COMPUTAÇÃO DE ALTO DESEMPENHO NA LINGUAGEM R

Paulo Ricardo Rodrigues de Souza Júnior
Acadêmico do curso de Ciência da Computação – Universidade de Passo Fundo
119711@upf.br
Henrique Gavioli Flôres
Acadêmico do curso de Ciência da Computação – Universidade de Passo Fundo
119694@upf.br
Willingthon Pavan
Professor/Pesquisador do curso de Ciência da Computação – Universidade de Passo Fundo
pavan@upf.br
Carlos Amaral Hölbig
Professor/Pesquisador do curso de Ciência da Computação – Universidade de Passo Fundo
holbig@upf.br

Resumo. *Uma das tecnologias que estão sendo mais utilizadas atualmente para implementar modelos de simulação de crescimento e desenvolvimento de culturas é a linguagem R. Devido a crescente complexidade destes modelos que, em certos casos, levam horas ou até dias para serem executados, este artigo visa apresentar alternativas para se obter o alto desempenho em programas escritos em R visando, em um segundo momento, aplicar estas alternativas em modelos de simulação de culturas de plantas e doenças já utilizados na área agrícola.*

Palavras-chave: *Alto desempenho. Linguagem R. Modelos de simulação*

1. INTRODUÇÃO

A linguagem R é um projeto *open source* que está disponível para a maioria das plataformas computacionais. Além de ser uma linguagem de programação também é um ambiente para computação estatística, modelagem e visualização de dados. Trata-se se uma suíte de softwares integrados que proporcionam facilidades na manipulação de dados, no uso de funções estatísticas e na geração de gráficos (ADLER, 2012; R-PROJECT, 2012).

Os modelos de simulação do crescimento e desenvolvimento de culturas têm sido usados com sucesso ao redor do mundo na agricultura para aumentar a produtividade e reduzir custos (PAVAN, 2007). A linguagem R é uma das tecnologias que estão sendo utilizadas atualmente para implementar estes modelos. Estes modelos apresentam um constante aumento de dados tornando os problemas muito complexos e demandando um grande esforço computacional, o que eleva muito o seu tempo de processamento. Para ser viável trabalhar com este grande número de dados é cada vez mais importante o desenvolvimento de programas otimizados e que, com a utilização de pacotes, proporcionem a obtenção de um alto desempenho computacional. Devido a estes fatores, este trabalho foca o uso da linguagem R em ambientes computacionais paralelos e, em um segundo momento, estudará a paralelização de modelos de simulação de culturas implementados em R utilizando os pacotes avaliados nesta pesquisa. Na seção 2 é apresentado um estudo sobre a computação de alto desempenho na linguagem R; na seção 3 são apresentados testes e resultados a respeito do uso de alguns destes pacotes paralelos e, por fim, na seção 4 são apresentados os resultados desta pesquisa e

os trabalhos futuros que serão desenvolvidos.

2. COMPUTAÇÃO DE ALTO DESEMPENHO EM R

Nesta seção são apresentados alguns pacotes para a linguagem R que possibilitam a sua execução em ambientes computacionais paralelos (clusters e multicore), além do desenvolvimento de programas que possam utilizar as funcionalidades das placas gráficas (GPU). Além dos pacotes paralelos, o uso de funções compiladas e o desenvolvimento de funções "rápidas" no R possibilitam que se obtenha um melhor desempenho na execução de seus programas.

2.1 Computação Paralela em R

A linguagem R apresenta diversos pacotes que possibilitam a sua paralelização (SCHMIDBERGER, et al., 2009). Existem pacotes para Clusters, Grids, para o uso em GPUs e para máquinas multicore, conforme apresentado na Tabela 1. Por existirem vários pacotes para programação paralela em R foi realizado um estudo para analisar alguns destes pacotes e decidir quais os mais adequados para serem utilizados na paralelização dos programas em R e em uma futura paralelização dos modelos de simulação trabalhados nesta pesquisa. Uma lista atual destes pacotes poderá ser encontrada na página da CRAN Task View: High-Performance and Parallel Computing with R (http://cran.r-project.org/web/views/HighPerformanceComputing.html), que é a página da entidade que disponibiliza o R e seus pacotes oficiais.

