Algebra Diagrams: A HANDi Introduction

Peter C-H. Cheng

Department of Informatics, University of Sussex, Brighton, BN1 9QH, UK

Email: p.c.h.cheng@sussex.ac.uk

Abstract. A diagrammatic notation for algebra is presented – Hierarchical Algebra Network Diagrams, HANDi. The notation uses a 2D network notation with systematically designed icons to explicitly and coherently encode the fundamental concepts of algebra. The structure of the diagrams is described and the rules for making derivations are presented. The key design features of HANDi are discussed and compared with the conventional formula notation in order demonstrate that the new notation is a more logical codification of introductory algebra.

1 Introduction

This paper describes novel notational system for introductory (school level) algebra – Hierarchical Algebra Network Diagrams (HANDi). HANDi was invented as part of a programme of research that is developing the *Representational Epistemic* approach to the study of how notational systems encode knowledge and the potential cognitive benefits that novel codifications of knowledge may confer [1-3]. The core principle of the Representational Epistemic approach claims that notational systems designed to directly encode the fundamental conceptual structure of knowledge rich domains, using coherent notational schemes, will possess semantic transparent and thus enhance problem solving and conceptual learning (see [1-3]).

This paper that has two purposes: (1) to provide a detailed description of HANDi, that includes a set rules for making HANDi derivations; (2) to highlight the main design features of the notation that attempt to more coherently encode the fundamental conceptual structure of algebra than the conventional formula notation. Thus, the paper has the 4 following main sections: section 2 present the basic graphical structure the HANDi notation; section 3 describes the rules for manipulating expression in the notation;

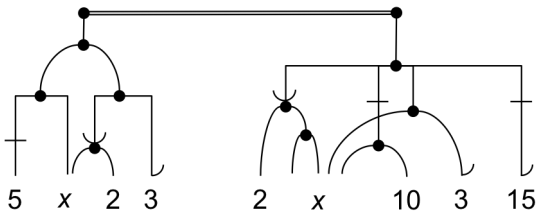


Fig. 1. A HANDi equation comprising two trees

section 4 provides examples of derivation using the rules; section 5 discuses the how the HANDi notation encodes the key concepts of the domain.

2 HANDi expressions

Fig. 1 shows an example of a Hierarchical Algebra Network Diagram for a quadratic equation in *x* that involves a complex number. The equation in the conventional formula notation equation is shown below the diagram. This diagram consists of two *trees*, left and right, but other diagrams may have multiple trees. The full interpretation of the diagram will become clear as the component parts of HANDi equations are introduced.

2.1 Basic operators

Fig. 2 shows the HANDi sub-tress for basic binary operators, which consist of a node, trunk and two branches. At the end of each branch is either an *argument* leaf that consists of a number, a variable, or another a node for an operator (as in Fig. 1). The *trunk* of a node is the line ascending from the top of the node. A *right-angle branch* represents addition of the arguments (or node) at the end of the branch. A right-angle branch with a horizontal bar represents subtraction of the argument at the end of the branch. An arc branch represents multiplication, in the same way. An arc branch with a diagonal bar represents division by the argument on the branch. The different styles of lines distinguish addition and subtraction from multiplication and division. The bars distinguish subtraction from addition and division from multiplication. All this conveys the notation that division is to multiplication as subtraction is to addition.

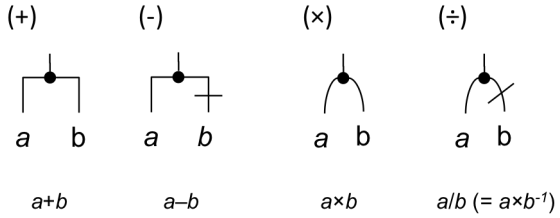


Fig. 2. Trees for the elementary algebraic operators

Different styles of branches may not be connected to the same node, and a diagonal bar may only be associated with an arc branch and a horizontal bar with a right-angle branch; Fig 3 shows three illegal diagrams.

2.2 Equation

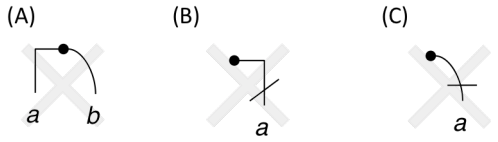


Fig. 3. Invalid tree and branch types

Relations among numbers and variables are encoded as hierarchical networks, *trees*, composed of the basic operators. The numerical equality of trees is shown a double horizontal line connecting their topmost trunk; for example Fig 1. (Inequalities may be shown by a double horizontal line with a $<$, $\leq$, $>$, $\geq$ symbol written on the line.) A tree may only have a single occurrence of each variable; there is just one

letter '*x*' in each of the two trees in Fig. 1. This is a key feature of HANDi. However, there may be multiple instances of the same number in a tree, for two reasons: (a) the same numbers may stand for quantities that have different units, in which case they necessarily represent different things (e.g., 3 metres versus 3 seconds), so should have different symbols; (b) pragmatically, this provides a means to manage the complexity of the diagrams, particularly in order to avoid crossing branches.

