Journal of Universal Computer Science, vol. 16, no. 18 (2010), 2657-2685
submitted: 14/12/09, accepted: 16/7/10, appeared: 28/9/10 © J.UCS

From Computing Sets of Optima, Pareto Sets, and Sets of Nash Equilibria to General Decision-Related Set Computations

Vladik Kreinovich
(University of Texas at El Paso, USA
vladik@utep.edu)

Bartłomiej Jacek Kubica
(Warsaw University of Technology, Poland
B.Kubica@elka.pw.edu.pl)

Abstract: Several algorithms have been proposed to compute sets of optima, Pareto sets, and sets of Nash equilibria. In this paper, we present a general algorithm for decision-related set computations that includes all these algorithms as particular cases.

To make our algorithm understandable to people working in optimization and in game theory, we also provide motivations and explanations for our formalizations of the corresponding problems and for the related notions of computable mathematics.

Key Words: computing sets, sets of optima, Pareto sets, Nash equilibria
Category: F.2.1, F.4, G.1.m

1 Introduction

The main objective of this paper is to provide algorithms which can be useful for solving practical decision-making problems.

We start, in Section 2, with describing the main types of decision-related problems like optimization. After describing each problem in decision-related terms, we explain reasonable ways to describe these problems in precise mathematical ("formal") terms.

Our objective is to "compute" solutions to these problems. So, before we explain how to compute these solutions, we first recall, in Section 3, what it means to compute a solution. Specifically, in this section, we recall the main definitions and results related to computability that we will use in our algorithms. These definitions and results are mostly taken from [Pour-El and Richards 1989] and [Weihrauch 2000]. To make our algorithms understandable to people working in optimization and in game theory, we also provide motivations and explanations for the related notions of computable mathematics.

Then, in Section 4, we formulate our main result – about the existence of the corresponding algorithms. The proof of this result is presented in Section 5.

Finally, in Section 6, we briefly discuss the questions of computational complexity and feasibility of the resulting algorithms.

2 Practical decision-related problems and their formal descriptions

Optimization is important. In many practical decision making problems, we are interested in finding the alternative which is the best (under given constraints). In many cases, an objective function $f(x)$ is explicitly given. In these cases, "the best" means that we want to find a solution which maximizes the value of this objective function, i.e., a solution x^* for which the value $f(x^*)$ cannot be improved – i.e., for every element x of the set X of all possible solutions, we have $f(x^*) \geq f(x)$.

In formal terms, the condition that x^* is a location of a global optimum can be described as

$$\forall x \ (f(x^*) \geq f(x)) . \qquad (1)$$

Computing sets of optima is important. Often, there are several optima. In this case, it is desirable to provide the user with the set of all these optima, so that the user can see all the options when selecting an alternative.

From this viewpoint, it is desirable to "compute" (in some reasonable sense) the set of all the optima – i.e., the set of all the points x^* that satisfy the condition (1).

Computing an exact global optimum of a computable function is, in general, not algorithmically possible. It is known that even for functions defined on the interval $[0,1]$, in general, it is not possible to have an algorithm that, given a computable function $f : [0,1] \to \mathbb{R}$, returns a location x^* of its local maxima. This result was first proven in [Specker 1959] (definitions of a computable function will be reminded later in this paper).

Comment. Since 1959, Specker's result has been expanded. For example, in [Kreinovich et al. 1998], it has been proven that no algorithm is possible that, given a computable polynomial of one variable which attains its optimum at exactly two points, will return these two optimizing points; see also [Nachbar and Zame 1996].

In practice, it is sufficient to be able to compute an approximation to the set of optima. From the practical viewpoint, the above algorithmic impossibility result is not that negative, since in practice, small differences in the values of the objective function can be safely ignored. Also, the objective function $f(x)$ describing the consequences of selecting an alternative x is usually only known approximately.

Let us denote, by ε, the accuracy below which differences in the values of $f(x)$ can be safely ignored. Thus,

- the given value $f(x^*)$ means that the actual (unknown) value $f_{\text{act}}(x^*)$ of the objective function can be any number from the interval $[f(x^*) - \varepsilon, f(x^*) + \varepsilon]$;
- the given value $f(x)$ means that the actual (unknown) value $f_{\text{act}}(x)$ of the objective function can be any number from the interval $[f(x) - \varepsilon, f(x) + \varepsilon]$;
- the optimization requirement $f_{\text{act}}(x^*) \geq f_{\text{act}}(x)$ means, in these terms, that there exist values $f_{\text{act}}(x^*) \in [f(x^*) - \varepsilon, f(x^*) + \varepsilon]$ and $f_{\text{act}}(x) \in [f(x) - \varepsilon, f(x) + \varepsilon]$ for which $f_{\text{act}}(x^*) \geq f_{\text{act}}(x)$.

If one of the elements of the interval $[f(x^*) - \varepsilon, f(x^*) + \varepsilon]$ is larger than or equal to one of the elements of the interval $[f(x) - \varepsilon, f(x) + \varepsilon]$, then the largest element $f(x^*) + \varepsilon$ of the interval $[f(x^*) - \varepsilon, f(x^*) + \varepsilon]$ is larger than or equal to the smallest element $f(x) - \varepsilon$ of the interval $[f(x) - \varepsilon, f(x) + \varepsilon]$, i.e., $f(x^*) + \varepsilon \geq f(x) - \varepsilon$.

Vice versa, if $f(x^*) + \varepsilon \geq f(x) - \varepsilon$, then an element $f(x^*) + \varepsilon$ of the interval $[f(x^*) - \varepsilon, f(x^*) + \varepsilon]$ is larger than or equal to the element $f(x) - \varepsilon$ of the interval $[f(x) - \varepsilon, f(x) + \varepsilon]$. Thus, the above condition is equivalent to $f(x^*) + \varepsilon \geq f(x) - \varepsilon$, i.e., to $f(x^*) \geq f(x) - 2 \cdot \varepsilon$.

So, it is desirable to describe the set of all the values x^* for which $f(x^*) \geq f(x) - 2 \cdot \varepsilon$ for all $x \in X$, i.e., in formal notations, for which

$$\forall x \; (f(x^*) \geq f(x) - 2 \cdot \varepsilon). \tag{2}$$

It is reasonable to call the alternatives x^* that satisfy this requirement $(2 \cdot \varepsilon)$-*optima.* In these terms, instead of computing the set of all the optima, we are arriving at a modified formalization of the original practical problem: compute the set of all $(2 \cdot \varepsilon)$-optima.

Additional complication: the approximation accuracy is also not exactly known. The above modified formulation implicitly assumes that we know the exact approximation accuracy ε. In other words, we assume that we know the exact value ε such that smaller differences between the values of the objective function $f(x)$ can be safely ignored.

In practice, of course, this "threshold" value ε is also known with uncertainty. A reasonable person can say that, e.g., 1% difference in the values of the objective function can be safely ignored but 2% difference is no longer negligible. However, it is difficult to expect a user to claim that a difference below 1.235% can be safely ignored, while any difference above this threshold value 1.235% is not negligible.

In other words, instead of a *single* exact value ε, we usually have *two* bounds $\underline{\varepsilon} < \overline{\varepsilon}$, so that:

- every difference smaller than $\underline{\varepsilon}$ can be safely ignored, while

– differences larger than $\overline{\varepsilon}$ cannot be ignored.

Computing the set of optima: the final formulation of the problem. If we take into account the uncertainty with which we know the accuracy ε, then we come to the following conclusion:

– every $(2 \cdot \underline{\varepsilon})$-optimal alternative is desirable, and

– every desirable alternative must be $(2 \cdot \overline{\varepsilon})$-optimal.

In other words, given an objective function f, we would like to compute a set S with the following two properties:

– every $(2 \cdot \underline{\varepsilon})$-optimal alternative belongs to the set S, and

– every alternative from the set S is $(2 \cdot \overline{\varepsilon})$-optimal.

Comment. In mathematical physics, there is a useful distinction between

– *strong* approximations which are close to the actual solution of the *original* problem, and

– *weak* approximations which solve the *approximate* problem, i.e., which have *properties* similar to the properties of the actual solution.

In these terms, our explanations show that in many practical problems, we are actually interested in weak approximations. In this paper, we produce an algorithm that computes such weak approximations.

The resulting problem is algorithmically solvable. It turns out that the problem of computing a set S with the above property is algorithmically solvable; see, e.g., [G.-Toth and Kreinovich 2009]. Specifically, it is possible to produce a finite list of elements L and a rational value $\delta > 0$ such that the set S of all the alternatives which are δ-close to one of the elements of L is the desired set.

Formally, if we denote the set of all $(2 \cdot \varepsilon)$-optimal alternatives by $M_{2\cdot\varepsilon}(f)$, then the following two conditions are satisfied:

– If $x_0 \in M_{2\cdot\underline{\varepsilon}}(f)$, then $d(x_0, \ell) \le \delta$ for some $\ell \in L$.

– If $d(x_0, \ell) \le \delta$ for some $\ell \in L$, then $x_0 \in M_{2\cdot\overline{\varepsilon}}(f)$.

In other words, the union S of the corresponding balls $B_\delta(\ell) \stackrel{\text{def}}{=} \{x : d(\ell, x) \le \delta\}$ satisfies the following property:

$$M_{2\cdot\underline{\varepsilon}}(f) \subseteq \bigcup_{\ell \in L} B_\delta(\ell) \subseteq M_{2\cdot\overline{\varepsilon}}(f)\,. \tag{3}$$

Comment. We do not describe here the algorithm for computing the list L, since later in this paper, we present an algorithm for solving a more general problem.

Comment about continuity. The above result is based on the (implicit) assumption that the objective function f is continuous. Continuous objective functions describe the usual consequences of different actions, since usually a small change in the solution only leads to a small change in the consequences.

In principle, there are some cases when the objective function is *not* continuous. For example, for some undesired side products of an industrial process, there is usually a threshold beyond which heavy fines start. In such situations, however, the desire is to avoid exceeding this threshold. Thus, the environmentally proper way of handling these situations is *not* to incorporate these fines into the profit estimates, but rather to avoid such undesirable situations altogether, and to view these restrictions as constraints that limit the set X of possible solutions. On thus restricted set, the objective function is continuous. Such constraint optimization problems will be discussed later in this section.

A more general problem: computing the Pareto set. The above description of the decision making problem assumes that we have a *single* objective function that we are trying to maximize – albeit an imprecisely known one. In other words, we assume that we have already agreed how to combine different characteristics describing different aspects of the problem into a single numerical quantity.

