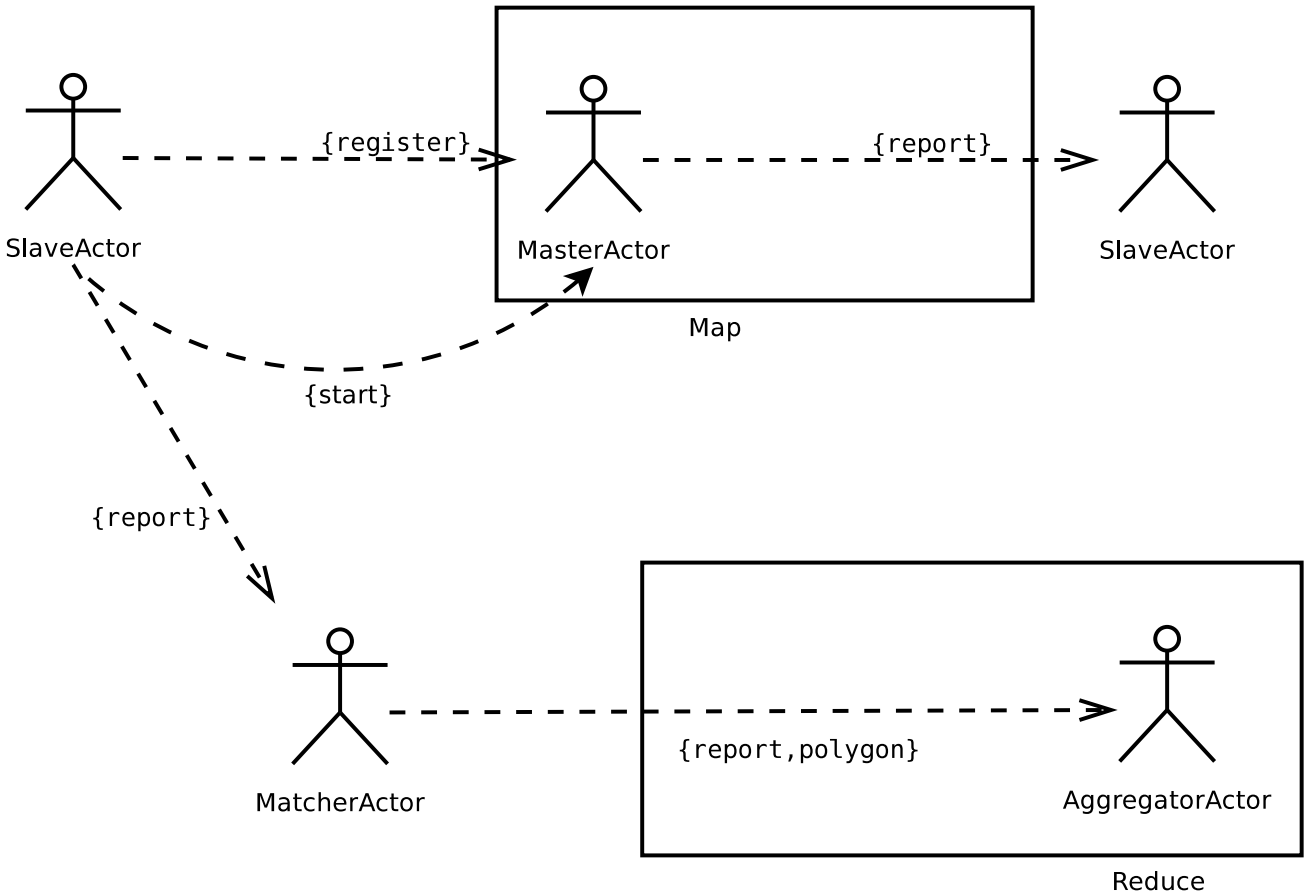


Figura: Visão geral da solução.

