REVUE FRANÇAISE D'AUTOMATIQUE, D'INFORMATIQUE ET DE RECHERCHE OPÉRATIONNELLE. RECHERCHE OPÉRATIONNELLE

D. MAURICIO
N. MACULAN

A trust region method for zero-one nonlinear programming

Revue française d'automatique, d'informatique et de recherche opérationnelle. Recherche opérationnelle, tome 31, n° 4 (1997), p. 331-341.

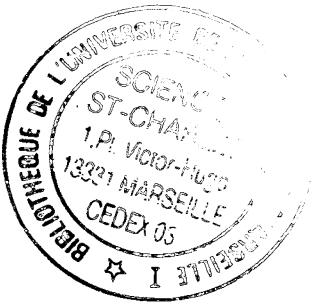
<http://www.numdam.org/item?id=RO_1997__31_4_331_0>

© AFCET, 1997, tous droits réservés.

L'accès aux archives de la revue « Revue française d'automatique, d'informatique et de recherche opérationnelle. Recherche opérationnelle » implique l'accord avec les conditions générales d'utilisation (http://www.numdam.org/legal.php). Toute utilisation commerciale ou impression systématique est constitutive d'une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM
Article numérisé dans le cadre du programme
Numérisation de documents anciens mathématiques
http://www.numdam.org/

Recherche opérationnelle/Operations Research
(vol. 31, n° 4, 1997, pp. 331 à 341)



A TRUST REGION METHOD FOR ZERO-ONE NONLINEAR PROGRAMMING (*)

by D. MAURICIO ([1]) and N. MACULAN ([2])

Communicated by Pierre TOLLA

Abstract. – *An $\mathcal{O}(n \log n)$ trust region approximation method to solve 0-1 non-linear programming is presented. Optimality conditions and numerical results are reported.*

Keywords: 0-1 nonlinear programming, trust region method, combinatorial optimization.

Résumé. – *Une méthode approchée de région de confiance de complexité $\mathcal{O}(n \log n)$ pour résoudre un programme nonlinéaire 0-1 est présentée. Des conditions d'optimalité et des résultats numériques sont aussi discutés.*

Mots clés : Programmation nonlinéaire 0-1, méthode de région de confiance, optimisation combinatoire.

1. INTRODUCTION

In this paper we have developped an approximation method to solve the following class of 0-1 nonlinear programming problems as follows:

(PNL 0-1):

$$f^* = \min f(x)$$
$$\text{s.t.} : x \in B^n = \{0, 1\}^n$$

where $f : R^n \to R$ is a nonlinear convex function. Many algorithms have been developed to solve (PNL 0-1). A review can be found in [3, 5, 6]. However, the great majority of methods are considered approximated and work well on special structures of f and a few variables.

It is proved that many exact algorithms to solve (PNL 0-1) are highly expensive and sensible in processing time to an initial chosen point

(*) Received March 1995.

([1]) State University of Norte Fluminense (UENF), Laboratory of Science and Engineering, CCT, Avenida Alberto Lamego, 200 Campos, RJ 28015-620, Brazil. e-mail: mauricio@uenf.br.

([2]) Systems Engineering and Computer Science, COPPE, Federal University of Rio de Janeiro, P.O. Box 68511, Rio de Janeiro, RJ 21945-970, Brazil, e-mail: maculan@cos.ufrj.br

Recherche opérationnelle/Operations Research, 0399-0559/97/04/$ 7.00
© AFCET-Gauthier-Villars

[3, 4, 9, 7]. Considering these difficulties, in this paper we have developed an approximated method, fast and efficient, which uses the inexact lineralization and trust region concept.

This paper is organized as follows. In the next section, practical conditions of optimality are described. In section 3 the method is showed. Numerical experiments are presented in section 4. The conclusion follows in section 5.

