Journal of Cognitive Psychology
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/pecp21

A theory of reverse engineering and its application to Boolean systems

N. Y. Louis Lee [a] & P. N. Johnson-Laird [b]

[a] Department of Educational Psychology and the Centre for Learning Enhancement and Research, The Chinese University of Hong Kong, Hong Kong

[b] Department of Psychology, Princeton University, Princeton, NJ, USA

Version of record first published: 05 Apr 2013.

To cite this article: N. Y. Louis Lee & P. N. Johnson-Laird (2013): A theory of reverse engineering and its application to Boolean systems, Journal of Cognitive Psychology, DOI:10.1080/20445911.2013.782033

To link to this article: http://dx.doi.org/10.1080/20445911.2013.782033

Journal of Cognitive Psychology, 2013
http://dx.doi.org/10.1080/20445911.2013.782033

Routledge
Taylor & Francis Group

A theory of reverse engineering and its application to Boolean systems

N. Y. Louis Lee[1] and P. N. Johnson-Laird[2]

[1]Department of Educational Psychology and the Centre for Learning Enhancement and Research, The Chinese University of Hong Kong, Hong Kong
[2]Department of Psychology, Princeton University, Princeton, NJ, USA

To reverse engineer a system is to infer how its underlying mechanism works. This paper presents a theory of the process, which postulates that individuals rely on an initial strategy of either focusing on the outputs of a system one by one, or on the components of the system one by one. They then try to assemble the system guided by both local and global constraints. The theory predicts that three main factors should affect the difficulty of reverse engineering: the number of variable components in the system, the number of their settings that yield an output, and, most importantly, the interdependence of components on one another in yielding outputs. Five experiments corroborated these predictions, using a test bed of electric light circuits and water-flow systems based on Boolean logic.

Keywords: Boolean algebra; Constraints; Problem solving; Reasoning; Reverse engineering.

Suppose that you want to control the lights in your hall with two switches, so that you can switch the lights on or off with either switch. The logic of the circuit is simple, e.g., when one switch is up and the other is down, the light is on. Change a switch, so that either both are up or both are down, and the light goes off. This logic is equivalent to an exclusive disjunction: The light is on when either one switch is up or else the other switch is up, but not both. If you are not familiar with electrical circuits, you will find the task of designing a circuit with this logic difficult. The process in general is known as *reverse* engineering, and it calls for knowledge and imagination. You are likely to know that it isn't a good idea to make a short circuit between the two terminals of a light. You possess a number of such *local* constraints concerning individual wires. You are also likely to know that you need to include both switches in the same circuit. You possess a number of such *global* constraints concerning the circuit as a whole. Guided by these constraints, you try to design a circuit. But, how do you go about it? The aim of the present paper is to answer this question.

In general terms, reverse engineering is the process of inferring how a particular mechanism works. In industry, it is sometimes merely the development of a way to duplicate an artifact, but our concern is the inference that works backwards from an existing device to an understanding of how its components should be put together to yield its behaviour. In most cases, individuals already know how the components work—if

Correspondence should be addressed to N. Y. Louis Lee, Centre for Learning Enhancement and Research, The Chinese University of Hong Kong, Shatin, New Territories, Hong Kong SAR of China. E-mail: louis@cuhk.edu.hk

We thank Linden Ball, John Darley, Sam Glucksberg, Adele Goldberg, Geoff Goodwin, Andy Johnson-Laird, Sunny Khemlani, and Alex Todorov for their helpful comments, and three anonymous reviewers for a critical reading of the paper. The research was supported in part by a grant from the National Science Foundation (SES 0844851) to the second author to study deductive and probabilistic reasoning.

they don't, they have to reverse engineer them—and so our focus is on the process of assembling the components into an appropriate system.

A sizable literature on reverse engineering exists in the fields of artificial intelligence (e.g., Clarkson, Gorse, & Taylor, 1992), computer science (e.g., Eilam, 2005; Várady, Martin, & Cox, 1997), biology (e.g., Csete & Doyle, 2002; Li et al., 2002), and biotechnology (e.g., Tegnér, Yeung, Hasty, & Collins, 2003). Psychologists have studied some of the component processes of reverse engineering, such as how individuals simulate mechanical and electrical systems (e.g., Hegarty, 1992; Schwartz & Black, 1996), how they explain causal mechanisms (e.g., Miyake, 1986; White, 1995), and how they diagnose faults (e.g., Goodwin & Johnson-Laird, 2005a; Moray, 1999; Rouse & Hunt, 1984). Likewise, a key requisite for reverse engineering is to understand what a system does; research in the areas of hypothesis testing and causal learning has shed light on the process of understanding. For example, Klahr and colleagues (e.g., Klahr, 2000, 2005; Klahr & Dunbar, 1988) investigated how individuals discover the function of a button that controls a toy robot. They found that individuals formulate new hypotheses about the function either by applying prior knowledge, e.g., by using the linguistic meaning of the button's label, or by generalising the robot's behaviours that they collected earlier. Likewise, research on causal reasoning has also shown that individuals are adept at determining causal relations using both top-down and bottom-up processes. For instance, individuals—even children as young as 2 years old—are adept at determining causal relations between two events through perceptual cues as well as covariational data, with the latter overriding the former if necessary (see, e.g., Gopnik, Sobel, Schulz, & Glymour, 2001; Schulz & Gopnik, 2004). Other cues that help determine causality include the temporal contiguity between the two events, as well as relevant prior knowledge (see, e.g., Hagmayer & Waldmann, 2002; Lagnado & Sloman, 2004; for a review, see Lagnado, Waldmann, Hagmayer, & Sloman, 2007); these cues may even override covariational data when individuals make causal judgements (White, 1995).

Yet, the task of reverse engineering depends on more than hypothesis testing and causal reasoning. You may have inferred that the two switches in your hall control the light independently. And you may have discovered their causal role by relying on many of the factors that the previously mentioned studies have identified. But, your discovery is the starting point of reverse engineering. It calls for you to work out in detail how to wire up the components in an electrical circuit to yield these causal relations. As we shall see later, the difficulty of grasping the functionality of a system is sometimes different from the difficulty of reverse engineering the system. The lights in the hall are a good example. Most of us understand their functionality; few of us, as we show presently, can reverse engineer the system. In fact, a major difficulty in reverse engineering arises from an important factor: the interdependence of variable components on one another in determining the outputs of a system (but cf. Dörner, 1996; Halford, Wilson, & Phillips, 1998; Simon, 1996). Psychologists are familiar with the concept of an interaction among independent variables in an experiment. Its physical analogy in reverse engineering is the interaction among variable components, that is, the performance of the system cannot be predicted solely from knowledge of the "setting" of one such component, because it depends on the "settings" of others too. Once again, we distinguish between inferring the existence of such an interaction from observations of a system and devising a mechanism that yields such an interaction. The reverse engineer has at least one advantage over the inventor: the reverse engineer knows that the task is feasible (Johnson-Laird, 2005). Our theory and investigations accordingly focus, not so much on the discovery of causal relations or interactions, but on the assembly of a mechanism that gives rise to them.

One way to study reverse engineering is to see how experts reverse engineer complex machines such as digital cameras, or complex systems such as computer programs. Such experiments would last for days, if not weeks. Fortunately, there is an alternative way to approach the topic. We have developed a general theory of reverse engineering, which aims to account for the process in domains ranging from cameras to computer programs. We have tested the theory in a domain that has three essential requirements. First, it is fundamental to many systems from human concepts to the central processing units of digital computers. Second, its components are familiar to most individuals and easy for them to grasp. And, third, it allows psychologists to carry out experiments in a feasible amount of time. This test bed is circuits of electrical switches, or water-flow

systems, that have a Boolean logic, i.e., a logic based on *not*, *and*, and *or*.

The paper begins with a theory of reverse engineering, which is based on plausible psychological assumptions, and which yields readily testable predictions. Next, it describes the testbed of Boolean systems, and show how the theory applies to this domain. It reports the results of five experiments in which the participants reverse-engineered electrical circuits and waterflow systems. The results corroborated the theory's predictions. Finally, the paper draws some conclusions about reverse engineering in general.

A THEORY OF REVERSE ENGINEERING

Any system to be reverse-engineered contains a finite number of components that work together in giving rise to the system's behaviour. Some of these components are *variable*, that is, they can be in more than one distinct state that affects the performance of the system, e.g., the setting on a digital camera that allows for the playback or erasing of photographs. Other components of the system do not vary, e.g., a wire leading from a switch to a bulb. The system has a number of distinct inputs from the user and a number of consequent outputs, and they are mediated by a finite number of interconnected components. In some systems, a component may have a potentially infinite number of particular states, e.g., different voltages. But, for purposes of reverse engineering, we assume that all variable components can be treated as having a finite number of distinct states, i.e., the system as a whole is equivalent to a finite-state automaton. In other words, analogue systems can be digitised, as in digital cameras, CDs, and other formerly analogue devices. We also assume that the device is intended to be deterministic, though a nondeterministic finite-state device can always be emulated by one that is deterministic (Hopcroft & Ullman, 1979).

A camera, of course, takes indefinitely many inputs depending on the scene in front of it, but they are of only indirect concern to reverse engineers. Their task is to interconnect components, so that the components match the performance of a given camera, i.e., producing the same outputs as the original device from the same inputs. A deterministic finite-state automaton, however, can yield infinitely many distinct outputs, such as strings of symbols or behaviours, and no algorithm exists that is guaranteed to learn even the function that such systems compute (see Gold, 1967, for a proof given plausible assumptions about what constitutes learning). A corollary of this proof is that the reverse engineering of such systems also lacks any guarantee of success. However, granted that a system has only a finite number of distinct input–output pairs, the task is feasible: Among the finite number of ways in which its components can be interconnected, at least one exists that emulates the input–output pairs of the original device. Digital cameras, microwaves, and mobile phones, have a finite number of settings, finite memories, and a finite number of ways of mapping their physical inputs to an output. In sum, a feasible domain of reverse engineering is the set of deterministic finite-state devices yielding finite sets of outputs.

The theory postulates three main principles concerning the difficulty of reverse engineering, which are outlined next:

(1) The principle of variable components
(2) The principle of positive outputs
(3) The principle of dependence.

The principle of variable components

This principle states that the greater the number of variable components that affect the performance of a system, the harder the system should be to reverse engineer. With a large number of variable components, individuals should take longer, be more likely to err, and be more likely to fail. The reason, of course, is that as the number of components increases, so the number of their potential configurations increases exponentially. Individuals are therefore likely to overlook some configurations, with potentially disastrous effects on their reverse engineering. This principle is analogous to the concept of ''relational complexity'' introduced by Halford et al. (1998), according to which the number of arguments in a relation determines the difficulty of reasoning with the relation (see also, e.g., Goodwin & Johnson-Laird, 2005b).

For many systems, including many electrical circuits, outputs are binary: Either a light comes on or it does not. When individuals reason, they tend to focus on what is true, not what is false (Johnson-Laird, 2006), and this principle of *truth*

generalises to other binary domains. Individuals tend to represent what is possible rather than impossible (Bucciarelli & Johnson-Laird, 2005; Johnson-Laird & Savary, 1999), and what is an instance of a concept rather than what is not (Goodwin & Johnson-Laird, 2010). The present theory therefore postulates that individuals tend to focus on outputs from a system rather than their nonoccurrence, i.e., they focus on the light coming on rather than on it not coming on. A second factor should therefore affect performance: the principle of positive outputs.

The principle of positive outputs

This principle states that the greater the number of distinct states of the system that yield positive outputs, the harder the system should be to reverse engineer. Once again, individuals should take longer, be more likely to err, and be more likely to fail, as the number of positive outputs increases. Logically, reverse engineering necessitates a full consideration of all binary occurrences, both outputs and their nonoccurrence. But, only in those extreme cases in which the proportion of positive outputs vastly exceeds the proportion of negative outputs are individuals likely to change their focus to negative outputs (see, e.g., Feldman's, 2000, principle of parity). Yet, even in such cases, individuals still have to construct a system that would produce the positive outputs.

