

Universidade Federal de Ouro Preto - UFOP
Instituto de Ciências Exatas e Biológicas - ICEB
Departamento de Computação - DECOM

TerraME HPA (High Performance Architecture)

Aluno: Saulo Henrique Cabral Silva
Matricula: 08.1.4137

Orientador: Joubert de Castro Lima

Co-orientador: Tiago Garcia de Senna Carneiro

Ouro Preto
15 de setembro de 2011

Universidade Federal de Ouro Preto - UFOP
Instituto de Ciências Exatas e Biológicas - ICEB
Departamento de Computação - DECOM

TerraME HPA (High Performance Architecture)

Proposta de monografia apresentada ao curso de Bacharelado em Ciência da Computação, Universidade Federal de Ouro Preto, como requisito parcial para a conclusão da disciplina Monografia II (BCC391).

Aluno: Saulo Henrique Cabral Silva
Matricula: 08.1.4137

Orientador: Joubert de Castro Lima

Co-orientador: Tiago Garcia de Senna Carneiro

Ouro Preto
15 de setembro de 2011

Resumo

Em 2008 foi concluído um projeto CNPq PIBIC que implementou uma arquitetura de alto desempenho para o ambiente de modelagem *Terra Modeling Environment* - TerraME. O TerraME é um arquitetura de software destinada ao desenvolvimento de modelos dinâmicos espacialmente explícitos, que vem sendo utilizada, por exemplo, para a modelagem e simulação de processos de mudança de uso e cobertura do solo (LUC) para toda região amazônica, no âmbito do projeto GEOMA, e para o desenvolvimento de modelos epidemiológicos junto à FIOCRUZ [7], como são os casos do controle da Dengue nas cidades do Rio de Janeiro e Recife. A arquitetura implementada em 2008 não permite um modelo possa ser executado em paralelo, impondo restrições a modelos que demandam uma grande quantidade de recursos computacionais em termos de velocidade de processamento matemático ou gráfico. Desta forma, este trabalho visa desenvolver uma API para a plataforma TerraME que permita que o usuário possa executar seu modelo de forma paralela, aproveitando os recursos de *hardware* disponíveis na máquina.

Palavras-chave: Paralização, TerraME, Modelagem.

Sumário

1	Introdução	1
2	Justificativa	3
3	Objetivos	4
3.1	Objetivo geral	4
3.2	Objetivos específicos	4
4	Metodologia	5
5	Atividades Desenvolvidas	6
6	Desenvolvimento	9
7	Trabalhos Futuros	9
8	Cronograma de atividades	10

Lista de Figuras

1	Ambiente de desenvolvimento e suporte a modelagem ambiental TerraME.	1
2	Desenho de solução atual.	7
3	Exemplo de tabela utilizada no modelo THROLL.	8
4	Exemplo de inserção em uma das estruturas do modelo THROLL(tabela tripla).	8

Lista de Tabelas

1	Cronograma de Atividades.	10
---	-----------------------------------	----

1 Introdução

Os sistemas Terrestres hoje estão submetidos em profundas mudanças cada vez mais intensas que vêm alterando seu funcionamento. Essas alterações têm trago um forte impacto direto na integridade do meio ambiente e na qualidade de vida das pessoas. Diversos estudos apontam as ações humanas como à principal causa das alterações dos biomas.

A intensificação das mudanças ambientais causadas por processos antrópicos, cujos impactos são quase sempre negativos ao ambiente, exige que pesquisadores, empresários, e governos forneçam respostas aos desafios científicos e tecnológicos ligados ao entendimento do funcionamento dos sistemas terrestres, mas para isso eles precisam de ferramentas de modelagem que sejam confiáveis e capazes de capturar a dinâmica e os resultados das dinâmicas das ações da sociedade humana. Modelos computacionais que reproduzem de uma forma satisfatória o fenômeno sob estudo contribuem para o ganho do conhecimento científico no que diz respeito ao seu funcionamento e este conhecimento científico no que diz respeito ao seu funcionamento, este conhecimento pode servir como uma base inicial para o planejamento e definição de políticas públicas através da sua otimização e obtenção de melhores resultados (chamadas políticas saudáveis).

