Escola Supercomputador SDumont 2020

O Supercomputador SDumont atende hoje cerca de 170 projetos de pesquisa pertencentes a 18 áreas de conhecimento, liderados por instituições de pesquisa de 12 estados brasileiros. A escola tem como objetivo prover à comunidade de usuários do Sdumont e à comunidade de programação em computação de alto desempenho em geral, minicursos relacionados com programação em computadores de alto desempenho tais como modelos de programação paralela, ferramentas de perfilagem e bibliotecas para o desenvolvimento de algoritmos paralelos otimizados.

A edição da escola de verão 2020 irá oferecer 20 minicursos. Dentre os minicursos a serem oferecidos, 5 minicursos serão oferecidos por docentes de instituições colaboradoras, 2 mini cursos serão oferecidos pela NVIDIA (https://developer.nvidia.com/developer-program) e um minicurso será oferecido pela Mellanox.(https://www.mellanox.com/) . Além dos minicursos, a escola irá apresentar um seminário oferecido pela ATOS (https://atos.net/en/) e dois seminários oferecidos pelo Pittsburgh Supercomputing Center (https://www.psc.edu/)

Metodologia: Minicursos com "hands-on" e seminários

Critérios para inscrição: Nível de instrução acadêmica: graduandos, pós-graduandos (mestrando e doutorando), pós-doutorandos ou pesquisadores. Atuação profissional do candidato: desenvolvimento de programas paralelos.

DOCENTES LNCC:

Roberto Souto (LNCC / http://buscatextual.cnpq.br/buscatextual/visualizacv.do?id=K4794811E7), Prof. Luis Gadelha (LNCC/http://buscatextual.cnpq.br/buscatextual/visualizacv.do?id=K4763360T6),
Carla Osthoff (LNCC/http://buscatextual.cnpq.br/buscatextual/visualizacv.do?id=K4780113Z3),
Amanda Sabatini (LNCC/http://buscatextual.cnpq.br/buscatextual/visualizacv.do?id=K4559384T8),
Frederico Carbal (LNCC/http://lattes.cnpq.br/6161165744422395)
Raquel Coelho (LNCC/INCA http://lattes.cnpq.br/6298932858686721)
André Carneiro (LNCC/ http://buscatextual.cnpq.br/buscatextual/visualizacv.do?id=K4111964Z3), Bruno Fagundes (LNCC/http://buscatextual.cnpq.br/buscatextual/visualizacv.do?id=K4420386H6),

Docentes Colaboradores:
Mateus Serpa (UFRGS http://lattes.cnpq.br/3041231438928613)
Pedro Pais Lopes (Exaflop http://lattes.cnpq.br/6240387512435840)
Esteban Meneses / Universidade de Costa Rica/Costa Rica)
Pedro Mario Cruz e Silva (NVIDIA / http://lattes.cnpq.br/6273068677376442)
Gilad Shainer (Mellanox)
John Urbanic (Pittsburgh Supercomputing Center)

EMENTAS DOS MINICURSOS

Modulo 1- Introdução ao Ambiente de programação no SDUMONT

M1-I - Introdução ao ambiente SDUMONT/SLURM e Ferramentas BULLX-DE
Carga Horária: 4h
Professores: Roberto Pinto Souto (LNCC), André Ramos Carneiro (LNCC), e Bruno Alves Fagundes (LNCC),

