

Relatório dos Seminários do Laboratório de Sistemas da Computação nos anos de 2003, 2004 e 2005

Lucas Mello Schnorr, Marcelo Veiga Neves, Marcelo Pasin

Dezembro 2005

1 Introdução

O Laboratório de Sistemas de Computação - LSC - tem finalidade de desenvolver atividades de ensino, pesquisa e extensão relacionadas a sistemas de computação. Dentro deste laboratório se desenvolvem trabalhos nas áreas de:

- Processamento Paralelo
- Sistemas Distribuídos
- Sistemas Operacionais
- Arquitetura de Computadores
- Redes de Computadores

O laboratório tem o suporte das agências de fomento tal, tal e tal e é composto por um grupo de pesquisadores doutores, um grupo de mestrandos e outro, composto de graduandos, de iniciação científica. Todos trabalham separados em alguma área de interesse, listadas acima.

O objetivo deste relatório é mostrar o resumo das apresentações feitas ao longo do ano de 2003 e 2004. Essas apresentações ocorrem quase que sempre semanalmente e tem como objetivo, além da integração do grupo, mostrar os resultados de uma determinada linha de pesquisa. Nestas reuniões ocorrem

discussões sobre a continuação do trabalho do apresentador, proporcionando uma pesquisa de maior qualidade, já que recebe sugestões construtivas de todo o grupo.

2 Trabalhos do ano de 2005

Esta seção inicialmente lista os trabalhos realizados durante o ano de 2005, até a data de edição deste relatório. Depois, são mostrados os resumos de cada trabalho, que detalham o assunto e o que foi feito. Os resumos foram desenvolvidos pelos seus autores.

Os trabalhos apresentados em 2005, até a data deste relatório, são:

1. **Cleverton Marlon Possani**, em 26/04/2005, apresentou: *Sistema CADEO*
2. **Elton Nicoletti Mathias, Marcelo Veiga Neves e Tiago Scheid**, em 10/05/2005, apresentou: *Computação P2P*
3. **Marcelo Veiga Neves, Tiago Scheid e Andrea Schwertner Charão**, em 31/05/2005, apresentou: *Monitoração de clusters com a ferramenta Ganglia: avaliação e adaptação*
4. **Daniel Michelin De Carli**, em 07/06/2005, apresentou: *Computadores Paralelos e Computação*
5. **Márcio Parise Bouffleur**, em 07/06/2005, apresentou: *Projetando Algoritmos Paralelos*
6. **Edmar Pessoa Araújo Neto**, em 22/09/2005, apresentou: *Reestruturação do Gerenciamento de Memória do Pajé*
7. **Elton Nicoletti Mathias**, em 29/09/2005, apresentou: *Improving Peer-to-Peer Resource Usage Through Idle Cycle Prediction*
8. **Geovani Ricardo Wiedenhof**, em 06/10/2005, apresentou: *Ferramenta para o rastreamento e visualização de programas java usando JVMTI*
9. **Juliano Foletto Reckziegel**, em 20/10/2005, apresentou: *Adicionando suporte a soquetes brutos na biblioteca MicroVAPI*

10. **Elton Nicoletti Mathias, Marcelo Veiga Neves, Tiago Scheid, Edmar Pessoa Araújo Neto, Andrea Charão**, em 21/11/2005, apresentou: *Extensão da Ferramenta IC2D para Monitoração de Carga em Clusters e Grids de Computadores*
11. **Edmar Pessoa Araújo Neto, Marcelo Veiga Neves, Tiago Scheid, Elton Nicoletti Mathias, Andrea Charão**, em 21/11/2005, apresentou: *Comparação entre Plataformas Peer-to-Peer para Desenvolvimento de Aplicações de Computação Distribuída*
12. **Guilherme Koslovski, Márcio Bouffleur, Andrea Charão**, em 01/12/2005, apresentou: *Migração de Processos Utilizando Xen*

2.1 Resumos

1. **Cleverton Marlon Possani**

Sistema CADEO

É de conhecimento geral, que em sistemas computacionais existe uma grande defasagem na relação potencialidade X disponibilidade dos computadores, ocorrendo na maioria dos casos, um grande desperdício de processamento, ou seja, um computador de última geração, de alto poder de computação por segundo, se ocupado durante seis horas diárias, sua relação custo/benefício é muito alto. Este desperdício de processamento é conhecido como tempo de ociosidade. Fazer bom uso da ociosidade de um grupo seletivo de computadores, significa, ter a oportunidade de aproveitar este tempo de desuso da maneira mais apropriada. Com o objetivo de suprir esta necessidade surge o sistema CADEO (Controle de Alocação Dinâmica de Estações Ociosas) que tem o objetivo de gerenciar computadores capazes de efetuar processamento sem interferir sua tarefa primária. Neste seminário será apresentado uma continuidade de trabalho dentro do sistema CADEO que enfatiza o módulo Alocador, este é responsável por criar aglomerados dinâmicos com objetivos de atender requisições de processamento de aplicações.

2. **Elton Nicoletti Mathias, Marcelo Veiga Neves e Tiago Scheid**

Computação P2P

O termo "peer-to-peer" (P2P) se refere a uma classe de sistemas e aplicações que utilizam recursos distribuídos para realizar uma função crítica de maneira descentralizada. Com a existência de uma grande disponibilidade de recursos distribuídos em todo o mundo, a utilização de sistemas P2P para agregar

esses recursos na solução de problemas específicos, tem atraído a atenção de pesquisadores em diversas áreas, inclusive no processamento de alto desempenho. Alguns dos benefícios do P2P incluem: aumento da escalabilidade graças a abordagem descentralizada; eliminação dos custos de infraestrutura (servidores) através da conexão direta entre os clientes; e agregação de recursos distribuídos. Esta apresentação abordará os conceitos e características em torno da computação P2P , bem como uma classificação dos sistemas existentes.

3. **Marcelo Veiga Neves, Tiago Scheid e Andrea Schwertner Charão**
Monitoração de clusters com a ferramenta Ganglia: avaliação e adaptação

Neste artigo apresenta-se uma avaliação de Ganglia, que é uma das principais ferramentas de Software Livre para monitoração de clusters de computadores. Descreve-se também algumas adaptações efetuadas nesta ferramenta, tornando-a capaz de monitorar programas paralelos e permitindo a sua integração com outras ferramentas de monitoração. Estas adaptações ampliam as possibilidades de utilização de Ganglia, contribuindo para sua valorização junto à comunidade de Software Livre aplicado ao processamento de alto desempenho.

4. **Daniel Michelon De Carli**
Computadores Paralelos e Computação

O poder computacional teve avanços gigantescos desde suas primeiras aplicações (cálculos balísticos por exemplo). Sendo assim muitos podem pensar que em algum determinado momento as máquinas iriam ter o poder computacional necessário para executar todas as aplicações. Entretanto modelo computacional tradicional (von Neumann) tem limitações físicas (como interferências elétricas e até mesmo a velocidade da luz, por exemplo). Além disso técnicas de paralelização já são utilizadas atualmente em uma granularização baixa nos computadores convencionais (pipelining, arquiteturas super escalares entre outros). Assim sendo a busca de um aumento de desempenho não pode ser conseguida levando somente em consideração o tempo de execução de uma única instrução por ciclo de clock. Desse modo será apresentado basicamente uma visão introdutória a respeito da computação paralela. Analizando o primeiro capítulo do livro *Designing and Building Parallel Programs*, de Ian Foster.

5. **Márcio Parise Bouffleur**
Projetando Algoritmos Paralelos

Este trabalho tem como objetivo expor um ambiente para projeto de algoritmos paralelos, fazendo um resumo do capítulo 2 do livro *Designing and Building Parallel Programs*, de Ian Foster. Nele será mostrado como a especificação de um problema pode ser traduzida em um algoritmo que reflete concorrência, escalabilidade e localidade. Para tal, será mostrada uma abordagem metódica que disponibiliza um mecanismo para encontrar as melhores alternativas de projeto.

