Algorithmic Social Sciences Research Unit

ASSRU

Department of Economics
University of Trento
Via Inama 5
381 22 Trento Italy

DISCUSSION PAPER SERIES

11 – 03

ON EXACT AND APPROXIMATE SOLUTIONS FOR HARD PROBLEMS: AN ALTERNATIVE LOOK

R. S. Bartholo, C. A. Cosenza, F. A. Doria, M. Doria & A. Teixeira

MARCH 2011

On exact and approximate solutions for hard problems in economics: an alternative look

R. S. Bartholo, C. A. Cosenza, F. A. Doria, M. Doria and A. Teixeira

Advanced Studies Research Group and Fuzzy Sets Laboratory
PIT, Production Engineering Program
COPPE, UFRJ
P.O. Box 68507
21945–972 Rio RJ Brazil.

BARTHOLO@PEP.UFRJ.BR
COSENZA@PEP.UFRJ.BR
FADORIA@GMAIL.COM
FADORIA@PEP.UFRJ.BR
MANUELDORIA@GMAIL.COM

Version 1.0
January 30, 2011

Abstract

We discuss in an informal, general audience style the da Costa–Doria conjecture about the independence of the $P = NP$ hypothesis and try to briefly assess its impact on practical situations in economics. The paper concludes with a discussion of the Coppe-Cosenza procedure, which is an approximate, partly heuristic algorithm for allocation problems.

Keywords: P vs. NP, allocation problem, assignment problem, traveling salesman, exact solution for NP problems, approximate solutions for NP problems, undecidability, incompleteness

1 Introduction

We deal here with NP–hard and NP–complete problems in economics. The most famous problem in the NP class is the *traveling salesman problem*: given N cities connected by a web of roads, which is the shortest route among them? Which route is such that it has length at least equal to, or larger than, some fixed value l?

In the first case we have an example of a NP–hard problem; in the second case, of a NP–complete problem.

Another example is the family of *allocation problems*. We have sites geographically dispersed over some region, and scarce resources to be distributed among them. We want to do it efficiently (for some previously established criteria of efficiency) and again can either impose an optimizing condition (which will make it into a NP hard problem) or fix some efficiency level (the NP complete case).

Both problems have strong import on practical, everyday economic situations, yet as we show here their roots appear to go deep in the foundations of mathematics and of computer science.

(Definitions of the main concepts appear in the course of the paper.)

Structure of the paper

This paper is divided into three parts. After a brief comment on the historical background together with another brief survey, of a few applications of NP–completeness to economics, we reach part two, which describes the ideas presented in [15] in a so to say "general audience" style. It is an approach to the P vs. NP question that leads to the conjecture that $P = NP$ and $P < NP$ adequately formalized within a strong axiomatic theory are independent of that axiomatic framework. An appendix gives a more technical approach to the matter.

Finally the third part describes two algorithms to deal with NP–complete problems. One is an exact algorithm which follows from our conjecture about P vs. NP; after that we exhibit an approximate, but very effective semi–heuristic procedure for the solution of NP–complete and NP–hard problems.

Algorithms for problems in the NP class

The literature about algorithms for NP–complete and NP–hard problems is vast and we won't touch it here. We will restrict our attention to the two specific examples we've just mentioned:

- *A nearly polynomial algorithm for NP–complete problems?* The first example we discuss directly originates in the analysis that led to da Costa et al. approach [16], but has never been much explored in a practical context. If $P = NP$ and $P < NP$ are independent of a strong axiom system such as Zermelo–Fraenkel set theory,[1] then there is a nearly polynomial algorithm for NP–complete problems. This result goes back to O'Donnell [24] and is again presented in detail but in a different form in [5].

- *An approximate algorithm.* The second example is the COPPE–COSENZA algorithm, which expands the Italian MASTERLI procedure, and is an approximate, partly heuristic solution technique which has been already tested in many real world situations.

  The COPPE–COSENZA procedure was originally conceived to deal with allocation problems, but has been applied to several situations, from medicine to architecture, and can be modified to deal with any kind of problem in the NP–class [4].

Recently S. Zambelli pointed out [33] that an user–friendly version of FAD's views on the P *vs.* NP question was needed [15], especially because it would also be of interest to an audience of economists. The same co–author has collaborated with the team that developed the so–called COPPE–COSENZA algorithm for the solution of allocation problems and planning strategies [4] and again would like to present that technique to a wider audience, while trying to evaluate its practical and theoretical impacts. The present paper originated in a discussion among Cosenza, Doria and the last author, Teixeira, and fuses both intents.

Undecidability and incompleteness in everyday economic situations?