Resumo:
Visão geral do ambiente computacional do supercomputador Santos Dumont, mostrando os principais aspectos com relação aos recursos de hardware e de software disponíveis aos usuários.
Visão geral das ferramentas de avaliação de desempenho disponível aos usuários do supercomputador Santos Dumont, mostrando aspectos que visam obter um melhor entendimento da execução da aplicação, obtendo-se perfil de desempenho, que podem orientar no sentido de otimização do código.
Objetivos:
Este curso contemplará a programação paralela heterogenia em arquiteturas de computadores de memória distribuída, com enfoque na área de computação científica. Serão abordados tópicos como conceitos fundamentais de paralelismo, arquiteturas do sistema Santos Dumont, técnicas de otimização e estudo de casos. É também objetivo do curso a aplicação prática em laboratório dos conhecimentos de programação paralela adquiridos pelos alunos.
Ementa:
Perfil de desempenho de aplicação obtido com BullxProf,
HPCTookit,
Scalasca,
Open|SpeedShop,
MPIAnalyser,
xPMPI;
medição de contadores de hardware com a bibliotca PAPI.
Configuração dos nós computacionais;
arquiteturas paralelas disponívies (CPU, MIC e GPU)
sistema operacional Linux RedHat;
sistema de gerenciamento de recursos SLURM: principais comandos, submissão e monitoramento de jobs, políticas de submissão no SDUMONT;compiladores, ambientes paralelos de programação distribuída(BullxMPI, Intel MPI),
carregando bibliotecas computacionais (module/source).
Bibliografia:
1.Evaluating scalability and efficiency of the Resource and Job Management System on large HPC Clusters, Yiannis Georgiou (BULL S.A.S, France); Matthieu Hautreux (CEA-DAM, France) (16th Workshop on Job Scheduling Strategies for Parallel Processing, May 2012)
2.Contributions for Resource and Job Management in High Performance Computing, Yiannis Georgiou, Universite Joseph Fourier (Thesis, December 2010
1. Parallel Programming by Thonas Rauber and Gudula Runger. Springer 2010.
3. Using MPI Portable parallel Programming with the Message-Passing Interface by Willian Gropp, Ewing Lusk and Anthony Skjllum. 2014
4. https://www.dkrz.de/pdfs/docs/docu-mistral/bullx_scs_4_r4_de_2014-01.pdf?lang=de
5. http://hpc-support.lboro.ac.uk/bullxprof.html
3. Parallel Programming by Thonas Rauber and Gudula Runger. Springer 2010.
4. https://slurm.schedmd.com/

M1-II Introdução à E/S Paralela no SDUMONT
Carga Horária: 4h
Professores: André Ramos e Bruno Fagundes (LNCC)

Resumo:
Visão geral do Sistema de Arquivos Paralelo Lustre e como está configurado no SDumont, apresentando as melhores práticas na utilização do Lustre e uma visão geral sobre o MPI-IO.
Objetivos:
Este curso contemplará a programação paralela em arquiteturas de computadores de memória distribuída com arquivos paralelos Lustre, com enfoque na área de computação científica. Serão abordados tópicos como conceitos fundamentais de paralelismo de dados, arquiteturas de sistemas de arquivos paralelos, técnicas de otimização e estudo de casos. É também objetivo do curso a aplicação prática em laboratório dos conhecimentos de programação paralela adquiridos pelos alunos.
Ementa:
Arquitetura de um Sistema de Arquivos Paralelo Lustre;
Funcionamento do Lustre no SDumont;
Parâmetros e Utilização do Lustre;
Introdução ao MPI-IO;
Exemplos
Referências:
1. High performance Parallel I/O by Prabhat and Quincey Koziol. Chapman & Hall/CRC Computaiocnal Science Series. 2015
2. Parallel Programming by Thonas Rauber and Gudula Runger. Springer 2010.
3. Using MPI Portable parallel Programming with the Message-Passing Interface by Willian Gropp, Ewing Lusk and Anthony Skjllum. 2014
4. https://www.mpi-forum.org/
5. http://lustre.org/

Modulo 2 – Programação em Ambiente de Sistemas Distribuídos (Supercomputadores e Clusters)

M2-I - Programação com MPI
Professora: Carla Osthoff (LNCC)
Carga Horária: 4h

Objetivos:
O objetivo deste curso é ensinar aos programadores que não estão familiarizados com programação paralela as noções básicas para desenvolver e executar programas paralelos de acordo com as normas do padrão MPI. Para isto iremos apresentar durante o curso apenas as rotinas básicas para se trabalhar com o padrão MPI. MPI é um padrão que vem sendo largamente utilizado para o desenvolvimento de programas paralelos em arquitetura de sistemas distribuídos, tais como clusters de PCs, A norma do padrão MPI foi implantada a mais de 25 anos e existe hoje muito material sobre este assunto. Este curso toma como base diversas apostilas que foram desenvolvidas ao longo dos últimos anos em centros de supercomputação, particularmente nos tutoriais do Laurence Livermore Laboratory (https://computing.llnl.gov/tutorials/mpi/). As demais referências utilizadas para a elaboração do curso estão relacionadas na seção bibliográfica. O curso inicia fornecendo informações sobre o ambiente de sistemas distribuídos e sobre a programação de troca de mensagens. A seguir apresentamos as rotinas básicas da norma MPI, incluindo as rotinas de gerenciamento de ambiente, de comunicação ponto a ponto e as rotinas de comunicações coletivas.