Solução - Overview

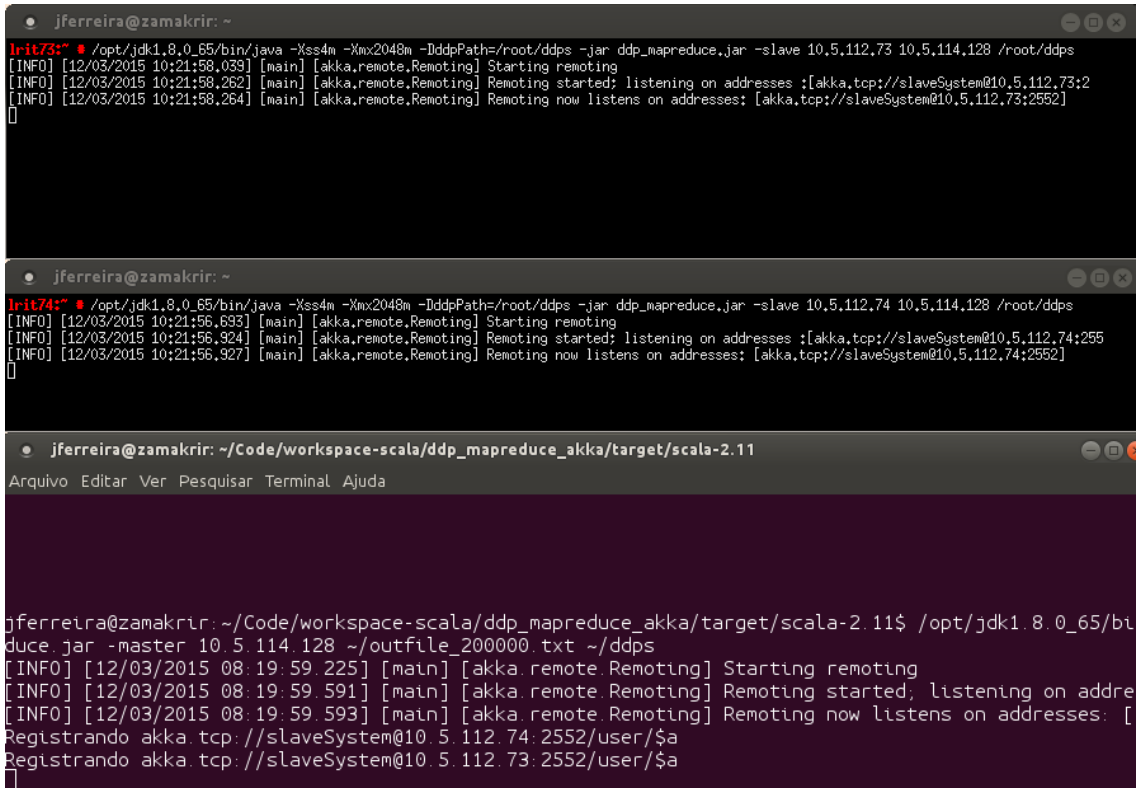


Figura: Exemplo com dois *slaves* conectados.

Instituto de Computação

Cenário

- Amostras de posições de navios de tamanhos: 10000, 20000, 100000, 200000;
- Cluster com 1, 2 e 3 slaves.
- Cada posição deve ser comparada com 1000 polígonos.

Instituto de Computação

Abordagem Sequencial

- CPU: i7-2600 CPU @ 3.40GHz (8-core)
- MEM: java -Xss4m -Xms4096m -Xmx4096m

#pos	Tempo(s)	Throughput(pos/s)
10000	231	43.29
20000	460	43.48
100000	2334	42.84
200000	4647	43.03

Instituto de Computação

Abordagem Paralela

- 1 slave;
- CPU: Xeon 5120 @ 1.86GHz (4-core)
- MEM: java -Xss4m -Xms2048m -Xmx2048m

#pos	Tempo(s)	Throughput(pos/s)
10000	660	15.15
20000	1171	17.07
100000	6062	16.49

Instituto de Computação

Abordagem Paralela

- 2 slaves;
- CPU: Xeon 5120 @ 1.86GHz (4-core)
- MEM: java -Xss4m -Xms2048m -Xmx2048m

#pos	Tempo(s)	Throughput(pos/s)
10000	368	27.17
20000	733	26.88
100000	3507	28.51
200000	6882	29.06

