TEMA Tend. Mat. Apl. Comput., **9**, No. 3 (2008), 395-404.

Aspectos Teóricos de Simulated Annealing e um Algoritmo duas Fases em Otimização Global[1]

G. HAESER[2], M. GOMES–RUGGIERO[3], Departamento de Matemática Aplicada, IMECC, UNICAMP, 13083-859 Campinas, SP, Brasil.

Resumo. Neste trabalho descrevemos a teoria da estratégia *simulated annealing*, e propomos um método híbrido para otimização global de problemas canalizados contínuos. A cada etapa deste método é realizada uma fase heurística, na qual empregamos *simulated annealing* e uma fase local, na qual empregamos o método `GENCAN`. O objetivo deste procedimento é explorar as propriedades de obtenção do ótimo global de *simulated annealing*, e acelerar esta estratégia acionando um procedimento de otimização local que possua boas propriedades de convergência para ótimos locais. Os resultados numéricos obtidos atestam a eficiência deste procedimento.

1. Introdução

Em pesquisa na literatura especializada sobre aplicações de técnicas em otimização contínua, é possível constatar que um grande número de problemas resulta em modelos não–lineares com funções não–convexas. As não–convexidades levam a múltiplos ótimos locais tornando difícil a tarefa de identificar o ótimo global, que é a solução de interesse nestas aplicações. A dificuldade central da busca pelo ótimo global resulta do fato que os algoritmos usuais em otimização dependem fortemente do ponto inicial e, no caso de convergência, é obtido um ponto estacionário, que pode não ser ótimo local do problema, muito menos ótimo global.

Neste trabalho, consideramos os modelos que podem ser formulados como problemas de minimização com restrições de canalização nas variáveis:

$$\min f(x), \quad l \leq x \leq u. \tag{1.1}$$

Propomos um algoritmo que a cada iteração realiza duas fases: fase heurística, na qual aplicamos uma etapa da técnica *simulated annealing* e fase local, na qual acionamos um otimizador local para resolução do problema (1.1), usando como aproximação inicial, a solução obtida na fase heurística. Escolhemos, como otimizador local, o método `GENCAN` [3], proposto e implementado por Birgin e Martínez para otimização de problemas canalizados contínuos. O objetivo com esta estratégia

[1]Trabalho financiado pela FAPESP, processo 03-11695/8
[2]ghaeser@ime.unicamp.br
[3]marcia@ime.unicamp.br

híbrida é construir um método que explore as propriedades de obtenção do ótimo global de *simulated annealing*, sendo que esta estratégia é acelerada por um procedimento de otimização local.

Na Seção 2 descrevemos a técnica *simulated annealing* [1, 7, 11], destacando os aspectos teóricos que a tornam uma estratégia eficiente em problemas de otimização global combinatória. Na Seção 3, apresentamos o algoritmo `SA-GENCAN`, assim denominado por envolver a técnica de *simulated annealing* e o método `GENCAN`. Na Seção 4, relatamos os experimentos numéricos realizados com 28 problemas clássicos com múltiplos ótimos locais [6], e o problema LOVO (*Low Order–Value Optimization*) [2], onde fizemos um ajuste por quadrados mínimos que descarta um número predeterminado de *outliers*. As conclusões são apresentadas na Seção 5.

2. Simulated Annealing: aspectos teóricos

Annealing é o processo utilizado para fundir um metal, onde este é aquecido a uma temperatura elevada e em seguida é resfriado lentamente, de modo que o produto final seja uma massa homogênea. O *simulated annealing* surgiu no contexto da mecânica estatística, desenvolvido por Kirkpatrick, Gelatt e Vecchi em 1983 [7] e independentemente por Černy em 1985 [5], utilizando o algoritmo de Metropolis de 1953 [9].

