Relational Constructions

John Cartmell

July 10, 2014

1 Relational Constructions

In mental operation and in everyday speech we have recourse to constructions in which relationship terms are refined or composed to enable identification of one or more entities from the context of another. For example, relationships are strung together as in phrases like *his father's cousin* or *my child's school's head teacher*; they are narrowed in scope, as in the definition of *sister* as *female sibling* and they are combined or aggregated such as in the definition of *parent* as *mother or father*. In technical contexts, programmers using high level programming languages have use of formal equivalents of these as means of expression of program logic and mathematicians too have their notational equivalents. In this section we are going to describe the significance of these constructions to entity modelling and to present a way of visualising them in diagrams.

2 Translational Equivalents

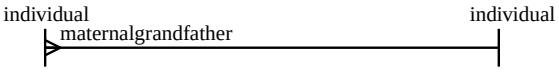
In some cases a single term exists that can be used in place of a composite relational construction; kinship terms provide a rich source of examples: *grandfather* substitutes for *male grandparent*, aunt for *parent's sister* and so on. Linguists would say in such cases that the single term and the composite phrase were translational equivalents.

Different languages vary with respect to the common relational constructions for which there are single term translational equivalents. For example,from the realm of temporal relationships rather those of kinship, in the Alamblak language of Papua New Guinea there is a word *yuanane* whose meaning is two days removed from the present which is to say its English translational equivalent is *the day before yesterday or day after tomorrow*. There is of course no single term in English which is translationally equivalent.

In regard to kinship terms, the English language is poorly endowed compared to many other languages and in English we need rely on explicit constructions such as elder brother, maternal uncle, wife's brother, cousin on mother's side to articulate relationships for which there are single word translational equivalents in other languages. For example, in the Sudanese kinship system also know as the descriptive system and into which category falls ancient Latin there is a specific term for all of the close family relationships. In Latin for example *amitinus* is father's sister's son, *patruelis* is father's brother's son, *matruelis* is mother's sister's son and, completing the male cousins, *consobrinus* is mother's brother's son.

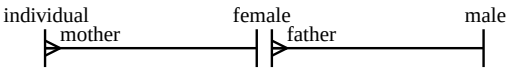
3 The Join Construction

In the definition of ‘maternal grandfather’ as ‘mother’s father’ the definiendum is a relationship between invididuals:



(1)

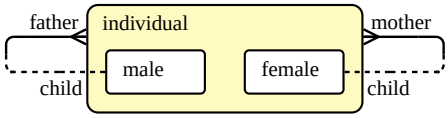
whereas the definiens is a phrasing which joins two relationships end to end:



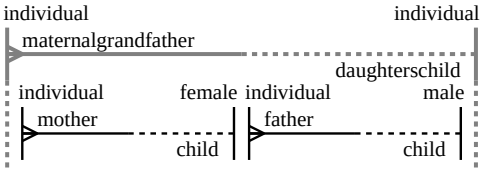
(2)

The definition presupposes types as shown in figure 1.

Figure 1 – The core family relationships

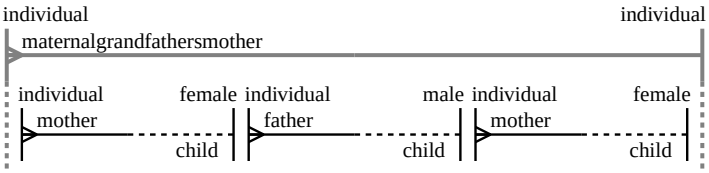


This construction by which a new relationship is defined from two others joined end to end is known simply as a *join* and the constructed relationship, the definiendum ‘maternal grandfather’, is said to be a join of the other two - in this case the join of the ‘mother’ relationship with the ‘father’ relationship. In definitions like this we shall put together definiendum and definiens is a single diagram:



(3)

There is no limit to the numbers of relationships which can be joined together, for example:



(4)

