A Population And Interval Constraint Propagation Algorithm

Vincent Barichard and Jin-Kao Hao

LERIA - Faculty of Sciences - University of Angers
2, Boulevard Lavoisier, 49045 Angers Cedex 01, France
Vincent.Barichard@info.univ-angers.fr
Jin-Kao.Hao@univ-angers.fr

Abstract. We present *PICPA*, a new algorithm for tackling constrained continuous multi-objective problems. The algorithm combines constraint propagation techniques and evolutionary concepts. Unlike other evolutionary algorithm which gives only heuristic solutions, *PICPA* is able to bound effectively the Pareto optimal front as well as to produce accurate approximate solutions.

1 Introduction

The multi-objective combinatorial optimization problems aim to model real world problems that involve many criteria and constraints. In this context, the optimum solution searched is not a single value but a set of good compromises or "trade-offs" that all satisfy the constraints.

A constrained continuous multi-objective problem (CCMO) can be defined as follows:

$$\text{CCMO}\begin{cases}\min & f_i(\overrightarrow{x}) & i = 1, ..., o\\ \text{s.t.} & C_l(\overrightarrow{x}) \geq 0 & l = 1, ..., m\\ & \overrightarrow{x} \in \mathbb{R}^n & \end{cases}$$

where n is the number of variables, $\overrightarrow{x}$ is a decision vector, o the number of objectives and m the number of constraints of the problem.

Over the past few years, many researchers developed some evolutionary algorithms which tackle multi-objective optimization problems. They demonstrated the advantage in using population-based search methods [11, 9, 17, 10, 20, 19, 3].

Unfortunately, when the constraints become difficult to satisfy, or when the feasible objective space is not connected, multi-objective evolutionary algorithms hardly converge to the whole Pareto optimal front. Furthermore, these algorithms don't give any bounds of the Pareto optimal front.

In this paper, we present *PICPA*, the "Population and Interval Constraint Propagation Algorithm" which is able to produce high quality approximate solutions while giving guaranteed bounds for the Pareto optimal front. These bounds allow us to know if the solutions found are close to or far away from the optimal front. *PICPA* combines "Interval Constraint Propagation" (ICP) techniques

[2, 4] with evolutionary concepts (population and Pareto selection process). Experimental evaluations of *PICPA* on some well known test problems show its effectiveness.

The paper is organized as follows: in the next section we briefly introduce the general principles of the ICP techniques. Then we present our *PICPA* algorithm in Section 3. Experimental results are the subject of the Section 4. Finally, conclusions and perspectives are given in the last section.

2 Interval Constraint Propagation

In this section, we explain briefly the basic idea of Interval Constraint Propagation (ICP). ICP combines interval computation [15] and constraint propagation [13] in order to solve non linear systems of equations and inequations. ICP algorithms have been first presented by Cleary [2] and Davis [4].

2.1 Interval representation of variables and constraints

Each variable of the problem is represented by an interval, and is linked to other variables by constraints. Let us consider two variables x and y, and assume that they belong to some prior feasible value domains:

$$\begin{array}{c} x \in [x^-, x^+] = [-2, 2] \\ y \in [y^-, y^+] = [-8, 9] \end{array}$$

Let us consider now the constraint: $y = x^3$. Because of its definition, we can consider the cubic constraint as a subset of $\mathbb{R}^2$:

$$cubic = \{(x, y) \in \mathbb{R}^2 \mid y = x^3\}$$

The cubic constraint is a binary constraint since it takes into account two variables x and y. In our example, the cubic constraint can be used to *remove some inconsistent values* in the domain of y. Indeed we see that: $\forall x \in [-2, 2], \quad x^3 \leq 8$. Therefore , all the values of the domain of y that are greater than 8 can be safely removed (cf. the hatched area of figure 1).

In a more formal way, the cubic constraint allows us to contract the domain of x and y thanks to the following projection operator:

$$\left\{ \begin{array}{l} [x^-, x^+] \longleftarrow [x^-, x^+] \bigcap (\sqrt[3]{[y^-, y^+]}) \\ [y^-, y^+] \longleftarrow [y^-, y^+] \bigcap ([x^-, x^+]^3) \end{array} \right.$$

$$\begin{array}{ll} \text{where} & \sqrt[3]{[y^-, y^+]} = [\sqrt[3]{y^-}, \sqrt[3]{y^+}] \\ \text{and} & [x^-, x^+]^3 \;\; = [(x^-)^3, (x^+)^3] \end{array}$$

More generally, for each *primitive constraint*, there is a projecting procedure allowing to contract each variable domain. For more information about these projection procedure, the reader is invited to consult a textbook, for example [12].