Durante o curso, serão apresentados exemplos em linguagem C e ao final serão passados exercícios. Assumimos que o usuário possui compreensão gerais sobre programação com a linguagem C.

Ementa

1- Introdução
Características de MPI:
Histórico de MPI:
Modelo de Programação.
Plataformas de Hardware
Como funciona MPI?

2- O padrão MPI.
Funcionalidades básicas do padrão MPI
Rotinas básicas de Comunicação.
Funcionamento Das Operações Ponto A Ponto.
Mensagens bloqueantes Versus não-bloqueantes
Ordem e Equidade:
3- Rotinas de Comunicação Coletivas
Exemplo Rotina de comunicação com MPI_Reduce
Exemplo Rotina de comunicação com SCATTER

Tipos de dados Derivados : Derived Data Types
Exemplo de um programa com Contiguous Derived Data Type (tipo contínuo)
Exemplo de um Tipo de dado Vetorial (Vector Derived Data Type)
Exemplo de um Indexed Derived Data Type (Tipo Indexado)
Exemplo de um Struct Derived Data Type (Tipo estruturado)

4- Comparação entre Grupos e Comunicadores

5- Considerações e Restrições sobre Programação com MPI:

6- Topologias Virtuais
7- MPI 3.1

Bibliografia

- Gabriel Silva, Programação Paralela com MPI: Um Curso Introdutório, Amazon Serviços de varejo do Brasil, 2018.
- Peter Pacheco. Parallel Programming with MPI . Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2019.
- Wesley Kendall, 2013, beginning MPI(An Introduction in C) .

M2-II- Introdução à Programação MPI com Extensões para E/S (MPI-IO)
Carga Horária: 4h
Professor: André Carneio

Resumo:

Na Computação de Alto Desempenho (High-Performance Computing), as operações de entrada e saída (I/O) são um
gargalo para um número crescente de aplicações, podendo comprometer a escalabilidade. Em ambientes de larga escala,
como clusters e supercomputadores, o uso de bibliotecas de alto-nível como MPI-IO permite obter melhor desempenho
através de I/O paralelo e de operações coletivas. Estas possibilitam o uso transparente de técnicas de otimização como

data sieving e two-phase I/O. Nesse contexto, este minicurso objetiva introduzir conceitos de I/O paralelo e MPI-IO, com um
enfoque prático, de forma que os participantes aprendam a utilizar a biblioteca em suas aplicações.

Pré-requisitos:
- Conhecimentos básicos sobre conceitos MPI:
- processos, ranks
- comunicadores
- operações bloqueantes e não bloqueantes (troca de mensagem)
- operações individuais e coletivas (troca de mensagem)
- MPI datatypes

MC2-III- Introdução ao "Adaptive MPI" (AMPI) (Esteban Meneses / Universidade de Costa Rica)
Professor: Esteban Meneses (Costa Rica National High Technology Center)
Carga Horária: 4h

Resumo:
Supercomputadores e grandes plataformas paralelas apresentam diversos desafios para o desenvolvimento de aplicações paralelas, tais como a exposição de concorrência, otimização de movimentação de dados, controle de balanceamento de carga, tratamento com heterogeneidade, tratamento de variações no comportamento de aplicações e tolerância a faltas. Para se lidar com esses desafios, uma ênfase em conceitos importantes como sobredecomposição, assincronia, migrabilidade e adaptatividade é necessária durante o desenvolvimento de aplicações. Para apresentar todas essas características, os ambientes de execução (runtimes) precisarão se tornar introspectivos e prover suporte automático a múltiplas tarefas que atualmente são responsabilidade de pessoas programadoras. Este mini-curso busca expor as pessoas participantes aos conceitos anteriormente mencionados, mostrando como suas implementações, em conjunto com um ambiente de execução introspectivo, pode levar ao desenvolvimento de aplicações que escalam independentes de plataformas. O minicurso foca no ambiente e paradigma de programação paralela Charm++ e sua implementação do padrão MPI através do Adaptive MPI (AMPI). Charm++ provê um modelo de programação paralela orientado a objetos baseado na troca de mensagens assíncronas. Seu ambiente possibilita a sobreposição de cálculo e comunicação de forma automática, balanceamento de carga, tolerância a faltas e checkpoints para execuções particionadas, entre outros. A abordagem seguida no mini-curso busca prover um guia para a migração de aplicações em MPI para o ambiente de execução de Charm++ e AMPI.

