On Competence and Meta-Knowledge

Gerhard Wickler **Louise Pryor**
Department of Artificial Intelligence
University of Edinburgh
80 South Bridge
Edinburgh EH1 1HN
Scotland
{gw|louisep}@aisb.ed.ac.uk

Abstract

In this paper we define and attack the problem of competence assessment for intelligent agents. The basic idea is that we use meta-knowledge to infer competence. The main contribution of this paper is a single rule that allows efficient competence assessment for any system with explicit strategic knowledge. The reason for this is that strategic knowledge already contains the right information. Cognitive evidence supports our theory.

Competence and Intelligent Agents

The Problem of Competence Assessment

The problem we attempt to address in this paper is best illustrated by looking at an example. Consider the problem-solving activity of human problem solvers given the following simple physics problem[1]:

> A block of mass m starts from rest down a plane of length l inclined at an angle Θ with the horizontal. If the coefficient of friction between block and plane is μ, what is the block's speed as it reaches the bottom of the plane?

Given that the human problem solvers have some knowledge of physics in the form of equations that are appropriate to the problem, they will most likely answer the question whether they *can* solve this problem with "yes", i.e. they will state that they are competent to solve this particular problem instance. We will refer to this question as the **competence assessment problem.** However, then asked *what* the solution is they would probably reply that to work out the *exact* speed of the block at the end of the plane they needed some more time.

[1]This example problem is taken from (Larkin *et al.* 1980), work we shall return to later in this paper.

Now, this is interesting. Assessing whether one *can* solve a problem seems to be much easier than actually solving the problem, at least for a human problem solver. This claim is supported by research on the knowing not phenomenon (Barr 1979). They point out that "people often know rapidly and reliably that they *do not know* something."

Intelligent Agents and Competence

To be able to solve the competence assessment problem efficiently would be a very useful and important ability for an intelligent (computational) agent trying to solve a problem as well.

Intelligent agents do not necessarily have all the knowledge they need to solve a problem but they might know of other agents they can communicate with that have the knowledge they lack. Thus, in order to solve its problem the first agent has to find another agent that can provide the desired knowledge, formulate and ask the relevant question, wait for the reply, and continue processing. The second agent receives the query and tries to find an answer. If it finds an answer it returns it and does something else.

A problem occurs when the agent is incompetent to solve the given problem and wastes time looking for the solution anyway. It may be the case that other queries cannot be worked on during this time. The simplest approach to trying to avoid this situation is for the question-answering agent to proceed in two steps:

1. Assess own *competence* with respect to the given problem instance;
2. Try to find the answer.

Questions about Competence

The first step, deciding the competence assessment problem, is what we want to look at more closely now. The **basic idea** of how this can be done could, if not must, look something like this:

We have to abstract from the given problem instance and from the knowledge base that will be used to try to solve this problem instance and test whether the two match in some way.

We can divide this problem into several categories by distinguishing different types of questions the intelligent agent should ask itself in order to assess its competence. These can be identified by the kind of object the question is about. What these questions all have in common is that they ask about the existence of some knowledge or capability within the agent. We have come up with the following basic categories:

1. Do I know about < *instance* >?

   e.g. Mont Blanc, Robert Burns, `ThatBlock` (from the above example), etc.

2. Do I know about < *concept* >?

   e.g. mountains, poets, sliding blocks, etc.

3. Can I derive < *proposition* >[2]?

   e.g. Mont Blanc is x metres high, etc.

4. Can I do < *action* >[3]?

   e.g. climbing the Mont Blanc, etc.

Looking at the sliding block problem again we need to assess competence for a proposition, i.e. a question of the third type. From the problem "What is the block's speed as it reaches the bottom of the plane?" we take the proposition "The block's speed as it reaches the bottom of the plane is x?" to insert into the competence question.