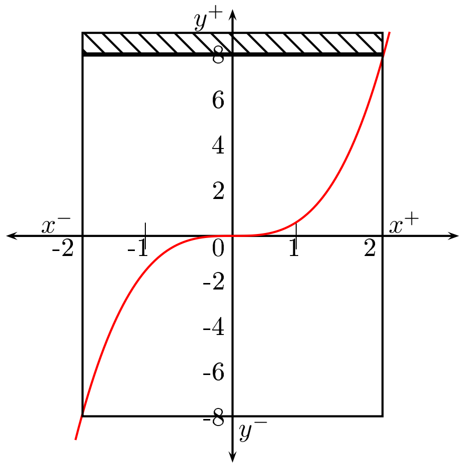


Fig. 1. Sample of the projection of the cubic constraint.

2.2 To reach a fix point

For a given set of constraints, an iterative application of the corresponding projection procedures over the constraints will lead to a state where no variable domain can be further reduced. That is, a fix point is reached. Notice that such a fix point doesn't constitute a solution because the variables are not instanced yet. To get a solution, other techniques have to be jointly applied.

In constraint programming, many algorithms for reaching fix points have been developed. We can mention various local (arc) consistency algorithms for discrete problems [13, 14], and ICP algorithms for continuous problems [1].

By considering a projection procedure as a local consistency procedure, it can be incorporated in a fix point algorithm. Thus efficient contracting algorithms were developed for the continuous constrained problems.

2.3 Discussion

An ICP algorithm is a polynomial time procedure which reduces the domains of the variables of the problem. This kind of algorithm reduces the search space by removing inconsistent values of the variables. It does not delete any solution of the problem. However, such an application of ICP leads only to an approximation of the solution. In order to increase the precision of the approximation, one may bisect any variable domain and apply ICP to the two different alternatives. If we iterate this process, we increase the precision but we get an exponential number of bisections, and so a huge resolution time. In practice, this approach is not usable for problems with a high number of variables.

In constrained multi-objective problems, we have to satisfy constraints but also to minimize (maximize) some objective functions. So, even with a small number of variables and objectives, the problem cannot be processed simply with a classical bisection algorithm.

3 Population and Interval Constraint Propagation Algorithm (*PICPA*)

The concept of population is very suitable in a multi-objectives context. Indeed, as the Pareto optimal front is most of the time a set of solutions, each individual of the population can hopefully become a particular solution of the Pareto optimal front. As a result, the whole population will be an approximation of the targeted Pareto optimal front.

Many population-based multi-objective optimization algorithms have thus been developed. We can mention, among others, *NSGA* [17], *SPEA* [20], *MOSA* [19], *MOTS* [10], *M-PAES* [3].

In this section, we present the *PICPA* algorithm which combines interval constraint propagation with a population.

3.1 A new dominance relation definition

PICPA uses a new dominance relation: the Point with Set dominance (PS-dominance). Let us recall first the classical dominance relation between two points:

Definition 1 (Dominance). *Let us consider a minimization problem. For any two vectors $\overrightarrow{x}$ and $\overrightarrow{y}$:*

- $\overrightarrow{x}$ *equals* $\overrightarrow{y}$, *iff* $\forall i \in \{1, 2, ..., n\} \,:\, x_i = y_i$
- $\overrightarrow{x}$ *dominates* $\overrightarrow{y}$, *iff* $\forall i \in \{1, 2, ..., n\} \,:\, x_i \leq y_i$ *and* $\exists j \in \{1, 2, ..., k\} \,:\, x_j < y_j$
- $\overrightarrow{x}$ *is dominated by* $\overrightarrow{y}$, *iff* $\forall i \in \{1, 2, ..., n\} \,:\, x_i \geq y_i$ *and* $\exists j \in \{1, 2, ..., n\} \,:\, x_j > y_j$
- $\overrightarrow{x}$ *is non dominated by* $\overrightarrow{y}$, *and* $\overrightarrow{y}$ *is non dominated by* $\overrightarrow{x}$ *otherwise*

We can now introduce the Point with Set Dominance definition:

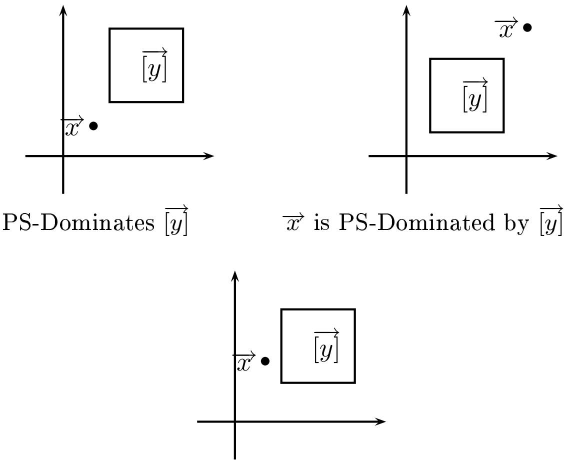
Definition 2 (PS-Dominance). *For any vector* $\overrightarrow{x}$ *and any vector set* $\{\overrightarrow{y}\}$, $|\{\overrightarrow{y}\}| > 0$*:*

