MoBiDiCk: Uma ferramenta para computação distribuída na internet

Apresentado por:
Marcos Camponogara
Tiago Silva da Silva
Tiago Thompsen Primo

Sumário da apresentação

- Tecnologia
- Protein Folding
- Introdução
- Arquitetura do Sistema
- Modelo de comunicação
- Registro de um nodo
- Execução de uma tarefa
- RAMAPLOT
- FOLDTRAJ
- Direções futuras

Tecnologia

- O que é um Grid?
  - O termo denota "uma proposta de infraestrutura de software e hardware para integração de recursos computacionais, dados e pessoas geograficamente dispersas de modo a formar um ambiente colaborativo de trabalho."

  Foster 2004 (Grid 2 Blueprint for a new computing infraestructure)

Tecnologia

- Aspectos considerados
  - Heterogeneidade – multiplicidade de recursos; trata das diferençasentre processadores, velocidade e arquitetura;
  - Escalabilidade – pode crescer de poucos recursos para milhões; problema em potencial é o desempenho a medida que o tamanho aumenta;
  - Dinamicidade ou adaptabilidade – falha de um recurso é regra, e não exceção! Gerenciar recursos e organizar seu comportamento para extrair desempenho a partir dos recursos e serviços disponíveis.

Tecnologia

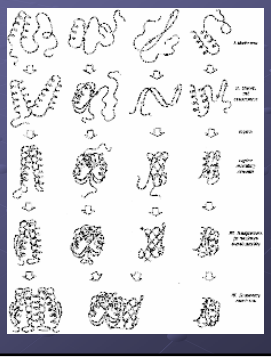
- Comparação entre sistemas distribuídos e grid

Sistemas distribuídos	Grid
1. Conjunto virtual de nodos	1. Conjunto virtual de recursos
2. O usuário tem acesso a todos os nodos	2. O usuário não tem acesso aos domínios individuais
3. O acesso a um nodo = a todos os recursos do nodo	3. O acesso a um recurso pode ser restrito
4. O usuário está ciente das potencialidades	4. O usuário possui pouco conhecimento sobre os recursos
5. Os nodos pertencem a uma organização	5. Recursos espalhados por múltiplas organizações

Protein Folding

Protein Folding:

Processo pelo qual uma proteína assume sua forma, ou configuração funcional.



Complexo

Diversas Variáveis a considerar

Simular de forma correta

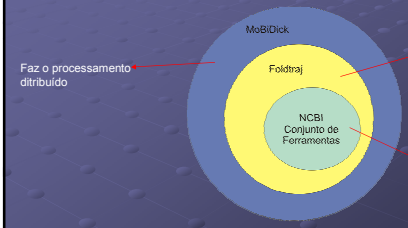
Introdução

Modular Bioinformatics Distributed Computing Kernel

Tem como objetivo solucionar o problema de Protein Folding

Parceria...

Introdução



Faz o processamento ditribuído

Junta todas as Ferramentas

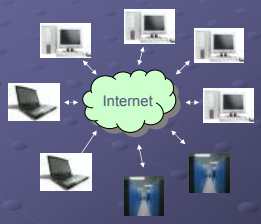
Entrez (base de dados integrada sobre Bioinformatica)

BLAST (Busca DNA e seqüência de proteínas)

Cn3D (visualização em 3d)

Introdução

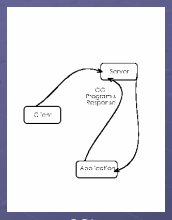
Como Juntar?



Pessoas compartilham ciclos de cpu para causas que julguem importantes.

Usa de diversos tipos de maquinas...

Arquitetura do Sistema



CGI

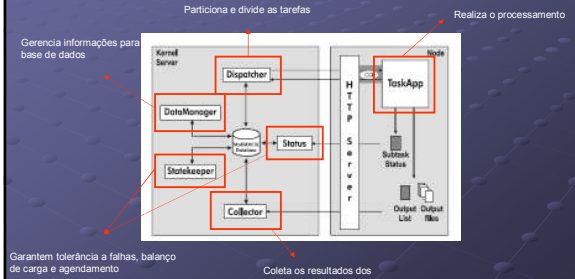
Opera sobre um conjunto de nodos interconectados

Um nodo pode vir a ser:
Estação de trabalho
Cluster
Estação Multiprocessada

Desempenho:

Apenas recursos para rodar o servidor web

Arquitetura do Sistema



Arquitetura do Sistema

- A modularidade do Kernel, permite que os recursos de CPU e I/O sejam otimizados.
- Funções de gerencia de sistema podem ser distribuídas através de um espalhamento dos módulos de Kernel em diversos servidores.
- Por exemplo, uma configuração de Kernel distribuída, pode envolver quatro servidores, onde:
  - o *dispatcher* é instalado em um determinado servidor,
  - o *collector* em um segundo,
  - *status* e *statekeeper* em um terceiro,
  - o *data manager* em um quarto.
- A base de dados tem cada parte montada em um servidor e é compartilhada dentre todos os servidores.
- O uso de um servidor de Kernel multiprocessado, é uma configuração alternativa que pode vir a beneficiar banda de CPU já que os módulos de Kernel podem rodar simultaneamente em diversos processadores.