"Naturally undecidable" problems in economics, that is problems where undecidability arises out of the very formulation of the question, have been investigated by K. V. Velupillai since the early 1990s [29]. The issue of whether we have undecidability, incompleteness — uncomputable stuff in general, so to say — in real world economic settings has already been touched upon in [3]. From the da Costa–Doria approach [15] which is still quite incomplete, it would follow, e.g., that for very simple economic situations we wouldn't be able to prove within reasonable axiom systems that a given procedure leads to a minimum cost. Would that fact result in some actual difficulty or obstacle which affects economic policies? This is one of the main questions which underlie the present paper. As stressed by Zambelli ([33], p. 3), we adress here the quite basic issue of "what an economic system can and cannot compute."

This is the gist of our work.

2 Notes on the history of NP–completeness

This section is *not* intended as a history of NP–completeness. Our purpose is merely to highlight a few steps that are relevant, in our opinion, in the development of these concepts.

[1]We elaborate a bit on the Zermelo–Fraenkel axioms in the later sections.

While we can trace the *traveling salesman problem* back to Sir William Rowan Hamilton in the 19th century, the first formulation of TSP is usually credited to Karl Menger in 1932 [23]. Menger's formulation of the TSP is quite straightforward:[2]

> *We call the Messenger Problem [...] the task of finding, for a finite number of points whose pairwise distances are known, the shortest path connecting the points. The problem is naturally solvable by making a finite number of trials. No rules are known that would reduce the number of trials below the number of permutations of the given point.*

The name "traveling salesman problem" makes one of its first appearances nearly two decades later, in a 1949 report prepared by Julia Robinson [25] for the RAND Corporation. Then follows Gödel's much–quoted letter to von Neumann in March 1956 [30] where the problem is again formulated, now in the context of a Boolean satisfiability problem.

Cook and Karp (see [22] for references and details) characterize NP–complete problems in the early 1970s; they are seen to pop up everywhere in both concrete and abstract situations.

One may now state the $P = NP$ question:

> *Is there a polynomial algorithm that settles all instances of some NP–complete problem?*

We must also briefly mention two important contributions to NP–completeness in economics: first, the Koopmans and Beckmann [21] consideration of assignment problems, and, last but not least, Herbert Simon's [32].

Velupillai gives an explicit, crystal–clear example of how a TSP problem can be converted into a Boolean satisfiability problem [30]. We have already mentioned the TSP problem and allocation problems as situations which lead do NP–complete problems. We now give a few more examples to support our case.[3]

3 NP completeness in economics

Do problems in the NP class matter for economics? They do — insofar as assignment or allocation problems, scheduling problems etc do appear in actual planning situations. Moreover, if the da Costa–Doria conjecture on NP completeness holds, then there will be lots of algorithmically undecidable situations in the actual consideration of NP–complete and NP–hard problems.

NP–complete and NP–hard problems are in general easily recognized because of the following features:

[2]We quote a translation of Menger's text.

[3]We thank V. Velupillai for an exacting and detailed analysis of the history of NP–completeness as it refers to economics.

- The obvious search for a solution involves a search over some combination or permutation of the elements involved.
- For the NP–hard case, we look for a combination of permutation of elements that maximize or minimize some condition K, which is easily checked.
- For the NP–complete case, there is a condition K' which must be satisfied by the solution; again it is easy to check it.

This is the general picture. Now let's go back to economics. Several theoretical situations in economics may also depend on the solution of a NP–complete problem. Consider the following case of an explicitly given game with coalitions:[4]

- We have a n–person game.
- We have some criterion K that has to be satisfied by coalitions in the game (say, returns for the coalitions).

In the general case the obvious solution technique is to examine all (explicitly given) possible coalitions in order to check for K, and that demands an exponential effort.

Yet if given some solution, we test for K, we'll see that it usually is a polynomial task. (This depends on the way we define K, but for most practical situations testing for K is "fast," that is, it can be made in polynomial time on the length of the input.)

There is a caveat here: in the general case, recursively presented competitive games are undecidable as implied by a result of A. Lewis — see [3] for a review. Also, in a very general context, finite games, be they competitive or not, are undecidable [3, 28]. That's the reason for the restrictive condition we added here, that of an explicitly given game.

More specific examples are now given. They show the widespread presence of NP–completeness in economics:

- *Computation of Nash equilibria.* Nash games are undecidable in the general case, a fact which is known since the Lewis' results obtained in the late 1980s (see the references in [3]; see also [14, 28]). However even for the simple case of explicitly given outcomes of games we get NP–completeness [2].

  We can offer an intuition about it: in order to compute Nash equilibria one must consider all possible combinations of players and strategies, so that out of the combinatorial explosion of alternatives we have the exponential growth typical of NP hard and NP complete problems. If we ask for a minimum return in the game, then to check whether some choice of strategies satisfies it is in general a polynomial task. Therefore the actual computation of Nash equilibria may then turn out to be a hard problem.

