Denotation in Algorithmic Perspective

Mihail Radu Solcan

2005; revised 2006-05-28

Contents

Abstract

The paper attempts to contribute to a new form of analytical philosophy. Classical analytical philosophy, illustrated by such masterpieces as Russell's "On Denoting"[11] used logical tools borrowed from the predicate calculus. The new tools are borrowed from new chapters in logic, such as the study of computability, recursion and programming. The idea exploited in this paper is that expressions in natural language contain clues for the design of corresponding algorithms. In particular, descriptions offer clues for the computation of denotations or the attempt to compute a denotation. The paper stresses the idea that such algorithms do not function as meanings. They must be considered from the perspective of the actions of the person who is computing a denotation.

1 From Philosophy to Computational Intelligence

A hundred years ago Bertrand Russell published a paper[11] in which he explained his version of the logical analysis of the natural language. The basic idea was that constructions in a natural language contain traps. We avoid such traps with the help of logic.

The analysis illustrated by Bertrand Russell[11] had also an impact beyond philosophy, but we are interested in this paper only in the philosophical significance of this type of analysis. Russell's analysis clearly separates the surface and the deep levels of the sentences formulated in a natural language. We reach the deep levels with the help of logic. Perhaps the most important feature here is the exactness, the precision that is obtained with the help of the symbolic apparatus.

Logic and even philosophical logic have had an impact well beyond philosophy. In logical programming the predicate calculus has stimulated a stress on declarations and the extraction of implicit knowledge from explicit knowledge. In a computer language like Prolog, used in computational intelligence, the programmer does not formulate instructions for the computer. Almost as in classical logical analysis, the programmer relies on the precise formalization of chunks of knowledge and inferences. The Prolog system performs automatically the inferences. Thus from chunks of explicitly formalized declarations follow the logical consequences.

Reading a treatise on artificial intelligence is quite an interesting experience for a philosopher. It is possible to read it as a kind of in-

troduction to philosophy (mainly analytical philosophy, of course).[1] Does however the connection work the other way round? Is it possible to track a distinct influence of the work done in computer science and computational intelligence upon philosophy? We tackle this question in the rest of the paper, but - following the tradition o analytical philosophy - we try to place it against the background of a more limited, specific problem. Therefore we have chosen a problem discussed in Russell[11].

2 The impact of algorithm theory on logic and philosophy

The question of the impact of the computer science upon philosophy presupposes an answer to another interrogation: which would be the lasting contribution of computing to the great ideas underlying theoretical thinking? Many people believe that the lasting contribution is summarized by one word: algorithm.

The theory of algorithms is a newcomer. It started as a by-product of the investigations in the 1930's of the decision procedures in logical and mathematical systems. The theory of algorithms is now strongly associated with the development of computer science.

There are a lot of interesting results in the pure theory of algorithms. There is, on the other hand, a lot of practical experience accumulated in the field of the design of algorithms. The practical design of algorithms displays a nice trait from the point of view of analytical philosophy. The design of an algorithm starts with an idea formulated in natural language. The algorithm is then translated into a more precise language and it is subjected to a process of stepwise clarification. Finally, there is a translation into a precise formal language. In analytical philosophy there is a tradition that stresses the importance of the clarification of ideas with the help of logic through a similar process.

The notion of algorithm is usually defined in terms of instructions. An algorithm is made up of clear and unambiguous instructions for transforming an input into an output.[2] We want to put

[1] We have mentioned in the bibliography the paper by Marcu and Hirst[6] because it covers some of the topics of the present paper. But it is also an example of a paper written for those interested in computational intelligence and computational linguistics which cites however philosophers: Frege, Russell, Meinong, Parsons. The authors use their contributions and build a system that, in its turn, could as well be studied by philosophers.

[2] Donald E. Knuth[5, §1.1] stresses first the **finite** and **clear** character of an

here this kind of definition in the perspective of the practical design of algorithms.

The practical design of an algorithm starts with an idea formulated in a natural language. It is a very bad habit to start writing a computer program directly in a computer language. From the point of view of programming practice, the clarification process is the key. This process closely resembles the process of clarification through logical analysis. The difference is however that we are not looking for exact declarations concerning various situations; we are looking for the actions that lead to the desired result.

