Representational Change is Integral to Reasoning

PHILOSOPHICAL TRANSACTIONS A

rsta.royalsocietypublishing.org

Research

Article submitted to journal

Subject Areas:
Knowledge evolution, Knowledge representation, Automated reasoning

Keywords:
Automated theory repair, Abduction, Belief revision, Conceptual change, Reformation

Author for correspondence:
Insert corresponding author name
e-mail: A.Bundy@ed.ac.uk

Representational Change is Integral to Reasoning

Alan Bundy[1] and Xue Li[1]

[1]University of Edinburgh

Reasoning is the derivation of new knowledge from old. The reasoner must represent both the old and new knowledge. This representation will change as reasoning proceeds. This change will not just be the addition of the new knowledge. The *representation* of the old knowledge will also often change as a side effect of the reasoning process. For instance, the old knowledge may contain errors, be insufficiently detailed or require new concepts to be introduced.

We start by illustrating our claim with some examples of such representational change. We then describe the Abduction, Belief Revision and Conceptual Change (ABC) theory repair system, which can automate such representational change.

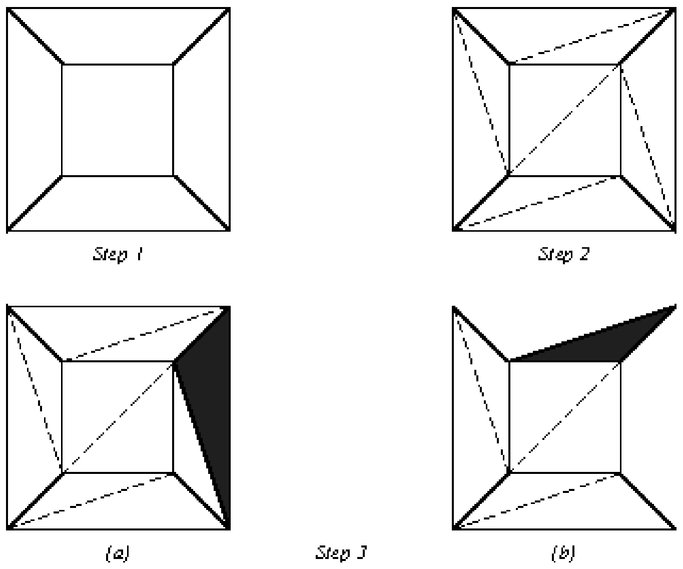
1. Lakatos' Proof and Refutations

In his book [1], Imre Lakatos illustrates the evolution of mathematical methodology via a rational reconstruction of Euler's 'Theorem' that in a polyhedron $V - E + F = 2$, where V is the number of vertices, F the number of faces and E is the number of edges. For instance, in a cube $V = 8$, $F = 6$ and $E = 12$.

The setting is a classroom of incredibly bright students whose teacher leads a Socratic dialogue in which the students echo the positions of various prominent mathematicians during the history of this 'Theorem'.

I have placed scare quotes around 'Theorem' because it rapidly becomes apparent that it has a wide variety of counterexamples. The evolution of mathematical methodology is illustrated by the different ways in which these counterexamples are regarded and the attempts to rescue the 'Theorem' in some form.

THE ROYAL SOCIETY
PUBLISHING



This 'proof' consists of a procedure. In step 1, a face is removed from the polyhedron and it is stretched onto the plane. In step 2, each face is triangulated. In step 3, these triangles are successively removed until only one remains. The last triangle has 3 vertices, 3 edges and 1 face, so $V - E + F = 1$. It is argued that this invariant is preserved by each step, except the first, which removes 1 face. So, working backwards, it has been shown that in the original polyhedron, $V - E + F = 2$. QED.

Figure 1. Cauchy's 'Proof' of Euler's 'Theorem'.

(a) Cauchy's 'Proof' and some Counterexamples to It

Lakatos' book starts with a 'proof', due to Cauchy, illustrated by Figure 1.

We have put scare quotes around 'proof' because Cauchy's 'Proof' is no more a proof than Euler's 'Theorem' is a theorem. It is claimed that this same procedure can be carried out on any polyhedron. But this claim has been shown to be false. Lakatos' students soon come up with counterexamples. Two of these are depicted in Figure 2.

(b) The Changing Definition of Polyhedron

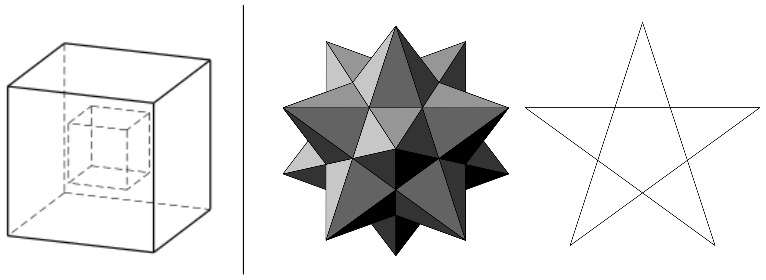
The students, however, are able to rescue these counterexamples and turn them into examples. They did this by choosing an appropriate definition of polyhedron or polygon. In the case of the hollow cube, they contrasted a solid structure with a plate structure.

Solid Structure: A polyhedron is a *solid* whose surface consists of polygonal faces.
Plate Structure: A polyhedron is a *surface* consisting of a system of polygons.

In the case of the *plate* structure, the hollow cube becomes one cube nested within another. For each one $V - E + F = 2$.

In the case of Kepler's Star Polyhedron, they chose an appropriate definition of a polygon.

Intersecting Edges: A polygon is a system of edges arranged in such a way that exactly two edges meet at each vertex.



The hollow cube has a cube-shaped hole in it, so the numbers of vertices, faces and edges is doubled: $V - E + F = 4$. The faces of Keplar's star polyhedron are intersecting pentagrams. There are 12 of these faces, making 30 edges and 12 vertices, so $V - E + F = 6$. Note that, in neither counterexample, is it possible to remove one of their faces and stretch the remaining polyhedron flat on the plane.

Figure 2. The Hollow Cube, Kepler's Star Polyhedron and Pentagram

Non-Intersecting Edges: A polygon is a system of edges arranged in such a way that exactly two edges meet at each vertex *and the edges have no points in common except the vertices.*

The second definition excludes edges that intersect, so the pentagram is ruled out as a polygon. If the faces are triangles then there are 32 of them, with 90 edges and 60 vertices, so $V - E + F = 60 - 90 + 32 = 2$.