If a system has more than two or three states producing positive outputs, the principal psychological problem in reverse engineering is to decompose the set of input–output pairs into tractable subsets for which the engineer can make a suitable assembly of components. This "divide-and-rule" strategy is akin to means–ends analysis in standard problem solving (Newell, 1990; Newell & Simon, 1972; Simon, 1996). It calls for individuals to divide the system up into separate subsystems that can be tackled, at least at first, independently from one another. The required input–output relations provide a *global* constraint on the assessment of the assembly of such subsystems. And, as individuals tackle these assemblies and assess their performance, they acquire knowledge of additional *local* constraints governing the placement of individual components. But, the crucial factor is the extent to which the system can be partitioned into separate components that can be solved independently from one another. Hence, the most decisive principle of all in determining the difficulty of reverse engineering is: the principle of dependence.

The principle of dependence

This principle states that the greater the dependence of components on one another in determining the performance of the system, the harder the system should be to reverse engineer. If each variable component has an independent effect on output, the system is easy to decompose; but if two or more variable components interact, then the system is harder to decompose because their joint effects have to be engineered. This condition is bound to call for a more complex system than an independent one containing the same number of variable components. Dependence is similar to the notion of element interactivity proposed by Sweller and colleagues as a cause of difficulty in learning (e.g., Carlson, Chandler, & Sweller, 2003; Sweller & Chandler, 1994; van Merriënboer & Sweller, 2005). Fortunately, as Simon (1996) pointed out, most systems in the world—both natural and artificial—are "nearly decomposable", because it makes them easier to build and evolve. When they are not, as we will show, reverse engineering is very difficult.

Granted that the process of reverse engineering is goal directed, there are two obvious strategies for tackling a problem, where a strategy refers to a systematic sequence of elementary mental steps that an individual follows in devising a solution (Van der Henst, Yang, & Johnson-Laird, 2002, p. 426). One strategy is to focus, one at a time, on each of the system's variable components and its contribution to the outputs. With a microwave, they can focus on one control and try to devise a circuit that yields its appropriate outputs. Once they have constructed a circuit for this control, they can move on to the next one, and try to modify their existing circuit to incorporate its effects. Another strategy is to focus instead, one at a time, on each of the system's outputs. With the oven, they can focus one at a time on each output, such as broiling for a fixed time and temperature, and devise a subsystem containing all the requisite components that brings about the required output. Once they have constructed this subsystem, they can move on to the next output, say, toasting, and modify the circuit so that it copes with this new output too, and so on.

The two strategies are not mutually exclusive, and they are likely to differ in their effectiveness for different sorts of problem. Individuals may change from one to the other, just as they change from one strategy to another in deductive reasoning (Van der Henst et al., 2002). Ultimately, in order to solve a problem in its entirety, they may have to consider the contributions of all the components to all the outputs. Nevertheless, at the outset, individuals should focus one at a time either on the variable components, or else on the outputs.

These three principles pertain to the complexity of the causal mechanism underlying the target device, but the complexity of its physical implementation matters too. Hence, some domains of reverse engineering introduce a further potential source of difficulty. When individuals devise a mechanism, the fewer spatial dimensions in which they have to envisage the layout of the components, the easier the task should be. From a spatial standpoint, the simplest possible problem has a physical solution in one dimension, a more complex problem has a solution in two dimensions, and the most complex problem has a solution in three dimensions. Temporal relations may similarly add to the difficulty of the task, as they are likely to affect individuals' understanding of the causal system (e.g., Hagmayer & Waldmann, 2002), although experienced reverse engineers ought to be able to use prior knowledge of cues to the causal system (Lagnado et al., 2007). We examine this extra factor of dimensionality in our studies of Boolean systems, and we turn to these systems now in order to illustrate how the theory applies to this particular domain.

BOOLEAN CIRCUITS

Boolean systems are those based on negation (*not*), and the connectives of conjunction (*and*), disjunction (or), and those that can be defined in terms of them. In fact, any Boolean connective can be defined in terms of a single operator, *nand*, the negation of a conjunction, but the three preceding primitives make better sense psychologically because they correspond to the primitives that are needed for mental models (Goodwin & Johnson-Laird, 2010). Boolean systems include the sentential calculus in logic, where the variables can be true or false (Jeffrey, 1981), genetic algorithms, where the variables are the presence or absence of properties (see, e.g., Holland, Holyoak, Nisbett, & Thagard, 1986), and theorems about what can, and cannot, be learned in a tractable way (Valiant, 1984). The processes of binary arithmetic are also Boolean. Hence, the circuit for computing the addition of two single binary digits, 1 and 0, and the carry, if any, is known as a ''half-adder'', and it can be decomposed into two wholly independent subsystems: One computes the sum (*a or else b*), and the other computes the carry (*a and b*). The realisation of half-adders in silicon chips is the basis for the arithmetical computations of the central processing unit in a modern computer. Such a circuit is a deterministic finite-state automaton, but if the system is equipped with unlimited memory for the results of intermediate computations, it then has the power of a Universal Turing machine, which is able to compute any computable function (Hopcroft & Ullman, 1979). Boolean systems can therefore be extraordinarily powerful.

In psychology, Boolean connectives underlie deductive reasoning in which individuals infer conclusions from assertions containing such connectives (e.g., Evans, Newstead, & Byrne, 1993; Johnson-Laird & Byrne, 1991). Given such assertions, individuals can also evaluate their truth or falsity, list the possibilities to which they refer, and find a possibility common to several assertions; conversely, given a set of possibilities, they can formulate a description of them (Goodwin & Johnson-Laird, 2011). Mental model theory, which explains these tasks, rests on the assumption that each possibility is represented in a separate mental model, and that the smaller the number of models of the premises, the easier the task should be. Hence, it is easier to make deductions from exclusive disjunctions of the form, *a or else b but not both*, where *a* and *b* denote assertions, than from inclusive disjunctions of the form, *a or b or both* (Johnson-Laird & Byrne, 1991), because exclusive disjunctions yield models of two possibilities, whereas inclusive disjunctions yield models of three possibilities. The theory also postulates that mental models represent only what is true. This constraint gives rise to predictable and systematic fallacies in sentential reasoning (e.g., Johnson-Laird & Savary, 1999).

There exists a large literature on how individuals learn Boolean concepts (Feldman, 2000, 2006; Kruschke, 1992; Love, Medin, & Gureckis, 2004; Nosofsky, Gluck, Palmeri, McKinley, & Glauthier, 1994; Vigo, 2009). The standard conceptual learning task is to present individuals with

instances and noninstances of a concept, and their task is to learn to identify all and only the instances of the concept (e.g., Shepard, Hovland, & Jenkins, 1961). In a comparative study, Goodwin and Johnson-Laird (2011) showed that the difficulty of Boolean concept acquisition is likely to be dependent on the number of mental models underlying the concept. One of the model theory's predictions, which experimental results corroborate, is that it is easier to acquire a concept that is an exclusive disjunction—it has two models of its instances—than to acquire a concept that is an inclusive disjunction—it has three models of its instances. However, as the present studies will show, the difficulty underlying the learning of Boolean concepts is very different from their reverse engineering.

The reverse engineering of a Boolean system, unlike the preceding tasks, calls for the construction of a working model of components that computes a Boolean function. The simplest realisation of such components is a circuit of switches, such as those in our opening example of switches controlling the lights in a hall. The main dependent variable in this case is whether or not the current flows as opposed to truth or falsity in the sentential calculus, or to the presence or absence of a property in an instance of a concept. We chose Boolean circuits as our test bed, because they underlie so many systems—from the central processing unit of a digital computer to the logic of programs, and because nearly everyone understands the operations of an electrical switch, which makes or breaks a circuit. As an illustration of Boolean circuits, consider three examples, each containing two switches. The first is the circuit for conjunction, *a and b*, in which both switches have to be up in order for the light to come on. The second is the circuit for inclusive disjunction, *a or b*, in which at least one switch has to be up for the light to come on. And the third is the circuit for exclusive disjunction, *a or else b*, as in the case of the lights in the hall in which one and only one switch has to be up for the light to come on. Henceforth, we use "or" to denote inclusive disjunction, and "or else" to denote exclusive disjunction, because that is how logically untrained individuals tend to interpret these terms. More than one circuit is possible for each function, but Figure 1 illustrates the three standard circuits.

Consider the circuit for *a and b* in Figure 1, which we can describe as a single possibility:

a b

in which both a and b are on. This diagram, however, has a spatial interpretation: The start terminal (which is connected to a source of electricity) is connected to a, the two switches have a single connection between their "on" positions, and b is connected to the end terminal (which is connected to in series to the rest of the circuit including the light and the battery). We could arrive at the circuit for *a and b* by starting with all possible circuits between the start and end terminals:

a	b
a	not-b
not-a	b
not-a	not-b

then eliminating those circuits that do not yield the required outputs. The result is:

a b

Finally, we eliminate any irrelevant wires. This procedure is guaranteed to reverse engineer any Boolean circuit, and Figure 2 presents its physical embodiment for dealing with circuits of three switches. It shows the layout for all possible configurations of three switches, including yoked switches that operate together. The next step eliminates those wires that yield an output when they should not, and Figure 3 shows the modification in which wires, and irrelevant yokes, are

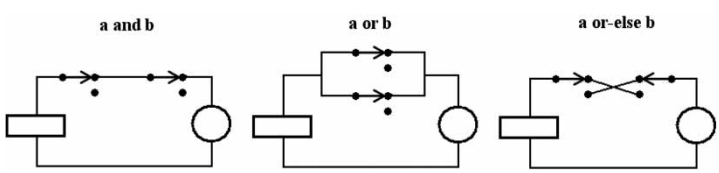

Figure 1. Minimal circuits for *a and b*, *a or b*, and *a or else b*. The rectangle and the circle represent the battery and the bulb respectively, and each black dot represents a terminal of a switch. The upper dot represents the "on" terminal of each switch. Hence, the bulb is on in the first two circuits, but not the last.

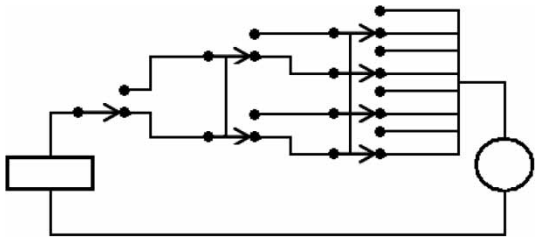

Figure 2. A general circuit yielding positive outputs for all eight possible configurations of three switches.

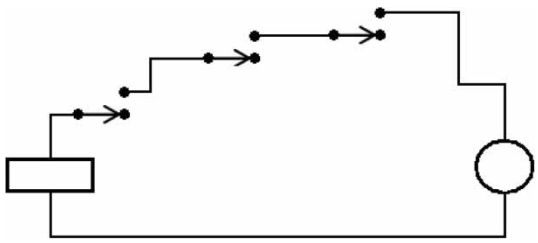

Figure 3. The circuit in Figure 2 with breaks in connections so that it is a circuit for *a and b and c*.

removed so that the positive instances correspond to those required for *a and b and c*. Naïve human reasoners are most unlikely to know this universal method or to discover it within an experiment. If they were to adopt it with the problems in Figure 1, then their relative difficulty should merely reflect the number of circuits to be eliminated from Figure 2, and so *a or else b* should be easier than *a and b*. How, then, do naïve individuals actually reverse engineer such circuits?

THE REVERSE ENGINEERING OF BOOLEAN SYSTEMS

The present theory of reverse engineering, which we described earlier, postulates that two sorts of constraints exist in reverse engineering, *local* constraints concerning the treatment of single individual components, such as single wires in Boolean circuits, and *global* constraints concerning more than one component and sometime the system as a whole. If an existing circuit containing the battery and light is already wired up between a start terminal and an end terminal, then the task is to complete the circuit between these two terminals by adding switches and wires in an appropriate way. Local constraints prevent an individual wire from insertion into a circuit in nonsensical ways:

(1) A terminal on a switch or a light should not be wired to itself in a loop.
(2) A duplicate of a wire already in the circuit should not be added to the circuit.
(3) The converse of a wire that is already in the circuit should not be added to the circuit: Wires are symmetric, and so a wire from A to B is equivalent to a wire from B to A, and so only one such wire is necessary.
(4) A single wire should not be introduced to connect one terminal on a switch to another terminal on the same switch, because that is what the switch itself does when it is on.
(5) A single wire should not be introduced to connect the start terminal to the end terminal, unless the circuit is supposed to keep the light on at all times.

