Supplementary Notes for the MAC-lectures October 20 - 27

October 24, 2003

On the definition of the substitution operator

In Chapter 7 the value of the term $[\ \langle \mathcal{A}|\ x := \mathcal{B}\ \rangle\ ]^{\cdot}$ is only defined in a *special case*, viz. when $[\ \mathcal{B}\ ]^{\cdot}$ is '*free for*' $[\ x\ ]^{\cdot}$ in $[\ \mathcal{A}\ ]^{\cdot}$:

> The value of $[\ \langle \mathcal{A}|\ x := \mathcal{B}\ \rangle\ ]^{\cdot}$ is found by substituting all free occurences of $[\ x\ ]^{\cdot}$ in $[\ \mathcal{A}\ ]^{\cdot}$ by $[\ \mathcal{B}\ ]^{\cdot}$ **if $[\ \mathcal{B}\ ]^{\cdot}$ is free for $[\ x\ ]^{\cdot}$ in $[\ \mathcal{A}\ ]^{\cdot}$**. (cf. page 217)

This gives rise to four questions:

1) How do we find the free occurences of $[\ x\ ]^{\cdot}$ in $[\ \mathcal{A}\ ]^{\cdot}$?
Answer: By drawing a binding diagram, e.g. by means of the syntax tree for $[\ \mathcal{A}\ ]^{\cdot}$.

2) How do we substitute the free occurences of $[\ x\ ]^{\cdot}$ in $[\ \mathcal{A}\ ]^{\cdot}$ by $[\ \mathcal{B}\ ]^{\cdot}$?
Answer: Since there are invisible parenthesis around each occurence of $[\ x\ ]^{\cdot}$ in $[\ \mathcal{A}\ ]^{\cdot}$, the section *Correct substitution of equals* in 'Extended Summary of the MAC-Lecture September 1' tells us, that (if $[\ \mathcal{B}\ ]^{\cdot}$ is free for $[\ x\ ]^{\cdot}$ in $[\ \mathcal{A}\ ]^{\cdot}$) we may substitute $[\ x\ ]^{\cdot}$ by $[\ \mathcal{B}\ ]^{\cdot}$ *enclosed by parenthesis*, and only remove the parentheses if they are superfluous.
For example:

$$[\ \langle\ x + 2\cdot x + x^{+} + x\ |\ x := 1 + y\ \rangle \equiv\ 1 + y + 2\cdot(1 + y) + (1 + y)^{+} + (1 + y)\ ]$$

3) What does it mean that $[\ \mathcal{B}\ ]^{\cdot}$ should be free for $[\ x\ ]^{\cdot}$ in $[\ \mathcal{A}\ ]^{\cdot}$?
Answer: As described on page 217 it means that *no* free variable in $[\ \mathcal{B}\ ]^{\cdot}$ may become bound by the substitution in **2)**. A direct translation of 'free for' into the Danish 'fri for' may seem rather misleading, since it is *not* the question whether the variable $[\ x\ ]^{\cdot}$ occurs in $[\ \mathcal{B}\ ]^{\cdot}$ or not! The phrase 'free as', i.e. 'fri (når $[\ \mathcal{B}\ ]^{\cdot}$ optræder) som' might have been more appropriate, but unfortunately 'free for' is the standard terminology found in the literature.

4) How do we find the value of $[\ \langle \mathcal{A}|\ x := \mathcal{B}\ \rangle\ ]^{\cdot}$ when $[\ \mathcal{B}\ ]^{\cdot}$ is <u>not</u> free for $[\ x\ ]^{\cdot}$ in $[\ \mathcal{A}\ ]^{\cdot}$?
Answer: A free variable in $[\ \mathcal{B}\ ]^{\cdot}$ can only become bound by the substitution in **2)** if that variable has the same name as a *binding* variable i $[\ \mathcal{A}\ ]^{\cdot}$ (Please think this over!) Hence, the problem does not occur in a *simple term* (where all binding variables are distinct and also distinct from any free variable). *Simplification* of the term $[\ \langle \mathcal{A}|\ x := \mathcal{B}\ \rangle\ ]^{\cdot}$ can always be obtained by renaming the **<u>binding</u>** (and any bound) variables (i.e not only the bound variables, cf. the errata list), and thus we may always obtain a simple term, where the substitution in **2)** is allowed.
For example: ($[\ \mathcal{A} \equiv \langle\ x\ |\ y := 2\ \rangle\ ]^{\cdot}$ and $[\ \mathcal{B} \equiv y\ ]^{\cdot}$):

$$\begin{aligned}[\ \langle \mathcal{A}|\ x := \mathcal{B}\ \rangle\ &\equiv\ \langle\ \langle\ x\ |\ y := 2\ \rangle\ |\ x := y\ \rangle\\ &\equiv\ \langle\ \langle\ v\ |\ u := 2\ \rangle\ |\ v := y\ \rangle \equiv \langle\ y\ |\ u := 2\ \rangle \equiv y\ ]\end{aligned}$$

due to simplification and substitution of any free occurences of the binding variables.