- $\overrightarrow{x}$ *PS-equals* $\{\overrightarrow{y}\}$, *iff* $\forall \overrightarrow{y} \in \{\overrightarrow{y}\} \,:\, \overrightarrow{x}$ *equals* $\overrightarrow{y}$
- $\overrightarrow{x}$ *PS-dominates* $\{\overrightarrow{y}\}$, *iff* $\forall \overrightarrow{y} \in \{\overrightarrow{y}\} \,:\, \overrightarrow{x}$ *dominates* $\overrightarrow{y}$
- $\overrightarrow{x}$ *is PS-dominated by* $\{\overrightarrow{y}\}$, *iff* $\forall \overrightarrow{y} \in \{\overrightarrow{y}\} \,:\, \overrightarrow{x}$ *is dominated by* $\overrightarrow{y}$
- $\overrightarrow{x}$ *is PS-non-dominated by* $\{\overrightarrow{y}\}$, *otherwise*

The time complexity of the PS-Dominance is in $\mathcal{O}(n \times |\{\overrightarrow{y}\}|)$ as we have to test $\overrightarrow{x}$ with each element of $\{\overrightarrow{y}\}$.

Afterwards, we use $[y]$ to denote any interval of $\mathbb{R}$ and $\overrightarrow{[y]}$ to denote any *interval* vector of $\mathbb{R}^n$. Furthermore, an *interval* vector of $\mathbb{R}^n$ may be also called a box of $\mathbb{R}^n$. So, we can use the PS-Dominance relation between any *decision* vector $\overrightarrow{x}$ of $\mathbb{R}^n$ and any box $\overrightarrow{[y]}$ of $\mathbb{R}^n$ (see figure 2). In this case, the time complexity of the PS-Dominance is in $\mathcal{O}(n)$. Indeed, we only need to test the dominance between $\overrightarrow{x}$ and the point located at the bottom left corner of $\overrightarrow{[y]}$.

Let us consider a box $\overrightarrow{[y]}$ of $\mathbb{R}^2$ and $\overrightarrow{x}$ a point of $\mathbb{R}^2$. In a minimization problem, we may have the examples given at Figure 2.



$\overrightarrow{x}$ PS-Dominates $\overrightarrow{[y]}$ $\overrightarrow{x}$ is PS-Dominated by $\overrightarrow{[y]}$

$\overrightarrow{x}$ is PS-Non-Dominated with $\overrightarrow{[y]}$

Fig. 2. Samples of some PS-Dominance cases.

3.2 Representation of the search and objective space

In most of the population-based algorithms, an individual or configuration of the population is a decision vector, each variable of which being given a single value. Under this representation, each individual corresponds to a particular point in the objective space.

In our approach, each individual is also a vector, but each variable is now represented by an *interval* instead of a single value. Consequently, each individual of the population corresponds now to a *box* $\overrightarrow{[x]}$ of $\mathbb{R}^o$ (o is the number of objectives of the problem) in the objective space.

Consider a problem with two variables x_1, x_2 and two objectives f_1, f_2. A population of three individuals might be represented as in Figure 3.

In this representation, we consider that the whole population gives a sub-paving (i.e. union of non-overlapping boxes) of $\mathbb{R}^o$. As a consequence, the matching boxes of $\mathbb{R}^n$ may overlap (see figure 3).

3.3 Pareto selection of boxes

Consider a set of individuals represented as introduced in Section 3.2. We ensure that all the feasible configurations are contained in the sub-paving described by the population. As a consequence, the Pareto optimal front is also enclosed by the population.

In order to remove the individuals which do not contain any solution of the Pareto optimal set, we apply the following Pareto selection procedure:

1. Try to instantiate all the individuals of the population with bounded local search effort.

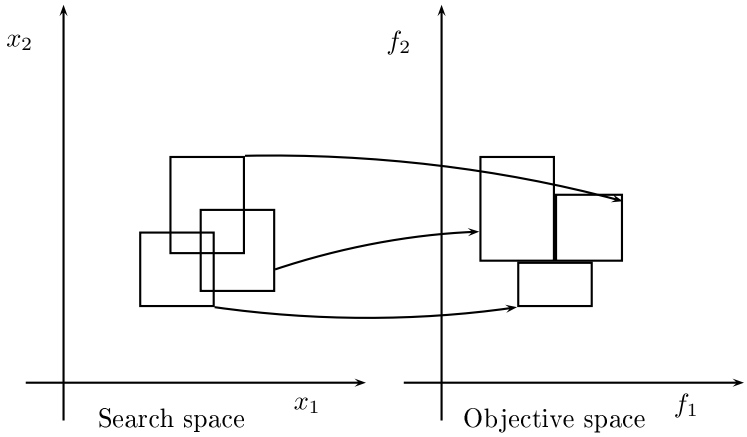


Fig. 3. Sample of the representation with intervals.

