BIROn - Birkbeck Institutional Research Online

On the relation between Marr's levels: A response to Blokpoel (2017)

Richard P. Cooper
Centre for Cognition, Computation and Modelling
Department of Psychological Sciences
Birkbeck, University of London

David Peebles
Department of Psychology
University of Huddersfield

Blokpoel reminds us of the importance of consistency of function across Marr's levels, but we argue that the approach to ensuring consistency that he advocates — a strict relation through exact implementation of the higher-level function at the lower level — is unnecessarily restrictive. We show that it forces over-complication of the computational level (by requiring it to incorporate concerns from lower-levels) and results in the sacrifice of the distinct responsibilities associated with each level. We propose an alternative, no less rigorous, potential characterisation of the relation between levels.

Introduction

Blokpoel's attempt to constrain relationships between Marr's levels is to be welcomed. As Blokpoel notes, if one assumes only a loose relation between each of Marr's three levels then consistency between levels (however that might be defined) cannot be ensured. Blockpoel proposes a two-pronged approach to the problem of constraining relations between levels. First, he calls for a strict relation between successive levels, whereby each "subordinate level is an *exact* implementation of the higher level" (p. 3), and second he advocates the use of computational-level constraints on inputs (i.e., constraints at the top-most, computational, level) as a way of placing limits on lower-level accounts.

The Primary Difficulty for Strict Relations

While we support Blokpoel's goal, his call for a strict relation between successive levels seems to us to be unnecessarily restrictive. It fails to acknowledge that different levels are subject to qualitatively different types of constraint and in so doing it forces higher-level descriptions to incorporate consequences that derive from lower-level considerations (and arguably *vice versa*).

Consider the example of (well-defined) goal-directed problem solving. An informal computational-level theory based on the work of Newell and Simon (1972) might claim this requires finding a path through state-space from the initial state to a goal state by applying a sequence of operators (i.e., state transition functions). More formally, the problem may be characterised in the language of Blokpoel, Kwisthout, van der Weide, Wareham, and van Rooij (2013) as follows:

> *Input*: $\langle S, s_0, E, g \rangle$, where S is a set of states, $s_0 \in S$ is the start state, $E \subset [S \times S]$ is the set of edges between states, reflecting valid state transitions, and $g : S \rightarrow [0, 1]$ is a function that maps states to 1 if and only if they satisfy the goal (and 0 otherwise).
>
> *Output:* $p \in S^+$, where p_1, the first element of p, is s_0, each successive pair of elements in the path p is in E (i.e., $\langle p_{i-1}, p_i \rangle \in E$ for $0 < i \leq n$, where n is the length of the path p), and p_n (the last element of p) is a goal state (i.e., $g(p_n) = 1$).

This computational-level characterisation is intentionally minimal and one might argue for additional constraints on p (e.g., that it contains no loops or that it is a shortest path, etc.). One might even argue that the characterisation should specify output(s) for each potential input (as Blokpoel appears to suggest). Critically, however, even with such constraints, the computational level account makes no reference to algorithmic concepts (i.e., to concepts related to specific algorithms that might meet the computational requirement).

Consider now an algorithmic-level model of goal-directed problem solving that meets the computational-level specification. Algorithmic-level accounts typically incorporate some form of limited look-ahead, whereby participants are argued to imagine the consequences of different sequences of two or three possible moves, attach a valuation of the subsequent states, and choose the moves with the greatest valuation. This can be repeated until a goal state is achieved. More information is required to flesh out this sketch into a specific algorithm. Minimally we require a valuation function $v : S \rightarrow \mathcal{R}$ that maps states to values, and which is maximised for goal states, together with a look-ahead parameter (typically denoted k) that specifies the depth of look-ahead. While one can imagine different algorithmic-level models based on different value functions (v), the look-ahead parameter (k) reflects a resource constraint — a limitation on the algorithm imposed either by the human cognitive aparatus or by the requirement to act in a timely manner.

Consider now a specific problem, say one that requires at least d steps for its solution. For values of k less than d there is no guarantee that the algorithmic-level account will concur with the computational-level account, but for all solvable problems (i.e., all problems that can be solved in a finite number of moves), regardless of the value function, the output of the algorithmic account will match the computational-level for a sufficiently large value of k.

Our example is somewhat different from the example cited by Blokpoel. He discusses Bayesian Inverse Planning (BIP) as an account of how one might infer an agent's goals from its actions (and knowledge of the probabilistic relations between actions and goals). In the case of BIP, the argument (from tractability considerations) is that for the theory to be psychologically plausible one of two constraints must hold. These constraints concern the number of goals that must be considered, the maximum number of "values" for those goals (e.g., within the BIP framework a goal such as *satisfy hunger* might have three values: big-hunger, medium-hunger or little-hunger), and the probabilities of different combinations of goals. In this case the constraints relate to the environment within which BIP is tractable. But presumably even with a suitably constrained environment, different algorithms may introduce different resource constraints such that the algorithm will only approximate the computational-level BIP theory. Alternatively, the limits on the number of goals etc. required for BIP to be tractable may be imposed by architectural limitations (e.g., working memory capacity limitations), which flow from lower-level considerations (and not the computational level).