Now we are confronted by a conundrum:

How was it possible for Euler to state a conjecture about polyhedra and for Cauchy to claim to have proved that conjecture, if neither had a formal definition of polyhedra?

The answer is that both were genralising from a finite set of examples. They knew about the five Platonic solids of tetrahedron, cube, octahedron, dodecahedron, icosahedron, and a few more. Euler's conjecture and Cauchy's proof both worked for them. There were others, however, for which whether it worked or not depended on definitions that they had not formulated.

Our claim is that this state of affairs is commonplace. Not now in modern mathematics, which imposes a discipline of formal definitions, but in everyday life. It is a frequent cause of computer failures, for instance. Programs are successfully tested on a finite set of test examples then, but later encounter so called, 'edge cases', on which they fail.

Formal verification of computer programs can avoid this problem because it proves the correctness of the program for the, potentially infinite, set of all cases. It requires, though, a formal logical definition of *all cases* and this is still error prone since it is a non-trivial task to formulate such a definition.

This motivated us to attempt to automate the repair of faulty logical theories. This is not just a matter of enlarging or reducing the axioms of a theory. It also entails refining the *language* in which they are expressed. We saw such language refinement in the case of polyhedra in the debate over whether they were solid or plate structured and over whether their edges could intersect. In this spirit, we developed the *ABC Theory Repair System* [2,3].

2. The ABC Theory Repair System

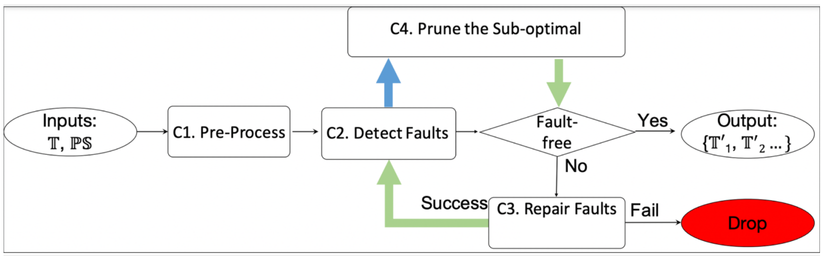
The ABC system takes a faulty theory $\mathbb{T}$ and a preferred structure $\mathbb{PS}$. It uses $\mathbb{PS}$ and inference on $\mathbb{T}$ to identify faults in $\mathbb{T}$. It then applies repair operations to $\mathbb{T}$ in an attempt to correct the faults. If it terminates, it outputs a fault-free theory.

Theories are expressed in the *Datalog* language. See §(a) for details. The preferred structure $\mathbb{PS}$ is a pair $\langle \mathcal{T}(\mathbb{PS}), \mathcal{F}(\mathbb{PS})\rangle$ of sets of ground assertions representing observations of the environment. $\mathcal{T}(\mathbb{PS})$ represents ground assertions that *are* observed and $\mathcal{F}(\mathbb{PS})$ represents ground assertions that *are not* observed. Faults are either *insufficiencies*, where something in $\mathcal{T}(\mathbb{PS})$ is not predicted or *incompatibilities*, where something in $\mathcal{F}(\mathbb{PS})$ is predicted. By predicting a ground assertion we mean that it is a theorem of $\mathbb{T}$. To prove theorems we use Selected Literal Resolution (SL) [4]. See §(b) for details.

The flowchart of the ABC system is depicted in Figure 3. The pre-process C1 reads and rewrites inputs into the internal format for later use. Then in C2, ABC applies SL to $\mathbb{T}$ to detect incompatibility and insufficiency faults.

Our hypothesis is that the ABC System has a diverse range of applications to successfully repairing faulty representations.

We evaluate this claim in §4 by presenting a diverse range of applications to which ABC has been successfully applied.



C4. Prune the Sub-optimal
Inputs: $\mathbb{T}$, $\mathbb{PS}$
C1. Pre-Process
C2. Detect Faults
Fault-free
Yes
No
Output: $\{\mathbb{T}'_1, \mathbb{T}'_2 \ldots\}$
Success
C3. Repair Faults
Fail
Drop

The green arrows deliver a set of theories one by one to the next process; the blue arrow delivers all faults of one theory as a set. When a faulty-theory is not repairable, it will be dropped.

Figure 3. The Flowchart of ABC.

(a) Datalog Theories

Datalog is a logic programming language consisting of Horn clauses in which there are no functions except constants [5]. We use this notation to define a subset of first-order logic that we also call *Datalog*. We choose Datalog because it is decidable under SL and ABC requires[1] decidability in the detection of faults. We represent formulae in Datalog as clauses in Kowalski normal form, shown in Definition 2.1 below.

Definition 2.1 (Datalog Formulae)**.**

Let the language *of a Datalog theory* $\mathbb{T}$ *be a triple* $\langle \mathcal{P}, \mathcal{C}, \mathcal{V}\rangle$*, where* $\mathcal{P}$ *are the propositions,* C *are the constants and* V *are the variables. We will adopt the convention that variables are written in lower case, and constants and predicates start with a capital letter*[2].

[1]This requirement can be relaxed providing one is willing to accept a heuristic ABC. That is, proof attempts can be terminated with failure when some time or space threshold is exceeded. So, a difficult to prove theorem might be wrongly classified as unprovable.
[2]The opposite of the Prolog convention.

A proposition is a formula of the form $P(t_1, \ldots, t_n)$, where $t_j \in \mathcal{C} \cup \mathcal{V}$ for $1 \le j \le n$, i.e., there are no compound terms. Let $R \in \mathcal{P}$ and $Q_i \in \mathcal{P}$ for $0 \le i \le m$ in $\mathbb{T}$. Datalog clauses are of the four types in Definition 2.2, R is called the head *of the clause and the conjunction of the Q_is forms the* body.

