Fecha de recepción: 12/11/2021 Fecha de aceptación: 17/04/2022 Pp. 116 – Pp. [141](#_bookmark59)

An Extension of the Nested Partitions Method

*Ebert Brea*

## [ebrea@ucab.edu.ve](mailto:ebrea@ucab.edu.ve) Centro de Investigación y Desarrollo de Ingeniería & Escuela de Ingeniería Industrial,

Universidad Católica Andrés Bello, Caracas, Venezuela

**Abstract**

This article addresses a new extension of the well known Nested Partitions (NP) method for globally solving mixed integer nonlinear optimization problems under bound constraints. The extension, called Mixed Integer Nested Partitions (MINP) method, is based on the same stages of the NP method at each iteration, i.e.: partitioning; random sampling; identifying of the promising region, which presumes to contain at least a global solution of the problem; and verifying of the stopping rule. Nevertheless, both a new scheme of partitioning and a stopping rule proposal are here presented as main contributions to mixed integer programming. The article has also included a theoretical study of the behavior of the MINP method from the point of view of the Markov chain. Numerical examples have made sure the correct functionality of the algorithmic method and its new stopping rule.

**Key words**: Nested Partitions method, mixed integer nonlinear programming, global optimization.

Una Extensión del Método de Particiones Anidadas

**Resumen**

Este artículo aborda una nueva extensión del conocido método de Particiones Anidadas (PA) para resolver globalmente problemas de optimización no lineales enteros mixtos bajo restricciones de bandas. La extensión, llamada método de Particiones Anidadas Enteros Mixtos (PAEM), se basa en las mismas etapas del método PA en cada iteración, es decir: partición; muestreo aleatorio; identificación de la región prometedora, que presume contener al menos una solución global del problema; y verificación de la regla de parada. Sin embargo, tanto un nuevo esquema de partición como una propuesta de regla de parada que se presentan aquí son las principales contribuciones a la programación entera mixta. El artículo también ha incluido un estudio teórico del comportamiento del método PAEM desde el punto de vista de la cadena de Markov. Ejemplos numéricos han asegurado la correcta funcionalidad del método algorítmico y su nueva regla de parada.

**Palabras clave**: método de Particiones Anidadas, programación no lineal entera mixta, optimización global.

Uma Extensão do Método de Partições Aninhadas

**Resumo**

Este artigo aborda uma nova extensão do conhecido método de Partições Aninhadas (PA) para resolver globalmente problemas de otimização não linear de inteiros mistos com restrições. A extensão, chamada método de Partições Aninhadas Inteiras Mistas (PAIM), é baseada nos mesmos estágios do método PA em cada iteração: particionamento; amostragem aleatória; identificação da região promissora, que presume conter pelo menos uma solução global do problema; e verificação do critério de parada. Não obstante, tanto um novo esquema de particionamento como uma proposta de critério de parada são aqui apresentados como principais contribuições à programação inteira mista. O artigo também incluiu um estudo teórico do comportamento do método PAIM do ponto de vista da cadeia de Markov. Exemplos numéricos garantem o correto funcionamento do método algorítmico e do novo critério de parada.

**Palavras chave**: Método de Partições Aninhadas, programação não linear inteira mista, otimização global.

Revista TEKHNÉ No 25.1 Semestre octubre-febrero 2022

ISSN electrónico: 2790-5195

ISSN: 1316-3930

## INTRODUCTION

Consider the following bound constrained mixed integer nonlinear minimization problem:

**Problem 1**

## minimize *f* (*z*); (1a)

*𝑥*∈R*n*×Z*m*

*subject to*

*l* ≤ *z* ≤ *u,* (1b)

where: *z* will henceforward denote any mixed integer vector, i.e., *z* = (*z*(1)*, . . . , z*(*n*); *z*(*n*+1)*, . . . , z*(*n*+*m*))*t*, in the (n+m)-multidimensional Euclidean space R*n* × Z*m*; *f* (*z*) : R*n* × Z*m* → R is a nonlinear function, for which

analytical and explicit mathematical expression cannot be obtained; and *l, u* ∈ R*n* × Z*m* respectively are both the lower and upper bounds of the mixed integer feasible region Θ.

Note that we have here denoted by the symbol ≤ for

indicating that a vector precedes to another vector, what

will be defined later.

In this case, the objective function must be evaluated by an appropriate simulation model or by solving a nonlin- ear system equations, and besides the objective function has no gradient function, because the objective function

domain is defined in the set mixed integer R*n* × Z*m*.

Nowadays this kind of problems has had an important

presence in the industry, due to fact the enormous challenges facing the industry, which must find answers for efficiently designing equipment and systems, and therefore friendly with the environment, and at the same time taking into account the economic sustainability.

Examples of these problems in the branch of chemical engineering are presented by Floudas **[**[**1**](#_bookmark60)**]**, who intro- duces an important number of mixed integer nonlin- ear optimization problems on: design, scheduling and planning of batch processes; heat exchanger network synthesis, etc. Grossmann and Kravanja also present an overview of the applications in many areas within the engineering process **[**[**2**](#_bookmark61)**,** [**3**](#_bookmark62)**]**. Tawarmalani and Sahinidis also address the mixed integer nonlinear programming through a rigorous study presented in **[**[**4**](#_bookmark63)**,** [**5**](#_bookmark64)**]**. For his part, Brea has found optimum solutions for the design of equipment making use of the mixed integer optimization viewpoint **[**[**6**](#_bookmark65)**,** [**7**](#_bookmark66)**]**, and he also has developed a new meta- heuristic based on a game framework of random pattern search algorithms, offering a new approach for globally solving mixed integer nonlinear problems **[**[**8**](#_bookmark67)**,** [**9**](#_bookmark68)**,** [**10**](#_bookmark69)**]**. Another example worthwhile of commenting, it is the recently published article by Kantor and coworkers, who

#### EBERT BREA

propose a model for finding an integrated solution, which allows the industry to provide optimum integral solutions within plants and potential industrial symbiosis options, using a mixed integer linear programming approach **[**[**11**](#_bookmark70)**]**. There exists a vast amount of algorithmic optimization methods via a wide variety metaheuristic approaches for globally solving constrained and unconstrained mixed integer nonlinear optimization problems, which could be grouped in two large classes, namely: the bio-inspired metaheuristic algorithms and the non bio-inspired meta- heuristic algorithms. Among the bio-inspired meta- heuristic algorithms, it can be included: Genetic Al- gorithms **[**[**12**](#_bookmark71)**,** [**13**](#_bookmark72)**]**; Ant Colony **[**[**14**](#_bookmark73)**]**; Particle Swarm optimization, and Artificial Bee Colony **[**[**15**](#_bookmark74)**]**, whereas in the group of non bio-inspired metaheuristic algorithms, we have: Simulated Annealing **[**[**16**](#_bookmark75)**]**; Branch-and-Bound method **[**[**17**](#_bookmark76)**]**, Nested Partitions method **[**[**18**](#_bookmark77)**]**; Game of Patterns **[**[**9**](#_bookmark68)**]**, etc.

The best of our knowledge, there exists only one pub-

lished article concerning an extension the Nested Parti- tions method for globally solving mixed integer optimiza- tion problems **[**[**19**](#_bookmark78)**]**, which is based on a hybrid of the Nested Partitions method **[**[**18**](#_bookmark77)**,** [**20**](#_bookmark79)**]** and the well known CPLEX **[**[**21**](#_bookmark80)**]**. Nevertheless, we here propose a new ex- tension of the Nested Partitions method for finding global solutions to bound constrained mixed integer nonlinear optimization problems based on the main idea of the Nested Partitions, namely: partition of the promising region; sampling of each subregion from partitioned promising region and the surrounding region to the promising region; and identification of the next promising region, which presumes to have a global solution of our problem. Nonetheless, the algorithmic extension, which has been called Mixed Integer Nested Partitions (MINP) method includes: both a novel partitions scheme and a new algorithm stopping rule represent the main contributions of this article.

Numerical examples have allowed us to assert that the MINP method has successfully identified promising regions, in a relative small amount of iterations. Nev- ertheless, the MINP method has resulted to be a very expensive method, from the viewpoint of times the ob- jective function requires to be evaluated by the algorithm for globally solving this kind of problems.

Due to the fact that the MINP method has resulted to be an expensive algorithm from the viewpoint of the number of function evaluations for identifying at least a global solution, we have therefore considered to undertake fur- thermore research in the future, for hybridizing the MINP method with some local search method, e.g., the mixed integer randomized pattern search algorithm developed

by Brea [[**10**](#_bookmark69)], and is thought that by this via, it could be improved its performance.

The rest of the article is organized as follows. Next section introduces basic concepts and main idea of the NP method **[**[**18**](#_bookmark77)**,** [**20**](#_bookmark79)**]** through a very simple didactic example in R2. Section iii presents the theoretical main results of the MINP method, which are based on the NP method approach. A pseudocode of the MINP method is proposed in Section iv, whereby its procedures and functions have been included in appendices. Section v addresses the main properties of the MINP method,

which has been studied from a Markov chain viewpoint.

*x*(2)

9

8

7

6

5

4

3

2

1

#### EBERT BREA

Section vi presents a software, which operates the MINP method for taking *r* independent samplings of perfor-

(0)

0 1 2 3 4 5 6 7 8 9

*x*(1)

mance measurements of the MINP method. Section vii shows a set of numerical experiments for statistically analyzing the performance of the MINP method. Finally, Section viii discusses advantages and disadvantages of the MINP method, and future research for improving performance of the MINP method. A list of mixed integer nonlinear problems for testing the MINP method has been included in Appendix [A](#_bookmark55).

# THE NESTED PARTITIONS METHOD

With the aim of giving a didactic explanation of the NP method principles, consider the following two dimen- sional real optimization example.

**Example 1**

minimize*f* (*x*) (2a)

*x*∈R2

Figure 1. Entire feasible region, promising region *σ*(0)

As can be seen from Figure [2](#_bookmark3), the promising region *σ*(0) is partitioned or divided into 4 subregions, yielding the set of subregions {*σj*(0)}4 , and at this iteration *k* = 0, there exists no surrounding region to *σ*(0), i.e., *S*(*σ*(0)) = ∅.

*j*=1

*x*(2)

9

|  |  |
| --- | --- |
| 2(0) | 1(0) |
| 4(0) | 3(0) |

8

7

6

5

4

3

2

*subject to*

(1*,* 1)*t* ≤ *x* ≤ (9*,* 9)*t,* (2b)

1

0 1 2 3 4 5 6 7 8 9

*x*(1)

*where f* (*x*) : R2 → R *is a nonlinear objective function.*

The feasible region given by ([2b](#_bookmark4)) is indubitably depicted by a square region with both width and height equal to 8 units, such as is shown in Figure [1](#_bookmark2) in green color.

Note that this region is considered the initial promising region *σ*(0), because the entire feasible region at least contains, and without any discussion, a global solution of the problem.

Figure 2. Partitions of promising region *σ*(0)

According to the NP method **[**[**18**](#_bookmark77)**]**, the algorithm takes random trial points from each *j*th subregion, denoted by *σj*(0), for estimating the next promising region by finding an index ˆ*j*, which indicates where the best function value has come from, as it can then be seen from Figure [3](#_bookmark5) by a red cross on *σ*3(0), i.e., ˆ*j* = 3, and hence, the next promising region will be given by the subregion *σ*3(0).

*x*(2)

9

|  |  |
| --- | --- |
| 2(0) | 1(0) |
| 4(0) | 3(0) |

8

7

6

5

4

3

2

1

0 1 2 3 4 5 6 7 8 9

*x*(1)

#### EBERT BREA

and *σ*5(1) = *S*(*σ*(1)), what allows the NP method to identify the best subregion, this means, if the best subregion results ˆ*i* ∈ {1*,* 2*,* 3*,* 4} the algorithm go toward to next iteration, where the promising region is therefore smaller that the current promising region, whilst if ˆ*i* = 5, then the NP method backtracks to the initial promising region *σ*(0).

*x*(2)

9

|  |  |  |
| --- | --- | --- |
| *S*((1)) | | |
|  | 2(1) | 1(1) |
| 4(1) | 3(1) |

8

7

6

Figure 3. Sampling of each *j*th subregion {*σj* (0)}4

*j*=1

We must say that if at this stage, the NP method identi- fies more than one subregion likewise promising, the NP method will arbitrarily break this draw, for choosing just one subregion, of course.

In the example shown from Figure [4](#_bookmark7), we have hence assumed that the promising region resulted to be *σ*3(0),

5

4

3

2

1

0 1 2 3 4 5 6 7 8 9

*x*(1)

which will therefore be the next promising region *σ*(1),

and so starts a next iteration of the NP method.

*x*(2)

9

*S*((1))

(1)

8

7

6

5

4

3

2

1

Figure 5. Partitions of promising region *σ*(1)

We have assumed that the sampling procedure yielded that the best function value belongs to the subregion *σ*2(1), which has thus been marked by a red cross on

subregion *σ*2(1), and therefore ˆ*j* = 2.

We must point out that two backtracking rules have been proposed by Shi and Ólafsson, namely: the first one causes a backtracking process to the previa promising region; and the second one effectuates a backtracking process to the entire feasible region **[**[**18**](#_bookmark77)**]**.

*x*(2)

9

|  |  |  |
| --- | --- | --- |
| *S*((1)) | | |
|  | 2(1) | 1(1) |
| 4(1) | 3(1) |

8

0 1 2 3 4 5 6 7 8 9

Figure 4. promising region *σ*(1)

*x*(1) 7

6

5

The partitions of the promising region *σ*(1), into a new 4

set of four subregions {*σj*(1)}4

*j*=1

, and that had been

3

denoted by *σ*3(0) at the 0th iteration, which is depicted in

Figure [5](#_bookmark6) in green color, and also the surrounding region to *σ*(1), that is here denoted by *S*(*σ*(1)), which is shown by the yellow area of the figure.

