Desenvolvimento de uma interface para uma biblioteca “open-source” de algoritmos de programação linear

William Pereira dos Santos Wada[1]
Luís Gustavo Barioni[2]

Introdução

Um grande número de problemas de otimização em diversas áreas de aplicação faz uso da técnica de programação linear (SANTOS, 2003). Na agricultura, a PL tem sido amplamente utilizada, particularmente em problemas de alocação de recursos e planejamento (SOUZA et al., 2008). Além disso, a formulação de dietas para animais constitui-se um problema clássico de aplicação dessa técnica (LANNA et al., 1999).

O desenvolvimento de sistemas de suporte à tomada de decisão (SSD) que façam uso de PL requer a implementação de um algoritmo robusto ou o uso de implementações confiáveis, disponibilizadas por terceiros.

Um grande número de produtos comerciais tem sido desenvolvido para esse fim. Entretanto, o custo da maioria das ferramentas inviabiliza o desenvolvimento e distribuição de SSDs livres, gratuitos ou de baixo custo para agropecuária.

[1] *Faculdade de Tecnologia Universidade Estadual de Campinas; willianpsw@cnptia.embrapa.br*

[2] *Embrapa Informática Agropecuária; barioni@cnptia.embrapa.br*

Nesse sentido, identificou-se a biblioteca LPSolve, a qual possui código aberto, distribuição gratuita, pode ser usada tanto em ambiente Linux quanto Windows e implementa diversos algoritmos de PL. No caso do sistema operacional Windows, os algoritmos são disponibilizados na forma de uma DLL (biblioteca de ligação dinâmica, do inglês Dynamic Link Library).

Visando a facilitar a aplicação da DLL em SSDs, desenvolveu-se uma interface, implementada utilizando-se de conceitos de programação orientada a objetos, descrita neste artigo. Essa interface está sendo incorporada ao SSD Invernada, desenvolvido no Laboratório de Matemática Computacional da Embrapa Informática Agropecuária.

Material e método

Foi utilizada a linguagem Delphi e o paradigma de orientação a objeto (OO). A implementação utilizou os conceitos de interface, ou seja um componente de software que permite o isolamento dos detalhes de implementação do mundo exterior (W3C,1995) evitando o acesso direto à regra do negócio da implementação e permitindo a padronização de código.

As chamadas das funções da DLL LPSolve utilizaram o conceito de tipos procedurais, “um recurso da linguagem Delphi que permite armazenar métodos, funções e procedimentos em variáveis” (CANTÚ, 2003, p.397). Nessas variáveis atribui-se os endereços de memória das rotinas da DLL. Essa técnica é necessária, pois a linguagem Delphi não permite chamadas de funções da DLL como métodos de classe. Além disso, o carregamento da DLL ocorre em tempo de execução, permitindo maior flexibilidade.

Resultados e discussão

Apesar da DLL permitir a resolução de vários problemas simultaneamente, optou-se por desenvolver uma classe de interface de maneira

que cada instância resolva apenas um problema. O problema é identificado por meio do atributo FnumeroProblema que é atribuída pela função make_lp (da DLL LPSolve), a qual inicia a configuração de um problema de PL. Para resolver dois ou mais problemas, a classe deve ser instanciada o número de vezes necessário. Outro atributo da classe, chamado Flibhandle, possibilita a utilização de outras DLLs, pois tem o papel de identificar qual DLL será utilizada, bastando passar como parâmetro da função LoadLibrary o nome da DLL. O suporte para outras DLLs não foi implementado até o momento. A classe de interface ainda gera tratamento de exceções, facilitando a identificação de erros pelo sistema que irá utilizá-la.

Na Figura 1 segue o código responsável pela invocação da rotina set_bounds da DLL LPSolve, extraído da classe de interface desenvolvida.

```
// Declaração de interface
interface

//Declaração do método da DLL pela palavra chave stdcall atribuído para a variável tipo procedural Tser_bounds
TSet_bounds = function(LP:PlongInt;colunas:integer;lower,upper:Real): Boolean; stdcall;


//Assinatura da Função que manipulará a DLL
function LP_Set_Bounds(n_col:integer;min,max:REAl):Boolean;


// Início da função para manipulação da DLL executando a estipulação dos limites para a resolução do sistema
function TLinearProgramming.LP_Set_Bounds(n_col: Integer; min: Real; max: Real):Boolean;
var
//Declaração da variável tipo procedural do tipo function Set_bounds da DLL
 Set_Bounds : TSet_bounds;

begin
//  Início do Bloco de captura de Excessão
 try
   //Apontando o endereço de memória da variável Set_Bounds para o endereço da função da DLL.
   @Set_Bounds := GetProcAddress(FLibHandle,'set_bounds');
   //Resolução utilizando a função da DLL.Como atributos o número para identificação do problema,
   // a coluna desejada para a aplicação dos limites e o intervalo inferior e superior do limite.
   Result:= Set_Bounds(FNumeroProblema,n_col,min,max);
  //Instrução de tratamento de exceção
  except
  //Levantamento da Exceção
    raise Exception.Create('Nao foi possivel estipular limites');
 end;
 // Fim do bloco de captura de Excessão
//Fim da função
end;
```

Figura 1. Trecho responsável pela invocação da rotina set_bounds da DLL LPSolve.

Referências

CANTÚ, M. **Delphi 7.** Marina Village Parkway: Sybex, 2003.

LANNA, D. P.;TEDESCHI, L. O.; BELTRAME, J. A. **modelos lineares e não-lineares de uso de nutrientes para formulação de dietas de ruminantes**. Piracicaba. Disponível em: <http://www.scielo.br/scielo.php?pid=S0103-90161999000200031&script=sci_arttext > Acesso em: 01 Ago 2010.

SANTOS, M. P. **Pesquisa Operacional**, Rio de Janeiro: UFRJ, 2003, Apostila -Departamento de Matemática Aplicada Instituto de Matemática e Estatística .

SOUZA, P. M.; FERREIRA, V. R.; PONCIANO N. J.; BRITO, M. N. **Otimização econômica, sob condições de risco, para agricultores familiares das regiões Norte e Noroeste do Estado do Rio de Janeiro.** Rio de Janeiro: Disponível em: <http://www.scielo.br/scielo.php?pid=S0101-74382008000100007&script=sci_arttext >. Acesso em: 02 Ago 2010.

W3C. A Java-VM Implementation Based on the Modula-3 Runtime (em inglês). Disponível em: <http://www.w3.org/OOP/m3-java.html > Acesso em: 15 jul. 2010.