We denote by $\| \, . \, \|$ the euclidean norm and call the set $\partial_r f(\overline{x}) = \{s \in R^n : f(x) \geq f(\overline{x}) + s^t (x - \overline{x}),\ x \in B^n\}$, $\overline{x} \in B^n$, by discrete subdifferential of function f at point $\overline{x}$. An element $s \in \partial_r f(\overline{x})$ is called discrete subgradient of the function at the point $\overline{x}$. We also denote the set of feasible solutions of a given problem P by $D(P)$.

2. OPTIMALITY CONDITIONS

In order to verify necessary and sufficient optimality conditions to the class of nonlinear integer programming problems we present some practical conditions.

THEOREM 2.1: *Let* $\delta = \min\{f(x) - f^* : f(x) > f^*,\ x \in B^n\}$ *and* $s \in \partial_r f(\overline{x})$, $\overline{x} \in B^n$. *If* $\min\{s^t (x - \overline{x}) : x \in B^n\} > -\delta$, *then* $\overline{x}$ *is an optimal solution for* $(PNL\ 0\text{-}1)$.

Proof: From theorem hypothesis and discrete subgradient definition we therefore have

$$f(x) \geq f(\overline{x}) + s^t (x - \overline{x}) > f(\overline{x}) - \delta, \qquad \forall x \in B^n.$$

In particular, the relation above is valid for x^* optimal solution of (PNL 0-1), *i.e.*

$$f(x^*) = f^* > f(\overline{x}) - \delta,$$
$$f(\overline{x}) - f^* < \delta,$$

therefore from δ definition follows the complete proof. □

The theorem 2.1 above establishes an optimality sufficient condition. In some cases, δ can easily be determined; for example, if f is a polynomial function with integer coefficients, δ can be fixed as the unit. When the δ

computing is as hard as solving the studied problem, we can use a weaker optimality condition as follows.

THEOREM 2.2: *Let* $\overline{x} = (\overline{x}_1\, \overline{x}_2 ... \overline{x}_n)^t \in B^n$ *and* $s = (s_1\, s_2 ... s_n)^t \in \partial f_r(\overline{x})$. *If*

$$\begin{aligned} &\overline{x}_i = 1, && \text{for } i \text{ such that } s_i < 0, \\ &\overline{x}_i = 0, && \text{for } i \text{ such that } s_i > 0, \\ &\overline{x}_i \in \{0, 1\}, && \text{for } i \text{ such that } s_i = 0, \end{aligned}$$

then $\overline{x}$ *is an optimal solution of (PNL 0-1).*

Proof: We observe from the hypotheses above that $s_i \overline{x}_i \leq s_i x_i$ for all i such that $x_i \in \{0, 1\}$. Then $s_i x_i - s_i \overline{x}_i \geq 0$, for $i = 1, 2, ..., n$, we can write

$$\sum_{i=1}^{n} s_i (x_i - \overline{x}_i) = s^t (x - \overline{x}) \geq 0.$$

Thus $\min \{s^t (x - \overline{x}) : x \in B^n\} \geq 0$, and as $\delta = \min \{f(x) - f^* : f(x) > f^*, x \in B\} > 0$, we have $s^t (x - \overline{x}) > -\delta$, $\forall x \in B^n$ and from theorem 2.1, $\overline{x}$ is an optimal solution for (PNL 0-1). □

3. THE MODEL

3.1. Theoretical results

A local approach of f at point $x^k \in B^n$ is given by the following linear function:

$$f(x^k) + s^t (x - x^k), \qquad \text{for} \quad s \in \partial_r f(x^k).$$

Thus, a linear integer local model for (PNL 0-1) at $x^k \in B^n$ is given for some $\gamma \geq 0$:
$ML(x^k, \gamma)$:

$$\begin{aligned} \text{minimize} \quad & s^t x \\ \text{s.t.:} \quad & \|x - x^k\| \leq \gamma \\ & x \in B^n \end{aligned}$$

By the philosophy of trust region method, the parameter γ (trust region radius) must be updated, the model $ML(x^k, \gamma)$ tries to produce a better solution than x^k.