The following global constraints apply to a Boolean circuit:

(1) The circuit should yield the appropriate output for each switch position.
(2) The start terminal must be connected to at least one switch.
(3) The end terminal must be connected to at least one switch.
(4) Each switch must have a wire connected to its single terminal, and a separate wire connected to at least one of its double terminals, where one of them is for the ''on'' position and the other is for the ''off'' position.
(5) Except in the case that the target is to produce a circuit whose light will come on regardless of the switches' positions, there should be no circuit of wires without switches from the start to the end terminal.
(6) A circuit in which two switches control a single light should not contain more than six wires. This number is arbitrary, but is a reasonable maximum number of wires needed for a problem containing two switches (see Figure 1).

Undergraduates, even those who are not studying science, readily grasp some of these local and global constraints (see Experiment 5 later) from instructions to solve the problems. However, they have little strategic knowledge. Yet, as they tackle problems, they acquire other constraints that allow them to solve problems much more readily. We have described elsewhere a mechanism for a strategic shift of this sort, in which individuals

deduce the consequences of a move while tackling a problem, and these consequences come to operate as constraints on the process of creating new moves (see Lee & Johnson-Laird, 2013). Since the functionality of each individual component constrains how the components are assembled, individuals in principle should be more likely to violate global than local constraints as they learn to reverse engineer any system. No individual can hope to solve any but the most trivial reverse engineering problem by trial and error, i.e., without using any a priori constraints, or without making any errors. We exclude, of course, the possibility that the individual already knows the solution.

The two main strategies for reverse engineering are to focus either on outputs or on components. With Boolean circuits, however, a focus on components is less useful. For example, in the case of the disjunction, *a or b*, a focus on switch *a* shows that the light can be on when it is up and when it is down, and likewise the light can be off when it is up and when it is down. So, a focus on a single switch in this case is not very useful. In contrast, a focus on outputs is more useful: The light is on when switch *a* is up and switch *b* is down. A circuit that yields this performance is a step in the right direction. Hence, the theory predicts that individuals should be more likely to adopt the strategy that focuses on outputs in the case of Boolean systems. Granted that individuals use the strategies we have described, the three main principles of the theory predict the difficulty of reverse engineering the circuits. The principle of variable components is the same for all three circuits in Figure 1, because they each have the same number of these components. But, the principles of dependence and positive outputs apply to the circuits. Consider, first, dependence, which is the crucial principle for reverse engineering. In a circuit for a conjunction, *a and b*, each of the two switches must be on in order for the light to come on, but either switch operates *independently* to switch the light off. Conversely, in a circuit for an inclusive disjunction, *a or b*, each of the two switches operates *independently* in order to switch the light on, but both switches must both be off in order to switch the light off. However, in a circuit for exclusive disjunction, *a or else b*, neither switch can operate independently either to switch the light on, or to switch the light off; the positions of both switches control the status of the light. Dependence therefore predicts that the circuits for *a and b* and *a or b*, which are partially independent, should be easier to reverse engineer than the circuit for *a or else b*, which is wholly dependent. The principle of positive outputs then discriminates between conjunction and inclusive disjunction. The switch positions for *a and b* yield only one positive output, whereas those for *a or b* yield three positive outputs. It follows that *and* should be easier to reverse engineer than *or*. Overall, the theory predicts the following trend in the difficulty in reverse engineering the three circuits: *and* should be easier than *or*, which in turn should be easier than *or else*. We emphasise that this prediction differs from the known facts about acquiring the three concepts from knowledge of their instances (Goodwin & Johnson-Laird, 2011).

Dependence in any system demands a more complicated solution than a partially or wholly independent problem with comparable numbers of variable components and positive outputs. Hence, in the case of a Boolean problem, it requires a more complicated circuit, typically with more wires. The reason is that no switch by itself can either turn the light on or turn it off, and so both positions of any switch have to be wired into the overall circuit. Dependence is also related to the notion of nonlinearity in learning systems: It is impossible for a connectionist network to acquire an exclusive disjunction unless it has a layer of hidden units (see, e.g., Minsky & Papert, 1969)—an idea that is similar to Vapnik's (1998) notion of a linear algorithm that can *shatter* a set of points, i.e., it can learn every possible training set of positive and negative instances. Systems are either linear or nonlinear, whereas the notion of dependence is gradable: the ease of reverse engineering a system depends on the degree to which it is possible to analyse parts of the system independently from other parts. For instance, a circuit of the form *a and (b or else c)* is decomposable to this extent: Individuals can first figure out that switch *a* needs to be on for the light to come on, and then they can figure out the circuit for *b or else c*, which they can plug into the circuit for conjunction. Such decomposition is impossible with a circuit of the form *a or else (b or else c)*.

The theory postulates that the spatial dimensions of a physical system affect the difficulty of reverse engineering it, and this factor holds for Boolean circuits. The circuit for *a and b* has to handle only a single possibility in which both *a* and *b* are on:

a b

As we remarked earlier, this diagram has a spatial interpretation: The start terminal is connected to *a*, the two switches have a single connection between their ''on'' positions, and *b* is connected to the end terminal. Hence, the problem has a one-dimensional solution: a single wire from *a* to *b*. The circuit for *a or b* calls for a disjunction of two possibilities:

a
b

where each row represents a separate connection between the start and end terminals. Hence, it has a two-dimensional solution. But, the circuit for *a or else b* calls for a more complex disjunction:

a not-b
not-a b

As this diagram shows, *a* in its on position is connected to *b* in its off position, which we represent as ''not-b'', and *a* in its off position is connected to *b* in its on position. One solution is accordingly for the wires to cross one another in a three-dimensional spatial configuration (see Figure 1 for this circuit). However, if one switch is rotated to uncross the wires, the problem has a two-dimensional solution but one in which the two switches are *incongruent*, e.g., the switch for *a* in the diagram above is the right way up but the switch for *b* is upside down. Topologically, it is then possible to rotate the switch in the plane so that it takes up a congruent position, dragging its wires around behind it, but their lengths must now be increased to allow for the rotation and so they are no longer of minimal lengths. None of the participants in any of our studies ever discovered this circuit (cf. the incongruent left-to-right solution to *or else* in Figure 4, later). In summary, *and* in Boolean circuits has a one-dimensional solution, *or* has a congruent solution in two dimensions, and *or else* has an incongruent solution in two dimensions with the switches in opposite orientations or a congruent solution in three dimensions. (The possibility of a two-dimensional solution has the happy consequence that integrated chips for computing Boolean functions can also be two dimensional.) Dimensionality as a psychological factor must accordingly take into account congruence. It then predicts the same trend in difficulty as the principles of dependence and positive outcomes. Our final experiments were accordingly designed to tease apart dimensionality from these other two variables.

A COMPUTER MODEL

In order to explore the role of constraints in reverse engineering, we wrote a computer program in Common Lisp that simulates the reverse engineering of Boolean circuits (the source code is available on http://mentalmodels.princeton.edu/models). It uses list structures to represent the components of the circuit, and it can test any circuit to determine whether or not it works correctly. It tries to make a circuit from the start terminal to the end terminal, and so the circuit includes switches, each of which has one terminal on one side and two terminals on the other side. Hence, if necessary, the switch can make or break one circuit, or instead make two alternative circuits. The program can install wires that interconnect the switches and the start and end terminals. In its simplest mode of operation, the program carries out a sequence of random moves to try to reverse engineer a problem. It starts with a representation of the input–output performance of the target system, and the set of components. It then wires up components at random, using separate wires. When it tries to generate a circuit, say, for *a and b*, it can be given the constraint to add only the required number of wires. Without any further constraints, it generates many nonsensical circuits. But, when the program is equipped with the local and the global constraints described earlier, it discovered some creative alternatives to the standard circuits. Figure 4, for example, shows a different solution for *a or else b*, which it discovered. To our surprise, this solution has only a single wire connecting the two switches. It does, however, call for incongruent left-to-right switch positions and more distinct wires connecting the start and end terminals than the solution in Figure 1.

As an initial test of the trend prediction over *and*, *or*, and *or else*, we examined the performance of the computer program in four conditions: (1) with a single constraint (the number of wires to be used was set at six), (2) with the addition of local constraints only, (3) with the addition of global constraints only, and (4) with the addition of both sets of constraints. Once a circuit included six wires (a global constraint), it

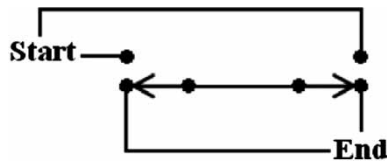

Figure 4. A circuit for *a or else b* discovered by the computer program: The current flows from start to end if and only if one switch is up and the other switch is down.

either solved the problem or not. Table 1 presents the frequencies over many trials with which the program solved the three sorts of problem, i.e., it came up with a circuit that yielded the correct output for the light for each switch position. The simulations showed, first, a reliable increase in performance as constraints were added, Jonckheere trend test, $z = 5.88$, $p < .001$; second, a reliable increase in difficulty over the three sorts of problem, Jonckheere trend test, $z = 7.30$, $p < .001$; and, third, an exception to this trend: *or* is easier than *and* when the program uses only local constraints, Mann-Whitney test, $z = 3.77$, $p < .001$. The reason for this phenomenon appears to be that local constraints alone have the emergent property of eliciting parallel circuits more often than serial circuits. The results of the simulations supported the theory, and so we carried out a series of experiments with human participants in order to test the theory's predictions.

EXPERIMENT 1: CIRCUITS OF TWO SWITCHES

The aim of our first experiment was to test the prediction of the difficulty of reverse engineering three sorts of circuit: conjunction (*a and b*) should be easier than inclusive disjunction (*a or b*), which should be easier than exclusive disjunction (*a or else b*). The participants' task was to reverse engineer these three circuits.

Method

The experiment was carried out under computer control. Each participant had to tackle all three problems, and each problem was presented on the screen as a "black box" with two binary, flick-on switches and a light bulb, and the computer displayed one at a time each of the four possible switch settings with the status of the bulb as on or off. As an example, Figure 5 presents the conjunctive problem. The participants' task was to devise a circuit of the switches, the battery, and the bulb, to yield the required outputs. We tested individually 18 Princeton undergraduates (10 male, eight female; mean age 21.3 years), who participated in the experiment for course credit, and who had no prior knowledge of switch circuits. The participants were assigned at random to each of the six possible orders of the problems. The experimenter explained the nature of the task, and that each switch had a terminal on one side and two terminals on the other side, so that the switch could make or break one or two circuits. He illustrated these uses of the switches. The key instructions were: "In each experimental trial, you will be shown a device with two switches and a light bulb. Certain combinations of the switches' positions turn the light on. However, the underlying wiring of the device is inside a box and hence is unknown. Your task is to draw the required wiring of the circuit." The participants were encouraged to draw putative circuits, and they were allowed to draw an unlimited number of them. They had 7 minutes for each problem, and were told that they would be timed from the presentation of a problem until they said, "done", to signal that they had completed a problem.

Results and discussion

Table 2 presents the percentage of correct solutions for the problems in Experiments 1 and 2. As predicted, the *and* problem was indeed easier for

TABLE 1
The mean number of times in 20 samples of 1000 trials, i.e., a total of 20,000 trials each, in which the program reverse engineered *and*, *or*, and *or else* switch circuits, depending on the constraints on generating moves

Type of problem	*Single constraint (only six wires)*	*Local constraints*	*Global constraints*	*Local and global constraints*
a and b	13	41	3316	4881
a or b	5	95	619	1359
a or else b	0	0	1	6

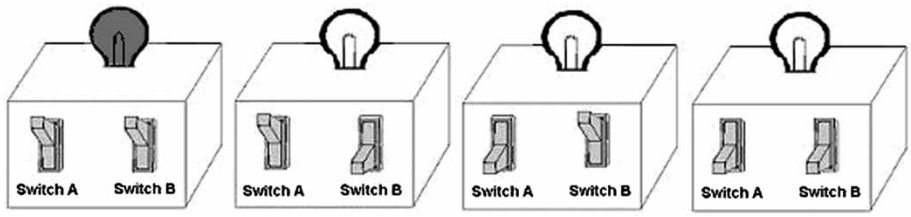

Figure 5. The presentation of the *a and b* problem on the computer screen in Experiment 1. The grey bulb indicates that the bulb is on.

participants than the *or* problem, which was in turn easier than the *or else* problem, Page's $L = 199.5$, $z = 2.75$, $p < .005$. Figure 6 presents the latencies for the three sorts of problem, and the trend was also reliable, Page's $L = 234.5$, $z = 3.08$, $p < .002$. The participants often drew more than one diagram for a given problem, and the mean numbers of diagrams that they drew to reach a solution or to exceed the time limit were 1.1 for the *and* problem, 2.4 for the *or* problem, and 3.8 diagrams for the *or else* problem, Page's $L = 194.0$, $z = 3.67$, $p < .0005$. To investigate if there is any difference between correct and incorrect responses, we break down the latencies and the number of diagrams of these two groups, and Table 3 shows them. The trends are broadly consistent with the general trend. The results thus corroborated the theory's predictions.

