Massachusetts Institute of Technology
6.844, Spring '03: Computability Theory of and with Scheme
Prof. Albert Meyer

Course Notes 2
February 24
revised February 28, 2003, 641 minutes

Substitution into Arithmetic Equations

1 Proof of the Substitution Lemma

Lemma (Substitution).

$$[\![e[x := f]]\!]V = [\![e]\!](V[x \leftarrow [\![f]\!]V]),$$

where for any function, F, and elements a, b, we define $F[a \leftarrow b]$ to be the function G such that

$$G(u) = \begin{cases} b & \text{if } u = a. \\ F(u) & \text{otherwise.} \end{cases}$$

A standard technique for proving things about inductively defined objects like arithmetic expressions is to use *structural induction*. This is an inductive proof in which the base cases are those of the definition – for ae's this means base cases considering expressions that are *constants* and expressions that are *variables*. The induction cases correspond to the clauses of the induction definition where new objects are defined from prior ones—for ae's this means cases cases for sums, products, and negations of ae's. In proving the induction cases, the hypothesis to be proved may be *assumed*—by induction – to hold for the constituents of an object. For example, to prove the hypothesis holds in the case of a sum of ae's, we may assume by induction that the hypothesis holds for each of the two summands.

Structural induction could be justified as a special case of induction on the size of ae's, but when a proof is organized in this way, it's clearer to describe it as proof by structural induction.

For this problem there are two ae's, e and f on which structural induction could conceivably be based, but structural induction on e is the one that works nicely:

Proof. Letting V' be the valuation $V[x \leftarrow [\![f]\!]V]$ and e' be the expression $e[x := f]$, we prove that $[\![e']\!]V = [\![e]\!]V'$ by structural induction on the arithmetic expression e.

Base case 1: e is a constant, c, namely, `0` or `1`. So by the definition of valuation, the value of e is the number 0 (respectively, 1) in all valuations. Also, by the definition of substitution, e' is still the same constant, c. So, the value of e' is also 0 (resp., 1) in all valuations, and in particular, $[\![e']\!]V = [\![e]\!]V'$.

Base case 2: e is a variable, y, distinct from x. Then by the definition of substitution, e' is also the variable y. So, the value of e' in V is $V(y)$ and the value of e in V' is $V'(y)$. But since x is distinct from y, we have $V'(y) = V(y)$ by definition of V'. So $[\![e']\!]V = V(y) = V'(y) = [\![e]\!]V'$.

Copyright © 2003, Prof. Albert Meyer.

Base case 3: e is the variable x. Then by the definition of substitution, e' is the expression f. Also, $[\![f]\!]V = V'(x)$ by definition of V'. So,

$$[\![e']\!]V = [\![f]\!]V = V'(x) = [\![x]\!]V' = [\![e]\!]V'.$$

Structural induction case 1: e is of the form $e_1 + e_2$. By induction we may assume $[\![e'_i]\!]V = [\![e_i]\!]V'$ for $i = 1, 2$.

Now by the definition of substitution, e' is the same as $e'_1 + e'_2$. Also, by the definition of the value of expressions, the sum of the values of two expressions is the value of the "+"-expression made from the two expressions. So

$$[\![e']\!]V = [\![e'_1 + e'_2]\!]V = [\![e'_1]\!]V + [\![e'_2]\!]V = [\![e_1]\!]V' + [\![e_2]\!]V' = [\![e_1 + e_2]\!]V' = [\![e]\!]V'$$

Remaining Structural induction cases: Similar to case 1.

□

2 Substitution into Inequalities

Note that $\models (e <= f)$ does *not* imply $\models (g[x := e] <= g[x := f])$. Just consider the case when g is $-x$, e is 3 and f is 4: we would conclude that since $\models 3 \leq 4$, also $\models -3 \leq -4$! On the other hand, we can safely substitute *into* inequalities:

Lemma 2.1.

$$\textit{If } \models (e <= f) \textit{ then } \models (e[x := g] <= f[x := g]).$$

Proof. The hypothesis $\models (e <= f)$ means that $[\![e]\!]W \leq [\![f]\!]W$ for every valuation W. Combining this inequality with the Subsitution Lemma, we have

$$[\![e[x := g]]\!]V = [\![e]\!]V' \leq [\![f]\!]V' = [\![f[x := g]]\!]V.$$

Since V was arbitrary, we conclude that for *all* valuations V, $[\![e[x := g]]\!]V \leq [\![f[x := g]]\!]V$, that is, $\models (e[x := g] <= f[x := g])$.

□

3 Substitution as a Derived Rule

Lemma 3.1.

$$\textit{If } \vdash (f = g), \textit{ then } \vdash (e[x := f] = e[x := g]).$$

Proof. The proof is by structural induction on e.