We can observe that $\|x - x^k\| \in \{\sqrt{0}, \sqrt{1}, \sqrt{2}, ..., \sqrt{n}\}$ for x and $x^k \in B^n$. We just pointed out that $\|x - x^k\| = 0 \Leftrightarrow x = x^k$ and $\|x - x^k\| = \sqrt{n} \Leftrightarrow x = e - x^k$, where $e = (11 \cdots 1)^t \in R^n$. All other different values of $x \in B^n$ give $\|x - x^k\| \in \{\sqrt{1}, \sqrt{2}, ..., \sqrt{n-1}\}$. Then the local model $ML(x^k, \gamma)$ can be solved by a sequency of problems: $ML(x^k, \sqrt{l})$:

$$\begin{array}{ll} \text{minimize} & s^t x \\ \text{s.t. :} & \|x - x^k\| = \sqrt{l} \\ & x \in B^n, \end{array}$$

for $l = 0, 1, 2, ..., n$.

About the feasible solutions of $ML(x^k, \sqrt{l})$, we can point the following results out:

$$\|x - x^k\| = \sqrt{l} \Leftrightarrow \sum_{i=1}^{n} (x_i - x_i^k)^2 = l \in \{0, 1, 2, ..., n\},$$

we define $N = \{1, 2, ..., n\}$, $L \subseteq N$ such that $|L| = l$.

Let

$$\bar{x}_i = \begin{cases} 1 - x_i^k & \text{if} \quad i \in L, \\ x_i^k & \text{if} \quad i \in N - L, \end{cases}$$

then $\bar{x}_i \in \{0, 1\}$, $\forall i \in N$ and

$$\begin{aligned} \sum_{i \in L} (\bar{x}_i - x_i^k)^2 + \sum_{i \in N-L} (\bar{x}_i - x_i^k)^2 &= \sum_{i \in L} (1 - x_i^k - x_i^k)^2 \\ &\quad + \sum_{i \in N-L} (x_i^k - x_i^k)^2 \\ &= \sum_{i \in L} (1 - 2\,x_i^k)^2 \\ &= \sum_{i \in L} [1 - 4\,x_i^k + 4\,(x_i^k)^2] \\ &= \sum_{i \in L} 1 \\ &= l. \end{aligned}$$

(We note that for $\lambda \in \{0, 1\}$, $\lambda^2 = \lambda$.)

Thus $\overline{x} \in D[ML(x^k, \sqrt{l})]$. For each $L \subseteq N$, such that $|L| = l$ we can define a feasible solution $\overline{x}$ of $ML(x^k, \sqrt{l})$, and we have a way to generate all feasible solutions of $ML(x^k, \sqrt{l})$.

The following result shows that local model $ML(x^k, \sqrt{l})$ for $l \in \{1, 2, ..., n\}$ can be solved by a sequency of simple problems.

THEOREM 3.1: *Suppose* $s_i(1-2x_i^k) \leq s_{i+1}(1-2x_{i+1}^k)$, $i = 1, 2, ..., n-1$ *and* $L^* = \{1, 2, ..., l\}$, $x_i^* = 1 - x_i^k$, $i \in L^*$ *and* $x_i^* = x_i^k$, $i \in N - L^*$, *then* x^* *is an optimal solution for* $ML(x^k, \sqrt{l})$.