2. Apply the PS-Dominance relation 3.1 to remove the individuals which are dominated by another instantiated individual.

At this stage, we get a reduced population of individuals, and we ensure that the union of these boxes (or individuals) contains the whole Pareto optimal set.

We give now some elements of the proof: let $\{\overrightarrow{f}\}$ be the feasible objective space, $\overrightarrow{y}$ a point of the Pareto optimal front and $\overrightarrow{[y]}$ a box that contains $\overrightarrow{y}$:

$$\begin{array}{l} \text{therefore, } \nexists \overrightarrow{x} \in \{\overrightarrow{f}\} \text{ and } \overrightarrow{x} \text{ dominates } \overrightarrow{y} \\ \text{so, } \nexists \overrightarrow{x} \in \{\overrightarrow{f}\} \text{ and } \overrightarrow{x} \text{ PS-dominates } \overrightarrow{[y]} \end{array}$$

As a result, $\overrightarrow{[y]}$ cannot be PS-Dominated and won't be removed from the population.

Figure 4 shows an example of a Pareto selection process with the PS-Dominance relation. We see that the hatched boxes can be safely removed because they are PS-Dominated by some feasible points of the objective space.

3.4 The *PICPA* algorithm

PICPA combines an ICP process (see Section 2) with a Pareto interval selection (see Section 3.3) into a single algorithm.

PICPA uses a population of variable size whose maximum is a parameter to be fixed. *PICPA* starts with a single individual $\overrightarrow{[x]}$ where each variable x_i is initialized with its interval (value domain). We suppose that each variable is locally consistent. If this is not the case, a first ICP process may be applied to reach a fix point. Clearly this individual corresponds to a box of $\mathbb{R}^o$.

Let $\overrightarrow{[f]}$ be this box. Take an objective f_i and bisect its value interval $[f_i]$. Such a bisection triggers two applications of the ICP process to contract variable

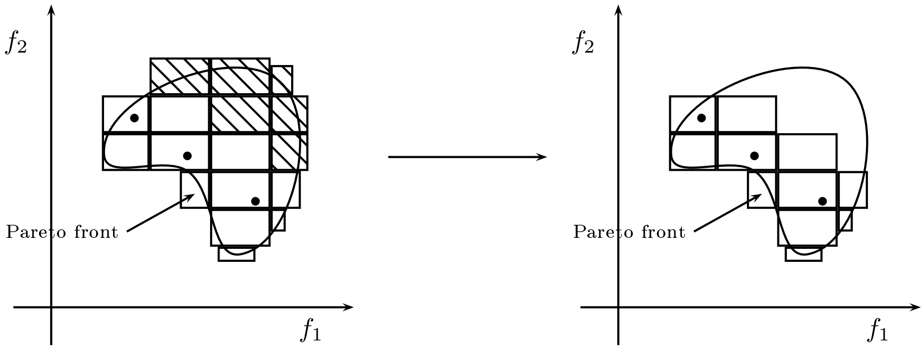


Fig. 4. A Pareto selection sample.

intervals on the individual $\overrightarrow{[x]}$, leading to two new individuals $\overrightarrow{[x']}$ and $\overrightarrow{[x'']}$. These individuals replace the parent individual. These two individuals are composed of reduced intervals and correspond thus to two reduced boxes $\overrightarrow{[f']}$ and $\overrightarrow{[f'']}$ of $\mathbb{R}^o$. This "bisection-contraction" process continues until the number of individuals in the population reaches its allowed maximum. Notice that if any variable domain is reduced to the empty set by the ICP process, the underlying individual will be not inserted into the population.

Once the population reached its full size, an instantiation process will try to create a complete feasible instantiation for each $\overrightarrow{[x]}$. That is, a particular value of the interval will be searched for each individual which satisfies the problem constraints. Since this stage may require a huge processing time, *PICPA* uses a *search effort* parameter as a stop criterion to stop the instantiation process. Thus, after this stage, some individuals will be completely instantiated, leading to a real vector $\overrightarrow{x}$ while others remain uninstantiated. Notice that the result of this instantiation is recorded in a separate data structure and the individuals in the current population will not be altered.

At this point, we execute a Pareto selection mechanism (cf. section 3.3) to eliminate dominated individuals from the population. Since the population is reduced, we start again the above "bisection-contraction" process to extend the population to its maximal size.

The *PICPA* algorithm stops if one of the following cases occurs:

1. an empty population is encountered, in this case, the problem is proved to be infeasible, consequently has no solution;
2. the Pareto selection process cannot remove any individuals from the population. In this case, the individuals of the population constitute the bounds of the problem. The individuals which are successfully instantiated (recorded in a separate structure) give an approximate solution of the Pareto optimal set.