The participants drew diagrams in order to try to solve the problems, and readers may wonder what happens when individuals have to assemble actual components into circuits. We carried out a replication in which 12 new participants from the same population as before worked with real switches, bulbs, batteries, and wires. The pattern of results was the same as before: 11 participants solved the *and* problem, five solved the *or* problem, and two solved the *or else* problem, Page's $L = 130.5$, $z = 2.76$, $p < .005$. In addition, the same predicted trend occurred for the latencies (82.7 s, 428.8 s, and 541.9 s), Page's $L = 162.0$, $z = 3.67$, $p < .001$. The use of diagrams therefore appears to be ecologically valid. Readers may suppose that exclusive disjunctions are in general difficult to deal with, but, as we have already emphasised, the supposition is false. They are easier to use in deductive reasoning than inclusive disjunctions (Johnson-Laird & Byrne, 1991), and easier to acquire than inclusive disjunctions in concept learning (Goodwin & Johnson-Laird, 2010).

TABLE 2
The percentages of correct solutions for the problems in Experiments 1 and 2

	and	*or*	*or else*
Experiment 1	89	61	28
Experiment 2			
Switches	100	45	20
Water pumps	100	45	15

EXPERIMENT 2: STRATEGIES IN REVERSE ENGINEERING

This experiment examined the participants' strategies in reverse engineering circuits. They had to think aloud as they drew diagrams to try to solve each problem. Think-aloud protocols are controversial (e.g., Ericsson & Simon, 1984; Fleck & Weisberg, 2004; Nisbett & Wilson, 1977; Schooler, Ohlsson, & Brooks, 1993), but the same patterns of difficulty occurred in reasoning and in problem solving whether or not the participants had to think aloud (Lee & Johnson-Laird, 2013; Van der Henst et al., 2002). A second aim of the experiment was to test whether the results generalised to a different, though isomorphic, domain of water-flow systems, and whether there would be transfer from electrical circuits to water-flow systems, and vice versa. The switch circuits were

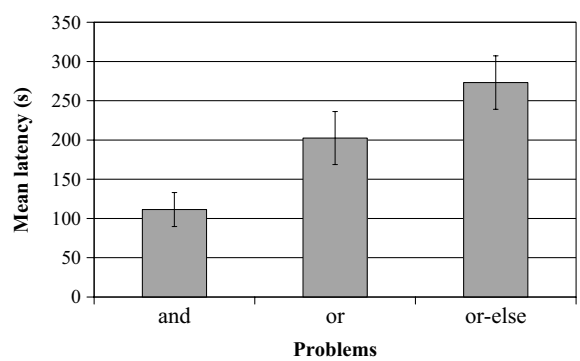

Figure 6. The mean latencies (s) of putative solutions, both accurate and inaccurate, for the three sorts of problem in Experiment 1.

TABLE 3
The mean latencies (s) and the mean number of diagrams participants drew for both correct and incorrect responses for the three sorts of problem (*and*, *or*, *or else*) in Experiments 1 and 2

			and	*or*	*or else*
Latencies					
Experiment 1	Correct		93.17	109.10	217.17
	Incorrect		257.13	319.18	294.74
Experiment 2	Switches	Correct	85.88	196.42	280.98
		Incorrect	n/a	589.49	483.34
	Water pumps	Correct	105.23	196.03	413.23
		Incorrect	n/a	410.86	588.22
Number of diagrams					
Experiment 1	Correct		0.88	1.55	3.60
	Incorrect		2.50	3.86	3.92
Experiment 2	Switches	Correct	0.55	2.11	2.00
		Incorrect	n/a	7.64	5.25
	Water pumps	Correct	1.20	2.33	4.27
		Incorrect	n/a	3.00	5.12

n/a = cases in which the participants made no errors.

the same as in the previous experiment, and the water-flow systems had to be assembled from the following components: a pump that circulated the water, two rotating faucets (each of which had only two positions, on or off, at 90 degrees from each other), a turbine, and pipes that were either straight or L-shaped. The task was to design a system so that the turbine ran only when the faucets were in appropriate positions, and so the problems in the two domains were isomorphic: The participants had to solve *and*, *or*, and *or else* problems in each domain. Figure 7 presents solutions to each of these problems in the water-flow domain. The theory predicted that the same trend in difficulty should occur in both domains, as the assembly process should not be affected by the surface features of the isomorphic problems.

Method

Twenty-one Princeton students participated in the experiment for course credit, but one was excluded from the data analysis for failing to understand the experimental instructions. The remaining participants (five male, 15 female; mean age = 19.1 years) were tested individually with two blocks of three problems (*and*, *or*, and *or else*). One block contained the three switch-circuit problems, and one block contained the three water-flow problems. The orders of the two blocks and the orders of the problems within each block were counterbalanced over the participants. The procedure was the same as in Experiment 1, with a simple training trial before each block of problems to familiarise the participants with the components. However, in the present experiment, the participants had to think aloud as they solved the problems. The experimenter instructed each participant: ''Please try to think aloud as you solve the problems; that is, please maintain a running commentary as you solve the problems. If you fall silent for more than a few seconds, I will probe you.'' The demand to think aloud tends to slow down performance, and so we allowed participants more time to solve each problem than in the previous experiment: they now had 11 minutes to solve each problem. They also had to describe the strategies that they had used in a

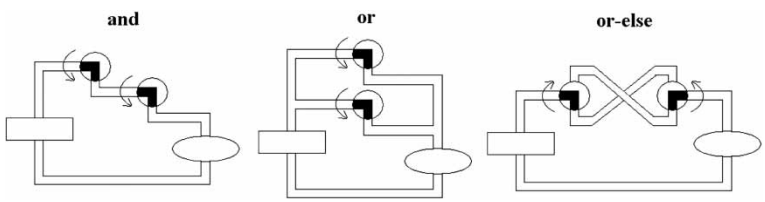

Figure 7. Minimal solutions to the three sorts of water-flow problem in Experiment 2. The rectangle and the oval represent the pump and the turbine respectively.

postexperimental interview. The experimenter recorded their protocols with a tape recorder.

Results

The numbers of correct solutions corroborated the predicted trend in both domains, and Table 2 show them: The *and* problem yielded more correct solutions than the *or* problem, which in turn yielded more correct solutions than the *or else* problem in both domains, Page's $L = 264.0$, $z = 3.79$, $p < .001$ for the switch domain, and Page's $L = 265.5$, $z = 4.03$, $p < .001$ for the water pump domain. Figure 8 presents the mean latencies for the problems, and the overall trend was highly reliable, Page's $L = 272.5$, $z = 5.14$, $p < .001$. The trend was also reliable in the number of diagrams that the participants drew (means, excluding the final diagram, were 0.9, 4.3, and 4.9 for the *and*, *or*, and *or else* problems, respectively), Page's $L = 270.0$, $z = 4.74$, $p < .001$. No reliable differences occurred in either accuracy or latency between the two domains, or even between the two blocks of trials, both in terms of accuracy and latency, Wilcoxon signed test, $z = 0.12$, $p = .45$, $z = 1.36$, $p = .09$, *ns*. Hence, no reliable transfer occurred from the first block of problems to the second block, and both blocks showed the same pattern of results. Indeed, out of four participants who solved the *or else* problem in the first block, only one managed to solve the problem in the second block. The difference between the *or* and the *or else* problem was smaller for the circuits than for the water-flow problems, Wilcoxon signed test, $z = 2.70$, $p < .005$,

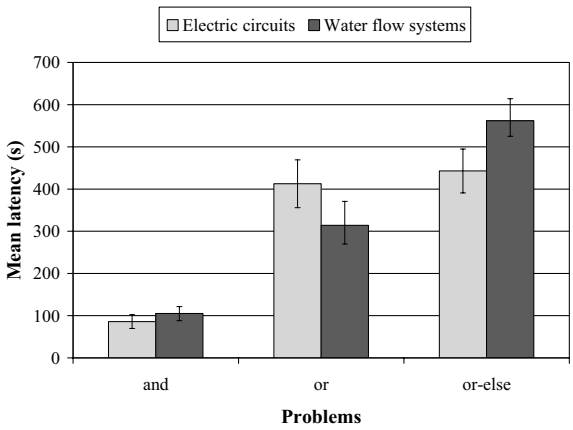

Figure 8. The mean latencies of putative solutions, both accurate and inaccurate, for the three sorts of problem (in both domains) in Experiment 2.

but the reason for this result is unclear. Table 3 shows the breakdown of latencies and the number of diagrams for both correct and incorrect responses. As in Experiment 1, the trends are broadly consistent with the general trends, the *or* switch problem being an exception.

The computer program, which we described earlier, makes random moves but they can be constrained by both local and global constraints. However, the think-aloud protocols showed that the participants did not use this simple method of trial and error but instead used the two main predicted strategies, which were also borne out in the postexperimental interviews. The participants did not always use the same strategy for all problems, and some participants even switched from one strategy to another whilst they were solving a problem. The first strategy is to consider each output separately. The participants synthesised a solution for one outputs, and then tried to modify this solution to capture the other outputs. In both Experiment 1 and the current experiment, the *or else* problem was both predicted and found to be most difficult.

Figure 9 shows a complete think-aloud transcript of how one participant (No. 11) tackled this problem in the water-flow domain. He started by going through all the outputs, noting that when the first faucet was up (i.e., the on position) and the other was rotated to the left (i.e., the off position), the turbine came on. He therefore drew a circuit to accommodate this output (see 1 in Figure 9). He then realised that this circuit would not account for the other outputs, and so he drew a new circuit (see 2 in the figure) to account for the turbine being on when the first faucet was off and the second faucet was on. However, as he went through the four outputs, he realised that this new circuit would not work either. He then reconsidered the given components of the system, and realised that he had only one pump and one turbine at his disposal, but he could add additional pipes to the system. He therefore connected a water pump to the first faucet but, instead of making this faucet control the water flow of one pipe, he added a second pipe to the faucet, so that the faucet acted as a switch, controlling the flow of water into one of two pipes. Then, he inserted a second faucet on the other ends of the two pipes (see 3 in the figure). He checked that this system produced all the required outputs, and announced that he had completed the problem.