Esta técnica é utilizada em problemas de otimização combinatória, $\min_x f(x)$, $x \in S$, onde $f : S \to \mathbb{R}$, S finito. Neste contexto, o processo de otimização é realizado por níveis, simulando os níveis de temperatura no resfriamento. Em cada nível, dado um ponto $u \in S$, vários pontos na vizinhança de u são gerados e o correspondente valor de f é calculado. Cada ponto gerado é aceito ou rejeitado de acordo com uma certa probabilidade. Esta probabilidade de aceitação decresce de acordo com o nível do processo, ou equivalentemente, de acordo com a temperatura.

O algoritmo a seguir apresenta os passos deste procedimento. Neste algoritmo, $T_k \in \mathbb{R}^*_+$ representa a temperatura no nível k e L_k o número de pontos que serão gerados neste nível. Inicialmente, T_0 e L_0 são fixados, e um ponto inicial, u, é escolhido em S.

`Inicializar` u, T_0, L_0
$k := 0$
`Repetir`
  `Para` $l = 1$ `ATÉ` L_k
    `Gerar` w `de` $V(u)$
    `Se` $f(w) \leq f(u)$, `então` $u := w$
    `Caso contrário, se` `random`$[0,1) < \exp\left(\dfrac{f(u) - f(w)}{T_k}\right)$
      `então` $u := w$
  $k := k + 1$
  `Calcular` L_k `e` T_k
`Até ‘‘critério de parada’’`

A idéia do algoritmo é inicialmente aceitar quase todas as transições propostas, a fim de escapar de um minimizador local (para isso, T_0 deve ser suficientemente

grande) e em seguida aceitar com probabilidade cada vez menor os pontos que pioram o valor da função objetivo, sendo que no limite $T_k \to 0^+$, só serão aceitos os pontos que melhoram o valor da função objetivo. Para descrevermos a teoria deste algoritmo, é preciso definir as cadeias de Markov e a proposta do algoritmo de Metropolis.

2.1. Cadeias de Markov

Definição 1. *Uma cadeia de Markov é uma seqüência de variáveis aleatórias X_k que assume valores em S tal que:*

$$\mathbf{P}\{X_k = i_k | X_{k-1} = i_{k-1}, \ldots, X_0 = i_0\} = \mathbf{P}\{X_k = i_k | X_{k-1} = i_{k-1}\},$$

além disso, a cadeia é dita finita se S é finito.

Ou seja, em uma cadeia de Markov, a probabilidade de mudarmos de um estado i para um estado j não depende dos estados anteriores. Assim, em uma cadeia de Markov finita, podemos definir para cada k, a *matriz de transição* $P^{(k)}$, onde

$$P_{ij}^{(k)} = \mathbf{P}\{X_k = j | X_{k-1} = i\}.$$

Definição 2. *Dizemos que a cadeia de Markov é homogênea se $P^{(k)}$ não depende de k, neste caso escrevemos apenas P, para a matriz de transição. Caso contrário dizemos que a cadeia é heterogênea.*

Definição 3. *Em uma cadeia de Markov finita e homogênea, dizemos que existe a distribuição limite se, para cada $i \in S$ existe o limite*

$$q_i = \lim_{k \to +\infty} \mathbf{P}\{X_k = i | X_0 = j\},$$

com q_i independente de j. O vetor q é chamado de distribuição limite da cadeia de Markov.

É possível mostrar que sob certas condições, se P é a matriz de transição de uma cadeia de Markov finita e homogênea e o vetor q é uma distribuição de probabilidade em S e $q_i P_{ij} = q_j P_{ji}$, $\forall\, i, j \in S$, então q é a distribuição limite desta cadeia de Markov. Para detalhes veja [6].

2.2. Algoritmo de Metropolis

Uma questão central que precisa ser respondida é a seguinte: "dado um conjunto finito S e uma distribuição de probabilidade q em S, como construir uma cadeia de Markov em S com distribuição limite q?" O algoritmo de Metropolis, responde a esta pergunta. A seguir definimos o algoritmo.