Figure [6](#_bookmark8) displays by points the set of random trial points

2

1

0 1 2 3 4 5 6 7 8 9

*x*(1)

*j*=1

that have been taken from each *j*th subregion *σj*(1), and from the current surrounding region *S*(*σ*(1)). Here, we

Figure 6. Sampling of each *j*th subregion {*σj* (1)}4

can say that we have 5 subregions, namely: {*σj*(1)}4 ;

*j*=1

As is shown in Figure [7](#_bookmark9), we have assumed that the best

objective function value comes from *σ*2(1), what allows us to identify the subregion *σ*2(1) as the next promising region, namely, *σ*(2) = *σ*2(1).

#### EBERT BREA

**Definition 1 (Precedent vector)** *Let z*1 *and z*2 *be two mixed integer vectors, i.e., z*1*, z*2 ∈ R*n* × Z*m. It is said that z*1 *precedes z*2 *in order, and will be denoted by*

*z*1 ≤ *z*2*, if their respective components z*(*i*) ≤ *z*(*i*) *for*

*x*(2)

9

8

7

6

5

1 2

*each i* ∈ {*ℓ*}*n*+*m.*

*ℓ*=1

**Definition 2 (Subsequent vector)** *Let z*1 *and z*2 *be two mixed integer vectors, i.e., z*1*, z*2 ∈ R*n* × Z*m. It is said that z*1 *succeeds z*2 *in order, and will be denoted by*

*z*1 ≥ *z*2*, if their respective components z*(*i*) ≥ *z*(*i*) *for*

*S*((2))

2(1) = (2)

1 2

4 *each i* ∈ {*ℓ*}*n*+*m.*

*ℓ*=1

3

2

1

0 1 2 3 4 5 6 7 8 9

Figure 7. promising region *σ*(2)

*x*(1)

The relationship between the vectors that have been defined above are respectively called strictly precedes

or strictly succeeds, if the above inequalities are true as strict inequalities for all *i*th components of *z*1 and *z*2.

**Definition 3 (Promising region)** *Let σ*(*k*) *be a*

This procedure is recurrently carried out until some stopping criterion is met.

Shi and Ólafsson have proposed two stopping rules for the NP method, namely: the first one is based on ordinal optimization concepts; and the second one has been developed applying concepts of order statistics for objective function values **[**[**22**](#_bookmark81)**]**.

# THE MIXED INTEGER NESTED PARTITIONS METHOD

The MINP method has been designed for globally solv- ing mixed integer bound constrained nonlinear problems. For that, we have developed the MINP method based on similarity stages to the NP method, namely: i) parti-

tioning of the promising region *σ*(*k*) into *Mσ* subregion *σi*(*k*) for all *i* ∈ {*ℓ*}*Mσ* ; ii) sampling of the subregion *σi*(*k*) by random mixed integer trial points from each

*ℓ*=1

subregion; iii) identifying of the next promising region *σ*(*k* + 1) = *σ*ˆ*i*(*k*), where ˆ*i* denotes the index where the best value of the objective function has come from; iv)

verifying whether the MINP method will go next iteration or it will stop, and as a result of this verifying stage, i.e., the algorithm will accordingly test at each iteration, if the stopping rule is met.

## PRELIMINARIES

This subsection deals with some definitions for famil- iarizing the reader with concepts and notations that have been included both in the explanation of the MINP method and its pseudocode.

*nonempty promising region at the current iteration k*

*of the MINP method. It is then said that σ*(*k*) *is given by*

*σ*(*k*) = {*z* ∈ R*n* ×Z*m* : *lσ*(*k*) ≤ *z* ≤ *uσ*(*k*)}*,* ∀*k* ∈ N*,* (3)

*where: k hereafter depicts the counter of iteration that has been carried out by the MINP method from the initial promising region σ*(0) = Θ = {*z* ∈ R*n* ×Z*m* : *l* ≤ *z* ≤ *u*}*;*

*lσ*(*k*)*, uσ*(*k*) ∈ R*n* × Z*m respectively are the lower and*

*upper bound vector of the kth promising region σ*(*k*)*,*

*which presumes to contain at least a global solution of Problem* [*1*](#_bookmark0)*.*

Note that according to Definition [3](#_bookmark10), each *k*th promising region must be a nonempty, because it presumes to contain at least a global solution of Problem [1](#_bookmark0). Moreover, the smallest promising region, in the number of element sense, that is contained on the feasible region of Prob- lem [1](#_bookmark0), is a singleton set.

**Definition 4 (Subregion)** *Let σi*(*k*) *denote the ith sub- region of the current kth promising region. It is then said that σi*(*k*) ⊂ *σ*(*k*) *for each i* ∈ {*ℓ*}*Mσ , and σi*(*k*) ∩ *σj*(*k*) = ∅ *for all i* =/ *j and i, j* ∈ {*ℓ*}*Mσ , where Mσ denotes the number of subregion.*

*ℓ*=1

*ℓ*=1

**Definition 5 (Surrounding region)** *Let S*(*σ*(*k*)) *be the surrounding region to the kth promising region σ*(*k*)*. It is then said that S*(*σ*(*k*)) = *σ*(0)\ *σ*(*k*)*.*

**Definition 6 (Depth vector)** *Let d*(*k*) *denote the kth mixed integer depth vector of the current kth promising region. It is then said that d*(*k*) = *uσ* − *lσ, i.e.,*

#### EBERT BREA

*promising mixed integer region σ*¯(*k*) ∈ Z*m. Then, an appropriate partition of the jth component of σ*¯(*j*)(*k*) *into two disjoint subsets is given by*

(1)

(1)

(*n*+*m*)

(*n*+*m*) *t*



(*j*)

(*j*)

(*j*)

(*j*)

(*ℓ*)

(*ℓ*)

*d*(*k*) =

*uσ*(*k*) − *lσ*(*k*)*,..., uσ*(*k*) − *lσ*(*k*)

*,* ∀*k* ∈ N*.*

## (4)

 [¯*lσ , δ*

 (*j*)

] ∪ [*δ*

+ 1*, u*¯*σ* ]*, if* ⌊*δ*

(*j*)

⌋ = ⌈*δ* ⌉*;*

(6)

Note that according to above definitions: i) *σ*(0) is the feasible region Θ of Problem [1](#_bookmark0); ii) the *k*th promising

[¯*lσ ,* ⌊*δ*(*ℓ*)⌋] ∪ [⌈*δ*(*ℓ*)⌉*, u*¯*σ* ]*, if* ⌊*δ*(*ℓ*)⌋ /= ⌈*δ*(*ℓ*)⌉*,*

*where δ*(*j*) = (¯*l*(*j*) + *u*¯(*j*))*/*2 *is the midpoint of the rect*

*σ σ σ*

region *σ*(*k*) is a convex hull, which is bounded by

*line segment defined by* ¯*l*(*j*) *and u*¯(*j*)*.*

*lσ*(*k*) ≤ *z* ≤ *uσ*(*k*), and it will hereafter be simplified its *σ σ*

notation by *l*

≤ *z* ≤ *u*

; and iii) *σ*(*k*) = S*Mσ σ* (*k*).

**Proof.** Because we have two cases, the first one,

*σ σ*

Finally, let N

*i*=1 *i*

(*j*)

¯(*j*)

(*j*)

*κ* ∈ be the cumulative counter iterations

that are executed by the MINP method, which always is greater than or equal to *k*.

## ON THE PARTITIONS OF THE PROMISING

when *δ* = (*lσ* + *u*¯*σ* )*/*2 is an integer number, and the second one, when *δ*(*j*) is a no integer number, these cases will separately be proved.

Part i) if *δ*(*j*) is an integer number, i.e., ⌊*δ*(*ℓ*)⌋ = ⌈*δ*(*ℓ*)⌉

the set given by *σ*¯(*j*)(*k*) = {*y* ∈ Z : ¯*l*(*j*) ≤ *y* ≤ *u*¯(*j*)}

REGION *σ σ*

can be partitioned into two subsets, namely, *σ*¯(*j*)(*k*) =

We now turn our attention to the proposed procedure of

1. (

1

(*j*)

partitioning of the *k*th promising region *σ*(*k*), which is based on a novel method for getting the boundaries of each *i*th subregion *σi*(*k*) in the current promising mixed

{*y* ∈ Z : ¯*lσ* ≤ *y* ≤ *δ j*)} and *σ*¯2 (*k*) = {*y* ∈ Z : *δ*(*j*) +

1. ≤ *y* ≤ *u*¯(*j*)}, where both mentioned subsets indeed

*σ*

are disjoint, i.e., *σ*¯(*j*)(*k*) ∩ *σ*¯(*j*)(*k*) = ∅, and *σ*¯(*j*)(*k*) =

integer region, as is shown in Figure [18](#_bookmark56) on page [135](#_bookmark56).

(*j*)

1 2

(*j*)

**Proposition 1** *Let σ*(*j*)(*k*) = {*x* ∈ R : *l*(*j*) ≤ *x* ≤ *u*(*j*)}

*σ*¯1 (*k*) ∪ *σ*¯2 (*k*).

Part ii) In the case when *δ*(*j*) is a no integer number,

*σ σ*

then the subsets

(*j*)

Z ¯(*j*)

(*j*)

*be the jth interval of real number or set of the jth*

*real value component that defines the current promising*

*σ*¯1 (*k*) = {*y* ∈ : *lσ* ≤ *y* ≤ ⌊*δ* ⌋}

and *σ*¯(*j*)(*k*) = {*y* ∈ Z : ⌈*δ*(*j*)⌉ ≤ *y* ≤ *u*¯(*j*)} are such that

*mixed integer region σ*(*k*) ∈ Z*m. Then, an adequate*

2 *σ*

( ) ( ) ( )

*σ*¯ *j* (*k*) ∩ *σ*¯ *j* (*k*) = ∅, and *σ*¯(*j*)(*k*) = *σ*¯(*j*)(*k*) ∪ *σ*¯ *j* (*k*).

*partition of the jth component of σ*(*j*)(*k*) *into two disjoint* 1 2 1 2

*subsets is given by*

*σ*(*j*)(*k*) = {*x* ∈ R : *l*(*j*) ≤ *x* ≤ *δ*(*j*)}; (5a)

Based on Propositions [1](#_bookmark12) and [2](#_bookmark15), it was designed Parti- tioning procedure shown in Figure [18](#_bookmark56).

**Remark 1** *Assume, without any loss of generality, that*

1 *σ σ*

*σ*(*j*)(*k*) = {*y* ∈ R : *δ*(*j*) *< y* ≤ *u*(*j*)}*,* (5b)

*Partitions procedure in Figure* [*18*](#_bookmark56) *is conducted on any*

2

(*j*)

(*j*)

*σ σ*

(*j*)

*kth promising region σ*(*k*) *for solving Problem* [*1*](#_bookmark0)*, as a*

*result of Propositions* [*1*](#_bookmark12) *and* [*2*](#_bookmark15)*. Then Partitions procedure*

*where δσ* = (*lσ* + *uσ* )*/*2 *is the midpoint of the rect*

*line segment defined by both l*(*j*) *and u*(*j*)*.*

*yields Mσ* = 2*n*+*m disjoint subregions from the current promising region σ*(*k*)*.*

*σ σ*

**Proof.** We know that a partition of an integer number set into two subsets must then yield two disjoint subsets and the union of both subsets must contain all elements of

the original set.

**Proof.** Due to the fact that according to Partitioning procedure in Figure [18](#_bookmark56), at each *k*th iteration, both each of the *n* real components is divided by into 2 subsets and each of the *m* integer components is also divided into 2

(*j*)

(*j*)

subsets, yielding respectively at each iteration 2*n* and 2*m*

As can be seen: first, *σ*1 (*k*) ∩ *σ*2 (*k*) = ∅, and second,

*σ*(*j*)(*k*) = *σ*(*j*)(*k*) ∪ *σ*(*j*)(*k*), what has allowed us to

subsets, thus, this fact allows us to obtain *Mσ* = 2*n*+*m*.

1 2

partition the set *σ*(*j*)(*k*) properly.

As can be seen from Partitions procedure in Figure [18](#_bookmark56),

at each *k*th iteration a set of {*σi*(*k*)}*Mσ* subregions from

(*j*)

¯(*j*)

(*j*)

the current promising region *σ*(*k*)

*i*=1

**Proposition 2** *Let σ*¯ (*k*) = {*y* ∈ Z : *lσ* ≤ *y* ≤ *u*¯*σ* }

*be the jth interval of integer number or set of the*

*jth integer value component that defines the current*

are mutually disjoint subsets, i.e., *σi*(*k*) ∩ *σi*(*k*) = ∅ for all *i* /= *j* and *i, j* ∈

{*ℓ*}*Mσ* , and moreover *σ*(*k*) = S*Mσ σ* (*k*).

*ℓ*=1 *i*=1 *i*

## ON THE STOPPING RULE

As was mentioned in the introduction, one of the con- tributions of this research has been the inclusion of a new stopping rule, which is based on the depth vector *d*(*k*) of the current promising region, without considering any objective function value, because our approach is only based on partitioning scheme, leaving for another

research the study of stopping rule based on statistical viewpoints, such as statistical considerations on the feasible global minimum function value applying non- standard parametric inference. A good approach on non- standard parametric inference is introduced by Cheng **[**[**23**](#_bookmark82)**]**, which could be taken into account for using it together with the concepts of order statistics presented

#### EBERT BREA

*where l, u* ∈ R*n* × Z*m respectively are the lower and upper bound constrains given by* ([1b](#_bookmark1))*.*

**Proof.** Due to the fact that the MINP method partitions each *i*th component of the promising region into two subsets, the *i*th real component is successively divided by 2 at each *k*th, resulting the geometric series given by ([7](#_bookmark21)), for each *i*th real component of the mixed integer depth vector.