[4]The "explicitly given" clause is essential here.

- *Risk management.*[5] The choice of a portfolio given simultaneous risk and return constraints is also NP–complete, for we must consider all possible securities combinations and see whether they satisfy the desired constraints or not.

  This fact has led Huberman and coworkers to the suggestion (among others) that we should look at NP–completeness from the viewpoint of economics [19]. The idea is a very interesting one, for in most tasks that lead to NP–complete problems we can evaluate each alternative in terms of gains or in terms of risks incurred. For instance, given the current status of DNA research, and given the parents' genetic makeup, if we define a level of risk of defects in the possible offspring, we have a NP–complete problem which can be evaluated in economic terms, that is, in terms of gains (the children) and risks (namely inherited defects and dseases).

- *Shapley values.* NP–complete questions appear in the theory of cooperative games in general.[6] A recent example is given in [8], which also lists related references.

  The chief result in the Conitzer and Sandholm paper is the proof that core membership in an explicitly given cooperative game is NP–complete, a result that mirrors an earlier result by Cheng and Papadimitriou [17]. However the calculation of Shapley values is easy, again as shown by the same authors Deng and Papadimitriou.

So, we believe that our case for the relevance of NP–hardness and NP–completeness for economists has been successfully pleaded. Not only the class of NP problems do matter for economics: we can even follow the converse path and look at some NP problems from the viewpoint of typical economic concepts such as gain, risk or value.

4 The da Costa–Doria approach to P *vs.* NP

We now sketch the main ideas presented in the da Costa–Doria paper ([15]; see also [7]), which support the conjecture that $P = NP$ and its negation $P < NP$ are independent of quite strong consistent axiomatic systems (the authors argue for Zermelo–Fraenkel axiomatic set theory with the Axiom of Choice). It is known that if independence holds then $P < NP$ will be true in the standard world for arithmetic, that is, in the real world of computers and the like.

Required concepts from mathematical logic and computer science can be found in [3]. We will essentially need a few intuitions about Turing machines, plus some results on logic which can be found in the reference.

[5] We use the concept of risk management as in [18].

[6] The authors thank V. Velupillai for reminding them of this application. For detailed discussions of complexity in economics see Velupillai's recent book [31].

Polynomial Turing machines

Turing machines are supposed to input sequences of 0s and 1s such as, say, 001010. They are called *bit strings.* A *polynomial Turing machine* (or *poly machine*) is a Turing machine whose operation time is bounded by some polynomial on the length of the input string. (The length of a string is the number of 0s and 1s in it; given a bit string x, its length is noted $|x|$.)

The NP class of problems and the $P = NP$ conjecture

NP stands for "nondeterministic polynomial"; we'll clarify the meaning of that later on.[7] Problems in the NP class are described by the slogan:

> *If you know the solution for a problem in NP then you can test for (check) it very fast. However if you don't know the solution then all known algorithms are such that it is very hard (it takes a long long time) to get one solution in the general case.*

More precisely: they are easy to check because they can be checked by a poly machine. They are hard to find because in the general case nobody knows an algorithm for it which is polynomial, i.e., can be implemented by a poly machine. Then follows the $P = NP$ question (this equation means, NP problems are solvable by poly machines):

> *Is there a poly machine that settles all instances of some NP problem?*

It is known that the so–called nondeterministic poly (that is, NP–) machines can settle any NP problem in polynomial time, whereas the use of the abbreviation NP to denote that class of problems [22].

Formally, a problem NP_H is NP–hard if and only if there is a polynomial Turing machine that reduces some NP–complete problem to it. This essentially means that if we can solve NP_H, then we can also settle the NP–complete problem we started from, modulo a "fast," polynomial computational map. Example:

- *NP–complete.* Get a route that goes through N cities, with length $l \leq L$, L fixed.

  Notice that depending on L, there may not be such a route.

- *NP–hard.* Get the minimal length route l that goes through N cities.

  This is an optimization problem.

[7]Basically this refers to a machine which can process an input in simultaneous parallel procedures.

On the da Costa–Doria conjecture for P vs. NP

The kernel of the da Costa–Doria approach to the P vs. NP problem has to do with the behavior of the so–called "counterexample function to $P = NP$." Let's describe it:

- We start from the usual Gödel numbering to all Turing machines. (Given each Turing machine M there is an integer e_{M}, its Gödel number, that codes the machine's program [22, 26].)

- Select some NP–complete problem, which is kept fixed. Define the counterexample function f:

  – If e_{M} codes a nonpolynomial machine, then put $f(e_{\mathsf{M}}) = 0$.