Definition 2.2. *Kowalski Form Horn Clauses*

Implication: $(Q_1 \wedge \ldots \wedge Q_m) \implies R$, *where $m > 0$ These usually represent the rules of $\mathbb{T}$.*

Assertion: $\implies R$, *i.e., the body is empty. When R contains no variables the assertion is called* ground. *These ground assertions represent the facts of $\mathbb{T}$ and the members of $\mathcal{T}(\mathbb{PS})$ and $\mathcal{F}(\mathbb{PS})$.*

Goals: $Q_1 \wedge \ldots \wedge Q_m \implies$, *i.e., the head is empty. These usually arise from the negation of the conjecture to be proved and from subsequent subgoals in a derivation.*

Empty Clause: $\implies$, *i.e., both the head and body are empty. This represents false, which is the target of a reductio ad absurdum proof. Deriving it, therefore, represents success in proving a conjecture.*

(b) Selected Literal Resolution

SL is a complete, reductio ad absurdum proof procedure for first order logic expressed in clausal form [4]. When restricted to Datalog clauses, such as those defined in Definition 2.2, SL is a decision procedure. This means that ABC can decide whether a given conjecture is or is not a theorem of a Datalog theory $\mathbb{T}$.

Definition 2.3 (A Deductive Step in SL). *A deductive step in SL is between a goal clause and either an assertion or a rule, which we will collectively call an* axiom. *A proposition in the goal clause and the head of the axiom must unify, that is it must be possible to instantiate each of them so that they are identical. An instantiation σ replaces variables by constants. We write $\phi\sigma$ to mean that proposition ϕ is instantiated by substitution σ. The substitution used in an SL step is called* a most general unifier *[6]. We will depict such an SL step as:*

$$\frac{Q_1 \wedge \ldots \wedge Q_i \wedge \ldots \wedge Q_m \implies}{(Q_1 \wedge \ldots \wedge P_1 \wedge \ldots \wedge P_m \wedge \ldots \wedge Q_m)\sigma \implies} \; P_1 \wedge \ldots \wedge P_m \implies R$$

where σ is the most general unifier of Q_i and R.

Definition 2.4 (An SL Refutation). *Because a proof in SL is by reductio ad absurdum, we call it a* refutation. *It consists of a series of SL steps. Each step takes as input the goal clauses produced by the previous step and outputs the goal clauses to be used in the next step. In the final step, one goal proposition remains and an assertion is unified with it to leave the empty goal clause $\implies$. We can depict such a refutation as follows.*

$$\begin{array}{cl} Goal_1 \implies & \\ \hline Goals_2 \implies & Axiom_1 \\ \hline Goals_3 \implies & Axiom_2 \\ \hline Goal_4 \implies & Axiom_3 \\ \hline \implies & Assertion \end{array}$$

If an SL refutation proves a formula ϕ using the axioms from theory $\mathbb{T}$. We write $\mathbb{T} \vdash \phi$.

(c) Types of Fault

Given a preferred structure $\mathbb{PS}$, a theory $\mathbb{T}$ could have two kinds of faults:

Incompatibility: Predictions that arise from the agent's representation conflict with observations of their environment: $\exists \phi.\ \mathbb{T} \vdash \phi \wedge \phi \in \mathcal{F}(\mathbb{PS})$.

Insufficiency: The agent fails to predict observations of its environment: $\exists \phi.\ \mathbb{T} \nvdash \phi \wedge \phi \in \mathcal{T}(\mathbb{PS})$

Since SL is decidable for Datalog theories [7], ABC can exhaustively test all members of both $\mathcal{T}(\mathbb{PS})$ and $\mathcal{F}(\mathbb{PS})$ for theorem-hood. So it can detect all occurrences of insufficiency and incompatibility in a Datalog theory.

(d) ABC Repair Operations

An insufficiency can be repaired by unblocking a proof with additional necessary SL steps, while an incompatibility can be repaired by blocking all its proofs, which can be done by breaking one SL step in each of them [2]. ABC repairs faulty theories using eleven *repair operations*. There are five for repairing incompatibilities and six for repairing insufficiencies. These are defined in Definitions 2.5 and 2.6.

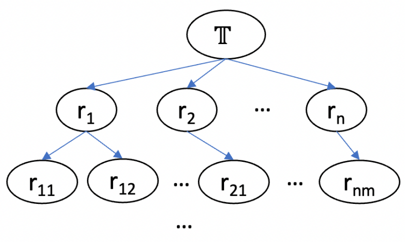
Definition 2.5 (Repair Operations for Incompatibility). *In the case of incompatibility, the unwanted proof can be blocked by causing any of the SL steps to fail. Suppose the targeted SL step is between a goal, $P(s_1, \ldots, s_n)$, and an axiom, Body $\Longrightarrow P(t_1, \ldots, t_n)$, where each s_i and t_i pair can be unified. Possible repair operations are as follows:*

Belief Revision 1: *Delete the targeted axiom: Body $\Longrightarrow P(t_1, \ldots, t_n)$.*
Belief Revision 2: *Add an additional precondition to the body of an earlier rule axiom which will become an unprovable subgoal in the unwanted proof.*
Reformation 1c: *Rename P in the targeted axiom to either a new predicate or a different existing predicate P'.*
Reformation 2c: *Increase the arity of all occurrences P in the axioms by adding a new argument. Ensure that the new arguments in the targeted occurrence of P, are not unifiable. In Datalog, this can only be ensured if they are unequal constants at the point of unification.*
Reformation 3c: *For some i, suppose s_i is C. Since s_i and t_i unify, t_i is either C or a variable. Change t_i to either a new constant or a different existing constant C'.*

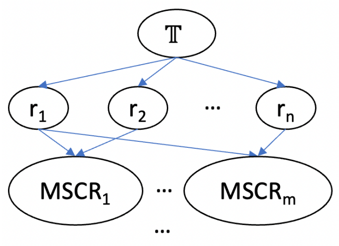
Definition 2.6 (Repair Operations for Insufficiency). *In the case of insufficiency, the wanted but failed proof can be unblocked by causing a currently failing SL step to succeed. Suppose the chosen SL step is between a goal $P(s_1, \ldots, s_m)$ and an axiom Body $\Longrightarrow P'(t_1, \ldots, t_n)$, where either $P \neq P'$ or for some i, s_i and t_i cannot be unified. Possible repair operations are:*

Abduction 1: *Add the goal $P(s_1, \ldots, s_m)$ as a new assertion and replace variables with constants.*
Abduction 2: *Add a new rule whose head unifies with the goal $P(s_1, \ldots, s_m)$ by analogising an existing rule or formalising a precondition based on a theorem whose arguments overlap with the ones of that goal.*
Abduction 1: *Locate the rule axiom whose precondition created this goal and delete this precondition from the rule.*
Reformation 1s: *Replace $P'(t_1, \ldots, t_n)$ in the axiom with $P(s_1, \ldots, s_m)$.*
Reformation 2s: *Suppose s_i and t_i are not unifiable. Decrease the arity of all occurrences P' by 1 by deleting its i^{th} argument.*
Reformation 3s: *If s_i and t_i are not unifiable, then they are unequal constants, say, C and C'. Either (a) rename all occurrences of C' in the axioms to C or (b) replace the offending occurrence of C' in the targeted axiom by a new variable.*

In a faulty theory, there can be multiple faults and each fault can have multiple repairs. The basic search is shown by Figure 4 (a), where repairs are applied individually and the fault detection and repair generation are recursively until the repair process terminates with fault-free theories or finds no further repairs to apply. This basic search is of low efficiency. In the next section, a refinement which improves the search efficiency will be introduced.