Proof (by contradiction): Let $\overline{x}_i = 1 - x_i^k$, $i \in \overline{L} \subseteq N$, $|\overline{L}| = l$, $\overline{L} \neq L^*$, and $\overline{x}_i = x_i^k$, $i \in N - \overline{L}$ a feasible solution of $ML(x^k, \sqrt{l})$ such that $s^t \overline{x} < s^t x^*$, that is

$$\sum_{i=1}^{n} s_i \overline{x}_i < \sum_{i=1}^{n} s_i x_i^*,$$

we can write

$$\sum_{i=1}^{n} s_i (\overline{x}_i - x_i^k) < \sum_{i=1}^{n} s_i (x_i^* - x_i^k)$$

or

$$\sum_{i \in \overline{L}} s_i (1 - x_i^k - x_i^k) + \sum_{i \in N - \overline{L}} s_i (x_i^k - x_i^k) < \sum_{i \in L^*} s_i (1 - x_i^k - x_i^k) + \sum_{i \in N - L^*} s_i (x_i^k - x_i^k)$$

then

$$\sum_{i \in \overline{L}} s_i (1 - 2x_i^k) < \sum_{i \in L^*} s_i (1 - 2x_i^k)$$

which contradicts $s_i(1-2x_i^k) \leq s_{i+1}(1-2x_{i+1}^k)$, $i = 1, 2, ..., n-1$. □

Using theorem 3.1 we can find an algorithm of $\mathcal{O}(n \log n)$ [1, 10] complexity to solve the sequency of problems $ML(x^k, \sqrt{l})$, $l = 1, 2, ..., n$.

3.2. Algorithm

Now we present a trust region algorithm to solve 0-1 nonlinear programming problems. This method is based on the fact that the cost

to solve the local model for a fixed radius is the same for its many values. That is, given an initial point, next point is determined at the best descent solution given by local model for all trust regions ($\gamma = \sqrt{1}, \sqrt{2}, ..., \sqrt{n}$). Proceeding continues until some stop is verified, we consider optimality tests shown in section 2, and heuristic stop test that is satisfied when a descent solution cannot be found.

Approximate algorithm

(0) Start

given $x^0 \in B^n$, $\delta \geq 0$, $s^0 \in \partial f_r(x^0)$, set $k := 0$

(1) Optimality test

if x^k verifies theorem 2.1 or 2.2, then x^k is an optimal solution, STOP

(2) Search a better descent solution

take $s^k \in \partial f_r(x^k)$

set $y := x^k$

for $l := 1$ to n

compute $z \in \arg\min ML(x^k, \sqrt{l})$ (theorem 3.1)

if $f(z) < f(y)$ then $y := z$

end-for

(3) Heuristic test

if $f(y) = f(x^k)$ then x^k is the best heuristic solution, STOP

(4) Next point

$x^{k+1} := y$, $k := k + 1$ goto (1).

4. NUMERICAL EXPERIMENTS

The approximate algorithm was implemented in FORTRAN F77L3. We have used a shell sort modification to determine the sequency order, as pointed out in theorem 3.1, and a subgradient was computed as a discrete one. All numerical experiments were realized in a PC-DX 486 computer with 8 MBytes RAM and 50 MHz.

4.1. Test problems

We indicated for F.O. and S.O. the objective function and the optimal solution respectively, and n is even.

• Prob. 1:

F.O.:

$$\sum_{i=1}^{n} x_i^2 - 1.8 \sum_{i=1}^{n} x_i + 0.81\, n$$

S.O.:

$$x_i = 1, \;\; i = 1, 2, ..., n$$

• Prob. 2:

F.O.:

$$2 \sum_{i=1}^{n} x_i^2 + \sum_{i=1}^{n-1} x_i\, x_{i+1} - 2 \sum_{i=1}^{n/2} (1.9\, x_{2\,i-1} + 1.1\, x_{2\,i}) + 1.205\, n$$

S.O.:

$$x_{2\,i-1} = 1, \; x_{2\,i} = 0, \; i = 1, ..., n/2$$

• Prob. 3:

F.O.:

$$\sum_{i=1}^{n} (x_i - 0.9)^{8/3}$$

S.O.:

$$x_i = 1, \;\; i = 1, 2, ..., n$$

• Prob. 4:

F.O.:

$$\sum_{i=1}^{n} x_i^2 - 0.8 \sum_{i=1}^{n} x_i + 0.16\, n$$

S.O.:

$$x_i = 0, \;\; i = 1, 2, ..., n$$

- Prob. 5:

F.O.:

$$0.4 \sum_{i=1}^{n/2} (x_i - 0.6)^4 + 0.6 \sum_{i=n/2+1}^{n} (x_i + 0.4)^2$$

S.O.:

$$x_i = 1, \ i = 1, ..., n/2;$$
$$x_i = 0, \ i = n/2 + 1, ..., n$$

- Prob. 6:

F.O.:

$$-\prod_{i=1}^{n/2} (x_{2i-1} - 1)(x_{2i-1} + 1)$$

S.O.:

if n is a multiple of 4
then $x_{2i-1} = 0, \ x_{2i} = 1, \ i = 1, ..., n;$
if n is an even but not multiple of 4
then $x_{2i-1} = 1$ for some $i \leq n/2$,
$x_j = 1 \quad \text{or } 0 \qquad \text{for} \quad j \in \{1, ..., n\} - \{2i - 1\}$

- Prob. 7:

F.O.:

$$\sum_{i=1}^{n/2} (2x_i - 1)^2 + \sum_{i=n/2+1}^{n} (3x_i^2 - 1)^2$$

S.O.:

$$x_i = 1 \quad \text{or } 0, \qquad \forall \ i \leq n/2;$$
$$x_i = 0, \qquad \forall i > n/2$$

- Prob. 8:

F.O.:

$$\sum_{i=1}^{n/2} (2x_i - 1)^2 + \sum_{i=n/2+1}^{n} (3x_i^2 - 1)^2 + \prod_{i=1}^{n} x_i$$

S.O.:

$$x_i = 1 \quad \text{or } 0, \qquad \forall\ i \leq n/2;$$
$$x_i = 0, \qquad \forall i > n/2$$

- Prob. 9:

F.O.:

$$\sum_{i=1}^{n/2} (0.1)^{x_i} + \sum_{i=n/2+1}^{n} (1.1)^{x_i}$$

S.O.:

$$x_i = 1, \qquad \forall\ i \leq n/2;$$
$$x_i = 0, \qquad \forall i > n/2$$

- Prob. 10:

F.O.:

$$(0.1)^{\sum_{i=1}^{n} x_i} + (1.1)^{\sum_{i=1}^{n} x_i}$$

S.O.:

$$x_i = 1 \quad \text{for some } j \leq n,$$
$$x_i = 0 \quad \text{for } i \in \{1, ..., n\} - \{j\}$$

4.2. Numerical results

Problems 1, 2, 3 and 4 were taken from [9]. In table 1 we can observe our numerical results for a sample of 30 problems, for each objective function we considered $n \in \{128, 512, 1024\}$. For 12 examples the heuristic solutions obtained did not verify the optimality conditions presented in theorems 2.1 and 2.2. For problems 1, 2, 4, and 8 we take $\delta = 0.2 = \frac{1}{5}$, because the multiplication of these objective functions by 5 will give polynomial functions with integer coefficients.

5. CONCLUSION

We presented a heurist algorithm based on inexact linearization concept and trust region to solve nonlinear 0-1 programming problems. Table 1 shows that our method works well for the sample of problems presented

in the last section. The algorithm converged to an optimal solution in few seconds. But for some problems in our sample the optimality conditions are not verified, as we can observe in table 1. In this last case optimality has to be tested using enumerative methods which can be found in [8, 9, 2]. When

TABLE I
Numerical results for heuristic trust region.