6. **Edmar Pessoa Araújo Neto**

Reestruturação do Gerenciamento de Memória do Pajé

A ferramenta de visualização Pajé é utilizada principalmente na depuração de programas paralelos em busca de aumento de desempenho. Para isso, ela foi desenvolvida para possuir três características (interatividade, extensibilidade e escalabilidade) importantes para viabilizar uma boa visualização. Esta apresentação irá propor uma reestruturação do gerenciamento de memória da ferramenta Pajé para aumento de escalabilidade.

7. **Elton Nicoletti Mathias**

Improving Peer-to-Peer Resource Usage Through Idle Cycle Prediction

ProActive provides a peer-to-peer infrastructure that can be deployed in a dynamic set of computers. It can take profit of idle CPU cycles, but their availability must be expressed in configuration files. In a very dynamic system, it is difficult to know in advance which or when the resources will be available. Besides, generating availability files can be a complex task if there is a large number of computers.

This work proposes an idleness detection model, based on time series forecast and resource usage pattern analysis. Such approach intends to provide a dynamic mechanism that make possible exploitation of idle resources, without disturbing normal operation of the computational environment. When used with the peer-to-peer infrastructure, this model allows to efficiently use non-dedicated environments for running large distributed applications.

A further benefit of monitoring the environment usage is an extension to the IC2D tool, that offers a graphical visualization of historical load values, such as CPU, memory and network usage, load average, etc.

8. **Geovani Ricardo Wiedenhof**

Ferramenta para o rastreamento e visualização de programas java usando JVMTI

A utilização da linguagem Java no desenvolvimento de aplicações paralelas e distribuídas que demandam alto desempenho e recursos de diferentes máquinas é crescente. Naturalmente, surge a necessidade de meios para a depuração do código Java desenvolvido. Essa depuração é mais complicada utilizando um depurador de aplicações Java seqüenciais quando o código Java possui fluxos de execução diferentes. Uma forma de auxiliar a depuração dessas aplicações é a geração de rastros durante a execução de uma aplicação Java para posterior visualização. Java oferece JVMTI, uma interface para recuperação dos principais eventos gerados pela JVM. Através dessa interface pode-se realizar o monitoramento e registro de eventos, necessários para a geração de rastros de execução. Esta apresentação descreverá o desenvolvimento de uma ferramenta de rastreamento de programas paralelos e distribuídos na linguagem de programação Java, configurável e transparente ao desenvolvedor da aplicação. Além disso, serão explicados sobre os decodificadores das informações registradas para as diferentes visualizações na ferramenta Pajé.

9. **Juliano Foletto Reckziegel**

Adicionando suporte a soquetes brutos na biblioteca MicroVAPI

A MicroVAPI é uma biblioteca de adaptação feita para porta chamadas de VAPI para DECK. O que faz com que aplicações escritas para rodar sobre Infiniband através da VAPI possam ser executadas nas redes suportadas pelo DECK (SCI, VIA, Myrinet e Ethernet). Quando utilizamos o DECK sobre Ethernet acabamos utilizando o protocolo TCP que possui uma grande sobrecarga. Pois implementa mecanismos de tolerância a falhas, que normalmente são dispensáveis em um aglomerado de computadores, pois estamos trabalhando sobre uma rede local e confiável. Para executar aplicações originalmente escritas para Infiniband sobre Ethernet com o máximo de aproveitamento proporcionado pela tecnologia esse trabalho propoe-se a implementar um módulo para a MicroVAPI que utilize soquetes brutos ao invés do DECK como alternativa de comunicação.

10. **Elton Nicoletti Mathias, Marcelo Veiga Neves, Tiago Scheid, Edmar Pessoa Araújo Neto, Andrea Charão**

Extensão da Ferramenta IC2D para Monitoração de Carga em Clusters e Grids de Computadores

A análise do ambiente de execução permite uma melhor depuração de programas distribuídos e a tomada de decisões mais apuradas a respeito da disponi-

bilidade de recursos para resolução de tarefas. Esta apresentação tratará de uma extensão ao ambiente gráfico de monitoração e controle IC2D, do middleware para computação distribuída ProActive. Esta extensão desenvolvida permite a coleta e visualização gráfica de índices de carga de ambientes de clusters e grids.

11. **Edmar Pessoa Araújo Neto, Marcelo Veiga Neves, Tiago Scheid, Elton Nicoletti Mathias, Andrea Charão**

Comparação entre Plataformas Peer-to-Peer para Desenvolvimento de Aplicações de Computação Distribuída

A apresentação expõe uma comparação entre plataformas Peer-to-Peer para desenvolvimento de aplicações de computação distribuída. Acredita-se que esta comparação possa fornecer subsídios para construção de novas aplicações distribuídas através da identificação das vantagens e desvantagens de algumas plataformas, em relação às necessidades mais frequentes e dos recursos frequentemente disponíveis.

12. **Guilherme Koslovski, Márcio Bouffleur, Andrea Charão**

Migração de Processos utilizando Xen

A migração de processos de um computador para outro é um tema recorrente em áreas ligadas à computação distribuída. No contexto do processamento de alto desempenho, mecanismos de migração de processos são potencialmente úteis para se alcançar objetivos clássicos, tais como distribuição dinâmica de carga, tolerância à falhas e localidade no acesso a dados. Atualmente, alguns sistemas com suporte à migração de processos vêm se tornando populares, como é o caso de Xen, um sistema baseado em virtualização, onde pode-se realizar a migração de uma determinada máquina virtual juntamente com seus processos em execução. Este seminário tem como objetivos apresentar alguns conceitos sobre virtualização e apresentar os procedimentos utilizados pelo sistema Xen para realizar a migração de máquinas virtuais.

3 Trabalhos do ano de 2004

Esta seção inicialmente lista os trabalhos realizados durante o ano de 2004. Depois, são mostrados os resumos de cada trabalho, que detalham o assunto e o que foi feito. Os resumos foram desenvolvidos pelos seus autores.

Os trabalhos apresentados em 2004, pelo Laboratório de Sistemas de Computação, são:

1. **Tiago Scheid**, em 25/03/2004, apresentou: *Estudo de Casos e Algoritmos*
2. **Guilherme Piegas Koslovski**, em 25/03/2004, apresentou: *Desenvolvendo Algoritmos Paralelos*
3. **Lucas Mello Schnorr**, em 02/04/2004, apresentou: *Estudo e Classificação de Ferramentas de Gerenciamento de Clusters*
4. **Rodrigo da Rosa Righi**, em 16/04/2004, apresentou: *Recursos que a tecnologia Infiniband oferece para aglomerados de computadores e a interface de programação de aplicação VAPI*
5. **Marcelo Veiga Neves e Tiago Scheid**, em 23/04/2004, apresentaram: *Ganglia - Características de Projeto e Implementação*
6. **Antônio Marcos de Oliveira Candia**, em 30/04/2004, apresentou: *Criptanálise Distribuída utilizando ataques de dicionário e de força-bruta em Java*
7. **Claudio Schepke**, em 07/05/2004, apresentou: *Comparando o desempenho entre Java e C++ em aplicações numéricas*
8. **Marcelo Veiga Neves e Tiago Scheid**, em 14/05/2004, apresentaram: *Monitoração de Aplicações Paralelas Utilizando Ganglia e Pajé*
9. **Rodrigo da Rosa Righi e Marcia Cristina Cera**, em 21/05/2004, apresentaram: *Características e facilidades do ProActive para a construção de aplicações e ferramentas Java*
10. **Lucas Mello Schnorr**, em 28/05/2004, apresentou: *Visualização Simultânea e Multinível de Informações de Monitoramento de Clusters*
11. **Rodrigo da Rosa Righi**, em 23/06/2004, apresentou: *Implementação de um ambiente para invocação remota e assíncrona de métodos sobre infiniband e DECK*
12. **Márcia Cristina Cera**, em 02/07/2004, apresentou: *Alocação Dinâmica de Recursos Computacionais Ociosos em Java*