We will first address the third question type because research on explanation indicates how to answer this question. Achinstein defines what it means to be in a knowledge-state with respect to an (explanation-seeking) question Q (Achinstein 1983, page 24). Although the above questions are not explanation seeking questions, the notion of "being in a knowledge-state" is very useful here. Let Q be a question asking for the proposition P. Then someone *(A) is in a knowledge-state with respect to Q iff A knows of P*[4] *that it is a correct answer to Q.* Similarly, extending this definition, we can say that A is in a complete knowledge-state with respect to P iff A is in a knowledge-state with respect to any Q asking for P. We can say that *A is in a knowledge-state with respect to P iff A is in a knowledge-state with respect to some Q asking for P.*

As this is work from philosophy, the knower A from above was intended to be human, not a computational intelligent agent. We now need to look at the concept of being in a knowledge-state from a computational point of view. In that sense we would like to view being in a knowledge-state as a property of the knowledge base rather than a whole system that might not be a question-answering system. This can be done by saying that *a knowledge base KB is in a knowledge-state with respect to P iff an appropriate mechanism could extract P from KB given some Q as above.*

The problem with this is that P is a proposition in some formalism and thus, extraction means derivation. Only in a rather simple formalism like propositional logic is it possible to decide whether a given proposition follows from the agents knowledge base or not. This can be done by just trying to derive the given proposition. However, this is not addressing the competence question directly but trying to solve the problem and obviously gives no efficiency gain. Furthermore, if the underlying formalism is more complex, as, for example, first order logic then the problem is known to be undecidable. Notice that the difficulty comes from the problem itself.

However, we can apply the above definition of knowledge-states to the first two question types simply by saying that P is an instance or concept. This gives us a very convenient computational criterion for deciding on competence questions about instances and concepts: basically we just need to search for knowledge involving them. For example, we will assume that an agent knows about the block from our sliding block example if the symbol `ThatBlock` representing the block is mentioned in sentences in the agent's knowledge base.

As we believe that a useful knowledge representation formalism will have the property of undecidability quite often, we can only hope for an *approximate* answer to a competence question about propositions. That is, we hope for an answer like "Yes, it is *likely* that I can solve this problem (e.g. derive a proposition like the one in the sliding block problem)." In this way we could overcome the undecidability and efficiency problem mentioned above. The last kind of competence question about actions has the same problem associated with it, we can only get a probabilistic evaluation here as well.

[2] This question can be seen as asking about the agent's *knowledge* if the *knowledge* refers to the deductive closure of the knowledge base.

[3] Given that an agent knows the primitive actions it can perform, this kind of competence question is equivalent to asking whether the agent can generate and execute a plan implementing the action in question. Whether and how the competence questions about propositions and actions are related remains an open issue.

[4] In the original work P is existentially quantified to cover equivalent statements. The idea itself is based on the concept of being "acquainted" with P (Russell 1910).

Looking at human problem solvers again, we find the same behaviour: the initial assessment might be wrong. This is simply a result of the fact that the competence decision is based upon an abstraction of the problem instance, and some details might reveal the problem as more difficult than initially thought. If, in the example from the beginning of this paper, the plane turns out to be so long that the effects of aerodynamics become important, then human problem solvers may fail at this problem instance although they originally thought they were competent.

Up to now, we know how to solve the competence assessment problem for instances and concepts and we know that the best we can hope for in the case of propositions and actions are probabilistic answers. Also, it is intuitively clear that answering the more difficult questions relies on the simpler cases.

Meta-Knowledge

The Type of Knowledge Required

We now have a set of question types representing the competence assessment problem, two of which we know how to deal with, and two we have yet to deal with. To address the latter we need two things: the knowledge required to answer the questions, and a method to get at this knowledge efficiently. We will first look at the type of knowledge required.

Competence knowledge is *knowledge about other knowledge.* Davis and Buchanan have used the term *meta-knowledge* to describe exactly this kind of knowledge (Davis & Buchanan 1977):

> ... meta-level knowledge is knowledge about knowledge. Its primary use here[5] is to enable a program to "know what it knows", and to make multiple uses of its knowledge. That is, the program is not only able to use its knowledge directly, but may also be able to examine it, abstract it, reason about it, or direct its application.

The kind of knowledge the meta-level knowledge examines is called object-level knowledge. This is knowledge of the domain that is used to solve a problem, the kind of knowledge you would find in a trace of the system.

What we want to do with the object-level knowledge in order to assess competence is to *abstract it*, i.e. abstract from the given problem instance, and then *reason* whether we can use the agents knowledge to solve the problem. Thus, let us look at the meta-knowledge TEIRESIAS contained.

[5] "here" refers to TEIRESIAS, the system they were building.