O *framework* TerraME é uma plataforma de domínio público para o desenvolvimento de modelos ambientais espacialmente explícitos integrados a um Sistema de Informação Geográfica (SIG), desenvolvido pela parceria TerraLAB-UFOP (Universidade Federal de Ouro Preto) e INPE (Instituto Nacional de Pesquisas Espaciais). Esta plataforma fornece uma linguagem de modelagem de alto nível que é utilizada para a descrição dos modelos/algoritmos e sua posterior integração aos bancos de dados geográficos.

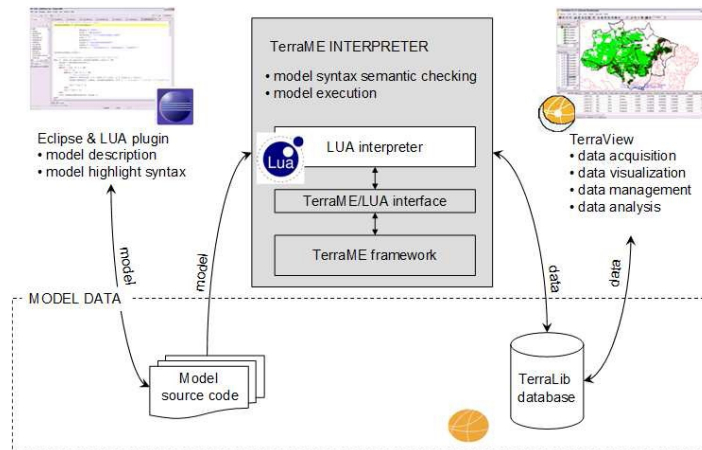


Figura 1: Ambiente de desenvolvimento e suporte a modelagem ambiental TerraME.

TerraME é um ambiente de desenvolvimento e suporte a modelagem ambiental espacial dinâmica que suporta modelos de computação baseados em autômatos celulares [13] e conceitos de autômatos celulares aninhados (Nested-CA) [2]. Um modelo espacial dinâmico é uma representação abstrata de um fenômeno que evolui no tempo e no espaço, baseado em descrições de entidades, processos e relações entre eles. Dessa forma, o TerraME está associado a um Sistema de Informação Geográfica (SIG) que

fornece a localização espacial dos dados. Os resultados destes modelos são mapas que mostram a distribuição espacial de um padrão ou de uma variável contínua. TerraME permite simulação em duas dimensões de espaços celulares regulares e irregulares.

Dentre os principais problemas entre as plataformas de simulação existentes hoje é, o tempo de execução do modelo e a quantidade de dados utilizada para a execução dos mesmos. Com o aumento das mudanças ambientais esses problemas tendem a piorar ainda mais, para tentar reproduzir a dinâmica dos fenômenos naturais, os modelos agora devem considerar mais variáveis, mais processos passam a interferir na dinâmica do fenômeno e mais dados são necessários. Com todas essas modificações os modelos passaram a ficar muito mais complexos, fazendo com que o espaço em disco utilizado seja muito grande e seu tempo de execução fique ainda maior. Para que o modelo seja executado em tempo hábil o modelador muitas das vezes tem que omitir alguns pontos do modelo que prejudicam a fidelidade do modelo ao tentar expressar a dinâmica do fenômeno.

Com o aumento a demanda por plataformas de hardware e software que possam oferecer maior capacidade de processamento para a execução de aplicações associadas a problemas complexos do mundo real, a programação paralela e a abordagem distribuída [1] surgiram como técnicas que tem sido utilizadas para implementar tais aplicações nos casos em que um único processador requeira muito tempo para processá-las sequencialmente e em que o espaço de armazenamento seja insuficiente.

