Substitution into Arithmetic Expressions

We describe an alternative notion of proof, called a substitution proof, for arithmetic equations.

Figure 1 in Notes 1: Proving Arithmetic Equations, in the Readings section, listed a formal proof consisting of a sequence of equations and ending with the equation

$$(f + g) + -g = f.$$

Another style of proof for this equation is the sequence of equalities

$$(f + g) + -g = f + (g + -g) = f + 0 = 0 + f = f.$$

Here the first equality follows from (associativity), the second from (inverse for +), and the third and fourth by (symmetry) and (identity for +), respectively. Proofs in this format are attractive and easy to follow because each equality follows solely from the immediately previous equality by **substituting equals for equals** -- in contrast to the sequence-of-equations proofs of the Notes, where justification of an equation may require appeal to more than one antecedent.

Here is a precise definition of such substitution proofs:

Definition. An arithmetic **substitution proof** is a sequence of ae's, e_i, separated by equality symbols:

$$e_1 = e_2 = e_3 = \ldots = e_n$$

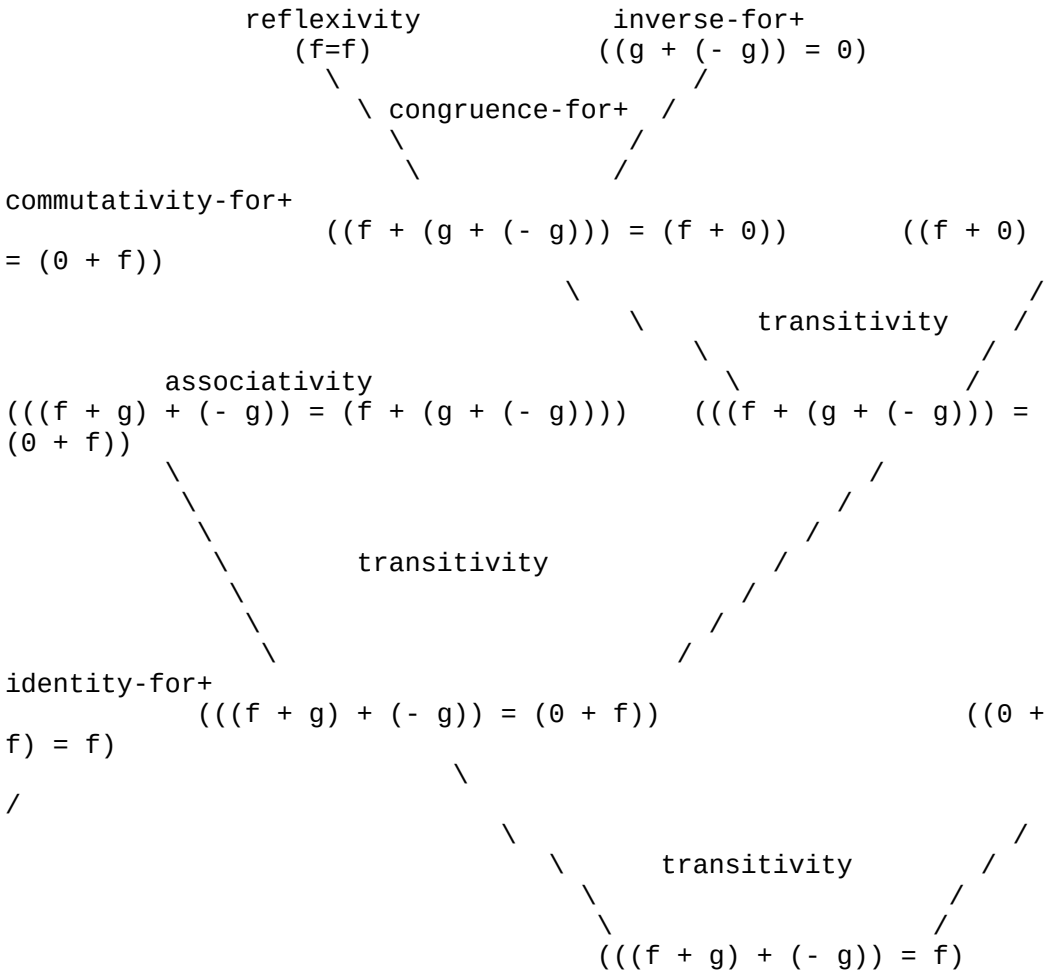
such that for $1 \leq i < n$, the expression e_{i+1} is obtained from e_i by *substituting equals for equals*. Namely there is an ae, h_i, with exactly one occurrence of some variable, x, and there is an axiom instance, $(f_i = g_i)$, such that the expressions e_i and e_{i+1} are the same as the expressions $h_i[x:=f_i]$ and $h_i[x:=g_i]$ (in either order). The equation proved by the proof is $(e_1 = e_n)$.