3.1 Expressing a Join in Mathematical Notation

Being as both the *mother* and *father* relationships are many-one relationships then they can be classified as mathematical functions; the relational join in this special case is precisely the mathematical operation of functional composition. In mathematical notation the functional

composition, written $f \circ g$, of two functions f and g is defined by

$$x \xrightarrow{f \circ g} g(f(x)) \tag{5}$$

and so we can write:

$$maternalgrandfather = mother \circ father \tag{6}$$

and

$$maternalgrandfather'smother = mother \circ father \circ mother \tag{7}$$

3.2 Join as Navigation

In computer software the / operation is often used in the definiens in these situations giving a construction which is sometimes described as a path or a navigation:

$$mother/father/mother \tag{8}$$

The words navigate and navigation have come to be used in recent years in describing the activity of following links through the internet or through a web site and this usage has been entered into modern dictionaries. Pre-internet the words had widespread use among programmers describing the behaviour of programs moving along links between data entities. Immersed in the contexts of their programs, programmers will say I navigate here and I navigate their meaning that they program their programs to do this and this was echoed as earlier as 1960 by C.W.Bachman writing about the structural representations of data in that he called his paper "'The programmer as navigator"'. Family relationships again are a good source of examples. Suppose a program had to extract details of a person's maternal grandfather's mother. The programmer might say that they navigated:

1. to the person's mother
2. to her father
3. to his mother

This is of course to follow the join construction from left to right in the definiens of 3 above.

A concept which has been used in theoretical contexts but not more widely is that of logical horizon (a term found in Kant's Critique of Pure Reason'??). The logical horizon of an entity is the set of entities which can be reached from it by navigation of single valued relationships. Thus father, mother, their fathers and mothers and so on to all ancestors are said to be with in the logical horizon of a particular subject person entity.

Constructed relationships such as the joins that we have seen in (3) and (4) above encapsulate possible navigations. This gives us another way of expressing what the logical horizon of an

entity is - we can define it to be the set of entities which can be reached by way of navigation of a single valued constructed relationship.

Whereas constructed relationships encapsulate directions for navigation from subject entities a set of directions such as:

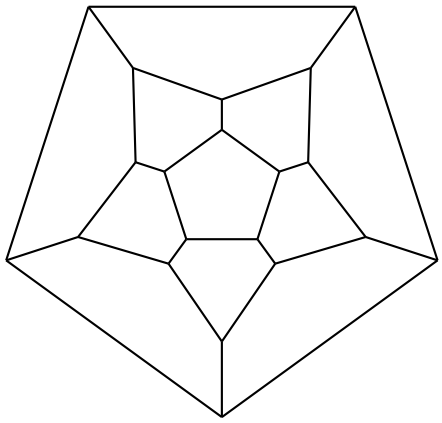
1. take the first left
2. when you get to the next lights turn right
3. now take the second right

equally can be seen as a relational construction between one subject entity which include place and direction to another place approached from a specific direction.

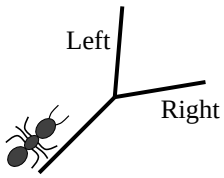
3.3 Navigating a Docdecahedron

Irish mathematician William Hamilton, suggested the game of searching for a circuit around the graph formed by the corners and edges of a dodecahedron which visited all nodes of the graph exactly once. Such a circuit around any graph became know as a 'Hamiltonian Circuit'. In a paper in 1856 he described a system of symbolic expressions every one of which:

> *... may be interpreted, as having reference to the passage from ... from corner to corner... of the dodecahedron...* [1]



We need imagine the directions to an ant facing along one of the edges of the graph of the dodecahedron. The ant is approaching a junction and can turn Left or Right.