As we have already observed, algorithms can be specified both in natural language and in the computer languages (which have the structure of a formal language). One can also specify algorithms in intermediary languages. A good programmer starts with a specification in a natural language and then she gives it a form in pseudocode (a mixture of natural language and formal constructions). This intermediate language is very useful and we are using it in this paper.[3]

What captures an algorithm? Any algorithm starts with the specification of an initial action. It then contains specifications for a finite number of actions that lead, after a finite number of steps, to a value. As we observed before, during the practical design of algorithms, there is a process of stepwise refinement of the initial formulation (in natural language) of the algorithm. For some authors, the algorithm captures the semantic content of the final ideal formulation in a symbolic language.[4] This leaves however aside the problem of the continuity of the process of stepwise refinement of the algorithm.

The view that we embrace here is that the algorithm is represented by what remained identical during the process of stepwise refinement. One may compare this view with the theories of per-

algorithm. An algorithm has also **input(s)** and output(s). The output(s), the result(s) of the algorithm correspond to the **value(s)** returned by the algorithm. Knuth also emphasizes the significance of **efficiency**: the algorithmic process (the transformation of the input into the output according to the algorithm) must lead to a value in a reasonable time interval. The finite character of the process is not enough.

[3]For a very good introduction to algorithms see Baldwin and Scragg[1]. Their book on the science of computing uses as pseudocode a language that is very close to usual English. This is also a very good book for philosophers who have few ideas about algorithm theory.

[4]I think this is the view expressed by Knuth[5]. See also in § 2.1 the presentation of the important point of view of Yiannis Moschovakis.

sonal identity. Look at your picture as a baby! This is **your** picture as a baby. It is not easy to specify the conditions for personal identity, but we talk about the **same** person. Likewise, in our quest for a role for algorithms in philosophical analysis we will use the principle "don't look for the ideal algorithm, look for **algorithmic identity**".

Let us take as an example the Euclid algorithm.[5] Let us say that we deal with two positive integers. The first is greater than the second. We try to find the value of their greatest common divisor. We divide the first integer by the second, but there is a remainder. The crucial observation is that if the remainder divides the second number (the smaller number) then it also divides the first number. If we get again a non-null remainder we repeat the procedure until we get a null remainder.

The above formulation is far from being crystal clear. If I add more observations in natural language the result would be even more awkward. Add a few symbols and you obtain a far better formulation. For example, one can easily express in this way the structure of the first number as the sum of the product of the quotient with the second number and of the remainder. The steps of the algorithm may also be stated using symbols.

However, during the whole reformulation process, the same "idea" remains behind the procedure used for the computation of the value of the greatest common divisor.

It is prudent to distinguish between the algorithm and the algorithmic process. Take, for example, three pairs of positive integers: (24, 12), (28, 21) and (30, 16). If we apply the above algorithm we get three different **processes**. For the first pair the remainder is null after the first division. We get a remainder (namely, 7) in the second process. We get two remainders (14 and 2) until the third process stops.

The old Euclid algorithm is fairly simple but it shows what an algorithm does: it shows us how to use a set of operations we already know (division, for example) in order to compute a value. This is indeed something Russell is interested in when he writes about denotations. We do not just talk about denotations; we also compute them. We find the value of a denoting phrase.

In the study of an algorithm, there are two very important aspects. First, we must show that the algorithmic process halts and returns a value. We have to prove that this result is correct. For ex-

[5] See, for example, Knuth[5, §1.1].

ample, we might prove that the Euclid algorithm returns the correct value (the value of the greatest common divisor).

The other aspect that is very important in the study of an algorithm is the complexity of the corresponding algorithmic process. Roughly speaking, the complexity is the number of steps in the algorithmic process. Some algorithms return a result faster than others; they are less complex.[6]

Excessive complexity or a process that ends up looping for ever must be detected both for theoretical and practical reasons.

Mathematical logic is also interested in algorithms. The short presentation of mathematical logic by Yiannis Moschovakis[7] shows clearly how logic has changed. Computability, recursion and programming are now distinct research domains in mathematical logic. And, as the work of Yiannis Moschovakis himself shows us, the abstract, formal characterization of the concept of algorithm is very important from a logical point of view.[8]

It would not be difficult to document the impact of the theory of algorithms on philosophical logic, but the main question here is the impact upon other areas of philosophy.

There is an indisputable impact of algorithm theory and computer programming upon philosophers interested in cognitive science. Daniel Dennett, for example, had a striking idea: he sees evolution as an algorithmic process.[9]

Among the philosophers of science, Paul Thagard is notable for his insistence on the idea that philosophers should not just talk about programming, but write and experiment with computer programs. Thagard recognizes that „the kind of training that most philosophers have, which includes little preparation for actually doing computational work“ prevents the spread of the computational approach in philosophy.[10]

In fact, for those philosophers who are familiar with logic, understanding the basic ideas of the science of computing is rather

[6] For a clear discussion of the distinction between correctness and efficiency see Skiena[13, §1.1].