  – If e_{M} codes a polynomial Turing machine then $f(e_{\mathsf{M}})$ = first instance (if any) of our previously fixed NP–complete problem so that machine M *fails* to output a correct answer to the problem.

  – If that never happens, then $f(e_{\mathsf{M}})$ = undefined and $\{e_{\mathsf{M}}\}$ polynomially settles our fixed NP–complete problem, and any other NP–complete problem, modulo a polynomial algorithmic transformation.

The counterexample function f lists all first failures to correctly solve some problem in an NP–complete example. *If there is some algorithm that polynomially settles all instances of some NP–complete problem, then f is undefined at that algorithm.* Now it has long been asserted as a folklore fact:

> *The counterexample function f grows in its peaks beyond all intuitively total recursive functions.*

This means that it grows at least as fast as the Busy Beaver function, and that it is a noncomputable function. But we can make a trick and transform that intractable function f into a recursive function f which is called the BGS counterexample function [1]. The BGS function f is defined over the BGS polynomial Turing machines. A BGS poly machine is a Turing machine coupled to a clock. The machines inputs a binary string, the clock evaluates out of the input's length some polynomial bound and begins to count the machine's operating steps. Just before it exceeds the bound the clock shuts down the machine, if it hasn'st stopped before. Then the output is the value computed if the machine stops before the clock shuts it down, or zero otherwise.

All BGS machines are polynomial Turing machines, as the clock can be simulated by some subroutine; and given any polynomial Turing machine there is a BGS machine that computes the same recursive function as the given machine. The BGS set is a recursive set.

But notice that we can expand the concept and define several BGS–like sets which are naïvely equivalent to the original BGS set. For instance:

- Define a *scale of functions* within our theory, say, ZFC.

- A scale of functions is an ordered set $\mathsf{F}_0 < \mathsf{F}|_1 < \mathsf{F}_2 < \ldots$, where for $i < j$, F_i is dominated by F_j.
- Moreover we require that the diagonal function given by:

$$\mathsf{F} = \{\mathsf{F}_0(0), \mathsf{F}_1(1), \mathsf{F}_2(2), \ldots\}$$

  while intuitively recursive and total, cannot be proved to be so in ZFC.

Such scales exist both in Peano Arithmetic (PA) and in ZFC.

We can define a BGS–like set as follows: form all pairs of Turing machines coupled to clocks that operate according to a polynomial $|x|^{\mathsf{F}_k(n)} + \mathsf{F}_k(n)$, where $|x|$ is the length of the input binarily coded. This extended BGS set looks like an infinite collection of the original BGS sets, each one with the exponent in the clock given by some F_k. The advantage? Large numbers are coded by a much shorter sequence of binary digits, namely the (fixed) program for F_k plus the binary version of n, instead of the full, explicitly written value of $\mathsf{F}_k(n)$. The counterexample function f over such a set will be partial recursive, and naïvely total iff $P < NP$. However is it provably total in ZFC? We suspect that it isn't so.

Yet there is now a surprising result. Consider some strong axiomatic theory, like ZFC (Zermelo–Fraenkel set theory with the Axiom of Choice). Then the following holds:

> *The sentence: "f is total, if and only if* f *is total" is independent of the* ZFC *axioms, supposed consistent.*

(The equivalence holds true of the standard model for arithmetic.) The independence result holds even if we strengthen ZFC with strong axioms such as large cardinal axioms.

Da Costa and Doria conjecture that the sentences $P = NP$ and $P < NP$ will turn out to be independent of such strong axiom systems.

5 The O'Donnell exact nearly efficient algorithm for NP problems

The following result depends on the conjecture already stated in this paper:

> *Both $P = NP$ and $P < NP$ are independent of the axioms of axiomatic set theory,*[8] *supposed consistent.*

Then there is a possibly quite efficient but not very much explored algorithm for those problems. It was first presented in 1979 [24] and then reintroduced in 1991 [5]. The present version follows [16].

The algorithm is exponential, but depends on an extremely slow growing function, and as such will be polynomial for all practical purposes, given the independence conjecture. (More about it at the end of this section.)

[8]That is, ZFC axiomatic set theory.

We will require the already mentioned BGS set of machines [1]. A BGS machine is a Turing machine coupled to a clock (which is also a Turing machine). One inputs a string x of length $|x|$ to the machine and to the clock. The clock computes some polynomial function of it, say, $|x|^a + b$, for a, b, positive integers, and watches the operation steps of the coupled Turing machine. The moment the machine reaches step $|x|^a + b$, the clock shuts down the operation of the Turing machine (of course if it has already stopped and given some output, this won't happen).

One usually agrees that the output of a shutting–down operation is zero, for zero is a "failed" answer. The couple Turing machine + clock is of course a poly machine, and every poly machine can be written in that form. So the BGS set adequately represents all poly machines but doesn't include all of them.

