Journal of Cognitive Psychology

ISSN: 2044-5911 (Print) 2044-592X (Online) Journal homepage: http://www.tandfonline.com/loi/pecp21

Children's creation of algorithms: simulations and gestures

Monica Bucciarelli, Robert Mackiewicz, Sangeet S. Khemlani & Philip N. Johnson-Laird

To cite this article: Monica Bucciarelli, Robert Mackiewicz, Sangeet S. Khemlani & Philip N. Johnson-Laird (2016): Children's creation of algorithms: simulations and gestures, Journal of Cognitive Psychology, DOI: 10.1080/20445911.2015.1134541

To link to this article: http://dx.doi.org/10.1080/20445911.2015.1134541

Published online: 17 Jan 2016.

JOURNAL OF COGNITIVE PSYCHOLOGY, 2016
http://dx.doi.org/10.1080/20445911.2015.1134541

Routledge
Taylor & Francis Group

Children's creation of algorithms: simulations and gestures

Monica Bucciarelli[a], Robert Mackiewicz[b], Sangeet S. Khemlani[c] and Philip N. Johnson-Laird[d,e]

[a]Centro di Scienza Cognitiva and Dipartimento di Psicologia, Università di Torino, Torino, Italy; [b]Department of Psychology, University of Social Sciences and Humanities, Chodakowska 19/31, Warsaw, Poland; [c]Navy Center for Applied Research in Artificial Intelligence, Naval Research Lab, Washington, DC, USA; [d]Stuart Professor of Psychology, Princeton University, Princeton, NJ, USA; [e]Department of Psychology, New York University, 301 E 22nd Street, New York, NY, USA

ABSTRACT

Experiments showed that children are able to create algorithms, that is, sequences of operations that solve problems, and that their gestures help them to do so. The theory of mental models, which is implemented in a computer program, postulates that the creation of algorithms depends on mental simulations that unfold in time. Gestures are outward signs of moves and they help the process. We tested 10-year-old children, because they can plan, and because they gesture more than adults. They were able to rearrange the order of 6 cars in a train (using a siding), and the difficulty of the task depended on the number of moves in minimal solutions (Experiment 1). They were also able to devise informal algorithms to rearrange the order of cars when they were not allowed to move the cars, and the difficulty of the task depended on the complexity of the algorithms (Experiment 2). When children were prevented from gesturing as they formulated algorithms, the accuracy of their algorithms declined by13% (Experiment 3). We discuss the implications of these results.

ARTICLE HISTORY
Received 16 March 2015
Accepted 15 December 2015

KEYWORDS
Abduction; algorithmic reasoning; gestures; mental simulation; mental models

Our research concerns the creation of informal algorithms. They are descriptions of a sequence of steps that transforms an input into an output, so they can solve problems. Formal algorithms are embodied in computer programs, but informal algorithms are the substance of much of human reasoning. People describe them for daily activities such as setting a table, repairing a bicycle, and driving directions to a destination. They are explanations of *how* to carry out a particular process that solves a problem. Not all explanations are algorithms, for example, an explanation of why New Orleans was flooded during hurricane Katrina. Computer programming presupposes the ability to devise informal algorithms, but adults who are not programmers are able to devise them too (Khemlani, Mackiewicz, Bucciarelli, & Johnson-Laird, 2013). It seems reasonable to suppose that the ability has its roots in childhood, but prior to our research no-one knew whether children were able to devise informal algorithms. Our first goal was to find out.

The first issue was to find a domain suitable for studying algorithms. Psychologists have studied many sorts of problem, such as the well-known X-ray problem, the Tower of Hanoi, Sudoku puzzles, and so on and on. Many of these problems are in reality just a single problem, for example, the X-ray puzzle, which is to devise a set up in which X-rays can kill a tumour without damaging the healthy surrounding tissue (Duncker, 1945). Others are a little more general, though a single algorithm can in principle solve all instances of them, for example, Tower problems (Aamodt-Leeper, Creswell, McGurk, & Skuse, 2001; Huizenga, Dolan, & van der Molen, 2006; Luciana & Nelson, 1998; Shallice, 1982). Studies of tower problems showed that working memory is needed to hold and to update information (Miyake et al., 2000; Unterrainer & Owen, 2006). It is also needed for informal programming. But, we needed a domain in which there was a plentiful supply of problems, which each called for a different algorithm, and for which we could manipulate the complexity of the algorithms. The rearrangement of cars on a toy railway track was promising, because children could at least understand the domain. Figure 1 shows the track we

CONTACT Monica Bucciarelli monica.bucciarelli@unito.it

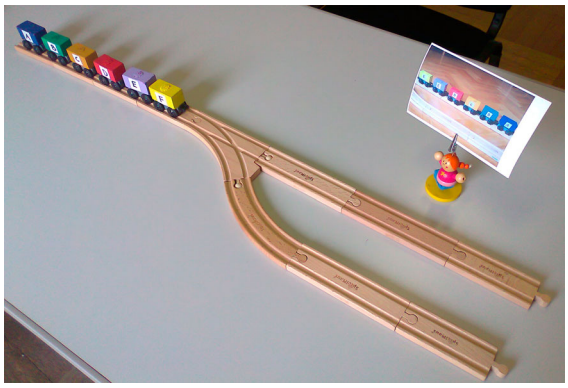

Figure 1. The railway with the cars in their initial arrangement (A B C D E F) on left track, the empty siding, the empty right track, and a picture behind the track of the rearrangement required on right track (which reads F E D C B A).

used in our investigations. It shows the arrangement at the start of a problem. Labelled cars are in an order on the left track. The task is to rearrange the cars, using the siding as necessary, so that they are lined up in the required order on the right track. This order is shown in the picture behind the right side of the track. Only three sorts of move are allowed: one or more cars can be moved from the left to the right track, from the left track to the siding, and from the siding to the left track. Hence, various moves are not allowed, including a move from the siding straight to the right track, a move from the right track back to the left track, or a move that lifts one car over another. The participants can move several cars at once: they hold one car, and then push it and any cars in front of it to their new position on the track.

The domain has three essential properties for the study of algorithms. First, there are many different sorts of rearrangement—in theory an infinite number. For example, the order of the cars in a train can be reversed, so that the order *abcdef* becomes *fedcba*, or they can be interleaved in a parity sort so that all the cars in odd-numbered positions precede those in even-numbered positions, for example, the order *abcdef* becomes *acebdf*. Readers familiar with Turing machines will notice the similarity of the railway domain to them. With the ability to scan the contents of cars (e.g. binary digits), and to add or remove these contents, the railway system has the power of a Universal Turing machine, because both the siding and the left track can function as stacks to store information (Hopcroft & Ullman, 1979). It is therefore possible to emulate the process of solving the Tower of Hanoi in the railway environment. However, our studies examined only simple rearrangements, such as reversals and parity sorts.

Second, the algorithms for rearrangements differ in complexity, and so we could manipulate their complexity as an independent variable in our experiments.

Third, a crucial difference exists between making a rearrangement and creating an algorithm to make it, and the railway environment offers the possibility to investigate both abilities. Making a rearrangement is like assembling a piece of furniture given the parts and a diagram representing the finished result: the person assembling the furniture moves the parts into the positions shown in the diagram. In contrast, devising an algorithm is like your telling this person what to do when you can see the diagram but she cannot. The contrast is clear in the instructions for the two sorts of task.

The key instruction for making a rearrangement is: "You have to form a train on the right track in the order shown in the photograph behind the track, and you can make the three sorts of move". The participants use the picture of the goal to decide which cars to move and where to move them. They can use a means to partial-ends strategy. That is, they can decompose the ends—the required order of the cars on the right track—in order to achieve them in a piecemeal way, one or more adjacent cars at a time (cf. Newell & Simon, 1972). The following procedure, for example, is a simple way to make any rearrangement:

(1) Find the positions of the car(s) on the left track that are furthest to the right in the goal.
(2) If cars are in front of these cars then move the blocking cars to the siding.
(3) Move the target car(s) to the right track.
(4) Return the cars on the siding, if any, to the left track.
(5) Repeat from Step 1 until the left track is empty.

The procedure makes rearrangements, but it fails to do so in a minimal number of moves, tending to move cars off the siding unnecessarily. We describe a more efficient algorithm below.

In contrast, the key instruction for devising an algorithm is: "You have to describe the moves needed to yield a train in the required order on the right track, but you are not allowed to move the cars". Here is an example of an algorithm, which reverses the order of cars:

(1) Move all but one car on the left track to the siding.
(2) While there are cars on the siding, move one car from the left track to the right track, and move one car from the siding to the left track.
(3) When there are no cars on the siding, move one car from the left track to the right track.

Step 2 in the algorithm calls for a loop of operations until all the cars have been shifted from the siding. We use diagrams, such as:

a[bcdef] -

to represent the current state of the track, where in this case car *a* is on the left track, cars *bcdef* are on the siding, which the square brackets demarcate, and "-" shows that the right track is empty. The algorithm above yields the following sequence of moves:

abcdef[-] -, a[bcdef] -, - [bcdef]a, b[cdef]a, [cdef]ba
… and so on until - [-]fedcba.

Each sort of rearrangement requires its own algorithm, and, as the preceding example illustrates, the algorithms are quite different from the one above for making rearrangements.

The creation of an algorithm is neither a deduction from observations nor a mere inductive generalisation from them, though it may use both processes. Following Peirce (1955, Chapter 11) we use the term *abduction* to refer to the special sort of induction that underlies the creation of explanations. The abduction of an algorithm according to the theory of mental models, which is described in detail in the next section, depends on a kinematic simulation, that is, a mental model that unfolds in time in the same temporal order as the sequence of events that it represents (e.g. Hegarty, 1992; Johnson-Laird, 1983). The simulation has to be translated into a verbal description of the algorithm.

The two tasks of making a rearrangement and of formulating an algorithm to make it should differ in difficulty. The difficulty of making a rearrangement should depend on the minimal numbers of moves and of cars to be moved that it requires—just as, for example, the difficulty of a tower problem depends on the number of disks to be rearranged. If the minimal numbers of moves and cars to be moved increases over a set of different rearrangements, then the difficulty of making them should increase too. In contrast, the difficulty of formulating an algorithm should depend on the *complexity* of the algorithm rather than on the number of moves that its execution calls for. As complexity increases over a set of algorithms, so too should the difficulty of formulating them. We return to these two points later in order to spell out the predictions for our experiments. Evidence corroborates the difference between the two tasks in a study of adults who made rearrangements, and who formulated algorithms to make them (Khemlani et al., 2013). The evidence was the actual numbers of moves that the participants took to make rearrangements and the accuracy of their algorithms. But, could there be more direct evidence for the use of mental simulations?