Os principais pacotes utilizados em clusters de computadores são Rmpi, Rpvm e RHadoop. O RPVM (R for Parallel Virtual Machine) é projetado para permitir que uma rede Unix heterogênea ou máquinas Windows sejam usadas como um único computador paralelo distribuído. O RPVM é complexo de ser utilizado por valer-se de funções de baixo nível. RMPI (R *for Message-Passing Interface*) é um sistema padronizado e portátil de transmissão de mensagens em computação paralela, fornecendo uma interface R para funções MPI de baixo nível. Desta forma, o utilizador R não precisa conhecer os detalhes das implementações de MPI. O RHadoop é um pacote que integra o R as funcionalidades do Hadoop, que é um sistema baseado no modelo Map/Reduce e é voltado para a manipulação de grande quantidades de dados (ADLER, 2012).

Tabela 1. Visão geral sobre os pacotes para computação paralela em R

Pacote	Descrição
rpvm	Interface R para PVM (*Parallel Virtual Machine*)
Rmpi	Interface (Wrapper) para MPI (*Message-Passing Interface*)
RHadoop	Pacote que integra o R ao Hadoop
snow	*Simple Network of Workstation*
foreach	Construtor foreach para R
doMC	foreach adaptado para o pacote multicore
doSNOW	foreach adaptado para o pacote snow
doMPI	foreach adaptado para o pacote Rmpi
fork	Funções para manipulação de múltiplos processos
multicore	Código para processamento paralelo do R em máquinas con múltiplos cores ou CPUs
gridR	Executa funções em hosts remotos, clusters ou grids
gputools	Aalgoritmos de mineração de dados implementados usando uma mistura da linguagem CUDA e da biblioteca cublas
magma	Biblioteca de classes e métodos para o processamento paralelo de operações matriciais em GPU´s

Para ambientes computacionais com processadores multicore os principais pacotes são Fork, Multicore e o foreach. O pacote Fork utiliza basicamente os recursos do sistema UNIX para efetuar a paralelização. Possui uma utilização relativamente simples por ter poucas funções mas não apresenta suporte para funções de alto nível como a função aplly. O pacote multicore apresenta além das chamadas de funções do sistema UNIX, outras rotinas próprias. Sua utilização é mais complexa por apresentar mais funções, porém tem suporte para funções de alto nível. O pacote foreach dá suporte para a construção do loop foreach (similar ao comando for) que é uma expressão que permite a iteração sobre os elementos de uma coleção de dados (em sequencial ou em paralelo), sem a utilização de um contador de ciclo explícito.

Para ambientes multicore, os pacotes Multicore e foreach fornecem melhores soluções pois reunem mais recursos que o pacote Fork. A partir da versão 2.14.0 a linguagem R oferece suporte direto ao paralelismo com a disponibilização do pacote "parallel" que incorpora cópias (ligeiramente revisadas) dos pacotes multicore e snow (mas excluindo clusters MPI, PVM e NWS). Além destes pacotes para clusters e multicore há pacotes para grids, GPUs e pacotes para aplicações específicas, todos disponíveis no site da CRAN.

2.2 Uso de funções compiladas

O código da linguagem R é interpretado quando é executado, ao contrário de algumas outras linguagens de programação. Esta é uma razão do porque as funções escritas em C são muitas vezes mais rápidas que as funções escritas em R. Com o uso da biblioteca "compiler" é possível tornar funções, em alguns casos, mais rápidas. Para fazer o uso de funções compiladas em C em programas em R é utilizada a função cmpfun(). Além das funções compiladas acessadas pelo pacote "compiler", o R possui o pacote chamado "Rcpp", o qual proporciona a integração de funções de R com rotinas escritas em programas em C++. Neste caso, os tipos de dados do R são associados a objetos no C++ em uma hierarquia de classes, onde cada tipo é mapeado com a sua classe dedicada. Os atributos do "Rcpp" fornecem uma sintaxe de alto nível para declarar funções do C++ chamadas pelo R, gerando automaticamente o código necessário para utilizá-lo.

2.3 Ferramentas e opções para escrita de funções "rápidas"

Em R existem algumas alternativas para a escrita de funções "rápidas". Estas alternativas abordam aspectos de vetorização de funções e o uso de estrutura de dados mais simples (ROSS, 2013).

A vetorização no R é um recurso muito importante, pois uma função vetorizada não funciona em apenas um valor mas sim em todo um vetor ao mesmo tempo, o que torna mais fácil a escrita do código. É natural o uso de laços de repetição para a modificação de valores de um vetor, o que não é necessário com o uso das funções vetorização no R. Um exemplo do uso de vetorização é a função sum(), que retorna a soma dos valores de um vetor, assim evitando a necessidade de usar um laço para todo o processo da soma. Grande parte das funções em R são vetorizadas e geralmente são implementadas em C sendo, por isso, mais rápidas do que o uso tradicionais com laços de repetição.

