Some things that seem to be particularly hard to get ones head around

Jonathan Cederberg
jonathan.cederberg@it.uu.se

November 1, 2011

Abstract

This is some comments on things that seems to be particularly hard to get your head around (obviously, it says so in the title). More specifically, it is a collection of oft-asked questions in the course programming theory. Any mistakes in the below text are entirely the author fault (again, obviously), and information of any such discovered errors are highly appreciated.

1 General Hints

- Look in the compendium, use the *exact same* pattern when writing your own proofs.
- Always list the arithmetic rule you are using. This will decrease the risk that you use it incorrectly.
- Write lemmas, but do it separately and not inside of your proof. You are not allowed to use the assumptions of your proofs when proving the lemmas.
- Make sure you type your proof correctly. In particular, make sure that you are aware that "=" can denote both arithmetic equality and logical equivalence.
- Do not use rules with implication in a proof in the same way as you use rules with equivalence. They are mainly for use with conditional substitution.
- Also the converse: make sure you do not weaken the statement and treat it as if you have an equivalence. For example, $x + 3 = y$ is not equivalent to $x < y$, so you cannot have a proof with equivalences where you go from the first to the second.
- Symmetric problems should not turn asymmetric. This is a very simple sanity-check for proofs. This means that if you start out

with an expression E, and manipulate it through a series of steps to get E', then E' should not be weaker than E. This might be hard to tell, but in some cases it is clear that something "got lost along the way".

- Be careful with disjunctions in assumptions. You have to assume the whole disjunction, not the parts individually, as you could do with a conjunction.
- Remember that since you write "=" in every step, the propositions need indeed be equivalent. Make sure that this is the case.

2 Layout of proofs

Although we have seen several proof strategies (see handout on proof theory), the layout should always be the same. It is important to keep a few basic things in mind. The number one rule is that everything should be *clearly* presented. This means, for example, that the only rules that one is allowed to skip are commutativity and associativity. For the sake of clarity, and also for the sake of not doing unnecessary mistakes, one should always keep parantheses. If one uses some rule of arithmetic, that rule should be explicitly stated.

In general, to prove $p1 \wedge p_2 \wedge \ldots \wedge p_k \Rightarrow (q_1 = q_n)$, the solution should be of this form:

Prove:

[Statement you want to prove]

Proof:

$$\left.\begin{array}{l}\textbf{Assume: } p_1\\ \ldots\\ \textbf{Assume: } p_k\end{array}\right\} \text{ k assumptions, where k might be 0}$$

$$\begin{array}{l} q_1 \\ = \{\text{[some proof rule]}\} \\ q_2 \\ = \{\text{[some proof rule]}\} \\ q_3 \\ \vdots \\ = \{\text{[some proof rule]}\} \\ q_n \ \square \end{array}$$

If you do not which to prove an implication but instead an equality like $q_1 = q_n$, just skip the "**Assume:**" statements.

In particular, this means that one can *not* assume something in the middle of the proof, and then continue. It is also important to note that the *only* rules one is allowed to use, are the ones in the compendium, plus any lemmas one might already have proved. If one wants to use for example substitution using a rule saying "$p = q$" to substitute p for q, we

need to have an expression involving *exactly* p at our current stage in the proof. We cannot have "almost" p.

To sum this up, always write proofs as they are presented in the compendium.

3 How not to use rules with implication

When looking at the set of axioms and theorems, one can easily spot quite a few that is of the form $p \Rightarrow q$. This, one might think, is perfect for proving implications, because one can drop the proofstyle with "$=$ {[some proof rule]}" everywhere, for the weaker statement "$\Rightarrow$ {[some proof rule involving implication]}". However, this highly problematic. As a small example, consider the deduction rule saying that from "$p = q$", we can deduce "$E(p) = E(q)$"(*). However, it is not true in general that from "$p \Rightarrow q$", one can deduce "$E(p) \Rightarrow E(q)$"(**). As an example, take $E(r) = \neg r$. Here clearly (*) is true, but not (**).