According to the model theory, individuals' gestures as they try to formulate algorithms provide an outward sign of inward simulations. When children are not allowed to move the cars in our studies, they have to imagine moving them. And this mental simulation should control their gestures, showing the moves of the cars that they envisage. But, their gestures should also improve the accuracy of the algorithms that they formulate. The effect should occur for two reasons (cf. Hostetter & Alibali, 2008, 2010; Marghetis & Benjamin, 2014). First, when children use a *deictic* gesture to point to a car and then an *iconic* gesture to show its movement from one position to another, they have no need to name the car or to describe its movement in their description of the algorithm. This reduced demand on descriptive processes allows them to focus more on the formulation of the algorithm. Second, although their gestures are under the control of a mental simulation, the very act of pointing and gesturing should help the children to remember what cars are where at the current juncture of the algorithm. This stage in the kinematic model is crucial for formulating the next move in the algorithm. Hence, gestures should have a causal effect: they should make it easier to formulate accurate algorithms. On this account, gestures are both an output from and an input to kinematic simulations.

Previous studies of other tasks are consistent with this account: during descriptions of spatial relations, iconic gestures reduce the need to describe referents (Chu & Kita, 2011; Sauter, Uttal, Alman, Goldin-Meadow, & Levine, 2012), and they help speakers to maintain information in memory (see e.g. Morsella & Krauss, 2004; Wesp, Hess, Keutmann, & Wheaton, 2001). Studies have likewise shown that deictic gestures can improve memory for their referents (Petkov & Nikolova, 2010). Their effects should therefore be important when children are trying to create an algorithm but are not allowed to move

the cars, because their gestures help them to envisage what is going on, and to finesse the description of moves. It follows that the task of formulating an algorithm should be more difficult—more prone to error, and take longer—if the participants are unable to gesture. The second aim of our research was to test this prediction, and to examine the effects of deictic and iconic gestures on descriptions of algorithms.

Three main hypotheses exist about the relation between gestures and thinking. The first is that gestures are epiphenomena: they are consequences of thinking but play no causal role in its process (e.g. Kita & Ozyurek, 2003; Melinger & Kita, 2007). Speakers on mobile phones, for example, often gesture for no apparent reason. The second hypothesis is that gestures are communicative (e.g. Hostetter, 2011). Conventional gestures such as those used in Naples, or well-known obscene signs, are intentional communications (McNeill, 1992), but the hypothesis allows that gestures can also be tacit communications that individuals generate unconsciously as they speak. The third hypothesis is that gestures not only reflect thinking, but also affect it and can improve it (e.g. Goldin-Meadow, Nusbaum, Kelly, & Wagner, 2001; Wagner, Nusbaum, & Goldin-Meadow, 2004). Evidence corroborates the hypothesis. Studies have compared performance when participants are allowed to gesture with performance when they are not allowed to gesture. They show that gesturing improves performance in learning math (e.g. Goldin-Meadow, Cook, & Mitchell, 2009), in mental rotation tasks (e.g. Chu & Kita, 2011), in spatial reasoning (Ehrlich, Levine, & Goldin-Meadow, 2006), in simulating a system of gears (Alibali, Spencer, Knox, & Kita, 2011), in learning science from texts (e.g. Cutica, Ianì, & Bucciarelli, 2014), and in the production of discourse (e.g. Alibali & Kita, 2010). As the model theory implies, gestures can be outward signs of mental simulations (Hostetter & Alibali, 2008, 2010) and a natural way of expressing their results (Hegarty, Mayer, Kriz, & Keehner, 2005).

Of course, the three hypotheses are not mutually exclusive. Some gestures may be epiphenomena, but others, as the model theory postulates, can both help speakers to communicate (by allowing them not to have to describe moves in rearrangement), and assist the process of thinking (by providing an aid to remembering where various cars are located on the railway).

An ideal population for our studies was 10-year-old children. On the one hand, they have the ability to plan (Keen, 2011), to carry out means-ends analyses (Kuhn, 2013), to make the rearrangements in the Tower of Hanoi problem (e.g. Aamodt-Leeper et al., 2001), and to make mental simulations of actions (Skoura, Vinter, & Papaxanthis, 2009). Hence, they should be able to cope with the rearrangements of trains. However, their working memories are not developed in full (e.g. Cowan, 2005; Unterrainer & Owen, 2006), are not near to adult capacity until late adolescence (Luna, Garver, Urban, Lazar, & Sweeney, 2004; Welsh, Pennington, & Grossier, 1991), and continue to develop into early adulthood (Albert & Steinberg, 2011). It is this capacity that predicts their performance in inferential tasks (Barrouillet, Grosset, & Lecas, 2000). On the other hand, they tend to make more gestures than adults even as their speech develops (e.g. Alibali, Evans, Hostetter, Ryan, & Mainela-Arnold, 2009; Chu & Kita, 2008; Colletta et al., 2015). We therefore investigated children of around the age of 10 years.

At the start of our investigation, we did not know whether 10-year-old children could formulate algorithms, whether their gestures would reflect mental simulations, or whether their gestures would help them to formulate more accurate algorithms. Thus, our research focused on these questions. In what follows, we outline the model theory of mental simulation and its computer implementation, and its predictions about making different sorts of rearrangement and its different predictions for abducing algorithms to make them. We then describe three experiments designed to test these predictions. Experiment 1 established that 10 year olds were able to make re-arrangements and that the minimal number of moves in making them predicted their difficulty. Experiment 2 presented 10 year olds with the different task of creating algorithms for different rearrangements. The difficulty of this task depended, not on the number of moves in a rearrangement, but on the complexity of the algorithm. Experiment 3 corroborated this result, and showed that 10 year olds were more accurate in formulating algorithms when they could gesture than when they could not gesture. Finally, we discuss the implications of the results.

The model theory

Mental simulations

The idea of a mental simulation depends on three assumptions formulated in the model theory (e.g.

Johnson-Laird & Khemlani, 2013; Johnson-Laird, Khemlani, & Goodwin, 2015). First, mental models are *iconic*, that is, the structure of a model corresponds to the structure of what it represents insofar as possible (see e.g. Johnson-Laird, 2006; Peirce, 1931–1958, Vol. 4). For example, the spatial relations among objects in a situation are mirrored in the spatial relations among them in their model. Second, mental models underlie visual images, but they are more parsimonious (Shepard & Metzler, 1971), because they do not represent a situation from a particular point of view or represent all, or any, of its visual features (Hegarty, Stieff, & Dixon, 2013). Models can represent what is common to many possibilities, which differ in their details, and so they yield faster inferences than images (Knauff, Fangmeier, Ruff, & Johnson-Laird, 2003). Third, a simulation depends on a kinematic sequence in a mental model that changes over time. The sequence itself represents the temporal order of events (Johnson-Laird, 1983, Chapter 15; Khemlani et al., 2013).

The capacity of working memory constrains simulations, which cannot animate all the parts of a moving machine at the same time—notwithstanding anecdotal reports to the contrary, for example, Nikola Tesla was said to be able to envisage running a dynamo in order to see which bearings would wear out first. However, as Hegarty (1992, 2004) has shown, naive individuals cannot simulate all the actions of a system of pulleys at the same time. Instead, they propagate the effects of one pulley in the system to the next, and so on, in a piecemeal simulation. Fortunately, in the mental simulations underlying the rearrangements in our tasks, individuals need to envisage only one move at a time.

The mAbducer *program*

The model theory explains how people make rearrangements, how they abduce algorithms to make them, and how they deduce the consequences of such algorithms. We have implemented its account of these three processes in a computer program, *mAbducer*. The program is written in Common Lisp, and its source code is available at: http://mentalmodels.princeton.edu/models/.

In order to make a rearrangement, *mAbducer* uses a single means-ends algorithm in the form of a loop:

(1) If no cars remain to be moved in order to achieve the goal, then the rearrangement is done.
(2) If there are more cars that match the goal at the head of the train on the left track than on the siding, then move the matching cars to the right track.
(3) Otherwise if the left track has the next car(s) in the goal but there are cars blocking their move, then move the blocking cars to the siding.
(4) Otherwise for the next car(s) in the goal, move them and any cars in front of them from the siding to the left track.

Unlike the algorithm outlined in the Introduction, this one makes the rearrangements in our experiments in a minimal number of moves. Two statistics should predict the difficulty of making a rearrangement: the minimal number of moves in the rearrangement, and the minimal number of cars to be moved. A reversal of 6 cars calls for 12 moves, and all but 1 of the cars moves more than once, yielding a minimal total of 16 cars in moves. These two variables had effects in predicting adults' difficulty in making rearrangements, both for the actual numbers of moves they made and the times they took, but the two variables did not interact (Khemlani et al., 2013).

Our experiments examined four sorts of rearrangement, which Table 1 presents with their minimal numbers of moves and minimal numbers of cars to be moved. Appendix 1 summarises the derivations of these numbers. The two measures yield the following predicted trend in increasing difficulty for making rearrangements:

palindrome < parity sort < reverse parity sort < reversal.

The difficulty of creating an algorithm for rearrangements should depend, not on the numbers of moves and cars to be moved, but on the *complexity* of the sequence of moves in the algorithm. The *mAbducer* program abduces algorithms that make rearrangements: it is an automatic programmer. It abduces programs by simulating

Table 1. The four rearrangements used in the experiments, their minimal number of moves and number of cars to be moved (see the Appendix for derivations).

	Initial state	Goal state	Number of moves	Number of cars to be moved
Palindrome	abccba	aabbcc	6	10
Parity sort	abcdef	acebdf	7	10
Reverse parity sort	acebdf	abcdef	9	12
Reversal	abcdef	fedcba	12	16

the rearrangements, abstracting from these simulations their underlying structure, and then translating this structure into LISP, a formal programming language, and into informal English. The algorithms are general, yielding a correct rearrangement for trains containing any number of cars. We present here *mAbducer*'s informal algorithms verbatim for the four rearrangements in our studies. The program uses the diagrams that we described earlier:

(1) Reversal: abcdef[-] - is to be rearranged as: - [-] fedcba, where "-" denotes an empty part of the track:

Move one less than the cars to the siding.
While there are more than zero cars on the siding,
    move one car to right track,
    move one car to left track.
Move one car to right track.

The sequence for the reversal of a six-car train has four repetitions of a loop of two moves, and so its structure is simple and salient.

(2) Palindrome: abccba[-] - is to be rearranged as: - [-]aabbcc.

Move one less than half the cars to the siding.
While there are more than two cars on left track,
    move two cars to right track,
    move one car to left track.
Move two cars to right track.

The palindrome has only two repetitions of a two-move loop, and one of the moves in the loop is of two cars. So, the algorithm is a little more complex than the one for the reversal.

(3) Parity sort: abcdef[-] - is to be rearranged as: - [-]acebdf.

While there are more than two cars on left track,
    move one car to right track,
    move one car to the siding.
Move one car to right track.
Move one less than half the cars to left track.
Move half the cars to right track.

The parity sort has two repetitions of a two-move loop, but they are followed by a sequence of three moves. Hence, the algorithm is more complex than the one for the palindrome.

(4) Reverse parity sort: acebdf[-] - is to be rearranged as: - [-]abcdef.

Set the number of cars to be dynamically moved, n-of-s, to one less than half the cars.
Set the decrement to one.
While n-of-s is more than zero,
    move one car to right track,
    move n-of-s cars to the siding,
    move one car to right track,
    move n-of-s cars to left track,
    take decrement from n-of-s.
Move two cars to right track.

The algorithm has two repetitions of a loop, but individuals are most unlikely to detect the loop, because it contains four moves. This factor and the use of the variables *n-of-s* and *decrement* imply that this algorithm should be the most complex of the four.