Seja $G \in \mathbb{R}^{n \times n}$ uma matriz simétrica, onde $n < +\infty$ é a cardinalidade de S e cada linha de G é uma distribuição de probabilidade. A matriz G é chamada *matriz de geração* e G_{ij} é a probabilidade de gerar j a partir de i. Sejam $\alpha_{ij} = \min\{q_j/q_i, 1\}$, as entradas da *matriz de aceitação* α. As quantidades α_{ij} definem o chamado *critério de Metropolis*.

O algoritmo consiste em: dado $X_k = i$, gerar j a partir de i de acordo com a destribuição de probabilidade dada pela i–ésima linha da matriz G. Aceitar $X_{k+1} = j$ com probabilidade α_{ij}, ou rejeitar j, isto é, $X_{k+1} = i$ com probabilidade $1 - \alpha_{ij}$. Deste modo X_k é uma cadeia de Markov finita e homogênea com

$$P_{ij} = \alpha_{ij} G_{ij}, \text{ se } i \neq j \text{ e } P_{ii} = 1 - \sum_{j \in S, j \neq i} P_{ij}.$$

Note que no algoritmo de Metropolis não é necessário conhecermos o valor da constante normalizadora da distribuição q, pois o algoritmo só depende de q na razão q_j/q_i.

Usando a simetria de G, temos que, para $i \neq j$:

$$q_i P_{ij} = q_i \alpha_{ij} G_{ij} = q_i \min\{q_j/q_i,\ 1\} G_{ij} = \begin{cases} q_j G_{ij} & \text{se } q_j \leq q_i \\ q_i G_{ij} & \text{se } q_j > q_i \end{cases}$$

$$= \begin{cases} q_i G_{ji} & \text{se } q_i < q_j \\ q_j G_{ji} & \text{se } q_i \geq q_j \end{cases} = q_j \min\{q_j/q_i,\ 1\} G_{ji} = q_j \alpha_{ji} G_{ji} = q_j P_{ji}.$$

Assim, é possível mostrar que a cadeia de Markov construída pelo algoritmo de Metropolis possui distribuição limite q, se q não é uniforme em S e $q_i > 0$ para todo $i \in S$. O *simulated annealing* consiste na aplicação do algoritmo de Metropolis repetidas vezes, onde a cada execução a distribuição limite desejada é modificada.

2.3. Distribuição de Boltzmann

Dados $f : S \to \mathbb{R}$, S finito e $T > 0$, a distribuição de Boltzmann é definida por:

$$q_i(T) = \frac{1}{N_0} \exp\left(-\frac{f(i)}{T}\right), \quad \forall\, i \in S,$$

onde $N_0 = \sum_{i \in S} \exp(-f(i)/T)$ é uma constante normalizadora. A escolha desta distribuição se justifica pela propriedade:

$$\lim_{T \to 0^+} q_i(T) = q_i^* \stackrel{\text{def}}{=} \begin{cases} \frac{1}{|S_{\texttt{opt}}|} & \text{se } i \in S_{\texttt{opt}} \\ 0 & \text{caso contrário} \end{cases},$$

onde $S_{\texttt{opt}} = \{i \in S | f(i) \leq f(j), \quad \forall\, j \in S\}$ e $|S_{\texttt{opt}}|$ é a cardinalidade de $S_{\texttt{opt}}$. Isto é, conforme $T \to 0^+$ a distribuição de Boltzmann tende para uma distribuição uniforme nos minimizadores globais. Note que $q_i(T) > 0$ para todo $i \in S$ e para todo $T > 0$. Assim, obtemos o seguinte resultado:

$$\lim_{T \to 0^+} \lim_{k \to +\infty} \mathbf{P}\{X_k = i\} = \lim_{T \to 0^+} q_i(T) = q_i^*,$$

ou seja,

$$\lim_{T \to 0^+} \lim_{k \to +\infty} \mathbf{P}\{X_k \in S_{\texttt{opt}}\} = 1.$$

Este resultado também se verifica no caso particular em que a distribuição de Boltzmann é a distribuição uniforme em S, pois neste caso f é constante em S e o resultado é trivial.