In practice, we usually have several objective functions

$$f(x) = (f_1(x), \ldots, f_n(x))$$

describing different aspects of the possible solution x, such as profit, environmental friendliness, safety, etc. Ideally, we should maximize the values of *all* these characteristics, but in reality, there is often a trade-off: e.g., to achieve more environmental friendliness, it is often necessary to slightly decrease the profit; there is a similar trade-off between cost and durability.

In many situations, the user does not have a clear a priori idea which trade-offs are beneficial and which are not; in other words, the user does not have a single combined objective function $f(x)$ that would enable him or her to make an ultimate decision. In such situations, it is reasonable to present the user with the *set* of all possible solutions – and let the user decided between different possible solutions from this set. The only possible solutions x^* that we do not want to present to the user are solutions x^* which can be improved in all the senses, i.e., solutions for which, for some other solution x, we have $f_j(x^*) \le f_j(x)$ for all j and $f_j(x^*) < f_j(x)$ for some j. The set of all such "non-improvable" solution is known as the *Pareto set.* The problem is how to compute the Pareto set. This problem has many practical applications; see, e.g., [Figueira et al. 2004].

In formal terms, an alternative x^* is *dominated* if

$$\exists x \ (f_1(x) \geq f_1(x^*) \ \& \ \ldots \ \& \ f_n(x) \geq f_n(x^*) \ \&$$
$$(f_1(x) > f_1(x^*) \vee \ldots \vee f_n(x) > f_n(x^*))). \tag{4}$$

Thus, the condition that x^* is *not* dominated (= Pareto optimal) takes the form

$$\forall x \ (f_1(x^*) > f_1(x) \vee \ldots \vee f_n(x^*) > f_n(x) \vee$$
$$(f_1(x^*) \geq f_1(x) \ \& \ \ldots \ \& \ f_n(x^*) \geq f_n(x))). \tag{5}$$

In these terms, the Pareto set is the set of all the alternatives x that satisfy the property (5).

In general, the computation of a Pareto set is an algorithmically undecidable problem. There exist efficient algorithms for computing the Pareto set for several important specific classes of problems: e.g., for special location problems [Nickel and Puerto 2005] and for problems with linear objective functions [Figueira et al. 2004].

In general, however, this problem is known to be computationally difficult; see, e.g., [Ruzika and Wiecek 2005]. This difficulty has a theoretical explanation – this problem is, in general, algorithmically undecidable. This undecidability directly follows from the fact that for $n = 1$, we get the problem of computing the set of optima, the problem which is (as we have mentioned earlier) algorithmically undecidable.

The problem of computing a Pareto set becomes decidable if we take into account that the objective functions are known with some accuracy. In practice, as we have mentioned, we know each of the objective functions $f_j(x)$ only with some accuracy ε_j. It turns out that if we appropriately take this uncertainty into account, then (verified) algorithms for computing the resulting Pareto set become possible. Such algorithms were described, for the case of $n = 2$ objective functions f_j defined on bounded subsets of $\mathbb{R}^m$, in [Fernández et al. 2006], [Fernández and Tóth 2006], [Tóth and Fernández 2006], [Fernández and Tóth 2007], [Fernández and Tóth 2009]. For the general case of arbitrary computable objective functions defined on a general "computable" compact set X, the result is given in [G.-Toth and Kreinovich 2009]. (A similar algorithm is presented in [Kubica and Woźniak 2008].)

Specifically, we assume that for every j, we know the bounds $\underline{\varepsilon}_j < \overline{\varepsilon}_j$ on the (unknown) accuracy ε_j. Similarly to the optimization case, for each combination of values $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_n)$, we say that an alternative x^* is $(2 \cdot \varepsilon)$-*Pareto optimal* if it satisfies the following property:

$$\forall x \ ((f_1(x^*) \geq f_1(x) - 2 \cdot \varepsilon_1) \vee \ldots \vee (f_n(x^*) \geq f_n(x) - 2 \cdot \varepsilon_n) \vee$$

$$((f_1(x^*) \geq f_1(x) - 2 \cdot \varepsilon_1) \ \& \ \ldots \ \& \ (f_n(x^*) \geq f_n(x) - 2 \cdot \varepsilon_n))) \,. \tag{6}$$

Comments.

- Please note that since we are considering approximate values anyway, we replaced the strict equality $f_1(x^*) > f_1(x) - 2 \cdot \varepsilon$ with a non-strict one $f_1(x^*) \geq f_1(x) - 2 \cdot \varepsilon$.
- After this replacement, the second part

  $$((f_1(x^*) \geq f_1(x) - 2 \cdot \varepsilon) \ \& \ \ldots \ \& \ (f_n(x^*) \geq f_n(x) - 2 \cdot \varepsilon))$$

  of the condition (6) follows from the first one, so the requirement (6) take the simplified form:

$$\forall x \ ((f_1(x^*) \geq f_1(x) - 2 \cdot \varepsilon_1) \vee \ldots \vee (f_n(x^*) \geq f_n(x) - 2 \cdot \varepsilon_n)) \,. \tag{7}$$

End of comments.

Given a tuple of objective functions

$$f = (f_1, \ldots, f_n) \,,$$

we would like to compute a set S with the following two properties:

- every $(2 \cdot \underline{\varepsilon})$-Pareto optimal alternative belongs to the set S, and
- every alternative from the set S is $(2 \cdot \overline{\varepsilon})$-Pareto optimal.

Specifically, it is possible to produce a finite list of elements L and a rational value $\delta > 0$ such that the set S of all the alternatives which are are δ-close to one of the elements of L is the desired set.

Formally, if we denote the set of all $(2 \cdot \varepsilon)$-Pareto optimal alternatives by $P_{2 \cdot \varepsilon}(f)$, then the following two conditions are satisfied:

- If $x^* \in P_{2 \cdot \underline{\varepsilon}}(f)$, then $d(x^*, \ell) \leq \delta$ for some $\ell \in L$.
- If $d(x^*, \ell) \leq \delta$ for some $\ell \in L$, then $x^* \in P_{2 \cdot \overline{\varepsilon}}(f)$.

In other words,

$$P_{2 \cdot \underline{\varepsilon}}(f) \subseteq \bigcup_{\ell \in L} B_\delta(\ell) \subseteq P_{2 \cdot \overline{\varepsilon}}(f) \,. \tag{8}$$

A similar problem: computing the set of Nash equilibria. In the previous text, we considered the problem of selecting an alternative in which one person (or one entity) makes the decision, and the results depend only on this person's decision.

In practice, often, we have several different persons, with potentially different objective functions $f_1(x), \ldots, f_m(x)$, and we need to take into account the interests of all the participants. Such decision problems are handled in game theory. One of the most widely used solution concept is the concept of *Nash equilibrium*: participants select a joint decision $x^* = (x_1^*, \ldots, x_m^*)$ in such a way that none of them has the incentive to unilaterally change the decision. In other words, for every $x_i \in X_i$, we have

$$f_i\left(x_1^*, \ldots, x_{i-1}^*, x_i^*, x_{i+1}^*, \ldots, x_m^*\right) \geq f_i\left(x_1^*, \ldots, x_{i-1}^*, x_i, x_{i+1}^*, \ldots, x_m^*\right). \quad (9)$$

Formally, we can say that $x^* = (x_1^*, \ldots, x_m^*)$ is a Nash equilibrium if the following condition holds:

$$(\forall x_1\ (f_1(x_1^*, x_2^*, \ldots, x_m^*) \geq f_1(x_1, x_2^*, \ldots, x_m^*))\ \&\ \ldots\ \&$$
$$\left(\forall x_m\ \left(f_m\left(x_1^*, \ldots, x_{m-1}^*, x_m^*\right) \geq f_m\left(x_1^*, \ldots, x_{m-1}^*, x_m\right)\right)\right). \quad (10)$$

In general, the computation of the set of all Nash equilibria is an algorithmically undecidable problem. Indeed, similarly to the Pareto case, this undecidability directly follows from the fact that for $m = 1$, we get the problem of computing the set of optima, the problem which is (as we have mentioned earlier) algorithmically undecidable.

The problem of computing the set of all Nash equilibria becomes decidable if we take into account that the objective functions are known with some accuracy. In practice, as we have mentioned, we know each of the objective functions $f_j(x)$ only with some accuracy ε_j. It turns out that if we appropriately take this uncertainty into account, then (verified) algorithms for computing the resulting set of (approximate) Nash equilibria become possible; see, e.g., [Kubica and Woźniak 2010].

The above three cases are similar. The above three cases are similar, the results and algorithms are similar. It is therefore desirable to find a general formulation of a decision making problem that would include these results as particular cases.

This desirability also comes from the fact that several other practically important decision making problems can be formulated in a similar manner – so a general result will enable us to solve all these problems as well.

Another example of a similar problem. An example of such a decision making problem is the problem of constraint optimization. In general, we can have constraints of the equality type $a(x) = b(x)$ and constraints of the inequality type $a(x) \geq b(x)$. By moving all the terms to one side, we can have an equivalent reformulation as $c(x) = 0$ and $c(x) \geq 0$, where $c(x) \stackrel{\text{def}}{=} a(x) - b(x)$. Every constraint $c(x) = 0$ of the equality type can be represented as two inequality constraints $c(x) \geq 0$ and $-c(x) \geq 0$. Thus, in general, we can formulate the constraints optimization problem as follows: optimizing a given objective function $f(x)$ under constraints $c_1(x) \geq 0, \ldots, c_n(x) \geq 0$.

For the alternative x^* to be the location of the conditional optimum, this alternative must satisfy the following two requirements:

– the alternative x^* must satisfy all n constraints $c_i(x^*) \geq 0$, and

– for every other alternative x that satisfies all n constraints, we must have $f(x^*) \geq f(x)$.

Formally, these two requirements have the following form

$$c_1(x^*) \geq 0 \,\&\, \ldots \,\&\, c_n(x^*) \geq 0 \,\&$$

$$\forall x\, ((c_1(x) \geq 0 \,\&\, \ldots \,\&\, c_n(x) \geq 0) \to f(x^*) \geq f(x)) . \tag{11}$$

Replacing implication $A \to B$ with the equivalent formula $B \vee \neg A$, we get the following equivalent reformulation of (11) which makes it even closer to our previous problems:

$$c_1(x^*) \geq 0 \,\&\, \ldots \,\&\, c_n(x^*) \geq 0 \,\&$$

$$\forall x\, (f(x^*) \geq f(x) \vee c_1(x) < 0 \vee \ldots \vee c_n(x) < 0) . \tag{12}$$

It is desirable to prove that – similarly to the above results – the natural ε-approximation is computable.