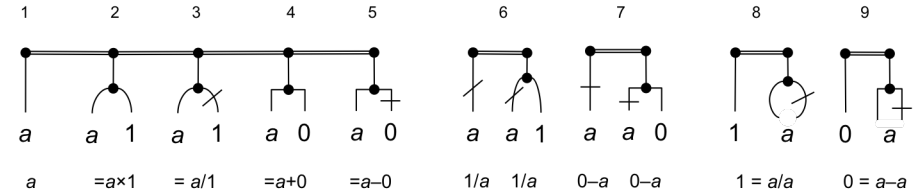


Fig. 4. Operations involving identify element

2.3 Identities and inverse operations

Fig. 4 shows how a sub-tree with a single branch may be drawn in place of a binary tree that has the same argument and a particular identity element. Fig 4.1 is a sub-tree with a single branch, and no bar, that is equivalent to Figs. 4.2-4.5. Figs. 4.6 and 4.7 show that single branches with a diagonal or horizontal bar are equivalent to trees in which the argument is the divisor of one or is subtracted from zero. Fig. 4.8 and 4.9 shows that an argument divided by or subtracted from itself are equal to one or zero, respectively.

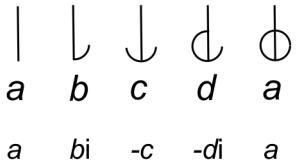


Fig. 5. Positive, imaginary, negative, negative imaginary and positive arguments

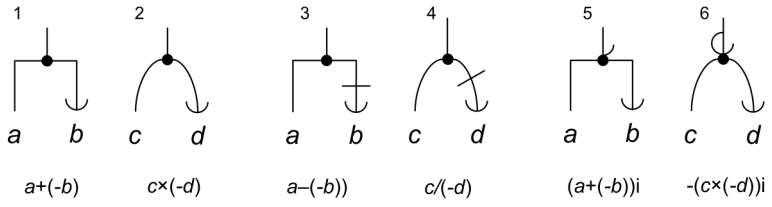


Fig. 6. Turns in trees

2.4 Negative and imaginary numbers – unary turn operators

HANDi provides a notational scheme that coherently unifies positive, negative and imaginary numbers; see Fig. 5. A *turn* is a unary operator at the end of branch that indicates the extent of rotation of the argument in an Argand diagram. A quarter turn (1/4 circle) represents an imaginary argument, a half turn (semi-circle) represents a

negative argument, a three quarter turn (3/4 circle) represents a negative imaginary argument, and positive argument is a branch with no turn or a complete (whole circle).

2.5 Combined unary and higher order operators

Fig. 6 shows unary turn operators in association with binary operators. Addition, subtraction, multiplication and division operators may act upon negative numbers; Figs. 6.1 to 6.4. HANDi makes an explicit distinction between negative numbers and subtraction in terms of the type of attachment to the branch, either a half turn or a bar, and in relation to the position of the attachment, either at the end or bisecting the branch. Fig. 6.5 and 6.6 indicate how unary turn operator may be applied to trees. Notice how in Fig. 6.6 the minus sign and 'i' occur at opposite ends of a conventional formula, but the same information is encoded as a single three quarter turn symbol in HANDi.

2.6 Repeated operations – recursive addition and powers

Figs. 7 and 8 show HANDi expressions that involve repeated applications of an operator to an argument. Fig. 7.1-7.4 shows multiple additions of a single argument, including recursive applications. Figs. 7.5-7.10 show how HANDi represents various power expressions for a single argument. In cases where multiple branches of a tree converge on a single sub-tree below (Figs. 7.4, 7.9 & 7.10 and Fig. 8.5), each branch represents a repeated operation on the sub-tree: Fig. 7.4 shows two additions of 3*a*; Fig. 7.9 shows the product of three a^3; Fig. 8.5 shows the product of two $a \times b$. Notice that the top tree in Fig. 7.9 is equivalent to the top tree in Fig. 7.8, which both represent the cube of their respective arguments. Consider Fig. 7.10; raising an index of an argument (2 in a^2) to a power (3 in $a\text{^}2^3$) is represented by a number of the hierarchical repetitions of the sub-tree for the index (2) by the magnitude of the power (3): there is a stack of three pairs of branches. Fig. 8.4 shows that arguments raised to some power may be embedded as a sub-tree within a higher tree. If all the branches of the tree have diagonal bars, which indicate division by the argument, then the tree represents a negative power, as in Fig. 8.1.

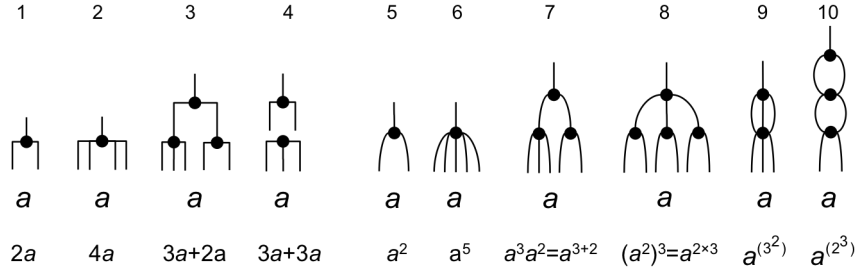


Fig. 7. Repeated addition and powers

To show non-integer fractional powers, branches that ascend as well as descend from the node are used. Fig. 8.2 has two ascending branches and one descending branch, so its power is 1/2 (square root). Fig 8.2 shows $a^{2/3}$, because two branches point down and three point up. By the same scheme, the two down and one up branches in Fig. 7.5 shows the argument raised to the "fractional" power of 2/1.