[7] See Moschovakis[7].

[8] See Moschovakis[8] for a mathematical treatment of the idea of *abstract algorithm*.

[9] See Dennett's ideas in [2]. Dennett's idea is that all Design in the universe is the result of „an algorithmic process that weds randomness and selection“. Intelligence is the the product of such an algorithmic process, not the originator of this process.

[10] See Thagard[14].

easy.[11] The real problem, we think, is connected to the question "does the computational turn really bring something new and interesting"? In the following subsections we refer to two contributions to philosophy for which, directly or indirectly, the answer to this question is positive. The concept of algorithm, a central concept for the computational turn, from the perspective of such contributions, does matter for philosophy as analytical philosophy.

2.1 Yiannis Moschovakis about algorithms and meaning

In a series of papers, Yiannis Moschovakis developed an original approach to the concept of algorithm. Moschovakis stresses the idea that algorithms are **semantic** objects. According to him this is quite obvious, but he wants to emphasize the contrast with such syntactic objects as computer programs.[12]

Moschovakis has proposed a formal framework for understanding ideal, abstract algorithms.[13] He also uses, for obvious reasons, a pseudocode for algorithms.[14]

The basic idea in Moschovakis's approach is simple. Let us say, for example, that we examine an usual declarative sentence. This might be "Athens and Bucharest are in the same meantime zone". Then let us examine the algorithm for determining the truth of this sentence. The algorithm remains the same if I translate this sentence into Romanian and professor Moschovakis translates it into Greek. Anyone may use the same algorithm in order to determine the truth of the Romanian or the truth of the Greek sentence.

According to Yiannis Moschovakis, an algorithm such as the one that we have mentioned above is Frege's *sense*. This is precisely the suggestion contained in the title of Moschovakis's paper[9]. Moschovakis argues that the interpretation of Frege's *sense* as algorithm is at the heart of the approach of practically all those who have written about Frege. Obviously, they do not use explicitly the concept of algorithm, but some idea like that of method or procedure for determining the denotation.[15]

[11] One can try to read, for example, Baldwin and Scragg[1].

[12] Computer programs are just texts. See Moschovakis[9, p.212].

[13] See Moschovakis[8].

[14] See, for example, his analysis if the *liar* in Moschovakis[9, pp.212–213].

[15] Moschovakis writes that "the idea that the sense of a sentence determines its denotation is at the heart of the Frege doctrine and practically everyone who has written on the matter uses some form of computational analogy to describe it"[9, p.216].

Moschovakis's idea is very powerful because he is able to translate it into a formal language of recursive algorithms. In such a framework it is possible to discuss precisely the relationships between senses.

2.2 Pavel Tichý and the intensional analysis

The work of Pavel Tichý corroborates and generalizes Moschovakis's observation concerning Frege's *sense*. It is possible to identify in logic and philosophy a clear tendency to connect intensions with algorithms. For Pavel Tichý, *intensions* are procedures. He distinguishes between the logical **type** of procedures and the type of the extensional entities.[16]

Tichý approaches directly the core of the method of analytical philosophy: the formalization of natural language in order to get rid of the traps that it contains. He suggests that a very important step in the formalization process is the disentanglement of the references to extension from the references to intension.[17]

In Tichý's approach there is a strong emphasis on types. But does he really use the concept of algorithm? As far as I can guess from what I know the answer is positive. Of course, one cannot find the **word** algorithm in his 1971 paper, but this is not the point. He writes about the possibility of talking about intensions in terms of Turing machines.[18] This would be different from Moschovakis's concept of algorithm, which is expressly **not** conceived in terms of computing machines. But the approach is exploiting the same idea of a connection between intensions and algorithms.

In Tichý's school of logic the focus is on the distinction between "the object that has been constructed and the way it has been constructed"[19]. The technical term they use for the sense of an expression is *construction*.[20] Constructions are non-linguistic. Distinct linguistic expressions may express the same construction.[21]

[16] See Tichý's arguments in his 1971 paper[15, p.278].

[17] Tichý's dual strategy for the formalization of natural language is clearly delineated in his paper on intensional analysis[15, p.282].

[18] Unfortunately, I had no opportunity to read his paper "Intension in Terms of Turing Machines" published in *Studia Logica*, volume 24.

[19] Duží and Materna[4, p.15].

[20] Duží and Materna[4, p.18].

[21] For the technical definition of *construction* see Duží and Materna[4, pp.26–30].

3 Russell's logical analysis

Let us now go back to the very idea of **analysis** as it is illustrated in the seminal 1905 paper by Bertrand Russell. Consider, for example, the following sentence:

The author of *Waverley* also wrote *Ivanhoe*.