Comment. For conditions of the type $c(x) \geq 0$, the effect of inaccuracy is somewhat different than for inequalities of the above type $f(x) \geq f(x')$.

Indeed, let us denote, by ε, the accuracy below which differences in the values of $c(x)$ can be safely ignored. Thus, the given value $c(x)$ means that the actual (unknown) value $c_{\text{act}}(x)$ of the corresponding quantity can be any number from the interval $[c(x) - \varepsilon, c(x) + \varepsilon]$.

The constraint $c(x) \geq 0$ means, in these terms, that there exist a value $c_{\text{act}}(x) \in [c(x) - \varepsilon, c(x) + \varepsilon]$ for which $c_{\text{act}}(x) \geq 0$.

If one of the elements of the interval $[c(x) - \varepsilon, c(x) + \varepsilon]$ is larger than or equal to 0, then the largest element $c(x) + \varepsilon$ of the interval

$$[c(x) - \varepsilon, c(x) + \varepsilon]$$

is larger than or equal to 0: $c(x) + \varepsilon \geq 0$.

Vice versa, if $c(x) + \varepsilon \geq 0$, then an element $c(x) + \varepsilon$ of the interval

$$[c(x) - \varepsilon, c(x) + \varepsilon]$$

is larger than or equal to 0. Thus, the above condition is equivalent to $c(x)+\varepsilon \geq 0$, i.e., to $c(x) \geq -\varepsilon$.

Note that here we have $-\varepsilon$ instead of $-2 \cdot \varepsilon$.

Other possible examples. Similarly, we can consider Pareto optimization under constraints, Nash equilibrium under constraints, etc.

Another case is when we do not have any objective function, we simply want to find all the alternatives that satisfy the given constraint(s).

Yet another case is when we want to find a alternative x^* which guarantees a certain level of outcome no matter what alternative y is selected by the second participant: $\forall y\ (f(x^*, y) \geq f_0)$, i.e., equivalently, $\forall y\ (f(x^*, y) - f_0 \geq 0)$.

It is also possible to look for a *maximin* solution, a solution x^* for which the worst-case outcome $\min\limits_y f(x^*, y)$ is the largest possible. In this solution,

- there is an alternative y_w for which the value $f(x^*, y_w)$ is the smallest possible, and
- for every other selection x, the value $f(x, y)$ can be smaller than or equal to $f(x^*, y_w)$, i.e., there exists y for which $f(x, y) \leq f(x^*, y_w)$.

Formally, this property has the form

$$\exists y_w\ (\forall y\ (f(x^*, y_w) \leq f(x^*, y))\ \&\ \forall x\, \exists y\ (f(x^*, y_w) \geq f(x, y)))\,.$$

In some problems, we look for *local optima*, i.e., for a value x^* for which, for all x within a certain radius d from x^*, we have $f(x^*) \geq f(x)$. Local optima are important in many practical problems: to separate an image of an astronomical object into components; in spectroscopy to subdivide the observed spectrum into individual lines corresponding to different ions and chemical substances, etc.; see, e.g., [Villaverde and Kreinovich 1993].

Formally, a local maximum at x^* means that we have

$$\forall x\ (d(x, x^*) \leq d \rightarrow f(x^*) \geq f(x))$$

i.e.,

$$\forall x\ (d - d(x, x^*) \geq 0 \rightarrow f(x^*) \geq f(x))\,.$$

3 What is a computable function, what is a computable set: a brief reminder

Computable numbers and computable functions. In the global optimization problem, we have a set of alternatives X, we have an objective function

$f : X \to \mathbb{R}$, and we are interested in computing the set of all optima. In other problems, we have one or more functions, and we want to compute an appropriate set. In order to analyze these problems from the algorithmic viewpoint, we need to know how this information is represented in a computer. I n other words, we must start with a "computable" set X and "computable" function(s) f, and we must "compute" the corresponding solution set S.

The notions of computable numbers and computable functions are known. These notions, originated proposed in the pioneer works [Grzegorczyk 1955, Grzegorczyk 1957], form the basis of *computable mathematics*; see, e.g., [Pour-El and Richards 1989], [Ko 1991], [Edalat and Heckmann 1998], [Weihrauch 2000] and references therein. The notions of computable sets (and computable compact sets) are also known; see, e.g., [Weihrauch 2000] and [Brattka and Presser 2003].

As mentioned in [Weihrauch 2000], there are different approaches to computable mathematics, but all these approaches are (largely) equivalent. In this paper, we will mainly use notations from [Pour-El and Richards 1989] and results from [Pour-El and Richards 1989] and [Weihrauch 2000].

For example, we call a real number x *computable* if there exists an algorithmic sequence of rational numbers r_n for which $|r_n - x| \le 2^{-n}$.

A function $f(x_1, \ldots, x_m)$ from a box $[\underline{x}_1, \overline{x}_1] \times \ldots \times [\underline{x}_n, \overline{x}_m]$ with computable endpoints $\underline{x}_i$ and $\overline{x}_i$ is called *computable* if:

– there is an algorithm that, given a rational-valued tuple $r = (r_1, \ldots, r_m)$ from the given box and an integer n, computes an 2^{-n}-approximation to $f(r)$, i.e., a rational number s for which $|s - f(r)| \le 2^{-n}$, and

– there exists an algorithm transforming every integer k into an integer ℓ for which $d(x, x') \le 2^{-\ell}$ implies $|f(x) - f(x')| \le 2^{-k}$.

Once we have these two algorithms, then, for every tuple $x = (x_1, \ldots, x_m)$ of computable real numbers x_i, we can compute $f(x)$ with a given accuracy 2^{-n}. First, we use the second algorithm to find an integer ℓ for which $d(x, x') \le 2^{-\ell}$ implies $|f(x) - f(x')| \le 2^{-(n+1)}$. Since the values $x_1, \ldots, x_m$ are computable real numbers, we can algorithmically find rational approximations $r = (r_1, \ldots, r_m)$ for which $d(r, x) \le 2^{-\ell}$ – and thus, for which $|f(r) - f(x)| \le 2^{-(n+1)}$. Now, we use the first algorithm to compute the rational number s for which $|s - f(r)| \le 2^{-(n+1)}$. For this rational number, we have

$$|s - f(x)| \le |s - f(r)| + |f(r) - f(x)| \le 2^{-(n+1)} + 2^{-(n+1)} = 2^{-n};$$

thus s is the desired 2^{-n}-approximation to $f(x)$.

Alternatively, instead of an algorithm transforming k into ℓ, we can require that there is an algorithm for computing the corresponding modulus of continuity, i.e., an algorithm that transforms every real number $\varepsilon > 0$ into a number $\delta = \omega_f(\varepsilon) > 0$ for which $d(x, x') \le \delta$ implies $|f(x) - f(x')| \le \varepsilon$.

As we have mentioned earlier, all standard computer-implemented functions such as $\sqrt{}$, exp, sin, ln, etc., are computable in this sense. In particular, the possibility to find δ from ε is based on the fact that most of these functions have a Lipschitz property $|f(x) - f(x')| \le L \cdot d(x, x')$ for a known L. It is also known that a composition of computable functions is also computable. Thus, all practical objective functions are computable in this sense.

Computable metric spaces and computable sets: motivation. Which definition of a computable set is the most appropriate for our applied problems? In a computer, at any given moment of time, we can only represent finitely many objects, and we can only represent them approximately – e.g., to produce the exact real number, we need to describe infinitely many digits, and a computer can only produce finitely many digits in any given time interval. As we increase the memory size, we can get more and more accurate representations of each object.

Thus, it is reasonable to require that for each approximation accuracy $\varepsilon = 2^{-k}$, we have a finite set $X^{(k)}$ of elements that can approximate any element x from the universal set X with this accuracy, i.e., for which, for every $x \in X$, there exists an element $z \in X^{(k)}$ for which $d(x, z) \le 2^{-k}$.

The elements z corresponding to all possible k's form a sequence whose elements can approximate an arbitrary element $x \in X$ with an arbitrary accuracy, i.e., sequence which is *dense* in X.

In metric spaces, a finite list $X^{(k)}$ with the above ε-approximation property is called a *ε-net*, and a metric space that has a finite ε-net for every $\varepsilon > 0$ is called *totally bounded*. This notion is closely related to the more well-known notion of compactness: namely, a metric space is compact if and only if it is complete and totally bounded. Thus, what we need are effectively totally bounded computable metric spaces.

Similarly, when we say that we want to generate a set S (e.g., the set of all ε-optimal points), what we want is to be able to generate the corresponding approximations $S^{(k)}$ for this set – i.e., we want a set which is also effectively totally bounded.

We will therefore use the notion of effectively totally bounded sets and metric spaces that has indeed been introduced in computable analysis; see, e.g., [Yasugi et al. 1999, Iljazovic 2009].

Comment 1. From the approximation viewpoint, a limit point $x = \lim x_n$ is indistinguishable from the points x_n in the following sense: whatever accuracy $\varepsilon > 0$ we select, there exists an n for which $d(x, x_n) \le \varepsilon$, i.e., for which, within the given accuracy, x and x_n are indistinguishable. Thus, without changing any observable properties, we can assume that the metric space X contains all its limit points, i.e., that it is *complete*.

Since we have already assumed that X is totally bounded, we can therefore conclude that X is a compact metric space.

Comment 2. The notion of an ε-net can be reformulated in terms of the Hausdorff distance $d_H(A,B) \stackrel{\text{def}}{=} \max\left(\max\limits_{a\in A} d(a,B), \max\limits_{b\in B} d(b,A)\right)$, where $d(a,B) \stackrel{\text{def}}{=} \min\limits_{b\in B} d(a,b)$. Namely, a set S is an ε-net for a space X if and only if $d_H(S,X) \leq 2^{-k}$.

Computable metric spaces, computable sets, and computable functions on computable sets: resulting definitions. Let (X,d) be a metric space. A tuple $(X, d, \{x_1, \ldots, x_n, \ldots\})$, where $x_i \in X$, is called a *computable metric space* if the sequence x is dense in X and the corresponding distance function $i, j \to d(x_i, x_j)$ is computable. An element $x \in X$ is called *computable* if there is an algorithm that, given k, return ℓ for which $d(x, x_\ell) \leq 2^{-k}$.

According to [Yasugi et al. 1999, Iljazovic 2009], a computable metric space is called *effectively totally bounded* if there exists an algorithm that, given a natural number k, returns a finite set $X^{(k)} \subseteq X$ which is a 2^{-k}-net for X. One can easily prove that this definition is equivalent to the requirement that there is an algorithm that transform every k into a finite list of computable elements that forms a 2^{-k}-net for the space X. In this paper, we will consider effectively totally bounded spaces which are complete – and hence compact.