On Chapter 8: Functions

In Chapter 8 **two binary operators** and **a new type of values of nullary operators** are introduced. The two new constructs (which are operators, although this is only specified implicitly in the chapter) are:

a) The application operator : $\mathcal{A}$ ' $\mathcal{B}$ (i.e. the term $[\ \mathcal{A}\ ]^{\cdot}$ applied to the term $[\ \mathcal{B}\ ]^{\cdot}$)
b) The lambda operator: λx.$\mathcal{A}$ (i.e. 'lambda x dot $\mathcal{A}$', where x is a variable and $\mathcal{A}$ a term)

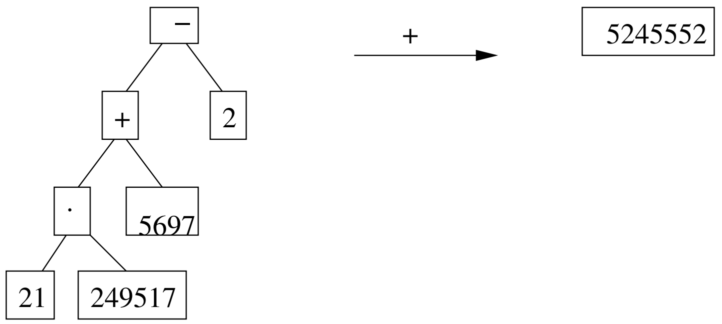
Since $[\ \mathcal{A}$ ' $\mathcal{B}\ ]^{\cdot}$ and $[\ \lambda x.\mathcal{A}\ ]^{\cdot}$ are syntactically correct compositions of mere operators, they are **terms** and thus have a value if all the free variables in $[\ \mathcal{A}\ ]^{\cdot}$ and $[\ \mathcal{B}\ ]^{\cdot}$ have one.
What the value of terms such as $[\ \mathrm{T}$ ' $\perp\ ]^{\cdot}$ or $[\ (1 :: 5)$ ' $(2{\cdot}x + 4)\ ]^{\cdot}$ (for $x \equiv 1$, say) are, is, however, not revealed until Chapter 9, but all terms *do* have a value when any free variable is given one!!

Until now all values of nullary operators have *either* been $[\ \perp\ ]^{\cdot}$ *or* a value belonging to one of the sets in the type tables of Volume 3 (e.g. **B**, **D**, **X** or **E**). When we have drawed syntax trees, the **leaves of the syntax trees** were **nullary operators** with such values, and we have not distinguished between these nullary operators and **their values**. Hence, we have claimed that e.g. numbers are nullary operators, but in fact it is only the *value* of certain nullary operators that may be numbers!
The reason is that 'numbers' is a **semantic** concept (the 'meaning/value' of the operator $[\ 2\ ]^{\cdot}$ is a certain number), whereas 'operators' is a **syntactic** concept telling us e.g. how many arguments the operator need (nullary, unary, binary, ternary, n-ary operators) and where to put the operator (prefix, infix, suffix, outfix, 'none') in order to write a term.

When the term consists of more than one operator, the difference between syntax and semantics may become more clear. Consider e.g. the expression

$$[21 \cdot 249517 + 5697 - 2]^{\cdot}$$

Most people will consider this as being a term (with a certain value), whereas very few will consider it to be the number 5245552 (at least until they find the *value* of the term!).
When we draw the *syntax* tree of the term and perform graph reductions:



we thus see how a term composed of 3 binary operators and 4 nullary operators are reduced to a term consisting of only one nullary operator. Now it is much more easy to find the value, since the value of the first term equals the *value* of the last one. In other words: the latter term is almost *transparent* in the sense that the semantics/meaning to which it *refers* is seen immediately by those familiar with the base-10 number system. Hopefully, you also find the lecture notes and the supplementary notes **referential transparent** instead of wondering about the syntax (i.e. the composition of the single characters and words) in the English texts!