The preceding analysis of complexity is intuitive but sensible. Complexity can be assessed in many other ways, including the number of instructions and the number of their operands in each algorithm. They corroborate the intuitive trend too. It is also borne out by a simple universal metric that has been used in psychological studies (e.g. Chater & Vitányi, 2003; Khemlani et al., 2013): Kolmogorov complexity. This measure assesses the complexity of a string of symbols from the length of its shortest possible description in a reference language, such as Common Lisp. We computed Kolmogorov complexity from the numbers of characters in *mAbducer*'s Common Lisp algorithms, multiplied by the number of bits in a character. This measure predicts the same trend in increasing difficulty as the intuitive account above, which we show here with their Kolmogorov complexities:

reversal (1288) < palindrome (1295) < parity sort (1519) < reverse parity sort (1771).

We refer to the "complexity" of an algorithm, whichever way it is assessed, and it should predict the difficulty of formulating algorithms.

The difference between the trend predictions for making and for programming rearrangements concerns only reversals. They should be the hardest rearrangements to make, but the easiest to program. In our studies, the two tasks differed in several ways. First, the children moved the cars in making rearrangements, whereas they were not allowed to move them in formulating algorithms. Second, the children did not have to describe anything as they made a rearrangement, whereas they had to describe the algorithms that they abduced. A corollary is that making rearrangements should

be much easier than formulating algorithms. There was a marked contrast between the two tasks for adults when they had to formulate algorithms that worked for trains of any length. In such cases, an algorithm must depend on a loop of moves, and *mAbducer* can formulate loops that are carried out *for* a specified number of times, and loops that are carried out *while* a given condition obtains. While-loops are more powerful than for-loops, that is, they can compute functions that for-loops cannot compute (see e.g. Johnson-Laird, 1983, Chapter 1; Rogers, 1967). But, while-loops are a natural by-product of mental simulations, whereas for-loops call for the solution of simultaneous linear equations. Adults used while-loops more often than for-loops in devising informal algorithms (Khemlani et al., 2013). Each algorithm in the present studies was for a train of only six cars, and so children should not invoke explicit loops of operations—adults seldom did for trains with a small number of cars (Khemlani et al., 2013), but, as we now explain, the children's algorithms should contain precursors to loops.

When a repeated sequence of moves is salient, children should tend to abbreviate the description of these moves in what we refer to as a "proto-loop", because such descriptions are precursors to proper loops. Hence, a proto-loop should occur when a child grasps that the same moves can be carried out on new cars, and abbreviates the description of these moves. For example, a child might describe moving car *d* from the siding to the left track and then to the right track, and continue the description, "and then do the same for *e* and then *f*". This description is an abbreviation, because the reference of "do the same" depends on context, and it is a proto-loop because it refers to carrying out the same sequence of moves on each member of set of cars. A further prediction is that the occurrence of proto-loops should be inversely related to the complexity of algorithms. For example, proto-loops should be more likely in an algorithm for a reversal, which repeats a pair of moves four times, than for a reverse parity sort, which has just two repeats of four moves, which should therefore be much harder to detect.

The five predictions

In summary, the model theory makes five main predictions:

(1) The difficulty of making a rearrangement should depend on the minimal numbers of moves and cars to be moved. These two factors yield a trend of increasing difficulty over the four rearrangements in our studies:
palindrome < parity sort < reverse parity sort < reversal.
(2) The difficulty of formulating an algorithm should depend on its complexity. This prediction yields a trend of increasing difficulty over the four sorts of rearrangement:
reversal < palindrome < parity sort < reverse parity sort.
(3) Children's tendency to use proto-loops should correlate inversely with the complexity of the algorithms.
(4) Children should spontaneously use gestures that mimic moves as they try to formulate algorithms, because gestures should reduce the descriptive demands of the task and help them to remember the current positions of the cars on the track.
(5) Gestures should help children to formulate accurate algorithms, which should be more accurate when they can gesture than when they cannot.

We now turn to the experiments designed to test these predictions.

Experiment 1

Our first goal was to determine whether or not 10 year olds were able to make the 4 sorts of rearrangement of 6 cars. If they were not able to do so, it would be pointless to ask them to formulate algorithms for making them. So, the aim of the first experiment was to examine children's ability to make rearrangements and to test whether their performance corroborated the model theory, that is, the difficulty of a rearrangement should yield the predicted trend (1). We assessed the children's difficulty in making rearrangements from the actual numbers of moves that they took to make them, and their overall times in making the correct moves.

Participants

We tested 20 fifth-grade children (age $M = 9;7$ years, ranging from 8;2 to 10;10 years; 10 M and 10 F) from a primary school in Turin, Italy. As in all our experiments, they took part voluntarily, after their parents had given their informed consent.

Design

The experiment manipulated the theoretical difficulty of rearrangements, and the participants acted as their own controls and carried out the four rearrangements shown in Table 1: a palindrome, a parity sort, a reverse parity sort, and a reversal. They differ in difficulty according to the minimal numbers of moves and cars to be moved (as corroborated by *mAbducer*). To assess the children's performance, we measured (from video-recordings) the accuracy of their solutions, the actual numbers of moves in their solutions, and their overall times to make rearrangements. The rearrangements were presented in a different random order to each participant.

Procedure

The children were each tested in a quiet room in the sole presence of the experimenter. The materials were the railway track and six cars of different colours (blue, green, beige, red, violet, and yellow), labelled with different letters (*a*, *b*, *c*, *d*, *e*, and *f*, respectively), and a photograph of the required rearrangement behind the right track (see Figure 1).

The instructions explained the general nature of the task: the children had to use the siding to rearrange the cars so that they ended up on the right track in the order shown in the photograph. The experimenter pointed to each relevant part of the track, and demonstrated the rules, as she read the following key instructions (translated from Italian):

> This is a game on how we can form trains. In order to form a train you can make moves using these tracks: the left track, the right track, and the siding. These are the six cars to use to form trains. At the beginning, the cars are on the left track. The train you construct has to be on the right track. In order to remember this, each time I'll put a photo of the train you have to construct behind the right side of the track. In order to form the trains you have to follow some rules:
> • You can't lift the cars: they have to run on the tracks.
> • You can't move the cars from the right track to the left track. You can move cars from the left track either to the right track or to the siding, and from the siding back to the left track.
> • You have to try to construct a train in the fewest moves. In order to do so, when you have to move more than one car on a track, move them all together. If, for instance, you want to move two cars from the left track to the siding, you should move the two cars together in a single move rather than in two separate moves.

The experimenter then asked the child to do a practice rearrangement in which two black cars had to be moved from left to right track.

On each trial, the experimenter first placed the six cars on the left track, next she put the photo of the goal state behind the right track, and then she invited the child to construct the required train. The experimental session was video-recorded, and transcribed to show the children's moves, using the same diagrams as *mAbducer*'s, and to record their overall times from the presentation of a goal to the end of the child's final move.

Results and discussion

All 20 participants were accurate with the palindrome, parity sort, and reverse parity sort, and 19 of them were accurate with the reversal. The overall mean time to make the rearrangements was 54 s. Table 2 summarises the participants' accuracy, mean number of moves, and the latencies of their correct solutions. The theoretical minimal numbers of moves (and number of cars to be moved) predicted the difficulty of the rearrangements, both the children's actual numbers of moves over the four rearrangements (Page's $L = 581.5$, $z = 6.31$, $p < .00001$), and the times they took to make correct solutions (Page's $L = 544$, $z = 3.41$, $p < .0003$). The predictions concern monotonic trends, and so we used Page's L test to assess the reliability of such an increase in the rank order of the number of moves, and of their latencies, over the four arrangements.

The children seldom succeeded in making minimal rearrangements, especially for the reversal. Nineteen out of the 20 participants made at least 1 redundant move on 3 or more of the rearrangements, that is, more often than chance (Binomial test, $p < .000025$). The redundancies occurred because children made a move they revised at once, or because they perseverated. For example, a child moved a car from the siding to the left track, and then over to the right track, failing to notice that a more parsimonious solution was to move two cars together from the left track to the right track. There was a reliable difference among the participants, not in the accuracy with which they made the rearrangements (Friedman non-parametric analysis of variance, $\chi^2 = 3.0$, $df = 3$, $p > .3$), but in

Table 2. The percentages of accurate rearrangements in Experiment 1, the mean number of moves, and their latencies (in s) to the four sorts of rearrangement.

	The four sorts of rearrangement and their minimal numbers of moves			
	Palindrome(6 moves)	Parity sort(7 moves)	Reverse parity sort (9 moves)	Reversal(12 moves)
Percent accurate rearrangements	100	100	100	95
Mean number of moves	8.2 (SD = 1.8)	9.9 (SD = 1.5)	11.6 (SD = 1.5)	16.3 (SD = 5.2)
Mean latencies (s)	42 (SD = 17)	49 (SD = 18)	55 (SD = 25)	70 (SD = 33)

the times that they took to make them (the fastest child had a mean time of 32 s, and the slowest child had a mean time of 81 s; Friedman non-parametric analysis of variance, $\chi^2 = 17.37$, df = 3, $p < .001$). The ages of the participants expressed in months correlated neither with accuracy (Spearman's $\rho = 0.12$, $p > .61$), nor with mean times (Spearman's $\rho = 0.21$, $p > .35$). Likewise, gender did not have a significant affect either on accuracy (98% for males and 100% for females; Mann–Whitney test, $z = 1.0$, $p > .32$, Cliff's $\delta = 0.03$) or on mean times (51 s for males and 58 s for females; Mann–Whitney test, $z = 0.51$, $p = .61$, Cliff's $\delta = 0.07$). In sum, 10 year olds could make the rearrangements, and their accuracy and latency corroborated the model theory's predicted trend (Prediction 1 above), but they differed in the speed with which they made the rearrangements.

Experiment 2

We devised Experiment 2 to find out whether children were able to abduce informal algorithms for making the four sorts of rearrangement. Adults who knew nothing about programming could formulate accurate algorithms (Khemlani et al., 2013), but no studies had been carried out on children. When they made rearrangements in Experiment 1, the goal in the picture behind the right track governed their choice of moves. They could use a single strategy to make any rearrangement, because they could decompose the goal—the required order of the cars on right track—and rearrange one or more adjacent cars at a time until they were in the order that matched the goal. In contrast, when they had to formulate an algorithm to make a particular rearrangement, the algorithm itself could not rely on the goal. The children could look at the goal, but their task was to formulate an algorithm that describes a sequence of moves that produce the goal as output. Hence, each rearrangement calls for its own algorithm (see the four algorithms described in the section above on the model theory). The formulation of algorithms should therefore be more complicated than making rearrangements. Likewise, the children had to formulate these algorithms without being able to move the actual cars, and so abduction required them to keep in mind the current positions of the cars.

The experiment tested three predictions. First, the complexity of the algorithms, as opposed to their minimal numbers of moves and cars to be moved, should predict the following trend of increasing difficulty of abducing them (Prediction 2 in the summary of predictions above):

> reversal < palindrome < parity sort < reverse parity sort.