The bottom line here is of course that one should use the rules involving implication carefully. And, as remarked in the previous Section, be sure to write proofs exactly as in the compendium. There is a reason that the compendium only has proofs with equality.

Also see Section 5 for an example of using rules with implication.

4 The use of conditional substitution

A very powerful derivation rule, and also a rule which can be hard to use correctly, is the rule of conditional substitution:

$$\frac{q_1\ ,\ q_2\ ,\ \ldots\ ,\ q_n\ ,\ (q_1 \wedge q_2 \wedge \ldots \wedge q_n)\ \Rightarrow\ (p_1 = p_2)}{E(p_1) = E(p_2)\ ,\ E(p_2) = E(p_2)}$$

Let us take a closer look at this derivation rule. It says that if we know that $q_1, \ldots, q_n$ are true, and that $(q_1 \wedge q_2 \wedge \ldots \wedge q_n) \Rightarrow (p_1 = p_2)$ is true, we can deduce $E(p_1) = E(p_2)$ and $E(p_2) = E(p_1)$. That is, we can deduce that given the premises, any expression involving p_1 (written as $E(p_1)$ in the derivation rule) is equivalent to the same expression with p_1 substituted for p_2 (written $E(p_2)$).

Suppose we are in the middle of a proof and we want to use conditional substitution with some rule saying "$(q_1 \wedge q_2) \Rightarrow (p_1 = p_2)$" to replace "$E(p_1)$" with "$E(p_2)$". According to the rule above, we need establish that q_1 and q_2 are true, in order for the substitution to be valid. The question is how this can be established. The answer here is very simple. The premises q_1 and q_2 can be in three places: they can be assumptions, lemmas proven by you earlier, or axioms/theorems. In other words, the following proof outline would be the only valid way to use conditional substitution and the rule above (the [...] means a skipped part of the proof that is irrelevant to illustrate the point here, but one should remember that the proof *always* should conform to the rules of Section 2):

Proof outline

Prove:

Proof:

Assume: p_1
...
Assume: p_k
} k assumptions, where k might be 0

q_1
$=$ {[some proof rule]}
q_2
$=$ {[some proof rule]}
q_3
$\vdots$
$=$ {[some proof rule]}
q_n $\Box$

(There is actually some more cases, namely those where q_1, q_2 or both q_1 and q_2 are theorems, axioms or lemmas and not assumptions.)

4.1 Conditional Substitution as a Macro

Actually, we can prove that the use of conditional substitution is fair game by proving that it is actually a variant of things we already have, such as standard substitution:

$(q_1 \wedge q_2 \wedge \ldots \wedge q_n \wedge ((q_1 \wedge q_2 \wedge \ldots \wedge q_n) \Rightarrow (p_1 = p_2))) \Rightarrow E(p_1) = E(p_2)$

Proof:

Assume: q_1
$\vdots$
Assume: q_n
Assume: $(q_1 \wedge q_2 \wedge \ldots \wedge q_n) \Rightarrow (p_1 = p_2)$

$E(p_1)$
$=$ {And simplification}
$T \wedge E(p_1)$
$=$ {Assumption $(q_1 \wedge q_2 \wedge \ldots \wedge q_n) \Rightarrow (p_1 = p_2)$}
$((q_1 \wedge q_2 \wedge \ldots \wedge q_n) \Rightarrow (p_1 = p_2)) \wedge E(p_1)$
$=$ {Assumptions $q_1, q_2, \ldots, q_n$}
$((T \wedge T \wedge \ldots \wedge T) \Rightarrow (p_1 = p_2)) \wedge E(p_1)$
$=$ {And simplification}
$(T \Rightarrow (p_1 = p_2)) \wedge E(p_1)$
$=$ {Left identity of $\Rightarrow$}
$(p_1 = p_2) \wedge E(p_1)$
$=$ {Substitution (2.33 in `exam-comp.pdf`)}

