Computação de alto desempenho em R: paralelização e técnicas de otimização

Angela Mazzonetto, Prof. Dr. Carlos Amaral Hölbig

Programa de Pós-Graduação em Computação Aplicada – PPGCA
Universidade de Passo Fundo (UPF) 99.001-970 - Passo Fundo – RS – Brasil

144981@upf.br, holbig@upf.br

Abstract. *The processing and analysis of large amounts of data are present in several areas such as biology, chemistry, physics, statistics, geography, among others, a fact that can make the task computationally complex and exhaustive. The R language is an efficient computational tool to perform these kinds of tasks, allowing them to be performed in high-performance computing environments, aiming its parallelization and consequently a better computational performance. Because of this, this work aims at the study and demonstration of implementations that demonstrate that the use of R language is a feasible alternative when working with large amounts of data and to obtain efficient and quick responses when of the resolution of the most various kinds of computational models and simulations.*

Resumo. *O processamento e análise de grande quantidade de dados estão presentes nas mais diversas áreas como biologia, química, física, estatística, geografia, entre outras, fato este que pode tornar a tarefa computacionalmente complexa e exaustiva. A linguagem R é uma ferramenta computacional eficiente para realizar estes tipos de tarefas, possibilitando, ainda, que elas sejam realizadas em ambientes computacionais de alto desempenho, objetivando sua paralelização e, consequentemente, um melhor desempenho computacional. Por este motivo, este trabalho visa o estudo e a demonstração de implementações que comprovam que a utilização da linguagem R é uma alternativa viável quando da necessidade de se trabalhar com grandes quantidades de dados e de se obter respostas eficientes e rápidas quando da resolução dos mais diversos tipos de modelos e simulações computacionais.*

1. Introdução

Inúmeras organizações atualmente têm a necessidade de realizar o processamento e análise de uma grande quantidade de dados em tempo computacional hábil. Por este motivo a utilização de ferramentas computacionais torna-se indispensável para a realização destas atividades.

Uma ferramenta que pode ser utilizada para suprir estas necessidades é a linguagem R. De acordo com Torgo (2009), o R é uma linguagem de programação *open source* e um ambiente para computação estatística, modelação e visualização de dados. Trata-se de uma linguagem de programação especializada em análise de dados. Além disso, está disponível para uma vários sistemas operacionais, tais como Linux, Unix, Windows e MacOS. Outra grande vantagem desta linguagem é a grande disponibilidade de pacotes, ferramentas, bibliotecas e funções que possibilitam, entre

várias funcionalidades, o processamento paralelo das aplicações nela desenvolvidas e a otimização de seus programas por meio do uso de funções especiais e de códigos compilados em outras linguagens de programação.

Este artigo visa apresentar algumas técnicas e ferramentas da linguagem R que possibilitam o processamento de grandes quantidades de dados com mais eficiência e em ambientes computacionais de alto desempenho.

2. Computação de alto desempenho na linguagem R

Na linguagem R foram desenvolvidos inúmeros pacotes que possibilitam a computação de alto desempenho, pacotes voltados para os mais diversos ambientes computacionais como, por exemplo, para grids de computadores (`GridR`), para clusters (`RPVM`, `Rmpi`, `snow`, `snowFT`, `snowfall`, `papply`, `taskPR`, `foreach`, `doMC`, `doSnow`, `doMPI` e `Rdsm`), para computadores multicore (`fork` e `multicore`) e para GPUs (`gputools`, `magma` e `HiPLARM`). Além deles pode-se citar os pacotes `parallel` (uma suíte de vários dos pacotes citados acima em um único pacote do R) e o `RHadoop` (pacote do R que possibilita a sua integração com a ferramenta Apache Hadoop, que é um *framework* que permite o processamento distribuído de grandes conjuntos de dados em clusters de computadores, usando modelos de programação simples). Uma lista atual destes pacotes poderá ser encontrada na página da CRAN *Task View: High-Performance and Parallel Computing with* R[5], que é a página da entidade que disponibiliza o R e seus pacotes oficiais. Detalhes sobre estes pacotes podem ser obtidos em Schmidberger (2009), Eugster (2011) e McCallum and Weston (2011).

Alternativas para se obter um melhor desempenho com a linguagem é o uso de funções compiladas e de funções vetorizáveis. 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 do pacote `compiler` é possível tornar as 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++.

Além disso, 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. 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 vetorizadas no R. Um exemplo do uso da vetorização é a função `sum()`, que retorna a soma dos valores de um vetor ou matriz evitando, assim, 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.