**Lemma 2** *Assume Condition* [*1*](#_bookmark18) *holds and the ex- pected mixed integer depth vector ε is fixed to* (*ǫ, . . . , ǫ*; 0*,...,* 0)*t. The MINP method hence stops when*

by De Haan **[**[**24**](#_bookmark83)**]**.

Taking into account the main pseudocode of the MINP method in Figure [8](#_bookmark25) on page [124](#_bookmark25), we can notice that when

*d*(*k* + 1) ≤ *d*(*k*) holds, and the ˆ*i* ∈ {*ℓ*}*Mσ* the algorithm

*ℓ*=1

max

*i*∈{1*,...,n*}

(*u*(*i*) − *l*(*i*))

2*k < ǫ* (8)

carries out a new partitioning on the new promising region, otherwise, as a result of that ˆ*i* = *Mσ* + 1, the algorithm backtracks to the entire feasible region, and it restarts the iteration counter by letting *k* be equal to 0. Note that the surrounding to the promising region *σ*(*k*) has been here denoted by *σMσ* +1(*k*), i.e., *S*(*σ*(*k*)) =

*σM* +1(*k*).

*is met.*

**Proof.** During the execution of the algorithm, successive divisions by two are carried out, both to the real and integer components, what allows us to assert that any

integer component *d*¯(*i*)(*k*) for all *i* ∈ {*ℓ*}*m* of the

*ℓ*=*n*+1

*σ* mixed integer depth vector becomes 0, faster than any

Since the MINP method estimates the current promising region at each *k*th, the current depth mixed integer vector *d*(*k*) can be then easily calculated, for being

real component *d*(*i*)(*k*) for all *i* ∈ {*ℓ*}*n* of the mixed integer depth vector. Besides, all integer components

of the mixed integer vector *d*¯(*i*)(*k*) for all *i* ∈ {*ℓ*}*m*

*ℓ*=1

afterward tested at the end of each *k*th iteration, and so

become 0 in a finite number of iteration *k*

*ℓ*=*n*+1

, whereas that

verifying whether *ε* ≤ *d*(*k*) or not, whereby in the case

the real components *d*(*i*)(*k*) for all *i* ∈ {*ℓ*}*n*

approach

that *ε* ≤ *d*(*k*) the MINP method will go to next iteration,

to 0 when *k* → ∞

*ℓ*=1

. Since Lemma [1](#_bookmark19) holds, the depth

otherwise, the MINP method will stop.

We shall now focus on calculating the maximum num- ber of iteration, denoted by *k*ˆ, that the MINP method

needs for stopping the algorithm, considering that any backtracking operation has not been carried out during the identification of a global solution to Problem [1](#_bookmark0).

Due to the fact that we have still no stopping rule for the MINP method, it is reasonable to enunciate the following condition.

**Condition 1** *Suppose that is executed the iterative loop of the MINP method, that will afterward be shown in Fig- ure* [*8*](#_bookmark25)*, without considering any stopping rule for globally solving Problem* [*1*](#_bookmark0)*.*

**Lemma 1** *Assume Condition* [*1*](#_bookmark18) *holds. Then the ith real*

vector *d*(*k*) will therefore be precede to the expected mixed integer depth vector *ε*, when ([8](#_bookmark17)) is met.

In what follows, we provide evidence that the proposed stopping rule successfully works, avoiding an over itera- tions of the algorithm, when the stopping rule had been met, as a result that exists a positive probability that is reached.

**Theorem 1** *Let k*ˆ = ⌈*k*⌉ *be the closest integer greater than or equal to k, assume Condition* [*1*](#_bookmark18) *holds, and the expected mixed integer vector ε has been fixed to* (*ǫ, . . . , ǫ*; 0*,...,* 0)*t. Then minimum number of required*

*iterations by the MINP method for being stopped, i.e., when d*(*k*ˆ) ≺ *ε has just been met, it is given by*

*component of the mixed integer depth d*(*k*) *is given by*

 

## max

2

(*u*(*i*) − *l*(*i*)) 

(*i*)

(*i*)

*k*ˆ 

 *i*∈{1*,...,n*} 

*d*(*i*)

*u* − *l*

## (*k*) =

2*k*

*,* ∀*i* ∈ {*ℓ*}

*n ℓ*=1

*, k* ∈ N*,* (7)

= log 

#### 

*ǫ*



 *.* (9)

#### 



#### EBERT BREA

**Proof.** Getting *k* by solving ([8](#_bookmark17)) from Lemma [2](#_bookmark16), the MINP method stops at iteration *k*ˆ given by ([9](#_bookmark20)).

Now, we shall pay attention on the estimation of the probability of carrying out a backtracking operation to the

entire region Θ at any *k*th iteration, such that 0 *< k < k*ˆ.

**Remark 2** *Assume Condition* [*1*](#_bookmark18) *holds. Each kth promis- ing region is therefore given by*

*σ*(*k*) = {*z* ∈ R*n* × Z*m* : *lσ* ≤ *z* ≤ *uσ*}*,* (10)

## ON THE PERFORMANCE MEASUREMENT

We now consider the issue of measuring the perfor- mance of the MINP method, when it has been collected a set of solutions that has been yielded from *r* independent runs of the MINP method, under the same conditions.

For this reason, we shall here show the main results on the proposed performance measurement, which was introduced by Brea **[**[**25**](#_bookmark84)**]** for comparing the Game of Patterns **[**[**9**](#_bookmark68)**]** versus two implementations of Genetic Al- gorithms, when these last mentioned algorithms are

*where l*

= (*l*(1)*, . . . , l*(*n*); *l*¯(*n*+1)*, . . . , l*¯(*n*+*m*))*t and u* =

used for globally solving constrained nonlinear problems

*σ σ σ σ σ*

(1) ( ) ( +1) + )

*σ* in R*n*

### (uσ , . . . , u n ; u¯ n

*σ*

*σ*

*σ*

*, . . . , u*¯(*n*

*m* )*t respectively are the*

*lower and upper mixed integer bounds of the kth promis-*

*ing region.*

**Proof.** Since Condition [1](#_bookmark18) is always satisfied, at each *k*th iteration is rightly partitioned the current promising region *σ*(*k*) due to Propositions [1](#_bookmark12) and [2](#_bookmark15), what allow us to describe exactly *σ*(*k*) by ([10](#_bookmark22)).

Note that by Partitioning procedure in Figure [18](#_bookmark56) clearly defines each generated *k*th promising region *σ*(*k*).

**Theorem 2** *Suppose that there is no any stopping rule*

**Proposition 3** *Let p be a mixed integer bound con- strained problem in* R*n* × Z*m; let η*(*ℓ*)(*p, n, m*) *>* 0 *be the number of times that the objective function has been evaluated during ℓth run of the algorithm for searching of*

*a solution to Problem p; and let λ*(*ℓ*)(*p, n, m*) ≥ 0 *be the distance between the true point or solution of Problem p,*

*and the best achieved solution by the MINP method at the ℓth replication. Then, for each ℓ* ∈ N+*,*

*for the MINP method. If ν and N are both positive quantities, there will then exist a positive probability that the algorithm goes forward in depth, getting a new depth vector d*(*k* + 1) *that precedes to d*(*k*)*, at any kth*

*iteration, i.e., Pr*{*d*(*k* + 1) ≺ *d*(*k*)} *>* 0 *for all k* ∈ N*.*

*q*(*ℓ*)(*p, n, m*) = 1

1 + *η*(*ℓ*)(*p, n, m*) *· λ*(*ℓ*)(*p, n, m*)

*is on* (0*,* 1]*.*

*,* (11)

**Proof.** Using Propositions [1](#_bookmark12) and [2](#_bookmark15), we clearly have that the event {*d*(*k* + 1) ≺ *d*(*k*)} takes place with a positive probability.

The next result shows how Theorem [2](#_bookmark23) allows us to affirm that the MINP method does not over run for getting a global solution, namely, it finishes with a positive probability.

**Corollary 1** *Suppose Condition* [*1*](#_bookmark18) *holds for globally solving Problem* [*1*](#_bookmark0)*. Besides, assume that ν and N are both positive quantities; and the expected mixed integer vector ε is fixed to both real and integer no*

*negative components. Then the MINP method will meet the stopping rule, i.e., d*(*k*) ≺ *ε in a limited number of successive iterations with a positive probability.*

**Proof.** Because of Theorem [2](#_bookmark23), there exists a positive

probability that the MINP method reaches the stopping rule *d*(*k*) ≺ *ε*, and then it does not carry out any more iterations, causing a stopping of the algorithm.

Further results on the behavior of the MINP method will afterward be presented in next section.

**Proof.** It suffices to verify the implication ([11](#_bookmark24)), due to the fact that *η*(*ℓ*)(*p, n, m*) *>* 0 and *λ*(*ℓ*)(*p, n, m*) ≥ 0, what easily allows us to prove that for each *ℓ*th algo-

rithm run or also called replication is yielded a sampling

*q*(*ℓ*)(*p, n, m*) ∈ (0*,* 1].

Of course, that any collection {*q*(*ℓ*)(*p, n, m*)}*r* of *r*

*ℓ*=1

independent executions of the algorithm would then yield

an empirical distribution so far from being a normal distribution, whereby a parametric statistical analysis, under standard situations, of these data would be unsuit- able for obtaining a right description of the here called random variable quality, which will hereafter be denoted

by *Q*(*p, n, m*), and given by

## 1

*Q*(*p, n, m*) = *,* (12)

1 + *N* (*p, n, m*) *L*(*p, n, m*)

where *N* (*p, n, m*) and *L*(*p, n, m*) respectively are the random variables: number of function evaluation and dis- tance to the true point, and whose distribution functions are unknown.

#### EBERT BREA

Algorithm. The Mixed Integer Nested Partitions Method

Initialization Given:

the number of real components, *n*;

the number of integer components, *m*;

the problem minimize *f* (*z*)*,*

*z*∈R*n*×Z*m*

subject to *l* ≤ *z* ≤ *u,*

where *l, u* ∈ R*n* × Z*m*, e.i.: *l* = (*l*(1))*, . . . , l*(*n*)); ¯*l*(*n*+1)*, . . . ,* ¯*l*(*n*+*m*))*t* and *u* = (*u*(1))*, . . . , u*(*n*)); *u*¯(*n*+1)*, . . . , u*¯(*n*+*m*))*t*;

` ˛¸ x ` ˛¸ x ` ˛¸ x ` ˛¸ x

*n m n m*

the number *Mσ*(0) of subregions of the current most promising region *σ*(0) = Θ;

the number of sample points *Nj* (*k*) from each *j*th subregion *σj* (*k*) at each *k*th iteration;

Choose:

an *ε*(*i*) ∈ R+ for each *i* ∈ {1*, . . . , n*};

an *ε*¯(*i*) ∈ N for each *i* ∈ {*n* + 1*, . . . , n* + *m*}, what allows us to define the expected maximum depth of the mixed

integer vector, *ε* = (*ε*(1)*, . . . , ε*(*n*); *ε*¯(*n*+1)*, . . . , ε*¯(*n*+*m*))*t* ∈ R*n* × N*m*;

+

` ˛¸ x ` ˛¸ x

*n m*

Declare:

the mixed integer depth vector *d*(*k*) = (*d*(1)(*k*)*, . . . , d*(*n*)(*k*); *d*¯(*n*+1)(*k*)*, . . . , d*¯(*n*+*m*)(*k*))*t*;

` ˛¸ x ` ˛¸ x

*n m*

Let *k* = 0;

Calculate:

the initial mixed integer depth vector, e.i., *d*(0) = D(0*, n, m, σ*(0)*,* Θ), of the promising region *σ*(0);

while *ε* ≤ *d*(*k*) do Partitioning

Partition the current promising region *σ*(*k*) into

*M*

{*σj* (*k*)}*j*=1 subregion;

*σ*(*k*)

Aggregate the current surrounding region *σMσ*(*k*)+1 (*k*) = Θ \ *σ*(*k*), and denote it as *S*(*σ*(*k*));

Random Sampling

Execute the Sampling Procedure for taking *Nj* (*k*) random points from each *j*th subregion

*Mσ*(*k*)

{ *j* ( )}*j*=1

*σ k* ,

namely: for each *j*th subregion, get the set of sampled points {*θ*

*Nj* (*k*)

*j,s s*=1

} ;

Execute the Sampling Procedure for also taking *Nσ*(*k*)+1(*k*) random points from surrounding region

*σMσ*(*k*)+1

1. , that is, for getting {*θ*

*NMσ*(*k*)+1 (*k*)

*Mσ*(*k*)+1 *,s*}*s*=1 ;

Measuring

Calculate the performance of each got sampled points using the objective function *f* (*θ*), namely: *f* (*θ*)|*θj,s* for

*j*th subregion at each *s*th random sampling, and also the sampled points from the surrounding region

*σMσ*(*k*)+1 (*k*);

Estimating

Estimate the promising region, that is, finding the index of the best performance, namely, first estimate

*I*ˆ(*σj* (*k*)) = min

*s*∈{1*,...,Nj* }

*f* (*θj,s*)*,*

and then ˆ*jk* = min

*j*∈{1*,...,Mσ*(*k*)+1 }

*I*ˆ(*σj* (*k*))*.*

In the case that two or more regions are equally promising, the tie can be randomized broken.