Conteúdo
Parte I: Introdução a objetos paralelos e ao ambiente de execução
1. Projeto de objetos paralelos: decomposição em termos de entidades lógicas
2. Modelo de execução: invocação de métodos assíncronos, execução guiada por mensagens
3. Charm++
a) Programação em termos de objetos

b) Exemplos
c) Benefícios: sobreposição de cálculo e comunicação automática, modularidade, gerenciamento de recursos automático e adaptativo
d) Programação: definição de objetos e interfaces, construção, mensagens, mensagens coletivas
e) Compilação e execução

Parte II: Migração de Aplicações Paralelas em MPI para Charm++ e AMPI

1. AMPI
a) Introdução
b) Transformação de aplicações de MPI para AMPI
c) Remoção de variáveis globais
d) Exemplos
e) Sobredecomposição
f) Balanceamento de carga dinâmico: métodos de empacotamento e desempacotamento, isomalloc, inserção de chamadas, escolha de estratégias
g) Checkpointing
h) Visualização de desempenho com Projections
2. Projeto de aplicações para Charm++
a) Exemplos
b) Grupos de objetos ativos: compartilhamento de dados entre objetos em mesmos nós ou processadores
c) Mensagens
d) Controle de fluxo com Structured Dagger (SDAG)

Bibliografia

Acun, B.; Gupta, A.; Jain, N.; Langer, A.; Menon, H.; Mikida, E.; Xiang Ni; Robson, M.; Yanhua Sun; Totoni, E.; Wesolowski, L.; Kale, L., "Parallel Programming with Migratable Objects: Charm++ in Practice", in High Performance Computing, Networking, Storage and Analysis, SC14: International Conference for , vol., no., pp.647-658, 16-21 Nov. 2014
Laxmikant V. Kale, Abhinav Bhatele, "Parallel Science and Engineering Applications: The Charm++ Approach", Chapter 1: The Charm++ Programming Model, CRC Press, 2013. ISBN 9781466504127
Laxmikant V. Kale, Eric Bohm, Celso L. Mendes, Terry Wilmarth, Gengbin Zheng, "Programming Petascale Applications with Charm++ and AMPI", Petascale Computing: Algorithms and Applications, Chapman & Hall / CRC Press, 2007. ISBN 9781584889090
Laércio Lima Pilla, Esteban Meneses, "Anais da 15ª Escola Regional de Alto Desempenho do Rio Grande do Sul", Capítulo 2: Programação Paralela em Charm++, SBC, 2015. ISSN 2177-0085

Modulo 3- Programação em GPU

M3-I – Introdução ao OpenCL
Carga Horária: 4h
Professor: Amanda Sabatini (LNCC)

Resumo:
Caracterizados pelo alto grau e densidade de paralelismo, os aceleradores são coprocessadores paralelos que se destacam pela enorme capacidade e eficiência computacional, tornando-os atrativos para a computação de alto desempenho. Exemplos populares destes incluem as tradicionais GPUs ("Graphics Processing Units") de propósito

geral e os recentes MICs ("Many Integrated Core"). Embora tenham princípios em comum, os aceleradores compreendem diferentes arquiteturas computacionais, exigindo-se para o fim de portabilidade de código a programação heterogênea. É nesse contexto que surge a linguagem de programação massivamente paralela denominada OpenCL, contração de Open Computing Language. A OpenCL é uma linguagem de padrão aberto baseada em C/C++, orientada à eficiência e flexibilidade, para programação uniforme e portável de sistemas heterogêneos, isto é, compostos por qualquer combinação de processadores convencionais e dispositivos aceleradores.

Objetivos:
Este curso contemplará a programação paralela heterogênea para dispositivos aceleradores usando a linguagem OpenCL, com enfoque na área de computação científica. Serão abordados tópicos como conceitos fundamentais de paralelismo, arquiteturas de aceleradores, técnicas de otimização e estudo de casos. É também objetivo do curso a aplicação prática em laboratório dos conhecimentos de programação paralela adquiridos pelos alunos.