The crucial difference between complexity and number of moves concerns the reversal. In Experiment 1, it was the hardest rearrangement to make, but here its algorithm should be the easiest to formulate. Second, children should be more likely to use proto-loops, that is, the precursors to full-fledged loops, in algorithms with a simple structure, such as a reversal as opposed to a reverse parity sort (Prediction 3). Third, children should make a spontaneous use of gestures as they try to formulate algorithms (Prediction 4). Their gestures should make visible their mental simulations of moves, because the children were not allowed to make actual moves of cars.

Participants

We tested a new sample of 24 children from the same population as before (age $M = 10;1$ years, ranging from 9;6 to 10;6 years; 13 M and 11 F).

Design

The participants in the experiment had to describe in their own words how to make the four rearrangements of six cars, but they were not allowed to move the cars. The experiment manipulated complexity, which differed over the four algorithms, and which should predict the difficulty of formulating them.

We video-recorded the children's performance, and measured the accuracy of their algorithms and the overall times that they took to formulate them. Each participant carried out the task in one of the 24 possible orders allocated at random.

Procedure

There was an initial "warm up" phase in which the children tackled simple rearrangements of four cars in order to familiarise them with the railway environment. Its procedure was identical to that in Experiment 1, with the goal displayed on a photograph behind the right track, and the children moved the cars to make the rearrangements. For the experiment proper, the experimenter told the children that they had to describe in their own words how to make rearrangements of six cars, that is, to describe the moves needed to yield the train in its required order on the right track. The instructions also made clear that the participants could look at the railway track with the train on the left track, and the photograph of the goal on right track, but they were not allowed to move any of the cars.

The transcriptions of the videos

The entire session was video-recorded, and transcribed to record the children's descriptions of their algorithms, the overall times from the presentation of the goal to the end of each description of an algorithm, and the gestures that accompanied their descriptions of the algorithms. Two independent judges coded the video-recordings in order to make explicit each algorithm. The algorithm included all the moves described by the children either through a verbal description, a gesture, or both. Thus, for example, given the following initial arrangement of the cars:

abcdef [] -

if the child pointed to all the cars *b*, *c*, *d*, *e*, and *f* , and drew in the air a trajectory from the left track to the siding while saying "I move these here", the judges transcribed the move as:

a[bcdef] -

The two judges also identified the occurrence of proto-loops. Thus, for example, given the following arrangement of the cars:

- [bcdef]a

if the child said "I move *b* from the siding to the left track then to the right track, and the same with *c*, *d*, *e*, and *f*", the judges coded the description as a proto-loop. Table 3 is a translation from the Italian of a child's protocol for a reversal. The table also presents the transcription of the protocol into notation, and includes the identification of a proto-loop. Examples of transcriptions for the other sorts of rearrangement are in Appendix 2. The two judges agreed in their coding of the algorithms on 84% of trials (Cohen's $\kappa = 0.81$, $p < .0001$); and a third

Table 3. The translation from Italian of a participant's (P 4) description in Experiment 2 of how to make a reversal, its transcription using *mAbducer*'s diagrams, and the identification of a proto-loop.

Task: abcdef has to be rearranged as: fedcba.
Child: "I'd make move these (*points from b to f*) to the siding (*draws in the air a trajectory from the left track to the right track*)." Transcription: abcdef [] a[bcdef] Child: "Then I make *a* run and go there (*draws in the air a trajectory from a to the right track*)." Transcription: [bcdef]a Child: "Then I take off *b* (*points to b and makes a little clock-wise arc in the air*)." Transcription: b[cdef]a Child: "and I put it there (*draws in the air a trajectory from b to the right track*)." Transcription: [cdef]ba Child: "Then the *c* (*points to c and makes a little clock-wise arc in the air*)." Transcription: c[def]ba Child: "and I put it there (*draws in the air a trajectory from c to the right track*)." Transcription: [def]cba Child: "Then the *d* (*points to d and makes a little clock-wise arc in the air*)." Transcription: d[ef]cba Child: "and I put it there" (*draws in the air a trajectory from d to the right track*). Transcription: [ef]dcba Child: "Then the *e* (*points to e, makes a little clock-wise arc in the air, and then draws in the air a trajectory from e to the right track*)." The assertion is a proto-loop, because the utterance refers to a repetition of two moves, and cannot be interpreted correctly out of context: Transcription: e[f]dcba [f]edcba Child: "Then the *f* (*points to f and makes a little clock-wise arc in the air then draws in the air a trajectory from f to the right track*)." The proto-loop continues. Transcription: f[]edcba []fedcba

independent judge, the first author, resolved the discrepancies prior to the statistical analyses. The two judges agreed on 88% of trials about the occurrence of a proto-loop in an algorithm (Cohen's $\kappa = 0.73$, $p < .0001$). The third independent judge resolved the discrepancies between the two judges.

In some algorithms, the children described an illegal move in which a car jumped over an intervening one, as in the move from:

a[ce]bdf

to:

ae[c]bdf

Any algorithms describing such moves or for which the final train on the right track did not match the goal were scored as inaccurate.

Results and discussion

We excluded from analysis the data from three participants because more than half of their descriptions included illegal moves or unclear descriptions of moves. Table 4 summarises the results from the remaining 21 participants: the number of correct algorithms, the number of proto-loops they contained, and the overall times that the participants took to describe their correct algorithms.

Despite the slight anomaly of the palindrome—perhaps because it contains only three sorts of car (abc), theoretical complexity predicted the trend in accuracy of the children's algorithms (Page's $L = 502$, $z = 4.24$, $p < .00003$), and in their latencies (Page's $L = 228$, $z = 13.22$, $p < .000001$). In contrast, the minimal number of moves in making each rearrangement (see Table 1) did not predict a reliable trend in accuracy (Page's $L = 462$, $z = 0.98$, $p > .16$). The use of proto-loops occurred almost always on reversals (Prediction 3), and this bias reflected the simple nature of its loop of operations. The participants differed one from another in the accuracy of their algorithms (Friedman non-parametric analysis of variance, $\chi^2 = 30.75$, $df = 3$, $p < .0001$), and once again this difference correlated neither with age in months (Spearman $\rho = -0.09$, $p > .69$) nor with gender (63% for males and 50% for females; Mann–Whitney test, $z = 1.55$, $p > .1$, Cliff's $\delta = 0.38$).

The children accompanied 99% of their descriptions of moves with gestures. The exceptions were due to a child who described four moves without an accompanying gesture. This massive use of spontaneous gestures was striking. Children used deictic gestures to point at cars, and iconic gestures through the air to indicate the movement of a car from its current location to one that they either described or left unsaid. Most of these gestures were movements of a hand, but some were movements of the head, for example, turning it in the direction described in the algorithm. As the protocol in Table 3 illustrates, gestures often characterised a movement without the child describing its starting position or its destination. We had not anticipated the extent to which the children gestured, and so the video-recordings were not suitable for a fine-grained analysis—a problem that we remedied in the next experiment.

In sum, the experiment corroborated the model theory's predictions: the theoretical complexity of the algorithms determined the children's difficulty in formulating them (Prediction 2) and inversely their use of proto-loops (Prediction 3), and their gestures showed the moves that they were imagining and describing (Prediction 4).

Experiment 3

The ubiquity of gestures suggests that they are a consequence of children's kinematic simulations. They should allow children to finesse the description of cars or their moves (see Table 3). They should also help children to keep in mind what cars are where on the railway track. Hence, if children were unable to gesture, then they should be more likely

Table 4. The total numbers of correct algorithms in Experiment 2 ($N = 21$) depending on their predicted complexity, the total number of proto-loops they contained, and the mean times (s) that the participants took to formulate the correct algorithms.

	The four rearrangements in the predicted order of their increasing complexity			
	Reversal	Palindrome	Parity sort	Reverse parity sort
Total numbers of correct algorithms ($N = 21$)	17	19	9	3
Total number of correct algorithms with a proto-loop ($N = 21$)	8	1	3	1
Mean times (s) to formulate correct algorithms	28.1 (SD = 11.5)	26.8 (SD = 8.5)	43.8 (SD = 21.0)	43.7 (SD = 21.1)

to err in abducing algorithms (Prediction 5). Our final experiment tested this prediction. The participants formulated algorithms to make the four rearrangements in two conditions, once when they could gesture, and once when they could not gesture because their arms were crossed inside a cloth muff.

Participants

We tested a new sample of 34 children from the same population as before (age $M = 10;5$ years, ranging from 9;10 to 11;5 years; 13 M and 21 F).

Design

The participants acted as their own controls for two experimental manipulations: the four different rearrangements, and whether or not they could make gestures. Hence, they formulated algorithms for the four rearrangements twice in two different blocks of trials: in one block they were allowed to gesture (gesture condition), and in one block they were prevented from gesturing (no-gesture condition). The order of the two blocks was counterbalanced so that half the participants began with the gesture condition, and the other half began with the no-gesture condition. The trains had different labels on the cars in the two blocks: in one block the labels were numbers and in one block they were letters, and the occurrence of the two sorts of labels in the gesture and no-gesture conditions was counterbalanced. Within each block, the four rearrangements were presented in a random order to each child. We video-recorded the experiment, and recorded the accuracy of the children's algorithms, and their mean latencies for correct algorithms. We also analysed their gestures in the condition that allowed them to gesture.

Procedure

Each participant was tested in a single session in a quiet room with the sole presence of the experimenter. The participants carried out a "warm up" in which they made rearrangements in order to familiarise themselves with the railway environment. In these problems, the initial state was: abcd[-] -, and the train had to be rearranged in four different orders:

-[-]acdb; -[-]bcda; -[-]adcb; and -[-]cdba.

In the experiment proper, the children were presented with an arrangement of cars on the left track, and a photo of the required goal state behind the right track. The experimenter told them that they had to explain in their own words how they would make the rearrangement, that is, to describe the moves needed to yield the train in its required order on the right track. They were not allowed to move the actual cars. They were allowed to gesture in the gesture condition, though the experimenter did not prompt them to do so. The no-gesture condition was the same except that the children were prevented from gesturing because they had to wear a soft cloth muff with their arms crossed inside it. The experimenter introduced the no-gesture condition with the words: "Before you start, please wear this muff to keep your arms crossed", and she showed the child how to wear the muff. Likewise, the experimenter told the participants when the gesture condition was second: "Now, you'll do the same task without the muff", and the child took off the muff. We video-recorded the experimental sessions, and transcribed them.