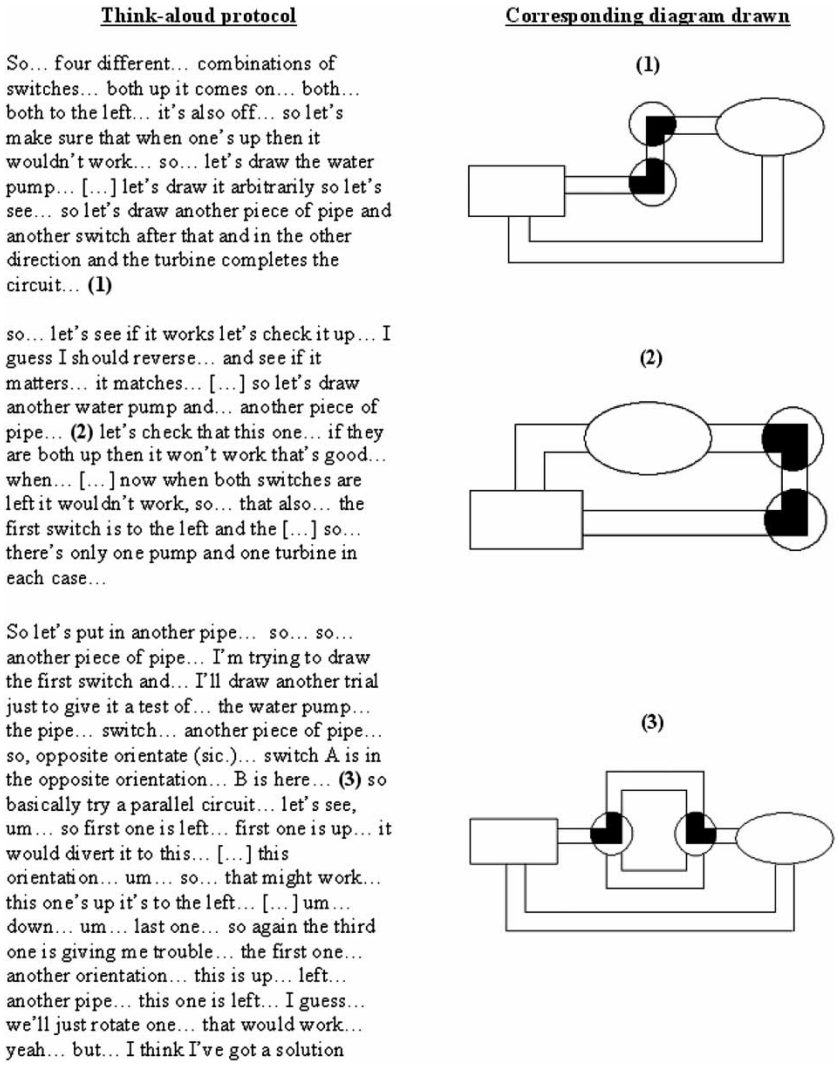

Figure 9. An example of a full protocol of a participant's think-aloud and the corresponding diagrams that he drew while tackling a problem in Experiment 2. Here, Participant 11 tackles the *or else* problem in the water-flow domain. "[…]" denotes unclear utterances, whereas "…" denotes a pause. (While the circuit may appear to be one for *a or else not b*, the participant in fact labelled the positions of the two faucets such that the effect of a *or else* circuit was produced. We have omitted labels for switch positions in the figures in this paper for easier reading.)

Figure 10 shows how another participant (No. 3) used this same strategy to try to solve another problem: the *or* problem in the water-flow domain. She started with the output in which faucets *a* and *b* were both up and the turbine was running, and constructed a working model for this output (see 1 in Figure 10). She modified this model to capture the output in which faucet *a* was on, faucet *b* was off, and the turbine was running. She accordingly added a branching pipe between the

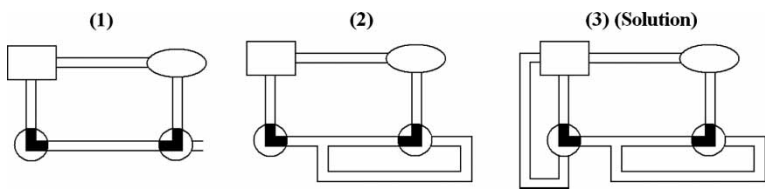

Figure 10. A protocol illustrating the strategy of focusing on one positive output at a time from Participant 3 in Experiment 2, working on the *or* problem in the water-flow domain.

two faucets, which she connected to the other end of faucet *b* so that the system would still work when faucet *b* was off (see 2 in the figure). She repeated this step to account for the third output in which faucet *a* was off and faucet *b* was on, and the turbine was running (see 3 in the figure). This solution, however, was incorrect, because the turbine would still come on when both faucets were off.

The second strategy is to focus on the effects of a single variable component, either a switch or a faucet depending on the domain. Figure 11 shows how one participant (No. 16) used this strategy to solve the switch-circuit *or* problem. She first focused on the fact that the bulb always came on when switch *a* was up. She drew a complete circuit with only switch *a* (see 2 in the figure). She then considered the possibilities in which switch *a* was down (see 3 in the figure). She began again, adding switch *b* to the circuit (see 4 in the figure). She worked out which output terminals in the circuit corresponded to which switch positions (see 5–8 in the figure). Finally, she drew out the resulting circuit in full (see 9 in the figure); it is a correct solution, though not the standard parallel circuit.

The two strategies are bound to converge in order to solve a problem, and so we categorised their use in terms of the first complete circuit that a participant created. If a participant focused on one output at a time, then the circuit contained both switches (or faucets). If a participant focused on one component at a time, then the circuit contained only one switch (or faucet). Of course, the subsequent diagrams and the participants' remarks as they thought aloud helped us to check this interpretation. For instance, the following remarks indicated the use of the strategy of focusing on one output at a time: ''consider a scenario in each box, then try to combine them'' (Participant 4), ''taking a diagram [output] at a time'' (Participant 5), and ''try to make 1 condition [output] meet, and then alter it to see if it fits the other conditions [outputs]'' (Participant 6).

We used the following criteria to classify the strategies:

(1) If a participant's first diagram of a complete circuit contains only one switch, then the participant is trying to focus on the effect of one switch at a time.
(2) If a participant's first diagram contains two switches, then the participant's strategy may be clarified by the subsequent diagrams. For example, if a subsequent diagram rotates the positions of the switches, then the participant is focusing on one output at a time.
(3) If a participant's first diagram is the solution to a problem, then, unless the ''think-aloud'' protocol clarifies matters, it is impossible to classify the participant's strategy.

In fact, as the theory predicted, the participants had a strong bias to focus on one output at a time

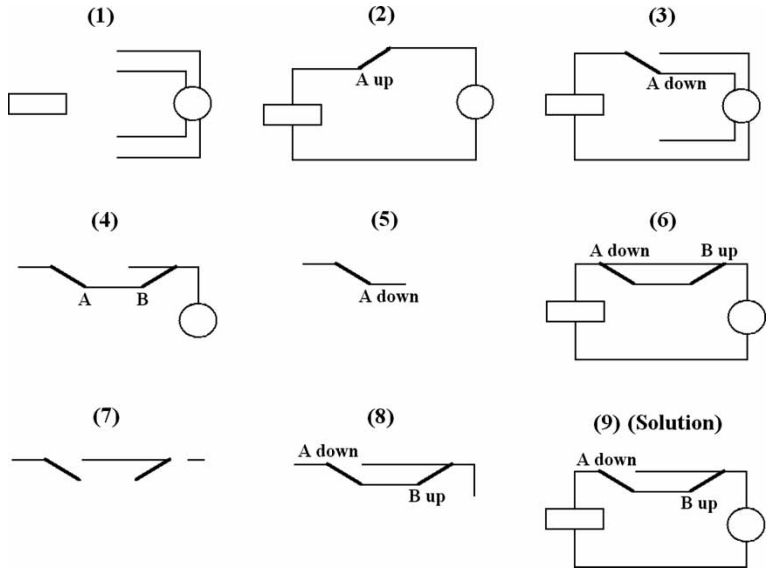

Figure 11. A protocol illustrating the strategy of focusing on one component at a time from Participant 16 in Experiment 2, working on the *or* problem in the circuit domain. We have added the verbal labels from the participant's think-aloud protocol.

(92%) as opposed to one component at a time (8%), Wilcoxon $z = 4.06$, $p < .001$. Their protocols also revealed that just occasionally they drew circuits that violated the local constraints that we described earlier: 2.5%, 50%, and 45% of the participants violated the fourth constraint (do not wire one terminal on a switch to another terminal on the same switch) for the *and*, *or*, and *or else* problems, respectively, but fewer than 10% of the participants violated each of the other four local constraints. Violations of global constraints were impossible to assess, because of the fragmentary nature of many drawings. But, few putative solutions (8%) ever violated any local or global constraints.

Experiment 2 replicated the results for switch circuits in the previous experiment, but it also showed that the same trend occurred in engineering water-flow systems, even though the isomorphism between the two domains was disguised by the difference between their components. The think-aloud protocols and the post-experimental interviews established that the participants did use the two main strategies, and that, as predicted, they tended to focus on positive outputs rather than on individual components.

EXPERIMENT 3: THREE-SWITCH PROBLEMS

An orthogonal comparison between the principle of dependence and the principle of positive outputs is impossible in circuits of only two switches, because any dependent circuit yields only two positive outputs. But, the comparison is possible for circuits of three switches. Hence, this experiment examined the reverse engineering of circuits containing three switches.

Method

Twenty-four Princeton University undergraduates took part in the experiment for course credit; four participants were excluded from the analysis because they failed to obey the instructions or to solve at least one problem. Each of the remaining participants (12 male, eight female; mean age = 20.3 years) tackled five problems. The initial problem for each participant was a simple conjunction: *a and b and c*: The light came on only when all three switches were up. The remaining four problems crossed two manipulations: The problems had either three or five positive outputs, and were either partially independent or dependent. The descriptions of the problems are as follows:

(1) *a and (b and c)* (1 positive instance)

Independent problems:

(2) *a and (b or c)* (3 positive instances)
(3) *a or (b and c)* (5 positive instances)

Dependent problems:

(4) *((a or else b) or else c) and (a or b)* (3 positive instances)
(5) *(not (a or else b)) or else (c or (a and b))* (5 positive instances)

In Problem 2, when switch *a* is up, either *b* or *c* has to be up in order to turn the light on, but when switch *a* is down, the light is off regardless of the configurations of the other switches: The problem is therefore partially independent. In Problem 4, however, when switch *a* is up, *b* and *c* have both to be up or both to be down for the light to come on, and when switch *a* is down, *b* has to be up, and *c* has to be down, for the light to come on: No switch has any independent control over the light. Problems 3 and 5 mirror Problems 2 and 4. The instances of the problems are summarised in Figure 12. For the independent problems only, a single plane through the cube representing a problem can divide the positive from the negative outputs (see Vapnik, 1998).

After Problem 1, the participants received the four other problems in one of the 24 possible orders. The problems were presented in the same way as in the previous experiments except that all eight configurations were shown at once. Likewise, the only difference in the instructions was that the participants were told that they also had yoked switches at their disposal when they built the circuits. A yoked switch, as illustrated in Figure 13, yokes two binary switches with a bar so that they make or break two separate circuits together. But, as the participants were told, switches can be yoked so that when one circuit is closed the other circuit is broken. The illustration and explanation of how a yoked switch works reduced the possibility of attributing a participants' failure to solve a problem to the lack of knowledge about this component. The partici-

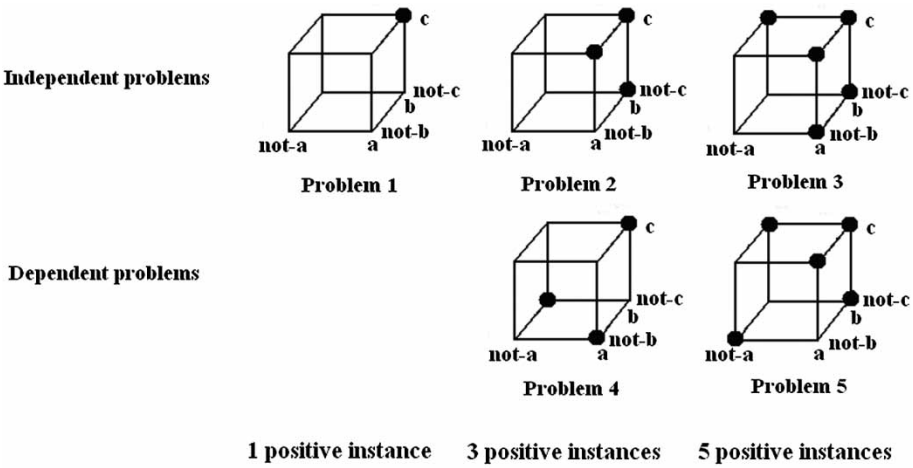

Figure 12. The problems in Experiment 3. Each dimension of a cube represents a switch with two values: on or not on, e.g., *a* and *not a*. A black dot indicates a positive output in which the light comes on. For example, in Problem 1 the light comes on only when all the three switches are up, i.e., *a and b and c*.

pants had 10 minutes to solve each problem. Figure 14 presents solutions to the problems.