One might be tempted to treat "the author of *Waverley*" as the subject of the above sentence. For Russell, this is a trap and one should avoid it.[22]

According to Russell, in the case of the sentences of a natural language we have to go beyond their surface. "The author of *Waverley*" is dissolved by the logical analysis. First, Russell stresses a strict interpretation of "the".[23] When "the" is involved, a uniqueness condition must be satisfied. Thus, there is one and only one author of *Waverly*. The other condition that must be satisfied is an existential condition. Therefore, when one talks or writes about "the author of *Waverley*" we have to expand logically this expression into "there is one and only one person who wrote *Waverley*"[24].

The most famous example offered by Russell is, of course:

The present King of France is bald.

Applying again the analysis sketched above, we have to realize that the above sentence is expanded into "there is one and only one person who is now King of France and that person is bald". The first part of the conjunction is (in 2004) false, therefore the whole sentence is false.[25]

Russell's analysis may seem strange, but it is not at all unnatural. Try to examine a sentence like:

The present King of Belgium is bald.

This sentence is expanded into "there is one and only one person who is now King of Belgium and that person is bald". The left part

[22] Russell writes that "if I say 'the author of *Waverley* was a man,' that is not a statement of the form 'x was a man,' and does not have 'the author of *Waverley*' for its subject"[11, p.488].

[23] See Russell's paper[11, p.481].

[24] Cf. Russell[11, p.489].

[25] Russell[11, p.484] stresses expressly the idea that the sentence is false. It is not a nonsense.

of the conjunction being true, it all depends on the right side of the conjunction. The idea is however that the two sides have a similar status.[26]

There is a huge literature on the topic of Russell's theory. It does not make sense to summarize it here.[27] As it is natural with a century-old theory, Russell's theory is dismissed by many authors today.[28] We may just mention that for P.F.Strawson "communication is much less a matter of explicit or disguised assertion than logicians used to suppose"[29]; there is a variety of contexts in which we use the natural language and different presuppositions help us to make sense of expressions in these contexts.[30]

4 Algorithmic analysis and denotation

Euclid's algorithm (see §2) computes a **value** for the greatest common divisor of two integers. Given two numbers a and b, it is a way or a procedure for the computation of the **denotation** of the expression "the greatest common divisor of a and b". In this section we discuss a generalization of this observation in the case of linguistic expressions.

What means that I find "the author of the *Peloponnesian War*"? I also have to compute a denotation for this expression. "Thucydides" is simply another **clue** for finding out the same denotation.

In contrast with the approach that establishes a correspondence between algorithms and meaning or sense, we do not treat algorithms as meanings. They are just ways of computing a denotation, in this approach. Consider, for example, the following case:

[26] Critics, like Strawson[10], rejected this kind of similarity between the two sides of the conjunction. Therefore the conjunction itself is illegal and should not be constructed. One should rely on the analysis of the **presuppositions** involved. From a russellian point of view, one might say that the conjunction mixes illegally different types.

[27] Recent introductions into the philosophy of language, like Devitt and Sterelny[3], still provide ample space for Russell's theory and its criticism.

[28] For critical comments see, for example, the introduction to the philosophy of language by Devitt and Sterelny[3, chap.3].

[29] P.F.Strawson[10, p.333].

[30] Russell wrote that he was "unable to see any validity whatever in any of Mr.Strawson's arguments"[12, p.385]. Russell points out that he disagrees with philosophers who are "persuaded that common speech is good enough not only for daily life, but also for philosophy"; he is "persuaded that common speech is full of vagueness and inaccuracy, and that any attempt to be precise and accurate requires modification of common speech both as regards vocabulary and as regards syntax"[12, p.387].

I number the shelves in my personal library; then I examine the expression "the author of the third book on the 22^{nd} shelf". The denotation happens to be again the same as that of "Thucydides". But is this changing something that we might call the **meaning**? I doubt. This would be an endless process with no connection with the denotation.

Let us imagine that I am looking to a picture and I say that I am contemplating the picture of the author of the third book on the 22^{nd} shelf in my library. This is a statement about my actions. It is not adding anything to the properties of the author, despite the fact that someone might use my words as **clues** for finding out that I am talking about a person whom she calls "Thucydides".

In Russell's classical analysis, as well as in the tradition that it inaugurated, the focus is on knowledge about the denotation. Russell's subtle accent is on descriptive knowledge, on the knowledge about something.[31] What knowledge could exist about the present King of France? Obviously, no knowledge, except the knowledge of the fact that there is no individual who is now King in France.

