

TRACTABLE COMPETENCE

MARCELLO FRIXIONE

*Department of Communication Sciences, University of Salerno, Italy
via Ponte Don Melillo
I-84084 Fisciano (Salerno) - Italy*

*e.mail: frix@dist.unige.it
<http://www.dif.unige.it/epi/hp/frix/hpf.htm>*

Abstract In the study of cognitive processes, limitations on computational resources (computing time and memory space) are usually considered to be beyond the scope of a theory of competence, and to be exclusively relevant to the study of performance. Starting from considerations derived from the theory of computational complexity, in this paper I argue that there are good reasons for claiming that some aspects of resource limitations pertain to the domain of a theory of competence.

Key words: Competence vs. performance, computational complexity, levels of cognitive explanation.

1. Introduction

The distinction between *competence* and *performance* was originally formulated by Noam Chomsky in the field of linguistics, with special reference to the syntax of natural languages. The competence/performance opposition is well summarised in the following passage on the initial pages of *Aspects of the Theory of Syntax*:

Linguistic theory is concerned primarily with an ideal speaker-listener, in a completely homogeneous speech-community, who knows its language perfectly and is unaffected by such grammatically irrelevant conditions as memory limitations, distractions, shifts of attention and interest, and errors (random or characteristic) in applying his knowledge of language in actual performance [...]. To study the actual linguistic performance, we must consider the interaction of a variety of factors, of which the underlying competence of the speaker-

hearer is only one. [...] We thus make a fundamental distinction between *competence* (the speaker-hearer's knowledge of his language) and performance (the actual use of language in concrete situations). [Chomsky (1965), pp. 3-4].

According to Chomsky, linguistic competence consists in an *idealised* speaker's knowledge of his/her language. Such an idealised competent speaker is not constrained by extralinguistic factors. Limitations on computational resources are usually included in such extrinsic constraints. In the above passage, Chomsky explicitly refers to memory limitations, but similar considerations can be extended to limitations on processing time.

Originally concerned with the syntax of natural languages, the notion of competence subsequently assumed major influence in the study of cognitive phenomena, and was brought to bear on different areas of cognitive sciences. In the field of linguistics, various notions of *semantic competence* have been proposed; in many cases, they have been derived from formal theories of meaning [Cresswell (1978), Partee (1982), Chierchia and McConnell-Ginet (1990), Larson and Segal (1995)]. A theory of *lexical competence* has been developed by Diego Marconi (1997). Other notions of competence have also been applied to cognitive tasks different from language, e.g., logical or probabilistic reasoning [Cohen (1981), Johnson Laird and Byrne (1991)]. A review of various notions of competence, along with their classification in terms of their computational properties, has been provided by Frixione and Tamburrini (1996).

In this paper, I shall argue the following thesis: if we adopt a computational approach to the study of cognitive phenomena, then there are good reasons for maintaining that some aspects concerning resource limitations pertain to the domain of a theory of competence. Such a conclusion stems from considerations based on the results of the theory of computational complexity. In a nutshell, my argument can be summarised as follows: on the one hand, the properties related to computational complexity concern the *function* that is calculated and not the *algorithms* that calculate it; on the other hand, it is reasonable to assume that the considera-

tions about the functions modelling cognitive abilities fall in the domain of a theory of competence.

Therefore, there are aspects of resource limitations that can be accounted for within a theory of competence. On such a basis, a notion of *tractable competence* can be developed, that is, a notion of competence constrained by considerations on computational tractability. In my opinion, the importance of this move can be justified on different grounds. First, it imposes the adoption of more demanding criteria on the adequacy of competence theories, and enforces the development of models of competence endowed with a greater explicative value and a greater empirical content. The cost of computational resources is a crucial aspect of cognition if we take seriously the computational approach to the study of the mind; if we adopt a notion of tractable competence, this aspect is not entirely relegated to the domain of performance. In addition, a notion of tractable competence allows us to distinguish between rather different aspects of resource limitations, i.e., the aspects that are determined by competence factors and the aspects that depend exclusively on performance.

The rest of the article is organised as follows. In the next section, I outline the particular notion of competence that I shall use in the following, and I argue that the conclusions I shall draw on this basis are general enough to be relevant to the majority of the notions of competence proposed in the literature. Section 3 presents two examples of tasks that are «difficult» in terms of resource limitations; they are aimed to distinguishing between cases in which resource limitations affect exclusively performance and cases in which resource limitations could pertain to a theory of competence. In order to justify this distinction, in Section 4 I give some basic notions of the theory of computational complexity. In Section 5, the notion of tractable competence is discussed. Section 6 reviews some positions that have been presented in the literature and that are in some respect related to the theses of the present article. Finally, Section 7 is devoted to some concluding remarks.

2. Competence theories and computational theories

David Marr (1977) proposed a general framework for the analysis of the levels of explanation in cognitive sciences; it helps in situating the notion of competence within the overall study of cognition. According to Marr, the computational account of a cognitive phenomenon can be formulated at three different levels: the level of the *computational theory* (also called *level 1*), the *algorithmic level* (*level 2*), and the *implementation level* (*level 3*) [Marr (1982), Chapter 1]. The level of the computational theory is the most abstract; it concerns the specification of the task associated with a given cognitive phenomenon. At this level, cognitive tasks are characterised only in terms of their input data, the results they produce, and the overall aim (the «goal») of the computation, without any reference to the specific processes and cognitive mechanisms involved. In other words, at the level of the computational theory, a cognitive task is accounted in terms of a purely functional correspondence (mapping) between inputs and outputs. The algorithmic and implementation levels concern, to different degrees of abstraction, the realisation of the task described at the level of the computational theory. The aim of the algorithmic level is to specify «how» a certain task is performed: it deals with the particular procedures that are carried out and with the representation structures on which such procedures operate. The implementation level is concerned with the features of the physical device (e.g., the structures of the nervous system) that implement the representations and the procedures singled out at the algorithmic level. The relation existing between a computational theory and the algorithmic level can be regarded as the relation between a function (in a mathematical sense) that is computable and a specific algorithm for calculating its values. The aim of a computational theory is to single out a function that models the cognitive phenomenon to be studied. Within the framework of a computational approach, such a function must be effectively computable. However, at the level of the computational theory, no assumption

is made about the nature of the algorithms and their implementation.