We can also form exotic BGS sets, say, BGS$^{\mathsf{F}}$, where the clock bound is given by $|x|^{\mathsf{F}(a)} + \mathsf{F}(b)$. The counterexample function over the exotic set is recursive (as any such BGS set is intuitively recursive), and is noted f^{F}.

The O'Donnell algorithm

We can now describe the O'Donnell quasi–polynomial algorithm for an NP problem. We consider the exotic counterexample function f^{F} to $P = NP$, restricted to the exotic BGS set, BGS$^{\mathsf{F}}$. (But we will write f for short here.)

- We suppose independence and we argue in the intuitive world of arithmetic, which is also the real world of computers and the like.

- We consider the set of all problems in the NP class. We restrict our attention to discrete problems, which turn out to be enumerable, and we consider both NP–hard and NP–complete problems, that is we include those that are unsatisfiable.

- Consider the poly Turing machine $\mathsf{V}(x, s)$, where $\mathsf{V}(x, s) = 1$ if and only if s satisfies x, and $\mathsf{V}(x, s) = 0$ if and only if s doesn't satisfy x.

  That machine is polynomial, as it checks, given some possible solution for our problem, wheter it fits the picture or not.

- Consider the enumeration of the BGS$^{\mathsf{F}}$ [1] machines, $\mathsf{P}_0, \mathsf{P}_1, \mathsf{P}_2, \ldots$.

- Consider x, the binary code for a problem in NP.

- Alternatively check for $\mathsf{V}(x, 0)$, $\mathsf{V}(x, 1)$, ... up to — if it ever happens — some s so that $\mathsf{V}(x, s) = 1$; or,

- Input x to $\mathsf{P}_0, \mathsf{P}_1, \mathsf{P}_2, \ldots$ up to the first P_j so that $\mathsf{P}_j(x) = s_j$ and s_j satisfies x.

  Notice that the index $j = \mathsf{f}^{-1}(x)$.

- Now, if f is fast–growing, then as the operation time of P_j is bounded by $|x|^k + k$, we have that $k \leq j$, and therefore it grows as $\mathsf{f}^{-1}(x)$. This will turn out to be a very slowly growing function.

  More precisely, it will have to be tested up to j, that is the operation time will be bound by $\mathsf{f}^{-1}(x)(|x|^{\mathsf{f}^{-1}(x)} + \mathsf{f}^{-1}(x))$.

Then either x is unsatisfiable — and therefore one will have to test all possible s ad infinitum — or, if satisfiable, operation time has the nearly polynomial time given above. This means that if independence holds, then we will have something that might be informally written as $P \approx NP$ — there are nearly polynomial algorithms. (To deal with the unsatisfiable case we simply put some bound that tops $\mathsf{f}^{-1}(x)(|x|^{\mathsf{f}^{-1}(x)} + \mathsf{f}^{-1}(x))$ above on the operation time of our algorithm, and anything that goes beyond it is rejected.)

Notice that the actual bound is still weaker, as one should in fact use the (noncomputable) counterexample function f instead of f.

Undecidability and incompleteness

Given the independence hypothesis, we cannot prove in our theory S that the algorithm performs in an adequate way, even if it is clear from the preceding intuitive analysis. Does this affect actual economic situations? We address this issue at the end of our discussion.

Other well–behaved algorithms

There is a piece of good news here, which does not depend on the independence hypothesis of da Costa and Doria: for the typical situation in the Satisfiability Problem (SAT), the usual exponential algorithm based on truth–table calculations is polynomial.

Let's elaborate a bit on that:

- A Boolean expression in conjunctive normal form (the usual fare for SAT) with n propositional variables generates a truth–table that can be coded by a 2^n–bit binary string.

- Given any 2^n–bit binary string there is a corresponding Boolean expression in SAT.

- In the most frequent situation, the truth–table thus represented will be Chaitin incompressible, and therefore the corresponding Boolean expression will be of length of the order of 2^n.

- Thus the usual truth–table algorithm will input a code for the Boolean expression that is about as long as its output.

The truly nasty cases for the truth–table algorithm are infrequent.

6 The COPPE–COSENZA procedure

For a review of approximation techniques for NP–hard and NP–complete problems see [20].

The COPPE–COSENZA technique for approximate solutions of allocation problems is now briefly described [9]. It expands an earlier Italian model, the MASTERLI model (MODELO DI ASSENTO TERRITORIALE E LOCALIZZAZIONE INDUSTRIALE) which was conceived and applied in the early 1970s. The COPPE–COSENZA procedure is partly heuristic and uses a fuzzy–logic approach in its handling of data. It is widely used in locational studies in Brazil [4] and in dealing with allocation problems from economics and engineering to medicine [12, 13]; empirical data support the contention that it in general gives better, more efficient, solutions to the issues considered.