It is not difficult to identify the alternative to the focus on descriptive knowledge. We may focus upon the **procedures** we use for the discovery of the denotation.[32] This is what we propose here.

We illustrate the shift from the focus on descriptive knowledge to the focus on action with the help of two imaginary situations. Let us suppose that we are transported back in time, in Ancient Greece. We are still mainly animated by the passion for knowledge. We examine the truth of the following sentence:

The King of Sparta is lame.

First, the denoting phrase, as Russell would say, "the King of Sparta" is deficient. The problem is not now that there is no king in Sparta. Existence, in this case, is not the problem. The problem is created by the fact that Spartans have two kings. The uniqueness condition is unsatisfied.

[31] This is obvious from the start of Russell[11, p.479]. Russell comes back to the significance of the knowledge without acquaintance in the final part of his paper[11, p.493]. The fact that knowledge is in focus is bracketing the whole study.

[32] In the wittgensteinean tradition one would talk about a change of stress from meaning to use. Our intention, though obviously parallel, is however rather different. We still think that Russell's insistence on the tricky nature of the natural language and the need to go beyond its surface should be taken seriously.

If we think from the point of view of action, the phrase "the King of Sparta" leaves us in an unclear situation. Suppose that you have to give something to "the King of Sparta". Which one?

Let us further imagine that we walk into a room in which, on the shelves, there are all the writings of all the ancient Greek authors. We look for an author called "Xenophon". We ask for help from the keeper of the collection. Which Xenophon? - asks the keeper. There are two authors called "Xenophon".

Now, we remember that the author we are looking for wrote about a Spartan lame king. The keeper shows us the shelf on which we find *Hellenica* and *Agesilaus*. We found our author.

The mystery of the lame king is solved too if we refer somehow to Agesilaus. Agesilaus was a Spartan King and he was lame. Xenophon wrote about him. There is, no doubt, knowledge involved in all this. But, if we put action in focus, we see linguistic constructions rather as sets of clues than elements of a system of truth-bearers. We use clues that can lead us to Agesilaus or the author who wrote about him. The ability to use linguistic clues may well be illustrated by the hypothetical dialogue with the keeper.

If we put descriptive knowledge in focus, the keeper needs an adequate description in order *to know* what we want. If we put action in focus, the keeper just needs enough **clues** in order to provide us with the clues that we need. "Xenophon" and "the author who wrote about a lame King of Sparta" are good clues for the keeper of the collection. Simple words like "left", "right", "above" or "below" can be used as clues that help us find on the shelves the works we are looking for.

Before we go into the details of the algorithms involved, we should observe that we must distinguish between various actions in which we use sets of linguistic clues. For example, I do not live (now, at least) in France. I might go to France or use some other method to find out facts concerning present-day France. I might, on the other hand, listen to stories about a mythical France. With respect to the mythical France, I would **imagine** a lot of things, an action which is very different from that of searching and finding mere facts.

4.1 Looking for the present King of France

Russell is like someone who would send a search-team in France and expects a report. Writing the report is like the postcondition of an

algorithm. But in order to be able to write the report the team has to fulfill a precondition. The algorithm would be:

Precondition: *The existence of an unique individual who is King of France.*
The body of the algorithm.
Postcondition: *A report about the King of France.*

The problem with this algorithm is that the precondition (the preliminary requirement) conflicts with the postcondition. It would be impossible to generate any genuine report if there is no King in France.[33]

Let us modify the algorithm:

Precondition: *The existence of at least one individual in France.*
***While** you have not found a King of France or you have not examined all the individuals in France*
If** you have found a King of France **then
***For all** the other (yet unexamined) individuals in France*
Check if there is another king.
End for.
If** there is no other king **then
Check if the King of France is bald.
Make a report.
Else
Report that France has two or more kings.
Report that the sentence "There is one and only one individual who is now King of France" is false.
End if.
Else
Report that the sentence "There is an individual who is now King of France" is false.
End if.
Examine the next individual in France.

[33] Moschovakis[9, p.225] writes about the "programmer's solution" to Russell's problem. He asked four friends the following question: *do you think that the King of France is bald?*. Three of them answered "there is no King of France". I interpret this as a focus on preconditions. The problem with this answer is that it misses the point: which is the truth-value of the sentence "the present King of France is bald". It is symptomatic however that the majority seems to try instinctively to design some algorithm.