1. Define a Scheme procedure

   `tree-proof-->subst-proof`

   that transforms a tree proof corresponding to the recursive definition of *Arithmetic Equational Theorems* (aet's) given in the Notes into a substitution proof of the same equation. For input to the procedure, the tree proof will be represented in the usual way as a Scheme list structure (see below).

   In designing the Scheme procedure `tree-proof-->subst-proof`, we assume for simplicity that an equational proof as defined in the Notes will be represented as a tree showing the antecedents and rule used to arrive at each equation in the proof. For example, the proof in Fig.1 of the Notes, would be represented in tree form as:



This tree would then be represented as a Scheme list structure. Namely, the proof is represented as a list of the equation proved, a token indicating the inference rule by which it was deduced, and, if the inference rule was not an axiom, the lists representing the proofs of the antecedents of the inference rule. So the representation of the proof tree above would be:

```
(define Fig1-tree-proof
  '((((f + g) + (- g)) = f)
      transitivity
      ((((f + g) + (- g)) = (0 + f))
         transitivity
         ((((f + g) + (- g)) = (f + (g + (- g)))) associativity-
of+)
         (((f + (g + (- g))) = (0 + f))
            transitivity
            (((f + (g + (- g))) = (f + 0))
                congruence-for+
                ((f = f) reflexivity)
                (((g + (- g)) = 0) inverse-for+))
            (((f + 0) = (0 + f)) commutativity-for+)))
      (((0 + f) = f) identity-for+)))
```

Now applying the procedure to this input might produce the following printout:

```
(tree-proof-->subst-proof Fig1-tree-proof)
;Value: (((f + g) + (- g)) = (f + (g + (- g))) = (f + 0) = (0 +
f) = f)
```

This proof-transforming procedure shows that the class of equations provable by substitution proofs is at least as large as the equations provable using the tree proofs for aet's. Since we've already argued that the sequence-of-equations proof system is *complete* and that the tree-proofs are interconvertible with sequence-of-equations proofs, we can conclude that the substitution proof system indeed also proves all the valid equations. But since substitution proofs are *sound* (see below), it follows that they prove exactly the valid equations. That is, substitution proofs are another complete proof system for arithmetic equations.

2. The Substitution Lemma below relates the value of an expression obtained by substituting f for x in e to the values of f and e. Namely, suppose $F{:}A \to B$, that is, F is a total function with domain A and codomain B. For any a in A and b in B, we define

$$F[a \leftarrow b]$$

to be the function $G{:}A \to B$ such that

$$G(a) = b, \text{ and } G(x) = F(x) \text{ for } x \neq a.$$

Substitution Lemma: For all valuations, V,

$$val(e[x{:=}f], V) = val(e, V[x \leftarrow val(f, V)]).$$

Prove it!

3. Consider the following variations of the substitution rules for equality, adapted for inequalities:
    1. (f ≤ g) => => (e[x:=f] ≤ e[x:=g])
    2. (f ≤ g) => => (f[x:=e] ≤ g[x:=e])

Only one of these rules preserves validity. Give a simple example of arithmetic expressions *e,f,g* where one of these rules fails to preserve validity. Prove that the other rule does preserve it.