Ementa:
1) Introdução à computação paralela
2) Arquiteturas de aceleradores
3) Linguagem OpenCL
4) Técnicas de eficiência e otimizações
5) Estudo de casos

Bibliografia:
1. Heterogeneous Computing with OpenCL, by Benedict Gaster, Lee Howes, David R. Kaeli, Perhaad Mistry, Dana Schaa
2. Programming Massively Parallel Processors: A Hands-on Approach, by David B. Kirk, Wen-mmei W. Hwu- 2nd Ed.
3. http://www.khronos.org/opencl

M3-II - Introdução a Programação em CUDA
M3-III – Programação Avançada em Cuda
Carga Horária: 8h
Professor: Roberto Pinto Souto (LNCC)

Resumo:
Curso introdutório em linguagem CUDA, uma extensão da linguagem C utilizada para a programação de aplicações de propósito geral, sendo executadas em arquitetura massivamente paralela de placas gráficas, denominadas GPU.

Ementa:
A arquitetura das GPUs (“Graphics Processing Unit”) foi projetada para efetuar os cálculos em ponto flutuante mais frequentemente realizados em aplicações gráficas. Esta é uma arquitetura bem mais especializada que a presente em CPUs (“Central processing unit”) , o que faz com que estes programas rodem mais rapidamente nestas plataformas. Além disso, as GPUs são altamente paralelizadas, já havendo atualmente modelos com milhares de núcleos computacionais (“cores”). Embora originalmente desenvolvida para executar aplicações gráficas com maior eficiência, também é cada vez maior o uso de GPUs em aplicações não-gráficas. São aplicações que envolvem a resolução de métodos de álgebra

numérica computacional, frequentemente também utilizados nas aplicações gráficas, obtendo-se um significativo ganho de desempenho. Neste curso será apresentada a arquitetura de placas gráficas NVIDIA e a conexão destas placas com o computador hospedeiro. Em seguida será apresentado a plataforma de programação CUDA, que é uma extensão do C/C++ processada tanto no computador hospedeiro quanto nas placas gráficas conectadas a este. Serão apresentados exemplos de códigos de cada um dos tópicos abordados, os quais os alunos poderão executá-los nas máquinas disponíveis em sala de aula.

Tópicos abordados:
1. Introdução (Arquitetura)
2. Modelo de Paralelismo em CUDA
2.1 Organização e Identificação
2.2 Atribuição
2.3 Escalonamento
3. Hieraquia de Memoria
6. Métricas desempenho e otimização
7. Estudo de caso: multiplicação de matrizes

Bibliografia:

1) CUDA by Example: An Introduction to General Purpose GPU Programming, by David Weller,
2) Programming Massively Parallel Processors: A Hands-on Approach (Applications of GPU Computing Series), by David B. Kirk and Wen-mmei W. Hwu- Second Edition
3) http://docs.nvidia.com/cuda/cuda-cprogramming-guide/index.htm
4) http://devblogs.nvidia.com/parallelforall

M3-IV- Introdução à Programação em Aceleradores com Diretivas
MC3-V- Programação Avançada em em Aceleradores com Diretivas
Professora: Pedro Pais Lopes/Exaflop
Carga Horária: 8h
Local: **Laboratório 5**

Objetivos:

Processadores com centenas a milhares de núcleos de processamento estão presentes no cotidiano da computação. Este número expressivo de núcleos estão em placas gráficas (denominadas GPUs) como também em aceleradores x86 (como os Xeon Phi) ou mesmo em chips ARM, com arquiteturas complexas de acesso a memória e de operações de ponto flutuante. A programação para utilizar este grande poder de processamento pode envolver linguagens especializadas, o que demanda profunda modificação do programa e possivelmente redução da sua portabilidade. Padrões de programação baseadas em diretivas, amplamente utilizadas para expor paralelismo em arquitetura de memória central, surgiram ou foram expandidos para permitir expor o paralelismo destes processadores massivamente paralelos. Neste minicurso serão tratados os padrões existentes (OpenACC e OpenMP v4.5), suas principais diretivas, compiladores com suporte a estes padrões e desempenho obtido em aplicações simples em algumas arquiteturas (GPU, CPU multicore e Xeon Phi).