A idéia do *simulated annealing* é obter $i \in S$ com distribuição q^*. Para isso construímos a cadeia de Markov homogênea com T fixo até atingirmos a distribuição limite, em seguida diminuímos T e restauramos o equilíbrio atingindo novamente a distribuição limite para a nova temperatura. O processo é repetido até que T seja aproximadamente zero.

Em [1] é obtido um resultado de convergência como este, para um algoritmo onde o número de iterações para cada T fixo é finito, com a hipótese adicional de que o decréscimo da temperatura deve ser suficientemente lento.

Surgem então alguns parâmetros em aberto quanto à implementação do algoritmo:

- o valor inicial para a temperatura (T_0);
- uma função para ditar o decréscimo da temperatura;
- um valor final para a temperatura, ou seja, um critério de parada;
- um número finito de transições para cada temperatura (L_k).

Um esquema que especifica os parâmetros acima é denominado um *esquema de resfriamento*.

3. Simulated Annealing com Acelerador Local

Nesta seção descrevemos o método `SA-GENCAN`, para minimização de funções com restrições de canalização. Trata–se de um processo híbrido, no qual acionamos a estratégia *simulated annealing* e o método `GENCAN`.

Em princípio, qualquer método proposto para resolução de problemas do tipo (1.1) pode ser empregado como acelerador local. Optamos por `GENCAN` pelos resultados teóricos, que incluem propriedade de convergência global e pelo bom desempenho computacional, [3]. Este método foi proposto e implementado por Birgin e Martínez em 2002, e é um processo de restrições ativas com gradiente espectral projetado, onde a cada iteração uma aproximação quadrática do problema é resolvido em uma região de confiança.

Em `SA-GENCAN`, a técnica *simulated annealing* é aplicada sem discretização explícita do domínio, isto é, um novo elemento deve ser escolhido da vizinhança do ponto atual, sendo esta vizinhança um subconjunto de $\mathbb{R}^n$ com interior não vazio. Fixado $\varepsilon > 0$, definimos a vizinhança V de cada ponto $\bar{x}$ da caixa, definida pelas restrições $l \leq x \leq u$, como o conjunto dos pontos da caixa que distam no máximo ε de $\bar{x}$, na norma infinito. Um novo ponto é obtido gerando uma variável aleatória uniforme na vizinhança V. Este processo pode ser realizado por componentes já que cada vizinhança também é uma caixa, devido a utilização da norma infinito.

A cada troca de temperatura, empregamos a rotina `GENCAN` usando como aproximação inicial o ponto gerado nesta etapa de *simulated annealing*. A solução ótima obtida por `GENCAN` não é empregada como ponto inicial para o próximo nível de temperatura, ao invés disto, construímos uma lista de soluções encontradas por `GENCAN`, sendo que as iterações seguem com o ponto que possuíamos antes de aplicarmos `GENCAN`. Acreditamos que deste modo estamos sendo mais coerentes com a teoria de *simulated annealing*, pois evitamos mudanças bruscas no valor da função objetivo ao longo da cadeia de Markov gerada.

Executamos GENCAN com os parâmetros padrões, propostos pelos autores, e acrescentamos um novo critério de parada: a cada iteração de GENCAN, calculamos a distância Euclidiana entre o ponto atual e cada ponto da lista de soluções já obtidas por GENCAN nas etapas anteriores de SA-GENCAN. Caso uma destas distâncias seja menor que uma tolerância estabelecida, a execução de GENCAN é interrompida. Se este critério não for acionado, acrescentamos este novo ponto à lista (mesmo que GENCAN não tenha declarado convergência para um minimizador local). O objetivo da lista, além de armazenar a melhor solução, é evitar que GENCAN encontre uma solução já obtida anteriormente.