The transcriptions of the videos

Two independent judges made transcriptions of the moves described by the children through both speech and gestures, following the same coding procedures used in Experiment 2. Examples of transcriptions for all four problems are in Appendix 3. The judges agreed in their coding of the accuracy of the algorithms on 95% of trials (Cohen's $\kappa = 0.925$, $p < .0001$), they agreed about the occurrence or non-occurrence of proto-loops in the algorithms on 93% of trials (Cohen's $\kappa = 0.73$, $p < .0001$). A second pair of judges viewed the videos of the children's gestures in the gesture condition 3 times. First, they judged whether or not the children's descriptions of moves were accompanied by a gesture, and whether or not it corresponded to the description. They agreed on 89% of instances (Cohen's $\kappa = 0.73$, $p < .0001$). Second, they judged whether each child's gestures accompanying the description of a move were deictic, iconic, or both, where any movement of the hands, the head, or the shoulder, counted as a gesture. They agreed about the occurrence of deictic gestures on 94% of instances (Cohen's $\kappa = 0.81$, $p < .0001$) and about iconic gestures on 98% of instances (Cohen's $\kappa = 0.92$, $p < .0001$). Third, they judged whether deictic gestures accompanied full descriptions of their

referent (e.g. "I move *b*") or not (e.g. "I move this"). Likewise, they judged whether iconic gestures accompanied descriptions of the starting point of a move, its end point, both points, or neither of them. For all of these evaluations, a third independent judge (the first author) resolved the discrepancies prior to analyses. Finally, two different judges counted the ratios of deictic gestures and iconic gestures to the number of moves that the children described in their correct algorithms. They then reached agreement about any discrepancies in their judgments for deictic gestures (Kendall's tau = 0.8) and for iconic gestures (Kendall's tau = 0.71, $p < .0001$ in both cases).

Results and discussion

We analysed the results for 32 participants, excluding those for two participants whose descriptions of algorithms were too unclear to categorise as correct or incorrect. Table 5 summarises the percentages of correct algorithms, and their mean latencies (in s) in the gesture and no-gesture conditions. The children formulated correct algorithms more often when they could gesture (65%) than when they could not (52%; Wilcoxon test, $z = 2.37$, $p < .01$, Cliff's $\delta = 0.23$). Likewise, the mean times that the children took to create correct algorithms were faster when they could gesture (40 s) than when they could not (44 s; Wilcoxon test, $z = 2.37$, $p = .02$, Cliff's $\delta = 0.17$). The effects of gesturing therefore corroborated the model theory's prediction (5) that gestures should improve performance. Overall, it did not differ reliably between the two groups used to control the order of blocks in either accuracy (Mann–Whitney test, $z = 0.96$, $p = .34$, Cliff's $\delta = 0.19$) or in response times (Mann–Whitney test, $z = 1.06$, $p = .29$, Cliff's $\delta = 0.23$). But, there were reliable effects of the order of the two blocks: the children were more accurate (Mann–Whitney test, $z = 2.35$, $p < .02$, Cliff's $\delta = 0.46$) and faster (Mann–Whitney, $z = 2.46$, $p < .015$, Cliff's $\delta = 0.55$) in the no gesture condition if they had already carried out the gesture condition than if they had not (see Table 5). The advantage of gestures carried over to a small but reliable degree to the no gesture condition.

Table 5. The overall percentages of correct algorithms in Experiment 3 ($N = 32$) and the mean times (in s) to produce them in the gesture and no gesture conditions, depending on the order of these two blocks of trials.

	Gestures in first block		Gestures in second block	
	Gestures	No gestures	Gestures	No gestures
Percent correct algorithms	67	56	63	48
Mean times (s) to formulate correct algorithms	40.7 (SD = 13.2)	39.5 (SD = 14.7)	37.2 (SD = 13.4)	48.1 (SD = 16.8)

Table 6 presents the percentages of correct algorithms for the four sorts of rearrangement both in the gesture and in the no gesture conditions, the numbers of proto-loops they contained, and the mean times for their accurate formulations. Despite the unexpected good performance on the reverse parity sort, the theoretical complexity of the algorithms predicted the rank-order decline in their accuracy over the four sorts of rearrangement (Page's $L = 785.5$, $z = 3.89$, $p < .0001$) and the increasing trend in the overall times to formulate correct algorithms (Page's $L = 199.5$, $z = 3.21$, $p < .001$), though this trend was marginal in the no-gesture condition ($L = 135$, $z = 1.55$, $p = .06$). These trends corroborated the previous experiment and the model theory's prediction (2). But, as the theory predicts, the minimal numbers of moves required to make the rearrangements did not account for the difficulty of formulating accurate algorithms for them (Page's $L = 711.5$, $z = 0.868$, $p > .15$). Proto-loops occurred in 38 of the 256 algorithms (18%), equally distributed between the gesture and

Table 6. The percentages of correct algorithms in Experiment 3 ($N = 32$) for each of the four sorts of rearrangement, the total numbers of proto-loops in them, and the mean times (s) that the participants took to formulate correct algorithms.

	The four rearrangements in the order of their theoretical complexity			
	Reversal	Palindrome	Parity sort	Reverse parity sort
Percent correct: Gestures	84	78	38	59
Percent correct: No gestures	78	66	31	34
Number of proto-loops: Gestures	18	0	1	0
Number of proto-loops: No gestures	17	2	0	0
Mean times (s): Gestures	31.5 (SD = 13.7)	36.4 (SD = 17.5)	50.9 (SD = 19.5)	51.9 (SD = 17.9)
Mean times (s): No gestures	36.4 (SD = 11.7)	47.8 (SD = 25.6)	46.8 (SD = 16.0)	48.4 (SD = 14.3)

no-gesture conditions. But, as in the previous experiment, they tended to occur in reversals (see Table 6), the algorithm with the lowest complexity (Prediction 3).

On 74% of their descriptions of moves, the 32 children made gestures, which always corresponded to the move, but on 26% of the descriptions of moves, they made no gestures.

Six children failed to describe any correct algorithms, and of the remaining children, two made no gestures whatsoever. The remaining 24 children all made more gestures corresponding to a move than not (Binomial $p = .5^{24}$). For these 24 children, Table 7 presents the percentages of descriptions of moves that the children accompanied with gestures (in the gesture condition), and the mean ratios of number of gestures to number of described moves for deictic gestures, iconic gestures, and both sorts of gesture overall. Such a ratio is necessary because the numbers of moves to be described differs over the four sorts of problem. Although the trend in the percentages of gestures increases with the theoretical complexity of the algorithms, it is not possible to test its reliability because only four participants formulated accurate algorithms for all four problems in the gesture condition. However, one notable observation pertaining to these data is the proportions of algorithms in which children accompanied every one of their descriptions of a move with a gesture (69%) as opposed to accompanying only some of their descriptions of a move with a gesture (31%). Of these 24 children, most of them (16) described more algorithms with gestures for each move than not (7 children), and there was one tie (Binomial $p < .05$). The salient exceptions were descriptions containing proto-loops, where the children glossed over descriptions of moves of specific cars (only 51% of algorithms for reversal were described with gestures accompanying every move).

Table 7. The percentages of descriptions of moves in Experiment 3 ($N = 24$) for the four sorts of problem that the children accompanied with gestures (in the gesture condition), and the mean ratios of number of gestures to number of described moves for deictic gestures, iconic gestures, and overall gestures.

		Reversal	Palindrome	Parity sort	Reverse parity sort
Percentages of moves accompanied with gestures		64	85	81	95
Mean ratios of gestures to moves	Deictic	0.32	0.42	0.41	0.41
	Iconic	0.49	0.73	0.58	0.61
	Overall	0.78	1.15	0.99	1.02

The children in the present experiment accompanied 74% of their descriptions of moves with gestures, whereas those in the previous experiment accompanied 99% of descriptions of moves with gestures. The reason could be that preventing children from gesturing in the first block had a residual effect on the number of moves that they accompanied with gestures in the second block. It was only 60% whereas when the gesture condition occurred first, it was 85%. The difference, however, was not significant (Mann–Whitney test, $z = 1.14$, $p = .255$, Cliff's $\delta = 0.17$).

To examine the relation between gestures and descriptions of moves, we were forced to drop a further participant from the analysis because this child's descriptions were often unclear. The remaining 23 children accompanied their deictic gestures with complete description of the referents (85%) more often than not (15%, Wilcoxon test, $z = 3.87$, $p < .0001$, Cliff's $\delta = 0.92$), but they accompanied their iconic gestures with no verbal descriptions of the moves (74%) more often than they accompanied them with an incomplete verbal description of the move (21%: $z = 2.81$, $p < .005$, Cliff's $\delta = 0.61$), or a complete verbal description of the move (5%: $z = 3.52$, $p < .0001$, Cliff's $\delta = 0.76$).

The pattern of individual differences was much the same as before. The participants differed in their ability to formulate accurate algorithms (Friedman non-parametric analysis of variance: $\chi^2 = 28.319$, df = 3, $p < .0001$). The only discrepancy with previous studies was that the males (a mean of 5.5 accurate algorithms out of 8) were better than the females (a mean of 4.1 accurate algorithms; Mann–Whitney test, $z = 2.058$, $p < .05$, Cliff's $\delta = 0.43$), and the effect was much the same whether or not the participants were able to gesture. The children differed reliably neither in using full descriptions with deictic gestures (Friedman non-parametric analysis of variance, $\chi^2 = 3$, df = 3, $p > .38$) nor in failing to accompany iconic gestures with verbal descriptions of the corresponding moves (Friedman non-parametric analysis of variance, $\chi^2 = 1$, df = 3, $p > .80$).

General discussion

The two most striking findings of our investigation are that children as young as 10 can abduce algorithms to make rearrangements, and that their

gestures help them to formulate accurate algorithms. The task presupposes the ability to make rearrangements, and 10 year olds are able to do so. For instance, given a train of six cars on the left track, *abcdef*, and the goal of rearranging them into the order *acebdf* on the right track, that is, a parity sort, all the children in Experiment 1 made the rearrangement. Our computer program *mAbducer* represents the starting state of the rearrangement as follows:

abcdef[-] -

where the left side of the string represents the order of the cars on the left track, the square brackets represent the contents of the siding, "-" denotes an empty part of the track, and the right side of the string represents the right track. The program uses a means-ends analysis to make the rearrangement in a minimal number of 7 moves of 10 cars:

abcde[-]f abcd[e]f abc[e]df ab[ce]df a[ce]bdface[-]
bdf[-]acebdf

The program's means-ends analysis uses successive parts of the ends, that is, it solves for one or more cars in the goal at a time (for *mAbducer*'s algorithm for making rearrangements, see the earlier section on the model theory). The children were able to use a similar procedure for making rearrangements, and the program's minimal numbers of moves predicted the children's difficulty with the task, as measured in the actual numbers of moves that they took—they were seldom parsimonious—and in the overall times that they took (Prediction 1 in our earlier summary of the predictions). Hence, as Experiment 1 showed, a palindrome was easier than a parity sort, which was easier than a reverse parity sort, and a reversal of the train was hardest of all. The result replicates the same trend in difficulty for adults making these four rearrangements (Khemlani et al., 2013).

A subtle difference exists between a rearrangement and an algorithm for a rearrangement. When individuals rearrange cars, their moves are constrained by the goal—the required order shown in the picture behind the right track. When they formulate an algorithm to make a particular rearrangement, such as a reversal of the order of the cars in a train, they use the goal to simulate the moves. But, the resulting algorithm itself cannot be under the control of the goal. Programmers can see the goal, but the program that they are constructing cannot. Hence, for example, an algorithm to reverse the order of cars in a train can be described in these terms:

(1) Move all but one car on the left track to the siding.
(2) While there are cars on the siding, move one car from the left track to the right track, and move one car from the siding to the left track.
(3) Otherwise, move one car from the left track to the right track.

When this algorithm is applied to the train: abcdef[-] -, the result is the reversal: - [-]fedcba. The algorithm therefore does not use the goal to make the rearrangement; in effect, it constructs the goal.