End while.
Postcondition: *A report about the truth or falsity of the sentence "the present King of France is bald".*

There is a problem with the complexity of this algorithm. There are a lot of individuals in France. In the worst case, the search team has to examine the situation of each of all these individuals. The best case would be to find, after examining two individuals, that France has two kings (like ancient Sparta). This algorithm is obviously unnatural. Most of us would probably proceed like this:

Precondition: *The existence of a clear (written or unwritten) constitution of France.*
If** there is no king in France (according to the constitution) **then
Report that the sentence "There is an individual who is now King of France" is false.
***Else if** there is a King of France*
Find the king.
If** the king is bald **then
Report that the king is bald.
Else
Report that the king is not bald.
End if.
End if.
Postcondition: *A report about the baldness of the King of France.*

4.2 Dreaming about a king of present-day France

Fiction seems to generate a lot of problems, if we are focused on knowledge. Suppose that we talk about an author who lives in present-day France and writes in Ancient Greek. There are persons in France who write in Ancient Greek, but the individual I imagine is an entirely fictitious person. The existence condition is not satisfied. Probably, there are at least some difficulties with the uniqueness condition too. Anything I say primarily[34] about this

[34] Anything I say in sentences in which "the French author who writes in Ancient Greek" has a primary occurrence; i.e. is a direct ingredient of the sentence. Russell[11, p.490] stresses the importance of the distinction between primary and secondary occurrences (occurrences in sentences that are components of a complex sentence). He admits that the distinction is not quite clear in the case of natural languages. It is easy to make it in symbolic languages.

author is bound to be false. It would be quite ridiculous, in a literary text, to use secondarily the denoting phrase and announce repeatedly that I am talking about a fictitious character.

Would things look differently in algorithmic perspective? Let us reexamine the algorithms discussed in section 4. The precondition must be modified: we must allow some room for imaginary situations. The postcondition has also to undergo some changes: if we want a report, then we must admit reports about imaginary situations. The most spectacular changes take place however in the algorithm itself.

In an imaginary situation it does not make sense to look for "the King of France". We add such an individual to the imaginary situation. We change the state of the world in order to make room for that individual.

Let us look to a possible algorithm:

> **Precondition:** *The possibility to imagine that there is at least another person added to the set of persons living in France.*
>
> ***Start*** *with the introduction of a new person.*
> *Imagine that this is a king of France.*
> *Make sure that there is no other king of France.*
> ***If*** *you have imagined that the King of France exists* ***then***
> *Imagine that the King of France is bald.*
> *Make a report about the imaginary King of France.*
> ***End if.***
>
> **Postcondition:** *A report about the imaginary situation of the King of France.*

Surprisingly, this algorithm is less complex than an algorithm which would force us to examine the situation of all the individuals in France. The complexity is similar to the algorithm in which we examine the constitutional situation, but there are less decisions to be taken. I might even eliminate the "if" from it; the algorithm is just a series of actions to be taken by someone who imagines a France with a King. This is not surprising: the whole operation is like an imaginary revision of the Constitution.

The advantage of the algorithmic perspective is that it stresses the fact that we **construct** the imaginary situation. An algorithm specifies the steps that we take during the construction process.

We use imaginary situations beyond pure fiction. We make imag-

inary experiments. But in imaginary experiments we do not want to accumulate knowledge about the imaginary situation as such. This does not make much sense, since we stipulate it. We want to investigate the consequences of the situation. These consequences might or might not corroborate the theory that we test.

From the algorithmic point of view, we are focusing on the construction process. Think about a play in a theater. We watch the actions of the characters, we are not interested in truths about the characters. If one knows after a few dialogues who is the bad guy and who is the good guy this is a sign of a rather stupid play. The distinction between the algorithm and the algorithmic process is here really helpful. Each performance of the play is like a new algorithmic process. If the algorithm has a lot of decision points in it, each time we reread the text there is the possibility to unfold a different algorithmic process.

4.3 Looking for ghosts

Russell rejected Meinong's objects. There are now many authors who have gone back to Meinong and try to figure out how a logical system that accommodates "round squares" and similar things looks like. My idea is rather different. Let us look to what happens if we modify the algorithm for the King of France and try to find out round squares.

Let us examine the following algorithm:

Precondition: *The examination of all possible squares.*
While *you have not found a suitable square or you have not examined all the squares* **do**
Check the current square.
If *the square is round* ***then***
You have found a suitable square.
Else
You did not find a suitable square.
End if.
Examine the next square.
End while.
Postcondition: *A report about squares.*

Despite the superficial similarity with the algorithm which checked the situation of all the persons in France, this algorithm has a nasty flaw. The algorithmic process will never halt.

First, if we assume that "squares" refers to normal, usual squares, there is an infinity of such squares and there is no chance to find a round square among them (as Russell pointed out long ago, this would be logically contradictory).