The skeleton of the *PICPA* is shown in algorithm 1.

- Initialize the population with a single locally consistent individual
- `While` $0 < |Population| < MaxPopulationSize$ `do`
  - `While` $0 < |Population| < MaxPopulationSize$ `do`
    1. Select an individual (parent) and bisect it according to one objective, leading to two distinct individuals (children)
    2. Contract the children
    3. Update the population:
       (a) Remove the father
       (b) Add the locally consistent children

    `EndWhile`
  - Potential instantiation of each individual
  - PS-Dominance selection process

  `EndWhile`

Algorithm 1: Skeleton of the *PICPA* algorithm .

As we see on algorithm 1, only two parameters (size of population and search effort) are required by *PICPA*.

PICPA has several advantages compared to other population-based algorithms. Firstly, it requires a small number of parameters. Secondly, it can sometimes answer "No" when the problem is infeasible. Thirdly, it gives in a single run bounds and an approximation of the Pareto optimal front.

3.5 Discussion

PICPA ensures that the Pareto optimal front will be enclosed in the returned population. As *PICPA* mainly bisects in the objective space, it is not very sensitive to the increase of the number of variables. The originalities of *PICPA* are:

1. to bound the number of bisections thanks to the population size
2. to bisect mainly in the objectives space
3. to apply a Pareto selection in order to converge to the Pareto optimal front

The computational effort of *PICPA* can be tuned thanks to the population size parameter. Indeed, larger population sizes lead to higher solution precisions. It is true that increasing the population size will increase the computing time. But this gives us a guaranteed way to get better approximation of the Pareto optimal front.

4 Experimental results

This section gives experimental results of *PICPA* on some famous test problems. Given the deterministic nature of *PICPA*, the quality of solutions of *PICPA* can be directly assessed with respect to the final bounds found. To show its practical performance however, we contrast the results of *PICPA* with those of

NSGA-IIc [7][1]. Notice that the version of the *NSGA-IIc* algorithm used here gives better results than those given in [7]. For these test experiments, the following parameter settings are used:

- for *NSGA-IIc*, we used the settings given in [8], i.e. simulated binary crossover [6] with $n_c = 20$ and the polynomial mutation operator with $n_m = 20$. A crossover probability of 0.9 and a mutation probability of 0.15 are chosen. The population size and the maximum number of generation were set according to the problem difficulty.
- for *PICPA*, we set the population size to 1000 and the search effort to 0.2.

Notice that these settings lead to very close computing time for the two algorithms, ranging from some few seconds to about three minutes according to the test problems. For each test problem, *NSGA-IIc* was run ten times and the best run was taken for our comparisons. As *PICPA* doesn't use any random value, only one run is required to get the result.

4.1 The Tanaka test problem

We first used a test problem introduced by Tanaka [18]:

$$\text{TNK}\begin{cases}\text{Minimize} & f_1(x) = x_1 \\ \text{Minimize} & f_2(x) = x_2 \\ \text{s.t.} & c_1(x) \equiv x_1^2 + x_2^2 - 1 - 0.1\cos(16\arctan(\frac{x_1}{x_2})) \geq 0 \\ & c_2(x) \equiv (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.5 \\ \text{and} & x_1, x_2 \in [0..\pi]\end{cases}$$

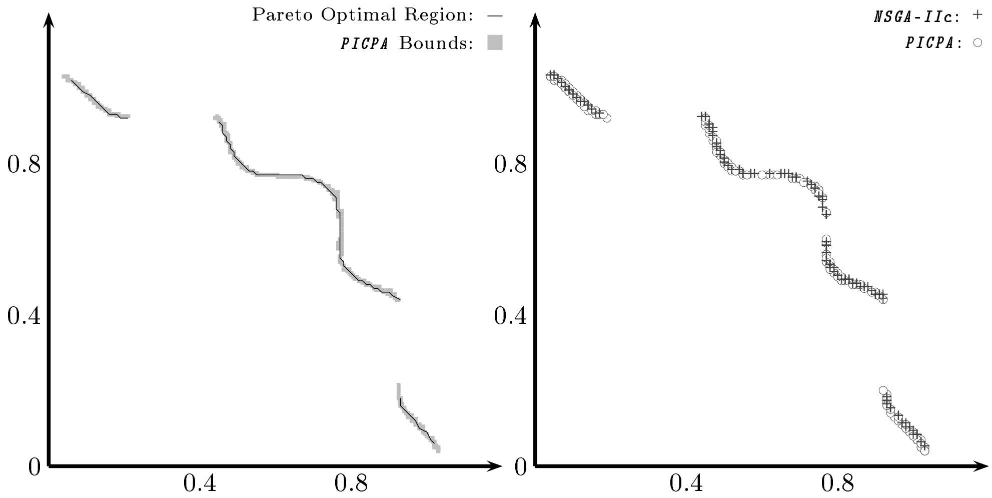
In this problem, the feasible objective space is the same as the feasible decision variable space. In the TNK experiments, we used a population of size 150 and a maximum generation of 500 for *NSGA-IIc*. The optimal Pareto set as well as the bounds given by *PICPA* are presented in figure 5(a). Figure 5(b) shows the return set given by *PICPA* and *NSGA-IIc*.