A proposta deste trabalho visa efetuar a paralelização do Kernel TerraME. Para isto é preciso criar uma API para que o usuário possa efetuar a paralelização do seu modelo de forma simples e intuitiva. Como resultado deste projeto, a plataforma TerraME ganhará robustez, pois modelos criados a partir da plataforma poderão ser executados de forma paralela, aproveitando assim, todo o poder de hardware disponível da máquina.

2 Justificativa

É clara e reconhecida a importância do desenvolvimento de modelos ambientais baseados no impacto que eles podem causar no cotidiano da sociedade. A simulação de processos ambientais ou a simulação das interações humano-ambiente são instrumentos de pesquisa de impacto e predição. Mas processos de simulações de sistemas terrestres consomem uma enorme quantidade de recursos computacionais. Modelos ambientais devem ser avaliados em diversas escalas. Atualmente, os modelos ambientais de larga escalas desenvolvidos em TerraME levam muito tempo para serem executados (O modelo THROLL[4] leva até 90 dias para ser executado). Este fato torna penosa a atividade de modelagem e, às vezes, a inviabiliza.

As pesquisas nesse sentido levarão ao desenvolvimento de novas técnicas computacionais e de modelagem que poderão ajudar a minimizar o impacto de desastres naturais ou epidemiológicos. Além de contribuir para a elaboração de prognósticos mais precisos acerca do comportamento das interações entre o homem e o ambiente.

Por fim, o trabalho está em conformidade com os desafios traçados pela Sociedade Brasileira de Computação (SBC) 2006-2016 [3], sejam estes: (i) gestão da informação em grandes volumes de dados, (ii) modelagem computacional de sistemas complexos artificiais, naturais e sócio-culturais e da interação homem-natureza.

3 Objetivos

3.1 Objetivo geral

Este trabalho tem por objetivo o desenvolvimento e documentação de mais um núcleo distribuído para o TerraME, chamado TerraME HPA.

O novo núcleo construído deverá produzir os seguintes subprodutos:

1. Documento de Especificação que identifica os principais requisitos a serem atendidos pela arquitetura TerraME HPA.
2. Documento de Desenvolvimento explicando as estruturas de dados, algoritmos, arquitetura de software e decisões de projeto envolvidas no desenvolvimento do TerraME HPA.
3. Guia de Programação descrevendo como sua Application Programming Interface (API) deve ser utilizada.
4. Tutoriais explicando sua instalação e utilização para o desenvolvimento de modelos dinâmicos espacialmente explícitos de larga escala.
5. Relatórios contendo análises de desempenho da utilização do TerraME HPA na execução de modelos reais, entre eles, modelos que não podem ser eficientemente executados na atual versão de alto desempenho do TerraME.

A principal meta deste projeto é reduzir, tanto quanto possível, o tempo de execução de modelos espaciais dinâmicos desenvolvidos através do uso do ambiente de modelagem TerraME.

3.2 Objetivos específicos

- Estender o núcleo do TerraME, permitindo que os dados de um modelo possam estar compartilhados.
- Contribuir para a consolidação das linhas de pesquisa em Modelagem de Sistemas Terrestres, Banco de Dados e Computação de Alto Desempenho do Departamento de Computação da UFOP.

4 Metodologia

Esta sendo utilizada a metodologia de desenvolvimento de software em espiral com prototipação de realize e versões. Assim, as etapas de concepção, projeto, implementação, teste e documentação serão executadas de forma cíclica, e cada ciclo uma nova versão documentada do aplicativo para planejamento de rotas será obtida.

As seguintes etapas devem ser realizadas para o desenvolvimento.