Decision

if ˆ*jk* ≤ *Mσ*(*k*) then

Let *σ*(*k* + 1) = *σ*ˆ (*k*);

*j*

*k*

Update by using *d*(*k* + 1) = D(*k* + 1*, n, m, σ*(*k* + 1)*,* Θ);

Let *k* ← *k* + 1;

else

Backtrack to the entire feasible region Θ;

Let *k* ← 0;

Let *σ*(*k*) = Θ;

Update by using *d*(0) = D(0*, n, m, σ*(*k*)*,* Θ);

Figure 8. The MINP method

# THE PSEUDOCODE OF THE MINP

METHOD

Taking into account the theoretical main results above, it

*d*˜(*k*) =

*u*(1) − *l*(1)

2*k , . . . ,*

*u*(*n*) − *l*(*n*)

2*k* ;

#### EBERT BREA

has been proposed the MINP method, which is shown in

Figure [8](#_bookmark25) using an easy description of its source code. The figure shows: the preamble of the algorithm, in which given: i) the problem; ii) a chosen expected depth vector; and iii) a declared vector *d*(*k*), the MINP method

,

*u*¯(*n*+1) − ¯*l*(*n*+1)

2*k*

,

*, . . . ,*

,

*u*¯(*n*+*m*) − ¯*l*(*n*+*m*)

2*k*

,!*t*

(14)

executes iterative loops for solving Problem [1](#_bookmark0), whilst the stopping rule has not been met.

# ON THE MINP METHOD

In this section we study the main properties of the MINP method from a stochastic viewpoint. According to the MINP method, it is reasonable to figure up that the MINP method generates a Markov stochastic process on a countable and infinite discrete stochastic state space, i.e., a Markov chain that can be described by a set

of {*D*(*κ*)}∞ , where *κ* ∈ N here depicts a cumulative

*κ*=0

iteration counter of the MINP method, what means in a

nonmathematical language: *the future only depends on*

*Then d*˜(*k*) *depicts the measurements of the smallest baggy hull in* R*n* × Z*m and thus d*˜(*k*) ≥ *d*(*k*) *for each k* ∈ N*.*

**Proof.** For proving this statement, we first pay attention to the real components, and later to the integer compo- nents of the promising region *σ*(*k*).

Because of Proposition [1](#_bookmark12), each *j*th real component of the current promising region *σ*(*k*) has successively been

divided by 2, yielding as a result consecutive partitions into two subsets per each *k*th iteration, we therefore have by ([5a](#_bookmark13)) and ([5b](#_bookmark14)) that

*today, and not what have occurred in past time*

(*j*)

(*j*)

*u*(*j*) − *l*(*j*)

We shall therefore turn up our attention to the Markov

chain state space that is generated by the MINP method,

*δσ*(*k*) − *lσ*(*k*) ≤

2*k* ; (15a)

when it is run for globally solving any mixed integer prob-

(*j*)

(*j*)

*u*(*j*) − *l*(*j*)

lem given by Problem [1](#_bookmark0). However, before introducing our

main results on the Markov chain that is generated by the

*σ*(*k*)

*uσ*(*k*) − *δσ*(*k*) ≤

2*k ,* (15b)

MINP method (MINPMC), we must point out that each

where *δ*(*j*)

(*j*)

*σ*(*k*)

= (*l*

+ *u*

(*j*)

*σ*(*k*)

)*/*2.

state of the MINPMC space will be measured by a depth vector of the *k*th baggy hull of the current promising region *σ*(*k*), which will be called baggy hull depth vector, and we shall then be defined it as follows.

Secondly, due to Proposition [2](#_bookmark15), each *j*th integer compo- nent of the current promising region *σ*(*k*) has succes- sively been partitioned into a pair of subsets, consecu-

tively generating two subsets not necessarily the same size per each *k*th iteration, hence, by ([6](#_bookmark11)) we know that

if *δ*

= (*l*

+ *u*¯

##### Definition 7 (Baggy hull of a promising region) *Let*

(*j*)

*σ*(*k*)

⌊*δ*

⌋ = ⌈*δ*

¯(*j*)

*σ*(*k*)

(*j*)

*σ*(*k*)

)*/*2 is an integer number, i.e.,

*σ*(*k*) *be a kth nonempty promising region given by*

*σ*(*k*) = {*z* ∈ R*n* × Z*m* : *lσ*(*k*) ≤ *z* ≤ *uσ*(*k*)}*,* (13)

(*j*)

*σ*(*k*)

(*j*)

*σ*(*k*)

⌉, then

, ,

(*j*) ¯(*j*)

*u*¯(*j*) − ¯*l*(*j*)

*with mixed integer depth vector d*(*k*) = *u*

*σ*(*k*)

− *lσ*(*k*)

*. It*

*δσ*(*k*) − *lσ*(*k*) ≤

; (16a)

2*k*

*says to be a baggy hull of σ*(*k*)*, denoted by σ*˜(*k*)*, is the smallest oversized hull σ*(*k*) *such that σ*˜(*k*) ⊇ *σ*(*k*)*.*

(*j*)

*u*¯

*σ*(*k*)

(*j*)

*σ*(*k*)

− *δ*

## − 1 ≤

, *u*¯(*j*) − ¯*l*(*j*) ,

2*k*

*.* (16b)

Basing on this last definition, we thus have the following result.

Whereas, if *δ*(*j*)

is a non integer number, i.e., ⌊*δ*(*j*)

## ⌋ /=

##### Proposition 4 *Let*

*d*˜(*k*) *be the kth baggy hull depth*

*σ*(*k*)

*σ*(*k*)

(*j*)

*σ*(*k*)

⌈*δ*

⌉, then

, ,

*vector of the current promising region σ*(*k*)*, which is*

(*j*)

¯(*j*)

*u*¯(*j*) − ¯*l*(*j*)

*given by*

⌊*δσ*(*k*)⌋ − *lσ*(*k*) ≤

## ; (17a)

2*k*

(*j*)

(*j*)

, *u*¯(*j*)

#### ,

− ¯*l*(*j*)

#### EBERT BREA

*u*¯*σ*(*k*) − ⌈*δσ*(*k*)⌉ ≤

*.* (17b)

2*k*



*p*(*k,*0)

*k*

. . .

*p*(*k*−1*,*0)

*k*−1

. . .

*p*(2*,*0)

2

*p*(1*,*0)

1

*d*˜(0)

*d*˜(1)

*d*˜(2) · · · *d*˜(*k*−1)

*d*˜(*k*)

*p*(0*,*1)

0

*p*(1*,*2)

1

*p*(2*,*3) *p*(*k*−2*,k*−1)

2 *k*−2

*p*(*k*−1*,k*)

*k*−1

*p*(*k,k*+1)

*k*

Note that the measurements of any *k*th promising region *σ*(*k*) are given by all the components of *d*(*k*), and sim- ilarly denoting *σ*˜(*k*) as the baggy hull of the correspon- dent promising region *σ*(*k*), and whose measurements are given by all the components of *d*˜(*k*) ∈ N, we can

hence say that *σ*(*k*) ⊆ *σ*˜(*k*).

We are now interested in describing the Markov chain that is generated by the MINP method, we can hence say that due to Proposition [4](#_bookmark27), the Markov chain from

the MINP method (MINPMC) can rigorously be ex- plained using ([14](#_bookmark26)), yielding either, under Condition [1](#_bookmark18) an unlimited countable set of discrete stochastic states

{*D*(*κ*) = *d*˜(*k*)}∞ for all *k* ∈ N; or under Condition

*k*=0

Figure 9. The MINP Markov chain

**Condition 2** *Suppose that the MINP method has suc- cessfully been run for globally solving Problem* [*1*](#_bookmark0) *apply- ing proposed the algorithm in Figure* [*8*](#_bookmark25)*.*

**Condition 3** *Assume that the MINP method uniformly takes random samplings by randomized located trial points from both each subregion and the current sur-*

*rounding region, using both independent and identically*

1. a limited countable set of discrete stochastic states

{*D*(*κ*) = *d*˜(*k*)}*k*ˆ , where *k*ˆ is given by ([9](#_bookmark20)).

*k*=0

*distributed (i.d.d.) continuous uniform random variables*

*and i.d.d. discrete uniform random variables.*

Observe the difference that has been made between the counters *k* and *κ* of the algorithm. The first one means the number of iterations that are executed by

the algorithm from initial state *D*(*κ*) = *d*˜(0); whilst the

second one depicts the number of cumulative iterations that has been carried out by the MINP method from its starting.

We shall now introduce a measurement of mixed integer mixed region size, which will allows us to estimate the probability of the MINPMC events.

**Definition 8 (Hypervolume of a region)** *Let* V[*σ*(*k*)]

*denote the hypervolume of σ*(*k*)*, which is defined by*

Let

*p*(*k,ℓ*) = *Pr*{*D*(*κ* + 1) = *d*˜(*ℓ*) | *D*(*κ*) = *d*˜(*k*)} (18)

*κ*

Y*n*

V[*σ*(*k*)] =

*ℓ*=1

(*ℓ*)

*σ*(*k*)

*u*

(*ℓ*)

*σ*(*k*)

− *l*

Y*m*

## +

*ℓ*=1

(*ℓ*)

*σ*(*k*)

*u*¯

¯(*ℓ*)

*σ*(*k*)

− *l ,*

denote the single-step transition probability that the MINPMC will visit state *D*(*κ* + 1) = *d*˜(*ℓ*) at the cumu-

∀*k* ∈ N*.*

## (20)

lative iteration *κ* + 1, if the MINPMC is at state *D*(*κ*) = *d*˜(*k*) at the *κ*th cumulative iteration counter, for all *k, κ* ∈ N and *ℓ* ∈ {0*, k* + 1}. Nevertheless, it is reasonable

to assume that ([18](#_bookmark32)) dependents on *k*, whereby the

Note that Definition [8](#_bookmark31) can also be used for calculating the baggy hull hypervolume of any promising region, which is

MINPMC is said to be an nonhomogeneous Markov chain, what will hereafter be denoted by

## V[*σ*˜(*k*)] =

Y*n*

*ℓ*=1

*u*(*ℓ*) − *l*(*ℓ*) !

2*k* +

Y*m* , *u*¯(*ℓ*) − ¯*l*(*ℓ*) ,

*k ,*

2

*ℓ*=1

## (21)

*p*(*k,ℓ*) =*Pr*{*D*(*k* + 1) = *d*˜(*ℓ*) | *D*(*k*) = *d*˜(*k*)}*,*

*k*

∀ *k, κ* ∈ N*, ℓ* ∈ {0*, k* + 1}*.*

## (19)

∀*k* ∈ N*.*

Based on ([21](#_bookmark33)), we shall now consider the following proposition.

Figure [9](#_bookmark28) illustrates the MINPMC, wherein each state is symbolized by denoted rounded node *d*˜(*k*), and each

transition probability *p*(*k,ℓ*) by directed arcs, which rep- resent the probability that the MINPMC changing from *d*˜(*k*) state to either *d*˜(*ℓ*) state for each *ℓ* ∈ {0*, k* + 1}, at

*k*

*k*th iteration counter.

We now consider as a successful event F, which has occurred as a consequence of that the MINP method

has discovered the existence of the best value of the objective function comes from any subregion, among

all sampled trial points at a *k*th iteration, i.e., at a *d*˜(*k*)

state, and by contrast, let B denote the event that a

#### EBERT BREA

*i*=1

backtracking operation is carried out by the algorithm as a result that the MINP method identified the best

function value comes from the *k*th surrounding region at the same *k*th counter iteration, therefore, a geometric

distribution is taken place in the estimation of the proba-

got from a binomial distribution with *Mσ* subregions, because a set {*σi*(*k*)}*Mσ* is generated by Partition pro- cedure in Figure [18](#_bookmark56) at each *k*th iteration, and parameter *γk* expressed by

bility distribution of the MINPMC states.

V[*σ*˜*i*(*k*)]

Q*n u*(*ℓ*) −*l*(*ℓ*)

2*k*+1 +

Q*m* , *u*¯(*ℓ*)−¯*l*(*ℓ*) ,

2*k*+1

**Proposition 5** *Assume, without loss of generality (w.l.o.g.), Condition* [*1*](#_bookmark18) *and* [*3*](#_bookmark30) *are met, and let Ni* = *ν*(*>* 0) *for all i* ∈ {*ℓ*}*Mσ be the number of random samplings*

*ℓ*=1

*γk* =

V[*σ*˜(*k*)]

*ℓ*=1

Q*n*

=

*ℓ*=1

*u*(*ℓ*) −*l*(*ℓ*)

2*k*

*ℓ*=1

Q*m* ,

## +

*ℓ*=1

*u*¯(*ℓ*)−¯*l*(*ℓ*)

2*k*

, *,* (27)

*that are taken from each ith subregion σi*(*k*) *at the*

for all *k* ∈ N+, because V[*σ*˜1(*k*)] =*,* · · · *,* = V[*σM*˜

(*k*)],

*κth iteration, and let N* (*>* 0) *be the number of random*

and hence *γk*

*σ*

= V[*σ*˜*i*(*k*)]*/*V[*σ*˜(*k*)] for any *i* ∈

*trial points that are taken from the surrounding region*

*S*(*σ*(*k*))*. Then*

#### (

1*, if k* = 0*;*

{1*,..., Mσ*}, what yields

#### !

*Pr*{Σ | F} = *Mσ γ* (1 − *γ* )*Mσ* −1*,* ∀*k* ∈ N

*.* (28)

*p*(*k,k*+1) =

*k*

*ϕ Mσ γk* (1 − *γk*)*Mσ* −1*, if k* ∈ N+*,*

*i* 1 *k k* +

*where:*

*ϕ* =

*Mσν*

## (22)

; (23a)

By substituting ([26](#_bookmark38)) and ([28](#_bookmark36)) in ([25](#_bookmark37)), we obtain

*p*(*k,k*+1) = *ϕ Mσγk* (1 − *γk*)*Mσ* −1*,* ∀*k* ∈ N+*.* (29)

*k*

*Mσν* + *N*

Q*n u*(*ℓ*)−*l*(*ℓ*)

2*k*+1

Q*m* ,

## +

*u*¯(*ℓ*)−¯*l*(*ℓ*) ,

2*k*+1

Besides, the event that the MINPMC visits the {*D* = 1}

state after it has been at the {*D* = 0} state always oc-

curs, leaving no doubt, because *S*(*σ*(0)) = ∅, whereby

*γk* = *ℓ*=1

Q*n*

*ℓ*=1

*u*(*ℓ*)−*l*(*ℓ*)

2*k*

*ℓ*=1

Q*m* ,

## +

*ℓ*=1

*u*¯(*ℓ*)−¯*l*(*ℓ*)

2*k*

, *,* ∀ *k* ∈ N+*.*

## (23b)

*p*(0*,*1) = 1.