For short, we will call effectively totally bounded computable compact metric spaces *effectively compact*.

Similarly, we say that a compact set $S \subseteq X$ in an effectively compact metric space is *effectively compact* if there exists an algorithm that, given a natural number k, returns a finite list $S^{(k)} \subseteq S$ of computable elements that forms a 2^{-k}-net for the set S.

A function $f : X \to \mathbb{R}$ from an effectively compact metric space x to real numbers is called *computable* if we have two algorithms:

- an algorithm that, given n, computes $f(x_n)$, i.e., more precisely, an algorithm that, given integers n and k, computes a rational number r which is 2^{-k}-close to $f(x_n)$: $|r - f(x_n)| \leq 2^{-k}$, and
- an algorithm that computes the modulus of continuity, i.e., an algorithm that transforms every real number $\varepsilon > 0$ into a number $\delta = \omega_f(\varepsilon) > 0$ for which $d(x, x') \leq \delta$ implies $|f(x) - f(x')| \leq \varepsilon$.

Comments. One can easily check that an interval $[\underline{a}, \overline{a}]$ with computable endpoints are effectively compact subsets of $\mathbb{R}$: as $S^{(k)}$, we can take, e.g., points $\underline{a} + \dfrac{i}{q(k)} \cdot (\overline{a} - \underline{a})$, $i = 0, \ldots, q(k)$, for a sufficiently large $q(k)$. Similarly, a multi-D box $[\underline{a}_1, \overline{a}_1] \times \ldots \times [\underline{a}_m, \overline{a}_m]$ with computable endpoints $\underline{a}_i$ and $\overline{a}_i$ is an effectively compact subset of $\mathbb{R}^m$.

One can also easily check that a Cartesian product $X = X_1 \times \ldots \times X_m$ of effectively compact sets, with a metric $d(x, y) = \max_i d(x_i, y_i)$, is also effectively compact: if $X_i^{(k)}$ are 2^{-k}-nets for the space X_i, then the finite set $X^{(k)} = X_1^{(k)} \times \ldots \times X_m^{(k)}$ is a 2^{-k}-net for X.

Operations on computable functions. In the proof of our main result, we will need the following properties of computable functions:

- if $f(x)$ is a computable function and c is a computable number, then the function $c \cdot f(x)$ is also computable;
- if $f(x)$ and $g(x)$ are computable functions, then the difference $f(x) - g(x)$, the minimum $\min(f(x), g(x))$ and $\max(f(x), g(x))$ are also computable.

These results are proven, e.g., in [Pour-El and Richards 1989] and in [Weihrauch 2000] (Theorems 6.2.1, 6.2.4, 6.2.7, and 6.2.9 and Corollary 6.2.5).

Comment. Strictly speaking, in [Weihrauch 2000], these results are only proven for functions on $\mathbb{R}^n$, but the proofs can be almost verbatim applied to the more general case of effectively compact metric spaces; see, e.g., Section 8.1 from [Weihrauch 2000], see also [Bridges and Popa 2003] and [Weihrauch and Grubba 2009].

Minimum and maximum over an effectively compact set. We will also need an auxiliary result that if $f : X \times Y \to \mathbb{R}$ is a computable function and X and Y are effectively compact metric spaces, then the functions $\min\limits_{x \in X} f(x, y)$ and $\max\limits_{x \in X} f(x, y)$ are also computable. Let us prove this result.

Minimum over an effectively compact set: computability. To compute the value

$$h(y) = \min_{x \in X} f(x, y)$$

with a given accuracy $\varepsilon > 0$, we:

- take $\delta = \omega_f\left(\dfrac{\varepsilon}{2}\right)$,
- find a δ-net $x^{(1)}, \ldots, x^{(m)}$ for the constructive set X,
- compute the values $f\left(x^{(i)}, y\right)$ with accuracy $\dfrac{\varepsilon}{2}$, resulting in approximate values $\widetilde{f}\left(x^{(1)}, y\right), \ldots, \widetilde{f}\left(x^{(m)}, y\right)$, and
- compute $\widetilde{h}(y) \stackrel{\text{def}}{=} \min\left(\widetilde{f}\left(x^{(1)}, y\right), \ldots, \widetilde{f}\left(x^{(m)}, y\right)\right)$.

Let us show that this value is indeed an ε-approximation to $h(y)$.

Indeed, from the fact that each value $\widetilde{f}\left(x^{(i)}, y\right)$ is an $(\varepsilon/2)$-approximation to $f\left(x^{(i)}, y\right)$, we conclude that $f\left(x^{(i)}, y\right) \le \widetilde{f}\left(x^{(i)}, y\right) + \dfrac{\varepsilon}{2}$ for all $i = 1, \ldots, m$.

Thus, the smallest of the left-hand sides is smaller than or equal to the smallest of the right-hand sides:

$$\min_i \left(f\left(x^{(i)}, y\right)\right) \le \min_i \left(\widetilde{f}\left(x^{(i)}, y\right) + \frac{\varepsilon}{2}\right) = \min_i \widetilde{f}\left(x^{(i)}, y\right) + \frac{\varepsilon}{2} = \widetilde{h}\,(y) + \frac{\varepsilon}{2}.$$

Since $\min\limits_i \left(f\left(x^{(i)}, y\right)\right) \ge \min\limits_{x \in X} f\,(x, y) = h\,(y)$, we thus conclude that

$$h\,(y) \le \widetilde{h}\,(y) + \frac{\varepsilon}{2} < \widetilde{h}\,(y) + \varepsilon. \tag{13}$$

Vice versa, since X is a compact, the minimum $h\,(y) = \min\limits_{x \in X} f\,(x, y)$ of the function $f\,(x, y)$ is attained for some $x_0 \in X$: $f\,(x_0, y) = h\,(y)$. Since the values $x^{(1)}, \ldots, x^{(m)}$ form a δ-net, there exists an i for which $d\left(x_0, x^{(i)}\right) \le \delta$. Due to the choice of $\delta = \omega_f\left(\dfrac{\varepsilon}{2}\right)$, this implies $\left|f\,(x_0, y) - f\left(x^{(i)}, y\right)\right| \le \dfrac{\varepsilon}{2}$, hence

$$f\left(x^{(i)}, y\right) \le f\,(x_0, y) + \frac{\varepsilon}{2} = h\,(y) + \frac{\varepsilon}{2}.$$

Since $\widetilde{f}\left(x^{(i)}, y\right)$ is an $(\varepsilon/2)$-approximation to $f\left(x^{(i)}, y\right)$, we conclude that

$$\widetilde{f}\left(x^{(i)}, y\right) \le f\left(x^{(i)}, y\right) + \frac{\varepsilon}{2}$$

hence

$$\widetilde{f}\left(x^{(i)}, y\right) \le \left(h\,(y) + \frac{\varepsilon}{2}\right) + \frac{\varepsilon}{2} = h\,(y) + \varepsilon.$$

Since one of the values $\widetilde{f}\left(x^{(i)}, y\right)$ does not exceed $h\,(y) + \varepsilon$, the smallest $\widetilde{h}\,(y)$ of these values also does not exceed $h\,(y) + \varepsilon$: $\widetilde{h}\,(y) \le h\,(y) + \varepsilon$. Together with (13), this implies that $|\widetilde{h}\,(y) - h\,(y)| \le \varepsilon$.

Minimum over an effectively compact set: modulus of continuity. Let us show that for $h\,(y) = \min\limits_{x \in X} f\,(x, y)$, we can take $\omega_h\,(\varepsilon) = \omega_f\,(\varepsilon)$.

Let us show that if $d\,(y, y') \le \delta = \omega_f\,(\varepsilon)$, then $h\,(y) \le h\,(y') + \varepsilon$. Indeed, since X is a compact set and f is a continuous function, there exists a value x_0 for which $h\,(y') = \min\limits_{x \in X} f\,(x, y') = f\,(x_0, y')$. Here,

$$d\,((x_0, y)\,, (x_0, y')) = \max\,(0, d\,(y, y')) = d\,(y, y') \le \delta.$$

Due to our choice of $\delta = \omega_f\,(\varepsilon)$, we have hence $|f\,(x_0, y) - f\,(x_0, y')\,| \le \varepsilon$, hence $f\,(x_0, y) \le f\,(x_0, y') + \varepsilon = h\,(y') + \varepsilon$. Since $h\,(y) = \min\limits_{x \in X} f\,(x, y) \le f\,(x_0, y)$, we thus conclude that $h\,(y) \le h\,(y') + \varepsilon$.

Similarly, we can prove that $h\,(y') \le h\,(y) + \varepsilon$, so indeed $|h\,(y) - h\,(y')\,| \le \varepsilon$.

Maximum over an effectively compact set. The value $h(y) = \max\limits_{x \in X} f(x, y)$ can be written as $h(y) = -\min\limits_{x \in X}(-f(x, y))$ and can thus be approximated using the above algorithm for the minimum over an effectively compact set.

Towards the main result. Now, we have listed all the desired definitions and the auxiliary results, so we are ready to start the formulation and the analysis of our problem – of computing different types of sets.

4 Definitions and the main result

Discussion. All above definitions of decision-related properties were formed in a similar manner: we had basic expressions of the type $a \geq b$ or $a > b$ or $a = b$, and we combined them by using logical connectives $\vee$, &, $\neg$ ("not"), $\rightarrow$, and quantifiers $\exists t$ and $\forall t$. An additional restriction is that the basic expressions contained one unknown function: we had expressions of the type $f_i(\ldots) \geq f_i(\ldots)$ or $f_i(\ldots) \geq 0$ – but not, e.g., expressions of the type $f_i \geq f_j$ for $i \neq j$.

To analyze such expressions, let us perform some simplifications. First, we can eliminate equalities by replacing each equality $a = b$ with an equivalent combination of two inequalities $(a \geq b)$ & $(b \geq a)$.

Second, we can replace each implication $A \rightarrow B$ with an equivalent logical formula $B \vee \neg A$, thus eliminating all implication symbols too.

Third, we can move negations inside the formulas, so that negations appear only in front of basic expressions:

- we replace $\neg(A \,\&\, B)$ with an equivalent formula $\neg A \vee \neg B$;
- we replace $\neg(A \vee B)$ with an equivalent formula $\neg A \,\&\, \neg B$;
- we replace $\neg(\exists t\, A(t))$ with an equivalent formula $\forall t\ (\neg A(t))$; and
- we replace $\neg(\forall t\, A(t))$ with an equivalent formula $\exists t\ (\neg A(t))$.