(a) The Naive search space.



(b) The reduced search space based on MSCRs.

The length of each search branch can be different. By applying all repairs in one search branch, that branch terminates with a fault-free theory or with failure, if no repair is available to fix a detected fault.

Figure 4. The Search Space for Fault-Free Theories.

(e) The Optimal Maximal Set of Commutative Repairs

Commutative repairs are the ones that can be applied in any order, for instance, because they repair different parts of a theory to solve different faults. Thus, we refine the naive search method so that it computes only Maximal Sets of Commutatives Repairs (MSCR). As the repairs in a MSCR can be applied together, the search space of fault-free theories using MSCR is reduced dramatically.

Not all repairs are commutative.

(i) It is possible that after applying one repair r_1, another repair r_2 won't be needed because r_1 has also solved the fault which r_2 targeted.

(ii) On the other hand, r_1 may make r_2 inapplicable. For instance, r_1 may merge predicate $mother$ with predicate mum. Then, if r_2 would have deleted an axiom of $mother$, it cannot find it after r_1's application.

These are the scenarios where repairs are not commutative because applying them in different orders results in different repaired theories.

The commutation between repairs r_1 and r_2 is defined in Definition 2.7. $\mathbb{T} \cdot r$ represents the application of a repair r to a theory $\mathbb{T}$.

Definition 2.7 (Commutative Repairs)**.** *Two repairs r_1 and r_2 are commutative if applying them in different orders to theory $\mathbb{T}$ results in the same repaired theory.*

$$\mathbb{T} \cdot r_1 \cdot r_2 = \mathbb{T} \cdot r_2 \cdot r_1 \tag{2.1}$$

Accordingly, the maximal set of commutative repairs are defined below.

Definition 2.8 (Maximal Set of Commutative Repairs)**.** *Given the whole set of all possible repairs $\mathbb{R}$ for all detected faults in the theory $\mathbb{T}$, a MSCR $\mathbb{M}$ is a maximum set of $\mathbb{T}$'s commutative repairs if they avoid the scenarios illustrated in (i) and (ii). We can formalise this as follows:*

$$\exists r_m \in \mathbb{M},\ \forall r \in \mathbb{R} \setminus \mathbb{M},\ \mathbb{S}(\mathbb{T}, r) \wedge \mathbb{S}(\mathbb{T}, r_m) \neq \emptyset \tag{2.2}$$

$$\forall r_1, r_2 \in \mathbb{M},\ if\, r_1 \neq r_2,\ then\ \mathcal{F}(\mathbb{T},\ r_1) \neq \mathcal{F}(\mathbb{T},\ r_2) \tag{2.3}$$

where $\mathbb{S}(\mathbb{T}, r) = \{\alpha | \alpha \in \mathbb{T} \wedge \alpha \notin \mathbb{T} \cdot r\}$ is the scope of a repair r in the theory $\mathbb{T}$, and $\mathcal{F}(\mathbb{T}, r)$ is the fault in $\mathbb{T}$ that a repair r solves.

There could be n MSCRs, where $n \geq 1$, and a repair can belong to more than one MSCRs. ABC computes all MSCRs and applies each MSCR separately to produce n semi-repaired theories, which will be delivered for the next round of fault detection and repair generation. By applying repairs in a MSCR together, the search space is reduced because the search branches of grouped repairs are merged into one. The comparison of search spaces are drawn in Figure 4.

Inspired by the sub-optimal pruning of ABC repairs based on Max-Sat [8], an optimal MSCR is defined as follows:

Definition 2.9 (Optimal MSCR). *A MSCR, $\mathbb{M}_1$, is optimal for the theory $\mathbb{T}$ if and only its estimated cost $c(\mathbb{T}\ \mathbb{M}_1)$ is not bigger than any of the MSCRs of that theory, denoted as $\mathbb{M}_2$:*

$$\forall \mathbb{M}_2.\ c(\mathbb{T},\ \mathbb{M}_1) \leq c(\mathbb{T},\ \mathbb{M}_2) \tag{2.4}$$

where the estimated cost is:

$$c(\mathbb{T},\ \mathbb{M}) \quad = \quad |\mathbb{M}| + \mathcal{N}_{insuff}(\mathbb{T}_m) + \mathcal{N}_{incomp}(\mathbb{T}_m)$$

where $|\mathbb{M}|$ is the number of repairs in $\mathbb{M}$ and $\mathcal{N}_{insuff}(\mathbb{T}_m)$ and $\mathcal{N}_{incomp}(\mathbb{T}_m)$ are the number of insufficiencies and incompatibilities of $\mathbb{T}_m$, respectively, and where $\mathbb{T}_m$ is the repaired theory produced by applying all repairs in $\mathbb{M}$ to $\mathbb{T}$.

By only taking the optimal MSCRs to the remaining repair process, the search space of fault-free theories is further reduced. This method dramatically saves time and space.

3. Illustrative Example

In this section we illustrate the operation of ABC using the black swan theory, which is given in Example 3.1. This example was drawn from the belief revision literature [9], where the proposed repair operations are to remove one of the four axioms: $A1 - A4$.

Let the theory in 3.1 be $\mathbb{T}$.