13. **Cleverton Marlon Possani**, em 09/07/2004, apresentou: *Uma Proposta de Monitoramento de Rede para Laboratórios de Informática*
14. **Juliano Reckziegel**, em 16/07/2004, apresentou: *Utilização de DECK como suporte para a comunicação em programas Java*
15. **Diego Kreutz**, em 22/07/2004, apresentou: *Um Calculador de Capacidade de Computação para Nós de Máquinas Virtuais LAM/MPI*
16. **Lucas Mello Schnorr**, em 22/07/2004, apresentou: *Visualização da biblioteca de comunicação DECK em Pajé*
17. **Rodrigo Righi e Juliano Reckziegel**, em 22/07/2004, apresentou: *Invocação Remota de Métodos de Alto Desempenho*
18. **Edmar Pessoa Araujo Neto**, em 06/08/2004, apresentou: *Visualização online de aplicações paralelas no Pajé*
19. **Claudio Schepke**, em 12/08/2004, apresentou: *Avaliação da Biblioteca JMP para Paralelização de Métodos Numéricos em Arquitetura com Memória Compartilhada*
20. **Pablo Furlan da Silva**, em 19/08/2004, apresentou: *Desenvolvimento de um Módulo de Visualização Não Temporal de Dados para o Programa de Visualização Interativa de Aplicações Paralelas - Pajé*
21. **Geovani Ricardo Wiedenhof**, em 26/08/2004, apresentou: *Rastreamento de programas Java usando JVMTI*
22. **Andre Schmidt Dellamea**, em 02/09/2004, apresentou: *Implementação de um sistema distribuído em estrutura hierárquica para suporte para metacomputação*
23. **Elton Nicoletti Mathias e Marcelo Veiga Neves**, em 09/09/2004, apresentaram: *Balanceamento de Carga e Detecção de Ociosidade em Sistemas Distribuídos*
24. **Juliano Foletto Reckziegel e Rodrigo Righi**, em 16/09/2004, apresentaram: *Aldeia Sockets*
25. **Marcia Pasin**, em 23/09/2004, apresentou: *Alta disponibilidade para servidores de aplicação Enterprise JavaBeans*

26. **Claudio Schepke e Andrea Schwertner Charão**, em 07/10/2004, apresentaram: *Implementação Paralela do Método do Gradiente Conjugado Utilizando MPIJava*
27. **Marcelo Veiga Neves, Tiago Scheid, Lucas Mello Schnorr e Andrea Charão**, em 14/10/2004, apresentaram: *Integração de Garglia, libRastro e Pajé para o Monitoramento de Aplicações Paralelas*
28. **Rodrigo Righi, Philippe Navaux e Marcelo Pasin**, em 21/10/2004, apresentaram: *Sistema Aldeia: Invocação Remota e Assíncrona de Métodos sobre Infiniband e DECK*
29. **Diego Kreutz, Lucas Mello Schnorr e Marlon Possani**, em 21/10/2004, apresentaram: *Comportamento de Aplicações Paralelas em Aglomerados de Computadores Heterogêneos*
30. **Claudio Schepke, Andrea Schwertner Charão**, em 11/11/2004, apresentaram: *Bibliotecas Java para o Cálculo Numérico*
31. **Elton Mathias**, em 26/11/2004, apresentou: *Detecção de Ociosidade em Sistemas Distribuídos*
32. **Tiago Scheid**, em 26/11/2004, apresentou: *Comparação entre ferramentas de monitoração de clusters*
33. **Guilherme Piegas Koslovski**, em 02/12/2004, apresentou: *Estudo de Técnicas para Melhora do Desempenho de Aplicações Java*
34. **Edmar Pessoa Araujo Neto**, em 09/12/2004, apresentou: *Comparação de Desempenho Entre Objetos Distribuídos, RMI e RMI assíncrono*
35. **Antonio Marcos de Oliveira Candia**, em 09/12/2004, apresentou: *Criptanálise Distribuída de Alto Desempenho*

3.1 Resumos

1. **Tiago Scheid**

Estudo de Casos e Algoritmos

A computação paralela é utilizada amplamente em pesquisas científicas. Dentre as várias aplicações nessa área tomou-se três exemplos (problemas) para analisá-los e discutí-los, que são: Modelo Atmosférico, redução de área

em chips e química computacional. O meu trabalho visa analisar cada um dos casos e analisar os algoritmos usados para solucionar cada problema. Esse trabalho tem como objetivo principal estudar e compreender todos os processos de resolução do problema como particionamento, comunicação, aglomeração e mapeamento. A apresentação abordará justamente isso, como foi feito o particionamento, a comunicação, a aglomeração e o mapeamento em cada um dos três problemas citados anteriormente.

2. **Guilherme Piegas Koslovski**

Desenvolvendo Algoritmos Paralelos

O desenvolvimento de algoritmos paralelos é uma tarefa muito difícil, pois exige que o programador possua uma visão completa do problema, desde sua análise, implementação até a obtenção dos resultados. A metodologia apresentada neste trabalho estrutura o processo de desenvolvimento em quatro etapas: Particionamento, Comunicação, Aglomeração e Mapeamento, sendo ressaltado que, apesar da divisão, o desenvolvimento de um algoritmo paralelo deve ser visto como um todo, sempre observando os efeitos que a análise de uma etapa pode causar sobre as outras. As análises realizadas dão ênfase à concorrência, escalabilidade, localidade e outros temas relacionados com a eficiência do algoritmo. Dentro das etapas citadas, foram abordados tópicos como: técnicas de decomposição funcional e em domínios; métodos de comunicação globais, estáticos e dinâmicos, estruturados e não estruturados, síncronos e assíncronos; agrupamento de tarefas buscando diminuir custos de comunicação e implementação, bem como alguns algoritmos de balanceamento de carga e escalonamento de tarefas.

3. **Lucas Mello Schnorr**

Estudo e Classificação de Ferramentas de Gerenciamento de Clusters

Este trabalho tem como objetivo principal estudar e tentar classificar o gerenciamento de cluster como um todo. Nesta tentativa de classificação, observou-se que o gerenciamento pode ser dividido em três sub-áreas: escalonamento, monitoramento e configuração. O trabalho aprofunda a discussão nas duas primeiras sub-áreas, classificando as aplicações paralelas e os tipos de escalonamentos possíveis e analisando técnicas, arquiteturas e características na construção de monitoradores de cluster. O trabalho consiste também em uma análise de um conjunto de ferramentas de escalonamento, analisando-se as características que as diferem uma das outras, as suas arquiteturas e principalmente as suas características. Para cada ferramenta

de escalonamento estudada, que são CCS, PBS, Maui, Crono e Condor, tenta-se classificá-las de acordo com a proposta feita anteriormente. No fim do trabalho, algumas ferramentas de monitoramento são delineadas, dentre as quais podemos citar: Ganglia, Performance Co-Pilot, SCMS/RMS, Ka-Admin, Parmon, ClusterProbe, SIMONE e RVision. Na conclusão do trabalho apresenta-se a necessidade de se obter uma solução de gerenciamento integrada, que tenha sob si as três sub-áreas do gerenciamento.

4. **Rodrigo da Rosa Righi**

Recursos que a tecnologia Infiniband oferece para aglomerados de computadores e a interface de programação de aplicação VAPI

A seguinte palestra irá abordar o assunto que foi tema do trabalho individual elaborado pelo autor. A computação em aglomerados de computadores é cada vez mais utilizada para suporte a aplicações que necessitem de alto poder de processamento. Nessa arquitetura, os componentes do sistema, também chamados de nós, realizam a comunicação através da rede de interconexão, e dependendo da aplicação, estas operações podem ocasionar um gargalo de desempenho. Para melhorar o desempenho desse tipo de máquina paralela, existe pesquisa para minimizar o tempo de uma troca de mensagem. Nesse contexto, a arquitetura Infiniband propõe uma padronização de tecnologias de rede eficientes e realiza a transferência de dados com o sobrepasso do sistema operacional, podendo atingir uma largura de banda de até 30 Gbits/s. O presente trabalho descreve as capacidades da arquitetura Infiniband e os benefícios que esta pode oferecer para a arquitetura de aglomerados. Além disso, esse trabalho apresenta a interface de programação Infiniband VAPI, que foi escolhida para o desenvolvimento de aplicações em aglomerados.