Now, when negations are only at basic inequality expressions:

- we replace $\neg(a \geq b)$ with $b > a$ and
- we replace $\neg(a > b)$ with $b \geq a$.

Thus, we eliminate all the negation symbols as well.

Thus, we have basic expressions of the type $a \geq b$ and $a > b$, and a general formula can be obtained by using $\vee$, &, and quantifiers. Finally, as we have mentioned in Section 1, when we have a strict inequality, we replace it with a non-strict one anyway. Thus, we arrive at the following definition.

Definition 1. Let $X_1, \ldots, X_m$ be effectively compact sets, and let f_i, $i = 1, \ldots, n$, be computable functions from $X = X_1 \times \ldots \times X_m$ to the set $\mathbb{R}$ of real numbers.

– By a f_i-*expression*, we mean a formula of the type $f_i(x_1, \ldots, x_m) \geq 0$ or of the type $f_i(x_1, \ldots, x_m) \geq f_i(x'_1, \ldots, x'_m)$, where x_i and x'_i are variables (possibly coinciding).

– By a *decision-related property*, we mean an expression that is obtained from f_i-expressions by using $\vee$, &, and quantifiers $\forall t_{t \in X_i}$ and $\exists t_{t \in X_i}$.

– For each decision-related property P with free variables $z = (z_1, \ldots)$, by a *decision-related set* $S(P)$, we mean the set of all the tuples z that satisfy the property P.

Comment. One can easily check that all the above examples are particular cases of this general definition.

Definition 2. Let $X_1, \ldots, X_m$ be effectively compact sets, let f_i, $i = 1, \ldots, n$, be computable functions from $X = X_1 \times \ldots \times X_m$ to the set $\mathbb{R}$ of real numbers, and let $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_n)$ be a tuple of positive real numbers.

– By an ε-*version* of a decision-related property P, we mean the formula P_ε in which

  • each f_i-expression of the type $f_i(x_1, \ldots, x_m) \geq 0$ is replaced by

$$f_i(x_1, \ldots, x_m) \geq -\varepsilon_i; \quad \text{and}$$

  • each f_i-expression of the type $f_i(x_1, \ldots, x_m) \geq f_i(x'_1, \ldots, x'_m)$ is replaced by $f_i(x_1, \ldots, x_m) \geq f_i(x'_1, \ldots, x'_m) - 2 \cdot \varepsilon_i$.

– We say that a tuple z ε-*satisfies* the property P if it satisfies the formula P_ε.

– The set of all the tuples that ε-satisfy the property P will be denoted by $S_\varepsilon(P)$.

Example: inequality. In particular, an inequality $f_i \geq 0$ is transformed into $f_i \geq -\varepsilon_i$. For the constraint example, the ε-modification can be justified by the following simple result:

Definition 3. Let $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_n)$ be a tuple of positive real numbers.

– We say that the functions $f_i(x)$ and $g_i(x)$ are ε_i-*close* if $|f_i(x) - g_i(x)| \leq \varepsilon_i$ for all x.

– We say that a tuple of functions $f(x) = (f_1(x), \ldots, f_n(x))$ is ε-*close* to a tuple $g(x) = (g_1(x), \ldots, g_n(x))$ if $|f_i(x) - g_i(x)| \leq \varepsilon_i$ for all x and i.

Proposition 4. *Let $f_i(x)$ be a function, $\varepsilon_i > 0$ be a number, and x^* be a value. Then, the following two conditions are equivalent to each other:*

– *the inequality $f_i(x^*) \geq -\varepsilon_i$ holds for a given function $f_i(x)$;*

– *the inequality $g_i(x^*) \geq 0$ holds for some function g_i which is ε_i-close to $f_i(x)$.*

Example: equality. The original equality $f_i = 0$ is first represented as two inequalities $f_i \geq 0$ and $f_j \geq 0$, where $f_j \stackrel{\text{def}}{=} -f_i$. For the computable function $f_j = -f_i$, the accuracy ε_j with which we know f_j is the same as the accuracy ε_i with which we know f_i: indeed, $|f_j - \widetilde{f}_j| = |-f_i - \left(-\widetilde{f}_i\right)| = |f_i - \widetilde{f}_i|$. Thus, the two inequalities get replaced with two modified inequalities $f_i \geq -\varepsilon_i$ and $-f_i \geq -\varepsilon_i$. The second modified inequality is equivalent to $f_i \leq \varepsilon$ and thus, this system of two modified inequalities is equivalent to $|f_i| \leq \varepsilon_i$. This is a reasonable ε_i-approximate analogue of the original equality $f_i = 0$.

Proposition 5. *Let $f_i(x)$ be a function, $\varepsilon_i > 0$ be a number, and x^* be a value. Then, the following two conditions are equivalent to each other:*

– *the inequality $|f_i(x^*)| \leq \varepsilon_i$ holds for a given function $f_i(x)$;*

– *the equality $g_i(x^*) = 0$ holds for some function g_i which is ε_i-close to $f_i(x)$.*

Equalities can be safely added. In general, our main theorem (see below) remains true if we allow equalities $f_i(\cdot) = f_i(\cdot)$ and $f_i(\cdot) = 0$ as f_i-expressions, and say that these equalities are ε_i-satisfied if, correspondingly, $|f_i(\cdot) - f_i(\cdot)| \leq 2 \cdot \varepsilon_i$ and $|f_i(\cdot)| \leq \varepsilon_i$.

Examples: optimality, Pareto optimality, and Nash equilibrium. For optimality and Pareto optimality, results similar to Propositions 4 and 5 have been proven in [G.-Toth and Kreinovich 2009]. A similar result holds for the Nash equilibrium:

Proposition 6. *Let $f : X_1 \times \ldots \times X_m \to \mathbb{R}$ be a tuple of functions, ε be a tuple of numbers, and $x_0 = (x_1, \ldots, x_m) \in X_1 \times \ldots \times X_m$ be a value. Then, the following two conditions are equivalent to each other:*

– *the value x^* is a ε-Nash equilibrium, i.e., for every i and for every $x_i \in X_i$, we have*

$$f_i\left(x_1^*, \ldots, x_{i-1}^*, x_i^*, x_{i+1}^*, \ldots, x_m^*\right) \geq$$
$$f_i\left(x_1^*, \ldots, x_{i-1}^*, x_i, x_{i+1}^*, \ldots, x_m^*\right) - 2 \cdot \varepsilon_i; \tag{14}$$

– *for some tuple of functions $g(x)$ which is ε-close to $f(x)$, the value x^* is a Nash equilibrium, i.e., for every i and for every $x_i \in X_i$, we have:*

$$g_i\left(x_1^*, \ldots, x_{i-1}^*, x_i^*, x_{i+1}^*, \ldots, x_m^*\right) \geq g_i\left(x_1^*, \ldots, x_{i-1}^*, x_i, x_{i+1}^*, \ldots, x_m^*\right). \quad (15)$$

In general, negation cannot be safely added. In the original transformation of general formulas, we eliminated equality and we eliminated negation. Let us show that while equality can be easily added to this formulation, it is not possible to consistently add negation. Indeed, let us consider a condition $\neg(f_i > 0)$.

– According to the current methodology, we replace $\neg(f_i > 0)$ with $-f_i \geq 0$ and then replace it with a modified inequality $-f_i \geq -\varepsilon_i$, i.e., $f_i \leq \varepsilon$.

– On the other hand, if we could simply allow negations in the above definition, then, to modify the above formula, we would replace $f_i > 0$ with $f_i \geq -\varepsilon$. Then, the original formula $\neg(f_i > 0)$ would be replaced with a modified formula $\neg(f_i \geq -\varepsilon)$, i.e., $f_i < -\varepsilon$.

One can see that the resulting formulas $f_i \leq \varepsilon$ and $f_i < -\varepsilon$ are indeed different. Moreover, the second formula $f_i < -\varepsilon$ does not allow the possibility $f_i = 0$ which is perfectly in line with the original formula $\neg(f_i > 0)$.

Because of the difficulty with negation, the ε-modification is not always meaningful. In Section 1, we gave examples of several problems for which ε-modifications make sense. It should be mentioned, however, that because of the above difficulty with negation, not all formulas get a meaningful modification. As an example, let us take the requirement $\forall x\ (f_i(x) = 0 \rightarrow f_j(x) = 0)$.

Intuitively, if $f_i(x)$ is known only up to some precision, then it is always consistent to assume $f_i(x) \neq 0$. Thus, in the ε-approximation, the given implication, the antecedent can always be assumed as false, so the implication is always true.

This intuitive conclusion can be confirmed if we follow the above methodology. First, we eliminate equality and transform this requirement into

$$\forall x\ ((f_i(x) \geq 0 \,\&\, -f_i(x) \geq 0) \rightarrow (f_j(x) \geq 0 \,\&\, -f_j(x) \geq 0)).$$

Second, we eliminate implication, resulting in

$$\forall x\ (\neg(f_i(x) \geq 0 \,\&\, -f_i(x) \geq 0) \vee (f_j(x) \geq 0 \,\&\, -f_j(x) \geq 0)).$$

Third, we move negations inside, resulting in

$$\forall x\ (\neg(f_i(x) \geq 0) \vee \neg(-f_i(x) \geq 0) \vee (f_j(x) \geq 0 \,\&\, -f_j(x) \geq 0))$$

and in

$$\forall x\ ((-f_i(x) > 0) \vee (f_i(x) > 0) \vee (f_j(x) \geq 0 \,\&\, -f_j(x) \geq 0)).$$

The corresponding ε-modification has the form

$$\forall x\ ((-f_i(x) \geq -\varepsilon_i) \vee (f_i(x) \geq -\varepsilon_i) \vee (f_j(x) \geq -\varepsilon_j \,\&\, -f_j(x) \geq -\varepsilon_j)).$$

This modified condition is, however, meaningless because it is satisfied for all possible values x. Indeed:

– If $f_i(x) \geq -\varepsilon_i$, then the second condition $f_i(x) \geq -\varepsilon_i$ from the disjunction is satisfied.

– Otherwise, if $f_i(x) < -\varepsilon_i$, then $-f_i(x) > \varepsilon_i$ and since $\varepsilon_i > -\varepsilon_i$, the first condition $-f_i(x) \geq -\varepsilon_i$ from the disjunction is satisfied.

Monotonicity of ε-satisfaction. In our proofs and algorithms, we will use the fact that if $\underline{\varepsilon}_i \leq \overline{\varepsilon}_i$ for each i, then:

– the inequality $f_i(x_1, \ldots, x_m) \geq -\underline{\varepsilon}_i$ implies $f_i(x_1, \ldots, x_m) \geq -\overline{\varepsilon}_i$, and

– the inequality $f_i(x_1, \ldots, x_m) \geq f_i(x'_1, \ldots, x'_m) - 2 \cdot \underline{\varepsilon}_i$ implies $f_i(x_1, \ldots, x_m) \geq f_i(x'_1, \ldots, x'_m) - 2 \cdot \overline{\varepsilon}_i$.