Prob	n	dd	sol	test	iter	CPU	f^*
1	128	0.2	opt	V	2	0.05	1.28
	512	0.2	opt	V	2	0.93	5.12
	1024	0.2	opt	V	2	3.85	10.24
2	128	0.2	opt	N	2	0.11	39.04
	512	0.2	opt	N	2	2.20	156.16
	1024	0.2	opt	N	2	8.84	312.32
3	128	0.0	opt	N	2	0.11	0.2275767
	512	0.0	opt	N	2	1.82	1.1030705
	1024	0.0	opt	N	2	7.19	2.20611
4	128	0.2	opt	V	1	0.0	20.48
	512	0.2	opt	V	1	0.0	81.92
	1024	0.2	opt	V	1	0.0	163.84
5	128	0.0	opt	V	2	0.06	6.79936
	512	0.0	opt	V	2	0.60	27.1974
	1024	0.0	opt	V	2	2.53	54.39488
6	128	0.0	opt	V	2	0.05	-2^{64}
	512	0.0	opt	V	2	0.10	-2^{116}
	1024	0.0	opt	V	2	0.11	0.0
7	128	1.0	opt	N	1	0.0	128.
	512	1.0	opt	N	1	0.49	512.
	1024	1.0	opt	N	1	2.14	1024.
8	128	0.2	opt	V	1	0.0	128.
	512	0.2	opt	V	1	0.0	512.
	1024	0.2	opt	V	1	0.0	1024.
9	128	0.0	opt	V	2	0.06	70.3999
	512	0.0	opt	V	2	0.77	281.6001
	1024	0.0	opt	V	2	3.13	563.14983
10	128	0.0	opt	N	2	0.11	1.2
	512	0.0	opt	N	2	1.1	1.2
	1024	0.0	opt	N	2	2.53	1.2

Prob: # test problem; n: number of variables; dd: indicate parameter value δ; sol: type of solution; opt: optimal solution; test: optimality test; N: the optimality test is not verified; V: the optimality test is verified; iter: number of iterations; CPU: processing time in seconds; f^*: objective function value.

f is not convex, it is always possible to convert it to (PNL 0-1) form, adding a penalty term, *see*, for example [9].

ACKNOWLEDGEMENT

The authors wish to thank Frederico da Cruz for his helpful advice and suggestions. This work was partially supported by CNPq and FAPERJ.

REFERENCES

1. A. V. Aho, J. E. Hopocroft and J. D. Ullman, *The Design and Analysis of Computer Algorithms.* Addison Wesley Publication Company, 1974.
2. J. Cha, Mixed Discrete Constrained Nonlinear Programming via Recursive Quadratic Programming. University of New York at Buffalo, 1987.
3. R. S. Garfinkel and G. L. Nenhauser, *Integer Programming.* John Willey & Sons, Inc., NY, 1972.
4. O. K. Gupta and A. Ravindran, Nonlinear Integer Programming and Discrete Optimization, *ASME Journal of Mechanism, Transmission and Automation in Design,* 1981, *105*, pp. 160-164.
5. P. Hansen, Methods of Nonlinear 0-1 Programming and Discrete Mathematical Programming. *Mathematical Programming,* 1979, *5*, pp. 53-70.
6. H. Loh and N. Papalambros, A sequential Linearization Approach for Solving Mixed-Discrete Nonlinear Design Optimization Problems. Design Laboratory, University of Michigan, Technical Report UM-MEAM-89-08, 1989.
7. D. Mauricio and S. Scheimberg Um Método de Linearização Sequencial com Cortes Eficientes para Programação Inteira com Restrições Lineares. COPPE/Federal University of Rio de Janeiro, ES-283.93, Relatório Técnico, 1993.
8. D. Mauricio, Métodos de Resolução de Problemas de Programação Não Linear Inteira. Ph. D. dissertation, Systems Engineering and Computer Science, COPPE/Federal University of Rio de Janeiro, 1994.
9. Z. A. Wu, A Subgradient Algorithm for Nonlinear Integer Programming and its Implementation. Ph. D. dissertation, Rice University, Houston, Texas, 1991.
10. N. Ziviani, *Projeto de Algoritmos – Com Implementações em Pascal e C.* Pioneira, São Paulo, SP, Brazil, 1993.