1. **Concepção de software:** Nesta fase, são definidas as questões que a arquitetura TerraME HPA deve responder. Os trabalhos correlatos serão investigados. As informações disponíveis serão analisadas na esperança de que sejam identificados todos os requisitos de software da arquitetura. O documento de especificação será elaborado nesta etapa.
2. **Projeto de software:** Nesta etapa, serão projetados as estruturas de dados, os algoritmos, as interfaces com o usuário que permitirão ao modelador simular em tempo razoável modelos ambientais em larga escala. Nesta etapa, também será projetada a arquitetura de software do sistema TerraME HPA. Os diagramas de classe, de sequência e de transição de estados, da metodologia UML serão utilizados para a documentação das decisões de projeto. O documento de projeto será elaborado nesta fase.
3. **Implementação de software:** Nesta etapa, cada módulo de software projeto na fase etapa anterior será implementado na linguagem de programação C++. O código fonte da arquitetura TerraME HPA será documentado através do sistema de documentação DOXYGEN ou similar.
4. **Teste de software:** Cada módulo de software implementado na etapa anterior será avaliado quanto sua corretude e desempenho de forma separa. O uso integrados desses componentes será avaliado através sua aplicação para a simulações de modelos ambientais de larga escala e em múltiplas escalas. Os relatórios de desempenho da arquitetura deverão ser elaborados durante esta etapa.
5. **Documentação de software:** Nesta etapa, são escritos artigos e relatórios sobre os avanços científicos e tecnológicos gerados pela pesquisa, o manual do usuário, o guia de programação, e os tutorias da arquitetura TerraME HPA baseada em dados distribuídos.

5 Atividades Desenvolvidas

Nos primeiros meses do projeto, houve um foco em pesquisar simuladores [9] [6] [11] [12], bibliotecas, métodos e trabalhos correlatos que pudessem auxiliar, na implementação da solução paralela *Shared Memory* para o *Kernel* do TerraME.

Sabendo que a parte fundamental do TerraME é o interpretador. E que ele é responsável por ler um programa escrito na linguagem de modelagem TerraME, que por sua vez é uma linguagem estendida da linguagem LUA, que interpreta o código fonte, e chama as funções da plataforma TerraME. A plataforma TerraME é um conjunto de módulos escritos em C++, e como já mencionado anteriormente ela oferece funções e classes para modelagem espacial dinâmica e acesso ao banco de dados espacial *TerraLib* [5].

Para a implementação do TerraME foi utilizado recursos da biblioteca QT e da biblioteca de integração *LUA to C*. Logo os primeiros caminhos buscados para auxiliar na paralelização do *Kernel* TerraME, foi pesquisar (estudar) a fundo ambas bibliotecas apresentadas acima. Inicialmente tentamos resolver a paralelização dos modelos em nível LUA (No mesmo nível do modelador, utilizando apenas recursos da linguagem LUA). Para tal solução o conceito de *co-routines* seria utilizado, no entanto tal solução não foi possível ser aplicada, pois o recurso de *co-routine* disponível em LUA, não é executado concorrentemente (difere do conceito de *thread's*), *co-routines* cooperam entre si e *thread's* concorrem entre si por recurso computacional.

Durante as fases de compreensão e estudos para traçar a solução *Shared Memory* para este problema, observamos alguns inconvenientes pelos quais usuários do TerraME passavam, ao realizar as calibrações de seus modelos utilizando o TerraME. O problema ocorre pelo seguinte fato. Os métodos de calibração de modelos existentes no TerraME (método de Monte Carlo e Algoritmo Genético) demandam um tempo demasiadamente grande, para realizar a calibração dos modelos, e isso muitas é um empecilho para o modelador, que aguarda dias, semanas e até mesmo meses, para obter o resultado final da calibração do modelo. Analisando o problema mais a fundo foi proposto a solução deste problema, pois efetuando a solução do mesmo, teríamos um maior conhecimento do ambiente de trabalho (*Kernel* TerraME) e familiarizaríamos com as tecnologias utilizadas no *Kernel* do TerraME (podemos citar como principais elementos, biblioteca QT e biblioteca de integração *LUA to C*).