Therefore, we shall hereafter denote, without causing a loss of its meaning, *Pr*{*D*(*k*) = *d*˜(*k*)} as *Pr*{*D*(*k*) =

0

**Proof.** Let F denote the event that the MINP method has identified that the best value of *f* (*z*) has come from the current subregion *σ*(*k*), let Σ*i* denote the event that the best value of *f* (*z*) has come from any *i*th subregion *σi*(*k*) at the *k*th iteration, then,

*p*(*k,k*+1) = *Pr*{Σ*i,* F}*,* ∀*k* ∈ N+*.* (24)

*k*

Using conditional probability, we have

*p*(*k,k*+1) = *Pr*{Σ*i* | F} *Pr*{F}*,* ∀*k* ∈ N+*.* (25)

*k*

Under Condition [3](#_bookmark30), we can say that letting *Pr*{F} = *ϕ*

*Mσν*

*ϕ* = *.* (26)

*k*} = *π*(*k*).

We shall now pay especial attention to the probability distribution estimating of the MINPMC states, which

may be expressed by a (*k*ˆ+1) dimensional vector *π* =

(*π*(0)*, . . . , π*(*k*ˆ))*t*, which depicts

ˆ

*k*

*π*(*k*) = Σ *π*(*ℓ*) *δ*[*k* − *ℓ*]*,* ∀*k* ∈ Z*,* (30)

*ℓ*=0

where *δ*[*k*] : Z → {0*,* 1} is here called discrete impulse function, which is a special case of a 2-dimensional Kronecker delta function *δij*, and a generalized *ℓ*-shifted mathematical expression of *δ*[*k* − *ℓ*] is given by

#### 

1*,* if *k* = *ℓ*;

*Mσν* + *N*

On the other hand, due to Conditions [2](#_bookmark29) and [3](#_bookmark30), and

*δ*[*k* − *ℓ*] =

0*,* another case of *k* ∈ Z. (31)

Sampling procedure in Figures [20](#_bookmark57) and [21](#_bookmark58), and since sampling trial points are taken from uniform random dis- tributions for each subregion and the current surrounding region *S*(*σ*(*k*)), we can then say that ([22](#_bookmark35)) is easily

Note that we have used in ([30](#_bookmark39)) *π*(·) for denoting the

probability distribution function, and this is not to be confused with the superscript argument notation *π*(·)

conventionally used for denoting the component of a

vector, in this case, *π* vector.

**Proposition 6** *Suppose that Conditions* [*2*](#_bookmark29) *and* [*3*](#_bookmark30) *hold, and let π*(0) = *Pr*{*D*(0) = 0} *be the probability that the MINPMC visits the state D*(0) = *d*˜(0)*, which of course*

*always occurs by the starting of the MINP method; or by backtracking operations, which can be eventually carried*

#### EBERT BREA

Knowing that *p*(*j,j*+1) is estimated by Proposition [5](#_bookmark34), and using both ([34](#_bookmark41)) and ([37](#_bookmark44)) the proof is completed.

*j*

For almost ending this analysis, we apply the above results and ([30](#_bookmark39)) together for getting our mathematical expression of the probability distribution function of the MINPMC states, yielding

*out by the algorithm. Then*

1 *k*ˆ

*ℓ*−1

#### 

*π*(0)*, if ℓ* ∈ {0*,* 1}*;*

*π*(*k*) =Σ *π*(0) *δ*[*k* − *ℓ*] +Σ *π*(0) Y *p*(*j,j*+1)*δ*[*k* − *ℓ*]*,*

*π*(*ℓ*) =

*π*(0)

Q*ℓ*−1 *p*(*j,j*+1)*, if ℓ* ∈ {2*,..., k*ˆ}*,*

## (32)

*ℓ*=0

∀*k* ∈ Z*.*

*ℓ*=2

*j*=1

*where:*

*j*=1 *j*

*π*(0) = 1

## ; (33)

(38)

The results heretofore obtained allow us to describe

Σ*k*ˆ

## 2 +

*ℓ*Q−1

(*j,j*+1)

*j*

*p*

the behavior of the MINP method, which is a nonho- mogeneous Markov chain with a conditional geometric

(*j,j*+1)

*ℓ*=2 *j*=1

distribution of states given by ([38](#_bookmark40)).

*and pj is given by* ([22](#_bookmark35))*.*

**Proof.** By induction, we have

Finally, we shall make the follows statements for justi- fying our approach: By Proposition [4](#_bookmark27) and taking into

account that *σ*(*k*) ⊆ *σ*˜(*k*), we can assert that for each

*π*(*ℓ*) = *π*(0)

We besides know,

*ℓ*Y−1

*j*=0

*k*ˆ

*p*(*j,j*+1)*,* ∀*ℓ* ∈ N+*.* (34)

*k* ∈ N, V[*σ*˜(*k*)] ≥ V[*σ*(*k*)], and besides, by ([21](#_bookmark33)) we have that V[*σ*˜(*k*)] ≥ V[*σ*˜(*k* + 1)], what would allow us to

say as a reasonable conjecture, that the MINP method concentrates its finding better promising regions during the progress of its iterative process, due to the fact of the continuous process of reduction of the baggy hull of the

*j*

Σ *π*(*ℓ*) = 1*.* (35)

*ℓ*=0

Substituting ([34](#_bookmark41)) in ([35](#_bookmark42)), and knowing that *p*(0*,*1) = 1

0

because of ([22](#_bookmark35)), we therefore deduce

identified promising region.

# OPERATING THE MINP METHOD

In this section, we shall show a main software that will operate the MINP method for taking the collec- tion of the performance measurements: *η*(*ℓ*)(*p, n, m*);

*j*

ˆ

*k*

2 *π*(0) +Σ *π*(0)

*ℓ*=2

*ℓ*Y−1

*j*=1

*p*(*j,j*+1) = 1*.* (36)

*λ*(*ℓ*)(*p, n, m*); and *q*(*ℓ*)(*p, n, m*) from each *ℓ*th indepen- dent replication of the MINP method is depicted in Figure [10](#_bookmark45).

Solving ([36](#_bookmark43)) for *π*(0), we effortlessly obtain

As is shown in Figure [10](#_bookmark45) on page [129](#_bookmark45), the MINP method is called for being r-times executed with different random

*π*(0) =

Σ*k*ˆ

## 2 +

1

*ℓ*Q−1 *p*(*j,j*+1)

*.* (37)

seed each, thus achieving r independent runs for taking

i.d.d. unknown function empirical distribution of the performance measurements, when it is used for globally

*j*

*j*

*ℓ*=2 *j*=1

solving Problem [1](#_bookmark0).

#### EBERT BREA

**Main software** of the Mixed Integer Nested Partitions

##### Given:

a *p*th bound constrained mixed integer nonlinear problem P*p*

minimize *f* (*z*)*,*

*z*∈R*n* ×Z*m*

subject to *l* ≤ *z* ≤ *u,*

where *l, u* ∈ R*n* × Z*m*;

the global minimum or considered true point *z*ˆ*p* of the bounded constrained mixed integer nonlinear problem P*p*; a maximum number of replication, *r*;

a set of *S* random generator, namely, Ω = {*u*(*s*)}I(*S*)

*s*=I(1)

, which depends on the *s*th random seed I(*s*);

the MINP(*s, p, n, m*) algorithmic method;

##### Declare:

a counter replication, *ℓ* ∈ {1*, . . . , r*};

an *ℓ*th *z*ˇ*ℓ* ∈ R*n* × Z*m* point, which will be used for saving the best identified point by the *ℓ*th running of the

MINP(*s, p, n, m*) method;

an *ℓ*th performance measure sample *q*(*ℓ*)(*p, n, m*) for the (*n* + *m*) multidimensional *p*th problem, which is given by

*q*(*ℓ*)(*p, n, m*) = 1

1 + *η*(*ℓ*)(*p, n, m*) · *λ*(*ℓ*)(*p, n, m*)

*,* ∀*ℓ* ∈ N+*,*

where: *η*(*ℓ*)(*p, n, m*) ∈ N is the number of times that has been evaluated the objective function during the *ℓ*th running of the MINP(*s, p, n, m*) method; and *λ*(*ℓ*)(*p, n, m*) = ||*z*ˆ − *z*ˇ|| ∈ R is the distance or norm between the true point or global optimum point *z*ˆ and the best point *z*ˇ identified by the MINP(*s, p, n, m*);

**for** *ℓ* ← 1 **to** *r* **do**

**Choose** a not used *s*th random seed for getting an i.i.d. random number generator for each *ℓ* replication;

**Run** the MINP(*s, p, n, m*) method for solving the (*n* + *m*) multidimensional *p*th problem;

**Compute** the *ℓ*th *q*(*ℓ*)(*p, n, m*);

##### Save:

the *ℓ*th of *η*(*ℓ*)(*p, n, m*), *λ*(*ℓ*)(*p, n, m*) and *q*(*ℓ*)(*p, n, m*);

##### Estimate:

minimum, mode, mean, maximum, range, and deviation of *N* (*p, n, m*), using the set of sampled {*η*(*ℓ*)(*p, n, m*)}*r* ;

*ℓ*=1

minimum, mode, mean, maximum, range, and deviation of *L*(*p, n, m*), using the set of sampled {*λ*(*ℓ*)(*p, n, m*)}*r* ; minimum, mode, mean, maximum, range, and deviation of *Q*(*p, n, m*), using the set of sampled {*q*(*ℓ*)(*p, n, m*)}*r* ;

*ℓ*=1

*ℓ*=1

Figure 10. Main software of the MINP for taking sampling

# NUMERICAL EXPERIMENTS

In this section, we shall summarize from a set of three numerical examples, which are described in Appendix

[A](#_bookmark55). The performance of the MINP method, and whose analysis will then be discussed latter for illustrating the eventual usefulness of the MINP method.

One of main noteworthy features of each one of following problems is the existence of 2*n*+*m* local minima within its correspondent feasible region, and only one of them is a global minimum, whereby could result a challenge, because these problems are relatively difficult to identify their respective global minimum.

The experiments were conducted for: a number of

random trial points per subregion *Nσj* (*k*) = 6; number of random trial points per surrounding region *NσM* +1(*k*) = 96; and an expected maximum depth vector *ε* = (*ǫ, . . . , ǫ*; *ǫ*¯*, . . . , ǫ*¯)*t*, where *ǫ* = 0*.*1 and ¯*ǫ* = 0.

## GOLDSTEIN-PRICE PROBLEM

In this first example, we have taken into account an extension of Goldstein-Price function to the mixed inte-

ger Euclidean field R2 × Z2, which has explicitly been

defined in Appendix [A](#_bookmark55). For this numerical experiment, we run *r* = 100 independent replications of the MINP

method, using the software of Figure [10](#_bookmark45).

Table I. Summary of the MINP method, in which just 1 sample reached 53 cumulative iterations

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *f* (*𝑥*) | | *N* (1*,* 2*,* 2) | *L*(1*,* 2*,* 2) | *Q*(1*,* 2*,* 2) |
| Mean | 108.65 | 2566.08 | 6.103 | 1.03e-2 |
| SSD | 107.49 | 1597.31 | 6.744 | 3.81e-2 |
| Min | 6.00 | 1056 | 0.001 | 9.29e-6 |
| Q1 | 27.45 | 1056 | 1.058 | 4.44e-5 |
| Q2 | 91.17 | 2064 | 3.185 | 1.26e-4 |
| Q3 | 166.06 | 3528 | 9.107 | 5.34e-4 |
| Max | 696.09 | 8736 | 24.41 | 2.67e-1 |

#### EBERT BREA

variable distance to the true point (DTP) or global solution point as a function of *d*. As can be seen from the figure, approximately a 25 % of the replications of the MINP method yielded good enough results, i.e., results

less than 1.

As is shown in Table [I](#_bookmark46), in a few replications the MINP method achieved to identify the global solution of the

problem. In fact, according to the reported summary from the table, the 25 % of the replications reach to iden- tify solutions to a distance less than 1.058. Moreover, as can be seen in Figure [11](#_bookmark48), the algorithm globally solved the problem as much as 16 % of the samplings. Besides, from Table [I](#_bookmark46) it may be concluded that the MINP method required, in average 2566 objective function evaluations for solving the problem, what could be considered as

*FN*()

1*.*0

0*.*9

0*.*8

0*.*7

0*.*6

0*.*5

0*.*4

0*.*3

0*.*2

0*.*1

0

2000 4000 6000 8000 

a good enough algorithmic method. Nevertheless, its quality performance measurement resulted to be very

low, yielding a maximum value equal to 2*.*67 × 10−1.

*FD*(*d*)

1*.*0

0*.*9

0*.*8

0*.*7

0*.*6

0*.*5

0*.*4

0*.*3

0*.*2

0*.*1

0

0 5 10 15 20 25 *d*

Figure 11. Empirical CDF of distance to the true point (DTP) for the Goldstein-Price problem.

As shown in Figure [11](#_bookmark48), it is depicted the empirical cumulative distribution function (CDF) of the random

Figure 12. Empirical CDF of number of function evaluations (NE) for the Goldstein-Price problem.