Second, if we enlarge somehow the set of possible squares with some unusual squares, there would be no procedure to check those squares. You have to believe in some unusual checking procedures if you say that you check such squares. This would push you out of the realm of mathematics into the world of linguistic tricks. After all, those who believe that "round square" refers to some kind of object do not believe that such objects subsist.

It is impossible to halt the above algorithm, it loops for ever. The algorithm is obviously wrong. There is nothing however that prevents us from talking about this algorithm. This is perfectly legitimate talking.

Would it be possible to use the same strategy as in the case of the King of France and examine the "constitution"? This would engage us in a debate about the "logical legislation" that applies here. If we stick to good old classic logic, "round square" is a contradiction and it should be avoided. If we adopt some fancy logic, the situation might look quite different, but it is not the aim of this paper to engage in such debates.

One way or another, the solution that we suggest is: talk about the algorithm! Do not bypass the algorithm and try to discuss the denotation as an object in itself. The denotation makes sense only if we can (or we cannot) illuminate it with the help of some algorithm.

5 The algorithmic turn

According to Bertrand Russell philosophical theories are tested using logical puzzles. The use of puzzles resemble the use of experiments in natural sciences. The capacity to deal with puzzles is similar to the capacity of a theory in physical science to pass empirical tests.[35]

5.1 What did the King want to know?

Let us now consider the famous sentence proposed by Bertrand Russell:[36]

[35] See Russell[11, pp.484–485].
[36] Russell[11, pp.485].

> George IV. wished to know whether Scott was the author of *Waverley*.

In the algorithmic interpretation of this sentence, "the author of *Waverley*" is the blueprint for an algorithm. This algorithm computes a denotation. The King George IV. wanted to know whether the result of this computation has the name "Scott".

What happens with the puzzle? The logical difficulty appears when we substitute "the author of *Waverley*" with "Scott". The substitution is based on the assumption that both the denoting phrase and the proper name have the same denotation. This substitution makes no sense however from an algorithmic point of view. "The author of *Waverley*" as such has no denotation whatsoever. It is simply the blueprint for an algorithm. The King wanted to know where leads the algorithm.

The name "Scott" raises its own problems. This paper has no intention to discuss them. We assume that, in the puzzle, the King already knows how to compute the denotation of the name "Scott". He might have known that Scott wrote *Ivanhoe* or something similar. This seems to be Russell's opinion. He might have known how to use the label "Scott" as a clue in order to find a certain individual.[37]

In the algorithmic interpretation, the puzzle involves a logical error. The question is **about** an algorithm. Does the algorithm lead to a certain result? We cannot replace this question with a question about the result itself.

What happens if we interpret the whole substitution as the replacement of an algorithm with another algorithm? The logical mistake is still there. I may replace one algorithm with another algorithm if I am just interested in getting the same result in a different way. The replacement is wrong if I want to know where the algorithm is leading to. In fact, the whole question might be about the legitimacy of replacing one algorithm with another algorithm. The substitution begs the question.

Somebody may object to the approach above and wonder if the restrictions on substitution are not too drastic. Actually, they are

[37] John Stuart Mill compares such an operation with the chalk mark that thieves in *1001 Nights* leave on Ali Baba's house. Morgiana confuses them by making the same sign on all the houses in the neighborhood. The thieves are lost since they use the mark as a clue (that plays a key role in the algorithm they use for finding the house). See John Stuart Mill's *A System of Logic*, Book 1, chapter II, §5 (I have used the French translation of the sixth edition, *Système de logique*, vol.I [Paris: Alcan, 1889]).

less severe than the usual restrictions recommended in logic. Consider the following sentence:

> X wants to know when did the author of *Waverley* live.

It all depends on the knowledge of X. If X knows that the author of *Waverley* is Scott, then the substitution raises no problems. X simply does not know when did Scott live.

It is a perfectly benign strategy to answer a question like "when did the author of *Waverley* live" by making first a substitution. Then one may look into an encyclopedia and come up with a time interval: 1771–1832.

The logical analysis as formalization translates vague sentences from the natural language into a symbolic language. It is then easier to analyze the arguments involved.

The scheme of algorithmic analysis follows, up to a point, a similar pattern: it replaces vague indications in natural language with algorithms in pseudocode. Then it is possible to refine the code and translate it into a formal language for algorithms.

The approach used here departs from Russell's approach far less than the theories which claim that natural language is all right and needs no translation into a symbolic language. Summing up, the approach we would favor is the following: don't look for the meaning, analyze the corresponding algorithmic process.