Ao solucionar o problema citado acima, obtemos os seguintes resultados:

1. Foi construído um módulo para o TerraME, que proporciona a opção de realizar a calibração de modelos desenvolvidos no mesmo de forma paralela. Vale ressaltar que houve uma redução no tempo de espera para o resultado final da calibração. Utilizando uma máquina com 8 cores, houve uma redução no tempo de execução de 55% ao realizar a comparação com a versão sequencial.
2. Maior compreensão do ambiente de trabalho (*Kernel* TerraME) e das tecnologias para construir o mesmo (bibliotecas QT e de integração LUA to C).
3. Com os resultados dessa solução, um artigo (*Parallel Calibration of Spatial Dynamic Models in TerraME*) foi submetido a *IADIS International Conference APPLIED COMPUTING*, Rio de Janeiro, Brazil, 6 - 8 Novembro 2011(Qualis B3). para uma possível publicação.

Em um segundo momento, aproveitando recursos providos das biblioteca QT e LUA, um outro caminho para a solução foi traçado. Utilizar *Lua_State* (biblioteca *lua to c*) e *QThread* (biblioteca QT), para possibilitar a paralelização de cada função do modelo.

Abaixo os recursos que possibilitaram traçar a solução que esta sendo desenvolvida atualmente:

1. **Lua_State:** Estrutura da biblioteca *Lua to C*, que mantém todo o estado do interpretador LUA. Todas as informações sobre um estado (um modelo por exemplo) são mantidas nessa estrutura. Todas as variáveis presentes nos modelos podem ser acessadas acessando a pilha de execução interna dessa estrutura.
2. **QThread:** QThread é um recurso da biblioteca QT. Ela representa um segmento individual de controle, dentro de um programa que compartilha dados com os outros segmentos do processo. A execução deste segmento é de forma independente (QThread's concorrem entre si por recurso computacional). QThread exige um método virtual *run()*.

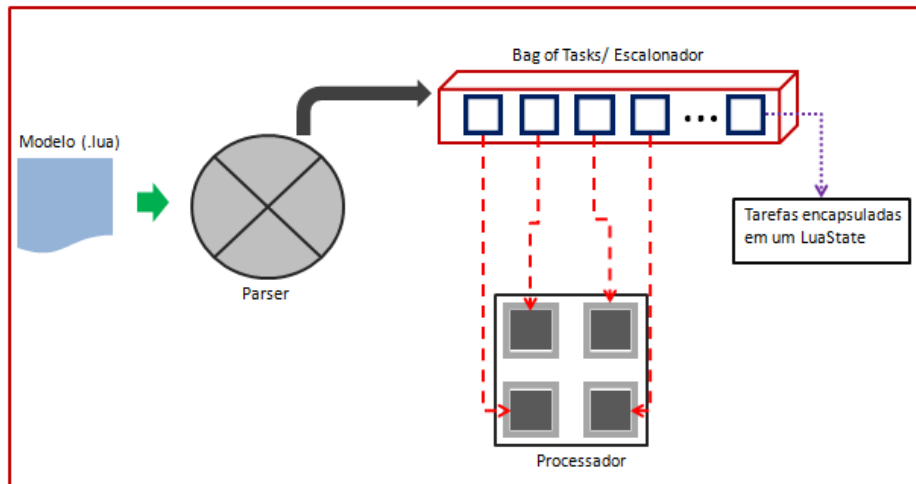


Figura 2: Desenho de solução atual.

Na Figura 2, é apresentado o desenho da solução que esta sendo implementada atualmente. Como primeiro dado de entrada, temos um arquivo com extensão *.lua*, que corresponde ao modelo implementado pelo usuário. Na fase seguinte, o modelo passa por um *parsing*, que é responsável por identificar partes do modelo a serem executadas em paralelo [10]. Após a fase de *parsing*, cada parte dividida deve ser encapsulada por uma estrutura *lua_state* (recurso da biblioteca *Lua to C*). Cada *lua_State* fica armazenado em um escalonador de atividades (tarefas), que têm como objetivo distribuir as tarefas (*lua_State's*) entre os *cores* da máquina (utilizando esta estratégia, reduzimos ao máximo o tempo de ociosidade do processador). Cada tarefa é alocada para um core (thread *QThread*). Assim que a tarefa é alocada, a *thread* é terminada, o escalonador aloca a próxima tarefa armazenada para esta *thread*. Se não houver tarefa a ser executada no momento, a *thread* fica em um estado de espera (*sleep*), até que uma nova tarefa seja enviada para o escalonador. Quando isto ocorre, o escalonador

inicia/reinicia (*start*, alerta) as thread's que estavam no momento ociosas (dormindo) para entrarem no estado de execução.