Types of Meta-Knowledge

In discussing their system TEIRESIAS, Davis and Buchanan identified four different types of meta-knowledge. They do not claim that these are all the types there are but just what was relevant for their work. If meta-knowledge is the kind of knowledge required for competence assessment then it makes sense to look at the types of meta-knowledge they have identified:

- *Schemata* hold knowledge about the representation of objects. This knowledge describes the structure and interrelations of the objects represented.
- *Function templates* hold knowledge about the representation of functions. It resembles a simplified procedure declaration.
- *Rule models* hold knowledge about inference rules. They are abstract descriptions of a set of rules, built from empirical generalizations about these rules.
- *Meta-rules* hold the knowledge about reasoning strategies, i.e. knowledge about how to use other knowledge. It helps in deciding which knowledge to invoke next in a situation where more than one chunk of knowledge may be applicable.

One can argue whether all the types of knowledge in this list are really meta-knowledge. For example, schemata representing internal structure and relations between objects like a conceptual hierarchy are very close to semantic networks. Function templates declare the types of the arguments to the function in this hierarchy. Rule models are an abstraction from other knowledge in the system. In fact, we would like to see these three types rather as additional knowledge than as meta-knowledge because when using this knowledge in the reasoning process, the objects being reasoned about are not the actual rules in the object-level knowledge base. This is only the case for meta-rules and hence we would see this as true meta-knowledge here.

In the rest of this paper we will focus on meta-rules as a formalism to represent meta-knowledge. However, there is no need for the rule-basedness of the formalism. The emphasis should be seen as on the kind of knowledge this is, i.e. explicit knowledge how to decide in which direction to proceed in the search space, knowledge to control search. This was exactly the aim in Davis's work on meta-rules (Davis 1980).

Although they mention that the same knowledge may be used for other purposes as well, they did not follow this up. There is no mention of assessing competence based on this knowledge.

Explicit Representation versus Inferred Knowledge

There is one principled concern here that we need to discuss before we have a closer look at the relation between meta-rules and competence. If we want an agent to be able to answer the above competence questions efficiently then there are two basic approaches to getting at the required knowledge: adding an explicit representation of competence versus trying to infer competence with respect to a given topic from the object-level knowledge base.

A specific representation of competence knowledge will most likely be much more efficient because it can be designed to supply the knowledge to our questions directly. On the other hand there is always the risk that the object-level and meta-level knowledge become inconsistent since they are held in separate representations.

Attempting to infer competence could turn out to be very expensive and, as we have mentioned above, it might not be decidable at all depending on the formalism used for the object-level knowledge. However, if new knowledge is added to the system or existing knowledge is updated then inferring competence will always result in a correct assessment. This aids maintainability of the knowledge base greatly and there is no need to guess in advance all the things one might want to do with the knowledge. Some systems even change their own competence by learning or forgetting (Smyth & Keane 1995) in which case it is impossible to have a static representation of competence. Concerning efficiency, inferring the knowledge might be more expensive, but the example of human problem solvers given above indicates that the cost might not be too high. We will consider this approach further.

The Thesis: Meta-Rules Indicate Competence

In this section we will show how meta-rules as used for search control can be used to indicate competence. First we will look closer at an example meta-rule to see exactly what knowledge it represents. Then we will show how this knowledge can be used to assess competence, and finally we will present some cognitive evidence that supports our work.

Meta-Rules

Meta-rules guide search by pruning the search space or reordering the set of plausibly useful knowledge sources, i.e. the object-level rules. Meta-rules, in TEIRESIAS, are associated with the goals of object-level rules. If such a goal is activated the meta-rules indicate which applicable object-level rule should be disregarded and which ones should be tried before others. The following example meta-rule is taken from (Davis 1980). The goal it is associated with is to determine an investment.

METARULE002:
If [1] the age of the client is greater than 60,
[2] there are rules which mention high risk,
[3] there are rules which mention low risk,
then it is very likely (.8) that the former should be used after the latter.

Notice that the goal itself is not mentioned in the rule but can be seen as an additional precondition that is checked in a different way. Then there are two kinds of precondition in this meta-rule: the implicit precondition (the goal) and the first explicit precondition refer to the problem instance to be solved, the second and third precondition refer to the object-level knowledge of the system. The conclusion tells the inference engine which rules should be preferred over others. In other words, the preconditions in meta-rules abstract from the actual problem instance and the object-level knowledge base in order to determine the search strategy. This sounds very much like the basic idea of how to address competence questions as mentioned above.