The main idea is disarmingly simple: we define two matrices, A and B. The first one, matrix A, tells us the required factors. The second matrix, B, exhibits the possible alternatives we have in the real world in order to implement our wishes.

Then there is a third matrix, C, which is a function of A and B, which allows us to compute optimal allocations out of our desiderata and out of the real–world alternatives we have at our disposal. That computation is both simple and fast, and may have heuristic components. The main ideas can be formulated for crisp (not fuzzy) sets, but a more sensitive algorithm may be obtained with the help of fuzzy objects.

Construction of matrix A

Suppose that we have several industries to distribute over a given geographic space, and suppose that we have different potential placements for those industries. The first matrix describes the industries we are interested in, and relates them to requirements for these industries (say, a shoe factory requires a continuous leather supply, water, energy, some chemical inputs and pollution control).

The first matrix, A, with k lines and m columns, has the following structure:

- *Lines* list the industries, $p_1, p_2, \ldots, p_k$.
- *Columns* list the requirements (factors) for these industries, $f_1, f_2, \ldots, f_m$.
- Given matrix A, its entries are linguistic variables A_{ij}, say:
  - Critical factor.
  - Decisive factor.
  - Indecisive factor.
  - Irrelevant factor.

Construction of matrix B

Matrix B has n lines and k columns and tells us what we have to offer to the demands in matrix A.

Matrix B has the same structure as A but in a transposed way:

- *Lines* list the same requirements $f_1, f_2, \ldots, f_n$ that appear in A. Matrix B tells us what is available in our prospective placements.
- *Columns* list possible placements for our industries, $z_1, z_2, \ldots, z_m$, where in general $n \neq m$.
- Again the entries B_{jk} of B are linguistic variables:
  - Optimal availability.
  - Good availability.
  - Regular availability.
  - Poor availability.

Construction of matrix C

Matrix C will be the tool we require to do actual computations. Its entries are given by a heuristic procedure:

1. Suppose that there is demand for factor f_i (1 value of demand), and that region z_j doesn't have that factor (0 value of offer). We put $C_{ik} = 0$.
2. Suppose that there is no demand for f_i (0 value of demand), and yet that region z_j has that factor (value equal to 1). We put $C_{ik} = 1/n$.
3. Suppose that there is no demand for f_i (0 value of demand), and that region z_j doesn't have that factor (value also equal to 0). We put $C_{ik} = 1/n!$.
4. Finally suppose that there is a demand for f_i (1 value of demand), and that region z_j has that factor (value equal to 1). We put $C_{ik} = 1$.

These are simply "marks" or "grades" we give for the possible alternatives. Cases 1. and 4. are obvious: they correspond to 0% and 100%, respectively. Case 2. gives an intermediate nonzero value because the fact that (momentarily) one doesn't require a factor that is available and which may be required in the future must be taken into account. Finally Case 3 — no demand and absence of a prescribed factor in the region — is given a nonzero value not to penalize the possibility, as 0 should only be given to a factor that is required and isn't available.

Ranking techniques

There are several alternative, empirically tests, ranking techniques, which depend on the optimal goals to be attained. We describe here a simple ranking scheme that has given very reliable results in actual optimizing situations:

- Consider the demand for industry j. Form the demand value $D_j = \sum_k D_{kj}$. That is to say:
  - Fix industry j. For site k, sum over all "grades" of the factors required for j. This gives the demand D_j of industry j.
- Examine the offer for site m: $O_m = \sum_i O_{ik}$. The offer is calculated as we do for the demand.
- The rank grade r_{jm} of site m with respect to industry j is given by:

$$r_{jm} = \frac{O_m}{D_j}.$$

Given matrices A, B, C, there are of course many other possibilities to rank locations for industries with respect to the required factors. This is just the first possibility.

We can also have global rankings:

- Compute the global demand, for all industries, $D = \sum_j D_j$.
- Compute the global offer, $O = \sum_j O_j$.
- The rank grade r_m of site m with respect to all industries is: $r_m = O_m/D$.

Of course $r_m < 1$ means that not all requirements are fulfilled, while $r_m \leq 1$ means the fulfilment of the requirements. One usually ranks the solutions with respect to the global demand D (and not the D_j), which allows for a fast algorithmic treatment of the procedure.

Comparison with an analytic solution

Given a fixed budget X, the distribution of F activities among Y locations, and the calculation of an arrangement, if any, that satisfies the prescribed budget is clearly NP–complete. For the obvious algorithm for the computation of an adequate arrangement of activities and locations is exponential, while testing whether some particular arrangement fits the budget can be done in polynomial time on the length of the input data.