An Alternative Proposal

Given the above arguments, we propose an alternative account (to that of Blokpoel, 2017) concerning the relationship between levels. In order to ensure consistency between levels, Blokpoel proposes:

$$\mathcal{A}(i) = C(i)$$

for all inputs i within the cognitive capacity's domain, where C is a computational-level theory and $\mathcal{A}$ is a corresponding algorithmic-level theory. We propose instead the following relation between the algorithmic and computational levels:

$$\lim_{r \to \infty} \mathcal{A}_r(i) = C(i)$$

for all valid inputs i, where r denotes the resources of the specific algorithm $\mathcal{A}_r$ which implements (in the sense of Blokpoel) C, the target computational-level theory. Critically, in this alternative formulation r concerns the algorithmic level and does not feature in the computational-level theory.[1]

In formal terms, it is of course possible to fold r into the computational level so as to preserve the position of Blokpoel (2017), viz.:

$$\mathcal{A}(i, r) = C(i, r)$$

However, this formulation adds unnecessary complication to the computational-level specification — one must consider resources and their availability as a further input to the computational level. Perhaps more critically it locates r at the wrong level as r is a property of a specific algorithm. Different algorithms may use qualitatively different resources, and pushing the resource into the computational level means that the computational-level description is no longer algorithm independent.

---

[1]We assume a similar formulation of the relation between the algorithmic and representational level and the implementation level.

These concerns are magnified if one adopts the logical extension of Blokpoel's approach to the implementation level. Here one would be required to fold neural constraints into both the algorithmic and representational level and then into the computational level. Doing so loses one of the main reasons for distinguishing between levels in the first place — namely that one can work at one level without being overly concerned by lower (and higher) level constraints.

An Additional Concern

A subsidiary argument made by Blokpoel (2017) is that "each computational-level constraint limits the set of possible algorithms" (p. 8). While this may well be true of some computational-level constraints, it is not true of all computational-level constraints. The tractability constraint is a case in point. As van Rooij (2008) notes, some researchers have dismissed various computational-level theories on the grounds that they are intractable, meaning that there is provably no known algorithm that can compute the output of the computational-level theory in a reasonable time (where "reasonable" time is defined as a polynomial function of some complexity parameter of the input, such as the input's length). van Rooij further argues that this dismissal is unjustified if the specific inputs which require unreasonable time are not typically encountered. In other words, van Rooij's argument is that restricting inputs effectively renders tractable some computational-level theories that would otherwise be intractable. This is a position that we, and Blokpoel (2017) endorse. However, restricting inputs typically increases the space of potential algorithms because algorithms that might be unreasonable on the full set of inputs may be reasonable when the set of inputs is restricted. Consequently, it is not the case that computational-level constraints necessarily limit the set of possible algorithms.

Conclusion

We have argued that the relation between the computational and algorithmic levels proposed by Blokpoel

(2017) is idealistic. It may hold in the limit as resource (and other algorithmic-level) constraints are relaxed, but demanding that it hold independently of algorithmic-level constraints does not fully appreciate the purpose of distinguishing between levels. In our view, the root of the difficulty arises from Blokpoel's assertion that "the competence/performance distinction [is] orthogonal to Marr's levels of analysis" (Blokpoel, 2017, p. 2). While the distinction is not without its own difficulties (e.g., in identifying competence based purely on performance), Marr (1982) explicitly identified the competence theory of Chomsky (1965) as a computational-level theory, contrasting it with an algorithmic-level performance theory.

References

Blokpoel, M. (2017). Sculpting computational-level models. *Topics in Cognitive Science*, *This issue*.

Blokpoel, M., Kwisthout, J., van der Weide, T. P., Wareham, T., & van Rooij, I. (2013). A computational-level explanation of the speed of goal inference. *Journal of Mathematical Psychology*, *57*(3), 117–133.

Chomsky, N. (1965). *Aspects of a theory of syntax*. Cambridge, MA: MIT Press.

Marr, D. (1982). *Vision: A computational investigation into the human representation and processing of visual information, henry holt and co*. Freeman and Company.

Newell, A., & Simon, H. A. (1972). *Human problem solving*. Englewood Cliffs, NJ: Prentice-Hall.

van Rooij, I. (2008). The tractable cognition thesis. *Cognitive Science*, *32*(6), 939–984.