[5] http://cran.r-project.org/web/views/HighPerformanceComputing.html

3. Testes e resultados

Com o intuito de validar a pesquisa realizada neste trabalho, alguns testes foram desenvolvidos no grupo de pesquisa ComPaDi da Universidade de Passo Fundo. O ambiente computacional foi composto por um computador com 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 14.04 64 bits e placa de vídeo GeForce GTS250 1GB DDR3 ECS. Os softwares utilizados foram a linguagem R (versão 3.1.1 de 64 bits), a IDE RStudio e o pacote `foreach` (1.4.1).

A Figura 1 apresenta um programa que realiza o cálculo sequencial da soma entre duas matrizes de 2000 elementos cada. Na linha 10 a soma é realizada e esta operação resultou em um tempo computacional de 0.011s. Posteriormente, utilizando a função vetorizada `sum` (linha 23), o tempo de execução ficou em 0.005s. Este fato demonstra a importância do uso das funções vetorizadas disponibilizadas pelo R.

```
1. ordem = 2000
2. S  = array(0, dim = c(ordem, ordem))
3. S1 = array(0, dim = c(ordem, ordem))
4.      for(i in 1:ordem)
5.      for(j in 1:ordem)
6.      {
7.      A[i,j] = round(runif(1)*10);
8.      B[i,j] = round(runif(1)*10);
9.      }
10.     system.time(S<-A+B)
11.     #  usuário   sistema decorrido
12.     #    0.012     0.000     0.011
13.
14.     system.time(
15.     for(i in 1:ordem)
16.     for(j in 1:ordem)
17.     {
18.     S1[i,j] <- A[i,j]+ B[i,j];
19.     })
20.     #  usuário   sistema decorrido
21.     #    9.112     0.019     9.134
22.
23.     system.time(s<-sum(A))
24.     #  usuário   sistema decorrido
25.     #    0.005     0.000     0.005
```

Figura 4. Programa em R com funções de otimização.

A Figura 2 apresenta um programa com aplicação do pacote `foreach`. Pode ser observado que o tempo decorrido foi de 1.827s com o laço `foreach` (linha 14) utilizando a opção `%dopar%` (execução em paralelo utilizando 8 processadores). Posteriormente, com um laço `for` simples (linha 17) o tempo foi de 7.197s (execução sequencial).

```
1. library(foreach)
2. require(doSNOW)
3. cl<-makeCluster(8) # numero de cores
4. registerDoSNOW(cl)l
5. # create a function to run in each itteration of the loop
6. check <-function(n) {
7. for(i in 1:1000)
8. {
9. sme <- matrix(rnorm(100), 10,10)
10.     solve(sme)
11.     }
12.     }
13.     times <- 100     # times to run the loop
14.     system.time(x   <-   foreach(j=1:times   )   %dopar%
   check(j))
15.     #  usuário   sistema decorrido
16.     #    0.091     0.008     1.827
17.     system.time(for(j in 1:times ) x <- check(j))
18.     #  usuário   sistema decorrido
19.     #    7.185     0.011     7.197
20.     stopCluster(cl)
```

Figura 2. Programa em R com uso do foreach.

4. Conclusão

Observa-se que o processamento e análise dos dados atualmente é algo imprescindível para as organizações que possuem base de dados com grande quantidade de informações. Por este motivo a linguagem R torna-se uma forte aliada para a criação de modelos de simulação que realizem tarefas relacionadas aos estas informações. Porém, inúmeras vezes, com apenas as funções básicas desta linguagem não é o suficiente para que o processamento dos dados seja feito em um tempo computacional hábil. Consequentemente surge a necessidade de buscar novos meios de proporcionar este processamento ainda mais eficiente em termos de desempenho computacional. Neste trabalho foram abordadas várias ferramentas para auxiliar o paralelismo em R. Também se realizou alguns testes utilizando a função `sum` própria do R e o pacote `foreach`. Estes testes demonstraram a viabilidade de sua utilização e o ganho em desempenho obtido, concluindo, assim, que a paralelização e a otimização em R são opções viáveis e eficientes quando da execução de aplicações reais de grande porte e com grande quantidade de dados. Como trabalho futuro, esta pesquisa visa a execução paralela de modelos de simulação de culturas e doenças de plantas implementados em R.

Referencias

Schmidberger, M. et al. (2009) "State of the Art in Parallel Computing with R", In: Journal of Statistical Software., v.31, n.1, p. 1-27.

Eugster, M. J. A. et al. (2011) "Hands-on tutorial for parallel computing with R", In: Computational Statistics, v. 26, n. 2, p. 219-239.

McCallum, Q.E. and Weston, S. (2011) “Parallel R”. O’Reilly Media, Inc.

Torgo, L. (2009) “A linguagem R: programação para a análise de dados.” Lisboa: Escolar Editora, p. 203.