Ementa:
• Introdução aos conceitos de processamento paralelo,
• diretivas de compilação,
• formato de diretivas,

• condicional de compilação,
• construtores paralelos,
• construtores de compartilhamento de trabalho,
• construtores combinados,
• clausulas,
• diretivas de sincronização,
• funções de ambiente de execução,
• funções de bloqueio,
• funções do tempo
• variáveis de ambiente.

Modulo 4- Programação com Processadores Multicore

M4-I- Introdução à Programação Paralela/ OpenMP
Professora: Carla Osthoff (LNCC)
Carga Horária: 4h

Resumo:
O objetivo deste curso é ensinar aos programadores que não estão familiarizados com programação paralela as noções básicas para desenvolver e executar programas paralelos de acordo com as normas do padrão OpenMP (www.openmp.org) . Para isto iremos apresentar durante o cursos rotinas básicas para se trabalhar com o padrão OpenMP v 5.0. OpenMP é um padrão que vem sendo largamente utilizado para o desenvolvimento de programas paralelos em arquitetura de sistemas de memória compartilhada, tais como ambientes computacionais com processadores multicore. A norma do padrão OpenMp foi implantada a mais de 20 anos e existe hoje muito material sobre este assunto. Este curso toma como base diversas apostilas que foram desenvolvidas ao longo dos últimos anos em centros de supercomputação, particularmente nos tutoriais do Laurence Livermore Laboratory (https://computing.llnl.gov/tutorials/openMP/). As demais referências utilizadas para a elaboração do curso estão relacionadas na seção bibliográfica. O curso inicia fornecendo informações sobre o ambiente de computação de memória compartilhada e sobre a programação conceitos de processamento paralelo.. A seguir apresentamos uma introdução às diretivas e os construtores OpenMP conforme padrão v5.0, rotinas da biblioteca OpenMP e as funções de ambiente. Durante o curso, serão apresentados exemplos em linguagem C e ao final serão passados exercícios. Assumimos que o usuário possui compreensão gerais sobre programação com a linguagem C.

Ementa:
- Introdução aos conceitos de processamento paralelo
- Introdução aos conceitos de processamento paralelo,
- Introdução ao OpenMP:
  • construtores paralelos,
  • construtores de compartilhamento de trabalho,
  • clausulas,
  • diretivas de sincronização,
  • funções de ambiente de execução,
  • variáveis de ambiente.

Bibliografia

- OpenMP Common Core: Making OpenMP Simple Again – by Tim Mattson, Helen He, Alice Koniges (2019)
- Using OpenMP – The Next Step – by Ruud van der Pas, Eric Stotzer and Christian Terboven (2017)
- Using OpenMP – Portable Shared Memory Parallel Programming – by Chapman, Jost, and Van Der Pas (2007).

MC4-II- Introdução à Programação Paralela e Vetorial (Matheus Serpa / UFRGS)
MC4–III - Programação Paralela e Vetorial Avançada (Matheus Serpa / UFRGS)

Professor: Matheus Serpa (UFRGS)
Carga Horária: 8h

Resumo:
Tradicionalmente o aumento de desempenho das aplicações se dava de forma transparente aos programadores devido ao aumento do paralelismo a nível de instruções e aumento de frequência dos processadores. Entretanto, este modelo não se sustenta mais. Atualmente para se ganhar desempenho nas arquiteturas modernas, é necessário conhecimentos sobre programação paralela e programação vetorial. Ambos paradigmas são tratados de forma lateral em cursos de computação, sendo que muitas vezes nem são abordados. Neste contexto, este minicurso objetiva propiciar um maior entendimento sobre os paradigmas de programação paralela e vetorial, de forma que os participantes aprendam a otimizar adequadamente suas aplicações para arquiteturas modernas. Como plataforma de desenvolvimento, serão utilizados os processadores Intel Xeon e Intel Xeon Phi do SDumont.

Ementa:
- Introdução ao OpenMP
- Introdução a Programação Vetorial (SIMD)
- Programação OpenMP/SIMD com Intel Xeon Phi

Pré-requisitos:
- Conhecimentos básicos em C
- Conhecimentos básicos em Linux

Bibliografia

- Jeffers, James, and James Reinders. **Intel Xeon Phi Coprocessor High Performance Programming.**. Newnes, 2013.
- http://www.openmp.org/.