Logical operations $\vee$, &, $\forall t_{t \in X_i}$, and $\exists t_{t \in X_i}$ are monotonic in terms of implication: e.g., if A implies A' and B implies B', then $A \vee B$ implies $A' \vee B'$.

We can therefore conclude that $P_{\underline{\varepsilon}}$ implies $P_{\overline{\varepsilon}}$, i.e., that $S_{\underline{\varepsilon}}(P) \subseteq S_{\overline{\varepsilon}}(P)$.

As usual, by $B_\delta(\ell) \stackrel{\text{def}}{=} \{z : d(\ell, z) \leq \delta\}$, we denote the ball of radius δ with a center at the point ℓ.

Theorem 7. *There exists an algorithm that, given a decision-related property P and two tuples $\underline{\varepsilon}$ and $\overline{\varepsilon}$ of rational numbers for which $0 < \underline{\varepsilon}_i < \overline{\varepsilon}_i$, produces a finite list of elements L and a rational number $\delta > 0$ with the following three properties:*

– *Every tuple z^* that $\underline{\varepsilon}$-satisfies the property P is δ-close to some element from the list L.*

– *Every tuple that is δ-close to some element of the list L $\overline{\varepsilon}$-satisfies the property P.*

– *The set $\bigcup_{\ell \in L} B_\delta(\ell)$ of all tuples which are δ-close to some element of the list L is an effectively compact set that satisfies the property*

$$S_{\underline{\varepsilon}}(P) \subseteq \bigcup_{\ell \in L} B_\delta(\ell) \subseteq S_{\overline{\varepsilon}}(P).$$

Comment 1. The list L and the accuracy δ provide a description of the desired decision-related set S – as the set of all the elements which are δ-close to one of the elements from the given list, i.e., as the union of the corresponding balls $B_\delta(\ell)$.

It is worth mentioning that while in some decision-related problems like optimization, the solution always exists, in other problems – such as constraint satisfaction – it is possible that there are no solutions. In such cases, the above algorithm will produce an empty list $L = \emptyset$.

Comment 2. In this paper, we mainly followed notations from [Pour-El and Richards 1989], because one of our main objectives is to provide algorithms of use to people working in optimization or game theory, and we believe that these notations are the easiest to explain to these people. Alternatively, we could use the toolkit of TTE as described in [Weihrauch 2000]. This would have allowed us to simplify the proofs by using known theorems about computable metric spaces and computably compact sets.

5 Proofs

Proof of Proposition 4. If $g_i(x^*) \geq 0$ and g_i is f_i is ε_i-close to f_i, then $|f_i(x^*) - g_i(x^*)| \leq \varepsilon_i$ hence $f_i(x^*) \geq g_i(x^*) - \varepsilon_i$ and so $f_i(x^*) \geq -\varepsilon_i$.

Vice versa, if $f_i(x^*) \geq -\varepsilon_i$, then for $g_i(x) \stackrel{\text{def}}{=} f_i(x) + \varepsilon_i$, we have $g_i(x^*) \geq 0$ and at the same time $|f_i(x) - g_i(x)| = \varepsilon_i \leq \varepsilon_i$, so g_i and f_i are indeed ε_i-close.

Proof of Proposition 5. If $g_i(x^*) = 0$ and g_i is f_i is ε_i-close to f_i, then $|f_i(x^*) - g_i(x^*)| \leq \varepsilon_i$ hence $|f_i(x^*)| \leq \varepsilon_i$.

Vice versa, if $|f_i(x^*)| \leq \varepsilon_i$, then for $g_i(x) \stackrel{\text{def}}{=} f_i(x) - f_i(x^*)$, we have $g_i(x^*) = 0$ and at the same time $|f_i(x) - g_i(x)| = |f_i(x^*)| \leq \varepsilon_i$, so g_i and f_i are indeed ε_i-close.

Proof of Proposition 6. Let us first assume that f and g are ε-close and the condition (15) holds, i.e.,

$$g_i\left(x_1^*, \ldots, x_{i-1}^*, x_i^*, x_{i+1}^*, \ldots, x_m^*\right) \geq g_i\left(x_1^*, \ldots, x_{i-1}^*, x_i, x_{i+1}^*, \ldots, x_m^*\right).$$

Then, due to $|f_i(\cdot) - g_i(\cdot)| \leq \varepsilon_i$, we have

$$f_i\left(x_1^*, \ldots, x_{i-1}^*, x_i^*, x_{i+1}^*, \ldots, x_m^*\right) \geq g_i\left(x_1^*, \ldots, x_{i-1}^*, x_i^*, x_{i+1}^*, \ldots, x_m^*\right) - \varepsilon_i \geq$$

$$g_i\left(x_1^*, \ldots, x_{i-1}^*, x_i, x_{i+1}^*, \ldots, x_m^*\right) - \varepsilon_i \geq f_i\left(x_1^*, \ldots, x_{i-1}^*, x_i, x_{i+1}^*, \ldots, x_m^*\right) - 2\cdot\varepsilon_i,$$

i.e., the desired inequality (14).

Vice versa, let us assume that the inequality (14) holds. Let us define the new functions g_i as follows:

– for the given x^*, we take $g_i(x^*) = f_i(x^*) + \varepsilon_i$;

– for all $x \neq x^*$, we take $g_i(x) = f_i(x) - \varepsilon_i$.

Then, every condition of the type

$$f_i(x^*) = f_i\left(x_1^*, \ldots, x_{i-1}^*, x_i^*, x_{i+1}^*, \ldots, x_m^*\right) \geq$$

$$f_i\left(x_1^*, \ldots, x_{i-1}^*, x_i, x_{i+1}^*, \ldots, x_m^*\right) - 2 \cdot \varepsilon_i$$

with $x_i \neq x_i^*$ implies

$$g_i(x^*) = f_i(x^*) + \varepsilon_i \geq \left(f_i\left(x_1^*, \ldots, x_{i-1}^*, x_i, x_{i+1}^*, \ldots, x_m^*\right) - 2 \cdot \varepsilon_i\right) + \varepsilon_i =$$

$$f_i\left(x_1^*, \ldots, x_{i-1}^*, x_i, x_{i+1}^*, \ldots, x_m^*\right) - \varepsilon_i = g_i\left(x_1^*, \ldots, x_{i-1}^*, x_i, x_{i+1}^*, \ldots, x_m^*\right).$$

Thus, for the tuple of functions $g(x)$, the value x^* is indeed a Nash equilibrium. The proposition is proven.

Comment. Please note that the functions $g_i(x)$ defined in the above proof are not continuous and thus, not computable. This is OK since the formulation of the proposition does not require computability of the functions $g_i(x)$.

Proof of Theorem 7.

1°. Let us denote $\alpha_0 \stackrel{\text{def}}{=} \max_i \left(\frac{\underline{\varepsilon}_i}{\overline{\varepsilon}_i}\right)$. Then, for every i, we have $\frac{\underline{\varepsilon}_i}{\overline{\varepsilon}_i} \leq \alpha_0$ hence $\underline{\varepsilon}_i \leq \alpha_0 \cdot \overline{\varepsilon}_i$. Due to the monotonicity property of the set $S_\varepsilon(P)$, we thus have

$$S_{\underline{\varepsilon}}(P) \subseteq S_{\alpha_0 \cdot \overline{\varepsilon}}(P). \tag{16}$$

We will show how to compute a finite set L and two rational values $\underline{\delta} < \overline{\delta}$ for which

$$S_{\alpha_0 \cdot \overline{\varepsilon}}(P) \subseteq \bigcup_{\ell \in L} B_{\underline{\delta}}(\ell) \subseteq \bigcup_{\ell \in L} B_{\overline{\delta}}(\ell) \subseteq S_{\overline{\varepsilon}}(P). \tag{17}$$

Due to (16), this implies that

$$S_{\underline{\varepsilon}}(P) \subseteq \bigcup_{\ell \in L} B_{\underline{\delta}}(\ell) \subseteq \bigcup_{\ell \in L} B_{\overline{\delta}}(\ell) \subseteq S_{\overline{\varepsilon}}(P). \tag{18}$$

2°. The sets $S_{\alpha_0 \cdot \overline{\varepsilon}}(P)$ and $S_{\overline{\varepsilon}}(P)$ are particular cases of the general set $S_{\alpha \cdot \overline{\varepsilon}}(P)$ corresponding to $\alpha = \alpha_0 < 1$ and to $\alpha = 1$. Let us therefore find a general description of the set $S_{\alpha \cdot \overline{\varepsilon}}(P)$. We will do it by following the structure of the properties: we start with the basic statements, and we show how this description can be extended to statements obtained by using logical connectives and quantifiers.

2.1°. A basic statement $f_i(x_1,\ldots,x_m) \ge -\alpha\cdot\overline{\varepsilon}_i$ can be equivalently reformulated as $g_i(x_1,\ldots,x_m) \ge -\alpha$, where

$$g_i(x_1,\ldots,x_m) \stackrel{\text{def}}{=} \frac{f_i(x_1,\ldots,x_m)}{\overline{\varepsilon}}.$$

Since f_i is a computable function and $\overline{\varepsilon}$ is a computable number, the function $g_i(x_1,\ldots,x_m)$ is also computable.

2.2°. A basic statement $f_i(x_1,\ldots,x_m) - f(x'_1,\ldots,x'_m) \ge -2\cdot\alpha\cdot\overline{\varepsilon}_i$ can be equivalently reformulated as $h_i(x_1,\ldots,x_m) \ge -\alpha$, where

$$h_i(x_1,\ldots,x_m) \stackrel{\text{def}}{=} \frac{f_i(x_1,\ldots,x_m) - f(x'_1,\ldots,x'_m)}{2\cdot\overline{\varepsilon}}.$$

Since f_i is a computable function and $\overline{\varepsilon}$ is a computable number, the function $h_i(x_1,\ldots,x_m)$ is also computable.

2.3°. Let us assume that we have a property $P\,\&\,P'$, where properties P and P' have already been represented in the form $f(z) \ge -\alpha$ and $f'(z) \ge -\alpha$ for computable functions $f(z)$ and $f'(z)$. Then, the property $P\,\&\,P'$ can be represented as $g(z) \ge -\alpha$, where $g(z) \stackrel{\text{def}}{=} \min(f(z), f'(z))$.