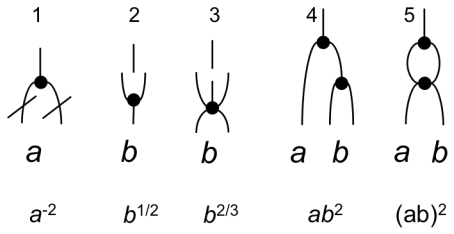


Fig. 8. Assorted power diagrams

That completes the description of the graphical structure of HANDi trees.

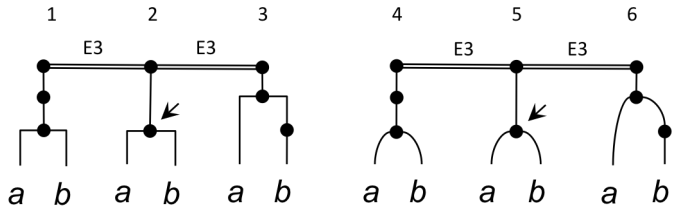


Fig. 9. Branch insertion or elimination

3 Elementary HANDi transformation rules

HANDi provides rules for the transformation of tree diagrams in to alternative forms. Elementary rules are the primitive transformations – considered in this section – and derived rules are more complex transformations composed by the successive applications of the elementary rules – considered in the next section.

HANDi has twelve elementary rules for transforming trees and equations.

Rule E0 specifies that horizontal positioning of argument and nodes is arbitrary so their relative positions may be swapped at will; for instance, to improve the clarity of a diagram. The *E0* labels on the double equality line in Figs. 15 and 23 (below) show the application of this rule.

Rule E1 allows binary trees for operations on an argument and an identity element to be replaced by a single branch for the argument, as described above and shown in Figs. 4.2-4.7. In the case of Rule E1.1 the single branches for the arguments have no bars (Fig 4.2-5), whereas in Rule E1.2 each branch has a bar (Figs. 4.6 & 4.7). The opposite also applies; a single

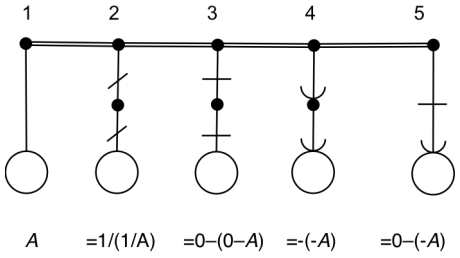


Fig. 10. Nest operations that are equivalent to a tree with a single plain branch

branch may be transformed in to a binary tree (e.g., Fig 4.1 in to any of Figs. 4.2-4.6)

Rule E2 also allows a binary tree to be replaced by a single branch, or vice versa, but in this case the argument on the branch is either one or zero, because the binary tree involves an argument divided by itself or subtract from itself, as described above and shown in Figs. 4.8 & 4.9.

Rule E3 allows a single branch with no bar to be inserted (or removed) from a tree. Going from Fig. 9.2 to 9.1, or Fig. 9.5 to 9.4, an extra node and branch are inserted above the node in the tree indicated by the arrow. Going from Fig. 9.2 to 9.3, or Fig. 9.5 to 9.6, an extra branch and node are inserted below the indicated node of the tree. In the opposite direction (to Fig. 9.2 or to Fig. 9.5) the node and branch are eliminated.

Rule E4 involves trees with single branches, Fig. 10. The circles and capital '*A*' stand for an argument or a sub-tree. Three trees have two nested branches for the same type the operator (Fig 10.2-4) and one has two different operators (Fig 10.5). In all cases the trees are equivalent to a single plain branch (Fig 10.1). For instance, Fig 10.4 states that taking the negative of a negative gives positive and Fig 10.5 shows that subtracting a negative is equivalent to a positive. Thus, the variants of E4 allow such nested branches to be replaced by a single plain branch, or vice versa.

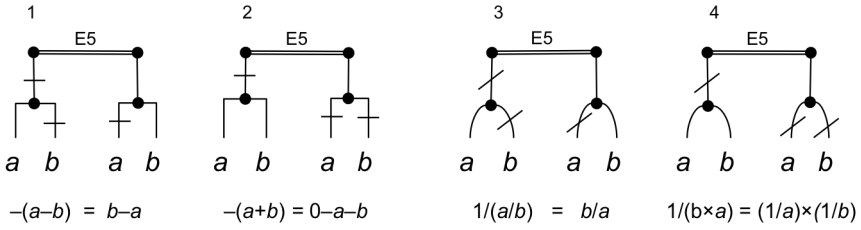


Fig. 11. Promotion and demotion of bars

Rule E5 concerns the movement of bars up and down branches, as shown in Fig. 11. A bar on a trunk can be move down to any branch that has no bar, but if a bar already exists on the branch it is cancelled out by the moved bar. The rule applies equally to right-angle and arc trees. For example, Fig. 11.1 and 11.3 show that the bar on the *b* branch is eliminated and a bar added to the *a* branch, when the bar of the trunk is demoted.

Rule E6 concerns the relation of right-angle trees to trees with arc branches. Fig. 12 is an example of how repeated additions of the same argument may be represented as a multiplication operation. A tree with multiple right-angle branches for a single argument may be transformed into a binary tree with arc branches, one for the argument and another with a number matching the quantity of right-angle branches.