According to Marr, the level that pertains to a theory of competence is level 1. For example, he states that: "Chomsky's (1965) notion of a 'competence' theory for English syntax is precisely what I mean for a computational theory for that problem" [Marr (1977), in Haugeland (1981), p. 130]. A theory of syntactic competence in sense of Chomsky would be a special case of computational theory in the sense of Marr.

Marr's characterisation of competence is partially different from the Chomskian notion. However, for reasons that will be clear in the following, it is well suited for the argument I am going to develop in this paper, and does not involve any loss of generality. Let us consider the differences between Marr's and Chomsky's positions in this regard. According to Marr's characterisation, two extensionally equivalent grammars (i.e., two grammars that generate the same language, and that cannot therefore be distinguished in terms of input-output relations) would be indiscernible from the point of view of a level-1 theory. However, Chomsky maintains that extensionally equivalent grammars may turn out not to be equivalent as descriptions of human syntactic competence. Therefore, a Chomskian theory of competence seems to represent something more than a mere level-1 theory, i.e., it seems to include some aspects that, according to Marr, would pertain to the algorithmic level of analysis.

According to Cristopher Peacocke (1986) not only the Chomskian notion of competence, but even the theories developed by Marr in the context of his actual scientific practice do not fully correspond to Marr's level 1 (i.e., to a purely extensional characterisation of the functions computed by the cognitive system). Rather, they correspond to a further level, which Peacocke calls level 1.5. This level turns out to be intermediate between level 1 (computational theories) and level 2 (algorithmic theories). For example, suppose that the same cognitive function can be characterised through two different formulations: a mere look-up table and a formulation that, in some way, suggests how the function

should be cognitively computed (and that is therefore more akin to level 2). At the level of a theory of competence the latter choice is preferred, even if, in terms of a pure level 1 theory, both formulations are fully equivalent.

May be the Marrian notion of competence is too "austere" as compared with the Chomskian one. One may wonder whether a theory of competence in the sense of Chomsky is concerned *only* with the identification of the computed function (and probably, as said before, this is not the case). One may even wonder whether Marr's "computational" theories are really level-1 theories or they are to be placed at Peacocke's level 1.5. However, for the purposes of the present discussion, it should be noted that all the aspects of the competence *sensu* Marr can be expected to be important also for the Chomskian notion of competence (or for level-1.5 theories). In particular, the considerations concerning the input-output functional relations are surely relevant also to the Chomskian competence, even though they do not exhaust it. As a consequence, for our present aim, the adoption of a notion of competence in the sense of Marr does not entail any loss of generality: if some aspect is relevant to the Marrian notion of competence, then, *a fortiori*, it should also be relevant to the Chomskian notion of competence. From the point of view of this paper, the Marrian definition has the advantage of isolating a crucial aspect, clearly defined in formal terms: a theory of competence for a given cognitive task must be concerned (at least) with the *function* (in a mathematical sense) modelling the task.

Such considerations can be extended to the positions mentioned in the previous section, which generalise the competence/performance dichotomy to fields of cognitive sciences different from the syntax of natural languages. In computationally oriented accounts, also some aspects of the algorithmic level are usually assumed to be relevant to the characterisation of competence. However, the fact that a theory of competence should *at least* account for the functional mapping between the inputs and

the outputs of the studied cognitive task could hardly be questioned.¹

3. Two difficult tasks

As said before, limitations on computational resources have traditionally been considered irrelevant to the level of competence, and confined to the sphere of performance. Of course, there are aspects of resource limitations that exclusively concern a theory of performance. However, there are good reasons for hypothesising that some aspects of computational resource limitations pertain to the level of competence.

Let us consider two examples of tasks that are «difficult» in terms of required computational resources, and that could be considered paradigmatic of the two cases mentioned above. The first task, let us call it (*a*), consists in memorising and repeating on the spot a sequence of natural numbers, for instance, ranging between 1 and 1000. If the numbers to be memorised are many (e.g., fifty), the task becomes so difficult as to be prohibitive for any human being. Indeed, the storage capacity of our short-term memory is much more limited. On the basis of a series of experiments concerning different cognitive tasks, psychologist G.A. Miller (1956) showed that the short-term human memory has a capacity of about seven distinct elements (on this regard, Miller speaks of the «magical number seven plus or minus two»).

The second task, let us call it (*b*), is the well known *SAT* problem, that is, the classical problem of deciding whether a pro-

¹ According to some standpoint in the literature, it is not mandatory that the input/output mapping in a theory of competence should be modelled in terms of a *computable* function. Typical in this respect is the opinion of Cresswell (1978). This author claims that some form of model theoretical account of natural language semantics could play the role of a theory of semantic competence. In this case, the judgements of logical entailment between sentences would play the same role as the judgements of grammaticality in a theory of syntax. According to Cresswell, the fact that logical entailment is, in general, undecidable does not constitute a problem in this respect. It is hard to understand how such a permissive notion of competence could turn out to be useful from a cognitive point of view. However, even according to such a position (and *a fortiori*, as no algorithmic constraint is available in this case) the functional mapping between inputs and outputs is relevant in characterising competence.

positional formula is satisfiable (or, in other terms, whether it is not a contradiction). Let F be a propositional formula, and let p_1, \dots, p_n be the n distinct propositional letters occurring in it. In order to decide on the satisfiability of F , let us assume to use an algorithm based on truth tables. If F has n different propositional letters, the truth table for F has 2^n rows. So, if $n = 10$, the rows of the truth table for F are $2^{10} = 1.024$, if $n = 20$, the rows are $2^{20} = 1.048.576$, if $n = 30$, the rows are $2^{30} = 1.073.741.824$, and so on. In the worst case, in order to establish whether F is satisfiable or not, all the 2^n rows of the truth table must be considered. Therefore, in such a case, the time required to compute the solution is exponential with respect to the number of different propositional letters of the formula. If F has 30 distinct propositional letters, then, in the worst case, more than one billion rows must be checked. This surely exceeds the abilities of actual cognitive agents.