A linguagem R tem várias maneiras de interagir com objetos, listas ou dataframes, estruturas mais flexíveis e que podem armazenar vários tipos de dados de uma só vez. Mas esta flexibilidade tem um custo, pois estruturas de dados que possuem apenas um tipo de dados são mais rápidas de serem manipuladas do que, por exemplo, um dataframe. Por isso é importante que estes tipos de estruturas mais complexas sejam

utilizadas com cuidado e que, na medida do possível, se opte pelo uso de estruturas mais simples.

3. TESTES E RESULTADOS

Os testes realizados até o momento para esta pesquisa foram desenvolvidos no grupo de pesquisa ComPaDi da Universidade de Passo Fundo. O computador utilizado possui um processador Intel Core i7 920, que opera à frequência de 2.66 Ghz, com 8 MB de cache L2, 8 GB de memória RAM, sistema operacional Ubuntu 12.04 64 bits e placa de vídeo GeForce GTS250 1GB DDR3 ECS. Os softwares utilizados foram a linguagem R (versão 3.0.1 de 64 bits), a IDE RStudio e os pacotes multicore, foreach, snow, doSNOW, iterators, parallel e gputools.

O primeiro teste abordou o uso do pacote multicore com a paralelização da função mapply do R utilizando a função paralela equivalente mcapply. Este teste utilizou apenas os quatro cores físicos do computador. Na execução sequencial o tempo de execução foi de 1.407 segundos e o tempo paralelo de 0.351 segundos.

O segundo teste abordou a paralelização de laços de repetições (comando for) por meio da função paralela equivalente foreach (utilizando o parâmetro %dopar%), disponibilizada pelo pacote foreach. Neste teste foi criada uma função estatística no R e o programa chamava esta função 100 vezes. Com o uso do laço em sequencial, o tempo de execução foi de 40.46 segundos, já em paralelo, o tempo de execução utilizando os oito processadores foi de 19.17 segundos.

O último teste abordou a realização de operações matriciais (uma multiplicação de matrizes de ordem 8192) utilizando a GPU da máquina. Para isso foi utilizado o pacote gputools com a função gpuMatMult. O tempo de execução sequencial foi de 139.1354 segundos e o utilizando a GPU foi de 11.1562 segundos.

É importante destacar que a linguagem R utiliza implicitamente para as operações de álgebra linear as rotinas da biblioteca BLAS

4. CONCLUSÕES E TRABALHOS EM ANDAMENTO

Com os modelos de simulação apresentando cada vez mais dados é imprescindível encontrar formas de otimizar o seu desempenho. Para tanto, a paralelização mostra ser uma das alternativas, pois pode melhorar efetivamente o tempo de execução dos programas. Os pacotes "multicore", "foreach" e "gputools" da linguagem R vêm ao encontro das necessidades dos modelos de simulação viabilizando a programação concorrente destes modelos. Atualmente, como consequência do uso destes pacotes, está se trabalhando na otimização e paralelização de modelos de doenças que atacam a cultura do morango e do trigo e que fazem parte de uma parceria entre a Universidade de Passo Fundo, a Embrapa Trigo e a Universidade da Flórida.

5. REFERÊNCIAS

ADLER, J. **R in a Nutshell.** 2. ed. EUA: O'Reilly, 2012.

PAVAN, W. **Técnicas de engenharia de software aplicadas à modelagem e simulação de doenças de plantas.** Tese (Doutorado em Agronomia) - Universidade de Passo Fundo, Passo Fundo. 2007.

R-PROJECT. **R Project for Statistical Computing.** Disponível em: <http://www.r-project.org/>. Acesso em 7 jan. 2012.

SCHMIDBERGER, M. et al. State of the Art in Parallel Computing with R. In **Journal of Statistical Software.** Vol. 31, Issue 1, p. 1-27, 2009.

ROSS, N. **FasteR! HigheR! StongeR! – A Guide to Speeding Up R Code for Busy People.** Disponível em: <http://www.r-bloggers.com/faster-higher-stonger-a-guide-to-speeding-up-r-code-for-busy-people/>. Acesso em 20 mai. 2013.