5. **Marcelo Veiga Neves e Tiago Scheid**

Ganglia - Características de Projeto e Implementação

Ganglia é um sistema de monitoramento distribuído e escalável para sistemas computacionais de alta performance como clusters e grids. Ele é baseado em um modelo hierárquico focado em federações de clusters. Utiliza um protocolo listen/announce baseado em multicast para monitorar o estado interno do cluster e usa uma árvore de conexões ponto-a-ponto entre um nó representativo do cluster e a federação de cluster, agregando o seu estado. Alavancado por tecnologias amplamente usadas como XML para representação dos dados, XDR para transmissão de dados de forma portátil

e compacta, e RRDtool para armazenamento e visualização de dados. Usa estruturas de dados e algoritmos cuidadosamente projetados para ter um baixo over-head por nó e alta concorrência. A implementação é robusta, tem sido portada para um grande conjunto de sistemas operacionais e arquitetura de processadores e atualmente é usado por mais de 500 clusters no mundo. A apresentação mostrará características de projeto e implementação do Ganglia.

6. Antônio Marcos de Oliveira Candia

Criptanálise Distribuída utilizando ataques de dicionário e de força-bruta em Java

A criptanálise permite a recuperação da chave utilizada para uma cifragem de texto a partir do texto cifrado. Dentre os vários métodos existentes de criptanálise, os ataques do tipo força-bruta são os que possuem maior demanda computacional. Neste tipo de ataque, testa-se todas as possibilidades de chave até se encontrar a correta. Já nos ataques baseados em dicionário utiliza-se um banco de palavras a serem testadas. Como trata-se de uma busca por varredura, embora não utilize todo o espaço de soluções possíveis como o ataque por força-bruta, também acaba tornando-se computacionalmente dispendiosa. Estes são os métodos mais genéricos quanto à possibilidade de utilização contra qualquer algoritmo criptográfico. A aplicação proposta - quebra-pedra (QP) - utiliza-se destes métodos de criptanálise para oferecer uma ferramenta flexível aos criptoanalistas. O problema de desempenho é atacado através da utilização do mecanismo de execução distribuída, visto tratar-se de uma tarefa do tipo "embaraçosamente paralelizável". Por tratar-se de uma aplicação de computação forense, várias preocupações adicionais surgem durante seu projeto/execução do ponto de vista de segurança de dados. Além destas características, o desenvolvimento deste sistema pode ser tratado como um estudo de caso de utilização da linguagem Java em processamento de alto desempenho. A apresentação mostrará o estágio atual da aplicação e quais os caminhos atualmente visualizados para o seu desenvolvimento.

7. Claudio Schepke

Comparando o desempenho entre Java e C++ em aplicações numéricas

A computação científica é um dos grandes ramos da pesquisa. O uso da mesma está associado a simulações de diversos problemas nas áreas das ciências naturais e exatas. O paradigma de programação orientado a ob-

jetos é uma das opções disponíveis para a criação de programas voltados a solução de problemas modelados matematicamente. A abstração de um problema como um objeto simplifica a criação de um algoritmo, mas tem impacto negativo no desempenho. Mesmo assim, as linguagens que oferecem este paradigma possuem diferenças entre si, em relação a suas características, cabendo destacar a eficiência, algo importante no caso de aplicações científicas. Para verificar as diferenças reais existentes entre as linguagens orientadas a objeto, foi feita uma comparação entre os tempos de execução usando Java e C++, em uma implementação linear do problema de Laplace, utilizando o Gradiente Conjugado como método de solução. O uso de Java devido as suas características de programação tem se mostrado uma das alternativas, mas ainda deixa a desejar em relação a eficiência.

8. **Marcelo Veiga Neves e Tiago Scheid**

Monitoração de Aplicações Paralelas Utilizando Ganglia e Pajé

A monitoração de aplicações paralelas pode ser feita em vários níveis. Um primeiro nível pode ser a instrumentação do código e geração rastros de execução. Um segundo nível pode ser o acompanhamento da execução através da coleta de informações do sistema. A apresentação mostrará uma solução em monitoração de programas paralelos, a nível de sistema, desde a coleta até a visualização das métricas. Para coleta e centralização dos dados foi utilizada a ferramenta de monitoração distribuída Ganglia. Essa ferramenta é robusta, escalável e possui um baixo overhead. Ela mantém um histórico do estado do cluster em uma base de dados RRD (sistema que permite armazenar séries de dados temporais em um banco de dados circular). Como o Ganglia não foi projetado para monitorar processos específicos, foi necessária uma adaptação para permitir a coleta de informações do /proc sobre um determinado PID. Para tornar a monitoração mais representativa também foi necessário uma redução de intervalos de coleta. A visualização das métricas é feita utilizando a ferramenta Pajé. Essa ferramenta possui um formato próprio de rastros de entrada, que permite representar os dados de forma hierárquica. Para tanto é necessário fazer uma conversão do formato RRD para o formato dos rastros do Pajé.

9. **Rodrigo da Rosa Righi e Marcia Cristina Cera**

Características e facilidades do ProActive para a construção de aplicações e ferramentas Java

Java vem se destacando como uma das mais populares e difundida lin-

guagem de programação orientada a objetos. Ela apresenta um modelo de execução independente de plataforma, mecanismos para a operação com processos concorrentes e distribuídos, multithreading, RMI, entre outras características que despertam interesses na sua utilização em aplicações paralelas e distribuídas. Porém, o desempenho de aplicações Java deixa a desejar quando comparado com aplicações que utilizam outras linguagens de programação. Muitos esforços vem sendo empregados para se obter melhor desempenho em aplicações Java. Nesse contexto, esta inserido o pacote ProActive, que é uma biblioteca Java para computação paralela, distribuída e concorrente. Chamada de métodos assíncronos, espera por necessidade, mobilidade e migração, segurança, objetos remotos e polimorfismo são as principais características do ProActive. A forma como o ProActive foi estruturado para proporcionar tais características, como ele funciona e quais as vantagens que oferece é o enfoque principal da apresentação desse seminário. Esta apresentação também abordará a utilização dese pacote para a construção de duas ferramentas que estão sendo desenvolvidas no laboratório LSC.

10. **Lucas Mello Schnorr**

Visualização Simultânea e Multinível de Informações de Monitoramento de Clusters

O tema central do trabalho é monitoramento de clusters, isto é, a coleta e visualização de informações que permitam conhecer o comportamento dos componentes de um cluster e/ou de suas aplicações. A motivação do trabalho advém da diversidade de ferramentas de monitoramento que fornecem informações em diferentes níveis de detalhe. Existem situações onde as informações de uma única ferramenta não são suficientes ou onde elementos diferentes de um cluster (ou de um grid) usem monitores diferentes para fornecer a mesma informação. Nestes casos, deve-se usar mais de uma ferramenta de monitoramento para se ter acesso aos dados necessários para uma tomada de decisão. O uso de ferramentas independentes para a visualização de dados obtidos de fontes distintas prejudica a análise da correlação entre fenômenos observados em diferentes níveis. No desenvolvimento de aplicações paralelas as informações da execução podem não ser suficientes para se encontrar as razões de eventuais problemas do programa. Nesses casos informações externas à aplicação podem permitir que se encontre a causa mais rapidamente, facilitando o desenvolvimento de programas paralelos. Centralizar as informações dessas diferentes fontes, permitindo a visualização

correlata e dando a possibilidade de facilitar o encontro da tomada de decisão tanto aos administradores do cluster, quanto aos usuários desenvolvedores de aplicação, é a idéia e objetivo principal desse trabalho. Para verificar a validade desta idéia, propõe-se a construção de uma ferramenta protótipo capaz de agrupar dados de diferentes fontes de informação, como Ganglia, DECK e MPI e visualizá-las conjuntamente através da ferramenta de visualização Pajé. Farão parte da apresentação a defesa do trabalho de dissertação e os resultados no monitoramento do sistema operacional e da biblioteca de comunicação DECK e suas respectivas visualizações em Pajé.

11. **Rodrigo da Rosa Righi**

Implementação de um ambiente para invocação remota e assíncrona de métodos sobre infiniband e DECK

sem resumo

12. **Márcia Cristina Cera**

Alocação Dinâmica de Recursos Computacionais Ociosos em Java

Os recursos computacionais disponíveis em ambientes de ensino, pesquisa e outros, muitas vezes acabam sendo sub-utilizados. Computadores executam tarefas que demandam por grande quantidade de processamento apenas em determinados períodos e os mesmos permanecem ociosos em uma fração considerável de tempo. Esse artigo propõe um sistema capaz de alocar recursos computacionais em ociosidade para integrarem uma plataforma de execução de aplicações paralelas e distribuídas. Com esse sistema serão proporcionados meios para obter um melhor aproveitamento do poder de processamento de computadores.