Both tasks (a) and (b) become quickly prohibitive for any human subject. However, the «difficulty» of task (a) is only due to the contingent limitations of the device that performs it (in this case, the limited storage capacity of the short-term human memory). The task per se is not *intrinsically* difficult. It is straightforward to design a device that performs it efficiently. And if it were possible to manipulate the short-term memory of a human being in order to extend its capacity, the task would become trivial even for him/her.

On the contrary, task (b) turns out to be difficult for every computational device. Whenever the number of distinct propositional letters of the formula F is increased by one element, the required computational steps are doubled. Obviously, even in the case of task (a), for every actual machine there always exists a sequence of numbers that is too long to be memorised. But a small increase in the machine capabilities is sufficient to overcome this limitation. In the case of tasks like (b), also small increments of the problem dimensions involve large increases in the resources needed. Soon, even for data whose dimensions seem to be fully «reasonable,» the required computing resources become impracti-

cable. For example, even supposing that available computing devices could reach the maximum speed that can be physically achieved, the duration of computations would soon exceed the lifetime of the entire universe.² In this sense, tasks are usually assumed to be *computationally intractable*, if they require computational resources (time and space of memory) growing exponentially with respect to the length of the input (measured in some suitable way; for example, in the case of *SAT*, the length of the input is given by the number of distinct propositional letters in the formula F). Tasks are assumed to be *computationally tractable*, if they require resources that grow at most polynomially with respect to the length of the input.

It could be objected that considerations like that on the lifetime of the universe concern purely contingent facts, that have nothing to do with in principle limitations. For example, it is easy to define computational tasks that are tractable according to the above definition, but that require prohibitive computational resources even for very small inputs. However, here the main issue is that for a task like (b) the ratio of the needed computation time to the input length grows so quickly as to suggest that, in a sense, the task itself, in its general form, cannot *in principle* be accomplished. In other words, hypothesising a device for a task like (b) seems to require a higher degree of idealisation than hypothesising a device for a task like (a) .

In addition, it could be objected that algorithms based on truth tables would not be a good choice to decide on the satisfiability of a propositional formula, and that more clever algorithms would allow things to go better. In a sense this is true: an algorithm based on truth tables is particularly inefficient, and more appropriate algorithms exist to solve the task. However, in the worst cases all known algorithms require computing times of the order of 2^n steps, where n is a suitable measure of the length of the formula.

4. Classes of computational complexity

² See, for example, Cherniak (1986), § 4.9.

The key point in the examples of Section 3 is that a task like (b) is likely to be *intrinsically* difficult, that is, its difficulty depends on the problem itself (i.e., on the computed function) and not on the algorithm chosen to solve it. This statement can be justified by introducing some notions developed within the theory of computational complexity.³

Let P be the class of decision problems (i.e., problems admitting a «yes or no» answer) that can be solved in polynomial time by using a deterministic Turing machine (from now on, DTM). A DTM is a usual Turing machine in which each computational step is fully determined by the configuration (i.e., the pair formed by the internal state of the machine and the observed symbol on the tape). Let NP be the class of decision problems that can be solved in polynomial time by a *non*-deterministic Turing machine (from now on, NTM). An NTM is a Turing machine whose table may comprise more than one quintuple (i.e., more than one instruction) corresponding to each machine configuration. As a consequence, an NTM starting from a certain input can produce more than one possible sequence of computational steps. Given the input and an initial configuration, in general there will exist a finitary tree of possible computations, each branch of the tree corresponding to a non-deterministic choice between alternative instructions. A decision problem is said to belong to NP if there exists an NTM such that, for every input, if the answer to the problem is «yes», there exists at least one computation (i.e., one

³ The present section reports some results of the theory of computational complexity. Such results are treated in depth in the specialistic literature. For example, Stockmeyer (1987) provides a synthetic introduction to computational complexity theory; more complete presentations are given by van Leeuwen (1990) and Papadimitriou (1994). Cadoli (1995) and Nebel (1996) discuss the role of complexity theory in artificial intelligence.

This section has been conceived to point out the aspects that are relevant for the arguments developed in this paper (for example, the discussion on parallel computation at the end of this section recalls the discussion on connectionism in Section 5). I have chosen to include this synthetic review in the article to make it self-contained also for readers that are not acquainted with such topics. Readers familiar with computational complexity theory can skip this section without loss of information.

"branch" of the tree) of polynomial length in the input that generates the answer.

The import of the above definition can be better explained by intuitively showing that *SAT* is an NP problem. Let TM_{SAT} be an NTM that, given as input a propositional formula F suitably coded, acts as described in the following in order to decide whether F is satisfiable or not. For each propositional letter p_i in F , TM_{SAT} non-deterministically assigns a truth value, true or false, to p_i . Then, TM_{SAT} produces a different sequence of computational steps (a different «branch» in a tree of possible computations) for each possible assignment of truth values to the propositional letters of F (i.e., for each row of the truth table of F). For each of such assignments, TM_{SAT} verifies if it satisfies F . If F is satisfiable, there must exist at least one of such computations that produces a positive answer. Deciding whether a given assignment of truth values satisfies or not a propositional formula takes a linear (and hence polynomial) time with respect to the length of the formula. As a consequence, each possible computation produced by TM_{SAT} has polynomial length with respect to F . Therefore, if F is satisfiable, then TM_{SAT} will provide at least one computation of polynomial length that decides the problem (in particular, this is the case of all computations in which TM_{SAT} «guesses» a right assignment, i.e., an assignment that satisfy F).