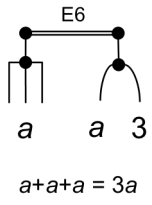
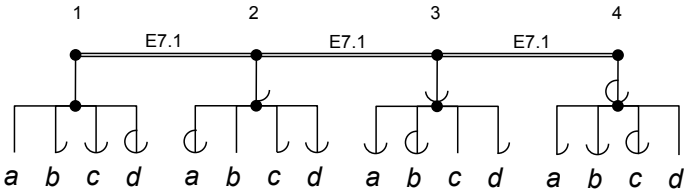


Fig. 12. Multiplication and repeated addition

Rule E7. Depending on the type of tree, transforms involving turns are governing by different conservation rules. In a tree with right-angle branches (with or without horizontal bars), the number of turns in successive right-

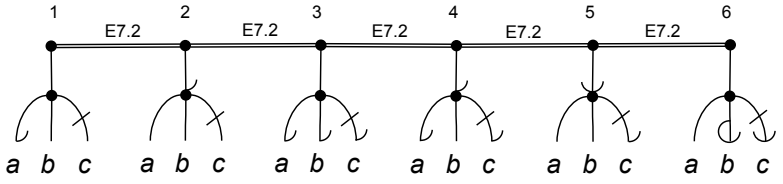
angle branches must be preserved, with the proviso that one complete turn is equivalent to no turn. For example, for variable *a* in Fig. 13.1, there are no turns along the branches from the top of the tree down to the *a* leaf, so when a quarter, half and three-quarter turn are introduced on the trunk, Figs. 13.2–13.4, the *a* branch must be augmented with sufficient fractional turns that the total number of turns is the same, a complete turn (or none). Similarly, with branches *b*, *c* and *d*, which start with a quarter, half and three-quarter turn, respectively.



$a+bi+(-c)+(-di) = ((-ai)+b+ci+(-d))i = -((-a)+(-bi)+c+di) = -(ai+(-b)+(-ci)+d)i$

Fig. 13. Conservation of turns along successive right-angle branches in equivalent trees

The conservation of turns in trees with arc branches applies to the whole tree, so all the turns on all the branches are summed, but with the exception that turns on branches that also have diagonal bars must be deducted. In Fig. 14 all the trees have a total of one quarter turn. In Fig 14.1 and 14.2 the quarter turn is associated with one argument or the whole tree, respectively. In Figs 14.3-5 a quarter turn is associated with a branch that has a diagonal bar, variable *c*, which means that there must be a total of a half turn elsewhere in the rest of each of the trees, so that all the trees possess the same number of turns. In Fig 14.6 the half turn associated with that branch will cancel most of the three-quarter turn on the middle branch (variable *b*) to leave the requisite quarter turn.



$ai \times b/c = (a \times b/c)i = ai \times bi/ci = (ai \times b/ci)i = -(a \times b/ci) = a \times (-bi)/(-c)$

Fig. 14. Conservation of turns throughout equivalent trees with arc branches

Rule E8. This rule encodes the property of distribution of multiplication over addition. In Fig 15.1 the topmost tree has arc branches, one of which is for the variable '*a*' and the other attaches to a tree with right-angle branches (*b*+*c*). Now, as the arc branch attached to the right-angle tree has *no bar*, the right-angle tree may be promoted and its branches attached to two sub-tree with arc branches below, as shown in Fig. 15.2. Note that the two new arc-branched trees now share the single variable (*a*) from

the original arc-branched tree. Figs. 15.4 and 15.3 are variants of Figs. 15.1 and 15.2 that simply change the horizontal position of the variables according to rule E0. This rule applies to right-angle trees with more than two branches and may be derived by repeated application of Rule E8 to successive right-angle branches.

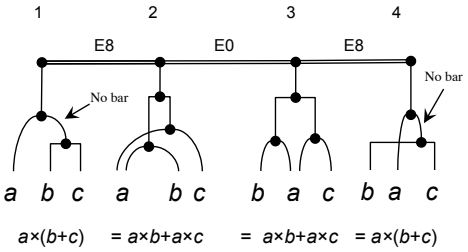


Fig. 15. Distributive property of multiplication over addition

Figs 16.1 and 16.2 show further two valid applications of rule E8 that also involve the presence of bars and turns. Again, note the absence of a bar on the critical arc branch, but the permitted presence of a turn on the branch in Fig. 16.2. In both cases the redistribution of argument *a* does not directly affect the bars or the half turn, because they remain attached to their original branches. In contrast Fig 16.3 shows when E8 cannot be applied, because there is a bar on the critical arc branch, which may not be re-distributed through the right-angle tree when that branch disappears.