5.2 The new clothes of the King of France?

The approach suggested here seems to go straight into a formidable obstacle. Let us say that someone exhorts you to write a good paper. What algorithm corresponds to her sentence?

There is obviously no direct algorithmic translation for "write a good paper!", because we cannot specify exactly the steps of the respective action. But this does not prevent us from **understanding** the respective sentence. First, it is (relatively) easy to use clues in "write a paper" for designing a sequence of actions that leads to the production of a paper. What is going to count as a **good** paper is much less definite. I think that, as with personal identity, branching becomes inevitable in the case of algorithmic identity. There are many ways in which a paper could be good.

Let us think about another possible instruction to do something:

Write a nice story!

This imperative sentence also lacks clear algorithmic identity. Context however might help. Let's imagine that a journalist hesitates. She has to write a story about a politician. She might tell the truth, but this implies the revelation of some unpleasant details. Her boss has an advice for her. He tells her to write a nice story. The tone of the of voice, in this case, is also helpful! We have no problem however to **imagine** the whole scene, to see it as a theatrical performance in which each character exhibits some patterns in her/his actions.

References

[1] Douglas Baldwin and Greg W. Scragg. *Algorithms and Data Structures: The Science of Computing.* Hingham, Mass.: Charles River Media, 2004. Quoted at pp. 5 and 8.

[2] Daniel C. Dennett. Evolution as an algorithm. In Harold Morowitz and Jerome Singer, editors, *The Mind, the Brain and Complex Adaptive Systems*, pp. 221–223. Boston: Addison-Wesley, 1995. Quoted at p. 7.

[3] Michael Devitt and Kim Sterelny. *Limbaj şi realitate* (translated by Radu Dudău). Iaşi: Polirom, 2000. Romanian translation of *Language and Reality: An Introduction to the Philosophy of Language*(Oxford: Blackwell, 1999). Quoted at p. 11.

[4] Marie Duží and Pavel Materna. *Constructions* [on the site dedicated to Transparent Intensional Logic]. <`www.phil.muni.cz/fil/logika/til/constructions_duzi_materna.pdf`>, 2000, visited at: 2004/09/30. Quoted at p. 9.

[5] Donald E. Knuth. *Arta programării calculatoarelor*, volume 1. Bucureşti: Teora, 2000. Romanian translation of *The Art of Computer Programming, Volume 1: Fundamental Algorithms*(Reading: Addison-Wesley, 1998). Quoted at pp. 4, 5, and 6.

[6] Daniel Marcu and Graeme Hirst. An implemented formalism for computing linguistic presuppositions and existential commitments. In *International Workshop on Computational Semantics*, pp. 141–150. Tilburg: 1994. <`http://www.isi.edu/~marcu/papers/exist-tilburg-94.ps`>. Quoted at p. 4.

[7] Yiannis Moschovakis. *Mathematical Logic* [posted on the website of the author]. <`http://www.math.ucla.edu/~ynm/`>, visited at: 2004/10/11. Published in the *Encyclopedia of Physical Science and Engineering*. Quoted at p. 7.

[8] Yiannis Moschovakis. The formal language of recursion. *The Journal of Symbolic Logic*, 54(4): 1216–1252, December 1989. Quoted at pp. 7 and 8.

[9] Yiannis Moschovakis. Sense and denotation as algorithm and value. In J. Oikkonen and J. Vaananen, editors, *Lecture Notes in Logic*, volume 2, pp. 210–249. New York:

Springer, 1994. <http://www.math.ucla.edu/~ynm/papers/frege.pdf>. Quoted at pp. 8 and 14.

[10] P.F.Strawson. On referring. *Mind*, 59(235): 320–344, July 1950. Quoted at p. 11.

[11] Bertrand Russell. On denoting. *Mind*, 14(56): 479–493, October 1905. Quoted at pp. 3, 4, 10, 12, 15, and 18.

[12] Bertrand Russell. Mr.Strawson on referring. *Mind*, 66(263): 385–389, July 1957. Quoted at p. 11.

[13] Steven S. Skiena. *The Algorithm Design Manual*. New York: Springer, 1998. The book has an extended site on the Internet at <http://www.cs.sunysb.edu/~algorith>. Quoted at p. 7.

[14] Paul Thagard. *Computation and the Philosophy of Science* [posted on the site of the author]. <http://cogsci.uwaterloo.ca/Articles/Pages/comp.phil.sci.html>, visited at: 2004/09/30. Quoted at p. 7.

[15] Pavel Tichý. An approach to intensional analysis. *Noûs*, 5(3): 273–297, September 1971. Quoted at p. 9.