Instituto de Computação

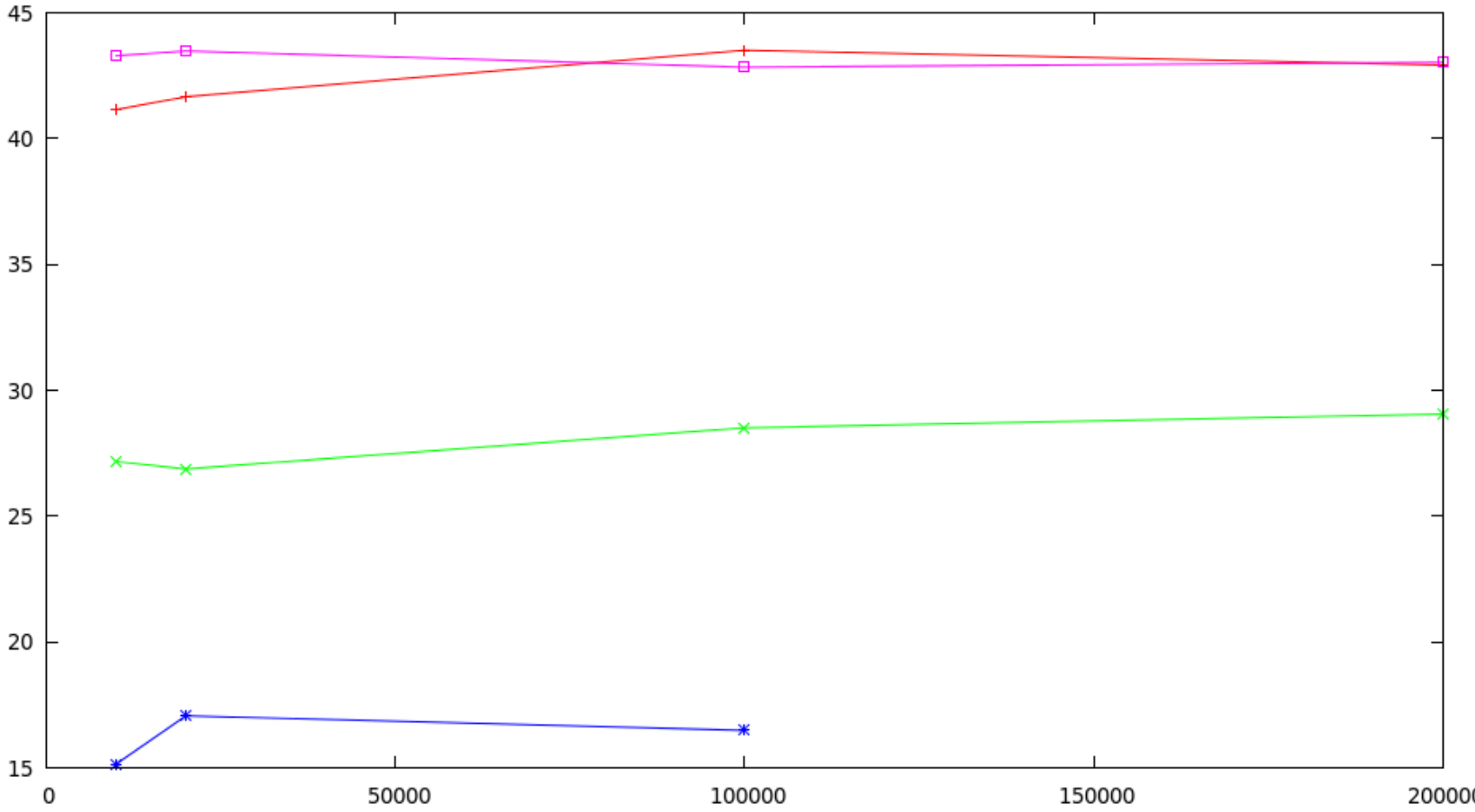
Abordagem Paralela

- 3 slaves;
- CPU: Xeon 5120 @ 1.86GHz (4-core)
- MEM: java -Xss4m -Xms2048m -Xmx2048m

#pos	Tempos(s)	Throughput(pos/s)
10000	243	41.15
20000	480	41.66
100000	2298	43.51
200000	4659	42.92

Instituto de Computação

Comparativo



Resultados

- Prova de conceito operacional e adequada para trabalhos em lote.
- *Slave nodes* podem ingressar e sair a qualquer momento, porém implementação ainda não contempla redistribuição do trabalho.
- *Overhead* aceitável considerando a facilidade de escalar recursos.
- *Throughput* quase constante em abordagens paralelas.

Instituto de Computação

Resultados

- Apesar do *framework* fornecer os meios para garantir resiliência e recuperação, implementação de fato é complexa.
- Facilidades como roteadores de mensagens com diferentes políticas: *RoundRobin*, *SmallestMailbox*, etc.
- Desempenho escalando linearmente de acordo com o número de *slaves*.

Instituto de Computação

Futuro

- Implementar um protocolo melhor e tolerância a erros;
- Testar com múltiplos *jobs*;
- Adicionar capacidade de elasticidade com *job* em andamento.

Instituto de Computação

Bibliografia I

Dean, J. and Ghemawat, S. (2004).
Mapreduce: Simplified data processing on large clusters.
In *Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6*, OSDI'04, pages 10–10, Berkeley, CA, USA. USENIX Association.

Instituto de Computação