13. **Cleverton Marlon Possani**

Uma Proposta de Monitoramento de Rede para Laboratórios de Informática

O gerenciamento dos computadores nas instituições de ensino, instituições governamentais e principalmente empresariais são tão importantes quanto a administração de recursos de borda da rede. Uma das funções dos administradores de redes é monitorar a rede, ver seu comportamento, suas limitações, seus defeitos, desta forma agir preventivamente contra uma possível parada de sistema. Podemos dizer que hoje "a rede é o próprio computador", não adianta ter os roteadores e switches funcionando corretamente se os servidores e as próprias estações de trabalho não estão. Uma proposta de monitoramento de rede será apresentada, utilizando ferramentas como SNMP, PHP, JGRAPH, APACHE e MySQL.

14. **Juliano Reckziegel**
Utilização de DECK como suporte para a comunicação em programas Java
Para o desenvolvimento de aplicações paralelas e distribuídas, observa-se a crescente utilização da linguagem Java. Ela oferece classes que implementam a troca de mensagens sobre soquetes TCP. Está em desenvolvimento uma biblioteca de adaptação chamada microVAPI que permite utilizar DECK como ambiente de comunicação para programas construídos com a linguagem Java.
15. **Diego Kreutz**
Um Calculador de Capacidade de Computação para Nós de Máquinas Virtuais LAM/MPI
sem resumo
16. **Lucas Mello Schnorr**
Visualização da biblioteca de comunicação DECK em Pajé
sem resumo
17. **Rodrigo Righi e Juliano Reckziegel**
Invocação Remota de Métodos de Alto Desempenho
sem resumo
18. **Edmar Pessoa Araujo Neto**
Visualização online de aplicações paralelas no Pajé
Pajé é uma ferramenta de visualização de rastros de aplicações em desenvolvimento no Laboratório de Sistemas de Computação da UFSM. Os rastros que ele visualiza são geralmente gerados durante a execução da aplicação e depois de seu término é possível visualizar o comportamento do tomado pelo programa. A visualização em linha permite que a aplicação seja analisada durante a execução, possibilitando uma captura de erros mais rápida. Este artigo descreve o processo de visualização em linha utilizando a ferramenta Pajé.
19. **Claudio Schepke**
Avaliação da Biblioteca JMP para Paralelização de Métodos Numéricos em Arquitetura com Memória Compartilhada
Ao longo do tempo, surgiram várias bibliotecas que facilitam a programação de aplicações envolvendo a resolução de sistemas de equações lineares através

do uso de métodos numéricos. O desenvolvimento desse tipo de biblioteca vem acompanhando a evolução, tanto das arquiteturas de computadores, como das linguagens e paradigmas de programação. Neste contexto, observa-se um interesse crescente na utilização de linguagens orientadas a objetos, e particularmente da linguagem Java, para o desenvolvimento de bibliotecas numéricas tanto seqüenciais como paralelas. O uso do processamento paralelo é uma das alternativas para diminuir os tempos de execução na resolução de sistemas cujas matrizes possuam uma grande ordem. Um exemplo recente neste cenário é a biblioteca JMP (Sparse Matrix Library in Java), desenvolvida por Bjørn-Ove Heimsund da Universidade de Bergen (Noruega), que engloba todos esses conceitos. Este trabalho tem como principal objetivo avaliar os recursos da biblioteca JMP para a paralelização de métodos numéricos em uma arquitetura com memória compartilhada. Para isso, serão utilizadas duas aplicações, a solução de um sistema linear e a multiplicação entre uma matriz por um vetor, sendo os tempos de execução medidos para cada caso.

20. **Pablo Furlan da Silva**

Desenvolvimento de um Módulo de Visualização Não Temporal de Dados para o Programa de Visualização Interativa de Aplicações Paralelas - Pajé

O corrente projeto tem como objetivo o desenvolvimento do módulo de visualização quantitativa de dados que permitirá uma análise, de um programa paralelo, em um diagrama não temporal. Esse módulo será posteriormente integrado a ferramenta Pajé, a fim de complementar a forma de análise existente na ferramenta. Este módulo fornecerá uma visão global da execução da aplicação e permitirá ao programador observar a dimensão real do desempenho de seu programa. Isto será possível devido à interatividade presente no módulo durante a escolha dos dados a serem visualizados e o formato de gráfico a ser utilizado.

21. **Geovani Ricardo Wiedenhof**

Rastreamento de programas Java usando JVMTI

A utilização da linguagem Java no desenvolvimento de aplicações paralelas que demandam alto desempenho é crescente. Naturalmente, surge a necessidade de meios para a depuração do código Java desenvolvido. Essa depuração é mais complicada utilizando um depurador de aplicações Java sequenciais quando o código Java possui fluxos de execução diferentes. Uma

forma de auxiliar a depuração dessas aplicações é a geração de rastros durante a execução de uma aplicação Java para posterior visualização. Java oferece JVMTI, uma interface para recuperação dos principais eventos gerados pela JVM. Através dessa interface pode-se realizar o monitoramento e registro de eventos, necessários para a geração de rastros de execução. Esse trabalho descreve a utilização da libRastro para o registro dos eventos monitorados através da JVMTI assim como a conversão dos eventos registrados para o formato da ferramenta de visualização genérica Pajé. A apresentação mostrará a evolução da biblioteca jRastro e a antiga interface JVMPI que irá deixar de ser incluída nas novas versões do Java. Bem como, as diferenças entre as interfaces.

22. Andre Schmidt Dellamea

Implementação de um sistema distribuído em estrutura hierárquica para suporte para metacomputação

Na área de processamento paralelo, a centralização de serviços apresenta potencial para ser um fator de sobrecarga para um sistema distribuído. Considere, por exemplo, que todos os computadores pertencentes a um sistema distribuído dependam de um único computador servidor. Conforme aumenta o número dos componentes do sistema distribuído, aumenta a comunicação entre essas componentes e o servidor, além do próprio processamento do servidor devido aos serviços oferecidos. A sobrecarga do processamento do servidor e da comunicação na rede pode afetar negativamente o desempenho do sistema distribuído. A idéia geral da metacomputação é abstrair as componentes do sistema distribuído, de modo que todo o sistema seja visto como um único computador. Nessa área, utiliza-se o conceito de tarefas e de recursos computacionais. As tarefas correspondem a atividades que devem ser realizadas, e os recursos computacionais correspondem aos recursos disponíveis e que serão utilizados para a realização de uma determinada tarefa. Um exemplo é a realização de uma integral complexa, que corresponde a uma tarefa, em um computador com mais de 32Mb de RAM, que corresponde aos recursos computacionais requeridos. A combinação entre tarefas requisitadas e recursos computacionais disponíveis é chamada de escalonamento. A combinação entre diferentes tarefas e diferentes recursos computacionais disponíveis, de forma eficiente, é uma área em constante pesquisa. A maioria das implementações de sistemas distribuídos que seguem a filosofia da metacomputação baseiam-se em sistemas centralizados, o que afeta o seu desempenho. O objetivo da palestra é propor a implementação de um sis-

tema distribuído que forneça suporte a metacomputação em um maior nível de abstração, utilizando uma estrutura descentralizada e hierárquica.

23. **Elton Nicoletti Mathias e Marcelo Veiga Neves**

Balanceamento de Carga e Detecção de Ociosidade em Sistemas Distribuídos

A utilização de recursos computacionais distribuídos tornou-se muito difundida na área do processamento de alto desempenho. Para uma melhor utilização da capacidade desses recursos torna-se necessário o uso de mecanismos de controle para alocação dos mesmos. Uma das formas mais adotadas é o balanceamento de cargas mediante coleta de índices para a tomada de decisão. A qualidade do balanceamento está intimamente ligada a esses índices de carga, que devem refletir, ao máximo, a realidade do estado atual do sistema. Para uma tomada de decisão mais apurada outras otimizações podem ser levadas em conta, como a tomada de decisão baseada no tipo de aplicação que ocupará o sistema e a utilização de técnicas para a predição de períodos ociosos. Esta apresentação abordará os conceitos de ociosidade e estratégias para a predição de períodos de ociosos, bem como a arquitetura de um detector de ociosidade hipotético. Além disso, serão abordados os principais índices de carga e as técnicas usuais de balanceamento de carga.