Apart from that, Figure [12](#_bookmark47) shows the empirical CDF of the number of function evaluation (NE) random variable for the Goldstein-Price problem, which allows us to see the performance of the algorithm from the viewpoint of the NE.

## W PROBLEM

Our second problem is a minimization problem of a ob- jective function, which has been proposed by the author as a challenge in the mixed integer programming, and whose mathematical expression is given in Appendix [A](#_bookmark55).

Table [II](#_bookmark49) presents our main performance measurements for the W problem, namely: *N* (2*,* 2*,* 2), *L*(2*,* 2*,* 2) and *Q*(2*,* 2*,* 2). As is shown in the table, the MINP method spent in average 5512*.*32 function evaluations for stop- ping the iterative process, and the 25 % of replications

achieved solutions to a distance to global minimum less than 20.84, in fact Figure [14](#_bookmark50) illustrates the value range of the NE. The table also shows the low values of quality performance *Q*(2*,* 2*,* 2), that has been reached by the algorithm during the computational experimentation for this problem.

#### EBERT BREA

Table II. Summary of the MINP, in which just 1 sample reached 163 cumulative iterations

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *f* (*𝑥*) | *N* (2*,* 2*,* 2) | *L*(2*,* 2*,* 2) | *Q*(2*,* 2*,* 2) |
| Mean | -95.30 | 5512.32 | 25.18 | 1.69e-5 |
| SSD | 524.23 | 4191.62 | 9.22 | 2.99e-5 |
| Min | -453.59 | 2016 | 1.45 | 1.25e-6 |
| Q1 | -306.94 | 2808 | 20.84 | 5.49e-6 |
| Q2 | -215.29 | 4272 | 25.05 | 9.25e-6 |
| Q3 | -108.82 | 6672 | 32.23 | 1.45e-5 |
| Max | 4265.34 | 28032 | 44.54 | 1.99e-4 |

From Figure [13](#_bookmark51) it may be inferred that less than 10 % of the reported replications identified solutions to a distance

*FN*()

1*.*0

0*.*9

0*.*8

0*.*7

0*.*6

0*.*5

0*.*4

0*.*3

0*.*2

0*.*1

0

0 5000 10000 15000 20000 25000 

to the true point less than 10.

Figure 14. Empirical CDF of the NE for the W function problem

*FD*(*d*)

1*.*0

0*.*9

0*.*8

0*.*7

0*.*6

0*.*5

0*.*4

0*.*3

0*.*2

0*.*1

0

0

10 20 30 40 *d*

## ICEBERG PROBLEM

The third problem of optimization is based on an unpub- lished objective function, whose mathematical expres- sion is given in Appendix [A](#_bookmark55).

Table III. Summary of the MINP, in which just 1 sample reached 101 cumulative iterations

*f* (*𝑥*) *N* (3*,* 2*,* 2) *L*(3*,* 2*,* 2) *Q*(3*,* 2*,* 2)

Figure 13. Empirical CDF of the DTP for the W function problem

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mean | -2649.00 | 4441.92 | 7.448 | 2.41e-4 |
| SSD | 583.91 | 3256.11 | 2.987 | 7.41e-4 |
| Min | -3773.35 | 1440 | 0.005 | 6.07e-6 |
| Q1 | -3041.65 | 1920 | 6.135 | 2.32e-5 |
| Q2 | -2598.08 | 3504 | 8.368 | 4.07e-5 |
| Q3 | -2188.85 | 5736 | 9.472 | 7.99e-5 |
| Max | -1326.02 | 16608 | 13.03 | 5.02e-3 |

Figure [14](#_bookmark50) shows cumulative distribution function of the NE when the MINP method is used for globally solving the W function problem. Note that the MINP at least required, in this case, about 28,000 objective function evaluations.

Table [III](#_bookmark52) shows a statistical summary of the random variables: *N* (3*,* 2*,* 2), *L*(3*,* 2*,* 2) and *Q*(3*,* 2*,* 2). As can be seen from the table, the MINP method reported good enough solutions, because the 25 % of replications identified solutions to the true point less than 6.135, and that can be verified from Figure [15](#_bookmark53), which illustrates the empirical CDF of the DTP for our problem.

*FD*(*d*)

1*.*0

0*.*9

0*.*8

0*.*7

0*.*6

0*.*5

0*.*4

0*.*3

0*.*2

0*.*1

0

0

5 10

#### EBERT BREA

number of function evaluations for globally solving the Iceberg problem.

It is worthwhile pointing out that the MINP method was tested without having been tuned for this group of prob- lems. However, some setting of its parameters were empirically fitted for improvement the performance of the algorithm, before running the numerical experiments.

1. DISCUSSION AND FUTURE RESEARCH The aim of this article has been to propose a new approach for globally solving bound constrained mixed integer nonlinear problems using, for reaching this tar- get, the principles of the NP method viewpoint, namely:
   1. partitioning into subregions of the current promising region; ii) sampling scheme for obtaining random trial

*d* points from both each subregions and surrounding re- gion to the current promising region; iii) locating of where

Figure 15. Empirical CDF of the DTP for the Iceberg

problem

Figure [15](#_bookmark53) shows the empirical CDF of the DTP for the Iceberg problem. As can be seen from the figure, the MINP required less than 5000 objective function evaluations in the 70 % of replications for identifying at least a solution, which can be either a local or global solution.

*FN*()

1*.*0

0*.*9

0*.*8

0*.*7

0*.*6

0*.*5

0*.*4

0*.*3

0*.*2

0*.*1

0

0 5000 10000 15000 

Figure 16. Empirical CDF of the NE for the Iceberg problem

Finally, Figure [16](#_bookmark54) allows us to infer the cost in function evaluations of the objective function, because as can be seen from the figure, the MINP method needed a high

has came from the best sampled trial point among all sampled trial points; and iv) testing of a stopping rule for making decision either executing a new iteration or finishing the iterative process of solving of the minimiza- tion problem. Nevertheless, heretofore this approach does not seem to have been effective enough, despite the theoretical foundations that have been developed in this research to reach our goal, if it is taken into account the results reported by Brea **[**[**25**](#_bookmark84)**]**, who carried out a comparative study among two implementations of Genetic Algorithms and the Game of Patterns in the n-dimensional real field.

Although, the MINP method has shown to be a powerful viewpoint for identifying promising regions, what would become a useful algorithmic procedure, and it could hence be hybridized with some local search algorithm, e.g., randomized pattern search algorithm **[**[**10**](#_bookmark69)**]**; pattern search algorithm **[**[**26**](#_bookmark85)**]**, because, the MINP method has experimentally shown to be effective enough for identi- fying promising regions, and hence with information of the promising region could be globally solved Problem [1](#_bookmark0). Besides, the approach that has been used in the MINP method could be easily parallelizable for encoding it in a parallel computer, what would be effective enough for finding global solutions to very large dimension mixed integer optimization problems.

This research has also raised several issues during the

development of the MINP method. Among them, one can remark: i) the MINP method parameter tuning for improving it, for this target, one could hence use the viewpoint of Adenso-Díaz and Laguna **[**[**27**](#_bookmark86)**]**; ii) statisti- cal analysis for the quality performance measurement applying non-standard parametric statistic, e.g., Cheng approach **[**[**23**](#_bookmark82)**]**; iii) the optimum quantity of random trial

#### EBERT BREA

points from both each subregion and surrounding region as a function of the depth measurement, what could

be solved by the approach of sampling budget, pro-

and

2Σ*m*−1

*y*(2*i*+1)

#### 2

*y*(2*i*+2)

*y*(2*i*+1)

posed by Chen and coworkers **[**[**28**](#_bookmark87)**]**, and including others approaches for improving the MINP method stopping

*f* (*y*) =

#### 

1 +

*i*=0

#### 

+ + 1

10 10

· 19 − 14

10

rule using, e.g., Berkhout viewpoint **[**[**29**](#_bookmark88)**]**, who, applying

*y*(2*i*+1)

+3

2

− 14

*y*(2*i*+2)

*y*(2*i*+1) *y*(2*i*+2)

+ 6

the results of Chen, et al., **[**[**28**](#_bookmark87)**]**, presents a new and

interesting approach for accelerating the NP method

10 10

#### !!

2

(2*i*+2)

10

(2*i*+1)

10

(2*i*+2) 2

stopping rule.

+3 *y*

10

· 30 + 2 *y*

10

− 3 *y*

10

On the other hand, we believe that the incorporation of criteria based on artificial intelligence (AI) for making

· 18 − 32

*y*(2*i*+1)

+ 12

10

*y*(2*i*+1)

10

2

+ 48

*y*(2*i*+2)

10

decisions on sampling quantity for taking from each

subregion and its surrounding region, it could be a large

*y*(2*i*+1) *y*(2*i*+2)

*y*(2*i*+2)

2!!

advance in the mixed integer programming.

Finally, as a future work we also propose a compar-

−36

10

+ 27 *.*

10 10

(39b)

ative study between the MINP method and the Game of Patterns algorithmic method, when they are applied for globally solving bound constrained mixed integer optimization problems.

# A LIST OF PROBLEMS

We here present the objective functions of the test problems used in our numerical experiments taking into account the formulation of Problem [1](#_bookmark0). Besides, both lower and upper bound vectors have also been specified, and their respective global solutions.

## EXTENDED GOLDSTEIN-PRICE PROBLEM

BOUND CONSTRAINTS. Let *l, u* ∈ R2*n* × Z2*m* be the bound constraints given by

*l* = (−2*.*5*,...,* −2*.*5; −25*,...,* −25)*t*; (40a)

*u* = (2*.*0*,...,* 2*.*0; 20*,...,* 20)*t.* (40b)

OPTIMUM SOLUTION. The unique global minimum is located at

*zt* = (0*,* −1*,...,* 0*,* −1; 0*,* −10*,...,* 0*,* −10)*,* (41)

#### ` ˛¸ x ` ˛¸ x

2*n* 2*m*

and

*f* (*z*ˆ) = 3(*n* + *m*) (42)

## W PROBLEM

OBJECTIVE FUNCTION. Let *f* (*z*) : R*n* × Z*m* → R be the W function, so called by the author, which is given by

2*n* 2*m*

Σ*n*  *x*(*i*) 4

2 Σ*m*  *y*(*i*) 4 2

OBJECTIVE FUNCTION. Let *f* (*z*) : R × Z → R be

*f* (*𝑥*)

− *x*(*i*) − 2 +

− *y*(*i*) − 2

the Extended Goldstein-Price function, so called by the author, which is given by *f* (*z*) = *f* (*x*) + *f* (*y*), where

4

*i*=1

4

*i*=1

(43)

*f* (*x*) =

2Σ*n*−1

1 + (*x*(2*i*+1) + *x*(2*i*+2) + 1)2 · (19 − 14*x*(2*i*+1)

BOUND CONSTRAINTS. Let *l, u* ∈ R*n* × Z*m* be the bound constraints given by

+3*x*

*i*=0

2

(2*i*+1)

− 14*x*

(2*i*+2)

+ 6 *x*

(2*i*+1)

*x*(2*i*+2)

+ 3 *x*

2

(2*i*+2) )

*l* = (−100*,...,* −100)*t*; (44a)

#### 2

· 30 + (2 *x*(2*i*+1) − 3 *x*(2*i*+2))2 18 − 32 *x*(2*i*+1) + 12 *x*(2*i*+1)

2

+48 *x*(2*i*+2) − 36 *x*(2*i*+1)*x*(2*i*+2) + 27 *x*(2*i*+2) *,*

*u* = (100*,...,* 100)*t.* (44b)

(39a)

SOLUTION. The unique global minimum is located at

*z*ˆ = (*x*ˆ*t*; *y*ˆ*t*)*t*, where

Σ*n* ( 4

Σ*m*  4

*x*ˆ*t*

## = (−12*.*2054969669241*,...,* −12*.*2054969669241);

#### ` ˛¸ x

*n*

*f* (*z*) =

*i*=1

*x i*)

− *α* sin(*x*(*i*))

+

*i*=1

*y*(*i*)

− *β* sin(*y*(*i*)) *,*

## (47)

(45a)

*y*ˆ*t* = (−12*,...,* −12)*,* (45b)

#### ` ˛¸ x

*m*

and

*f* (*z*ˆ) = −115*.*1035690056 *n* − 115 *m.* (46)

## ICEBERG PROBLEM

where *α* = *β* = 1000.

BOUND CONSTRAINTS. Let *l, u* ∈ R*n* ×Z*m* be the bound constraints given by

*l* = (−10*,...,* −10)*t*; (48a)

*u* = (10*,...,* 10)*t.* (48b)

SOLUTION. The unique global minimum is located at

*z*ˆ*t* = (1*.*55573432449541; *. . .*; 1*.*55573432449541; 2*,...,* 2),

#### ` ˛¸ x ` ˛¸ x

OBJECTIVE FUNCTION. Let *f* (*z*) : R*n* × Z*m* → R be the *n m*

Iceberg function, so called by the author, which is given by

and *f* (*z*ˆ) = −994*.*028673136238 *n* − 893*.*297426825682 *m*.