According to the model theory, the abduction of an algorithm calls for a kinematic simulation of a sequence of moves that the cars have to make, and for its conversion into a description. Experiments 2 and 3 are pertinent to the hypothesis, because the children were not allowed to move the cars. The difficulty of formulating an algorithm should depend, not on the number of moves that occur in executing the algorithm, but rather on the complexity of the algorithm itself—a factor that has no evident effect on making rearrangements. Complexity can be assessed in many ways. But, both an intuitive analysis of the structure of the four algorithms for our rearrangements (see the section on the model theory) and their Kolmogorov complexity yield the same predicted trend. As a consequence, the reversal of a train, which is the hardest rearrangement to make (Experiment 1), is the easiest algorithm to formulate because its structure is so transparent (Experiment 2). It repeats a loop of two moves 4 times. The palindrome has a structure that should be less easy to program, because its loop of two moves is repeated only twice. The parity sort, which calls for a train to be rearranged so that all cars in odd-numbered positions precede those in even-numbered positions, should be still harder to program, because after such a loop it adds a sequence of three moves. And the reverse parity sort, which is the converse of the parity sort, should be the most difficult of the four, because its structure is so complicated. At first, it appears not to contain a loop at all, but in fact it does have one, albeit one of four moves.

Experiment 2 showed that 10 year olds were able to formulate algorithms. They could do so even though they were not allowed to move the cars. They had to take the goal into account in

formulating an algorithm, but the algorithm they created did not refer to the goal, but aimed instead to produce the required rearrangement. The accuracy of the children's algorithms followed predicted complexity (Prediction 2), and their use of proto-loops—abbreviations of moves in a way that is a precursor to proper loops—also corroborated the theoretical complexity of the algorithms (Prediction 3). Almost all the proto-loops occurred in formulations of the algorithm with the least complex structure, that is, the reversal. To abduce algorithms, the children had to make mental simulations of sequences of moves, which became overt in their many gestures—deictic ones that picked out particular cars and iconic ones that mimicked the moves of one or more cars (Prediction 4). The gestures were indeed an outward sign of inward kinematic simulations. In sum, the theoretical complexity of an algorithm predicted both the accuracy and the time it took to formulate in both Experiments 2 and 3.

Most, but not all, of the children made gestures in formulating algorithms, and they did so in describing about three-quarters of the moves in their algorithms in the gesture condition of Experiment 3. Every single gesture corresponded to a move in their descriptions of algorithms. Gestures that accompany speech appear to serve several purposes, and there are three main views about them (see the Introduction). First, gestures could be epiphenomena that play no causal role in cognition, but that spill over from thinking into acting, perhaps because individuals encounter difficulties during speech production (see e.g. Kita & Ozyurek, 2003; Melinger & Kita, 2007). Second, they could be designed, perhaps in an unconscious way, to help listeners to understand what the speaker is saying (e.g. Hostetter, 2011). Third, gestures could be aids to thinking and to mental simulations (e.g. Goldin-Meadow et al., 2001; Hostetter & Alibali, 2010; Wagner et al., 2004). Within our theory, gestures reflect mental simulations but they also have causal effects. In the case of 10 year olds, who have limited cognitive resources, gestures should aid them in formulating algorithms. They should help, because they reduce the demand to describe actions, and because they aid children in keeping track of what cars are where in the current juncture of the simulation (see Marghetis & Benjamin, 2014).

Experiment 3 bore out the causal role of gestures in mental simulations. When children were unable to gesture, their ability to formulate accurate algorithms fell by 13% in accuracy (Prediction 5). So, children can create algorithms without using gestures, but in this case the task is more demanding, and so they have a greater tendency to err. A further result was that their gestures speeded children up, so they can gesture without taxing mental resources, again a corroboration of their use of mental simulations in order to formulate algorithms. This result contrasts with the findings of Wagner et al. (2004), who observed that when college students were told to inhibit their gestures, it adds to the load on working memory, and impairs their ability to solve math problems. In contrast, the children in our study did not have to concentrate on not making gestures, because the muff that they wore prevented them from gesturing, at least with their hands or arms. They were faster and more accurate in formulating algorithms when they could gesture than when they could not. As the model theory postulates, when simulation controls gestures, they reduce the demands of the task (see also Trafton et al., 2006). It is no longer necessary to describe movements, because iconic gestures communicate them, and likewise the gestures help them to keep track of the positions of cars.

Of course, there are potential alternative explanations. A radical hypothesis is that gestures have no effect on formulating algorithms, but that preventing children from gesturing distracts or inhibits them in some way and so they are less accurate than normal in formulating algorithms. The first proposition in this hypothesis is easy to refute. Children made spontaneous gestures in Experiments 2 and 3, which is unlikely if gestures had no effect on formulating algorithms. Moreover, gestures in Experiment 3 had a manifest effect on their algorithms: iconic gestures tended to replace explicit descriptions of moves. The second proposition in the hypothesis is a little harder to refute. Whenever two conditions yield differences in accuracy, it can be unclear whether one condition is improving performance, or the other condition is impairing performance, or both. One datum, however, is that children were more accurate and faster in the no gesture condition of Experiment 3 if they had already carried out the gesture condition than if they had not. Hence, there was a beneficial effect of gesturing that carried over to the trials in which the children could not gesture. Perhaps the children made covert gestures of a sort that we could not observe on video-recordings. But, however the

effect was mediated, it presupposes that gesturing *was* helpful to the children, contrary to this alternative account.

Another hypothesis is that gestures are more a consequence of the process of description than outward signs of mental simulation. According to the "information packaging hypothesis", gestures underlie the conceptual planning of speech (Alibali, Kita, & Young, 2000). These investigators showed that the gestures of five-year-old children differ depending on whether they are describing differences in quantities in conservation tasks or explaining these differences. They concluded that gestures relate to conceptual plans rather than to the generation of the surface forms of sentences. One interpretation of our results is that gestures reflect still deeper processes of thinking prior to description (see McNeill, 1992, p. 245). This claim is borne out by our finding that when children gesture, they abbreviate their descriptions of moves. They say nothing about a particular move but refer to it with an iconic gesture.

One final alternative hypothesis, which we owe to a reviewer, is that gestures might lead children to adopt a more reflective stance in which they review their own reasoning. It's possible. But, such reflections ought to lead to slower responses, whereas the children were faster to formulate correct algorithms in Experiment 3 when they could gesture. Which almost suggests that the children were more reflective when they could not gesture. Moreover, the children's gestures accompanied their descriptions of the moves in the algorithm, which suggests that the simulations yield both gestures and descriptions at the same time. The children were making both sorts of response in real time rather than reflecting off-line on the correctness of their performance.

The limits of our studies are pertinent to alternative hypotheses. One limit is that our method of preventing gestures in Experiment 3 was obvious to the children. This knowledge in turn may have affected their performance. Hence, it is conceivable that a more natural method of preventing gestures could have a less deleterious effect on performance. Adults appear to gesture less than children in the algorithmic task, but it may be that their eye movements play an analogous role to children's gestures. We are at present investigating this hypothesis. Another limit is that the use of a muff to prevent children from gesturing with their hands did not prevent them from moving their heads, shoulders, or eyes, in ways that were impossible to detect on the video-recordings. Perhaps the prevention of such moves would impair their performance still more.

In all our studies, both the present experiments with children and the earlier experiments with adults (Khemlani et al., 2013), the participants have differed from one another in their ability to formulate algorithms. This difference does not appear to depend on age or on gender. Verbal abilities as well as processing capacity of working memory could be the pertinent factors (cf. Kane, Conway, Hambrick, & Engle, 2007). Computer scientists often complain that no reliable and valid test exists that predicts the likelihood that individuals who know nothing of programming can become competent programmers (see e.g. Bornat, Dehnadi, & Dennis, 2008). Perhaps performance in formulating informal algorithms in the railway domain predicts programming ability.

In conclusion, children are able to make rearrangements and to formulate algorithms for making them. The minimal number of moves for a rearrangement predicts its difficulty for adults and for children. In contrast, the complexity of algorithms predicts the difficulty in abducing them for adults and for children. Children's mental simulations become manifest in their gestures when they are not allowed to move the actual cars on the railway track. These gestures in turn help them to abduce algorithms, most likely because iconic gestures reduce both the demand to describe actions and the load on working memory by helping children to keep in mind where the cars are on the track.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This research was supported by the Italian Ministry of Education University and Research [Grant 2010RP5RNM] (to M. B.) to study problem solving and decision making, by the Polish National Science Centre [Grant 2014/14/M/HS6/00916] (to RM), by a Jerome and Isabella Karle Fellowship from the Naval Research Laboratory (to S.S.K.), and by the National Science Foundation [Grant SES 0844851] (to P.N. J-L.) to study deductive and probabilistic reasoning. We thank Linden Ball, Ruth Byrne, Sam Glucksberg, Adele Goldberg, Geoffrey Goodwin, Louis Lee, David Lobina, Max Lotstein, Paula Rubio, Carlos Santamaría, and three anonymous reviewers for advice.

References

Aamodt-Leeper, G., Creswell, C., McGurk, R., & Skuse, D. H. (2001). Individual differences in cognitive planning on the tower of Hanoi task: Neuropsychological maturity or measurement error? *Journal of Child Psychology & Psychiatry*, *42*, 551–556. doi:10.1111/1469–7610.00749

Albert, D., & Steinberg, L. (2011). Age differences in strategic planning as indexed by the tower of London. *Child Development*, *82*, 1501–1517. doi:10.1111/j.1467-8624.2011.01613.x

Alibali, M. W., Evans, J. L., Hostetter, A. B., Ryan, K., & Mainela-Arnold, E. (2009). Gesture-speech integration in narrative discourse: Are children less redundant than adults? *Gesture*, *9*, 290–311. doi:10.1075/gest.9.3.02ali

Alibali, M. W., & Kita, S. (2010). Gesture highlights perceptually present information for speakers. *Gesture*, *10*, 3–28. doi:10.1075/gest.10.1.02ali

Alibali, M. W., Kita, S., & Young, A. J. (2000). Gesture and the process of speech production: We think, therefore we gesture. *Language and Cognitive Processes*, *15*, 593–613. doi:10.1080/016909600750040571

Alibali, M. W., Spencer, R. C., Knox, L., & Kita, S. (2011). Spontaneous gestures influence strategy choices in problem solving. *Psychological Science*, *22*, 1138–1144. doi:10.1177/0956797611417722

Barrouillet, P., Grosset, N., & Lecas, J. -F. (2000). Conditional reasoning by mental models: Chronometric and developmental evidence. *Cognition*, *75*, 237–266. doi:10.1016/S0010-0277(00)00066-4

Bornat, R., Dehnadi, S., & Dennis, S. (2008). Mental models, consistency and programming aptitude. *Proceedings of the Tenth Australasian Computing Education Conference*, *78*, 53–61. Wollongong: Australian Computer Society. doi:10.1145/1268784.1268845

Chater, N., & Vitányi, P. (2003). Simplicity: A unifying principle in cognitive science? *Trends in Cognitive Sciences*, *7*, 19–22. doi:10.1016/S1364-6613(02)00005-0

Chu, M., & Kita, S. (2008). The nature of gestures' beneficial role in spatial problem solving. *Journal of Experimental Psychology: General*, *140*, 102–116. doi:10.1037/a0021790

Chu, M., & Kita, S. (2011). The nature of gestures' beneficial role in spatial problem solving. *Journal of Experimental Psychology: General*, *140*, 102–116. doi:10.1037/a0021790

Colletta, J. M., Guidetti, M., Capirci, O., Cristilli, C., Demir, O. E., Kunene-Nicolas, R. N., & Levine, S. (2015). Effects of age and language on co-speech gesture production: An investigation of French, American, and Italian children's narratives. *Journal of Child Language*, *42*, 122–145. doi:10.1017/S0305000913000585

Cowan, N. (2005). *Working memory capacity*. New York, NY: Psychology Press.