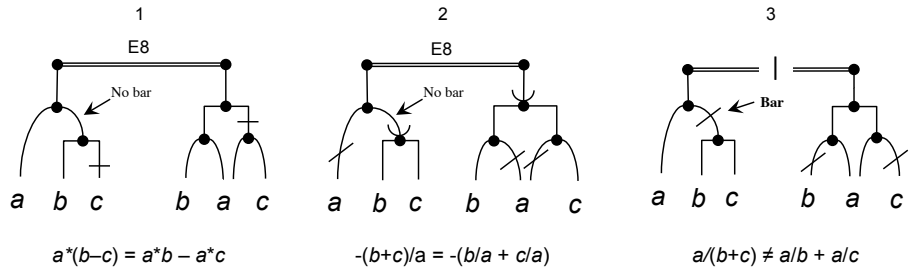


Fig. 16. Distributive property involving subtraction and division

Rule E9 concerns the transformation of trees for a single argument that have multiple arc branches, which may be nested, such as those shown in Figs. 7.5-7.10 and Fig. 8. Transformations of such trees should simply maintain the total effective number of branches. For example, in Fig. 17.1 a single branch for the upper sub-tree connects to a sub-tree with three arcs and a sub-tree with two arcs, hence there is a total of five branches overall, as shown in Fig. 17.2. In Fig 17.4 there are three lots of three arcs, which may be redrawn as nine branches, Fig. 17.5, or as the recursive application of a three-branch sub-tree to a lower level tree with three branches, Fig. 17.3. In general, successive sub-trees with multiple branches at different levels are multiplied, so as trees with upward pointing arcs represent fractional powers, we simply multiply by the appropriate fraction. In Fig. 17.6 there is a sub-tree with a pair

of arcs and above another with half an arc (two upside down arcs), thus the total number of branches is one, as shown in Fig. 17.7. Further, the transition from Fig. 17.1 to 17.2, and from Fig. 17.4 to 17.5 may also be interpreted as multiple applications of E3, which allows branches to be eliminated from trees.

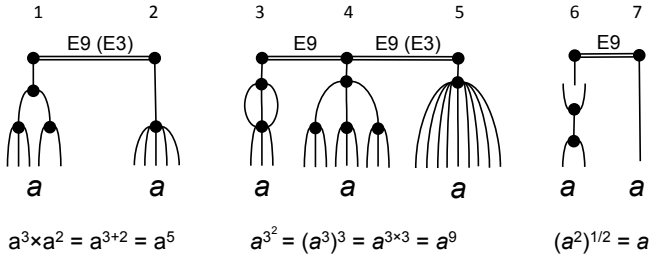


Fig. 17. Transformation of power expressions

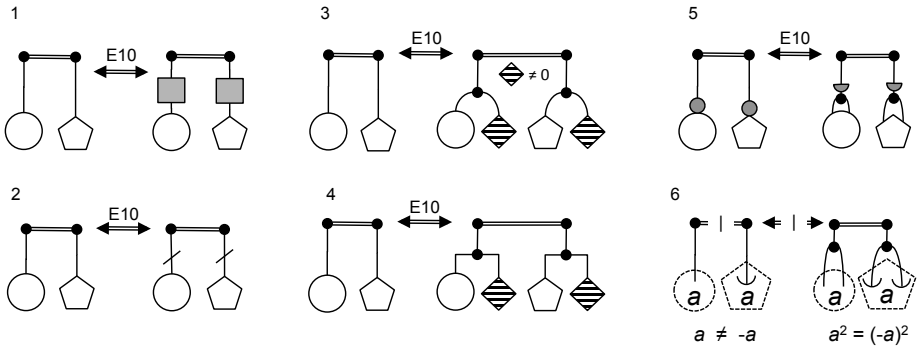


Fig. 18. Transformations to main trunks of equal trees

Rule E10. So far all the transformation rules have applied to individual trees. Consider now the transformation of multiple trees connected by the double horizontal line (for algebraic equality). This set of rules permits the same operation to be applied to all the trees attached to the double line, with some provisos. Rule E10.1 allows a horizontal or diagonal bar, or any turn, to be added to the top of both trees, or removed from both trees. Fig 17.1 shows this schematically, where the trees are represented by the circle and the pentagon and the grey squares stand for the same type of bar or number of turns that are to be added or removed: Fig. 17.2 is a particular example with diagonal bars. Rule E10.2 warrants the transformation of a tree with the incorporation (removal) of the same branch into both trees, as shown by the diamonds in Fig. 17.3 & 17.4. The one restriction is that the argument must not be a zero when the branches are curves, as this could yield trees are not equivalent (e.g., 2*0=3*0 but 2≠3). Further, Rule E10.3 states that the introduction of the operator must produce trees with equal overall numbers of turns, as illustrated schematically by the two grey circles and the two grey semi-circles on either side of Fig 17.5. Fig. 17.6 shows an invalid application of Rule E10.3, in which two trees with an equiva-

lent number of turns (right) are operated on to produce two new trees that do not have equal numbers of turn (left).

Rule E11. To calculate magnitudes of trees whose leaves are numbers, one may simply replace nodes with the number that results from applying the operators to the given values. For multi-level trees the process is performed in a recursive manner starting with the leaves.

Table 1 summarizes the 12 elementary rules for transforming HANDi expressions.