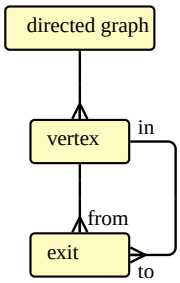


Our directions to the ant will be a series of Left Right instructions. The following series of instructions will take the ant around all the vertices of the graph:

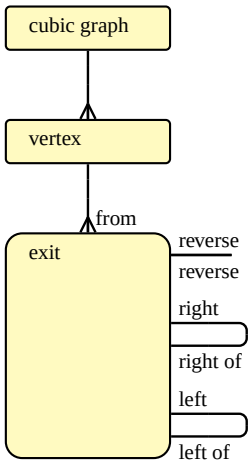
[1] For simplicity I have used ellipsis to hide his primary remark regarding navigation through the faces of an icosahedron and foregrounded his parenthetical and dual statement regarding the corners of the dodecahedron.

```
Left
Right
Left
etc
```

Each of these directions is a relationship between one position of the ant and another. We can model any graph like this: We can extend this model of a graph by a model which shows the possible positions and direction of the ant:



The directions left and right are in fact relationships between one possible position and another:



Each series of directions is a relationships constructed by join from the core Left and Right relationships in this model. Hamilton's symbolic expressions can be seen both as navigations and also as relational joins. What he was interested in was the fact the algebra that resulted. For the ant to turn around he needs to complete a circuit of one of the faces and exit. This can be achieved in a clockwise direction:

```
Left / Right / Right / Right /  Right / Left
```

or anti-clockwise

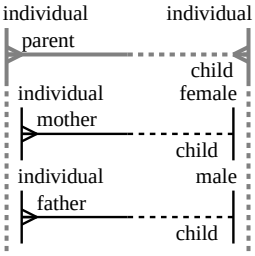
```
Right / Left / Left / Left / Left / Right
```

Turning around twice is do do nothing at all and so an equation results:

```
Right / Left / Left / Left / Left / Right / Right / Left / Left / Left / Left / Right = Iden
```

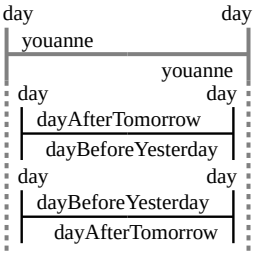
4 The Aggregation of Relationships

In contrast to the definition of ‘maternal grandfather’, we can define the ‘parent’ relationship to be the *aggregate* of the ‘mother’ and ‘father’ relationships. The definiens is now ‘mother’ *and* ‘father’ instead of being ‘mother’ *then* ‘father’ In this case the two relationships, ‘father’ and ‘mother’ are put together side by side and the whole definition can be viewed like this:



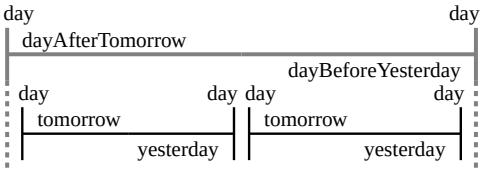
(9)

‘A day two days removed from the present’, or *yuanane*, is also an aggregation since it is ‘day before yesterday’ aggregated with ‘day after tomorrow’ :



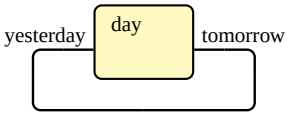
(10)

whereas ‘day before yesterday’ and ‘day after tomorrow’ are both joins:



(11)

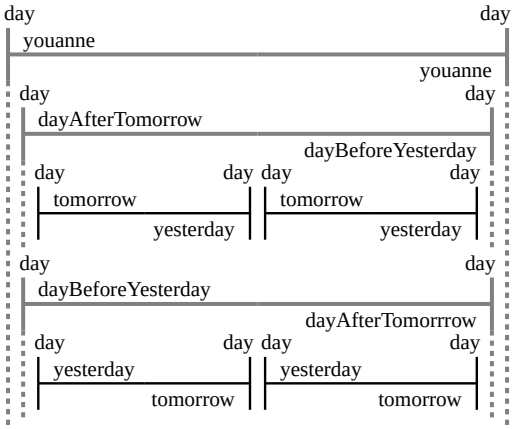
‘Tomorrow’ and ‘yesterday’ are different ends of the one binary relationship of succession between days:



Joining ‘tomorrow’ with ‘tomorrow’ is equally joining ‘yesterday’ with ‘yesterday’. Consequently the *yuanane* relationship is it’s own inverse, that is it is a symetric relationship:

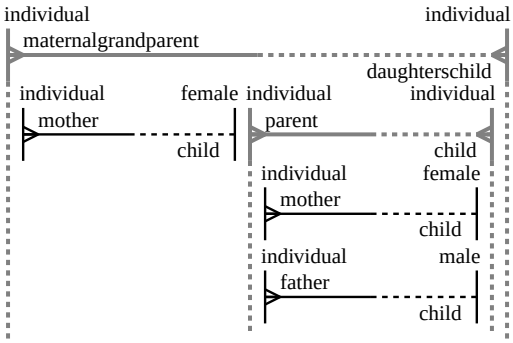
By expanding the definiens in the definition of *yuanane* and by also adding labels for all

inverse relationships we get a single diagram:



(12)

If either of the relationships in a join is many-valued then the resulting relationship is many-valued. Thus maternal grandparent is many-valued because parent is:



(13)

The existence of translational equivalents poses a particular danger to entity modellers as we now discuss.

5 Goodness of an Entity Model

Though we often have alternate, sometime conflicting, ways of seeing and articulating 'what is' in the world, it is the purpose of an entity model to present a single, self-consistent set of concepts that is minimal for a particular domain and, in a well-defined way that is to be explained here, does not introduce relationships needlessly.

For example, for minimally expressing, 'what is' in kinship, the *grandparent* relationship is needless - it should be omitted in favour of the *parent-child* relationship from which, as parent's parent, it can be derived. So it is with *brother* as *parent's male child*, *uncle* as *parent's brother* i.e. *parent's parent's male child*, and likewise *sister* and *aunt*, *grandfather*, *cousin* - all these relationships can be constructed from the *parent-child* relationship and the *male* and *female* typing of individuals. These latter relationships form a core, as shown in figure 1, from which the others can be derived.

We use the terms derived and constructed interchangeably and we need to be clear what it means to say that the relationship *grandparent* can be derived or constructed from the *parent-child* relationship - what it means is that if one party communicates to another (i) a set of individuals and (ii) the parent-child relationships between them then nothing new is added to this by a further communication of (iii) the grandparent-grandchild relationships between the same set of individuals.

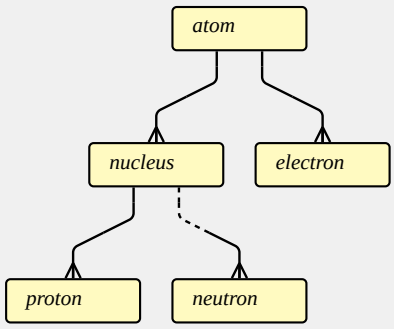
We say that the grandparent-grandchild relationship can be constructed from the parent-child relationship but note that what we actually mean by this is that in any given situation information about who is whose grand parent can be constructed from information about who is whose parent.

6 The Golden Rule

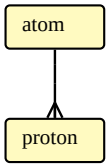
Inferrable relationships such as *grandparent-grandchild* one are said to be *constructed* and need to be contrasted with the *core* relationships from which they can be constructed. The golden rule of entity modelling is that it should not be possible to construct any of the relationships within an entity model from combinations of the others.

The golden rule is a prerequisite for a goodness criteria within database design technically referred to as ‘normal form’. Rather than just accept this condition we must understand what is achieved by meeting the condition; to do so it is helpful to keep in mind that meeting the condition and coming up with a good entity model requires a skill and that this skill is, to all intents and purposes, the ancient Euclidean skill of good theorizing by which I mean the articulation of a minimum set of assumptions or givens and their separation from that which can be derived or inferred from them. For our present purpose good modelling is good theorising and to borrow from Occam’s dictum it is *not multiplying relationships needlessly*.

Figure 2 – Atomic Structure



6.1 Illustration 1

Atoms are related to protons (and neutrons too for that matter), for each atom contains one or more protons, but the relationship 

was not represented on the model of figure 2 nor should it have been for to do so would have broken the golden rule since this is a relationship which can be constructed or inferred from the ones already present. The construction can be summarised thus:
From

(a) every atom contains exactly one nucleus

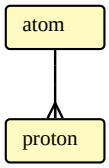
and

(b) every nucleus contains one or more protons

derive

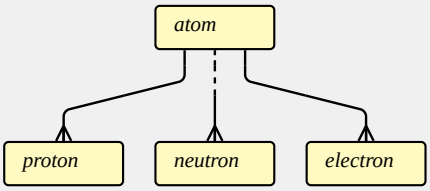
(c) every atom contains one or more protons.