O esquema de resfriamento implementado para o *simulated annealing* foi o descrito em [10], com os valores típicos para os parâmetros, juntamente com o critério de parada descrito em [1, 4]. Além disto limitamos em 100 o número máximo de trocas de temperatura.

Segue abaixo um esboço do algoritmo:

Inicializar x, T_0, L_0
$k := 0$
Repetir
  Para $l = 1$ até L_k
    Gerar y de $V(x)$
    Se $f(y) \leq f(x)$ então $x := y$
    Caso contrafio, se random$[0,1) < \exp\left(\dfrac{f(x) - f(y)}{T_k}\right)$
      ENTÃO $x := y$
  $k := k + 1$
  Calcular L_k e T_k
  Executar GENCAN a partir de x
Até ‘‘critério de parada’’

4. Experimentos Numéricos

Os experimentos numéricos incluem a resolução do problema LOVO e de um conjunto de 28 problemas clássicos da literatura, de pequeno porte e múltiplos ótimos locais [6], por *simulated annealing* puro e SA-GENCAN. A implementação foi realizada em Fortran 77 em uma máquina Pentium 4, 3.5 GHz, 2Gb Ram.

A seguir definimos o problema LOVO (*Low Order–Value Optimization*), de acordo com [2]. Nossa utilização do problema LOVO será como uma generalização do problema de quadrados mínimos, onde podemos descartar um número predeterminado de *outliers*. Outras aplicações bem sucedidas do problema LOVO se encontram no problema de alinhamento de proteínas e descobrimento de padrões ocultos [2].

Dadas m funções $f_1, \ldots, f_m$ definidas no conjunto $\Omega \subseteq \mathbb{R}^n$ e um inteiro positivo $p \leq m$, definimos as m funções índices $i_1, \ldots, i_m : \Omega \to \{1, 2, \ldots, m\}$ de modo que $\{i_1(x), i_2(x), \ldots, i_m(x)\}$ é uma permutação de $\{1, 2, \ldots, m\}$ e

$$f_{i_1(x)}(x) \leq f_{i_2(x)}(x) \leq \cdots \leq f_{i_p(x)}(x) \leq \cdots \leq f_{i_m(x)}(x), \quad \forall\, x \in \Omega.$$

Assim, podemos definir o problema LOVO de ordem p:

$$\min_x \sum_{j=1}^{p} f_{i_j(x)}(x)$$
$$\text{s.a} \quad x \in \Omega.$$

Em particular, queremos ajustar o modelo $y(t,x) = x_1 + x_2 t + x_3 t^2 + x_4 t^3$ em um conjunto de dados (t_i, y_i), $i = 1, \ldots, m$, desprezando os $m - p$ piores ajustes. As funções f_i são as funções erro $f_i(x) = (y(t_i, x) - y_i)^2$.

Sendo $\Omega = [-3, 3]^4$ e dada a solução $x^* = (x_1^*, x_2^*, x_3^*, x_4^*)$ escolhida aleatoriamente em Ω, $m = 101$ e $p = 80$ definimos $t_i = -1 + 0.02(i-1)$ e $w_i = y(t_i, x^*), \quad \forall\, i = 1, 2, \ldots\, m$.

Escolhemos aleatoriamente $m - p = 21$ elementos de $\{1, 2 \ldots, 101\}$, a saber $p_1, p_2, \ldots, p_{21}$ e definimos

$$y_i = \begin{cases} 10 & \text{se} \quad i \in \{p_1, p_2, \ldots, p_{21}\}, \\ w_i & \text{caso contrário.} \end{cases}$$

A solução deste problema é x^* com valor nulo para a função objetivo.