From the above definitions follows that every DTM is also an NTM. Therefore, all problems belonging to P are also in NP, i.e.,

$$P \subseteq NP.$$

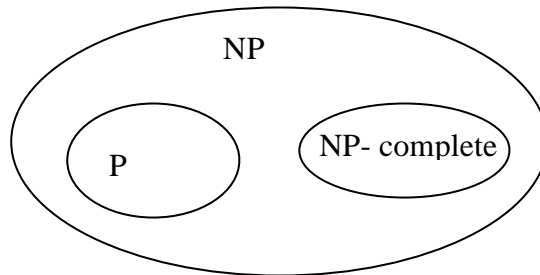
The question whether the inverse inclusion holds, and hence the question whether $P = NP$, is still an unsolved problem. There are, however, good reasons for maintaining that the identity of P and NP does not hold. Let us consider the evidence supporting this conjecture.

We say that an NP problem p is *NP-complete* if every other problem in NP can be reduced to p (in polynomial time), in the sense that, intuitively, if we had an «efficient» algorithm for p , we could obtain (in polynomial time) an efficient algorithm also for every NP problem. The class of NP -complete problems can be considered as the class of the «most difficult» problems in NP . It can be proved that *SAT* is NP -complete (for historical reasons, it can be considered, in a sense, as a «paradigmatic» example of this class). From the definition of NP -completeness, it follows that all NP -complete problems are reciprocally reducible. As a consequence, it would be sufficient to find a tractable algorithm for any NP -complete problem in order to have tractable algorithms for *all* NP -complete problems (and, accordingly, for all the NP problems). As a matter of fact and in spite of the lasting and systematic efforts of the research community, no algorithm has ever been found that solves some NP -complete problem in polynomial time. Therefore, it is usually conjectured that:

$$(*) P \neq NP,$$

even though no proof of (*) is currently available.

Under the assumption that the above conjecture holds, the situation would be the following:



In short, there would exist problems in NP (and NP-complete problems are among them) that, for their intrinsic nature, would not be computationally tractable, i.e., for such problems no algorithm would exist, that could solve them within reasonable bounds on resources.

For our present discussion, it is worth noting that the above considerations hold for *problems*, not for specific algorithms, nor for particular models of computational devices. Indeed, it is usually assumed that the above-mentioned results are independent of the specific way of characterising computational processes. Such an assumption stems from the fact that all classes of «reasonable» computing machines that have been studied until now have turned out to be equivalent in this respect, in the sense that, given any two classes of "reasonable" machines, for each machine of the first class, there exists a machine of the second class that can simulate it with at most a polynomial overhead, and vice versa. Therefore, it is unlikely that some class of machines that falsifies this hypothesis will be found in the future.⁴ The above assumption is called the *Invariance Thesis*. Intuitively, the Invariance Thesis can be stated by saying that, *if a problem is not computationally tractable with a (deterministic) Turing machine, then it is not computationally tractable with any reasonable computational device*. From the point of view of its epistemological status, the In-

⁴ Of course, it would be necessary to clarify what it is meant by the term «reasonable» machine in this context. For sake of brevity, I cannot examine this topic in depth. It was discussed, for example, by van Emde Boas (1990).

variance Thesis is similar to the Church Thesis: it is not a conjecture, in that it is not a proposition that some day may be proved and so become a theorem. Rather, it is a hypothesis that, at least in principle, could be falsified, but for which a great deal of positive evidence has been accumulated that makes this possibility very unlikely.

Therefore, the assumption under which NP-complete problems (like *SAT*) are not computationally tractable rests on two premises: 1) conjecture (*) (because in the case of $P=NP$, NP problems would be tractable, in that a DTM would exist that could compute them in polynomial time) and 2) the Invariance Thesis (which excludes the existence of some computational device that can efficiently compute problems that are not computationally tractable with a DTM).

The Invariance Thesis could be falsified if, for example, quantum computers, as theorised by Deutsch (1985), would turn out to be adequate and realistic models of computation, and if, in addition, quantum TMs would turn out to be able to efficiently compute tasks that cannot be computed in polynomial time by any DTM.⁵ It should be noted that, in this case, the Invariance Thesis would be falsified, but this would be insignificant for the plausibility of (*) (which is a conjecture about the capabilities of DTM).

Two observations are needed. First, in accordance with the definitions given above, P and NP classes include only decision problems (i.e., problems that admit exclusively "yes or no" answers). Such definitions can be generalised to include other types of problems (e.g., search problems or, more generally, computable functions of any kind). All the results I have previously stated holds true also in this more general case. Secondly, computational intractability does not affect only NP-complete problems. The class of NP-complete problems is of particular interest because it includes many problems that are important from both the theoretical and the practical points of view. However, there are other

⁵ Some evidence that seems to corroborate a similar hypothesis was obtained, for example, by Shor (1997). There are other results, however, that limit this possibility (see Bennett *et al.* (1997)).

classes of problems that have been assumed to be computationally intractable (and harder than NPs). There are also classes of problems for which it has been *proved* (independently of any conjecture) that they cannot be solved in polynomial time by any DTM. If we accept the Invariance Thesis, such problems *have been proved* to be intractable.⁶ An example of this kind of problems is the so-called *generalised chess problem*, i.e., the game of chess played on a generic board of dimensions $n \times n$ (instead of 8×8). Given as input a certain configuration of the pieces on the board, the problem consists in deciding if there exists a winning move for the player who moves first. In order to solve this problem, a DTM requires a computation time that is exponential with respect to the dimension n of the board.

Summing up, *if we accept the Invariance Thesis, the fact that some problems are not computationally tractable cannot be overcome by hypothesising the existence of more powerful machines, larger memories, more efficient architectures, faster processors, or parallel computing devices of some kind* (e.g., artificial or biological neural networks).