Table 1. Elementary HANDi transformation rules

Rule	Summary
E0	Horizontal position of arguments/nodes is arbitrary.
E1	Swapping a single branch with binary tree involving identities: E1.1) Single branch with no bar, Figs. 4.1-4.5. E1.2) Single branch with a bar, Figs. 4.6 & 4.7.
E2	Introduce/eliminate an argument with a pair of inverse operators: E2.1) Arc branches, Fig. 4.8 E2.2) Right-angle branches, Fig. 4.9
E3	Branch insertion/elimination at a node, Fig. 9.
E4	Equivalence of nested operations on successive single branches: E4.1) Two bars, Figs. 10.2 & 10.3; or two half turns, Fig. 10.4. E4.2) A horizontal bar and half turn, Fig. 10.5.
E5	Promotion/demotion of bars up/down equivalent branches, Fig. 11.
E6	Multiplication is repeated addition, Fig. 12.
E7	Conservation of number of turns with respect to: E7.1) Successive right-angle branches within a tree, Fig. 13. E7.2) Whole tree of arc branches (diagonal bars = subtraction), Fig 14.
E8	Distribute arc branch over a right-angle tree, Figs. 15 & 16.
E9	Equality of the total number arc branches for one argument, Fig. 17.
E10	Operations applied to (removed from) top trunk of equal trees, Fig. 18.
E11	Replace nodes with values calculated from number arguments.

4 Derived proofs and composite transformation rules

The elementary transformations of HANDi diagrams presented in the previous section can be applied to derive proofs or generate composite transformation rules. This section considers a few examples.

Fig. 19 shows the two applications of elementary rules: rule E1.1 turns a single branch tree into a binary tree by introducing a second arc branch with the number 1 as its argument; rule E2.1 then converts the 1 into to a further binary tree that involves the simultaneous multiplication and division of a new variable. For the purpose of a further example below that uses this derivation, it will be called composite rule C1.

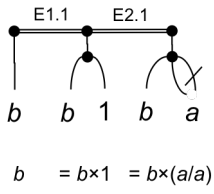


Fig. 19. Composite rule C1

Under the conventional approach the notion that addition and multiplication are associative is typically introduced in as fundamental property of algebraic formulas. In contrast, in HANDi the associative property may be treated as a composite rule that is derived from the elementary rule E3, for the insertion or elimination of a branch at a node. Fig. 20.2 may be redrawn as Fig. 20.1 or 20.3 by inserting a new right-angle branch to create a sub-tree. A diagram equivalent to Fig. 20.1-3 may also be drawn with arc braches for the associative property of multiplication. As E3 applies to any node, a branch may be inserted into a diagram with bars, for instance as shown in Fig. 20.4 and 20.5. However, branches with bar may not be inserted in this fashion, so Fig. 20.6 is not a valid transformation of Fig. 20.4 or 20.5: division is not associative. A diagram equivalent to Fig. 20.4-6 may also be drawn with right-angle branches showing the non-associative property of subtraction.

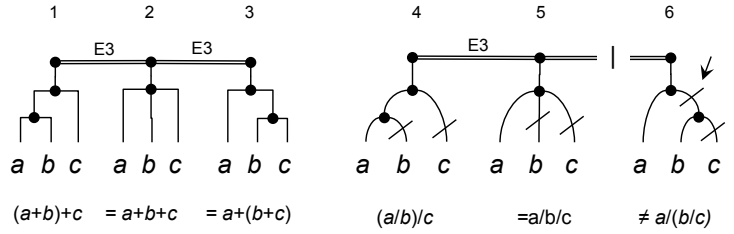


Fig. 20. Addition (& multiplication) is associative and division (& subtraction) is sometimes

Fig. 21 shows the proof that subtracting an argument is equivalent to applying the negation operator to the argument. Fig. 21.1 encodes elementary rule E4.2 that states that a positive argument is equal to the subtraction of the negative of the argument. A branch with a single horizontal bar may be inserted in both trees according to rule E10.2, so that left has one bar and the right has two, as shown in Fig. 21.2. Now, rule E4.1 permits the cancellation of the two bars on the right, so we obtain Fig. 21.3, which has just a half turn, the negation operator, to match the bar on the right, as required.

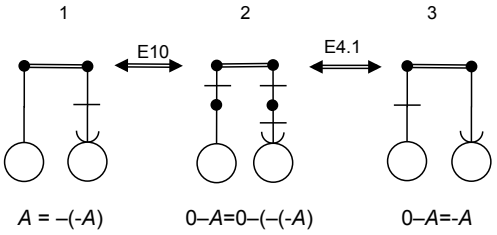


Fig. 21. Subtraction is equivalent to negation

The derivation of the product of two sums is shown in Fig. 22. This proceeds by applying the distributive transformation, rule E8, three times in succession. The parts of the diagram that are changed by each application of the rule are highlighted, which have the characteristic patterns found in Fig 15.1 and 15.2 (or their mirror image). In

the last step of the proof, Fig 22.4 to 22.5, all the unnecessary right-angle branches are eliminated, by rule E3, to reveal the four products of the variables. Of course, one may treat Figs. 22.1 and 22.5 as a composite transformation rule that simply says draw arc sub-trees for each combination of variables in the two right-angle trees and joint them all together with top level right-angle tree.