Consider now the expression

$$[f_3(\mathrm{x}) \doteq 2 \cdot \mathrm{x} + 4]^{\cdot}$$

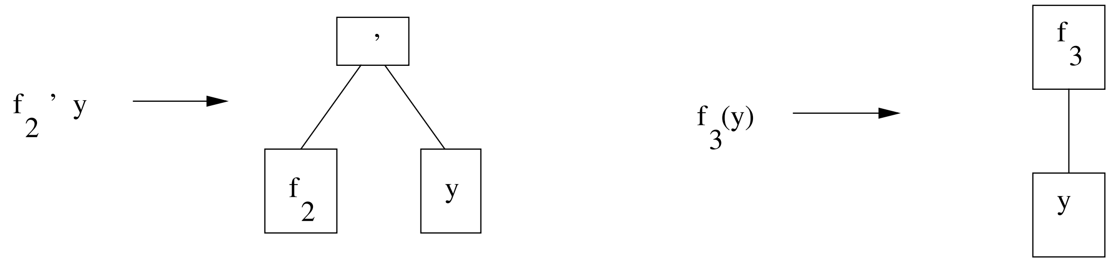
Since the expression contains the directive $[\doteq]^{o}$ we no longer have a term, but a statement defining the **unary** operator $[\ f_3\ ]^{o}$. Since $[\ f_3\ ]^{o}$ is no nullary operator it does not have a value (contrary to $[\ f_3(\mathrm{x})\ ]^{\cdot}$ for some value of $[\ \mathrm{x}\ ]^{\cdot}$), and thus $[\ f_3\ ]^{o}$ can never be **input** argument to any operator *nor* the **output** result from one.
Contrary to this

$$[f_2 \doteq \lambda\mathrm{x}.\ 2 \cdot \mathrm{x} + 4]^{\cdot}$$

is a statement defining the **nullary** operator $[\ f_2\ ]^{\cdot}$, whose **value** is a **function** which may be passed as input value to an operator or be obtained as the output value from one.
Due to

$$[\ \textbf{Mac rule ApplyLambda} :\ (\lambda\mathrm{x}.\mathcal{A})'\mathcal{S} \equiv \langle\mathcal{A}|\mathrm{x} := \mathcal{S}\rangle\ ]$$

the *values* of $[f_2\ '\ \mathrm{y}]^{\cdot}$ is the same as the *values* of $[\ f_3(\mathrm{y})\ ]^{\cdot}$, but the syntax trees show a difference in syntax, since the arity of the two operators $[\ f_2\ ]^{\cdot}$ and $[\ f_3\ ]^{o}$ are distinct:



Like the previous nullary operators $[\ f_2\ ]^{\cdot}$ is to be found among the leaves of the syntax tree and has a (new type of) value, whereas none of this is true for the operator $[\ f_3\ ]^{o}$.

Hence,

$[\ f_3\ ]^{o}$ is a **unary operator**, but **no term(!)**, since it has no value.
$[\ f_3(\mathrm{x})\ ]^{\cdot}$ is a term (a unary and a nullary operator). The value is the value of $[\ 2{\cdot}\mathrm{x} + 4\ ]^{\cdot}$.
$[\ 2{\cdot}\mathrm{x} + 4\ ]^{\cdot}$ is a term (two binary and three nullary operators). Due to $[\ f_3(\mathrm{x}) \equiv 2{\cdot}\mathrm{x} + 4\ ]^{\cdot}$ the value is the same as the value of $[\ f_3(\mathrm{x})\ ]^{\cdot}$.
$[\ f_2\ ]^{\cdot}$ is a term (a nullary operator). The value is the function which multiplies with 2 and adds 4.
$[\ \lambda\ \mathrm{x}.\ 2{\cdot}\mathrm{x}{+}4]^{\cdot}$ is a term (three binary and three nullary operators). Due to $[\ f_2 \equiv \lambda\ \mathrm{x}.\ 2{\cdot}\mathrm{x} + 4\ ]^{\cdot}$ the value is the same as the value of $[\ f_2\ ]^{\cdot}$.
$[\ f_2\ '\ \mathrm{x}\ ]^{\cdot}$ is a term (a binary and two nullary operators). The value is the value of $[\ 2{\cdot}\mathrm{x} + 4\ ]^{\cdot}$.

As is the case with substitution, renaming of **binding** (and any bound) variables are also important when applying (terms with values being) functions:

$$[(\lambda\mathrm{x}.\ \lambda\mathrm{y}.\ \mathrm{x})\ '\ \mathrm{y}\ '\ 2\ \equiv\ (\lambda\mathrm{v}.\ \lambda\mathrm{u}.\ \mathrm{v})\ '\ \mathrm{y}\ '\ 2\ \equiv\ \mathrm{y}]^{\cdot}$$