Results

All 20 participants solved the initial conjunctive problem. Eleven participants solved each of the two independent problems, one participant solved dependent Problem 4 with three positive outputs, but no participant solved dependent Problem 5 with five positive outputs. The dependent problems were, as predicted, much harder than the independent problems, Wilcoxon test, $z = 3.92$, $p < .001$. But, the "floor effect" in performance with the dependent problems made it hard to discern any effect of the number of positive outputs. The overall trend was reliable, Page's $L = 210.0$, $z = 4.74$, $p < .001$, but it was largely attributable to the ease of Problem 1 with one positive output. We therefore adopted a more refined scoring procedure in which we counted the maximum number of correct outputs, positive and negative, which each circuit yielded, and recorded the score for the circuit on each trial that yielded the highest score. (To use an extreme example, if a participant produced a circuit that comes on when *a* is switched on regardless of the positions of *b* and *c* for Problem 3, then the circuit is scored 7 out of 8, as it produces all outputs for Problem 3 except that of *(not a, and (b and c))*, an on possibility that the circuit fails to produce.) The means out of a maximum of 8 for each three-switch circuit were as follows:

- independent, 3 possibilities: 7.2 (89%)
- independent, 5 possibilities: 6.7 (59%)
- dependent, 3 possibilities: 4.7 (83%)
- dependent, 5 possibilities: 4.4 (55%)

The difference between the independent and the dependent problems was reliable, Wilcoxon $z = 3.30$, $p < .001$, but not the difference between three and five possibilities, Wilcoxon $z = 1.14$, $p = .13$, *ns*. Figure 15 presents the mean latencies for the problems. In general, the greater the number of positive instances, the longer the latencies, Page's $L = 270.5$, $z = 4.82$, $p < .001$, and dependent problems took longer than independent problems, often reaching the 10 minute time limit, Wilcoxon test, $z = 3.52$, $p < .001$.

The much greater difficulty of the dependent problems in comparison with the independent problems corroborated the theory. An effect of the number of positive outputs showed up in the latencies. In the previous experiments, however, the difference in accuracy between *and* and *or*, which are both independent problems, was robust. Yoked switches control multiple pathways simultaneously, and therefore are able to deal

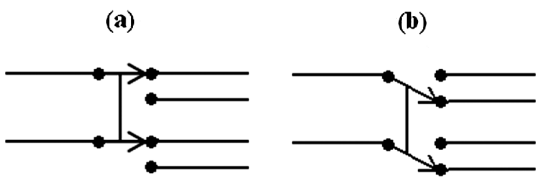

Figure 13. A yoked switch connects two wires on the left either (a) to two "up" wires on the right, or (b) to two "down" wires on the right.

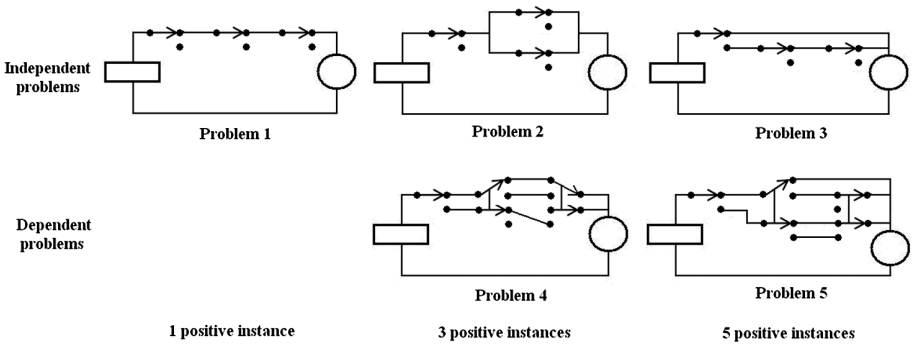

Figure 14. Minimal circuits for the five problems in Experiment 3.

with the interactive effects of the switches in dependent problems. The participants realised that these switches were useful for dependent problems: Among the 16 participants who used yoked switches in the experiment, 11 of them used yoked switches in a greater number of dependent than independent problems, Wilcoxon test, $z = 2.52$, $p < .01$, with two participants showing the opposite results, and three ties. Yet, the dependent effects of the three switches on the light were hard for the participants to engineer.

Figure 14 shows that the circuits for the dependent problems are more complicated than those for the independent problems: They call for nearly double the number of wires. But, as we pointed out earlier (in the section on reverse engineering Boolean systems), this complexity is an inevitable by-product of the theory. Problems 4 and 5 call for the use of yoked switches, and this difference—and the need for more wires—is a consequence of the dependency of their components in controlling the output. Could a measure of complexity of a circuit predict the difficulty of reverse engineering them? The answer, of course, depends on the measure. If complexity is defined in terms of the number of separate wires needed to make a circuit, then this measure fails to predict the difference between *or* and *or else* (see Figure 1). If complexity is defined in terms of the number of variable components, the number of positive outputs, and their dependence, then the measure does predict difficulty. But, the measure is merely a restatement of the present theory.

EXPERIMENTS 4 AND 5: THE CONGRUENCE OF CIRCUITS

The theory postulates an effect of the number of spatial dimensions of a device on the difficulty of its reverse engineering. But, as we pointed out earlier, dimensionality in Boolean circuits also includes the spatial congruency of its components. A congruent circuit for *and* is one dimensional, a congruent circuit for *or* is two dimensional, and a congruent circuit for *or else* is a three dimensional, because its wires have to cross one another. In the latter case, however, the wires can be uncrossed to yield an incongruent but two-dimensional layout of switches (see the section on reverse engineering Boolean systems). In Experiments 1 and 2, the factor of dimensionality was confounded with the effects of dependence and

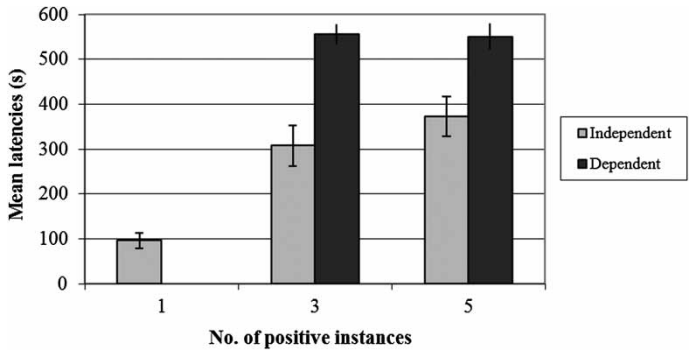

Figure 15. The mean latencies (s) of putative solutions for the five problems in Experiment 3.

positive outcomes. The aim of our final experiments was to unconfound the variables. The introduction of negation into a conjunction or a disjunction has no effect on their independence, but transforms them from congruent problems into incongruent problems: As Figure 16 shows, *a and not-b*, and *a or not-b*, call for one switch to be inserted into the circuit upside down. The introduction of negation into exclusive disjunction has no effect on its dependence, but transforms it from an incongruent problem into a congruent problem: As Figure 16 also shows, *a or else not b* has a two-dimensional solution with the switches in a congruent position, as the two possible circuits from the start terminal to the end terminal make clear:

a	b
not-a	not-b

The switches must either both be up or both be down for the light to come on, and so the circuit can also be described as: *a if and only if b*. Figure 16 shows the circuits for each of the six problems. The theory predicts two additive effects. First, the principles of dependence and positive outputs predict the trend that *and* should be easier than *or*, which should be easier than *or else*. Second, two-dimensional congruent problems should be easier than two-dimensional incongruent problems. Experiments 4 and 5 tested these two predictions.

Method

Experiment 4 examined a set of three congruent problems: *a and b*, *a or b*, and *a or else not-b*, and a set of three incongruent problems: *a and not-b*, *a or not-b*, *a or else b*. Twenty undergraduates from the same population as before (seven male, 13 female; mean age = 19.7 years) carried out all six problems, which were presented to each of them in a different random order. The instructions and procedure were identical to those in Experiment 1, except that the participants had 10 minutes to tackle each problem, and all four configurations of input and output were presented together. Experiment 5 was a replication of Experiment 4 with 20 volunteers from the Chinese University of Hong Kong (four male, 16 female; mean age = 21.6 years) in which the problems were presented in counterbalanced orders of two blocks (*a and b*, *a or b*, *a or else b*; and *a and not-b*, *a or b*, *a or else not-b*). Ten of these participants had studied arts subjects at high school, and 10 had studied science subjects at high school. None of them reported having encountered circuit problems before.

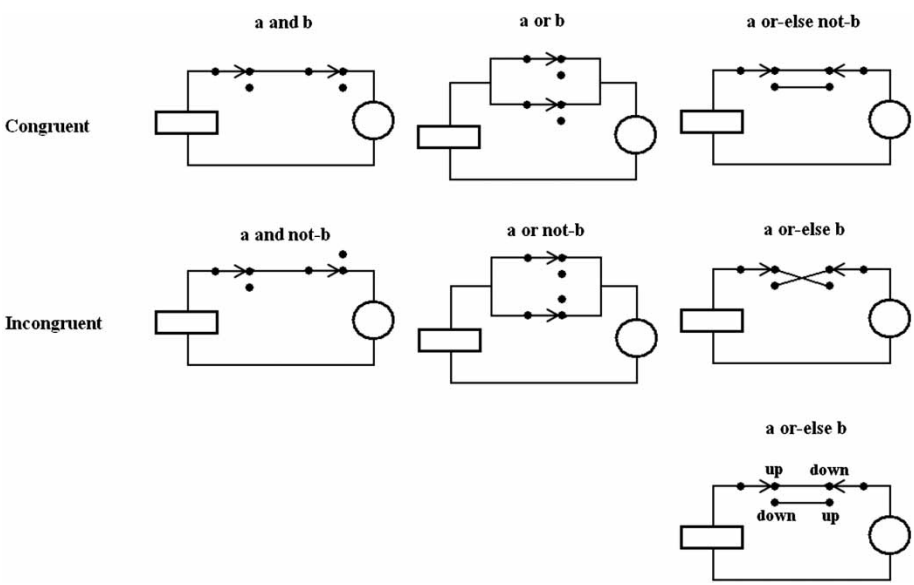

Figure 16. Minimal circuits for the congruent and incongruent problems in Experiment 4. The lowest figure in the right-hand column shows an alternative solution to the *a or else b* problem in which the second switch is inserted upside down, and relabelled as shown in the figure.

Results

Table 4 presents the percentage of correct solutions for the six problems in Experiments 4 and 5. As the table shows, the results corroborated the trend that conjunctions were easier than inclusive disjunctions, which were easier than exclusive disjunctions, Page's $L = 262.5$, $z = 3.56$, $p < .001$ for Experiment 4, and $L = 217.5$, $z = 3.56$, $p < .001$ for Experiment 5. Likewise, the congruent problems were easier than the incongruent problems, Wilcoxon tests, $z = 1.66$, $p < .05$ for Experiment 4, and $z = 3.22$, $p < .001$ for Experiment 5. There was no interaction between congruence and problem type in either experiment, Friedman tests, $\chi^2(2) = 0.95$, $p = .62$, *ns*; and $\chi^2(2) = 0.30$, $p = .86$, *ns*, respectively. The science students in Experiment 5 solved reliably more problems than the arts students, Mann-Whitney U = 18.0, $z = 2.48$, $p < .01$, but both groups showed the same trends in difficulty over the three sorts of problem, Page's $L = 107.0$, $z = 2.91$, $p < .002$, and $L = 129.5$, $z = 2.12$, $p < .02$, for the arts and science groups, respectively). Both groups also performed better with congruent than incongruent problems, Wilcoxon tests, $z = 2.37$, $p < .01$, and $z = 2.23$, $p < .02$.

Figure 17 presents the participants' latencies for putative solutions in Experiment 4, and Figure 18 presents these results for Experiment 5. The predicted trend was corroborated in both experiments, Page's $L = 212.5$, $z = 4.35$, $p < .001$ for Experiment 4, and $L = 263.0$, $z = 3.64$, $p < .001$ for Experiment 5. And the responses to the congruent problems were faster than to the incongruent problems, Wilcoxon tests $z = 2.50$, $p < .01$ for Experiment 4, and $z = 3.58$, $p < .001$ for Experiment 5. Again, no interaction occurred between congruence and problem type in latency in either experiment, Friedman tests, $\chi^2(2) = 2.05$, $p = .36$, *ns*; $\chi^2(2) = 5.3$, $p = .69$, *ns*, respectively. Although the arts participants in Experiment 5 took longer to solve the problems than the science participants (means = 294.1 s and 201.0 s, respectively), Mann Whitney U = 18.0, $z = 2.42$, $p < .01$, the general trend in latencies across the *and*, *or*, and *or else* problems held in both groups: arts (means = 169.7 s, 376.8 s, 335.8 s), Page's $L = 132.0$, $z = 2.68$, $p < .005$, and science (means = 86.8 s, 242.1 s, 270.6 s), Page's $L = 131.0$, $z = 2.46$, $p < .01$.