Example 3.1 *The Black Swan Theory:* $\mathbb{T}$

$$\begin{aligned} german(X) &\implies european(X) && \text{(A1)} \\ european(X) \wedge swan(X) &\implies white(X) && \text{(A2)} \\ &\implies german(bruce) && \text{(A3)} \\ &\implies swan(bruce) && \text{(A4)} \end{aligned}$$

$\mathcal{T}(\mathbb{PS}) = \{black(bruce)\},\ \mathcal{F}(\mathbb{PS}) = \{white(bruce)\}$

$\mathbb{T}$ has both an incompatibility fault and an insufficiency one.

$$\begin{aligned} \mathbb{T} \vdash white(bruce) \quad &\wedge \quad white(bruce) \in \mathcal{F}(\mathbb{PS}) \\ \mathbb{T} \not\vdash black(bluce) \quad &\wedge \quad black(bruce) \in \mathcal{T}(\mathbb{PS}) \end{aligned}$$

$\mathbb{T}$ illustrates a limitation of relying on just belief revision for repairing faulty theories. None of the four axiom removal operations results in what we suggest is the most natural repair to its incompatibility fault. We think this fault arises from the ambiguity of $european(X)$: it could mean 'X is a European variety' or 'X is resident in Europe'. In the first case, since $bruce$ is black then $european(bruce)$ is false, but in the second case it could be true if $bruce$ is resident in Germany, e.g., in a zoo[3]. ABC's Reformation 2c repair adds an extra argument to predicate $european$ that enables this distinction.

[3]A black swan can be seen in St James's Park, just opposite the Royal Society's HQ.

We will start by repairing this incompatibility fault. To see that $\mathbb{T} \vdash white(bruce)$, consider the SL proof in Figure 5.

$$\dfrac{\dfrac{\dfrac{\dfrac{white(bruce) \implies \qquad european(X) \wedge swan(X) \implies white(X)}{european(bruce) \wedge swan(bruce) \implies \qquad german(X) \implies european(X)}}{german(bruce) \wedge swan(bruce) \implies \qquad \implies german(bruce)}}{swan(bruce) \implies \qquad \implies swan(bruce)}}{\implies}$$

A different colour is used to highlight each pair of unifying propositions.

Figure 5. SL Resolution Steps of the Incompatibility.

The proof in Figure 5 can be broken at any of these four coloured unification steps. We will illustrate it being broken at the blue pair, i.e., between $european(bruce)$ and $european(X)$. We choose the repair operation Reformation 2c, which will add a new argument to $european$ to distinguish its two possible meaning. ABC is not able to assign meanings to new constants, so we use $abnormal$ to the instance in the goal clause and $normal$ to that in the axiom[4]. In this example, humans can interpret $abnormal$ as 'resident' and $normal$ as 'variety'. Note that instances in other rules in $\mathbb{T}$, such as (TA1) are assigned a new variable.The resulting (and desired) repair of $\mathbb{T}$ is given by Example 3.2 with changes highlighted in red.

Example 3.2 *The Desired Incompatibility Repaired Theory.*

$$german(X, Y) \implies european(X, Y) \qquad \text{(TA1)}$$
$$european(X, normal) \wedge swan(X) \implies white(X) \qquad \text{(TA2)}$$
$$\implies german(bruce, abnormal) \qquad \text{(TA3)}$$
$$\implies swan(bruce) \qquad \text{(TA4)}$$
$$\mathcal{T}(\mathbb{PS}) = \{black(bruce)\},\ \mathcal{F}(\mathbb{PS}) = \{white(bruce)\}$$

The incompatibility fault has been repaired, as the proof of $white(bruce)$ in Figure 5 is now broken.

We now illustrate ABC's repair of the insufficiency. Its simplest repair is adding the preferred proposition as an axiom directly using Abduction 1. This is illustrated in Example 3.3. The required proof of $black(bruce)$ consists of one step between the goal and this new axiom.

Example 3.3 A fully Repaired Black Swan Theory.

$$german(X, Y) \implies european(X, Y) \qquad \text{(A1')}$$
$$european(X, normal) \wedge swan(X) \implies white(X) \qquad \text{(A2')}$$
$$\implies german(bruce, abnormal) \qquad \text{(A3)}$$
$$\implies swan(bruce) \qquad \text{(A4)}$$
$$\implies black(bruce) \qquad \text{(A5)}$$

The current theory is faithful concerning $\mathbb{PS}$. The repaired theory is generated by combining reformation and abduction, and the solution satisfies the claimed repair postulates.

[4]This choice seems to work tolerably well for most examples.

ABC can find 39 ways to repair theory 3.1 by breaking the proof at different points and by choosing different repair operations to break it. Many of these repairs also support a meaningful interpretation. Work continues on mechanisms for preferring one repair over another.

4. Applications of Theory Repair

In §2 we claimed that:

The ABC System has a diverse range of applications in successfully repairing faulty representations.

Our evidence to support this claim is to present some diverse examples of faulty theories that the ABC system has successfully repaired.

(a) Defeasible Reasoning

Defeasible reasoning occurs when a rule is given, but there may be specific exceptions to it. We will call such a rule *defeasible*. In AI, defeasible reasoning has usually been formalised by some kind of *non-monotonic logic* [10]. Logical theories are normally[5] *monotonic*, i.e., adding extra axioms to a theory increases its set of theorems. In a non-monotonic theory adding a new axiom can sometimes override some applications of a defeasible rule, so that of proofs of some theorems will no longer hold.

We offer an alternative mechanism using only monotonic logical theories, i.e., a monotonic theory containing a faulty rule is repaired into another monotonic theory in which the fault has been eliminated.

The classic example of defeasible reasoning is about $tweety$ the penguin: a non-flying bird. We have formalised both the original faulty theory and its repair in Figure 4.1.

Example 4.1 Bird Theory $\mathbb{T}_b$ on the left and its Repair $\mathbb{T}_r$ on the right.

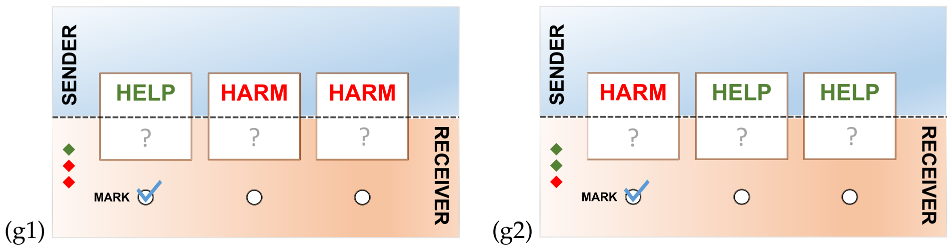
$\mathbb{T}_b$			$\mathbb{T}_r$		
$bird(X)$	$\Longrightarrow$	$fly(X)$	$bird(X, normal)$	$\Longrightarrow$	$fly(X)$
$bird(X)$	$\Longrightarrow$	$feathered(X)$	$bird(X, Y)$	$\Longrightarrow$	$feathered(X)$
$penguin(Y)$	$\Longrightarrow$	$bird(Y)$	$penguin(X)$	$\Longrightarrow$	$bird(X, abnormal)$
	$\Longrightarrow$	$penguin(tweety)$		$\Longrightarrow$	$penguin(tweety)$
	$\Longrightarrow$	$bird(polly)$		$\Longrightarrow$	$bird(polly, normal)$
	$\Longrightarrow$	$fly(polly)$		$\Longrightarrow$	$fly(polly)$

$$\mathcal{T}(\mathbb{PS}) = \{fly(polly), feathered(polly), feathered(tweety)\},\ \mathcal{F}(\mathbb{PS}) = \{fly(tweety)\}$$