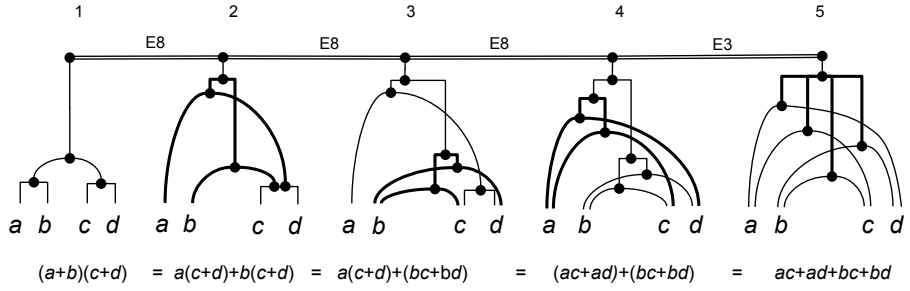


Fig. 22. Expansion of the product of two sums

Fig. 23 shows how the sum of reciprocals may be transformed by a diagrammatic procedure equivalent to cross multiplication. The first step is to form a sub-tree for each reciprocal in Fig. 23.1 with the variable of the other reciprocal by applying composite rule C1 (defined above, Fig. 19) to give Fig 23.2. The position of the pair of nodes may be swapped (applying rule E0) whilst the variables remain in place to yield Fig. 23.3, in which the interchanged nodes have been highlighted. Rule E3 may now be applied twice, as shown in Fig. 23.4, to introduce new branches to separate the arcs with bars from the ones without. As each sub-tree in Fig. 23.4 has branches with bars, the bars may be promoted to the next level by applying rule E5 (Fig. 23.5). The pairs of sub-trees in Fig. 23.5 are equivalent, so we have a situation like Fig 16.2 (right), where the sub-trees here map to '*a*' in that figure. Thus, rule E8 may be applied to coalesce the sub-trees to generate Fig. 23.6. With a little experience one learns to compose steps in similar proofs, for example jumping straight from Fig. 23.4 to Fig. 23.5 or even to Fig. 23.6. Again, the initial and final diagrams (Figs. 23.1 and 23.6) may be treated as a composite transformation rule.

As a final example, consider how laws of indices may be derived. Fig. 24 shows a

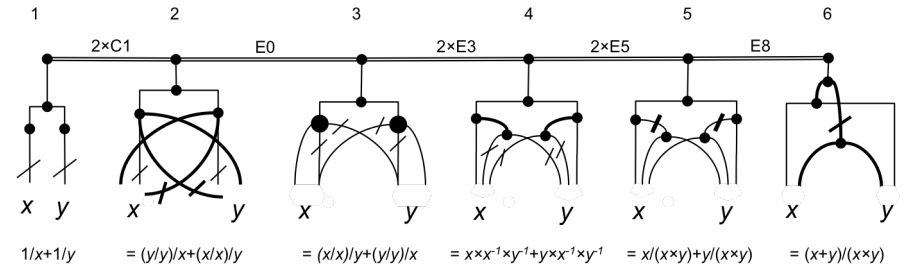


Fig. 23. Sum of two reciprocals

cubed variable divided by the square of the same variable. The proof simply involves: (i) demoting the diagonal bar on the trunk of the binary tree to its branches by applying E5 to give Fig. 24.2; (ii) eliminating branches of the upper tree using E3 twice, Fig. 24.3; (iii) eliminating pairs of arc branches with and without a bar to give branches ending in 1 by applying E2 twice, Fig. 24.4; and, (iv) deleting the branches ending in 1 by applying E1 twice, Fig 24.5.

That completes the introduction of HANDi and its transformation rules.

5 HANDi design

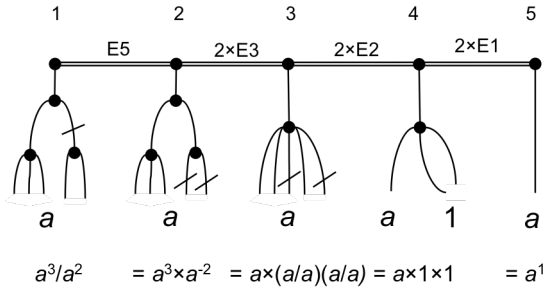


Fig. 24. Simplifying a power expression

The main motivation for the creation HANDi was to further test Representational Epistemic ideas about how to design notational systems to encode knowledge [1-3]. The core claim of this approach is that effective notational systems should directly encode the fundamental conceptual structure of their knowledge rich domains, within coherent notational schemes, and thereby possess semantic transparency that will enhance problem solving and conceptual learning [1-3]. Having presented HANDi in some detail above, this section makes explicit the design features of the new notation that attempt to achieve such a direct encoding of the knowledge of algebra.

HANDi attempts to provide a more rational codification than the conventional formula notation by coherently encoding fundamental concepts and properties of algebra. The notational schemes used to capture these concepts and properties will be considered in turn and contrasted with the conventional approach.

Representing variables and numbers, and relations among them, is obviously fundamental to algebra. HANDi uses individual sub-trees composed of a node, a trunk and branches to encode single relations among arguments. This is done for two reasons. First, a sub-tree is an explicit composite iconic symbol that stands for an operation among arguments, or other sub-trees (Fig. 5). Second, it is feasible to limit to just one instance the occurrence of letter standing for a variable in a tree. As the conventional notation (largely) relies upon the linear concatenation of symbols to capture relations among arguments, multiple occurrences of letters are often necessary, which means that the identity of arguments participating in a particular relations cannot always be read directly from the formula, so detailed examination of the arrangement of symbols in the formula is needed to determine the mathematical structure of the expression. In cognitive terms, HANDi exploits some of the well-understood benefits diagrams (e.g., [4]): information for each and every relation is co-located in the individual sub-trees of HANDi and the presence of just one letter for each variable in a tree reduces amount of deliberate search that is needed to match symbolic labels.