The two experiments corroborated the theory. The principles of dependence and positive outcomes yielded the predicted trend in difficulty of the three sorts of connective; and the manipulation of dimensionality yielded an additive effect on each sort of problem. In the domain of Boolean circuits, both factors yield separate effects on reverse engineering. Dependent problems are harder to solve than independent problems; and both sorts of problem become still harder when they call for the switches to be inserted into circuits in incongruent orientations. As Experiment 5 showed, the theory of reverse engineering held for both science students and arts students. We conclude that dimensionality, which includes the congruence of components, exerts a main effect in our studies, but cannot alone account for the effects of either the number of positive outcomes or the dependency of the components in yielding them.

GENERAL DISCUSSION

Reverse engineering is the process of working out how to assemble components with known properties into a system that has the input–output relations of a target system. It is therefore a special sort of problem solving, though not one that appears to have been studied by psychologists before. According to the theory presented in this paper, individuals tackle reverse engineering by adopting one of two main initial strategies: They focus either on single outputs one at a time, or on single components one at a time, and then search for a solution to this part of the problem guided by both local and global constraints. Once they have solved such a subproblem, they seek to extend its solution to the next part of the problem, and so on. Ultimately, they need to

TABLE 4
The percentages of correct solutions for the problems in Experiment 4 with Princeton students (left-hand column), and in Experiment 5 with Hong Kong science students (middle column) and Hong Kong arts students (right-hand column)

	and	*or*	*or else*
Congruent circuits	95 100 90	65 80 40	50 70 20
Incongruent circuits	93 80 60	58 50 10	38 30 10
Overall	94 90 75	62 65 25	44 50 15

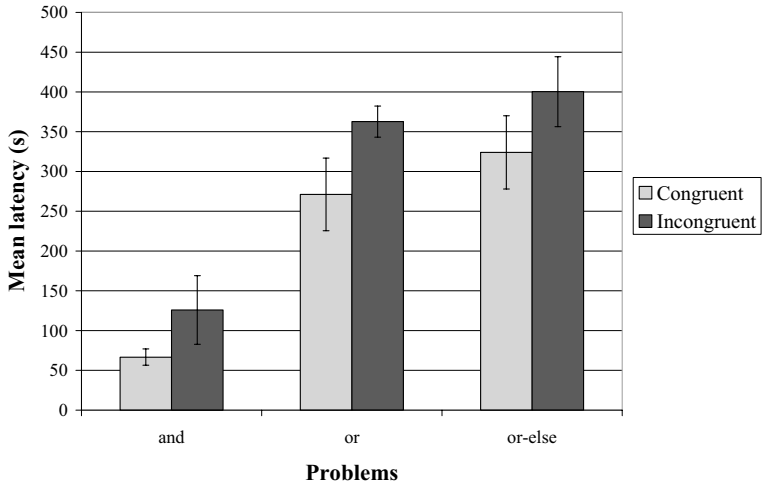

Figure 17. The mean latencies (s) of putative solutions for the six problems in Experiment 4.

devise a system that maps all inputs onto their appropriate outputs. A computer program that we implemented to solve reverse-engineering problems bore out the importance of constraints, and especially global constraints, on the search for solutions (see Table 1). And this factor was also corroborated in the sequences of diagrams that the participants drew in attempting to solve problems in Experiment 2. The theory implies that, for Boolean systems, a focus on outputs is more useful than a focus on components, because a circuit based on a single component often provides no basis for an extension to the correct circuit including the other components.

As an applied problem, reverse engineering calls for the use of a diverse range of cognitive processes including deduction, induction, and creativity. As a result, the principles governing reverse engineering must necessarily overlap with key principles in those fields. What, then, does the current theory add to our understanding of the psychological processes underlying reverse engineering? The theory identifies three principal factors that affect the difficulty of the process. First, the more variable components there are in a system, the harder the task should be (the principle of variable components). As the number of these components increases, so too does the number of distinct states into which a finite-state automaton can enter, which depends on the number of variables and the numbers of their possible values. Second, the more distinct states of the system that yield an output, as opposed to no output, the harder the task should be (the princi-

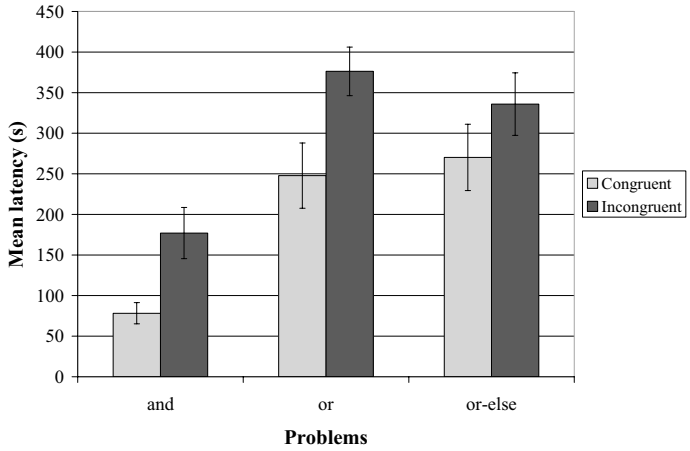

Figure 18. The mean latencies (s) of putative solutions for the six problems in Experiment 5.

ple of positive outputs). Third, the greater the degree to which the system resists decomposition into separate subsystems, because the effects of components are dependent on one another, the harder the task should be (the principle of dependence). Dependence is the crucial factor, and, as we showed, it predominates over the number of positive outputs. In devices with a spatial layout of components, such as Boolean circuits, the greater the number of dimensions or the need for incongruent orientations of components also adds to the difficulty of the task.

Our evidence corroborated the theory. When individuals have to reverse engineer Boolean systems, such as switch circuits or water-flow systems, they tend to focus on the performance of the system one output at a time, or, in a few cases, on one component at a time. This claim was borne out in the sequences of drawings that the participants made and in their remarks as they thought aloud (Experiments 1 and 2). As these experiments also showed, independent problems based on conjunction or inclusive disjunction were easier than dependent problems based on exclusive disjunction. The latter yields only two positive outputs, but the two components are dependent on one another to yield both positive outputs and negative outputs. Dependence is the major factor in reverse engineering Boolean systems (Experiments 1, 2, 4, and 5). Indeed, dependent problems with three switches were impossible to solve for almost all of our participants (Experiment 3).

Within independent problems, the experiments showed that the number of positive outputs had a robust effect: The reverse engineering of *a and b* (one positive output) was always easier than the reverse engineering of *a or b* in its inclusive sense (three positive outputs). Experiment 2 showed that the theory predicted the results for water-flow problems, and no transfer occurred from the circuit problems to this domain, or vice versa. The superficial differences between a switch and a faucet appeared to mask the underlying equivalences in Boolean logic. We made no direct experimental test of the effects of the number of variable components in a system. But, the contrast between three switches (Experiment 3) and two switches (the remaining experiments) suggests that there is such an effect, and we have no doubt that with, say, five positive outputs, a problem with 10 variable components is likely to be much harder than a problem with only three variable components.

Dimensionality in Boolean circuits includes the congruence of the components in the circuit, because one variable can compensate for the other. And, as the theory predicts, incongruence increases the difficulty of reverse engineering: The participants tended to discover two-dimensional solutions with incongruent switch orientations rather than three-dimensional solutions in which wires crossed one another. For example, in Experiment 1, 6 participants succeeded in reverse engineering *or else*, but none of them crossed wires in their solutions. They preferred instead to insert one switch into the circuit upside-down. In contrast, when an exclusive disjunction has a negated component, such as *a or else not-b*, it has a congruent two-dimensional solution. In this way, the results of Experiments 4 and 5 showed robust effects of congruence, which did not interact with those of dependence and the number of positive outcomes, and the predicted trend in difficulty over *and*, *or*, and *or else*.

In the present studies, we have employed different methodologies to understand the psychological processes underlying reverse engineering, including computer modelling, experiments with and without think-aloud requirements, and with and without real switches. There seems to be converging evidence to support our theory. However, do the results merely show that complex systems are more difficult to reverse engineer than simpler systems? Moreover, does the theory merely spell out the factors underlying the difficulty of reverse engineering Boolean systems, rather than revealing the underlying psychological processes themselves? First, as we showed, the complexity of a circuit is a direct consequence of the principles in our theory. It is impossible to convert a problem with independent components into one with dependent components without increasing the complexity of the circuit (see Figure 14). Dependency is bound to call for a more complex circuit precisely because outputs now depend, by definition, on the joint settings of more switches. Similarly, the hypothesis that difficulty in reverse engineering merely reflects the general difficulty of *and*, *or*, and *or else* cannot be true. In particular, exclusive disjunctions are not always more difficult to deal with than inclusive disjunctions. In deductive reasoning, exclusive disjunctions are reliably *easier* than inclusive disjunctions (see, e.g., Bauer & Johnson-Laird, 1993); the same difference occurs in the acquisition of concepts (see, e.g., Goodwin & Johnson-Laird, 2011). The reason is that exclusive disjunctions are consistent with fewer possibilities than inclusive disjunctions. The nature of the task

is therefore crucial. In deduction, the task is to determine what follows from all the possibilities consistent with the premises, and so a critical variable is the number of these possibilities. In reverse engineering, the task is to decompose a system into subsystems, and that task should be more difficult when it is impossible to divide the system into independent subsystems. More importantly, as both the computer model and Experiment 2 have demonstrated, the difficulty of reverse engineering depends also on how individuals master both local and global constraints to a problem, as well as the strategies that they choose to tackle the problem. Future studies should certainly investigate how individuals make the leap from focusing on either one possibility or one component at a time, to arriving at a solution for dependent problems.

Expert reverse engineers are likely to have developed an extensive knowledge of the domain of their expertise and to exploit it in their reverse engineering. Indeed, they may know at once how to reverse engineer, say, a computer program or a digital camera, just as an expert fire chief knows at once how to tackle a fire (e.g., Klein, 1997; Klein & Wolf, 1995), a chess master sees at once moves that are feasible (e.g., Chase & Simon, 1973), or a skilled practitioner knows at once how to make a diagnosis (Crandall & Calderwood, 1989). Their aim may be to satisfice (Simon, 1957), i.e., to find a workable solution, or it may be to optimise, i.e., to find the best solution. Experts have knowledge, which they can use to save them from thinking (Chi, Glaser, & Farr, 1988). But, when a problem lies outside this aspect of their competence, they are forced to fall back on their basic skills, and, in the case of expert reverse engineers, to fall back on the principles that our theory aims to capture.

Could the current theory be modelled under cognitive architectures such as SOAR (*S*tate, *O*perator, *A*nd *R*esult; Newell, 1990) or ACT-R (Atomic Components of Thought–Revised; Anderson & Lebiere, 1998)? As we have argued elsewhere (see Johnson-Laird, 1988), cognitive architectures have the power of a universal Turing machine, and therefore can in principle accommodate any consistent pattern of results. For instance, as a general model of human cognition, Newell's (1990) SOAR architecture has the problem solving heuristic of means–ends analysis as its core, and hence would readily explain why independent systems are much easier to reverse engineer than dependent ones. Yet, the key contribution of the theory here is its integration of pertinent cognitive principles to explain the specific psychological processes of reverse engineering, while also taking into account domain-specific constraints, such as congruence, as in the domain of Boolean circuits.

Boolean switch circuits are extraordinarily powerful, but a legitimate question is whether the present theory of reverse engineering extends to other domains, such as digital cameras or car engines. There are three reasons to suppose that the theory should extend to such domains. The first reason is that Boolean relations underlie many aspects of such systems, though plainly they also depend on other sorts of relation too. In the case of digital cameras, the control functions often depend on Boolean relations. Here, for example, is the instruction for switching from one mode, such as the flash is on, to another mode, for a common make of digital camera: ''Press the button (1) and use the left-arrow or right-arrow button to switch between modes (2).'' Many of the other instructions also call for the conjunction of one operation with another operation from a disjunctive set. As these instructions show, devices themselves can implement Boolean relations between their components.