The left hand theory $\mathbb{T}_b$ has an incompatibility fault: $\mathbb{T}_b \vdash fly(tweety)$ but $fly(tweety) \in \mathcal{F}(\mathbb{PS})$. The repair operation Reformation 2c is applied to add an additional argument to $bird$ in all occurrences in theory $\mathbb{T}_r$ and these are highlighted in red. This argument is given the value $normal$, $abnormal$ or a variable. This prevents the unification of $bird(X, normal)$ with $bird(X, abnormal)$, which is required to prove $fly(tweety)$. The proofs of the members of $\mathcal{T}(\mathbb{PS})$ are unaffected by this repair.

ABC also finds a repair in which Reformation 1c is used to rename $bird$ in the axiom to a new predicate $bird'$, representing flying birds, where $bird$ now means non-flying ones. To

[5]Some logicians regard monotonicity as a *defining* property. For them, a non-monotonic logic is an oxymoron.



The sender's view is at the top and the receiver's at the bottom. The sender knows what is in each box, the receiver doesn't. The receiver does, however, know how many helpful and harmful boxes there are, indicated by the red or green diamonds at bottom left. The sender marks one box with a tick and the receiver can see what was marked. The receiver can open as many boxes as they like, but must avoid harmful ones while opening as many helpful ones as possible.

Figure 6. Two Rounds of the Scorpion and Bananas Game.

avoid introducing an insufficiency, the axiom $bird(X) \implies feathered(X)$ axiom now has to be duplicated for $bird'$. This is a disadvantage of Reformation 1c over Reformation 2c in this case.

(b) Modelling Virtual Bargaining

Virtual Bargaining is a term coined by Cognitive Scientist, Nick Chater, to describe the extraordinary ability of cooperating humans to reach a, sometimes complex, agreement with only minimal channels of communication. It relies on their ability to put themselves in the shoes of their partner to imagine how they will understand these minimal communications [11].

To illustrate virtual bargaining, [11] invents a two-person cooperative game called *bananas and scorpions*. In this game, the human players need to guess or adjust the winning strategy based only on the others' game moves. The two players are the sender and the receiver. There are three boxes of two kinds: the helpful (containing bananas) and the harmful (containing scorpions). The sender knows all the boxes' contents and marks one of them to guide the receiver to choose a helpful box by marking only one box. On the other hand, the receiver only knows the number of each type of box, and which box is marked, and then aims to select as many helpful boxes as possible.

Two rounds of the game are illustrated in Figure 6. In the first round (g1), only box $b1$ is helpful and boxes $b2$ and $b3$ are harmful while the opposite is true in the second round (g2): $b2$ and $b3$ are helpful and $b1$ is harmful.

Given this limited bandwidth, each player has to imagine what the other is thinking and plan their play. In situation (g1), the sender has marked the only box marked 'Help'. This is a natural strategy to adopt. It's akin to pointing at the box you want to have opened. But in situation (g2), the strategy is less obvious. The sender could mark one of the two 'Help' boxes, but can't mark both, as only one tick is allowed. The receiver could then open this box, but this is a sub-optimal outcome, as the best outcome would be to open both 'Help' boxes. Bearing this in mind, the sender has changed strategy to mark the single 'Harm' box, intending the receiver to open both 'Help' boxes. Remarkably, the human players of the game frequently and spontaneous adopted this strategy, thus confirming their ability to do virtual bargaining. They did not first have to experiment to see which strategy was being used.

In [12], our research group modelled this process. Our Datalog theories were logic programs that the sender and receiver would invent and the receiver could then use to select which boxes to open. The faulty theory $\mathbb{T}$ that ABC generated an easily fixed insufficiency for g1 but failed completely on g2, because it would have opened a harmful box and would have failed to open either of the helpful ones.

The key rule in $\mathbb{T}$ was:

$$mark(X,Y) \implies select(X,Y) \tag{4.1}$$

which can be interpreted as 'select the marked box', where $mark(X,Y)$ means 'in game X mark box Y' and $select(X,Y)$ means 'in game X select box Y'. The preferred structure was:

$$\begin{aligned} \mathcal{T}(\mathbb{PS}) &= \{select(g_1,b_{green}),\ select(g_2,b_{green1}),\ select(g_2,b_{green2})\} \\ \mathcal{F}(\mathbb{PS}) &= \{select(g_1,b_{red1}), select(g_1,b_{red2}),\ select(g_2,b_{red})\} \end{aligned}$$

Using the red and green diamonds, this reflects the receiver's knowledge of the two games. In g_1, b_{green} stands for whichever box is known to be helpful and b_{red1} and b_{red2} the two boxes known to be harmful. In g_2, it's b_{green1} and b_{green2} that are known to be helpful and b_{red} that is harmful. Note that we must not assume that the receiver knows which actual boxes these correspond to, e.g., the receiver does not initially know that in g_1, b_{green} will turn out to be b_1, the marked box.

The insufficiency in g_1 is that $\mathbb{T} \not\vdash select(g_1, box_{green})$. The proof fails because $mark(g_1,b_{green})$ does not unify with $mark(g_1,b_1)$, where b_1 is the leftmost and marked box. This insufficiency is easily fixed using Reformation 3s, by merging b_1 and b_{green}.