From these figures, we notice that 1) the bounds of *PICPA* match almost perfectly the Pareto optimal front and 2) *PICPA* and *NSGA-IIc* give very similar solutions.

4.2 The Osyczka and Kundu test problem

For our second experiment, we chose a six variables test problem presented by Osyczka and Kundu [16]:

[1] downloadable at: `http://www.iitk.ac.in/kangal/soft.htm`



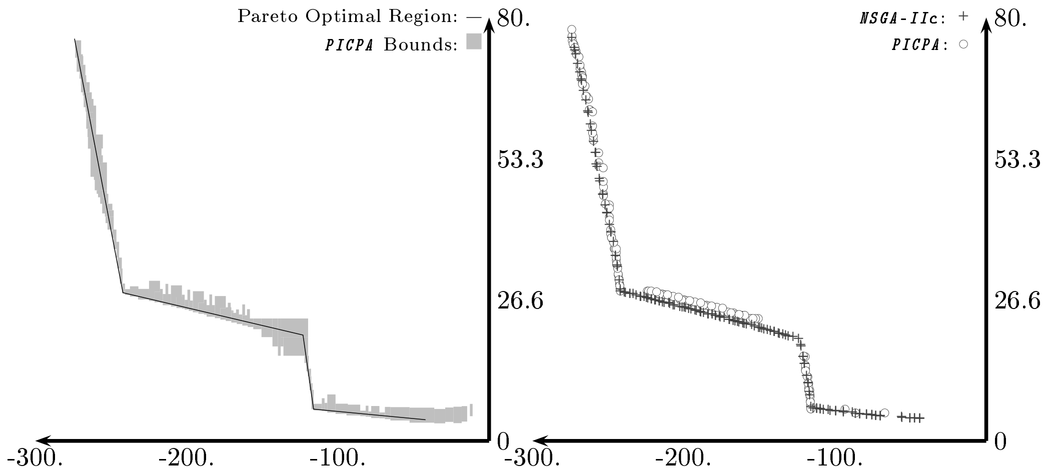
(a) Bound of the Pareto optimal front. (b) Non dominated points.

Fig. 5. Simulation results on TNK.

$$\text{OSY}\begin{cases}\text{Minimize} & f_1(x) = -(25(x_1-2)^2+(x_2-2)^2+(x_3-1)^2+(x_4-4)^2+ \\ & \quad (x_5-1)^2) \\ \text{Minimize} & f_2(x) = x_1^2+x_2^2+x_3^2+x_4^2+x_5^2+x_6^2 \\ \text{s.t.} & c_1(x) \equiv x_1+x_2-2 \geq 0 \\ & c_2(x) \equiv 6-x_1-x_2 \geq 0 \\ & c_3(x) \equiv 2-x_2+x_1 \geq 0 \\ & c_4(x) \equiv 2-x_1+3x_2 \geq 0 \\ & c_5(x) \equiv 4-(x_3-3)^2-x_4 \geq 0 \\ & c_6(x) \equiv (x_5-3)^2+x_6-4 \geq 0 \\ \text{and} & x_1, x_2, x_6 \in [0..10] \\ & x_3, x5, \in [1..5] \\ & x_4, \in [0..6]\end{cases}$$

Like in the TNK experiment (see Section 4.1), we set a population size of 150 and a maximum generation of 500 for `NSGA-IIc`.

From figure 6(a), we observe that the bounds of `PICPA` are globally very close to the optimal front, with exceptions for some areas. These larger areas are due to the great percent of additive constraints. Figure 6(b) shows that the quality of the two non dominated sets found by `PICPA` and `NSGA-IIc` are very close, even if some points found by `NSGA-IIc` dominate some of `PICPA`. But we notice that on this problem, different runs of `NSGA-IIc` give mixed results.



(a) Bound of the Pareto optimal front.

(b) Non dominated points.

Fig. 6. Simulation results on OSY.