Relations in algebraic expression are hierarchical, so knowing the specific level of sub-expressions is essential to the correct interpretation and transformation of formulas. HANDi specifically uses the network structure of sub-trees, with nodes spatially distributed in the vertical dimension, to show the hierarchal structure. In contrast, the conventional notation makes the hierarchical structure of expressions rather opaque, because it relies upon nested parenthesis to define sub-expressions or requires the reader to mentally apply parsing rules (e.g., "BODMAS"). Both of these schemes in the conventional notational clearly demand more mental effort than the direct visual inspection of HANDi expressions (e.g., the levels of the nodes in Fig. 1 is more obvious than the levels of the expressions in the formulas below the diagram.)

The four elementary arithmetic operators possess important conceptual similarities and differences. In HANDi the respective shapes of branches, arc versus right-angle, captures the similarity between multiplication and division but distinguishes it from addition and subtraction. Bars on branches are not only used to distinguish subtraction and division from addition and multiplication, but also encode the asymmetric nature of the former two operations. The perceptive reader will have noted that no explicit rule relating to the commutative property was included among the elementary rules. The explanation for its absence is that it is built directly into HANDi by the particular design of branch shapes and bars. The benefits of this scheme reach further by providing a single definition of the circumstances in which the distribution rule holds when subtraction, division and negation operators are present, as shown in Fig. 16. Distribution is valid whenever the arc branch above the right-angle tree does not have a bar. For example, the asymmetry of trees determined by the location of the bar neatly encodes the validity of the right distribution of division but not the left distribution (i.e., $(b+c)/a$, Fig. 16.2, versus $a/(b+c)$, Fig. 16.3].

Powers and imaginary numbers extend the range algebraic operations. In the conventional formulas supplementary notation devices are built on top of the basic formulas: superscripts for powers and the 'i' symbol as the imaginary unit. Each device has it own particular set of rules. In contrast, the approach in HANDi is simpler. To encode power relations HANDi exploits the idea that powers are operations that repeat multiplication, so the hierarchical network of sub-trees in the diagrams naturally encodes repetitions of arc branches at the same level and by recursively spanning levels (Fig. 7). HANDi has a single unified scheme to deal with imaginary numbers and negative numbers, Fig. 5, 13 and 14, that builds the fundamental relations that exist among positive, negative and imaginary numbers into the design of the notation at a foundational level. In both the case of powers and imaginary numbers, HANDi does not require the introduction of unique sets of rules associated with supplementary notations.

A potential disadvantage is that HANDi expressions are more complex in terms of the sheer number of symbols, when considered at the level of nodes, branches and bars. However, with a little experience one quickly begins to read HANDi expressions at the level of sub-trees or higher, in which case its complexity is comparable to the conventional notation. Taking this notion further, one potentially significant difference between the two notations, in cognitive terms, is the extent to which HANDi may allow its users to exploit perceptual operators to recognize meaningful patterns

and to make inferences (c.f. [4]). The primary notational scheme of the conventional approach is the linear concatenation of symbols, including parenthesizes, which tends to mask characteristic configurations of symbols. The network structure and principled design of the symbols (branches, bars, turns) of HANDi aims to provide distinctive patterns to associate with particular concepts. Thus, many of the HANDi transformation rules involve spotting patterns and drawing new configurations, such as the multiple application of the E8 distribution rule in Fig 15.

It has been argued that HANDi may be a more rational encoding of algebra than the conventional formula notation. However, reactions to a new notational system are sometimes negative, for at least two reasons. First, one may initially feel that HANDi expressions are arbitrary and its rules complex compared to the existing formula notation. However, such immediate judgments should be treated with caution, because one's relative expertise in the familiar notation masks the effort required to learn the notation in the first place, which is what one is experiencing during initial encounters with HANDi. Which notation better supports learning and problem solving is an empirical question, so studies to evaluate HANDi are planned. The second common negative reaction is to think that this is not what subject is "truly" about, because algebra *is* the writing and transformation of formulas. However, this falsely assumes that a topic and its notation are inseparable, perhaps because one has only experienced algebra in the one notation, and that there is a single valid codification of a topic. The scope of HANDi expressions and rules of derivations presented here provides an existence proof that an thoroughgoing rigorous alternative codification of algebra is feasible. HANDi is not a mere visualization of the formula notion, but a generative notation that re-codifies the content of this topic This, of course, opens up wider epistemic and pedagogic questions that must, unfortunately, be considered elsewhere.

6 Acknowledgements

My thanks go to members of the Representational Systems Lab in the Department of Informatics and to Alan Blackwell for their comments on early versions of this paper. Thanks also to the three anonymous reviewers for their handy comments.

7 References

1. Cheng, P. C-H. (2002). Electrifying diagrams for learning: principles for effective representational systems. Cognitive Science, 26(6), 685-736.
2. Cheng, P. C-H. (2011). Probably good diagrams for learning: Representational epistemic re-codification of probability theory Topics in Cognitive Science 3(3), 475-498.
3. Cheng, P. C-H., & Barone, R. (2007). Representing complex problems: A representational epistemic approach. In D. H. Jonassen (Ed.), Learning to solve complex scientific problems. (pp. 97-130). Mahmah, N.J.: Lawrence Erlbaum Associates.
4. Larkin, J. H., & Simon, H. A. (1987). Why a diagram is (sometimes) worth ten thousand words. *Cognitive Science, 11*, 65-99.