The second reason to suppose that the theory extends to other domains appeals directly to the theory's three main principles. The number of variable components and the number of positive outputs are likely to be major contributors to the difficulty of reverse engineering any system or artifact, and the number of positive outputs is the more critical factor, e.g., five variable components are easy to cope with if they yield only one positive output, but impossible to cope with if they yield 10 different positive outputs. By far the most dominant principle, however, is likely to be dependence. When variables interact in their effects, it is hard to predict the performance of a system (see Dörner's, 1996, studies of the simulation of complex systems), and if one cannot understand what the system is doing, then its reverse engineering is all but impossible. When dependence conflicts with, say, the number of variable components, or the number of positive outputs, dependence should be dominant. Hence, a device with a handful of components yielding only a few positive outputs may nevertheless be very hard to reverse engineer in case the outputs are highly dependent on the settings of all the components. Dimensionality is relevant only to problems that concern solutions in physical

devices, and it does not apply, say, to reverse engineering a computer program.

The final reason supporting the extension of the theory to other domains is due to Simon (1996). In his influential analysis of artificial systems ranging from cognitive mechanisms to economic systems, he argued that such systems—including the Boolean systems studied here—tend to have a hierarchical structure that is decomposable. This argument has stood the test of time, and thus we have good reasons to believe that Boolean systems are a reasonable test bed for a general theory of reverse engineering. The extension of the theory to non-Boolean devices—if an experimentally tractable domain can be found—may well yield surprises, but, from the evidence of our studies, the dependence of components, the number of positive outcomes, and the number of variable components, are likely to affect the reverse engineering of any system.

Original manuscript received June 2012
Revised manuscript received January 2013
First published online March 2013

REFERENCES

Anderson, J. R., & Lebiere, C. (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum Associates.

Bauer, M. I., & Johnson-Laird, P. N. (1993). How diagrams can improve reasoning. *Psychological Science*, *4*, 372–378.

Bucciarelli, M., & Johnson-Laird, P. N. (2005). Naïve deontics: A theory of meaning, representation, and reasoning. *Cognitive Psychology*, *50*, 159–193.

Carlson, R., Chandler, P., & Sweller, J. (2003). Learning and understanding science instructional material. *Journal of Educational Psychology*, *95*, 629–640.

Chase, W. G., & Simon, H. A. (1973). Perception in chess. *Cognitive Psychology*, *4*, 55–81.

Chi, M. T. H., Glaser, R., & Farr, M. J. (1988). *The nature of expertise*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Clarkson, T. G., Gorse, D., & Taylor, J. G. (1992). From wetware to hardware: Reverse engineering using probabilistic RAMs. *Journal of Intelligent Systems*, *2*, 11–30.

Crandall, B., & Calderwood, R. (1989). *Clinical assessment skills of experienced neonatal intensive care nurses (final report)*. Yellow Springs, OH: Klein Associates.

Csete, M. E., & Doyle, J. C. (2002). Reverse engineering of biological complexity. *Science*, *295*, 1664–1669.

Dörner, D. (1996). *The logic of failure: Why things go wrong and what we can do to make them right*. New York, NY: Henry Holt.

Eilam, E. (2005). *Reversing: Secrets of reverse engineering*. Indianapolis, IN: Wiley.

Ericsson, K. A., & Simon, H. A. (1984). *Protocol analysis: Verbal reports as data*. Cambridge, MA: MIT Press.

Evans, J. St. B., Newstead, S. E., & Byrne, R. M. J. (1993). *Human reasoning: The psychology of deduction*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Feldman, J. (2000). Minimization of Boolean complexity in human concept learning. *Nature*, *407*, 630–634.

Feldman, J. (2006). An algebra of human concept learning. *Journal of Mathematical Psychology*, *50*, 339–368.

Fleck, J. I., & Weisberg, R. W. (2004). The use of verbal protocols as data: An analysis of insight in the candle problem. *Memory and Cognition*, *32*, 990–1006.

Gold, E. M. (1967). Language identification in the limit. *Information and Control*, *10*, 447–474.

Goodwin, G. P., & Johnson-Laird, P. N. (2005a). Diagnosis of ambiguous faults in simple networks. *Proceedings of the 27th annual conference of the Cognitive Science Society, Stresa, Italy* (pp. 791–796). Mahwah, NJ: Lawrence Erlbaum Associates.

Goodwin, G. P., & Johnson-Laird, P. N. (2005b). Reasoning about relations. *Psychological Review*, *112*, 468–493.

Goodwin, G. P., & Johnson-Laird, P. N. (2010). Conceptual illusions. *Cognition*, *114*, 253–265.

Goodwin, G. P., & Johnson-Laird, P. N. (2011). Mental models of Boolean concepts. *Cognitive Psychology*, *63*, 34–59.

Gopnik, A., Sobel, D. M., Schulz, L. E., & Glymour, C. (2001). Causal learning mechanisms in very young children: Two-, three-, and four-year-olds infer causal relations from patterns of variation and covariation. *Developmental Psychology*, *37*, 620–629.

Hagmayer, Y., & Waldmann, M. R. (2002). How temporal assumptions influence causal judgements. *Memory and Cognition*, *30*, 1128–1137.

Halford, G. S., Wilson, W. H., & Phillips, S. (1998). Processing capacity defined by relational complexity: Implications for comparative, developmental, and cognitive psychology. *Behavioral and Brain Sciences*, *21*, 803–831.

Hegarty, M. (1992). Mental animation: Inferring motion from static diagrams of mechanical systems. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *18*, 1084–1102.

Holland, J. H., Holyoak, K. J., Nisbett, R. E., & Thagard, P. R. (1986). *Induction: Processes of inference, learning, and discovery*. Cambridge, MA: MIT Press.

Hopcroft, J. E., & Ullman, J. D. (1979). *Formal languages and their relation to automata*. Reading, MA: Addison-Wesley.

Jeffrey, R. (1981). *Formal logic: Its scope and limits* (2nd ed.). New York, NY: McGraw-Hill.

Johnson-Laird, P. N. (1988). *The computer and the mind: An introduction to cognitive science*. Cambridge, MA: Harvard University Press.

Johnson-Laird, P. N. (2005). Flying bicycles: How the Wright brothers invented the airplane. *Mind and Society*, *4*, 27–48.

Johnson-Laird, P. N. (2006). *How we reason*. New York, NY: Oxford University Press.

Johnson-Laird, P. N., & Byrne, R. M. J. (1991). *Deduction*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Johnson-Laird, P. N., & Savary, F. (1999). Illusory inferences: A novel class of erroneous deductions. *Cognition*, *71*, 191–229.

Klahr, D. (2000). *Exploring science: The cognition and development of discovery processes*. Cambridge, MA: MIT Press.

Klahr, D. (2005). A framework for cognitive studies of science and technology. In M. E. Gorman, R. D. Tweney, D. C. Gooding, & A. P. Kincannon (Eds.), *Scientific and technological thinking* (pp. 81–95). Hillsdale, NJ: Lawrence Erlbaum Associates.

Klahr, D., & Dunbar, K. (1988). Dual space search during scientific reasoning. *Cognitive Science*, *12*, 1–48.

Klein, G., & Wolf, S. (1995). Decision-centered training. In *Proceedings of the 39th annual meeting of the Human Factors and Ergonomics Society, San Diego, CA* (pp. 1249–1252). Santa Monica, CA: Human Factors and Ergonomics Society.

Klein, G. A. (1997). Developing expertise in decision making. *Thinking and Reasoning*, *3*, 337–352.

Kruschke, J. K. (1992). ALCOVE: An exemplar-based connectionist model of category learning. *Psychological Review*, *99*, 22–44.

Lagnado, D. A., & Sloman, S. A. (2004). The advantage of timely intervention. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *30*, 856–876.

Lagnado, D. A., Waldmann, M. R., Hagmayer, Y., & Sloman, S. A. (2007). Beyond covariation: Cues to causal structure. In A. Gopnik & L. E. Schulz (Eds.), *Causal learning: Psychology, philosophy, and computation* (pp. 154–172). New York, NY: Oxford University Press.

Lee, N. Y. L., & Johnson-Laird, P. N. (2013). Strategic changes in problem solving. *Journal of Cognitive Psychology*, *25*, 165–173.

Li, H., Linke, W., Oberhauser, A. F., Carrion-Vazquez, M., Kerkvliet, J. G., Lu, H., . . . Fernandez, J. M. (2002). Reverse engineering of the giant muscle protein titin. *Nature*, *418*, 998–1002.

Love, B. C., Medin, D. L., & Gureckis, T. M. (2004). SUSTAIN: A network model of category learning. *Psychological Review*, *111*, 309–332.

Minsky, M. L., & Papert, S. A. (1969). *Perceptrons*. Cambridge, MA: MIT Press.

Miyake, N. (1986). Constructive interaction and the iterative process of understanding. *Cognitive Science*, *10*, 151–177.

Moray, N. (1999). Mental models in theory and practice. In D. Gopher & A. Koriat (Eds.), *Attention and performance XVII: Cognitive regulation of performance: Interaction of theory and application* (pp. 223–258). Cambridge, MA: MIT Press.

Newell, A. (1990). *Unified theories of cognition*. Cambridge, MA: Harvard University Press.

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

Nisbett, R., & Wilson, T. (1977). Telling more than we can know: Verbal reports on mental processes. *Psychological Review*, *84*, 231–259.

Nosofsky, R. M., Gluck, M. A., Palmeri, T. J., McKinley, S. C., & Glauthier, P. (1994). Comparing models of rule-based classification learning: A replication and extension of Shepard, Holvand, and Jenkins (1961). *Memory and Cognition*, *22*, 352–369.

Rouse, W. B., & Hunt, R. M. (1984). Human problem solving in fault diagnosis tasks. In W. B. Rouse (Ed.), *Advances in man-machine systems research* (Vol. 1, pp. 195–222). Greenwich, CT: JAI Press.

Schooler, J. W., Ohlsson, S., & Brooks, K. (1993). Thoughts beyond words: When language overshadows insight. *Journal of Experimental Psychology: General*, *12*, 166–183.

Schulz, L. E., & Gopnik, A. (2004). Causal learning across domains. *Developmental Psychology*, *40*, 162–176.

Schwartz, D., & Black, J. B. (1996). Analog imagery in mental model reasoning: Depictive models. *Cognitive Psychology*, *30*, 154–219.

Shepard, R. N., Hovland, C. I., & Jenkins, H. M. (1961). Learning and memorization of classifications. *Psychological Monographs: General and Applied*, *75*, 1–42.

Simon, H. A. (1957). *Models of man: Social and rational*. New York, NY: Wiley.

Simon, H. A. (1996). *The sciences of the artificial* (3rd ed.). Cambridge, MA: MIT Press.

Sweller, J., & Chandler, P. (1994). Why some material is difficult to learn. *Cognition and Instruction*, *12*, 185–223.

Tegnér, J., Yeung, M. K. S., Hasty, J., & Collins, J. J. (2003). Reverse engineering gene networks: Integrating genetic perturbations with dynamical modelling. *Proceedings of the National Academy of Sciences of the USA*, *100*, 5944–5949.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the Association for Computing Machinery*, *27*, 1134–1142.

Van der Henst, J.-B., Yang, Y., & Johnson-Laird, P. N. (2002). Strategies in sentential reasoning. *Cognitive Science*, *26*, 425–468.

van Merriënboer, J. J. G., & Sweller, J. (2005). Cognitive load theory and complex learning: Recent developments and future directions. *Educational Psychology Review*, *17*, 147–177.

Vapnik, V. (1998). *Statistical learning theory*. New York, NY: Wiley.

Várady, T., Martin, R. R., & Cox, J. (1997). Reverse engineering of geometric models—an introduction. *Computer-Aided Design*, *29*, 255–268.

Vigo, R. (2009). Categorical invariance and structural complexity in human concept learning. *Journal of Mathematical Psychology*, *53*, 203–221.

White, P. A. (1995). Use of prior beliefs in the assignment of causal roles: Causal powers versus regularity-based accounts. *Memory and Cognition*, *23*, 243–254.