4.3 Constrained Test Problems (CTP)

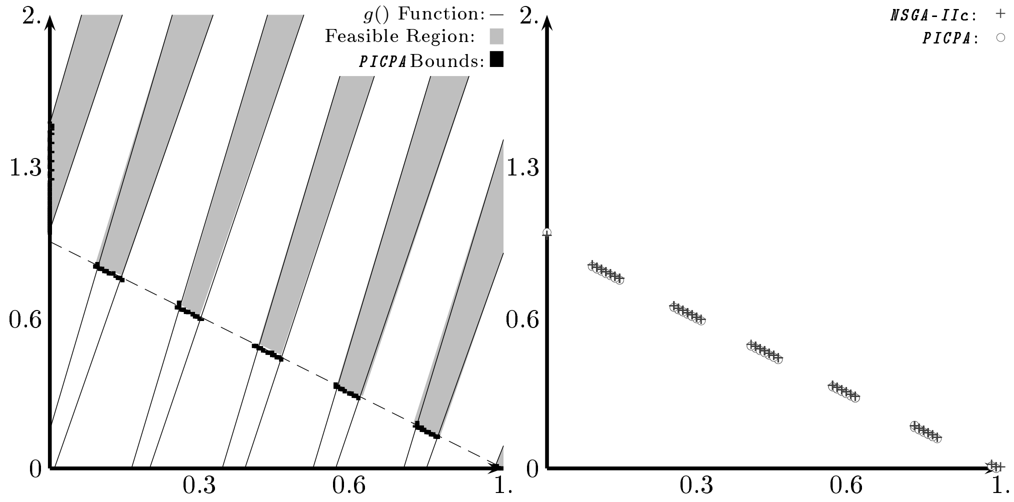
For the last experiments, we used two test problems presented in [8] and [5]. As these problems are tunable, we use a functional $g()$ which is a Rastrigin's function-base:

$$
\text{CTP}\left\{\begin{array}{lll}
\text{Minimize} & f_1(x) & = x_1 \\
\text{Minimize} & f_2(x) & = g(x) - f_1(x) \\
\text{s.t.} & g(x) & = 41 + x_2^2 - 10\cos(4\pi x_2) + x_3^2 - 10\cos(4\pi x_3) + \\
 & & \quad x_4^2 - 10\cos(4\pi x_4) + x_5^2 - 10\cos(4\pi x_5) \\
 & C_1(x) & \equiv \cos(\theta)(f_2(x) - e) - \sin(\theta) f_1(x) \geq \\
 & & \quad a|\sin(b\pi(\sin(\theta)(f_2(x) - e) + \cos(\theta) f_1(x))^c)|^d \\
\text{and} & x_1 \in [0..1] & \\
 & x_{i,i>1}, \in [-5..5] &
\end{array}\right.
$$

For the following experiments, we set a population of size 300 and a maximum generation of 1000 for *NSGA-IIc*. The *PICPA* parameters remain the same as in the previous sections.

CTP7 The parameter values used to get CTP7 are as follows:

$$\theta = -0.05\pi, \quad a = 40, \quad b = 5, \quad c = 1, \quad d = 6, \quad e = 0$$



(a) Bound of the Pareto optimal front.

(b) Non dominated points.

Fig. 7. Simulation results on CTP7.

The feasible search space, the corresponding disconnected Pareto optimal regions and the `PICPA` bounds are shown in figure 7(a).

Figure 7(a) shows the bounds found by `PICPA`, which are very tight with respect to the Pareto optimal front. On this more difficult test problem, the contraction procedures used by `PICPA` proved to be very useful. In figure 7(b), we see that `PICPA` and `NSGA-IIc` have comparable results.

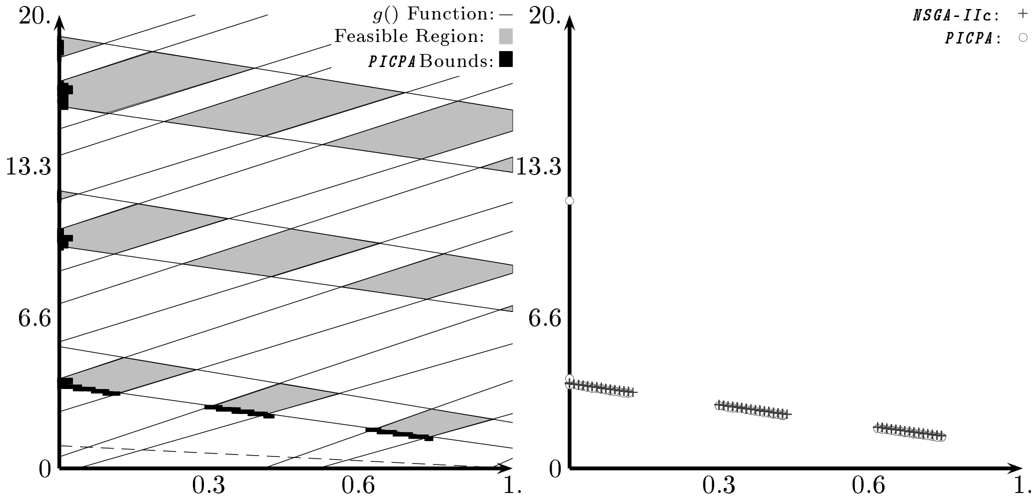
CTP8 CTP8 is composed of many disconnected feasible regions. In CTP8, unlike CTP7, we have two constraints. Here are the parameters used to generate these constraints:

$$C_1 \;:\; \theta = 0.1\pi, \quad a = 40, \quad b = 0.5, \quad c = 1, \quad d = 2, \quad e = -2$$

$$C_2 \;:\; \theta = -0.05\pi, \quad a = 40, \quad b = 2, \quad c = 1, \quad d = 6, \quad e = 0$$

Figure 8(a) shows the feasible search space, the corresponding disconnected Pareto optimal regions and the `PICPA` bounds. Figure 8(b) shows the solution sets by `NSGA-IIc` and `PICPA`.