$(p_1 = p_2) \wedge E(p_2)$
$=$ {Left identity of $\Rightarrow$}
$(T \Rightarrow (p_1 = p_2)) \wedge E(p_2)$
$=$ {And simplification}
$((T \wedge T \wedge \ldots \wedge T) \Rightarrow (p_1 = p_2)) \wedge E(p_2)$
$=$ {Assumptions $q_1, q_2, \ldots, q_n$}
$((q_1 \wedge q_2 \wedge \ldots \wedge q_n) \Rightarrow (p_1 = p_2)) \wedge E(p_2)$
$=$ {Assumption $(q_1 \wedge q_2 \wedge \ldots \wedge q_n) \Rightarrow (p_1 = p_2)$}
$T \wedge E(p_2)$
$=$ {And simplification}
$E(p_2)$

4.2 What if the assumptions do not match the rules?

Suppose you want to prove that $a \leq b \wedge a \neq b \wedge a \leq c \wedge b \leq c \Rightarrow (a < c)$ using conditional substitution. You start the proof as you always do for an implication:

Proof:

Assume: $a \leq b$
Assume: $a \neq b$
Assume: $a \leq c$
Assume: $b \leq c$

$a < c$

And now what? The arithmetic rule we might want to apply, "$a < b \wedge b \leq c \Rightarrow a < c$", is not applicable as it stands. the reason is that we cannot use the assumption straight away, because it is not *exactly* on the form we want. One way of solving this problem is by the use some initial manipulation of the whole expression, and then a lemma:

Proof:

$a \leq b \wedge a \neq b \wedge a \leq c \wedge b \leq c \Rightarrow (a < c)$
$=$ {Arithmetic: "$(a \leq b \wedge a \neq b) = (a < b)$"}
$a < b \wedge a \leq c \wedge b \leq c \Rightarrow (a < c)$
$=$ {Lemma}
T □

Lemma: $a < b \wedge a \leq c \wedge b \leq c \Rightarrow (a < c)$

Proof of Lemma:

Assume: $a < b$
Assume: $a \leq c$
Assume: $b \leq c$

$a < c$
$=$ {Conditional substitution using Arithmetic: "$(a < b \wedge b \leq c) \Rightarrow (a < c)$" and assumptions "$a < b$" and "$b \leq c$"}
T □

Let's sum up what we did here: we realized that we did not get the assumptions we wanted from just applying proof technique 3 (see the handout from proof theory tutorial). We then instead manipulated the whole expression to give us the right assumptions, and then we proved the manipulated statement as a lemma, using method 3.

There is also an alternative way to handle the situation is as follows: suppose that there is an assumption that "kind of" fits some rule for conditional substitution. One example might be that there was an assumption of the form "$m < n + 1$", but the rule one wants to use for conditional substitution is of the form "$(m \leq n) \Rightarrow (p = q)$". Instead of doing as we just discussed, we could prove a lemma stating "$(m < n+1) \Rightarrow (p = q)$" and then use conditional substitution using this lemma and the original assumption.

To sum this up, conditional substitution is powerful, but needs to be treated with respect. One should make sure to always state the rules and the assumptions that the conditional substitution is based on, if not only for the fact that this is an extra sanity test to see that is is really applicable.

5 Arithmetic

Even though the compendium does things differently, you are required in anything you hand in to write not only "Arithmetic", but also state the arithmetic rule you apply. As an example, let's prove the theorem "$(x = y + 1 \vee y = x + 1) \Rightarrow (x < y \vee y < x)$":