Já concluímos a implementação do Parser, do divisor/encapsulador de tarefas e do escalonador de tarefas. Atualmente estamos tornando o TerraME HPA robusto, para isso estamos executando diversos modelos e identificando possíveis dados que até então não são tratados pelo nosso módulo. Um destes modelos é o THROLL [4], este modelo tem como base o crescimento florestal, ele foi projetado para funcionar com grandes parcelas de florestas naturais, com especial ênfase para florestas neotropicais. É possível também observar a competição por luz entre as árvores dentro do campo de modelagem. Este modelo é de especial interesse para nossos testes, devido a sua complexidade computacional (o modelo gasta em torno de 90 dias para ser executado), e seu número elevado de variáveis (aproximadamente 75 variáveis, grande parte são de matrizes), possuindo 1800 linhas de código.

```
speciesGrouping = {
  ["LD1"] = { 21, 4.8, 0.08, 4, 5, 150, 60, 2.24, 0.07, 5, 1.76, 3},
  ["LD2"] = { 104, 4.8, 0.08, 2.5, 10, 120, 50, 4, 0.2, 15, 3.5, 3},
  ["LD3"] = { 52, 5, 0.08, 2, 15, 105, 40, 6.4, 0.4, 25, 6.1, 3},
  ["LD4"] = { 44, 5, 0.1, 2, 20, 75, 40, 9.6, 0.6, 40, 7.7, 3},
  ["I1"] = { 13, 6, 0.12, 2.2, 12, 60, 40, 1.76, 0.1, 5, 2.1, 2},
  ["I2"] = { 72, 5.8, 0.12, 1.4, 20, 50, 40, 2.46, 0.22, 14, 3.58, 2},
  ["I3"] = { 83, 5.6, 0.12, 1.4, 25, 40, 40, 4.48, 0.4, 23, 5.3, 2},
  ["I4"] = { 96, 6.4, 0.12, 1.4, 40, 24, 40, 8.96, 0.8, 40, 6.9, 2},
  ["ST1"] = { 151, 8, 0.22, 1.4, 20, 32, 20, 3.36, 0.3, 5, 5.3, 1},
  ["ST2"] = { 148, 7, 0.2, 1.2, 30, 16, 20, 4.8, 0.5, 15, 7.5, 1},
  ["ST3"] = { 116, 7, 0.2, 1, 40, 10, 20, 7.2, 0.9, 25, 11.3, 1},
  ["ST4"] = { 122, 7, 0.2, 1, 50, 8, 20, 9.6, 1.2, 50, 10.1, 1}
}
```

Figura 3: Exemplo de tabela utilizada no modelo THROLL.

```
sizeClassTable[ class ][specie][i] = 0;
```

Figura 4: Exemplo de inserção em uma das estruturas do modelo THROLL (tabela tripla).

Para auxiliar no reconhecimento de tipos definidos em LUA pelo usuário, estamos adicionando a biblioteca luabind[8], ao projeto TerraME. Esta biblioteca, nos auxiliará na ligação entre LUA e C++. Ela tem a capacidade de expor as funções e classes, escritas em C++ à LUA.

6 Desenvolvimento

Até o presente momento, já foram realizadas 4 etapas do projeto TerraME HPA. São elas: Revisão Bibliográfica, Estudo aprofundado do ambiente, Estudo aprofundado da Biblioteca, Elaboração do projeto do sistema.