Similar formalisms to encode strategic knowledge show the same principles, for example PRESS (Bundy *et al.* 1979), (Bundy & Welham 1981). There are also knowledge representation systems that support the representation of meta-knowledge, mostly by allowing sentences at the object-level to be associated with constants at the meta-level, for example SPHERE (Filman, Lamping, & Montalvo 1983), Omega (Attardi & Simi 1984), metaProlog (Bowen 1985), RM (Ginsberg 1986), etc. However, none of these addresses the competence assessment problem we are dealing with here.

Meta-Rules and Competence

Our claim here is that meta-rules contain knowledge about competence. Given a problem instance with its goal we can use the following simple rule to assess competence:

> If **there is a meta-rule** applicable to the given problem instance that tells us which object-level knowledge to apply then it is likely that **we are competent** with respect to this problem instance.

The intuition behind this rule is quite simple: in trying to assess our competence for a given problem we seek ways to attack the problem. If we know of a promising way to attack the problem then we assume that we are competent. The existence of meta-knowledge that tells the inference engine how to proceed in the search space represents exactly the condition of knowing how to attack the problem. Of course, it is possible that after actually starting to solve the problem the system discovers that the promising approach did not lead to the solution. This simply accounts for the fact that the competence assessment problem is not decidable in general.

Thus, in the above example of a meta-rule we can see that the system is competent to determine an investment for a client over the age of 60 if it has some object-level rules mentioning low risk. Otherwise the system could only try to find a high risk investment which we know from the meta-knowledge is unlikely to be suitable.

There are a number of possible objections at this point. For example, if the meta-knowledge still leaves us with a considerable number of promising rules then we know of quite few ways to attack the problem which might be interpreted as not really knowing how to attack the problem. Another objection could be that meta-rules do not contain all the knowledge we need to assess competence and we will fully agree with this point. We might need to explicitly represent further knowledge to get to a better assessment. Yet another objection could be that a rule-based approach is not general enough. However, a production system framework covers many AI techniques (Nilsson 1980). Still, one could argue that, for example, a case-based reasoning approach (as in JULIA (Kolodner 1987)) or a constraint-based approach is more suitable for the competence assessment problem. This is missing the point though.

The problem we are given is to decide on the basis of the knowledge-base (including all levels of knowledge and the problem instance at hand) whether it is likely that we can solve the problem. There are two issues here: how to abstract from the problem instance and knowledge base; and how to evaluate whether the two abstractions match. We believe that the hard part is the abstraction process because one has to identify some general features in terms of which the problem instance can be described. This problem is similar to the one in earlier work on concept learning systems: if the right attribute is not given to the system any mechanism must fail to derive an appropriate concept description. Only once the potentially important features have been identified it does makes sense to think about the mechanism. And this is also true for case-based reasoning as mentioned above: extracting the right features to classify and retrieve cases is a fundamental problem.

So why meta-rules? What we have shown above is how to use the knowledge in the meta-rules to get to an abstraction of the problem instance. Meta-rules perform a kind of abstraction from the problem instance and the object-level knowledge to decide what to do next. Thus, there are already a number of features contained in the meta-knowledge that are connected to features of the knowledge base. Instead of inventing new features, representing them in a different formalism, and doing the evaluation this way, it is more sensible to reuse the meta-knowledge in the system to decide the competence assessment problem.

Concerning knowledge that is necessary for competence assessment but not contained in the meta-rules, there is a possibility that this knowledge might be useful for pruning the search space as well and hence, should again be reused. We have not followed up this line of investigation as yet.

Some Cognitive Evidence

There is cognitive evidence supporting our claim that meta-knowledge indicates competence. (Larkin *et al.* 1980) have implemented two computational models simulating expert and novice reasoning showing that the expert's competence is based on additional meta-knowledge. The experiment they did was the following.

They prepared two example physics problems and had expert and novice subjects solve them. One of the problems is the sliding block example from the beginning of this paper. They asked the subjects to "think aloud" while solving the problems and recorded their reasoning. Then they took these data and adjusted the strategic knowledge of the two models so that their reasoning resembled that of expert and novice. The accuracy achieved in this way was around 92%.