There is a point relevant to parallel computing that needs some elucidation. In general, a correlation exists between the computation time required by a sequential device and the memory space used by a parallel device. Concerning this point, it is usually assumed that the so-called *Parallel Computation Thesis (PCT)* holds, according to which, if something can be computed within polynomial space bounds by some "reasonable" sequential machine, then it can be computed within polynomial space bounds also by some "reasonable" parallel machine, and vice versa. Now, as every NP problem can be computed within polynomial space bounds by a DTM, it seems that the PCT implies that NP problems can be computed within polynomial time bounds by a parallel machine. However, the PCT assumes that the number of computing units of a parallel computing device can be increased indefinitely. In general, according to the PCT the number of computing units can grow even exponentially with respect to the num-

⁶ See, for example, Johnson (1990).

ber of computational steps. It is worth noting that, in this respect, what is reasonable for the PCT is different from what is reasonable for the Invariance Thesis. In particular, according to the PCT, parallel computers can be considered reasonable, if they admit an exponential growth of the number of their processors. However, computers of this kind turn out to be "unreasonable" according to the Invariance Thesis.⁷

The assumptions required by the PCT are surely difficult to justify from a cognitive point of view. As a consequence, in the following I shall assume that parallel machines of the kind described above do not correspond to realistic idealisations of the capabilities of finite cognitive subjects. In other words, I shall assume that they do not constitute "reasonable" models from the cognitive point of view, and that, with respect to the problems discussed here, they must not therefore be considered a falsification of the Invariance Thesis.

5. Tractable competence

According to the considerations in the previous section, the properties associated with the computational tractability of a problem do not pertain to some specific way of solving that problem (e.g., a specific algorithm or a kind of machine); rather, they pertain to the problem itself. In other words, computational-complexity properties concern a computed *function*, not the methods used to compute it. As a consequence, in the field of cognitive processes, we can reasonably conclude that considerations on computational tractability pertain to Marr's *level of computational theories* and, therefore, to the sphere of a *theory of competence*.

As said previously, the fact that Marr seems to consider theories of competence identical with computational theories probably does not fully correspond to Chomsky's position. However, this is of minor importance in the present context. The Marrian notion of competence is rather impoverished as compared

⁷ On these points, see the chapter by van Emde Boas (1990).

with the Chomskian notion. According to the latter, also some aspects of the algorithmic level are probably significant for competence. But if considerations on computational tractability pertain to the mere level of computational theories, they would be *a fortiori* relevant to a broader notion of competence. Similarly, let us assume that competence theories correspond to level-1.5 theories in the sense of Peacocke (1986). This would be no problem for my thesis: if the properties of computational complexity pertain to level 1, they must also pertain to level 1.5.

So, it is reasonable to deduce that considerations on computational complexity should be taken into account when developing a theory of competence for some cognitive task. From such considerations, there emerges a notion of *tractable competence*, that is, a more constrained view of an idealised competent subject, according to which, to build a theory of competence for a certain cognitive task, a computationally tractable function modelling such a task should be singled out.

The notion of tractable competence is still an idealisation, as compared with the actual capabilities of cognitive subjects. Consider a certain cognitive task C , and suppose that a competence theory for C has been proposed (for the sake of simplicity, let us assume that a competence theory coincides with a computational theory in the sense of Marr). In addition, let us suppose that the function that models task C (e.g., f_C) is a computationally tractable function (i.e., the corresponding problem is a P problem). f_C is still certainly an idealisation of the cognitive task C as it is performed by human beings. For example, f_C does not take into account the specific bounds to the resources of the human beings in the flesh: it admits inputs of arbitrary dimensions such that nobody could ever process them, or computations that require memory dimensions transcending human limitations, or that are so long to exceed the life of any human being. In this sense, imposing a tractability constraint on f_C has nothing to do with the typical constraints of a theory of performance.

Broadly speaking, the importance of the facts concerning computational complexity within the context of cognitive sciences

could hardly be questioned. In the field of computational simulation of cognitive processes, considerable effort has been devoted to devising formal techniques for facing problems that, in their general form, are computationally intractable. For example, one of the classical (and historically oldest) subfields of artificial intelligence deals with the development of techniques (heuristics, etc.) for solving search problems, and with their applications in areas like game playing.

In more specific terms, a possible objection to the thesis of the cognitive pertinence of computational-complexity results (and, *a fortiori*, of their importance for theories of competence) is the following. Complexity results pertain *worst cases*: the fact that a given problem belongs to a certain complexity class is established on the basis of the instances of the problem that are more demanding in terms of computational resources. It might be objected that, from a cognitive point of view, average cases would be more significant than the worst cases. Two considerations are appropriate. First, up to now, for technical reasons, the study of average computational complexity has been less developed than the study of worst case complexity. However, also the average complexity properties of a problem are among the properties of the computed *function*, and not of some algorithm or of its implementation. As a consequence, the assumption about their cognitive relevance agrees with the main thesis of this paper, i.e., some aspects of the limitations on computational resources pertain to the level of a theory of competence. Secondly, the assumption about a greater cognitive importance of average complexity as compared with worst-case complexity is motivated by the observation that real cognitive subjects face intractable computational tasks by adopting strategies that *approximate* the global task and «discard,» in a sense, the most demanding cases. The study of approximate solutions to intractable problems is one of the main lines of research in the field of computational complexity. Given a computationally intractable function f , an approximation for f is a tractable function f' such that the outputs of f' are suitably related with the outputs of f . To assume that worst-case complexity is relevant to a theory of

competence amounts to assuming that the function f' pertains to the level of competence, i.e., that the study of the approximations used by cognitive systems cannot be entirely relegated to the sphere of performance.⁸

The advantages of taking into account some notion of «tractable» competence can be motivated by methodological and philosophical considerations concerning the plausibility and the feasibility of a computational view of the mind. A computational model of a given cognitive task that does not take into account computational-complexity constraints turns out to be scarcely informative about the way in which that task is actually carried out by the mind, and has therefore a limited explication value. According to Hector Levesque, such a model «tells us *nothing* about how the cognitive activity is physically possible» [Levesque (1988), p. 359]. The existence of a model for a certain process that is based on computationally intractable functions involves that there exists a (Turing) machine performing the process, but it gives no contribution to adequately explain how it is performed by the human mind. From a perspective of philosophy of the mind, a competence model that does not take into account tractability constraints is of little help in supporting a computational image of the mind, in that it is fully consistent with the objection that the human mind performs the task in a completely different way. Models of this kind give no support against the claim that human beings can do things that no «realistic» machine can do.