M4-IV - Otimização de código com Parallel Studio: um estudo de caso
Professor: Frederico Cabral(LNCC)
Carga Horária: 4h

Resumo:
Visão geral do Intel Parallel Studio XE e como está configurado no SDumont, apresentando as melhores práticas na utilização do perfilador e uma visão geral sobre ferramentas para perfilagem. O conjunto de desenvolvimento de software paralelo **Intel Parallel Studio XE** possibilita simplificar o desenvolvimento, depuração e ajuste de código, ajudando a utilizar o processamento paralelo para aumentar o desempenho do aplicativo

Objetivos:
Este curso contemplará a programação paralela em arquiteturas de computadores hibridas compostos por nós computacionais com processadores de memória compartilhada multicores e aceleradores MIC e GPU´s. Serão abordados tópicos como

conceitos fundamentais de paralelismo, arquiteturas de memória distribuída e de memória compartilhada, técnicas de otimização e estudo de casos. É também objetivo do curso a aplicação prática em laboratório dos conhecimentos de programação paralela adquiridos pelos alunos.

Ementa:

- Utilização do conjunto de ferramentas para simplificar a criação confiável e rápida de código paralelo, obtendo um aumento de desempenho usando relatórios de vetorização e otimização explícito.

-Apresentação do suporte do software para normas de suporte para OpenMP 4.0, 3.0 MPI, C ++ 2011 Full, suporte para Fortran 2003 e Fortran 2008 BLOCK completos.

- **Composer Edition:** inclui compiladores, bibliotecas de desempenho, e modelos paralelos otimizados para construir código paralelo rápido, em C++ e Fortran, para possibilitar o acréscimo de modelos paralelos intuitivos embutidos e suporte a vetorização.

-**Professional Edition**: possibilita a função de desempenho de perfil, design/desenvolvimento de threading, depurador de projeto de memória e thread, além de possibilitar a construção, depuração e otimização de código paralelo.

-**Cluster Edition:** Adiciona uma biblioteca de comunicações de cluster MPI, junto com verificação de erros MPI e projeto de afinação, além da construção, depuração e sintonia de código paralelo rápido que inclui MP.I

Bibliografia:

1. High Performance Parallelism Pearls- Multicore and Many-Core Programming Apporaches- James Reinders and Ji, jeffers- MK Moragan Kauffman- 2015
2. Intel "Xeon PHI" processor High performance Porgramming- Knights Landing Edition- Jjim Jeffers, James reinders and Avinash Sodani, 20163
3. Parallel Programming by Thonas Rauber and Gudula Runger. Springer 2010.
5 https://software.intel.com/en-us/intel-parallel-studio-xe-support/documentation

Modulo 5- Programação com Ambientes de Deep Learning (DGX-2)

M5-I -Introdução à Deep Learning
Professor: NVIDIA
Data: 21/02/2019
Carga Horária: 4h

M5-II- .Introdução à utilização de Containers para Deep
Professor: NVIDIA
Data: 21/02/2019
Carga Horária: 4h

Modulo 6- Programação com Bibliotecas de HPC

MC6-I - Introdução a workflows científicos paralelos em Python/Parsl
Professor: Luiz Gadelha (/LNCC)
Carga Horária: 4h

Resumo
Python é uma linguagem de programação de alto nível para programação de propósito geral lançada ao público em 1991. Possui contruções que permitem a criação de programas de pequena à grande escala e possui uma filosofia de design que prioriza principalmente a legibilidade de código. É cada vez mais usada em computação científica e numérica. Python tem tipagem dinâmica e gerência automática de memória. Suporta múltiplos paradigmas de programação, como programação orientada à objetos, imperativa, funcional e procedural. É citado frequentemente como uma de suas melhores características sua grande e abrangente biblioteca
padrão. O *Python Package Index*, o repositório oficial de pacotes de terceiros, possui outros 130000 pacotes. Interpretadores da linguagem Python estão disponíveis para diversos sistemas operacionais. A implmentação de referência, a CPython, é open source e possui um modelo de desenvolvimento comunitário assim como a maior parte das outras implementações alternativas**.**
Pré-requisitos
+ Conhecimentos básicos de programação estruturada
+ Conhecimentos básicos de programação orientada à objetos
+ Conhecimentos básicos do uso do shell *bash* ou similar no ambiente GNU/Linux
Ementa
+ Introdução geral à HPC
+ Processamento em memória compartilhada
- Threads vs processos
- Módulo threading vs módulo multiprocessing
+ Módulo multiprocessing
- Exemplo: aproximação de pi com Monte Carlo
+ Processamento em memória distribuída
- Introdução à MPI
- Módulo mpi4pi
- Módulo mpi4pi + módulo numpy
_ Exemplo: reimplementação da aproximação de pi com MPI
+ Biblioteca Parsl