24. **Juliano Foletto Reckziegel e Rodrigo Righi**

Aldeia Sockets

A linguagem de programação Java oferece a abstração de Sockets através de um conjunto de classes bem definido. Tal conjunto de classes permite comunicação TCP/IP entre computadores. Contudo, tal protocolo impõe algumas penalidades de software, que não seriam necessárias em um ambiente que priorize alto desempenho. Com base nesse contexto, foi desenvolvido um conjunto de software chamado AldeiaSockets. AldeiaSockets permite comunicação em tecnologias de rede suportadas pelas bibliotecas de comunicação DECK e VAPI. Dessa forma, é possível realizar a troca de mensagem em redes de alta velocidade, como Myrinet e Infiniband. Essa apresentação mostra o Aldeia Sockets, a sua estrutura e seu funcionamento, e as vantagens que ele pode oferecer se utilizado junto com as demais classes de E/S presentes nas distribuições padrão do Java.

25. **Marcia Pasin**

Alta disponibilidade para servidores de aplicação Enterprise JavaBeans

A especificação Enterprise JavaBeans (EJB) descreve serviços nãofuncionais de segurança, de gerenciamento de transações e de persistência em bancos de

dados para aplicações distribuídas, mas não descreve serviços que garantam alta disponibilidade. Neste trabalho, alta disponibilidade é oferecida como uma nova propriedade para as aplicações EJB através da adição de serviços não-funcionais usando abstrações de comunicação de grupo. Os serviços para alta disponibilidade são oferecidos através de uma arquitetura de múltiplas camadas. Esses serviços incluem gerenciamento de réplicas, chaveamento de servidor, gerenciamento de membros do grupo e detecção de membros falhos do grupo. A combinação de conceitos de bancos de dados com comunicação de grupo demonstra uma interessante solução para aplicações com requisitos de alta disponibilidade, como as aplicações EJB. Os serviços adicionais da arquitetura em múltiplas camadas foram implementados em um protótipo. A validação através de um protótipo possibilitou que experimentos fossem conduzidos dentro de um ambiente controlado, usando diferentes cargas de trabalho sintéticas.

26. **Claudio Schepke e Andrea Schwertner Charão**

Implementação Paralela do Método do Gradiente Conjugado Utilizando MPIJava

Java vem se destacando como uma linguagem na qual a representação de problemas científicos pode ser facilmente modelada através da orientação a objetos. Por ser simples, robusta e multiplataforma, ela pode ser considerada como uma linguagem especialmente desenvolvida para a computação distribuída, não obstante ela ser utilizada também para a computação de alto desempenho. Neste contexto, a comunicação entre processos é um dos fatores que necessita de uma atenção especial.

Além da concorrência disponibilizada pelo uso de threads, sockets e RMI é possível fazer uso de bibliotecas de comunicação específicas para processamento paralelo em clusters. Esta abordagem é bem atendida nas linguagens de programação C, C++ e Fortran com o uso de MPI. Já para Java existe a interface MPIJava. MPIJava permite a troca de informações via MPI, através de chamadas de métodos nativos via JNI, buscando disponibilizar as características de comunicação da biblioteca para programas em Java de maneira eficiente.

O presente trabalho visa apresentar uma implementação paralela do método do Gradiente Conjugado usando MPIJava em um aglomerado de computadores. Desta forma, pretende-se validar as funcionalidades da biblioteca, bem como apresentar os resultados obtidos com a utilização da implementação.

27. **Marcelo Veiga Neves, Tiago Scheid, Lucas Mello Schnorr e**

Andrea Charão

Integração de Ganglia, libRastro e Pajé para o Monitoramento de Aplicações Paralelas

Este artigo trata do uso integrado de diferentes ferramentas de monitoramento a fim de aprimorar a capacidade de análise das execuções de aplicações paralelas. Em particular, descreve-se o processo de integração dos dados coletados por Ganglia, que é uma ferramenta para monitoramento de clusters, aos rastros de execução gerados por libRastro, que é uma biblioteca para instrumentação de aplicações paralelas. A visualização dos dados integrados é feita com a ferramenta Pajé. Através de alguns exemplos de visualizações integradas, demonstra-se que as informações sobre o estado do cluster complementam os rastros de execução da aplicação, permitindo inclusive detectar eventuais problemas na execução da aplicação.

28. **Rodrigo Righi, Philippe Navaux e Marcelo Pasin**

Sistema Aldeia: Invocação Remota e Assíncrona de Métodos sobre Infiniband e DECK

A linguagem Java é cada vez mais utilizada para a construção de aplicações. O próprio sistema de invocação remota de métodos (RMI) da linguagem Java proporciona a escrita de aplicações distribuídas, permitindo comunicação através de TCP/IP entre computadores. Entretanto, tal protocolo impõem penalidades de software para a obtenção de alto desempenho na comunicação. Além disso, Java RMI realiza a comunicação de maneira síncrona, o que pode também contribuir para o decréscimo da eficiência de aplicações escritas com esse sistema. Visando o uso de Java para a programação de alto desempenho em aglomerados, está em desenvolvimento o sistema Aldeia. Ele possibilita a invocação remota e assíncrona de métodos sobre as interfaces de rede Infiniband e DECK. Esse artigo descreve a estrutura do sistema Aldeia, as tecnologias e as bibliotecas utilizadas para a sua confecção.

29. **Diego Kreutz, Lucas Mello Schnorr e Marlon Possani**

Comportamento de Aplicações Paralelas em Aglomerados de Computadores Heterogêneos

A computação em aglomerados heterogêneos de computadores está cada vez mais presente na área de computação de alto desempenho. Neste contexto, o objetivo desse trabalho é apresentar e analisar alguns dados de desempenho de aplicações paralelas em aglomerados desse gênero. Além disso, apresenta

um computador de capacidade de computação para nós de processamento de máquinas virtuais LAM/MPI. Com ele o usuário pode facilmente estabelecer um balanceamento de cargas para a sua aplicação, considerando as principais características da mesma.

30. **Claudio Schepke, Andrea Schwertner Charão**

Bibliotecas Java para o Cálculo Numérico

A linguagem de programação Java vem há algum tempo suscitando o interesse da comunidade ligada à computação científica. Isso se deve, em grande parte, às vantagens da orientação a objetos, aliadas à portabilidade das aplicações escritas nesta linguagem. Para o cálculo numérico, em particular, a orientação a objetos permite a programação em um nível de abstração bastante próximo das formulações algébricas. Para a codificação de algoritmos de álgebra linear, existem atualmente diversas bibliotecas de classes que permitem abstrair as operações de mais baixo nível, implementando recursos não existentes na linguagem, sanando assim, algumas das suas deficiências. Este trabalho tem como objetivo analisar algumas das bibliotecas Java que se aplicam à álgebra linear e, particularmente, que oferecem suporte a métodos numéricos computacionais. Para tanto, foram feitas avaliações dos principais pacotes disponíveis atualmente. Os critérios de avaliação consideraram essencialmente os tipos de problemas tratáveis, a existência de classes especiais como números complexos, matrizes e vetores, as funcionalidades para leitura e escrita de dados em formatos padrões, a paralelização de métodos e, também, alguns aspectos relativos ao suporte à biblioteca, incluindo documentação, facilidade de acesso ao código-fonte e atualização do mesmo.

31. **Elton Mathias**

Detecção de Ociosidade em Sistemas Distribuídos

O uso de sistemas distribuídos como suporte a aplicações que demandam alto desempenho tem se tornado uma forte tendência. Nesse tipo de ambiente, a existência de uma ocupação homogênea dos recursos favorece o desempenho do sistema como um todo. Para uma utilização mais eficiente da capacidade desses ambientes distribuídos, torna-se necessária a utilização de ferramentas que controlem a sua ocupação. Para um bom funcionamento desse tipo de ferramenta, pode-se utilizar algoritmos de detecção e predição de ociosidade que auxiliem na tomada de decisões relativas ao balanceamento de carga. Esta apresentação abordará a implementação de um detector de ociosidade baseado na coleta de índices de cargas e uma análise dos algoritmos de

predição por ele utilizados.