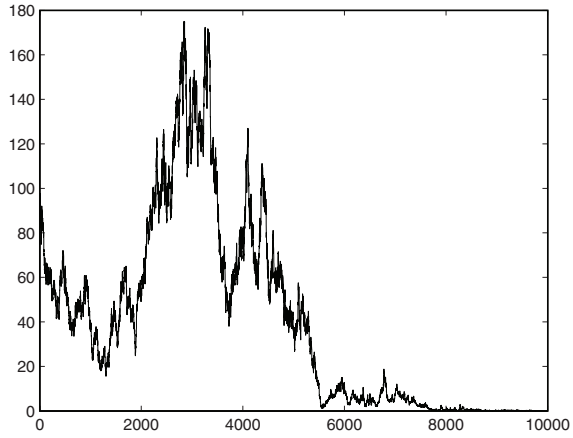

Figura 1: Valor da função objetivo de cada elemento da cadeia de Markov gerada.

Resolvemos o problema LOVO utilizando *simulated annealing* e `SA-GENCAN`. Na Figura 1 temos a cadeia de Markov gerada por *simulated annealing*, onde podemos observar que nas iterações iniciais há uma oscilação entre pontos bons e ruins, enquanto que nas iterações finais, pontos com valores piores da função objetivo não são mais aceitos.

Na Figura 2 temos a cadeia de Markov gerada por `SA-GENCAN`, onde antes de cada troca de temperatura melhoramos o valor da função objetivo através da execução de `GENCAN`. Cada execução está sinalizada na figura com o símbolo *. Notamos neste

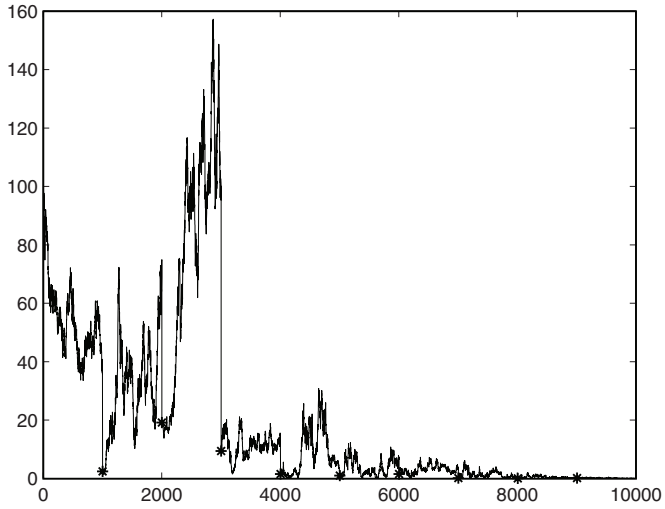

Figura 2: Valor da função objetivo da cadeia de Markov gerada por `SA-GENCAN`

caso uma oscilação pequena no valor da função objetivo após gerar 3000 pontos, o que demora mais para ocorrer no caso em que apenas *simulated annealing* é executado.

A Tabela 1 mostra o desempenho de `SA-GENCAN`, `GENCAN` e *simulated annealing* em 28 problemas de pequeno porte, com múltiplos ótimos locais e solução global conhecida, entre eles, Rosenbrock, Lennard–Jones e conformação molecular, de dimensões 100, 9 e 1 respectivamente (mais informações em [6]). Cada problema foi executado com 20 aproximações iniciais diferentes, sendo que um problema é considerado resolvido se o desvio percentual do valor da função no ponto encontrado em relação ao ótimo global é menor que 0.01.

Tabela 1: Comparação entre os algoritmos

	problemas resolvidos	avaliações de função	avaliações de gradiente
`SA-GENCAN`	98.92%	10378	1593
`GENCAN`	57.32%	183	62
simulated annealing	69.28%	9619	-

A Tabela 2 mostra a freqüência com que cada critério de parada foi acionado em `SA-GENCAN`.