A slightly different way of considering the importance of complexity results for competence is the following: What is important is not that a theory of competence *must* take into account limitations on computational resources. Rather, it is important that a model of competence *may* take into account such limitations. This allows one to develop competence models that are much more detailed and satisfactory. For example, one can avoid relegating a number of crucial aspects to the level of performance. I mentioned the problem of approximations. It can also be sug-

⁸ This kind of objections was also discussed by Hector Levesque [Levesque (1988), pp. 364-365].

gested that the theory of computational complexity could account for the systematic character of certain «errors» and of behaviours that deviate from the canons of normative rationality.

A further advantage consists in the possibility of distinguishing between different aspects of resource limitations. As said before (§ 2), not every resource constraint can be considered pertinent to a theory of «tractable» competence. The tractable-competence approach allows one to distinguish between two different kinds of resource constraints: (a) constraints that depend on the cognitive task as such (and that are therefore relevant to a competence theory for that task), and (b) constraints that depend on specific aspects of the cognitive architecture or on accidental or contextual factors (and that are therefore relevant to the level of performance, or even for Marr's level of implementation). An example of the latter kind is the previously quoted Miller's *magic number 7±2*.

Some connectionist arguments in favour of parallel distributed models and against classical symbolic approach pertain to type (b) constraints. An example is the so-called «hundred-step» constraint discussed by Feldman and Ballard (1982). This constraint stems from the fact that neurons are slow as compared with digital computers: the duration of a single «operation» of one neuron is of the order of ten milliseconds. Therefore, all cognitive phenomena lasting less than one second presuppose a computation whose length is at most equal to one hundred steps. This makes sense only if some parallel computational architecture is assumed. As a consequence, a strictly sequential model (based, for instance, on von Neumann architecture) cannot explain how information is processed by the nervous system. It is obvious that considerations of this type have nothing to do with the computational properties of problems; rather, they are relevant only to the organisation of the cognitive architecture and of the biophysical properties of its components. In this sense, the «hundred-step» constraint pertains exclusively to the algorithmic/implementation levels of Marr's hierarchy.

However, it is highly plausible that not all the aspects of connectionist models can be confined to the implementation or the algorithmic levels. Roughly speaking (and without debating the complex issues concerning the opposition between connectionist and classical cognitive sciences), at least three main views of the relationships between competence theories and connectionism can be singled out. The first view regard connectionist models as being mere implementations of classical competence theories. This is the position taken up by the supporters of the classical symbolic paradigm (e.g., Fodor and Pylyshyn (1988)). According to the second view, the relations between competence theories and connectionist models are more complex. Competence is still conceived in classical terms; however, connectionist models are not mere implementations. For example, according to Paul Smolensky (1988, 1989), a (classical) competence theory can be seen as a sort of macroscopic approximation for the real behaviour of a cognitive system, in the same sense that classical, Newtonian physics is an approximation for the real, microscopic behaviour of a physical system. According to this view, competence theories are in a sense fictitious: the true description of the cognitive phenomenon is provided by the connectionist model.⁹ The third view maintains that connectionism entails competence models that are different from classical models. In Marrian terms, connectionist and classical models compute different functions, and therefore presuppose different computational theories¹⁰. If we accept the third view, the computational (or competence) theories associated with connectionist models can be in their turn constrained by complexity theoretical factors.¹¹

6. Some comparisons with related positions

⁹ See also Clark (1990) for a discussion on these points.

¹⁰ According to a similar point of view, Golden (1996) wrote a handbook on artificial neural networks that follows Marr's hierarchy: neural nets are presented according to an implementation, a computational and an algorithmic point of view. For a connectionist counterpart of a theory of competence, see also Clark (1990).

¹¹ For example, Judd (1990) and Parberry (1994) investigated the computational complexity properties of neural network models.

The results of the theory of computational complexity received some attention from various authors in the field of cognitive sciences. In this section, I shall review some positions reported in the literature that are in some respect related to the theses of the present article.

I have claimed that considerations of computational complexity pertain David Marr's level-1 theories. A similar position, even though not concerned directly with the notion of competence, has been maintained by John Tsotsos in a series of papers on computational complexity in visual perception. For example, Tsotsos states that:

it is surprising that Marr did not even mention the problem of computational complexity *as part of the computational level of his theory*. [...] Yet considerations of efficiency or complexity are not just implementation details as Marr implies. [...] Complexity satisfaction is a major constraint on the possible solutions of the problem. It can distinguish between solutions that are realizable and those that are not [Tsotsos (1990), p. 424, italics added].

A thesis of the present paper is that there are cases in which effective computability could turn out to be too liberal an idealisation of cognitive abilities. A similar position has been argued for by Charles Parsons (1997) with reference to mathematical skills. He suggests that the appeal to the notion of effective computability alone results in an excessively permissive idealisation of human mathematical capabilities, and that more suitable and realistic constraints would be obtained by considering the theory of computational complexity.

Before the development of complexity theory it was in a sense mandatory to accept the notion of effective procedure as a suitable idealisation of human capabilities, in that any further restriction could rest exclusively on an extramathematical (e.g., physical or psychological) ground. However,

the matter looks somewhat different today, after the development of the theory of computational complexity. Any positive

solution to a decision problem still leaves a number of further problems open, whether the decision problem is P, NP, or any one of a number of complexity classes. *These concepts may be motivated by questions of physical possibility or some other considerations of feasibility, but in themselves they are perfectly mathematical notions.* [Parsons (1997), p. 348, italics added].

In the previous section, I have claimed that tractable competence is still an idealisation, as compared with actual human performance. Analogously, Parsons stresses that computational complexity notions are idealised notions. He claims that:

these notions still have elements of mathematical idealization, such as the lack of a bound on the length of the input and parameters that can take arbitrary values, such as the degree of the polynomial in the concept of polynomial-time computability [*ibid.*, p. 351].