32. **Tiago Scheid**

Comparação entre ferramentas de monitoração de clusters

Atualmente há uma ampla utilização de aglomerados de computadores (clusters) no processamento de alto desempenho. Para facilitar o gerenciamento de um cluster, é geralmente necessário monitorar a utilização de seus recursos. A monitoração consiste em observar o estado do cluster em um dado momento ou em momentos passados, através de um histórico. Hoje em dia existem várias ferramentas para monitorar-se um cluster, permitindo obter as mais diferentes informações e visualizá-las de diversas maneiras. Nesse contexto, esta apresentação busca expor a análise de algumas ferramentas de maneira aprofundada e, finalmente, um comparativo entre elas.

33. **Guilherme Piegas Koslovski**

Estudo de Técnicas para Melhora do Desempenho de Aplicações Java

Java vem sendo cada vez mais utilizada por ser uma linguagem de programação simples e flexível. Ela segue o modelo de programação orientado a objetos apresentando características como herança, polimorfismo, reusabilidade de código, portabilidade, limpeza de lixo automática, entre outras. Além de ser utilizada na implementação de programas seqüências, Java também vem sendo utilizada na programação paralela e distribuída. Isso porque ela oferece suporte nativo à programação com múltiplos fluxos de execução (multithreading) e com memória distribuída.

A fim de proporcionar flexibilidade e portabilidade, os códigos fonte Java podem ser traduzidos para uma arquitetura neutra chamada de bytecode. Um arquivo de bytecode pode ser executado em qualquer plataforma através de uma implementação da Máquina Virtual Java (JVM), capaz de interpretar o bytecode. Em contrapartida, as facilidades agregadas pelo paradigma orientado a objetos, processo de interpretação de bytecodes, e coleta automática de lixo acabam por prejudicar o desempenho final de aplicações Java.

O objetivo desse trabalho é apresentar algumas iniciativas que buscam proporcionar ganho de desempenho para aplicações Java. A fim de testar a eficiência de algumas dessas iniciativas, foram implementadas duas aplicações com características distintas, a partir das quais será possível identificar situações onde tais iniciativas podem ser eficazes.

34. **Edmar Pessoa Araujo Neto**

Comparação de Desempenho Entre Objetos Distribuídos, RMI e RMI

assíncrono

O desenvolvimento de algoritmos distribuídos é mais complexo que o de algoritmos seqüenciais. Por isso, é comum que os desenvolvedores busquem maneiras para abstrair essa complexidade. Algumas linguagens orientadas a objetos como Java e Objective-C possuem ambientes de desenvolvimento que facilitam ao programador a distribuir ou paralelizar suas aplicações de forma simples. A linguagem Objective-C possui um ambiente de desenvolvimento de aplicações chamado GNUstep, que trata a comunicação entre vários fluxos de execução através da abordagem de Objetos Distribuídos. Já a linguagem de programação Java possui um pacote que implementa a invocação de métodos remotos (RMI - Remote Method Invocation). Este trabalho apresenta uma comparação de desempenho na troca de mensagens através de rede entre Objetos Distribuídos e a invocação remota de métodos RMI.

35. **Antonio Marcos de Oliveira Candia**

Criptanálise Distribuída de Alto Desempenho

A criptoanálise computacional pode ser definida como o uso do computador para o ataque a um texto criptografado na tentativa de decifrá-lo. Quebra-pedra é um arcabouço Java para criptoanálise computacional que oferece as técnicas de ataque por força-bruta e busca dirigida por dicionário e foi projetado em torno de dois princípios básicos: independência de plataforma de execução e independência de algoritmos de criptografia. Os sistemas de criptoanálise atuais não são facilmente adaptáveis a algoritmos diferentes daqueles para os quais foram projetados. Além disso, a linguagem Java, até a versão 1.4.2, é freqüentemente citada por seu baixo desempenho em aplicações computacionalmente intensivas. No arcabouço, estes problemas foram atacados com o uso de dois conceitos disponíveis na plataforma Java: o código reflexivo e a interface nativa Java (Java Native Interface - JNI). O uso destes mecanismos permite a carga dinâmica de código nativo sem a necessidade de recompilação do sistema de criptoanálise. A programação da versão colaborativa de um aplicativo que utilize o Quebra-pedra para criptoanálise distribuída está sendo realizada através da biblioteca ProActive. Essa biblioteca faz parte do projeto ObjectWeb Middleware. Resultados preliminares mostram que o ganho de desempenho é semelhante aos resultados obtidos com o uso de Sockets. As vantagens de utilizar o ProActive incluem: um mecanismo transparente de migração de tarefas de processamento, programação simplificada, e o fato do mesmo ser escrito totalmente

em Java, permitindo sua portabilidade.

4 Trabalhos do ano de 2003

Esta seção inicialmente lista os trabalhos realizados durante o ano de 2003. Depois, são mostrados os resumos de cada trabalho, que detalham o assunto e o que foi feito. Os resumos foram desenvolvidos pelos seus autores.

Os trabalhos apresentados em 2003, pelo Laboratório de Sistemas de Computação, são:

1. **Claudio Schepke e Geovani Wiedenhof**t, em 27/05/2003, apresentaram: *Arquiteturas de Computadores Paralelos*
2. **Marcelo Veiga Neves e Elton Nicoletti Mathias**, em 06/2003, apresentaram: *Projetando Algoritmos Paralelos*
3. **Marcelo Pasin**, em 16/09/2003, apresentou: *Introdução à Metodologia Científica*
4. **Edmar Araujo Neto, Geovani Ricardo Wiedenhof**t e **Pablo Furlan da Silva**, em 23/09/2003, apresentaram: *Rastreamento e Visualização de um Programa Paralelo para a Aplicação de Filtros Morfológicos em Imagens*
5. **André Schmidt Dellamea**, em 30/09/2003, apresentou: *Metacomputação*
6. **Elton Nicoletti Mathias**, em 04/11/2003, apresentou: *Programação de Alto Desempenho utilizando RMI Assíncrono*
7. **Marcelo Veiga Neves**, em 18/11/2003, apresentou: *Ferramentas para Administração de Aglomerados*
8. **Claudio Schepke**, em 25/11/2003, apresentou: *Resolução de Sistemas Lineares em Matrizes Esparsas*
9. **Claudio Schepke**, em 02/12/2003, apresentou: *Algoritmo Paralelo de Busca de Ponto Mínimo*

4.1 Resumos

1. Claudio Schepke e Geovani Wiedenhof

Arquiteturas de Computadores Paralelos

A arquitetura de um computador é o modo como o hardware é estruturado. Os aspectos observados na arquitetura geram diferentes classificações. Uma das possíveis classificações está relacionado a distribuição de memória, que pode ser compartilhada, distribuída ou compartilhada distribuída. Já a classificação de Flynn se baseia no fluxo de instruções e no fluxo dos dados. Uma das possibilidades de se obter computadores paralelos está na utilização de aglomerados. Aglomerados fazem uso de diversos computadores interconectados por uma rede. A configuração desta rede pode ser feita em diversas topologias, como linha, anél, mesh, árvore e hipercubo. As arquiteturas paralelas apresentam diversos termos e definições que analisam o tempo de comunicação entre dois computadores, como o acréscimo do tempo de comunicação ao tempo de processamento, o poder de processamento relacionado a uma máquina sequencial e o desempenho relativo entre o uso de um determinado número de computadores para a solução de um problema.