Modelo de comunicação

Os passos envolvidos na transmissão da mensagem de m1 até m2 são os seguintes:

1- m_1 abre uma conexão TCP, C, entre N_1 e N_2
2- m_1 envia uma requisição http contendo a mensagem, M, para m_2 através de C
3- S_2 recebe a requisição, inicia m_2, e passa a requisição para m_2
4- m_2 extrai M da requisição, processa M, e da um retorno R
5- S_2 captura R, coloca um cabeçalho HTTP e o envia através de C
6- m_1 recebe a resposta de S_2 e lê R
7- m_1 fecha C

Registro de um nodo

- Browser
- Datamanager
  - Base de dados
- Atributos chave
  - Host name, IP, velocidade de CPU, número de CPUs, sistema operacional, capacidade de disco e memória, contato

Registro de um nodo

- Selecionar horários e dias para executar tarefas
- Agendamento *all-or-none*
- Diferente do SETI@home

Execução de uma tarefa

- Uma computação é solicitada ao módulo *Dispatcher (browser)*
- *Dispatcher* dá início a uma seleção de nodos
  - Compila uma lista de nodos candidatos
- Para se candidatar à execução de uma tarefa, um nodo deve cumprir algumas condições

Execução de uma tarefa

- Registro
- Participação
- Acessibilidade
- Conectividade
- Configuração

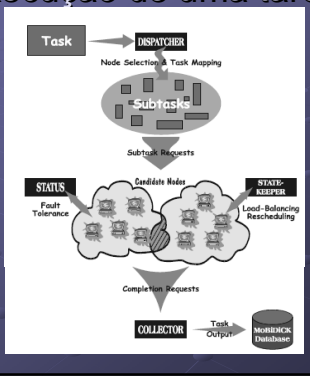
Execução de uma tarefa

- Registro e Participação
  - Consulta a base de dados
- Acessibilidade
  - Agenda de acesso ao nodo, verifica a disponibilidade e tempo
- Conectividade e Configuração
  - *Handshaking* único
  - *Dispatcher* envia uma requisição “teste” ao *TaskApp*

Execução de uma tarefa

- Se nenhuma resposta recebida ou erro no estabelecimento da conexão
  - Nenhuma das condições é suprida
- Servidor *web* responde com erro
  - Conectividade OK mas falha a condição de Configuração
- Uma resposta válida é recebida do *TaskApp*
  - Conectividade e Configuração OK
- Condições adicionais

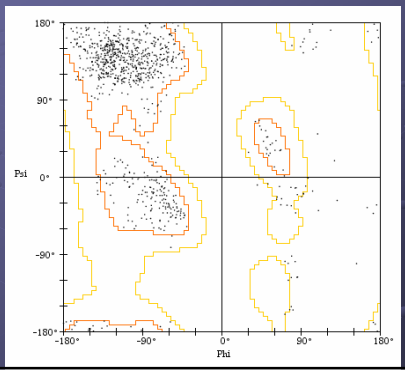
Execução de uma tarefa



RAMAPLOT

- TaskApp que usa estruturas tridimensionais de proteínas, provenientes da base de dados de modelagem molecular (MMDB)
- O objetivo é gerar o gráfico de Ramachandran, que é um gráfico de distribuição dos ângulos de rotação de proteínas de carbono α, para cada uma das 851 estruturas de proteínas da base de dados

Gráfico de Ramachandran



Configuração

- A tarefa foi realizada em um cluster com 15 nodos, cada um configurado com:
  - dois processadores Intel Pentium II 400MHz
  - 512Mb de memória
  - sistema operacional Linux RedHat
  - servidor HTTP apache
  - Servidor principal Sun Sparc Ultra-1 executando solares 2.6
  - A base de dados MMDB foi copiada para o disco rígido de cada nodo.

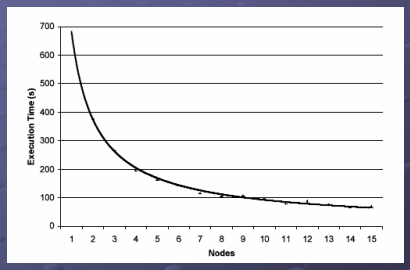
RAMAPLOT

- A tarefa utiliza dois parâmetros de entrada:
  - *dbsize:* representa o número de registros a ser processado, que varia de 1 a 851 (primeiro e último registro da base de dados, respectivamente)
  - *dbstart:* define o número do registro inicial na base de dados.
- Foram realizados testes com 15 instâncias da tarefa RAMAPLOT, iniciando com um único nodo e adicionando um novo nodo para cada nova instância, sendo que cada nodo possuía somente uma CPU.