Base case 1: e is a constant, c, or variable distinct from x. Then by the definition of substitution, $e[x := f]$ is actually the arithmetic expression c. Likewise, $e[x := g]$ is c. So $(e[x := f] = e[x := g])$ is actually the equation $(c = c)$, which is immediately provable by reflexivity Note that in this base case, we did not need to use the hypothesis $\vdash (f = g)$.

We could have said "$c = e[x := f]$" in the argument above, but because there are so many kinds of equality around— provable equality in our formal system, syntactic equality (*i.e.*, identical expressions), equality by definition, and equality of meaning—we were careful to say instead "$e[x := f]$ *is actually* the arithmetic expression c."

In handling this case, students sometimes fall into the following trap: they argue that since

1. $c = e[x := f]$ and
2. $c = e[x := g]$, it follows that
3. $(e[x := f] = e[x := g])$ is provable *by transitivity and symmetry*. (NO!)

This argument confuses our *mathematical proof* with a *formal proof in our system*. We proved that "$e[x := f]$" and "$e[x := g]$" describe the same constant expression, c, implicitly using the transitivity and symmetry properties of "sameness." On the other hand, the formal proof of the equation $(c = c)$ follows from reflexivity, not symmetry or transitivity.

Base case 2: e is the variable x. Then by the definition of substitution, $e[x := f]$ is the expression f. Likewise $e[x := g]$ is g. So this base case requires that we show that $(f = g)$ is provable, that is, $\vdash (f = g)$. But this is exactly the given hypothesis, so we are done.

Structural induction case 1: e is of the form $e_1 + e_2$. Assume $\vdash (f = g)$. We must now show that

$$\vdash ((e_1 + e_2)[x := f] = (e_1 + e_2)[x := g]). \tag{1}$$

Now by induction, we may assume that

$$\vdash (e_1[x := f] = e_1[x := g]),$$
$$\vdash (e_2[x := f] = e_2[x := g]).$$

By +-congruence from these two equations, we have

$$\vdash (e_1[x := f] + e_2[x := f] = e_1[x := g] + e_2[x := g]). \tag{2}$$

But by definition of substitution, the lefthand side of (2) describes the same expression as the lefthand side of (1), and likewise for the righthand sides of (2) and (1), so we are done.

Structural induction cases 2 & 3: e is of the form $e_1 * e_2$ or $-e_1$. Essentially the same as case 1.

□

Note that we could also have proved Lemma 3.1 from Lemma 2.1 by appealing to soundness and completeness of the equational proof system. But the proof by structural induction does not require completeness, and so will generalize to algrebraic number systems where completeness does not hold.

Lemma 3.2.

$$\vdash (e = f) \quad \textit{implies} \quad \vdash (e[x := g] = f[x := g]),$$
$$\vdash (e <= f) \quad \textit{implies} \quad \vdash (e[x := g] <= f[x := g]).$$

Structural induction on formulas is not always the way to go. This time to prove Lemma 3.2 we use induction on *the length of the formal proof* of $(e = f)$ (resp. $(e <= f)$) in our inequality proof system.

Base case: The proof of $(e = f)$ (resp. $(e <= f)$) is of length 1. This means that it is an axiom. But then we claim that $(e[x := g] = f[x := g])$ (resp. $\leq$) is another instance of the same axiom, and so also has a proof of length 1.

We illustrate why the claim holds for the +-commutativity axiom, $(e_1 + e_2 = e_2 + e_1)$. Namely, we claim that the equation

$$((e_1 + e_2)[x := g] = (e_2 + e_1)[x := g]) \tag{3}$$

is also an instance of +-commutativity. To see this, let h_1 be the expression $e_1[x := g]$ and h_2 be $e_2[x := g]$. But by the definition of substitution, the expression on the lefthand side of equation (3) describes the term $h_1 + h_2$ and the righthand side describes $h_2 + h_1$, proving the claim in this case.

The proofs of the claim for the other axioms follow similarly.

Induction: The proof of $(e = f)$ (resp. $(e <= f)$) is of length $n + 1$ for some $n \geq 1$.

In this case we use the fact that all the formal inference rules of Handout 3, Tables 1 & 3, have a property similar to that claimed for the axioms. Namely, if we perform the same substitution on all the antecedents and the consequent of a rule, we obtain another instance of the same rule. The argument proving this is similar to the argument above for the commutativity axiom. We leave it to the reader to check this fact.

Now, if $(e = f)$ (resp. $\leq$) is an axiom, the proof for the Base case shows that $(e[x := g] = f[x := g])$. Otherwise, $(e = f)$ must follow from an instance of an inference rule whose antecedents are equations earlier in the formal proof. By the fact stated above, substituting g for x into the antecedents and consequent of this inference rule is another instance of the rule. But the antecedents must have proofs of length at most n, so by induction, we may assume that the result of substituting g for x in any antecedent equation is also provable. Now the equation $(e[x := g] = f[x := g])$ is the consequent of the substituted instance of the inference rule, and the substituted antecedents are provable, so we conclude that this consequent is also provable.