Let's elaborate on that. In order to obtain one exact solution (if it exists) for one such NP–complete allocation problem as described, one would need:

- Crisp values for resources and demand at each location.

- We would have to consider all locations and the corresponding data; a solution for the problem might involve e.g. raw material obtained at places A_1 and A_2, plant location at A_4, administrative tasks at A_6, and so on.

  So, we would have to consider all possibilities at all places, which together with budget constraints makes the problem thus formulated into a NP–complete one.

The COPPE–COSENZA procedure is a kind of cutting the Gordian knot solution. It abandons crisp values for fuzzy data such as "critical" or "good." The ranking matrix sort of aggregates all data instead of considering each and one individual possibility, and the final ranking procedure derives from that aggregate consideration of all possible locations in a single demand index.

This semi–heuristical procedure however is known to provide efficient responses to most actual situations where it has been applied specifically because it markedly improves over already existing unplanned situations. An example is its application to the Brazilian Biodiesel Program [10, 11].

Anyway we believe that it is difficult or perhaps even impossible to analytically compare such a semi–heuristical procedure with a fully formal one.

7 Discussion

We take our cue from the last sentence in the preceding section. It has been once asked by John Casti[9]: is incompleteness a red herring? Is undecidability a red herring?

Not quite so. Undecidability is an everyday problem given our Turing–based computers. "Bugs" are quite often the result of the unsolvability of the Halting Problem. If we deal with a complex enough economic problem and try to handle it with some computer program, undecidability will assuredly creep up and disturb our work. Incompleteness is the way undecidability looks when seen from inside a formal theory.

The same situation happens to NP hard and NP–complete problems.

We presented here in an intuitive vein a long argument that purports to show that $P = NP$ and $P < NP$ are independent of strong axiomatic systems. Yet we conclude our examples with an approximate algorithm that is known — by practical testing — to perform quite well, and which has a fast, low degree polynomial implementation. A brief summary of what we have out of these situations is:

- If we deal with mathematical models for economic situations, we may have to cope with undecidability, incompleteness and NP–completeness.

- If we confront a practical, everyday, particular situation which can be usefully treated with a semi–heuristic procedure, then we will probably be able to forget about undecidability, incompleteness and NP–completeness.

[9]In a private discussion with F. A. Doria in 1994.

Of course there are crisp problems, say, cryptographic decoding procedures, that will always require exact solutionf for a NP–complete problem. But if we only need approximate solutions then an algorithm like the COPPE–COSENZA procedure is enough.

Yet — think about the situation in physics. When Einstein first published his general theory of relativity in 1916, nobody would even consider the possibility that it would affect us in ordinary, everyday situations. Well, today our homely GPS guiding devices use general relativity corrections to help is in establishing our location.

This is a sobering remark. Follows that undecidability, incompleteness and again NP–completeness may eventually matter, after all.

8 Acknowledgments

The ongoing research program that led to this text has been sponsored by the Advanced Studies Group, Production Engineering Program, COPPE–UFRJ, Rio, Brazil.

C. Cosenza wishes to thank Professor Philip Arestis for his hospitality at the Department of Land Economy, Cambridge University.

FAD acknowledges support from the Brazilian Academy of Philosophy and its chairman Professor J. R. Moderno. The same author acknowledges partial support from CNPq, Philosophy Section. MD acknowledges a scholarship from CNPq, Applied Sciences Section.

Finally we must thank Professor Vela Velupillai for an exacting criticism of this work, which greatly helped us in improving it. We must also thank Vela for his kind invitation to publish this paper in his prestigious preprint series.

References

[1] T. Baker, J. Gill and R. Solovay, "Relativizations of the $P = ?NP$ question," *SIAM J. Comp.* **4**, 431–442 (1975).

[2] R. Baron, J. Durieu, H. Haller, P. Solal, "Finding a Nash equilibrium in spatial games is an NP–complete problem," *Economic Theory* **23**, 445–454 (2004).

[3] R. Bartholo, C. A. Cosenza, F. A. Doria and C. Lessa, "Can economic systems be seen as computing devices?" *J. Econ. Behavior and Organization* **70**, 72–80 (2009).

[4] R. Bartholo, C. A. Cosenza, F. A. Doria and M. Doria, " A heuristic algorithmic procedure to solve allocation problems with fuzzy evaluations," to appear (2010).

[5] S. Ben–David and S. Halevi, "On the independence of $P\, vs.\, NP$, Technical Report # 699, Technion (1991).

[6] W. Carnielli and F. A. Doria, "Is computer science logic dependent?" in C. Dégremont et al., *Dialogues, Logics and other Strange Things: Essays in Honor of Shahid Rahman*, College Publications (2008).