Na Revisão Bibliográfica, identificamos as principais características do TerraME e seus diferenciais em comparação com as demais plataformas existentes no mercado. Após a revisão, foi possível traçar os primeiros passos a serem realizados, para a construção do novo Módulo TerraME HPA.

O estudo profundo do ambiente de trabalho e o estudo aprofundado da Biblioteca foram ambos realizados, ao solucionar o problema de calibração de modelos no TerraME (problema citado na Seção 5).

Estamos na fase de elaboração do projeto do sistema, uma estratégia para a solução já esta traçada e atualmente estamos tornando o TerraME robusto, adicionado poder de reconhecimento de novos tipos definidos pelo usuário em LUA.

7 Trabalhos Futuros

Para trabalhos futuros, as seguintes atividades devem ser desenvolvidas.

1. Adicionar a biblioteca *luabind* ao projeto TerraME, e resolver todas as dependências envolvidas.
2. Após à conclusão da implementação do Módulo TerraME HPA, a etapa de testes para verificar a corretude da solução deve ser realizada.
3. Redigir monografia (trabalho de conclusão de curso).
4. Apresentação da monografia
5. Conceber um tutorial, para demonstrar as novas funcionalidades do módulo TerraME HPA.
6. Publicação de um artigo, apresentando o módulo TerraME HPA

8 Cronograma de atividades

Na Tabela 1, são apresentadas as atividades a serem realizadas para a conclusão do trabalho.

1. Revisão Bibliográfica
2. Estudo aprofundado do ambiente
3. Estudo aprofundado da Biblioteca
4. Elaboração do projeto do sistema
5. Implementar a interface do sistema
6. Teste comparativo de rendimento
7. Teste do módulo
8. Teste integrado do sistema
9. Elaboração da documentação do sistema
10. Elaboração do Manual do Usuário
11. Redigir a Monografia
12. Apresentação do Trabalho

Atividades	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov	Dez
1	O.K									
2	O.K	O.K								
3		O.K	O.K							
4				O.K	O.K					
5						O.K	X			
6							X			
7							X			
8								X		
9								X		
10								X	X	
11								X	X	X
12										X

Tabela 1: Cronograma de Atividades.

Referências

- [1] et.al Ananth Grama. Introduction to parallel computing (2nd edition).
- [2] Tiago G.S. Carneiro. Nested-ca: a foundation for multiscale modeling of land use and land change. phd thesis in computer science. 2006.
- [3] A. P. L. Carvalho. Grandes desafios de pesquisa em ciência da computação no brasil. march 2006.
- [4] Jérôme Chave. Study of structural, sucessional and spatial patterns in tropical rain forest using troll, a spatially explicit forest model. *Ecological Modeling*, (124):233–254, 1999.
- [5] R. Souza et al Câmara, G. Terralib: Technology in support of gis innovation. ii brazilian symposium on geoinformatics. *GeoInfo*, 2000.
- [6] R. Costanza and A. Voinov. Modeling ecological and economic systems with stella. *Ecological Modelling*, (143):1 – 7, 2001.
- [7] Ministério da Saúde. Abordagens espaciais na saúde pública. 2006.
- [8] Arvid Norberg Daniel Wallin. Luabind. *Software Development and Consulting*, 2003.
- [9] R. Burkhart et al Minar, N. The swarm simulation system: A toolkit for building multi-agent simulation. *SFI Working Paper*, 1996.
- [10] Paul Breimyer Yekaterina Shpanskaya Nagiza F. Samatova Neil Shah, Guruprasad Kora. pr: Enabling automatic parallelization of data-parallel tasks and interfacing parallel computing libraries in r with application to fusion reaction simulations. 2011.
- [11] G. C. Cerqueira et al Soares, B. S. Dinamica - a stochastic cellular automata model designed to simulate the landscape dynamics in an amazonian colonization frontier. *Ecological Modelling*, (154):217–235.
- [12] Vensim. Ventana Systems. Vensim.
- [13] S. Wolfram. New kind of science.