Cutica, I., Ianì, F., & Bucciarelli, M. (2014). Learning from text benefits from enactment. *Memory & Cognition*, *42*, 1026–1037. doi:10.3758/s13421-014-0417-y

Duncker, K. (1945). On problem solving. *Psychological Monographs*, *58*, 270. doi:10.1037/h0093599

Ehrlich, S. B., Levine, S., & Goldin-Meadow, S. (2006). The importance of gesture in children's spatial reasoning. *Developmental Psychology*, *42*, 1259–1268. doi:10.1037/0012-1649.42.6.1259

Goldin-Meadow, S., Cook, S. W., & Mitchell, Z. A. (2009). Gesturing gives children new ideas about math. *Psychological Science*, *20*, 267–272. doi:10.1111/j.1467-9280.2009.02297.x

Goldin-Meadow, S., Nusbaum, H., Kelly, S. D., & Wagner, S. (2001). Explaining math: Gesturing lightens the load. *Psychological Science*, *12*, 516–522. doi:10.1080%2F01690965.2011.567074

Hegarty, M. (1992). Mental animation: Inferring motion from static diagrams of mechanical systems. *Journal of Experimental Psychology: Learning Memory & Cognition*, *18*, 1084–1102. doi:10.1037//0278-7393.18.5.1084

Hegarty, M. (2004). Mechanical reasoning by mental simulation. *Trends in Cognitive Sciences*, *8*, 280–285. doi:10.1016/j.tics.2004.04.001

Hegarty, M., Mayer, S., Kriz, S., & Keehner, M. (2005). The role of gestures in mental animation. *Spatial Cognition and Computation*, *5*, 333–356. doi:10.1207/s15427633scc0504_3

Hegarty, M., Stieff, M., & Dixon, B. L. (2013). Cognitive change in mental models with experience in the domain of organic chemistry. *Journal of Cognitive Psychology*, *25*, 220–228. doi:10.1080/20445911.2012.725044

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

Hostetter, A. B. (2011). When do gestures communicate? A meta-analysis. *Psychological Bulletin*, *137*, 297–315. doi:10.1037/a0022128

Hostetter, A. B., & Alibali, M. W. (2008). Visible embodiment: Gestures as simulated action. *Psychonomic Bulletin & Review*, *15*, 495–514. doi:10.3758/PBR.15.3.495

Hostetter, A. B., & Alibali, M. W. (2010). Language, gesture, action! A test of the gesture as simulated action frame work. *Journal of Memory & Language*, *63*, 245–257. doi:10.1016/j.jml.2010.04.003

Huizenga, M., Dolan, C. V., & van der Molen, M. W. (2006). Age-related change in executive function: Developmental trends and a latent variable analysis. *Neuropsychologia*, *44*, 2017–2036. doi:10.1080/09297049.2010.509715

Johnson-Laird, P. N. (1983). *Mental models*. Cambridge, MA: Cambridge University Press.

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

Johnson-Laird, P. N., & Khemlani, S. S. (2013). Toward a unified theory of reasoning. In B. Ross (Ed.), *The psychology of learning and motivation* (pp. 1–42, Vol. 59). New York: Elsevier.

Johnson-Laird, P. N., Khemlani, S. S., & Goodwin, G. P. (2015). Logic, probability, and human reasoning. *Trends in Cognitive Sciences*. doi:10.1016/j.tics.2015.02.006

Kane, M. J., Conway, A. R. A., Hambrick, D. Z., & Engle, R. W. (2007). Variation in working memory capacity as variation in executive attention and control. In A. R. A.

Conway, C. Jarrold, M. J. Kane, A. Miyake, & J. N. Towse (Eds.), *Variation in working memory* (pp. 21–48). New York: Oxford University Press.

Keen, R. (2011). The development of problem solving in young children: A critical cognitive skill. *Annual Review of Psychology*, *62*, 1–21. doi:10.1146/annurev.psych.031809.130730

Khemlani, S. S., Mackiewicz, R., Bucciarelli, M., & Johnson-Laird, P. N. (2013). Kinematic mental simulations in abduction and deduction. *Proceedings of the National Academy of Sciences of the United States of America*, *110*, 16766–16771. doi:10.1073/pnas.1316275110

Kita, S., & Ozyurek, A. (2003). What does cross-linguistic variation in semantic coordination of speech and gesture reveal? Evidence for an interface representation of spatial thinking and speaking. *Journal of Memory and Language*, *48*, 16–32. doi:10.1016/S0749-596X(02)00505-3

Knauff, M., Fangmeier, T., Ruff, C. C., & Johnson-Laird, P. N. (2003). Reasoning, models, and images: Behavioral measures and cortical activity. *Journal of Cognitive Neuroscience*, *15*, 559–573. doi:10.1162/089892903321662949

Kuhn, D. (2013). Reasoning. In P. D. Zelazo (Ed.), *The oxford handbook of developmental psychology* (pp. 744–764). Oxford: Oxford University Press.

Luciana, M., & Nelson, C. A. (1998). The functional emergence of prefrontally guided working memory systems in four- to eight-year-old children. *Neuropsychologia*, *36*, 273–293. doi:10.1037/a0016224

Luna, B., Garver, K., Urban, T. A., Lazar, N. A., & Sweeney, J. A. (2004). Maturation of cognitive processes from late childhood to adulthood. *Child Development*, *75*, 1357–1372. doi:10.1111/j.1467-8624.2004.00745.x

Marghetis, T., & Benjamin, B. (2014). Embodied meaning, inside and out: The coupling of gesture and mental simulation. In C. Müller, A. Cienki, E. Fricke, S. Ladewig, D. McNeill, & S. Teßendor (Eds.), *Body-language-communication. An international handbook on multimodality in human interaction* (pp. 2000–2007). New York: Mouton de Gruyter.

McNeill, D. (1992). *Hand and mind: What gestures reveal about thought*. Chicago, IL: University of Chicago Press.

Melinger, A., & Kita, S. (2007). Conceptualisation load triggers gesture production. *Language & Cognitive Processes*, *22*, 473–500. doi:10.1080/01690960600696916

Miyake, A., Friedman, N. P., Emerson, M. J., Witzki, A. H., Howerter, A., & Wager, T. D. (2000). The unity and diversity of executive functions and their contributions to complex frontal lobe tasks: A latent variable analysis. *Cognitive Psychology*, *41*, 49–100. doi:10.1006/cogp.1999.0734

Morsella, E., & Krauss, R. M. (2004). The role of gestures in spatial working memory and speech. *American Journal of Psychology*, *117*, 411–424. doi:10.2307/4149008

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

Peirce, C. S. (1931–1958). *Collected papers of Charles Sanders Peirce*. C. Hartshorne, P. Weiss, & A. Burks (Eds.). Cambridge, MA: Harvard University Press.

Peirce, C. S. (1955). *Philosophical writings of Peirce*. J. Buchler (Ed.). New York: Dover.

Petkov, G., & Nikolova, P. (2010). Simple pointing to objects may facilitate remembering. In S. Ohlsson & R. Catrambone (Eds.), *Proceedings of the 32nd annual conference of the cognitive science society* (pp. 1904–1909). Austin, TX: Cognitive Science Society.

Rogers, H. (1967). *Theory of recursive functions and effective computability*. New York: McGraw–Hill.

Sauter, M., Uttal, D. H., Alman, A. S., Goldin-Meadow, S., & Levine, S. C. (2012). Learning what children know about space from looking at their hands: The added value of gesture in spatial communication. *Journal of Experimental Child Psychology*, *111*, 587–606. doi:10.1016/j.jecp.2011.11.009

Shallice, T. (1982). Specific impairments of planning. *Philosophical Transactions of the Royal Society of London. B: Biological Sciences*, *298*, 199–209. doi:10.1098/rstb.1982.0082

Shepard, R. N., & Metzler, J. (1971). Mental rotation of three-dimensional objects. *Science*, *171*, 701–703. doi:10.1126/science.171.3972.701

Skoura, X., Vinter, A., & Papaxanthis, C. (2009). Mentally simulated motor actions in children. *Developmental Neuropsychology*, *34*, 356–367. doi:10.1080/87565640902801874

Trafton, J. G., Trickett, S. B., Stitzlein, C. A., Saner, L., Schunn, C. D., & Kirschenbaum, S. S. (2006). The relationship between spatial transformations and iconic gestures. *Spatial Cognition and Computation*, *6*, 1–29. doi:10.1207/s15427633scc0601_1

Unterrainer, J. M., & Owen, A. M. (2006). Planning and problem solving: From neuropsychology to functional neuroimaging. *Journal of Physiology-Paris*, *99*, 308–317. doi:10.1016/j.jphysparis.2006.03.014

Wagner, S. M., Nusbaum, H., & Goldin-Meadow, S. (2004). Probing the mental representation of gesture: Is hand-waving spatial? *Journal of Memory & Language*, *50*, 395–407. doi:10.1016/j.jml.2004.01.002

Welsh, M. C., Pennington, B. F., & Grossier, D. B. (1991). A normative-developmental study of executive function: A window on prefrontal function in children. *Developmental Neuropsychology*, *7*, 131–149. doi:10.1080/87565649109540483

Wesp, R., Hess, J., Keutmann, D., & Wheaton, K. (2001). Gestures maintain spatial imagery. *American Journal of Psychology*, *114*, 591–600. doi:10.2307/1423612

Appendix 1. *mAbducer*'s minimal solutions to the four rearrangements in the experiments

To make rearrangements, *mAbducer* constructs a kinematic sequence of models containing the smallest number of moves. It represents the railway environment as follows:

a[bcdef]

where a is on left track, bcdef are on the siding demarcated by the square brackets, and the right track is empty. (For simplicity, we have dropped the use of "-" to represent an empty part of the track in this appendix.) Trains can be rearranged using three sorts of move:

S: move one or more cars from left track to siding,
R: move one or more cars from left track to right track,
L: move one or more cars from the siding to left track.

The number following each of these abbreviations in the solutions below denotes the number of cars in the move. *mAbducer* yields the following minimal solutions.

(1) Reversal: abcdef[ ] is to be rearranged as: [ ] fedcba.