This represents the inference of a new relationship from old and is a further example of a join construction:

What is illustrated here is a general rule that if ‘every A has parts of type B’ and ‘every B has parts of type C’ then ‘every A has parts of type C’. This is precisely the relationship join construction applied to a pair of many-valued composition relationships. The rub is that the relationship 

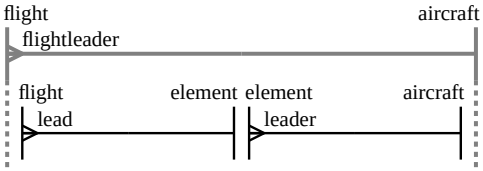
can appear in a core model - it is just that it cannot appear in the model of figure 2. Figure 3 illustrates this. This is a model which does contain the relationship but which does not does not break the golden rule. The model in figure 3 and is appropriate in modelling situations where the localisation of protons and neutrons to the nucleus is irrelevant to the task at hand. There is nothing necessarily inherent, therefore, in the distinction between core and constructed - in one model a relationship may be core while in a different model it may be constructed.

Figure 3 – Atomic Structure



6.2 Illustration 2

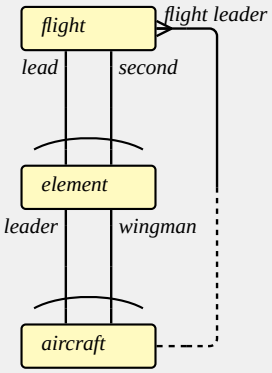
Consider the model of a four-finger flight formation in figure 4. All is well with this model until examination of the subject matter reveals that in all cases the flight leader is the leader of the lead element of the flight. This is to say that 'flight leader' is a constructed relationship:



(14)

With this additional information the model of figure 4 is seen to break the golden rule by representing the *flight leader* relationship unnecessarily.

Figure 4 – The Four-Finger Flight Formation



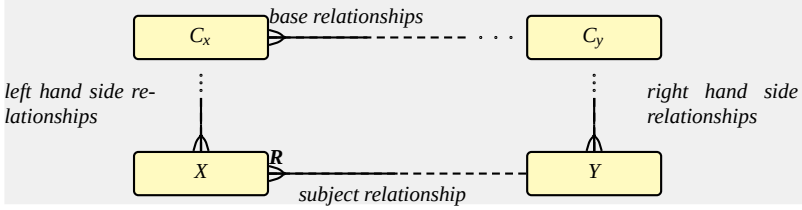
7 Finally

The main purpose of an entity model is to describe core relationships and it is wrong to include constructed relationships in an entity model and so to confuse them with the core. Depending on the task at hand, there may be interest and purpose in describing constructed relationships but the constructed relationships must not be included in the core.

8 Interpretation of Scope Squares

The scope squares of figures **??** and **??** are actually formed from four relationships. Figure **??** is a little different it has two relationships down the left hand side. Figure **??** is different again it has two relationships along the base. More generally, as shown in figure5 there may be a sequence of relationships forming the left hand entity type with context (an enclosing part of an enclosing part of an enclosing part, some number of times) and a sequence of relationships gives context to the right hand side. Along the base joining the left and right context types is a sequence of reference relationships.

Figure 5 – The general case.



Such a diagram, if it is a scope constraint, is a constraint of the relationship R. As a scope constraint it asserts that following

> if x of type X is in relationship R with y of type Y then the context c_x of x at level C_x is related through the sequence of base relationships to the context c_y of y at level C_y.

Mathematicians have a simpler way of saying this - they simply say that the diagram *commutes*.

Relational algebraists may say that the join of relationships followed clockwise is relationally identical to the join of relationships followed anti-clockwise.