An author who has taken into account the relationships between computational complexity and cognitive phenomena, and whose positions are worth some discussion is Christopher Cherniak (1986). He stresses the importance of computational-complexity results in order to develop realistic psychological models of rationality (see, for example, §§ 4.3 - 4.9). He also mentions the possible significance of computational complexity for a theory of competence. In a footnote, he states that, if we interpret traditional logical systems as models of *logical competence*, then «the fundamental datum of departures of actual human reasoning from ideal models is [...] treated as a *performance deficit*» [Cherniak (1986), p. 144, italics added]. Therefore, «the antagonism between formal correctness and tractability makes the ideal competence account seem particularly unrealistic» [*ibid.*, p. 145]. Cherniak's conclusion is that «a psychologically realistic concept of 'minimal competence' seems to have some advantages» [*ibid.*]. However, the notion of such 'minimal competence' is not further developed, so it remains difficult to evaluate its features and implications. In particular, it is not clear whether Cherniak would agree on the distinction between constraints of type (a) and (b), as sketched above. It is

likely that Cherniak's position does not assume a similar distinction. He formulates the notion of minimal competence in the context of his discussion on *minimal rationality*. Cherniak's notion of minimal rationality includes aspects that, even according to the canons of tractable competence, would pertain to the sphere of performance. For example, consistently with the canons of minimal rationality, a model of inferential capability could not be deductively closed. In general, a minimal agent «has *fixed* limits on cognitive resources such as time and memory» [*ibid.*, p. 3, italics added]. Again:

The most important unsatisfactoriness of the ideal general rationality condition arises from its denial of a fundamental feature of human existence, that human beings are in the *finitary predicament* of having fixed limits on their cognitive capacities and the time available to them. Unlike Turing machines, actual human beings in everyday situations or even in scientific inquiry do not have potentially infinite memory and computing time [*ibid.* p. 8].

Cherniak does not say whether, and in what sense, minimal competence is an abstraction with respect to the capabilities of actual minimal agents. As seen before, tractable competence is a mathematical idealisation, according to which an idealised competent subject is not submitted to specific resource bounds. On the contrary, a notion of competence developed along the lines of Cherniak's minimal rationality would seem to encompass most aspects concerning performance. As a consequence, it seems to be difficult to defend a notion of *competence* developed on this ground: it would not be clear in what sense it could be contrasted to *performance*.

A position that is closer to the proposal of a tractable competence has been developed by Hector Levesque (1988), even if he does not propose his thesis couched in terms of a notion of competence. Levesque is specifically concerned with logical models of reasoning. As said before, he argues that a model of a cognitive activity that does not take into account constraints on computational resources gives no information on how that activity

can be achieved by a cognitive subject. According to Levesque, there are two strategies to overcome this problem. One consists in considering the properties of specific computational architectures (e.g., classical von Neumann architectures, connectionist architectures, etc.) and in studying their properties and reciprocal advantages. The other lies in regarding cognitive tasks as information-processing tasks, i.e., as functional mappings from sets of input values into sets of output values [Levesque (1988), p. 360]. The latter alternative, which is the one favoured by Levesque, is implicit in the choice of the theory of computational complexity as a formal tool. There are evident analogies between Levesque's definition of the second strategy and Marr's notion of computational theory.

The latter strategy allows one to distinguish between tasks that are physically plausible and tasks that are physically unrealisable: «Those in the first category can be eliminated from further consideration as part of a computational model of cognitive activity, and those in the second category are tasks that we will be prepared to examine further, perhaps (though not necessarily) using specific computational architectures» [*ibid.*, p.361]. Levesque stresses that this kind of approach allows one to face, from an information-processing perspective, aspects that are traditionally considered to pertain the level of performance: «[...] memory limitations, attention lapses, or anything else that may be viewed as less than ideal. This is all part of the subject matter, and needs to be modelled deliberately as part of an IP [Information Processing] task» [*ibid.*, p 364]. As far as models of reasoning are concerned, Levesque's position is in many respects akin to the theses of Cherniak:

My thesis is as follows. The deviations from classical logic that will be necessary to ensure the tractability of reasoning stand in very close correspondence to the deviations from logic that we would have to make anyway to be psychologically realistic. If we look at the kinds of mistakes people make, the kinds of problems people run into, and the corners that are cut to get around them, we will find modifications to classical logic that ensure the computational tractability of the associated thinking. [*ibid.*, pp. 369-370].

7. Conclusions

The notion of competence plays a basic role in many developments of cognitive sciences. In this article, I have argued that there are aspects of resource limitations that can be accounted for within the context of a theory of competence. My arguments stem from the results of the theory of computational complexity. On such a basis, I have proposed a notion of *tractable competence*, that is, a notion of competence that is constrained by considerations of computational tractability. The adoption of tractable competence imposes the adoption of more demanding adequacy criteria on competence theories. As a result, it enforces the development of competence models endowed with a greater explicative value and a greater empirical content. A notion of tractable competence allows the cost of computational resources not to be entirely relegated to the domain of performance.

In addition, the introduction of tractable competence enables one to distinguish between different aspects of resource limitations, namely, the aspects that are determined by competence factors, and the aspects that depend exclusively on performance. From this point of view, an interesting subject for further investigations might consist in determining in what cases and to what extent systematic human deviations from the canons of normative rationality (as coded, for example, by logic, probability theory, game theory or the theory of rational choice) could be explained in terms of computational tractability. In other terms, it might be interesting to determine to what extent human «errors» could be accounted for in terms of competence.

Acknowledgements

I would like to thank Margherita Benzi, Alessandro Lenci, Diego Marconi, Carlo Marletti, Dario Palladino, Salvatore Ram-

pone, Guglielmo Tamburrini and Achille Varzi for reading earlier versions of this paper and/or for discussing the theses presented in it. I gave a talk on these topics at the workshop on *Logical Normativity and Commonsense Reasoning* held in Trento, Italy, in October 1996. I wish to thank the participants to the discussion. Subsequently, a short version of the paper has been published in Italian in the book *Normatività logica e ragionamento di senso comune*, edited by Francesca Castellani and Luisa Montecucco, Bologna, Il Mulino, 1998.