In g_2, $\mathbb{T}$ has both incompatibility and insufficiency faults, namely:

$$\begin{aligned} \mathbb{T} \vdash select(g_2,b_{red}) &\quad\wedge\quad select(g_2,b_{red}) \in \mathcal{F}(\mathbb{PS}) \\ \mathbb{T} \not\vdash select(g_2,b_{green1}) &\quad\wedge\quad select(g_2,b_{green1}) \in \mathcal{T}(\mathbb{PS}) \\ \mathbb{T} \not\vdash select(g_2,b_{green2}) &\quad\wedge\quad select(g_2,b_{green2}) \in \mathcal{T}(\mathbb{PS}) \end{aligned}$$

The simple repair that worked for g_1 will not work for g_2 because both $select(g_2,b_{green1})$ and $select(g_2,b_{green2})$ have to proved and b_1 cannot be merged with both b_{green1} and b_{green2} because they are unequal.

Instead, ABC repairs the faulty $\mathbb{T}$ using the operation Belief Revision 2, namely adding extra preconditions to rule 4.1. This rule is duplicated and the two new rules are given different preconditions to distinguish the two situations: when there are more harmful than helpful boxes or the other way around. These two new rules are:

$$hp < hm \wedge mark(X,Y) \implies select(X,Y) \tag{4.2}$$

$$hm < hp \ \wedge\ Y \neq Z \ \wedge\ mark(X,Z) \implies select(X,Z) \tag{4.3}$$

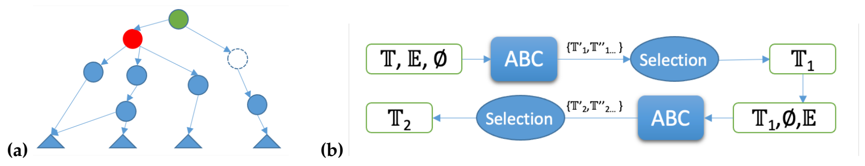
where hm is the number of harmful boxes and hp the number of helpful ones in the game. Using rule 4.3, the repaired theory suggests opening all boxes that are not marked[6] With these new rules, the repaired theory is able to correct the incompatibility and both the insufficiencies.

(c) Root Cause Analysis

Root Cause Analysis based on system logs of network systems [7], where single causes can trigger multiple failures. Taking the input theory containing the information from system logs and domain rules, ABC can detect missing information (MI) that is essential to cause failures and then suggest repairs to fix root causes [13].

Figure 7 (a) shows the damage caused by MI in RCA. All nodes are explicitly in the theory, which models the software system, except the dashed node[8]. Due to MI, the green node will not be diagnosed as the root cause of all four failures, as it should be. Figure 7 (b) depicts ABC's workflow in RCA, where the faulty $\mathbb{T}$, which lacks MI, is repaired into $\mathbb{T}_1$ to cover the previous missed information and then $\mathbb{T}_2$ includes repairs that fixes root causes.

[6]Note that, in order to stay within the Datalog grammar, $\neq$ is not the negation of $=$, but a binary predicate in its own right with the same meaning.
[7]To avoid misunderstanding, 'system' in this subsection only refers to the object where system failures occur, rather than the ABC repair system.
[8]A cause may be missing while its logical consequence exists in a KM, e.g., only the latter is recorded in the log.


(a)
(b)
$\mathbb{T}, \mathbb{E}, \emptyset$
ABC
$\{\mathbb{T}'_1, \mathbb{T}''_{1\ldots}\}$
Selection
$\mathbb{T}_1$
$\mathbb{T}_1, \emptyset, \mathbb{E}$
ABC
$\{\mathbb{T}'_2, \mathbb{T}''_{2\ldots}\}$
Selection
$\mathbb{T}_2$

(a) Failed RCA due to MI. *Triangles are propositions describing system failures; circle nodes are axioms or theorems representing system behaviours; an arrow starts from a behaviour's representation to its logical consequence's; the dashed node corresponds to the axiom that should be added to represent MI, which is not in the original theory that describing the network system;*

(b) ABC's flow of two-step RCA. *RCA's input are 1) KM $\mathbb{T}$; 2) the observed system failures as a set of assertions $\mathbb{E}$. RCA's output is the repaired KM $\mathbb{T}_2$ where the root cause is addressed. Here ABC's inputs are a KM, $\mathcal{T}(\mathbb{PS})$ and $\mathcal{F}(\mathbb{PS})$ in turn: in the first step $\mathcal{T}(\mathbb{PS}) = \mathbb{E}$, $\mathcal{F}(\mathbb{PS}) = \emptyset$; ABC outputs potential repairs $\{\mathbb{T}'_1, \mathbb{T}''_1 \ldots\}$, from which the selected $\mathbb{T}_1$ is the input KM of the second step, where $\mathcal{T}(\mathbb{PS}) = \emptyset$, $\mathcal{F}(\mathbb{PS}) = \mathbb{E}$.*

Figure 7. ABC in Root Cause Analysis.

The example given by [13] is about microservices in a network system. *In the first step of RCA*, ABC detects two insufficiencies in the original faulty theory, because the theory fails to predict two microservices failures. The first insufficiency is caused by the MI of a microservice session: that id is built on a full board $d1$. Thus, the repair is to add the corresponding axiom (4.4) (in green). The other insufficiency is cased by the mismatch between the predicate $microservice$ in (4.5) and the predicate ms in (4.6) (in red). The repair is to merge them, e.g., rename ms to $microservice$. In the second step of RCA, ABC changes the full board to not being full, which solves all system failures: two microservices $id1$ and $id2$ deployed on that board and another microservice $id3$ depends on the failed $id2$ according to rule (4.6).

$$\Longrightarrow \quad createOn(id1, d1) \tag{4.4}$$

$$\Longrightarrow \quad microservice(id1, s1) \tag{4.5}$$

$$ms(X, s1) \wedge ms(Y, s2) \wedge sameRoute(X, Y) \quad \Longrightarrow \quad depend(Y, X) \tag{4.6}$$

In this example, the two insufficiencies are caused by the incompleteness of the system log: the information of (4.4) was missing, and the theory is formalised from multiple data sources which caused the mismatch between (4.5) and (4.6), respectively. Then the repair of the former reminds engineers to improve the log quality by adding the MI. The repair of renaming ms to $microservice$ contributes to aligning the knowledge from the different data sources. In addition, the repair of the second step of RCA provides the solution of fixing these system failures: ensuring that board $d1$ is not full.

(d) New Physics by Analogy

For his 2016 MSc project, Cheng-Hao Cai applied reformation to the problem of correcting faulty analogies [14,15]. One of these faulty analogies was between gravitational attraction and electrostatic attraction/repulsion. In particular, an approximation to Coulomb's Law of electrostatic force can be generated from Newton's law of universal gravitation

$$F \quad = \quad G.\frac{m_1.m_2}{r^2}$$

where F is the gravitational force acting between two objects of mass m_1 and m_2, r is the distance between their centres of mass, and G is the gravitational constant. However, corresponding electrostatic charges repel rather than attract and G must be replaced by Coloumb's constant k_e.