# B PSEUDOCODE OF PROCEDURES

Procedure Surrounding(*n, m,* Θ*, σ*)

Given:

The number of real components, *n*; The number of integer components, *m*;

The bounded feasible region Θ, namely:

*l*(*ℓ*) ≤ *x*(*ℓ*) ≤ *u*(*ℓ*)*,* ∀*ℓ* ∈ {1*, . . . , n*}*,*

*l*¯(*ℓ*) ≤ *y*(*ℓ*) ≤ *u*¯(*ℓ*)*,* ∀*ℓ* ∈ {1*, . . . , m*};

A promising region *σ* ⊆ Θ, defined by

*l*(*ℓ*) ≤ *x*(*ℓ*) ≤ *u*(*ℓ*)*,* ∀*ℓ* ∈ {1*, . . . , n*}*,*

*σ σ*

*l*¯(*ℓ*) ≤ *y*(*ℓ*) ≤ *u*¯(*ℓ*)*,* ∀*ℓ* ∈ {1*, . . . , m*};

*σ σ*

Declare:

A matrix *X*˘ = [*x*˘(*ij*)]4*,n*

*i*=1*,j*=1

A matrix *Y*˘ = [*y*˘(*ij*)]4*,m*

*i*=1*,j*=1

∈ R4×*n*for saving real boundary components of the set *S*(*σ*(*k*));

∈ Z4×*m*for saving integer boundary components of the set *S*(*σ*(*k*));

for *ℓ* ← 1 to *n* do

*x*ˇ(1*,ℓ*) ← *l*(*ℓ*);

*x*˘(2*,ℓ*) ← *l*(*ℓ*);

*σ*

*x*˘(3*,ℓ*) ← *u*(*ℓ*);

*σ*

*x*˘(4*,ℓ*) ← *u*(*ℓ*);

for *ℓ* ← *n* + 1 to *n* + *m* do

*y*ˇ(1*,ℓ*−*n*) ← ¯*l*(*ℓ*);

*y*ˇ(2*,ℓ*−*n*) ← ¯*l*(*ℓ*);

*σ*

*y*˘(3*,ℓ*−*n*) ← *u*¯(*ℓ*);

*σ*

*y*˘(4*,ℓ*−*n*) ← *u*¯(*ℓ*);

Figure 17. Surrounding procedure

Procedure. Partitioning(*n, m,* Θ*, σ, d*(*k*))

Given:

the number of real components, *n*;

the number of integer components, *m*; the bounded feasible region Θ, namely:

*l*(*i*) ≤ *x*(*i*) ≤ *u*(*i*)*,* ∀*i* ∈ {1*, . . . , n*}*,*

*l*¯(*i*) ≤ *y*(*i*) ≤ *u*¯(*i*)*,* ∀*i* ∈ {1*, . . . , m*}; the promising region *σ* ⊆ Θ to be partitioned, which is defined by

*l*(*i*) ≤ *x*(*i*) ≤ *u*(*i*)*,* ∀*i* ∈ {1*, . . . , n*}*,*

*σ σ*

*l*¯(*i*) ≤ *y*(*i*) ≤ *u*¯(*i*)*,* ∀*i* ∈ {1*, . . . , m*};

*σ σ*

the mixed integer depth vector of the current promising region *σ*(*k*) to be partitioned

*dt*(*k*) = (*d*(1)(*k*)*, . . . , d*(*n*)(*k*); *d*¯(*n*+1)(*k*)*, . . . , d*¯(*n*+*m*)(*k*));

` ˛¸ x ` ˛¸ x

*n m*

Let *Mσ* = 2*n*+*m* be the number of subregions to be denoted by {*σj* }*Mσ* ;

*j*=1

Declare:

a matrix *X* = [*x*(*ij*)]2*Mσ,n*

*i*=1*,j*=1

∈ R2*Mσ* ×*n* for saving real boundary components of the set {*σj* }*Mσ* ;

a matrix *Y* = [*y*(*ij*)]2*Mσ,m* ∈ Z2*Mσ* ×*m* for saving integer boundary components of the set {*σj* }*Mσ* ;

*j*=1

*i*=1*,j*=1

Let *q* = 1 be the counter of subregion *σj* ;

Initialization

Let *i* = 1;

for *q* ← 0 to *Mσ* − 1 do

*j*=1

(*b*(*n*+*m*−1)*, b*(*n*+*m*−2)*, . . . , b*(0))2 ← *c*(*q*), where *c*(*q*) is a convertor function, which transforms decimal numbers to binary numbers, namely, (*b*(*n*+*m*−1)*, b*(*n*+*m*−2)*, . . . , b*(0))2 ∈ {0*,* 1}*n*+*m*;

for *ℓ* ← 1 to *n* do

if *u*(*ℓ*) − *l*(*ℓ*) *> ε*(*ℓ*) then

*σ σ*

*δ*(*ℓ*) = (*l*(*ℓ*) + *u*(*ℓ*))*/*2;

*σ σ*

if *b*(*ℓ*−1) = 0 then

*x*(*i,ℓ*) ← *l*(*ℓ*);

*σ*

*x*(*i*+1*,ℓ*) ← *δ*(*ℓ*);

else

*x*(*i,ℓ*) ← *δ*(*ℓ*);

*x*(*i*+1*,ℓ*) ← *u*(*ℓ*);

*σ*

for *ℓ* ← *n* + 1 to *n* + *m* do if *u*¯(*ℓ*) − ¯*l*(*ℓ*) *> ε*¯(*ℓ*) then

*σ σ*

*δ*(*ℓ*) = (¯*l*(*ℓ*) + *u*¯(*ℓ*))*/*2;

*σ σ*

if (*b*(*ℓ*−1) = 0) ∧ (⌊*δ*(*ℓ*)⌋ = ⌈*δ*(*ℓ*)⌉) then

*y*(*i,ℓ*−*n*) ← ¯*l*(*ℓ*);

*σ*

*y*(*i*+1*,ℓ*−*n*) ← *δ*(*ℓ*);

if (*b*(*ℓ*−1) = 1) ∧ (⌊*δ*(*ℓ*)⌋ = ⌈*δ*(*ℓ*)⌉) then

*y*(*i,ℓ*−*n*) ← *δ*(*ℓ*) + 1;

*y*(*i*+1*,ℓ*−*n*) ← *u*¯(*ℓ*);

*σ*

if (*b*(*ℓ*−1) = 0) ∧ (⌊*δ*(*ℓ*)⌋

*y*(*i,ℓ*−*n*) ← ¯*l*(*ℓ*);

*σ*

*y*(*i*+1*,ℓ*−*n*) ← ⌊*δ*(*ℓ*)⌋;

if (*b*(*ℓ*−1) = 1) ∧ (⌊*δ*(*ℓ*)⌋

*y*(*i,ℓ*−*n*) ← ⌈*δ*(*ℓ*)⌉;

*y*(*i*+1*,ℓ*−*n*) ← *u*¯(*ℓ*);

*σ*

⌈*δ*(*ℓ*)⌉) then

⌈*δ*(*ℓ*)⌉) then

Let *i* ← *i* + 2

Figure 18. Partitioning procedure

Preamble of Sampling and the Measuring of the Objective Function

Given:

The number of real components, *n*; The number of integer components, *m*;

Let *Mσ* ← 2*n*+*m*;

Given:

the number of random sample for being taken from each *j*th subregion *σj* (*k*), *Nj* ;

the number of random sample for being taken from the surrounding region *S*(*σ*(*k*)) = *σMσ* +1(*k*), *N* ;

the real boundaries of each *j*th subregion *σj* (*k*) for each *k*th iteration, which must be read from the real matrix *X* = [*x*(*jℓ*)]2*Mσ,n* ;

*j*=1*,ℓ*=1

the integer boundaries of each *j*th subregion *σj* (*k*) for each *k*th iteration, which must be read from the integer matrix *Y* = [*y*(*jℓ*)]2*Mσ,m* ;

*j*=1*,ℓ*=1

the real boundaries of the surrounding region *S*(*σ*(*k*)) for each *k*th iteration, which must be read from

the real matrix *X*˘ = [*x*˘(*ij*)]4*,n*

*i*=1*,j*=1

∈ R4×*n*;

the integer boundaries of the surrounding region *S*(*σ*(*k*)) for each *k*th iteration, which must be read

from the integer matrix *Y*˘ = [*y*˘(*ij*)]4*,m*

*i*=1*,j*=1

∈ Z4×*m*;

a convertor function *c*(*q*), which converts any *q* ∈ N number, which is given by its decimal representation, to its binary representation, namely, *q* = (*b*(⌈lg2(*q*)⌉−1)*, . . . , b*(0))2 ∈ {0*,* 1}⌈lg2 (*q*)⌉;

Declare:

the best current point *z*ˆ*t* = (*x*ˆ(1)*, . . . , x*ˆ(*n*); *y*ˆ(*n*+1)*, . . . , y*ˆ(*n*+*m*)) ∈ R*n* × Z*m*; the index of the best performance of the objective function, given by

*I*ˆ(*σj* (*k*)) = min

*s*∈{1*,...,Nj* }

*f* (*zs,j* )*,*

where *zs,j* denotes the *s*th mixed integer sample point, which has been taken from the subregion *σj* (*k*);

Choose:

an *n*th index seed, namely, *n* ∈ N , which depends on the *s*th random seed I(*s*);

an *Ns* ∈ N+ number of sampling per sector of surrounding region *σ*(*k*) at the *k*th iteration;

if *k* = 0 then Assign the current objective function value *f*ˆ(*z*ˆ), the largest possible value that can be represented in an x-bit computer;

Figure 19. Preamble of the sampling procedure

Procedure Sampling of Surrounding and Promising Region and the Measuring of the Objective Function (Part i)

Let ˆ*i* ← 0;

if *k* ≥ 1 then

for *ℓ* ← 0 to 2*n* − 2 do

for *ℓ*¯ ← 0 to 2*m* − 2 do

(*b*(*n*−1)*, . . . , b*(0))2 ← *c*(*ℓ*);

(¯*b*(*m*−1)*, . . . ,* ¯*b*(0))2 ← *c*(*ℓ*¯);

for *r* ← 1 to *Ns* do for *j* ← 1 to *n* do

if *b*(*j*−1) = 0 then

*x*(*j*) ← u(*x*ˇ(2*,j*) *, x*ˇ(3*,j*)*, s*)

else

*x*(*j*) ← w(*x*ˇ(1*,j*)*, x*ˇ(2*,j*) *, x*ˇ(3*,j*)*, x*ˇ(4*,j*)*, s*)

for *j* ← 1 to *m* do

if ¯*b*(*j*−1) = 0 then

*y*(*j*) ← u¯(*y*ˇ(2*,j*)*, y*ˇ(3*,j*)*, s*)

else

*y*(*j*) ← w(*y*ˇ(1*,j*)*, y*ˇ(2*,j*)*, y*ˇ(3*,j*)*, y*ˇ(4*,j*)*, s*)

Let *zt* ← (*x*(1)*, . . . , x*(*n*); *y*(*n*+1)*, . . . , y*(*n*+*m*))

*s* ` ˛¸ x ` ˛¸ x

*n m*

Measure the objective function by *f* (*z*)|*zs*

if *f* (*zs*) *< f*ˆ(*z*ˆ) then Let *f*ˆ(*z*ˆ) ← *f* (*zs*); Let *z*ˆ ← *zs*;

Let ˆ*i* ← *Mσ* + 1;

for *d* ← 1 to 2 do switch *d* do

case *d=1* do

Let *ℓ* = 2*n* − 1;

Let *ℓ*¯ = 2*m* − 2;

case *d=2* do

Let *ℓ* = 2*n* − 2;

Let *ℓ*¯ = 2*m* − 1;

(*b*(*n*−1)*, . . . , b*(0))2 ← *c*(*ℓ*);

(¯*b*(*m*−1)*, . . . ,* ¯*b*(0))2 ← *c*(*ℓ*¯);

for *r* ← 1 to *Ns* do for *j* ← 1 to *n* do

if *b*(*j*−1) = 0 then

*x*(*j*) ← u(*x*ˇ(2*,j*)*, x*ˇ(3*,j*)*, s*)

else

*x*(*j*) ← w(*x*ˇ(1*,j*)*, x*ˇ(2*,j*)*, x*ˇ(3*,j*)*, x*ˇ(4*,j*)*, s*)

for *j* ← 1 to *m* do

if ¯*b*(*j*−1) = 0 then

*y*(*j*) ← u¯(*y*ˇ(2*,j*)*, y*ˇ(3*,j*)*, s*)

else

*y*(*j*) ← w(*y*ˇ(1*,j*)*, y*ˇ(2*,j*)*, y*ˇ(3*,j*)*, y*ˇ(4*,j*)*, s*)

Let *zt* ← (*x*(1)*, . . . , x*(*n*); *y*(*n*+1)*, . . . , y*(*n*+*m*))

*s* ` ˛¸ x ` ˛¸ x

*n m*

Measure the objective function by *f* (*z*)|*zs*

if *f* (*zs*) *< f*ˆ(*z*ˆ) then Let *f*ˆ(*z*ˆ) ← *f* (*zs*); Let *z*ˆ ← *zs*;

Let ˆ*i* ← *Mσ* + 1;

Figure 20. Sampling procedure, part i

Procedure Sampling of Surrounding and Promising Region and the Measuring of the Objective Function (Part ii)

for *i* ← 0 to *Mσ* − 1 do for *r* ← 1 to *Ns* do

for *j* ← 1 to *n* do

if *d*(*j*) ≤ *ε*(*j*) then

*x*(*j*) ← *x*ˆ(*j*)

else

*x*(*j*) ← u(*x*(2*i*+1*,j*) *, x*(2*i*+2*,j*) *, s*)

for *j* ← 1 to *m* do

if *d*¯(*j*) ≤ *ε*¯(*j*) then

*y*(*j*) ← *y*ˆ(*j*)

else

*y*(*j*) ← u¯(*y*(2*i*+1*,j*)*, y*(2*i*+2*,j*)*, s*)

Let *zt* ← (*x*(1)*, . . . , x*(*n*); *y*(*n*+1)*, . . . , y*(*n*+*m*))

*s* ` ˛¸ x ` ˛¸ x

*n m*

Measure the objective function by *f* (*z*)|*zs*

if *f* (*zs*) *< f*ˆ(*z*ˆ) then Let *f*ˆ(*z*ˆ) ← *f* (*zs*); Let *z*ˆ ← *zs*;

Let ˆ*i* ← *i*;

Let *i* ← *i* + 1;

switch ˆ*i* do

case ˆ*i* = 0 do

Backtrack to entire feasible region Θ;