The differences between the two computational models were in the strategic knowledge where the expert used more forward reasoning and has more sophisticated strategic knowledge. Looking at the search space it is obvious that unguided forward search results in a much larger search space than unguided backward search. Hence, for the expert to solve the problem with forward inferences more efficiently, more knowledge on guiding this search must be available. Anderson shows a similar behaviour in a different domain (Anderson 1981).

They way we like to see this result here is that strategic knowledge can be represented in meta-rules and more expertise means greater competence. Hence, having appropriate meta-rules and being competent are strongly related at least.

Further Related Work

Reflective systems are systems that can reason about their own knowledge-base by means of an explicit representation of it (Smith 1982), (Maes 1986), (Maes & Nardi 1988). Thus, the ability to assess ones own competence can be called a reflective ability of an intelligent agent. However, there are many other abilities which a fully reflective system must have: it should be able to reason about any aspect of its own knowledge. In the light of this it is not surprising that the question of competence assessment is not treated in the work on reflective systems mentioned above.

The competence assessment problem was addressed in the project REFLECT (Voß *et al.* 1990). Their aim was similar to ours in that they wanted to add efficient, non-constructive feasibility studies to their system by abstracting from the given problem. However, they chose to implement this by adding explicit necessary criteria to test for solvability of the problem instance. For example, the requirements are tested for inconsistency before the system attempts to find a configuration. Using meta-knowledge as described above results in a probabilistic assessment.

A question we have not addressed in this paper is how to get at this meta-knowledge that we use for competence assessment and this seems to be very hard indeed. (Laske 1986) has analyzed the process of knowledge elicitation and described the knowledge states of an expert during some problem-solving activity as snapshots in the problem space. He argues that even the operations that lead from one state to another are only inferred by the knowledge engineer and are, thus, unreliable. He would further argue that the elicitation of meta-knowledge is very difficult because of the large *epistemological distance*. However, as shown above, (Larkin *et al.* 1980) came up with models that seem to fit expert and novice meta-knowledge rather well. Still, the problem of meta-knowledge elicitation has to be solved in order to be able to exploit meta-rules for competence assessment on a larger scale.

Future Work and Conclusions

In summary, we have defined and illustrated the competence assessment problem and argued that it is of relevance for intelligent agents. We have identified several questions that manifest that problem, and have given an efficient way to decide the simpler cases. For the more complex cases, we identified meta-knowledge as the kind of knowledge required to address these questions, and did a thorough analysis of different types of meta-knowledge. We have argued that it is desirable to infer competence from the given knowledge rather than adding specific competence knowledge.

Then, looking at meta-rules in more detail, we have given a criterion based on this knowledge to decide the competence assessment problem. We have explained, illustrated and discussed our method, emphasizing the point that its usefulness stems from the fact that meta-rules contain the right knowledge, not from the particular representation formalism of meta-rules. Finally, we have given supporting evidence that strategic knowledge and competence are strongly related.

Our aim for the future is to apply these ideas to planning, more specifically the O-Plan framework (Tate, Drabble, & Kirby 1994). This implies that we need to identify planning-specific meta-knowledge and analyse it for features as described above. This is not to be mixed up with meta-plans (Wilensky 1981) but is the seeking of control knowledge for the planning task. We expect domain knowledge to play an important role here. More generally, we will also follow up different approaches in order to *characterize planning capabilities*, but this is a much wider issue.

Another point worth mentioning here is that the ideas presented are not yet implemented. The reason is simply that O-Plan uses an evaluation function rather than explicit knowledge to decide in which direction to search for a plan. Extracting explicit knowledge is hence the first step to competence assessment with meta-knowledge.

Acknowledgements

The work reported is sponsored as part of the O-Plan project by the Advanced Research Projects Agency (ARPA) and Rome Laboratory, Air Force Materiel Command, USAF, under grant number F30602-95-1-0022. The O-Plan project is monitored by Dr. Northrup Fowler III at the USAF Rome Laboratory. The U.S. Government is authorised to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing official policies or endorsements, either express or implied, of ARPA, Rome Laboratory or the U.S. Government.

Several members of Edinburgh University have contributed useful comments to this paper, especially Austin Tate and Andrew Tuson. Thank you.