From figure 8(a), we see once again that the `PICPA` bounds are very precise. For this problem, which is the most difficult problem tested here, `NSGA-IIc` and `PICPA` found some solutions on the most of disconnected Pareto optimal regions.



(a) Bound of the Pareto optimal front.

(b) Non dominated points.

Fig. 8. Simulation results on CTP8.

5 Conclusions

In this paper, we presented *PICPA*, a new algorithm to solve continuous constrained multi-objective problems. *PICPA* combines interval constraint propagation with evolutionary concepts (population and selection). This algorithm has the desirable property of bounding the Pareto optimal front. Experimental evaluation of *PICPA* on some well-known test problems show its practical efficiency to find high quality approximation solutions and very tight bounds of the Pareto optimal front. Also, a new dominance relation called PS-Dominance was proposed which allows to compare a point to a set of points. We think this work fills in a gap of existing population-based methods. *PICPA* strongly depends on the efficiency of the projection and on instantiation procedures. Currently, we are investigating these two issues.

Acknowledgments We would like to thank the referees of the paper for their useful comments.

References

1. F. Benhamou, F. Goualard, L. Granvilliers, and J.F. Puget. Revising hull and box consistency. In *Proceedings of the International Conference on Logic Programming*, pages 230–244, 1999.

2. J.G. Cleary. Logical arithmetic. *Future Computing Systems*, 2(2):125–149, 1987.
3. D.W. Corne and J.D. Knowles. M-paes: a memetic algorithm for multiobjective optimization. In *Proceedings of the 2000 Congres on Evolutionary Computation*, pages 325–332, 2000.
4. E. Davis. Constraint propagation with interval labels. *Artificial Intelligence*, 32(3):281–331, 1987.
5. K. Deb. *Multi-objective optimization using evolutionary algorithms.* John Wiley, 2001.
6. K. Deb and R.B. Agrawal. Simulated binary crossover for continuous search space. *Complex Systems*, 9:115–148, 1995.
7. K. Deb and T. Goel. Controlled elitist non-dominated sorting genetic algorithms for better convergence. In *Proceedings of Evolutionary Multi-Criterion Optimization*, pages 67–81, 2001.
8. K. Deb, A. Pratap, and T. Meyarivan. Constrained test problems for multi-objective evolutionary optimization. In *Proceedings of Evolutionary Multi-Criterion Optimization*, pages 284–298, 2001.
9. C.M. Fonseca and P.J. Fleming. Genetic algorithms for multi-objective optimization: Formulation, discussion and generalization. In *Proceedings of The Fifth International Conference on Genetic Algorithms*, pages 416–423, 1993.
10. X. Gandibleux, N. Mezdaoui, and A. Freville. A multiobjective tabu search procedure to solve combinatorial optimization problems. In *Lecture Notes in Economics and Mathematical Systems*, volume 455, pages 291–300. Springer, 1997.
11. D.E. Goldberg. Genetic algorithms for search, optimization, and machine learning. *Reading, MA: Addison-Wesley*, 1989.
12. L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. *Applied Interval Analysis, with Examples in Parameter and State Estimation, Robust Control and Robotics.* Springer-Verlag, London, 2001.
13. A.K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8:99–118, 1977.
14. R. Mohr and T.C. Henderson. Arc and path consistency revisited. *Artificial Intelligence*, 28:225–233, 1986.
15. R.E. Moore. Methods and applications of interval analysis. *SIAM, Philadelphia, PA*, 1979.
16. A. Osyczka and S. Kundu. A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm. *Structural Optimization*, 10:94–99, 1995.
17. N. Srinivas and K. Deb. Multiobjective optimization using non dominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.
18. M. Tanaka. Ga-based decision support system for multi-criteria optimization. In *Proceedings of the International Conference on Systems, Man and Cybernetics-2*, pages 1556–1561, 1995.
19. E.L. Ulungu, J. Teghem, Ph. Fortemps, and D. Tuyttens. Mosa method: a tool for solving multiobjective combinatorial optimization problems. *Journal of Multi-Criteria Decision Analysis*, 8:221–336, 1999.
20. E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3:257–271, 1999.