This work was partially supported by the projects «Programma Cofinanziato CERTAMEN» and «Programma Cofinanziato Verità, dimostrazione e rappresentazione nella storia e nella filosofia della logica,» both funded by the Italian Ministry for the University and Scientific and Technological Research (MURST).

References

- Bennett, C.H., E. Bernstein, G. Brassard, and U. Vazirani (1997), 'Strengths and weaknesses of quantum computing', *SIAM Journal on Computing*, 26(5), pp. 1510-1523.
- Cadoli, M. (1995), *Tractable Reasoning in Artificial Intelligence*, Berlin: Springer Verlag.
- Cherniak, C. (1986), *Minimal Rationality*, Cambridge, MA: MIT Press.
- Chierchia, G., and S. McConnell-Ginet (1990), *Meaning and Grammar. An Introduction to Semantics*, Cambridge, MA: MIT Press.
- Chomsky, N. (1965), *Aspects of the Theory of Syntax*, Cambridge, MA: MIT Press.
- Clark, A. (1990), 'Connectionism, competence and explanation', *British Journal for the Philosophy of Science* 41, pp. 195-222, reprinted in M. Boden, ed., *Readings in the Philosophy of Artificial Intelligence*, Oxford, UK: Oxford University Press, 1990.
- Cohen, L.J. (1981), 'Can human irrationality be experimentally demonstrated?', *The Behavioral and Brain Sciences* 4, pp. 317-370.
- Cresswell, M.J. (1978), 'Semantic competence', in F. Guenther e M. Guenther-Reuter, eds., *Meaning and Translation*, London, UK: Duckworth.
- Deutsch, D. (1985), 'Quantum theory, the Church-Turing principle and the universal quantum computer', *Proc. Royal Society of London A* 400, pp. 97-117.

- van Emde Boas, P. (1990), 'Machine models and simulations', in (van Leeuwen 1990).
- Feldman, J.A., and F.H. Ballard (1982), 'Connectionist models and their properties', *Cognitive Science* 6, pp. 205-254.
- Fodor, J. and Z. Pylyshyn (1988), 'Connectionism and cognitive architecture', *Cognition* 28, pp. 3-71.
- Frixione, M., and G. Tamburrini (1996), 'Conflitti di competenza', *Sistemi intelligenti* 8(3), pp. 375-402.
- Golden, R.M. (1996), *Mathematical Methods for Neural Network Analysis and design*, Cambridge, MA: MIT Press.
- Haugeland, J., ed. (1981), *Mind Design* (1st edition), Cambridge, MA: MIT Press.
- Johnson, D.S. (1990), 'A catalog of complexity classes', in (van Leeuwen 1990).
- Johnson-Laird, Ph.N., and R.M.J. Byrne (1991), *Deduction (Essays in Cognitive Psychology)*, Hove, UK: Erlbaum.
- Judd, J.S. (1990), *Neural Network Design and the Complexity of Learning*, Cambridge, MA: MIT Press.
- Larson, R., and G. Segal (1995), *Knowledge of Meaning : An Introduction to Semantic Theory*, Cambridge, MA: MIT Press.
- van Leeuwen, J., ed. (1990), *Handbook of Theoretical Computer Science. Volume A. Algorithms and Complexity*, Amsterdam, NE: Elsevier.
- Levesque, H.J. (1988), 'Logic and the complexity of reasoning', *Journal of Philosophical Logic* 17(4), pp. 355-389.
- Marconi, D. (1997), *Lexical Competence*, Cambridge, MA: MIT Press.
- Marr, D. (1977), 'Artificial intelligence: a personal view', *Artificial Intelligence* 9, pp. 37-48, reprinted in J. Haugeland (1981).
- Marr, D. (1982), *Vision*, New York, NY: Freeman.
- Miller, G.A. (1956), 'The magical number seven, plus or minus two: Some limits in our capacity for processing information', *Psychological Review* 63, pp. 81-97.
- Nebel, B. (1996), 'Artificial intelligence: a computational perspective', in G. Brewka, ed., *Principles of Knowledge Representation*, Stanford, CA: CSLI Publications, Studies in Logic, Language and Information.
- Papadimitriou, C.H. (1994), *Computational Complexity*, Reading, MA: Addison-Wesley.
- Parberry, I. (1994), *Circuit Complexity and Neural Networks*, Cambridge, MA: MIT Press.
- Partee, B. (1982), 'Belief sentences and the limits of semantics', in S. Peters e E. Saarinen, eds., *Processes, Beliefs and Questions*, Dordrecht and Boston: Reidel.
- Parsons, Ch. (1997), 'What can we do "in principle"?', in M.L. Dalla Chiara, K. Doets, D. Mundici, J. Van Benthem, eds., *Logic and Scientific Methods*, Dordrecht and Boston: Kluwer.

- Peacocke, C. (1986), 'Explanation in computational psychology: Language, perception and level 1.5', *Mind and Language* 1(2), pp. 1010-23.
- Shor, P.W. (1997), 'Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer', *SIAM Journal on Computing*, 26(5), pp. 1484-1509.
- Smolensky, P. (1988), 'On the proper treatment of connectionism', *Behavioral and Brain Sciences* 11, pp.1-74.
- Smolensky, P. (1989), 'Connectionist modeling: Neural computation/mental connections', in L. Nadel, L.A. Cooper, P. Culicover, and R.M. Harnish, eds., *Neural Connections, Mental Computation*, Cambridge, MA: MIT Press.
- Stockmeyer, L. (1987), 'Classifying the computational complexity of problems', *Journal of Symbolic Logic* 52(1), pp. 1-43.
- Tsotsos, J. (1990), 'Analyzing vision at the complexity level', *Behavioral and Brain Sciences* 13(3), pp. 423-469.