References

Achinstein, P. 1983. *The Nature of Explanation.* New York, NY: Oxford University Press.

Anderson, J. R. 1981. Tuning of search of the problem space for geometry proofs. In *Proc. 7th IJCAI*, 165–170. Vancouver, Canada: University of British Columbia.

Attardi, G., and Simi, M. 1984. Metalanguage and reasoning across viewpoints. In O'Shea, T., ed., *Proc. 6th ECAI*, 413–422. Pisa, Italy: North-Holland.

Barr, A. 1979. Meta-knowledge and cognition. In *Proc. 6th IJCAI*, 31–33. Tokyo, Japan: William Kaufmann.

Bowen, K. A. 1985. Meta-level programmimg and knowledge representation. *New Generation Computing* 3(4):359–383.

Brachman, R. J., and Levesque, H. J., eds. 1985. *Readings in Knowledge Representation.* Los Altos, CA: Morgan Kaufmann.

Bundy, A., and Welham, B. 1981. Using meta-level inference for selective application of multiple rewrite rule sets in algebraic manipulation. *Artificial Intelligence* 16(2):189–212.

Bundy, A.; Byrd, L.; Luger, G.; Mellish, C.; and Palmer, M. 1979. Solving mechanics problems using meta-level inference. In *Proc. 6th IJCAI*, 1017–1027. Tokyo, Japan: William Kaufmann.

Davis, R., and Buchanan, B. G. 1977. Meta-level knowledge: Overview and applications. In *Proc. 5th IJCAI*, 920–927. Cambridge, MA: MIT. Also in: (Brachman & Levesque 1985, pages 389–396).

Davis, R. 1980. Meta-rules: Reasoning about control. *Artificial Intelligence* 15(3):179–222.

Filman, R. E.; Lamping, J.; and Montalvo, F. S. 1983. Meta-language and meta-reasoning. In *Proc. 8th IJCAI*, 365–369. Karlsruhe, Germany: William Kaufmann.

Ginsberg, A. 1986. A metalinguistic approach to the construction of knowledge base refinement systems. In *Proc. 5th AAAI*, 436–441. Philadelphia, PA: Morgan Kaufmann.

Kolodner, J. L. 1987. Extending problem solver capabilities through case-based inference. In Langley, P., ed., *Proc. 4th International Workshop on Machine Learning*, 167–178. Irvine, CA: University of California.

Larkin, J. H.; McDermott, J.; Simon, D. P.; and Simon, H. A. 1980. Models of competence in solving physics problems. *Cognitive Science* 4(4):317–345.

Laske, O. E. 1986. On competence and performance notions in expert system design: A critique of rapid prototyping. In *Proc. 6th International Workshop Expert Systems and their Applications*, 257–297.

Maes, P., and Nardi, D., eds. 1988. *Meta-Level Architectures and Reflection.* Amsterdam, The Netherlands: North-Holland.

Maes, P. 1986. Introspection in knowledge representation. In *Proc. 7th ECAI, Vol. I*, 256–269.

Nilsson, N. J. 1980. *Principles of Artificial Intelligence.* Palo Alto, CA: Tioga.

Russell, B. 1910. Knowledge by acquaintance and knowledge by description. *Aristotelian Society Proceedings* 11:108–128.

Smith, B. C. 1982. *Reflection and Semantics in a Procadural Language.* Ph.D. Dissertation, MIT, Cambridge, MA. Prologue in: (Brachman & Levesque 1985, pages 32–39).

Smyth, B., and Keane, M. T. 1995. Remembering to forget. In *Proc. 14th IJCAI*, 377–382. Montréal, Canada: Morgan Kaufmann.

Tate, A.; Drabble, B.; and Kirby, R. 1994. O-plan2: an open architecture for command, planning and control. In Zweben, M., and Fox, M. S., eds., *Intelligent Scheduling.* San Francisco: Morgan Kaufmann. chapter 7, 213–239.

Voß, A.; Karbach, W.; Drouven, U.; and Lorek, D. 1990. Competence assessment in configuration tasks. In *Proc. 9th ECAI*, 676–681. Stokholm, Sweden: Pitman.

Wilensky, R. 1981. Meta-planning: Representing and using knowledge about planning in problem solving and natural language understanding. *Cognitive Science* 5(3):197–233.