Since both functions $f(z)$ and $f'(z)$ are computable, we can use a result from the previous section to conclude that the new function $g(z)$ is also computable.

2.4°. Let us assume that we have a property $P \vee P'$, where properties P and P' have already been represented in the form $f(z) \ge -\alpha$ and $f'(z) \ge -\alpha$ for computable functions $f(z)$ and $f'(z)$. Then, the property $P\,\&\,P'$ can be represented as $g(z) \ge -\alpha$, where $g(z) \stackrel{\text{def}}{=} \max(f(z), f'(z))$.

Since both functions $f(z)$ and $f'(z)$ are computable, we can use a result from the previous section to conclude that the new function $g(z)$ is also computable.

2.5°. Let us assume that we have a property $\exists t_{t\in X_i} P(t,z)$, where the property $P(t,z)$ has already been represented in the form $f(t,z) \ge -\alpha$ for a computable function $f(t,z)$. Then, the property $\exists t_{t\in X_i} P(t,z)$ can be represented as $g(z) \ge -\alpha$, where $g(z) \stackrel{\text{def}}{=} \max\limits_{t\in X_i} f(t,z)$.

Since the function $f(t,z)$ is computable, we can use a result from the previous section to conclude that the new function $g(z)$ is also computable.

2.6°. Let us assume that we have a property $\forall t_{t\in X_i} P(t,z)$, where the property $P(t,z)$ has already been represented in the form $f(t,z) \ge -\alpha$ for a computable function $f(t,z)$. Then, the property $\forall t_{t\in X_i} P(t,z)$ can be represented as $g(z) \ge -\alpha$, where $g(z) \stackrel{\text{def}}{=} \min\limits_{t\in X_i} f(t,z)$.

Since the function $f(t,z)$ is computable, we can use a result from the previous section to conclude that the new function $g(z)$ is also computable.

2.7°. Thus, an $(\alpha\cdot\overline{\varepsilon})$-version of each decision-related property can be reformulated in the equivalent form $f(z)\ge -\alpha$ for an appropriate computable function $f(z)$. Thus, $S_{\alpha\cdot\overline{\varepsilon}}(P)=\{z: f(z)\ge -\alpha\}$.

3°. Since the function $f(z)$ corresponding to the desired decision-related property P is computable, there exists a corresponding algorithm for computing a modulus of continuity $\omega(\varepsilon)$.

Let us take, as ε, the rational number $\varepsilon=\dfrac{1-\alpha_0}{4}$, and let us take $\overline{\delta}=\omega_f(\varepsilon)$, and $\underline{\delta}=\dfrac{\overline{\delta}}{2}$.

Each z is a tuple of elements from computable (compact) sets X_i. Thus, the set Z of all possible values z is a Cartesian product of effectively compact sets and thus, an effectively compact set itself. In particular, this means that we can algorithmically compute a $\underline{\delta}$-net $z^{(1)},\ldots,z^{(M)}$ for the set Z.

Since the function $f(z)$ is computable, we can compute all the values $f\left(z^{(i)}\right)$ with arbitrary accuracy. Let us denote the rational number resulting from computing $f\left(z^{(i)}\right)$ with accuracy ε by $\widetilde{f}\left(z^{(i)}\right)$. As the desired set L, we will now take

$$L=\left\{z^{(i)}:\widetilde{f}\left(z^{(i)}\right)\ge -\alpha_0-2\cdot\varepsilon\right\}. \tag{19}$$

(Both compared values $\widetilde{f}\left(z^{(i)}\right)$ and $-\alpha_0-2\cdot\varepsilon$ are rational, so we can algorithmically check the above inequality.)

Let us prove that for these $\underline{\delta}$, $\overline{\delta}$, and L, we indeed have the property (17).

3.1°. Let us first prove that $S_{\alpha_0\cdot\overline{\varepsilon}}(P)\subseteq\bigcup\limits_{\ell\in L}B_{\underline{\delta}}(\ell)$, i.e., that if $f(z)\ge-\alpha_0$, then there exists an $z^{(i)}\in L$ for which $d\left(z,z^{(i)}\right)\le\underline{\delta}$.

Indeed, since the values $z^{(i)}$ form a $\underline{\delta}$-net for the set Z, there exists an i for which $d\left(z,z^{(i)}\right)\le\underline{\delta}$. All we need to prove now is that $z^{(i)}\in L$, i.e., by definition of the set L, that $\widetilde{f}\left(z^{(i)}\right)\ge-\alpha_0-2\cdot\varepsilon$.

Indeed, because of our choice of $\underline{\delta}$ as $\dfrac{1}{2}\cdot\omega_f(\varepsilon)$, the condition $d\left(z,z^{(i)}\right)\le\underline{\delta}=\dfrac{1}{2}\cdot\omega_f(\varepsilon)$ implies that $d\left(z,z^{(i)}\right)\le\omega_f(\varepsilon)$. By definition of the modulus of continuity, this implies that $|f(z)-f\left(z^{(i)}\right)|\le\varepsilon$. Thus, $f\left(z^{(i)}\right)\ge f(z)-\varepsilon$ and since $f(z)\ge-\alpha_0$, we conclude that $f\left(z^{(i)}\right)\ge-\alpha_0-\varepsilon$.

Now, by definition of $\widetilde{f}\left(z^{(i)}\right)$, we have $|\widetilde{f}\left(z^{(i)}\right)-f\left(z^{(i)}\right)|\le\varepsilon$ hence $\widetilde{f}\left(z^{(i)}\right)\ge f\left(z^{(i)}\right)-\varepsilon$. We already know that $f\left(z^{(i)}\right)\ge-\alpha_0-\varepsilon$, so we conclude that $\widetilde{f}\left(z^{(i)}\right)\ge-\alpha_0-2\cdot\varepsilon$ hence $z^{(i)}\in L$. The statement is proven.

3.2°. Let us now prove that $\bigcup\limits_{\ell\in L}B_{\overline{\delta}}(\ell)\subseteq S_{\overline{\varepsilon}}(P)$, i.e., that if for some $z\in Z$ and i for which $z^{(i)}\in L$, we have $d\left(z,z^{(i)}\right)\le\overline{\delta}$, then $f(z)\ge-1$.

Indeed, $z^{(i)} \in L$ means that $\widetilde{f}\left(z^{(i)}\right) \geq -\alpha_0 - 2 \cdot \varepsilon$. Since $\widetilde{f}\left(z^{(i)}\right)$ is an ε-approximation to $f\left(z^{(i)}\right)$, we have $f\left(z^{(i)}\right) \geq \widetilde{f}\left(z^{(i)}\right) - \varepsilon$ and therefore, $\widetilde{f}\left(z^{(i)}\right) \geq -\alpha_0 - 3 \cdot \varepsilon$.

Due to $d\left(z, z^{(i)}\right) \leq \overline{\delta} = \omega_f(\varepsilon)$, we have $|f(z) - f\left(z^{(i)}\right)| \leq \varepsilon$ hence $f(z) \geq f\left(z^{(i)}\right) - \varepsilon$ and thus, $f(z) \geq -\alpha_0 - 4 \cdot \varepsilon$. By definition of ε, this means that $f(z) \geq -1$, i.e., that indeed $\bigcup\limits_{\ell \in L} B_{\overline{\delta}}(\ell) \subseteq S_{\overline{\varepsilon}}(P)$

The property (17) is proven.

4°. To complete the proof, we must show that there exists a δ for which $\underline{\delta} \leq \delta \leq \overline{\delta}$ – and for which therefore

$$\bigcup_{\ell \in L} B_{\underline{\delta}}(\ell) \subseteq \bigcup_{\ell \in L} B_{\delta}(\ell) \subseteq \bigcup_{\ell \in L} B_{\overline{\delta}}(\ell),$$

for which the union $\bigcup\limits_{\ell \in L} B_\delta(\ell)$ is an effectively compact set.

Indeed, for an arbitrary δ, the condition that $z \in \bigcup\limits_{\ell \in L} B_\delta(\ell)$ means that there exists $\ell \in L$ for which $d(z, \ell) \leq \delta$. This condition is equivalent to $g(z) \leq \delta$, where $g(z) \stackrel{\text{def}}{=} \min\limits_{\ell \in L} d(z, \ell)$. Thus, $\bigcup\limits_{\ell \in L} B_\delta(\ell) = \{z : g(z) \leq \delta\}$.

According to the properties of the minima of computable functions, the function $g(z)$ is also a computable function on an effectively compact set. To complete our proof, it is now sufficient to show that for every computable function $g(z)$ and for every two computable numbers $\underline{\delta} < \overline{\delta}$, we can algorithmically find a value $\delta \in \left(\underline{\delta}, \overline{\delta}\right)$ for which the set $\{z : g(z) \leq \delta\}$ is an effectively compact set.

This result is, in effect, proven in [Bishop and Bridges 1985]. To be more precise, the result from [Bishop and Bridges 1985] is formulated in terms of a *constructive* function, i.e., a function which is defined only for computable inputs x, but one can easily check that the proof is applicable to *computable* functions as well.

For thus found value δ, the set $\bigcup\limits_{\ell \in L} B_\delta(\ell) = \{z : g(z) \leq \delta\}$ is effectively compact. The theorem is proven.

6 Computational complexity and feasibility of the resulting algorithms

Computational complexity: general case. Once we established that the algorithms exist, the natural next question is: how efficient are these algorithms? According to the proofs, the above algorithms require that we consider all the elements of the corresponding ε-net, its number of steps grows as the number of these elements. For an m-dimensional box this number is $\approx V/\varepsilon^m$, so it grows exponentially with the dimension m of the box.

This is, however, acceptable, since in general, the optimization problems are NP-hard [Kreinovich et al. 1998], and therefore, the worst-case exponential time is inevitable (unless, of course, it turns out that, contrary to the expectations of most computer scientists, P = NP and thus, all such problems can be solved in feasible (polynomial) time).

Implementation of the above algorithms: interval computations. The usual implementation of the above algorithms involve interval computations; see, e.g., [Jaulin et el. 2001, Moore at al. 2009].

Interval computations were originally designed to estimate the uncertainty of the result of data processing in situations in which we only know the upper bounds Δ on the measurement errors. In this case, based on the measurement result $\widetilde{x}$, we can only conclude that the actual (unknown) value x of the desired quantity is in the interval $[\widetilde{x} - \Delta, \widetilde{x} + \Delta]$.

In interval computations, at each intermediate stage of the computation, we have intervals of possible values of the corresponding quantities.