The program yields the solution:

S 5: a [bcdef]
R 1: [bcdef]a
L 1: b[cdef]a
R 1: [cdef]ba
L 1: c [def]ba
R 1: [def]cba
L 1: d[ef]cba
R 1: [ef]dcba
L 1: e [f]dcba
R 1: [f]edcba
L 1: f [ ]edcba
R 1: [ ]fedcba

The total number of moves is 12, and the number of cars moved is16. The sequence contains four iterations of a loop of two moves: R 1 and L 1.

(2) Palindrome: abccba[ ] to be rearranged as: [ ] aabbcc.

The program yields the solution:

S 2: abcc[ba]
R 2: ab[ba]cc
L 1: abb[a]cc
R 2: a[a]bbcc
L 1: aa[ ]bbcc
R 2: [ ]aabbcc

The number of moves is 6, and the number of cars to be moved is 10. The sequence has an initial move, two iterations of a loop of two moves, and then a final move.

(3) Parity sort: abcdef[ ] is to be rearranged as: [ ] acebdf.

The program yields the solution:

R 1: abcde[ ]f
S 1: abcd [e]f
R 1: abc[e]df
S 1: ab[ce]df
R 1: a[ce]bdf
L 2: ace[ ]bdf
R 3: [ ]acebdf

The number of moves is 7, and the number of cars to be moved is 10. The sequence has two iterations of a loop of two moves, but then a complex sequence of three moves.

(4) Reverse parity sort: acebdf[ ] is to be rearranged as: [ ]abcdef.

The program yields the solution:

R 1: acebd[ ]f
S 2: ace[bd]f
R 1: ac[bd]ef
L 2: acbd[ ]ef
R 1: acb[ ]def
S 1: ac[b]def
R 1: a[b]cdef
L 1: ab[ ]cdef
R 2: [ ]abcdef

The number of moves is 9, and the number of cars moved is 12. The sequence contains two iterations of a complex loop of four moves.

Appendix 2. The translations from Italian of three descriptions from three participants in Experiment 2 of how to make palindromes, parity sorts and reverse parity sorts and their relative transcriptions using *mAbducer*'s notation

Participant's 10 description of how to make a palindrome.

abccba[] has to be rearranged as: []aabbcc.

Child: "I would put this (*points to a*) there (*draws in the air a trajectory from the left track to the side track*)."
Transcription: abccba[] abccb[a]
Child: "I would put *b* (*points to b*) there (*draws in the air a trajectory from the left track to the side track*)."
Transcription: abbcc[ba]
Child: "I would put these two cars (*points to the two c*) here (*draws in the air a trajectory from the left track to the right track*)."
Transcription: ab[ba]cc
Child: "Then I would move *b* (*points to side track and draws in the air a trajectory from the side track to left track*) and these two are attached (*points to the two b*)."
Transcription: abb[a]cc
Child: "Then, *a* is here (*points to side track*), I move *b* and *b* ahead (*draws in the air a trajectory from the left track to the right track*)."
Transcription: a[a]bbcc

(*Continued*)

Continued.

abccba[] has to be rearranged as: []aabbcc.

Child: "Then I move *a* backwards (*makes a movement with the right hand so to grasp a car on the side track and draws in the air a trajectory from the side track to the left track*)."
Transcription: aa[]bbcc
Child: "Thus these two are equal (*points to one of the a*) and I put them ahead (*draws in the air a trajectory from the left track to the right track*)."
Transcription: []aabbcc

Participant's three description of how to make a parity sort.

abcdef[] has to be rearranged as: []acebdf.

Child: "I'd make move this (*points to f*) there (*draws in the air a trajectory from the left track to the right track*)."
Transcription: abcdef [] abcde[]f
Child: "Then this (*points to e*) here (*draws in the air a trajectory from the left track to the side track*)."
Transcription: abcd[e]f
Child: "This (*points to d*) ahead (*draws in the air a trajectory from the left track to the right track*)."
Transcription: abc[e]df
Child: "I'd move this (*points to c*) here (*draws in the air a trajectory from the left track to the side track*)."
Transcription: ab[ce]df
Child: "This (*points to b*) here (*draws in the air a trajectory from the left track to the right track*)."
Transcription: a[ce]bdf
Child: "I make go back (*draws in the air a trajectory from the side track to the left track*) *c* and *e* (*points to c and e*)."
Transcription: ace[]bdf
Child: "Then I make go ahead *e* (*makes a grasp movement with the right hand as to grasp a car on the left track and draws in the air a trajectory from the left track to the right track*)."
Transcription: ac[]ebdf
Child: "I make go back *c* (*makes a grasp movement with the right hand as to grasp a car on the left track and draws in the air a trajectory from the left track to the right track*)."
Transcription: a[]cebdf
Child: "Then *a* (*draws in the air a trajectory from the left track to the right track*)."
Transcription: []acebdf

Participant's 25 description of how to make a reverse parity sort.

acebdf[] has to be rearranged as: []abcdef.

Child: "I make *f* (*points to f*) go first (*draws in the air a trajectory from the left track to the right track*)."
Transcription: acebdf [] acebd[]f
Child: "Then we take *d* and *b* (*points to d and b*) and put them here (*draws in the air a trajectory from the left track to the side track*)."
Transcription: ace[bd]f
Child: "The *e* (*points to e*) there (*draws in the air a trajectory from the left track to the right track*)."
Transcription: ac[bd]ef
Child: "Then I make *b* (*points to side track*) go back (*draws in the air a trajectory from the side track to the left track*)."
Transcription: acb[d]ef
Child: "We take *d* (*draws in the air a trajectory from the side track to the right track*)."
Transcription: acbd[]ef
Child: "I take *d* (*points to d*) and put it there (*draws in the air a trajectory from the left track to the side track*)."
Transcription: acb[]def

(*Continued*)

Continued.

acebdf[] has to be rearranged as: []abcdef.

Child: "We put *b* (*points to b*) here again (*draws in the air a trajectory from the left track to the side track*)."
Transcription: ac[b]def
Child: "We put *c* (*points to c*) after *d* (*draws in the air a trajectory from the left track to the right track*)."
Transcription: a[b]cdef
Child: "Then we take again the *b* (*makes with the right hand a grasp movement as to take a car*) and put it behind the *c* (*draws in the air a clock-wise trajectory from the side track to the right track*)."
Transcription: ab[]cdef, a []bcdef
Child: "And we put the *a* (*points to a*) behind the *b* (*draws in the air a trajectory from the left track to the right track*)."
Transcription: []abcdef

Appendix 3. The translations from Italian of four descriptions from four participants in Experiment 3 of how to make reversal, palindromes, parity sorts and reverse parity sorts and relative transcriptions using *mAbducer*'s notation, and the identification of proto-loops

Participant's 5 description of how to make a reversal.

abcdef[] has to be rearranged as: []fedcba.

Child: "I move all here (*open the five fingers of the right hand and draws in the air a small trajectory from the left track to the right track*)."
Transcription: abcdef [] [abcdef]
Child: "Then I put *a* here (*draws in the air a trajectory from the side track to the left track*)."
Transcription: a[bcdef]
Child: "then here (*draws in the air a trajectory from the left track to the right track*)."
Transcription: [bcdef]a
Child: "*b* then here (*draws in the air a trajectory from the side track to the left track*)."
Transcription b[cdef]a
Child: "then there (*draws in the air a trajectory from the left track to the right track*)."
Transcription: [cdef]ba
Child: "The *c* (*makes a little clock-wise arc in the air from the side track to the left then to the right track*)."
Transcription: c[def]ba [def]cba
Child: "until the *f* the same". The assertion is a proto-loop, because the utterance refers to a repetition of two moves, and cannot be interpreted correctly out of context:
Transcription: d[ef]cba [ef]dcba e[f]dcba [f]edcba f[]edcba []fedcba

Participant's 11 description of how to make a palindrome.

abccba[] has to be rearranged as: []aabbcc.

Child: "I move a and b (points to a and b) to the side track (draws in the air a trajectory from the left track to the side track)."
Transcription: abccba[] abcc[ba]
Child: "The two c (points to the two c with two fingers) ahead (draws in the air a trajectory from the left track to the right track)."
Transcription: ab[ba]cc

(*Continued*)

Continued.

abccba[] has to be rearranged as: []aabbcc.

Child: "I leave a there (points to the side track) and I move b back (points to the side track and draws in the air a trajectory from the side track to the left track)."
Transcription: abb[a]cc
Child: "and I move the two b in the right track (points to left track and draws in the air a trajectory from the left track to right track)."
Transcription a[a]bbcc
Child: "I take the a on the side track and put it near the a on the left track (draws in the air a trajectory from the side track to the left track)."
Transcription: aa[]bbcc
Child: "Then I move them ahead (draws in the air a trajectory from the left track to the right track)."
Transcription: []aabbcc

Participant's 31 description of how to make a parity sort.

abcdef[] has to be rearranged as: []acebdf.

Child: "I put *f* (*points to f*) ahead (*draws in the air a trajectory from the left track to the right track*)."
Transcription: abcde f[] abcde[]f
Child: "I move *e* on the side track (*points to e and draws in the air a trajectory from the left track to the side track*)."
Transcription: abcd[e]f
Child: "I move *d* ahead (*points to d and draws in the air a trajectory from the left track to the right track*)."
Transcription: abc[e]df
Child: "Then the *c* where is the *e* (*points to left track and draws in the air a trajectory from the left track to side track*)."
Transcription: ab[ce]df
Child: "I move ahead *b* (*points to b and draws in the air a trajectory from the left track to the right track*)."
Transcription: a[ce]bdf
Child: "Then you make go backwards those that were on the side track (*points to the side track and draws in the air a trajectory from the side track to the left track*)."
Transcription: ace[]bdf

(*Continued*)

Continued.

abcdef[] has to be rearranged as: []acebdf.

Child: "You make them go ahead together (*points to the left track and draws in the air a trajectory from the left track to the right track*)."
Transcription: a[]cebdf
Child: "Then you pass the *a* (*draws in the air a trajectory from the left track to the right track*)."
Transcription: []acebdf

Participant's 32 description of how to make a reverse parity sort.

acebdf[] has to be rearranged as: []abcdef.

Child: "I put f (points to f) there (draws in the air a trajectory from the left track to the right track)."
Transcription: acebdf [] acebd[]f
Child: "I move these two (points to b and d) there (draws in the air a trajectory from the left track to the side track)."
Transcription: ace[bd]f
Child: "I send e ahead (points to e and draws in the air a trajectory from the left track to the right track)."
Transcription: ac[bd]ef
Child: "These two (points to b and d) are still here (points to side track) then I … yes I move backwards both (points to the side track and draws in the air a trajectory from the side track to the left track) the b and the d."
Transcription: acbd[]ef
Chil: "and I make move only the d (draws in the air a trajectory from the left track to the right track)."
Transcription: acb[]def
Child: "Then I move again ahead b (points to b and draws in the air a trajectory from the left track to the side track)."
Transcription: ac[b]def
Child: "Then I make move c (points to c and draws in the air a trajectory from the left track to the right track)."
Transcription: a[b]cdef
Child: "Then I make go back again b (points to side track and draws in the air a trajectory from the side track to the left track)."
Transcription: ab[]cdef
Child: "and I make go ahead a and b (points to a and b) directly (draws in the air a trajectory from the left track to the right track)."
Transcription: []abcdef