[7] G. J. Chaitin, N. C. A. da Costa and F. A. Doria, *After Gödel: Exploits into an Undecidable World*, to appear (201X).

[8] V. Conitzer and T. Sandholm. "Computing Shapley values, manipulating value division schemes, and checking core membership in multi–issue domains," in *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI-04)*, 219–225, San Jose, California, USA (2004).

[9] C. A. N. Cosenza, "Industrial location model: a proposal," preprint, Martin Centre, Cambridge U., Cambridge (1981).

[10] C. A. N. Cosenza, "Brazil's biodiesel project," Centre for Brazilian Studies, Oxford University (2005).

[11] C. A. N. Cosenza, C. Neves, F. Rodrigues Lima, "A decision–making method for the selection of biodiesel sites in Brazilian regional planning," internal report, PIT–Coppe (2005).

[12] C. A. N. Cosenza, M. E. Cosenza–Andraus, C. F. Andraus, S. Leon, R. G. Nunes, "Monitoración prolongada por vídeo EEG de pacientes con diagnostico ambulatorial de epilepsía del lobo frontal de difícil control," *Rev. Neurología de Barcelona* **43**, 1–20 (2006).

[13] C. A. N. Cosenza, R. C. M. Nobre, O. C. Rotunno, W. J. Mansur, M. M. M. Nobre, "Ground water vulnerability and risk mapping using GIS modeling and a fuzzy logic tool," *J. Contaminant Hydrology* **1**, 1–36 (2007).

[14] N. C. A. da Costa and F. A. Doria, "Computing the future," in K. V. Velupillai, ed., *Computability, Complexity and Constructivity in Economic Analysis*, 15–50, Blackwell (2005).

[15] N. C. A. da Costa and F. A. Doria, "Hypotheses that imply the independence of $P = NP$ from strong axiomatic systems," in S. Zambelli, ed., *Computable, Constructive and Behavioural Economic Dynamics*, 79–102, Routledge (2010).

[16] N. C. A. da Costa, F. A. Doria and E. Bir, "On the metamathematics of the P *vs.* NP question," *Applied Mathematics and Computation* **189**, 1223–1240 (2007).

[17] X. Deng and Ch. Papadimitriou, "On the complexity of cooperative solution concepts," *Mathematics of Operations Research* **19**, 257–266 (1994).

[18] D. Hester and J. Tobin, *Risk Aversion and Portfolio Choice*, John Wiley (1967).

[19] B. Huberman, R. M. Lukose, T. Hogg, "An economics approach to hard computational problems," *Science* **275**, 51–54 (1997).

[20] P. N. Klein and N. E. Young, "Approximation algorithms for NP–hard optimization problems," *Algorithms and Theory of Computation Handbook*, Ch. 34, CRC Press (1999).

[21] T. C. Koopmans and M. J. Beckmann, "Assignment problems and the location of economic activities," *Econometrica* **25**, 53-76 (1957).

[22] M. Machtey and P. Young, *An Introduction to the General Theory of Algorithms*, North–Holland (1979).

[23] K. Menger, "Das Botenproblem," in K. Menger, ed., *Ergebnisse eines Mathematischen Kolloquiums* **2**, 11–12 (1932).

[24] M. O'Donnell, "A programming language theorem which is independent of Peano Arithmetic," *Proc. 11th Ann. ACM Symp. on the Theory of Computation*, 176–188 (1979).

[25] J. Robinson, "On the Hamiltonian game (a traveling salesman problem," RAND Research Memorandum RM–303 (1949).

[26] H. Rogers Jr., *Theory of Recursive Functions and Effective Computability*, Addison–Wesley (1967).

[27] J. Stillwell, *Roads to Infinity*, A. K. Peters (2010).

[28] M. Tsuji, N. C. A. da Costa and F. A. Doria, "The incompleteness of theories of games," *J. Phil. Logic* **27** 553–563 (1998).

[29] K. V. Velupillai, *Computable Economics*, Oxford (2000).

[30] K. V. Velupillai, “A computable economist’s perspective on computational complexity,” in J. Barkley Rosser Jr., ed., *The Handbook of Complexity Research*, Ch. 4, 36–83, Edward Elgar Publishing Ltd. (2009).

[31] K. V. Velupillai, *Computable Foundations for Economics*, Routledge (2010).

[32] K. V. Velupillai, “Foundations of boundedly rational choice and satisficing decisions: reviving a Simon tradition,” *Second Herbert Simon Lecture*, National Chengchi University, Taipei, Taiwan (2010).

[33] S. Zambelli, “Computable and constructive dynamics. . . ,” in S. Zambelli, ed., *Computable, Constructive and Behavioural Economic Dynamics*, 1–12, Routledge (2010).