From interval computations to more sophisticated set computations. In interval computations, at every intermediate stage of the computations, we only keep the intervals of possible values of each quantity, but we do not keep the information about the relations between these quantities. As a result, we often have bounds with excess width.

To remedy this problem, in [Ceberio et al. 2006], [Ceberio et al. 2007], and [Kreinovich 2009], we proposed an extension of interval technique to *set computations*, where on each stage, in addition to intervals of possible values of the quantities, we also keep sets of possible values of pairs (triples, etc.). As a result, in several practical problems, such as

– estimating statistics (variance, correlation, etc.) under interval uncertainty, and

– solutions to ordinary differential equations (ODEs) with given accuracy,

this new formalism enables us to find estimates in feasible (polynomial) time; see [Ceberio et al. 2007] and [Kreinovich 2009].

Comment. The idea of using a grid to describe and compute sets is well known; see, e.g., [Weihrauch 2000] (Section 7.4), [Rettinger and Weihrauch 2003], and [Weihrauch 2003]. In particular, for the analysis of computational complexity, real numbers can be represented as $\sum a_n \cdot 2^{-n}$, with digits a_n from $\{-1, 0, 1\}$ (signed digit representation [Weihrauch 2000], Section 7.2). According to [Escardó 2009], sets can be described in terms of possible values of a_n.

Alternative approaches to efficient set representation and set computations can be found, e.g., in [Collins 2007] and [Collins 2007a].

Acknowledgments

This work was supported in part by the Polish Ministry of Science and Higher Education under grant N N514 416934, by NSF grant HRD-0734825, by Grant 1 T36 GM078000-01 from the National Institutes of Health, by CCA'2009 conference, and by Ettore Majorana Center for Scientific Culture, Erice, Italy.

We are very thankful to all the participants of CCA'2009 for valuable discussions, and to the anonymous referees for important suggestions.

References

[Brattka and Presser 2003] Brattka, V., Presser, G.: "Computability on subsets of metric spaces", Theoretical Computer Science 305 (2003) 43–76.

[Bishop and Bridges 1985] Bishop, E., Bridges, D. S.: "Constructive analysis", Springer-Verlag, Berlin-Heidelberg-New York, 1985.

[Bridges and Popa 2003] Bridges, D., Popa, G.: "Exact, continuous boundary crossings out of convex sets in R^N", The Quarterly Journal of Mathematics 54, 4 (2003) 391–398.

[Ceberio et al. 2006] Ceberio, M., Ferson, S., Kreinovich, V. et al.: "How To Take Into Account Dependence Between the Inputs: From Interval Computations to Constraint-Related Set Computations", In: Proc. 2nd Int'l Workshop on Reliable Engineering Computing, Savannah, Georgia, February 22–24, 2006, pp. 127–154; final version in Journal of Uncertain Systems 1, No. 1 (2007), 11–34.

[Ceberio et al. 2007] Ceberio, M., Kreinovich, V., Pownuk, A., Bede, B.: "From Interval Computations to Constraint-Related Set Computations: Towards Faster Estimation of Statistics and ODEs under Interval, p-Box, and Fuzzy Uncertainty", In: Melin, P., Castillo, O., Aguilar, L. T., Kacprzyk, J., Pedrycz, W.: (eds.), Foundations of Fuzzy Logic and Soft Computing, Proceedings of the World Congress of the International Fuzzy Systems Association IFSA'2007, Cancun, Mexico, June 18–21, 2007, Springer Lecture Notes on Artificial Intelligence, 2007, Vol. 4529, pp. 33–42.

[Collins 2007] Collins, P.: "Optimal semicomputable approximations to reachable and invariant sets", Theoretical Computer Science 41, No. 1 (2007), 33–48.

[Collins 2007a] Collins, P.: "Effective computations in linear systems", In: Computation and Logic in the Real World, Lecture Notes in Computer Science 4497 (2007), 169–178.

[Edalat and Heckmann 1998] Edalat, A., Heckmann, R.: "A computational model for metric spaces", Theoretical Computer Science 193 (1998) 53–73.

[Escardó 2009] Escardó, M.: "Theory and Practice of Higher-type Computation", In: Bauer, A., Dillhage, R., Hertling, P., Ko, K.-I., and Rettinger, R. (eds.), Proceedings of the Sixth International Conference on Computability and Complexity in Analysis CCA'2009, Ljubljana, Slovenia, August 18–22, 2009, pp. 21–22.

[Fernández and Tóth 2006] Fernández, J., Tóth, B.: "Obtaining the efficient set of biobjective competitive facility location and design problems"; In: "Proceedings of EURO XXI", Reykjavík, Iceland, July 2–5, 2006.

[Fernández and Tóth 2007] Fernández, J., Tóth, B.: "Obtaining an outer approximation of the efficient set of nonlinear biobjective problems"; Journal of Global Optimization 38, No. 2 (2007), 315-331.

[Fernández and Tóth 2009] Fernández, J., Tóth, B.: "Obtaining the efficient set of nonlinear biobjective optimization problems via interval branch-and-bound methods"; Computational Optimization and Applications 42, No. 3 (April 2009), 393-419.

[Fernández et al. 2006] Fernández, J., Tóth, B., Plastria, F., Pelegrín, B.: "Reconciling franchisor and franchisee: a planar multiobjective competitive location and design model"; In: "Recent Advances in Optimization", Springer Lecture Notes in Economics and Mathematical Systems 563 (2006), 375-398.

[Figueira et al. 2004] Figueira, J., Greco, S., Ehrgott, M. (eds.): "Multiple Criteria Decision Analysis: State of the Art Surveys", Kluwer, Dordrecht, 2004.

[Grzegorczyk 1955] Grzegorzyk, A.: "Computable functionals", Fundamenta Mathematicae 42 (1955), 168–202.

[Grzegorczyk 1957] Grzegorzyk, A.: "On the definitions of computable real continuous functions", Fundamenta Mathematicae 44 (1957), 61–71.

[Iljazovic 2009] Iljazovic, Z.: "Effective Dispersion in Computable Metric Spaces", In: Bauer, A., Hertling, P., Ko, K.-I., eds, "Proc. 6th Int'l Conf. on Computability and Complexity in Analysis", Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2009, http://drops.dagstuhl.de/opus/volltexte/2009/2268

[Jaulin et el. 2001] Jaulin, L. et al.: "Applied Interval Analysis", Springer, London, 2001.

[Ko 1991] Ko, K.-I., "Complexity Theory of Real Functions", Birkhäuser, Boston, 1991.

[Kreinovich 2009] Kreinovich, V.: "From Interval Computations to Constraint-Related Set Computations: Towards Faster Estimation of Statistics and ODEs Under Interval and P-Box Uncertainty", In: Bauer, A., Dillhage, R., Hertling, P., Ko, K.-I., and Rettinger, R. (eds.), Proceedings of the Sixth International Conference on Computability and Complexity in Analysis CCA'2009, Ljubljana, Slovenia, August 18–22, 2009, pp. 4–15.

[Kreinovich et al. 1998] Kreinovich, V., Lakeyev, A., Rohn, J., Kahl, P.: "Computational Complexity and Feasibility of Data Processing and Interval Computations", Kluwer, Dordrecht, 1998.

[Kubica and Woźniak 2008] Kubica, B. J., Woźniak, A.: "Interval methods for computing the Pareto-front of a multicriterial problem"; Lecture Notes in Computer Science, Springer, Berlin, 4967 (2008), 1382-1391.

[Kubica and Woźniak 2010] Kubica, B. J., Woźniak, A.: "An interval method for seeking the Nash equilibria of non-cooperative games"; In: Wyrzykowski, R. (ed.) "Proceedings of the Eighth International Conference on Parallel Processing and Applied Mathematics PPAM'2009, Wroclaw, Poland, September 13–16, 2009"; Lect. Notes Comp. Sci., Springer, Berlin (to appear).

[Moore at al. 2009] Moore, R. E., Kearfott, R. B., Cloud, M. J.: "Introduction to interval analysis", SIAM Press, Philadelphia, Pennsylvania, 2009.

[Nachbar and Zame 1996] Nachbar, J. H., Zame, W. R.: "Non-computable strategies and discounted repeated games"; Economic theory 8 (1996), 103–122

[Nickel and Puerto 2005] Nickel, S., Puerto, J. "Location Theory: A Unified Approach", Springer-Verlag, Berlin, 2005.

[Pour-El and Richards 1989] Pour-El, M. B., Richards, J. I.: "Computability in Analysis and Physics", Springer, Berlin, 1989.

[Rettinger and Weihrauch 2003] , Rettinger, R., Weihrauch, K.: The computational complexity of some Julia sets, In: Goemans, M. X. (ed.) Proceedings of the 35th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 9–11, 2003, ACM Press, 2003, 177–185.

[Ruzika and Wiecek 2005] Ruzika, S., Wiecek, M. M.: "Approximation methods in multiobjective programming", Journal of Optimization Theory and Applications 126 (2005), 473–501.

[Specker 1959] Specker, E.: Der Satz vom Maximum in der rekursiven Analysis. In: Heyting, A., ed., Constructivity in mathematics. Studies in Logic and The Foundations of Mathematics, North Holland, Amsterdam (1959), 254–265.

[Tóth and Fernández 2006] Tóth, B., Fernández, J.: "Obtaining the efficient set of nonlinear biobjective optimization problems via interval branch-and-bound methods";

In: "Proceedings of the 12th GAMM - IMACS International Symposium on Scientific Computing, Computer Arithmetic, and Validated Numerics SCAN'06", Duisburg, Germany, September 26–29, 2006.

[G.-Toth and Kreinovich 2009] G.-Toth, B., Kreinovich, V.: "Validated methods for computing Pareto-sets: general algorithmic analysis"; International Journal of Applied Mathematics and Computer Science 19, 3 (2009), 369-380.

[Villaverde and Kreinovich 1993] Villaverde, K., Kreinovich, V.: "A linear-time algorithm that locates local extrema of a function of one variable from interval measurement results"; Interval Computations, No. 4 (1993), 176–194.

[Weihrauch 2000] Weihrauch, K.: "Computable Analysis", Springer-Verlag, Berlin, 2000.

[Weihrauch 2003] Weihrauch, K.: "Computational complexity on computable metric spaces", Mathematical Logic Quarterly 249, 1 (2003), 3–21.

[Weihrauch and Grubba 2009] Weihrauch, K., Grubba, T.: "Elementary computable topology"; Journal of Universal Computer Science, 15, 6 (2009).

[Yasugi et al. 1999] Yasugi, M., Mori, T., Tsujji, Y.: "Effective properties of sets and functions in metric spaces with computability structure", Theoretical Computer Science 219 (1999), 467–486.