Update the depth vector *d* by using *d*(0) = D(0*, n, m, σ*(*k* + 1)*,* Θ);

Let *k* ← 0;

other wise do

Let *σ*(*k* + 1) = *σ*ˆ*i*(*k*);

Update by using *d*(*k* + 1) = D(*k* + 1*, n, m, σ*(*k* + 1)*,* Θ);

Let *k* ← *k* + 1;

Figure 21. Sampling procedure, part ii

# C PSEUDOCODE OF FUNCTIONS

Function Real Double Uniform(*a*1*, b*1*, a*2*, b*2*, n*)

Given:

*a*1 ∈ R: a lower real bound of the first interval uniform random distribution;

*b*1 ∈ R: an upper real bound of the first interval uniform random distribution;

*a*2 ∈ R: a lower real bound of the second interval uniform random distribution; *b*2 ∈ R: an upper real bound of the second interval uniform random distribution; *n* ∈ N : a *n*th seed from an available pseudorandom number generator set;

u(*n*) ∈ (0*,* 1): a uniformly distributed random number between 0 and 1 from the *n*th

index seed, namely, {u(*s*)}I(*S*)

*s*=I(1)

, which depends on the *s*th random seed I(*s*);

Output: *a uniformly distributed random real number belonging to two disjunct intervals, namely:* (*a*1*, b*1) *or* (*a*2*, b*2) *;*

Function w(*a*1*, b*1*, a*2*, b*2*, n*):

Calculate *θ* ∈ R and *x* ∈ R, namely:

*b*1 − *a*1

*θ* = ;

*b*1 + *b*2 − *a*1 − *a*2

*x* = *a*1 + (*b*1 + *b*2 − *a*1 − *a*2) u(*n*);

if *θ* ≤ u(*n*) *<* 1 then

Let *x* ← *x* + *a*2 − *b*1;

return *x*;

Figure 22. Double real uniform distribution function

#### EBERT BREA

Function Integer Double Uniform(*a*¯1*,* ¯*b*1*, a*¯2*,* ¯*b*2*, n*)

Given:

*a*¯1 ∈ Z: a lower integer bound of the first interval uniform random distribution;

¯*b*1 ∈ Z: an upper integer bound of the first interval uniform random distribution;

*a*¯2 ∈ Z: a lower integer bound of the second interval uniform random distribution;

¯*b*2 ∈ Z: an upper integer bound of the second interval uniform random distribution;

*n* ∈ N : a *n*th seed from an available pseudorandom number generator set;

u(*n*) ∈ (0*,* 1): a uniformly distributed random number between 0 and 1 from the *n*th

index seed, namely, {u(*s*)}I(*S*)

*s*=I(1)

, which depends on the *s*th random seed I(*s*);

Output: *a uniformly distributed random integer number belonging to two disjunct intervals, namely:* [*a*¯1*,* ¯*b*1] *or* [*a*¯2*,* ¯*b*2] *;*

Function w(*a*¯1*,* ¯*b*1*, a*¯2*,* ¯*b*2*, n*):

Calculate: *θ* ∈ R and *x* ∈ R, namely:

1 + ¯*b*1 − *a*¯1

*θ* = 2 + ¯*b* + ¯*b* − *a*¯ − *a*¯ ;

1

2 1 2

*x* = *a*¯1 + (2 + ¯*b*1 + ¯*b*2 − *a*¯1 − *a*¯2) u(*n*);

if *θ* ≤ *u*(*n*) *<* 1 then

Let *x* ← *x* + *a*¯2 − ¯*b*1 − 1;

return ⌊*x*⌋;

Figure 23. Double integer uniform distribution function

Function depth D(*k, n, m, σ*(*k*)*,* Θ)

Given:

The iteration counter, *k*;

The number of real components, *n*; The number of integer components, *m*;

The *k*th promising region *σ*(*k*) ⊆ Θ, which is then defined by

*l*(*i*) ≤ *x*(*i*) ≤ *u*(*i*)*,* ∀*i* ∈ {1*, . . . , n*}*,*

*σ σ*

¯*l*(*i*) ≤ *y*(*i*) ≤ *u*¯(*i*)*,* ∀*i* ∈ {1*, . . . , m*};

*σ σ*

the bounded feasible region Θ, namely:

*l*(*i*) ≤ *x*(*i*) ≤ *u*(*i*)*,* ∀*i* ∈ {1*, . . . , n*}*,*

¯*l*(*i*) ≤ *y*(*i*) ≤ *u*¯(*i*)*,* ∀*i* ∈ {1*, . . . , m*};

Declare:

*d* = (*d*(1)*, . . . , d*(*n*); *d*¯(*n*+1)*, . . . , d*¯(*n*+*m*))*t* ∈ R*n* × Z*m*;

` ˛¸ x ` ˛¸ x

*n m*

Output: *an updated depth vector d;*

Function D(*k, n, m, σ*(*k*)*,* Θ): switch *k* do

case *k* = 0 do

for *j* ← 1 to *n* do

*d*(*j*) ← *u*(*j*) − *l*(*j*);

for *j* ← *n* + 1 to *n* + *m* do

*d*¯(*j*) ← *u*¯(*j*) − ¯*l*(*j*);

other wise do

for *j* ← 1 to *n* do

*d*(*j*) ← *u*(*j*) − *l*(*j*);

*σ σ*

for *j* ← *n* + 1 to *n* + *m* do

*d*¯(*j*) ← *u*¯(*j*) − ¯*l*(*j*);

*σ σ*

return *d*

Figure 24. Depth function

# ACKNOWLEDGMENTS

This work was supported in part by the Centro de Investigación y Desarrollo de Ingeniería(CIDI) of the Universidad Católica Andrés Bello. Special thanks are due to Professor María I. López, Director of the CIDI for her efforts in obtaining the resources for this research. He also thanks Mr. Rolando Andrade for coding the al- gorithm under C++, and the two anonymous reviewers for very helpful comments and suggestions for clarifying the presentation of this article.

#### REFERENCES

1. Floudas, C. A. Nonlinear and mixed-integer optimiza- tion: fundamentals and applications. New York, Oxford University Press, 1995.

#### Grossmann, I. E. and Z. Kravanja,“Mixed-Integer Non- linear Programming: A Survey of Algorithms and Ap- plications”, pp. 73–100. New York, NY: Springer New York, June 1997.

<https://doi.org/10.1007/978-1-4612-1960-6_5>

#### Grossmann, I. E. “Review of Nonlinear Mixed-Integer and Disjunctive Programming Techniques”, Opti- mization and Engineering, vol. 3, no. 3, pp. 227–252, 2002. https://doi.org/10.1023/A:1021039126272

1. Tawarmalani, M. and N. V. Sahinidis, Convexification and global optimization in continuous and mixed- integer nonlinear programming: theory, algorithms, software, and applications. Nonconvex optimization and its applications; v. 65, Dordrecht Boston: Kluwer Academic Publishers, 2002.
2. Tawarmalani, M. and N. V. Sahinidis, “Global op- timization of mixed-integer nonlinear programs: A theoretical and computational study”, Mathematical Programming, vol. 99, no. 3, pp. 563–591, 2004. <https://doi.org/10.1007/s10107-003-0467-6>
3. Brea, E. Extensiones del método de Nelder Mead a problemas de variables enteras y enteras mixtas, tech. rep., Universidad Central de Venezuela, Septiembre 2009. Trabajo de Ascenso para categoría de Profesor Titular.
4. Brea, E. “Una extensión del método de Nelder Mead a problemas de optimización no lineales enteros mix- tos”, Revista Internacional de Métodos Numéricos para Cálculo y Diseño en Ingeniería, vol. 29, pp. 163– 174, July 2013. <https://doi.org/10.1016/j.rimni.2013.06.005>
5. Brea, E. (2016 July) On the Performance of the Mixed Integer Randomized Pattern Search Algorithm, in The 13th International Congress on Numerical Methods in Engineering and Applied Sciences (Y. González,

#### E. Dávila, V. Duarte, M. Candal, O. Pelliccioni,

J. Darías, and M. Cerrolaza, eds.), vol. 1, (Caracas, Venezuela), pp. Op 61–Op 72.

#### EBERT BREA

1. Brea, E. (2017 April). Game of Patterns: An approach for solving mixed integer nonlinear optimization prob- lems, in International Congress on Industrial Engi- neering and Operations Management (IEOM 2017), Bogota, Colombia. <http://ieomsociety.org/bogota2017/proceedings>
2. Brea, E. “On the mixed integer randomized pattern search algorithm”, Revista de la Unión de Matemáti- cos Argentinos, vol. 60, pp. 485-503, September 2019. <https://doi.org/10.33044/revuma.v60n2a14>
3. Kantor, I., J.-L. Robineau, H. Bütün, and F. Maréchal, “A mixed-integer linear programming for- mulation for optimizing multi-scale material and en- ergy integration”, Frontiers in Energy Research, vol. 8, pp. 49.1–49.20, April 2020. <https://doi.org/10.3389/fenrg.2020.00049>
4. Jalota, H. and M. Thakur, “Genetic Algorithm De- signed for Solving Linear or Nonlinear Mixed-Integer Constrained Optimization Problems”, International Proceedings on Advances in Soft Computing, Intelli- gent Systems and Applications, (Singapore), pp. 277– 290, Springer Singapore, December 2018. <https://doi.org/10.1007/978-981-10-5272-9_27>
5. Deep, K., K. P. Singh, M. Kansal, and C. Mohan, “A real coded genetic algorithm for solving integer and mixed integer optimization problems”, Applied Mathematics and Computation, vol. 212, pp. 505– 518, June 2009. <https://doi.org/10.1016/j.amc.2009.02.044>
6. Schlüter, M., J. A. Egea, and J. R. Banga, “Extended ant colony optimization for non-convex mixed integer nonlinear programming”, Computers & Operations Research, vol. 36, pp. 2217–2229, July 2009. <https://doi.org/10.1016/j.cor.2008.08.015>
7. Mohammadi, M., S. N. Musa, and A. Bahreininejad, “Optimization of mixed integer nonlinear economic lot scheduling problem with multiple setups and shelf life using metaheuristic algorithms”, Advances in Engi- neering Software, vol. 78, pp. 41–51, December 2014. <https://doi.org/10.1016/j.advengsoft.2014.08.004>

#### Cardoso, M. F., R. L. Salcedo, S. F. de Azevedo, and

D. Barbosa, “A simulated annealing approach to the solution of MINLP problems”, Computers & Chemical Engineering, vol. 21, pp. 1349–1364, April 1997. <https://doi.org/10.1016/S0098-1354(97)00015-X>

1. Gupta, O. K. and A. Ravindran, “Branch and bound experiments in convex nonlinear integer programming” Management Science, vol. 31, pp. 1533–1546, De- cember 1985. <http://www.jstor.org/stable/2631793>
2. Shi, L. and S. Ólafsson, “Nested Partitions method for global optimization”, Operations Research, vol. 48, pp. 390–407, May 2000. <https://doi.org/10.1287/opre.48.3.390.12436>

#### EBERT BREA

1. Xia, L., Y. Zhao, M. Xie, J. Shao, and J. Dong, “Mixed integer programming based nested partition algorithm for facility location optimization problems”, in 2008 IEEE International Conference on Service Operations and Logistics, and Informatics, vol. 2, pp. 2375–2381, October 2008.

#### Shi, L. and S. Ólafsson, Nested Partitions Method, Theory and Applications. No. 978-0-387-71909-2 in International Series in Operations Research and Management Science, Springer, 1st ed., December 2009.

<https://doi.org/10.1007/978-0-387-71909-2>

1. IBM, ILOG◎ CPLEX◎ Optimization Studio inter- faces, August 2021. <https://www.ibm.com/analytics/>
2. Shi, L. and S. Ólafsson, “Stopping rules for the stochastic Nested Partitions method”, Methodology and Computing in Applied Probability, vol. 2, pp. 37- 58, April 2000. https://doi.org/10.1023/A:1010055101140

#### Cheng, R. Non-standard parametric statistical infer- ence. Oxford, England: Oxford University Press, first ed., 2017.

1. de Haan, L. “Estimation of the Minimum of a Func- tion Using Order Statistics”, Journal of the American Statistical Association, vol. 76, pp. 467–469, June 1981. <http://www.jstor.org/stable/2287851>

#### Brea, E. (2019 January). Game of Patterns and Genetic Algorithms under a comparative study, in Hybrid Metaheuristics, HM2019 (M. B. Aguilera,

C. Blum, H. G. Santos, P. Pinacho-Davidson, and

J. G. del Campo, eds.), vol. 11299, (Concepcion, Chile), pp. 93–107, Springer, Cham. <https://doi.org/10.1007/978-3-030-05983-5_7>

1. Audet, C. and J. E. Dennis Jr., “Pattern Search Algorithms for Mixed Variable Programming”, SIAM Journal on Optimization, vol. 11, pp. 573–594, July 2001.

<https://doi.org/10.1137/S1052623499352024>

1. Adenso-Díaz, B. and M. Laguna, “Fine-Tuning of Algorithms Using Fractional Experimental Designs and Local Search”, Operations Research, vol. 54, pp. 99–114, February 2006. <https://doi.org/10.1287/opre.1050.0243>
2. Chen, C.-H., D. He, M. Fu, and L. H. Lee, “Effi- cient Simulation Budget Allocation for Selecting an Optimal Subset”, INFORMS Journal on Computing, vol. 20, pp. 579–595, May 2008. <https://doi.org/10.1287/ijoc.1080.0268>
3. Berkhout, J. “An accelerated stopping rule for the Nested Partition Hybrid Algorithm for discrete stochastic optimization”, Discrete Event Dynamic Systems, vol. 25, pp. 441–452, September 2015. <https://doi.org/10.1007/s10626-014-0191-9>