2016 predates ABC, so Cai applied just reformation. He also needed equational reasoning in addition to resolution, so adapted reformation to work with the Z3 solver [16]. The equivalents of Reformation 1s was used to change attraction to repulsion and Reformation 3s to change G to k_e.

(e) Modelling Student Misconceptions

When students learn arithmetic, they may make mistakes. Jovita Tang's 2016 MSc project used reformation to model students' misconceptions of arithmetic procedures [17]. The preferred structure was the students' incorrect answers to arithmetic problems $\mathcal{T}(\mathbb{PS})$, with the correct mathematical calculation rules $\mathbb{R}$ treated as the original faulty theory. ABC repaired $\mathbb{R}$ into a theory that models the student's incorrect mathematical calculation, i.e. theory $\mathbb{R}'$ is a logic program that derives the student's miscalculations in $\mathcal{T}(\mathbb{PS})$ as theorems. The repairs required to do this highlight the student's misconception.

5. Conclusion

We have argued that representational change is integral to reasoning for both humans and computers. As arguments are fleshed out, faults are exposed by the reasoning process that then have to be repaired. Such repair frequently involves an elaboration of the representation on which the reasoning is based. New concepts must be defined or existing ones refined. Unexpected distinctions are required and the language of the representation refined to attest them. For instance, an analysis of Lakatos'"Proofs and Refutations" shows that, even in mathematical proofs, the objects of the reasoning are revealed to be vague and require elaboration.

To formalise representational change, we have implemented the ABC system, that combines abduction, belief revision and conceptual change to repair faulty theories. To demonstrate the generality and range of ABC, we describe diverse successful applications to defeasible reasoning, virtual bargaining, root cause analysis, analogy repair and modelling student misconceptions. These applications include both technological and cognitive domains. It is sometimes necessary to model faults, e.g., in a 5G network or a student's understanding. In these cases, ABC can be used to work backwards, from a theory of the ideal situation, via symptoms of the fault, to the faulty situation being modelled.

Data Accessibility. Insert details of how to access any supporting data here.

The ABC system can be downloaded from GitHub at https://github.com/XuerLi/ABC_Datalog.

Authors' Contributions. Alan Bundy defined and led the representational change research programme that is summarised in this paper. Xue Li designed, implemented and evaluated the ABC system.

Competing Interests. The authors declare that they have no competing interests.

Funding. The EPSRC Human-Like Computing Network+ funded the work on virtual bargaining. Huawei project CIENG4721/LSC funded the further development of the ABC system and the application to root cause analysis. Also ELIAI (The Edinburgh Laboratory for Integrated Artificial Intelligence) EPSRC (grant no EP/W002876/1) funded Xue Li.

Acknowledgements. Thanks to the co-authors on the papers and dissertations outlined in this survey: Chenghao Cai, Jovita Tang and Eugene Philalithis. Also to Xue's other supervisors: Alan Smaill and Ewen Maclean. We gratefully acknowledge the support from Huawei, particularly Lei Xu, Stefano Mauceri and Zhenhao Zhou. We thank Predrag Janičić for redrawing the diagrams in Figures 1 and 2.

References

1. Lakatos I. 1976 *Proofs and Refutations: The Logic of Mathematical Discovery*. Cambridge University Press.
2. Li X, Bundy A, Smaill A. 2018 ABC repair system for Datalog-like theories. In *KEOD*, pp. 333–340.

3. Li X, Bundy A. 2022 An overview of the ABC repair system for Datalog-like theories. In *The Third of International Workshop on Human-Like Computing (HLC 2022)*, volume 28, p. 30.
4. Kowalski RA, Kuehner D. 1971 Linear resolution with selection function. *Artificial Intelligence* **2**, 227–60.
5. Ceri S, Gottlob G, Tanca L. 1990 *Logic Programming and Databases*. Surveys in Computer Science. Berlin: Springer-Verlag.
6. Robinson JA. 1965 A machine oriented logic based on the resolution principle. *J Assoc. Comput. Mach.* **12**, 23–41.
7. Pfenning F. 2006 *Datalog*. Lecture 26. 15-819K: Logic Programming.
8. Urbonas M, Bundy A, Casanova J, Li X. 2020 The use of max-sat for optimal choice of automated theory repairs. In *Artificial Intelligence XXXVII* (ed. M Bramer, R Ellis), pp. 49–63. Cham: Springer International Publishing.
9. Gärdenfors P. 1992 Belief revision: An introduction. In *Belief Revision* (ed. P Gärdenfors), pp. 1–28. Cambridge University Press. Cambridge Tracts in Theoretical Computer Science
10. Strasser C, Antonelli GA. 2019 Non-monotonic logic. In *The Stanford Encyclopedia of Philosophy* (ed. EN Zalta). Metaphysics Research Lab, Stanford University, summer 2019 edition.
11. Misyak J, Noguchi T, Chater N. 2016 Instantaneous conventions: The emergence of flexible communicative signals. *Psychological science* **27**, 1550–1561.
12. Bundy A, Philalithis E, Li X. 2021 Modelling repairs to virtual bargaining via representational change. In *Human-Like Machine Intelligence* (ed. SH Muggleton, N Chater), pp. 68–89. Oxford University Press.
13. Li X, Bundy A. 2022 ABC repair system in root cause analysis by adding missing information. In *The 8th International Online & Onsite Conference on Machine Learning, Optimization, and Data Science, special session of AI for Network/Cloud Management*.
14. Cai CH. 2016 *The Application of Reformation to Repair Faulty Analogical Blends*. MSc thesis, School of Informatics, University of Edinburgh.
15. Cai CH, Bundy A. 2022 Repairing numerical equations in analogically blended theories using reformation. In *The Third of International Workshop on Human-Like Computing* (ed. A Bundy, D Mareschal). CEUR.
16. Moura Ld, Bjørner N. 2008 Z3: An efficient smt solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, pp. 337–340. Springer.
17. Tang JWQ. 2016 *Arithmetic Errors Revisited: Diagnosis and Remediation of Erroneous Arithmetic Performance as Repair of Faulty Representations*. MSc thesis, School of Informatics, University of Edinburgh.