Tabela 2: Critérios de parada acionados em SA-GENCAN

Critérios de Parada	Porcentagem
Variação pequena da função objetivo	35.96%
Número máximo de iterações	47.31%
Número máximo de avaliações de função	3.28%
Temperatura muito pequena	3.85%
Nenhuma melhora nas últimas iterações	9.61%

5. Conclusões

Observamos que apenas GENCAN para a obtenção do ótimo global não é eficiente, uma vez que o número de problemas resolvidos é bem reduzido, o que era de se esperar pois GENCAN é um método local e os problemas apresentados possuem diversos minimizadores locais. O que pode ser feito para melhorar sua performance é executar GENCAN mais de uma vez com aproximações iniciais escolhidas aleatoriamente no espaço de busca, neste caso os resultados são semelhantes aos obtidos por SA-GENCAN nos problemas de dimensão pequena e se torna ineficiente para problemas de dimensões maiores. Além disto, em SA-GENCAN o procedimento realizado por *simulated annealing* escolhe aproximações iniciais mais promissoras o que resulta numa vantagem para este algoritmo.

Analisando os resultados temos que SA-GENCAN é o que consegue resolver o maior número de problemas, embora o custo computacional seja substancialmente maior. O alto custo já era esperado já que em SA-GENCAN são feitas chamadas de GENCAN e também há o mesmo esforço presente em *simulated annealing*.

Em [6], utilizamos como acelerador local para o *simulated annealing*, ao invés de GENCAN, o método DBMS [8], que não faz uso de derivadas. Nos testes realizados, foi possível observar o desempenho superior de SA-GENCAN, demonstrando assim que ao utilizar um algoritmo local robusto como GENCAN ao invés de uma busca local sem derivadas, temos uma forte melhora nos resultados obtidos.

Maiores detalhes teóricos, testes númericos e uma tentativa de aplicar a mesma idéia em problemas com restrições podem ser encontrados em [6].

Agradecimentos

Agradecemos aos revisores pelos comentários e sugestões, que contribuíram para a redação final deste trabalho.

Abstract. In this work we describe the theory for *simulated annealing*, and we propose a hybrid method for continuous box–constrained global optimization such that *simulated annealing* is used on the heuristic phase and GENCAN is used on the local phase. The purpose of this strategy is to explore the local convergence properties of GENCAN and the global properties of *simulated annealing*. Numerical results obtained show the efficiency of this procedure.

Referências

[1] E.H.L. Aarts, J.H.M. Korst, "Simulated Annealing and Boltzmann Machines", John Wiley & Sons, 1989.

[2] R. Andreani, J.M. Martínez, L. Martínez, F. Yano, Low order value optimization and applications, *Optimization Online Digest*, October (2005).

[3] E.G. Birgin, J.M. Martínez, Large–scale active–set box–constrained optimization method with spectral projected gradients, *Computational Optimization and Applications*, **23** (2002), 101–125.

[4] M.F. Cardoso, R.L. Salcedo, S. Feyo de Azevedo, The simplex–simulated annealing approach to continuous non–linear optimization, *Computers and Chemical Engineering*, **20**, No. 9 (1995), 1065–1080.

[5] V. Černy, Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm, *Journal of Optimization Theory and Applications*, **45** (1985), 41–51.

[6] G. Haeser, "Algoritmo Duas Fases em Otimização Global", Dissertação de Mestrado, T/UNICAMP H119a, IMECC, UNICAMP, Campinas, SP, 2006.

[7] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, *Science*, **220** (1983), 671–680.

[8] J. Ma, P. Tian, D.-M. Zhang, Global optimization by Darwin and Boltzmann mixed strategy, *Computers & Operations Research*, **27** (2000), 143–159.

[9] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, E. Teller, Equations of state calculations by fast computing machines, *The Journal of Chemical Physics*, **21**, No. 6 (1953), 1087–1092.

[10] P. Siarry, G. Berthiau, F. Durbin, J. Haussy, Enhanced simulated annealing for globally minimizing functions of many continuous variables, *ACM Transactions on Mathematical Software*, **23**, No. 2 (1997), 209–228.

[11] M.W. Trosset, What is simulated annealing?, *Optimization and Engineering*, **2**, No. 2 (2001),201–213.