2. Marcelo Veiga Neves e Elton Nicoletti Mathias

Projetando Algoritmos Paralelos

A maioria dos problemas computacionais tem várias soluções paralelas, e a melhor solução pode diferir daquela sugerida pelo algoritmo sequencial. A apresentação trata de uma metodologia de projeto de algoritmos paralelos, criada por Ian Foster, estruturada em 4 fases distintas. Primeiramente o problema em questão é particionado em pequenos pedaços, ou tarefas. Depois disso, temos de organizar estruturadamente a comunicação requerida entre as partes. Então, a fim de minimizar custos de comunicação, passamos à terceira fase, que visa aglomerar grupos de tarefas onde há maior comunicação. Finalmente, o projeto encerra-se com um mapeamento de tarefas à processadores com o objetivo de minimizar o tempo e execução total. Nesta fase final também procede-se com técnicas de escalonamento de tarefas e balanceamento de cargas entre os nós que executarão tal algoritmo. O objetivo final deste processo é apresentar um meio sistemático de construir programas capazes de criar e destruir tarefas dinamicamente, usando técnicas de balanceamento de cargas, a fim de explorar ao máximo a capacidade de processamento concorrente de uma máquina paralela.

3. **Marcelo Pasin**

Introdução à Metodologia Científica

A apresentação é uma introdução à Metodologia Científica, onde são mostrados os seus elementos básicos e suas partes ou etapas. São apresentados como elaborar um projeto de pesquisa, como redigir um trabalho científico (artigos, relatórios, livros ou capítulos) e como fazer uma apresentação oral.

4. **Edmar Araujo Neto, Geovani Ricardo Wiedenhof e Pablo Furlan da Silva**

Rastreamento e Visualização de um Programa Paralelo para a Aplicação de Filtros Morfológicos em Imagens

Com o uso dos métodos de paralelização de programas, ampliou-se a dificuldade de depuração e otimização de aplicações. Um meio utilizado para auxiliar no desenvolvimento de programas é o rastreamento. Os rastros são os registros dos eventos ocorridos durante a execução do programa. Assim, os rastros de uma execução possibilitam observar os acontecimentos ocorridos durante o decorrer da aplicação, favorecendo a verificação de erros e abrindo caminho para o melhoramento e correção do programa. Esta apresentação mostra o processo de rastreamento de um programa paralelo para a aplicação de filtros sobre imagens ruidosas. O objetivo é mostrar os processos de rastreamento e visualização de aplicações paralelas, utilizando a biblioteca de rastreamento libRastro e a ferramenta de visualização Paje.

5. **André Schmidt Dellamea**

Metacomputação

Atualmente, existe uma demanda crescente da utilização cada vez maior de mais recursos computacionais. Esses recursos computacionais podem ser processadores, memórias RAM, discos rígidos, impressoras, periféricos e qualquer tipo de componente que pode ser acoplado em um computador. Ao mesmo tempo, os computadores atuais são bastante ociosos quanto ao uso de seus próprios recursos computacionais. O princípio teórico da metacomputação é a criação de um computador virtual, que, de forma transparente aos usuários, é formado através do gerenciamento dos recursos computacionais dos computadores conectados. Esses computadores, juntamente com os seus recursos, formam fisicamente o computador virtual. Por exemplo, considere que existam quatro computadores que, individualmente, possuam capacidade de memória RAM de 100Mb e um único processador. Esses computadores formam, então, um computador virtual que possui 400Mb

e quatro processadores. Dessa forma, na teoria e segundo as percepções dos usuários, eles estão manipulando um computador real de 400Mb e com quatro processadores. A abstração do gerenciamento dos recursos computacionais possui como consequência que o metacomputador, em algum nível de implementação, deve associar recursos computacionais com a solicitação dos usuários do uso desses recursos. Essa solicitação é chamada de tarefa ou trabalho. Cada trabalho possui, necessariamente, a necessidade de um ou mais recursos computacionais para ser resolvido. O modo como é suprida a demanda dos trabalhos que são exigidos pelos usuários, conforme os recursos computacionais disponíveis, é chamado de balanceamento de carga ou escalonamento.

6. Elton Nicoletti Mathias

Programação de Alto Desempenho utilizando RMI Assíncrono

Graças à sua simplicidade, segurança e independência de arquitetura e SO, a linguagem de programação Java emergiu como uma poderosa ferramenta para a programação em ambientes distribuídos. A fim de facilitar a programação de tais sistemas a Sun Microsystems lançou o RMI, uma interface de chamada remota de métodos, que apresenta protocolos bem definidos possui fácil utilização. Entretanto, embora RMI seja de uso bastante simples, em determinadas aplicações, fica afetado o desempenho, devido à natureza assíncrona, que esta interface apresenta. Como alternativa para a implementação de aplicações paralelas, utilizando-se de mecanismo de RPC (Remote Procedure Call) em java, propõe-se a implementação de uma camada, em java, sobre o RMI original que oferece uma interface assíncrona de chamada de métodos remotos. A apresentação mostra primeiramente conceitos a respeito de RMI, seguindo-se com uma apresentação da estrutura e interface da camada que oferece o assíncronismo, finalizando com uma análise comparativa de desempenho entre o RMI original, da SUN, e essa, assíncrona, para dois aplicativos: um que efetua busca por números primos e o outro um filtro de imagem baseado em matrizes de convolução.

7. Marcelo Veiga Neves

Ferramentas para Administração de Aglomerados

Para a obtenção de alto desempenho em aglomerados é necessária a utilização de diversas ferramentas, as quais possuem como objetivo principal facilitar o gerenciamento da máquina paralela e solucionar os principais problemas de administração. Entre as principais funções dessas ferramentas estão:

automatização da instalação do sistema operacional e configuração e atualização de programas nos nós. Também é necessária uma forma de gerenciar os recursos disponíveis dividindo-os entre os usuários sem que nenhum deles seja prejudicado e ao mesmo tempo não haja desperdício. Outra função importante é a monitoração da máquina paralela. A monitoração provê uma visão global da máquina facilitando a identificação de possíveis falhas e gargalos. Esta apresentação tenta classificar as ferramentas, descrevendo os principais problemas em maiores detalhes e aponta as soluções enumerando as ferramentas disponíveis.

8. Claudio Schepke

Resolução de Sistemas Lineares em Matrizes Esparsas

Diversos problemas físicos podem ser resolvidos fazendo uso do poder computacional. Para tanto é necessário que eles sejam modelados matematicamente, para uma posterior discretização. Alguns destes problemas são resolvíveis por equações diferenciais parciais, através de métodos numéricos. Exemplos deste tipo de problema podem ser encontrados na termodinâmica, na distribuição de forças e cargas como também na dinâmica de fluidos. Para o processo de discretização são utilizadas diversas técnicas que possibilitam a geração de malhas. Já existem também bibliotecas que facilitam esta tarefa como a Triangle e a EasyMesh. Uma das questões referentes a este trabalho está na estrutura de dados para o armazenamento das informações, que darão geração a matriz de equações. Já para a aplicação de métodos numéricos existem duas possibilidades: os métodos diretos, que encontram sempre uma solução exata, e os métodos iterativos, geralmente aplicados em grandes sistemas, com o uso de computadores para a geração da solução. Quando o sistema possui uma grande quantidade de variáveis nulas, um método bastante eficiente é o método do Gradiente Conjugado, que explora as propriedades das matrizes simétricas, positivas e definidas, apresentando uma rápida convergência.

9. Claudio Schepke

Algoritmo Paralelo de Busca de Ponto Mínimo

Uma das opções existentes para a busca de soluções de um problema está no uso de heurísticas. Heurísticas buscam encontrar uma solução ótima em um tempo computacional aceitável, mesmo que a solução não seja a melhor. Esses algoritmos são aplicados em problemas de balanceamento de cargas, escolha de rotas que minimizem o custo ou tempo de transporte, sequencia-

mento genético, entre outros casos. Mesmo com o uso de heurísticas existem casos em que o tempo de processamento de um método é muito alto devido a dimensão do problema. Para solucionar este caso é possível fazer uso do processamento paralelo. Para verificar esta opção foi criado um algoritmo que busca a melhor solução na vizinhança de um ponto escolhido aleatoriamente. Cada processador é responsável por uma sub-região, informando a solução encontrada a um processador mestre que decide qual das soluções é a melhor. O algoritmo pode ser utilizado com qualquer número de processadores, apresentando pouco melhoramento do desempenho a partir de um determinado número de processadores, dependendo do problema, devido ao aumento de comunicação. A implementação foi escrita em na linguagem C utilizando a biblioteca de comunicação MPICH-1.2.5.2.