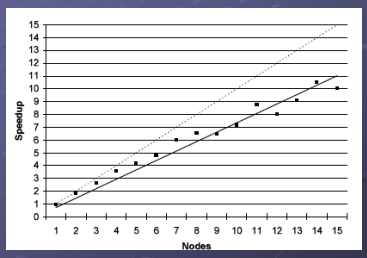
RAMAPLOT

- Para cada instância foram especificados o tempo de execução, o *speedup*, e a eficiência
- O *speedup = Tp/Ts em que:*
  - Tp = tempo de execução paralela
  - Ts = tempo de execução serial
- O melhor tempo serial obtido foi de 695 segundos.

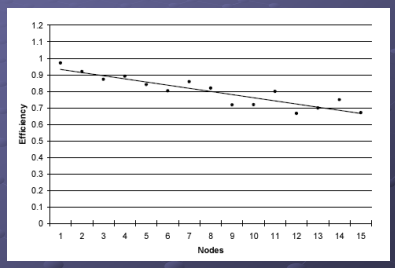
Tempo execução



Speedup



Eficiência



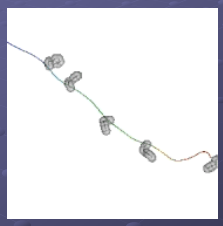
Resultados

Instance	Subtasks	Time (s)	Speedup	Efficiency
1	1	714	0.972	97.2
2	2	378	1.84	92.0
3	3	265	2.62	87.2
4	4	195	3.57	89.1
5	5	165	4.20	84.1
6	6	144	4.82	80.3
7	7	116	6.01	85.8
8	8	106	6.55	81.9
9	9	107	6.47	71.9
10	10	96	7.19	71.9
11	11	79	8.79	79.9
12	12	87	8.00	66.7
13	13	76	9.09	69.9
14	14	66	10.49	74.9
15	15	69	10.07	67.1

FOLDTRAJ

- Aplicação utilizada no problema do dobramento de proteínas, que provém do campo da biologia estrutural.
- Antes das proteínas poderem executar sua função bioquímica, elas se auto constroem, ou "dobram" (fold).
- O processo de dobrar proteínas permanece um mistério
- Quando as proteínas não dobram corretamente, podem ocorrer sérios efeitos, incluindo doenças conhecidas como Alzheimer, Vaca Louca, Parkinson e esclerose lateral amiotrófica.

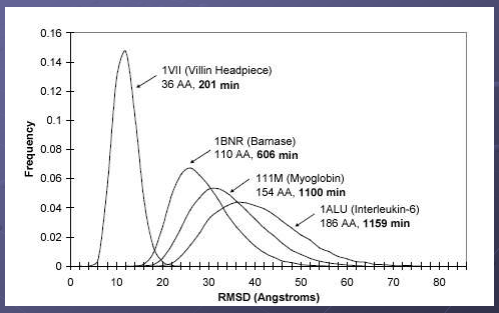
FOLDTRAJ



FOLDTRAJ

- O FOLDTRAJ pode gerar um número randômico de proteínas quimicamente válidas, colocando cada uma em um arquivo separado em formato binário ASN.1 ou ASCII
- O arquivo entrada conhecido como "distribuição de trajetória", contém informação de freqüência em um espaço angular 2D de um aminoacido sobre uma proteína particular
- A corretude de uma estrutura é medida pelo cálculo da RMSD (*Root Mean Squared Deviation*) relativo ao dobramento nativo das proteínas.

FOLDTRAJ



Direções futuras

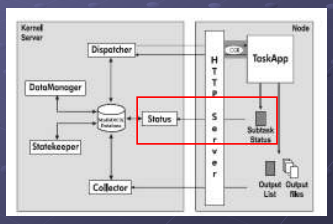
- Estimativa do tempo de execução das subtasks:
  - O tempo de execução é influenciado diretamente por:
    1. Carga da sub-tarefa
    2. *Rating* do nodo
  - Tempo de execução:
    T = *h*F(L)/R

    F = complexidade temporal da tarefa em função carga da sub-tarefa
    L = Carga da sub-tarefa
    R = *rating* do nodo
    h = F(Li)/Ri
    Li = média da carga das subtarefas em instâncias anteriores
    Ri = *rating* médio de todos os nodos que computaram estas subtarefas

Direções futuras

- Tolerância a falhas

Suspeita de falhas:
- Status não muda
- Não é possível obter status



Direções futuras

- Tolerância a falhas
  - Possíves tratamentos:
    - Continuar a computação, desconsiderando a falha
    - Cancelar a execução
    - Restartar a execução
    - Re-alocar a sub-tarefa que falhou

Direções futuras

- Migração de tarefas
  - A migração de uma tarefa deve acontecer se a tarefa necessitar de mais tempo que o disponível naquele nodo (*schedule overflow*)
  - O Statekeeper verifica periodicamente a possibilidade de *schedule* overflow checando o status das sub-tarefas
  - Se um overflow for antecipado a sub-tarefa é então migrada para um novo nodo