M6-II - R em ambiente HPC
Professora: Raquel Lopes (INCA/LNCC)
Carga Horária: 4h

Resumo
O R é uma linguagem bastante utilizada por cientistas de dados, estatísticos, biológos, epidemiologistas e funciona muito bem em muitas pesquisas, gerando resultados de qualidade em relatórios e artigos científicos. A flexibilidade e facilidade de programar avançando nas análises a partir de funções e pacotes disponíveis são vantagens decisivas na escolha dessa linguagem por pesquisadores. No entanto, a análise de dados massivos decorrentes principalmente da era 'Big Data' envolvem avançados algoritmos com grande número de parâmetros necessários para ser estimado. Analisar os dados massivos tem sido um fator limitante na linguagem R. Isso porque o R foi originalmente pensado para

ser executado em uma única *thread*, além de ser uma linguagem interpretada, sendo a maior parte das instruções executadas diretamente. Aliado a esses fatores, o conhecimento sobre MPI, SOCKET, C, C++ e Fortran são barreiras para grande de pesquisadores que lidam com cálculos estatísticos intensivos e estão tentando acelerar os cálculos implementando algoritmos paralelos. Ao longo dos anos, diversas iniciativas tem sido feitas para melhorar a eficiência e desempenho das aplicações em R. Esse curso tem como proposta apresentar algumas dessas estratégias de paralelismo e melhora na eficiência para ganho de escalabilidade na linguagem R.

Pré-requisitos

- R básico
- Conhecimentos em plataformas HPC

Ementa

- R, introdução geral
- O que é o R
- Vantagens do R
- Funções da família apply
- Pacotes de paralelismo do R
- Pacote parallel (funções análogas às da família apply)
- Pacote foreach e backends parelelos (estrutura análoga ao laço for nativo do R)
- Benchmark no R
- Exemplos com análise de agrupamento
- Otimização do código
- Estudo de caso, Model-R

Bibliografia

- State of art in parallel computing with R. Journal of Statistical Software. August, 2009. Vol. 31(1).
- A survey of R software for parelle computing (2014) Vol. 2(4). Americal Journal of Applied Mathematics and Statistics.
- Ethan McCallum e Stephen Weston (2012). Parallel R. (Book). O'Reilly.
- Lin, H., Yang, S. and Midkiff, S. P. (2013). A Parallel R Framework for Processing Large Dataset on Distributed Systems. Dataset on Distributed Systems, DataCloud.

Módulo 7- Gerenciamento de Clusters (Sistemas Distribuídos)

M7-I - Introdução à configuração e gerenciamento de Clusters(Andre Carneiro/Bruno Fagundes/LNCC)

Modulo 8 – InfiniBand In-Network Computing technology

MC8-I- InfiniBand In-Network Computing technology overview and advantages
Professor: Gilad Shainer / Mellanox
Carga Horária: 4h

Abstract:
The ever increasing demands for higher computation performance drive the creation of new datacenter accelerators and processing units. Previously CPUs and GPUs were the main sources for compute power. The exponential increase in data volume and in

problems complexity, drove the creation of a new processing unit – the I/O processing unit or IPU. IPUs are interconnect elements that include In-Network Computing engines, engines that can participate in the application run time, and analyze application data as it being transferred within the data center, or at the edge. The combination of CPUs, GPUs, and IPUs, creates the next generation of data center and edge computing architectures. The first generations of IPUs are already in use in leading HPC and Deep learning data centers, have been integrated into multiple MPI frameworks, NVIDIA NCCL, Charm++ and others, and have demonstrated accelerate performance by nearly 10X. The session will review the IPUs state of the art, how to usage the In-Network Computing technology for acceleration HPC and AI applications and algorithms, and the roadmap for future In-Network Computing technologies.