Assume $x = y + 1 \vee y = x + 1$

$$
\begin{aligned}
& x < y \vee y < x \\
= \quad & \{ \text{ and-simplification } \} \\
& (x < y \vee y < x) \wedge T \\
= \quad & \{ \text{ Assumption: "} x = y + 1 \vee y = x + 1\text{" } \} \\
& (x < y \vee y < x) \wedge (x = y + 1 \vee y = x + 1) \\
= \quad & \{ \text{ Distributivity } \} \\
& ((x < y \vee y < x) \wedge x = y + 1) \vee ((x < y \vee y < x) \wedge y = x + 1) \\
= \quad & \{ \text{ Distributivity } \} \\
& (x < y \wedge x = y + 1) \vee (y < x \wedge x = y + 1) \vee \\
& (x < y \wedge y = x + 1) \vee (y < x \wedge y = x + 1) \\
= \quad & \{ \text{ Arithmetic: "} (x < y \wedge x = y + 1) = F\text{" } \} \\
& F \vee (y < x \wedge x = y + 1) \vee \\
& (x < y \wedge y = x + 1) \vee (y < x \wedge y = x + 1) \\
= \quad & \{ \text{ Arithmetic: "} (y < x \wedge y = x + 1) = F\text{" } \} \\
& F \vee (y < x \wedge x = y + 1) \vee \\
& (x < y \wedge y = x + 1) \vee F \\
= \quad & \{ \text{ or-simplification } \} \\
& (y < x \wedge x = y + 1) \vee (x < y \wedge y = x + 1)
\end{aligned}
$$

$$
\begin{array}{ll}
= & \{ \text{ Arithmetic: “}(y < x \wedge x = y + 1) = (x = y + 1)\text{” } \} \\
& (x = y + 1) \vee (x < y \wedge y = x + 1) \\
= & \{ \text{ Arithmetic: “}(x < y \wedge y = x + 1) = (y = x + 1)\text{” } \} \\
& (x = y + 1) \vee (y = x + 1) \\
= & \{ \text{ Assumption: “}x = y + 1 \vee y = x + 1\text{” } \} \\
& T \quad \Box
\end{array}
$$

6 The definition of the alternative command and the alternative command theorem

The alternative command has the form

$$
\begin{array}{lllll}
IF: & \textbf{if} & B_1 & \rightarrow & S_1 \\
 & \Box & \vdots & \rightarrow & \vdots \\
 & \Box & B_n & \rightarrow & S_n \\
 & \textbf{fi} & & &
\end{array}
$$

Intuitively, the behaviour of this command is that *nondeterministically*, a guard B_i that evaluates to true in the current state is chosen, and the statement S_i is executed.

Definition of the weakest precondition for the alternative command The weakest precondition of the alternative commad theorem is defined as follows:

$$wp(IF, R) = (B_1 \vee \ldots \vee B_n) \wedge (B_1 \;\Rightarrow\; wp(S_1, R)) \wedge \ldots \wedge (B_n \;\Rightarrow\; wp(S_n, R))$$

Let us examine conjunction on the left-hand side of this definition closer. Recall that $wp(IF, R)$ defines the set of states such that IF can be executed and we end up in in a state satisfying R. The first conjunct $(B_1 \vee \ldots \vee B_n)$ says that for all the states, there should be at least one true guard. Each of the following conjuncts are of the form $B_i \;\Rightarrow\; wp(S_i, R)$. What does this mean? Suppose that the guard B_i is true. Then you must be sure to end up in a state satisfying R after executing S_i.

Alternative Command Theorem If a predicate Q satisfies

(1) $Q \;\Rightarrow\; (B_1 \vee \ldots \vee B_n)$
(2) $(Q \wedge B_i) \;\Rightarrow\; wp(S_i, R)$ for all i such that $1 \leq i \leq n$

Then, and only then, $Q \;\Rightarrow\; wp(IF, R)$.

The question is now why this theorem is useful. The answer here, is that the alternative command provides a shortcut for proving the correctness of programs of the form

$$
\begin{array}{c}
\{Q\} \\
IF \\
\{R\}
\end{array}
$$

The theorem can not be used (directly) to solve problems of the form "simplify $wp(IF, R)$", since the theorem is stated including Q. This might be more clear if we consider the following, alternative, formulation of the theorem:

$$((1) \wedge (2)) = (Q \ \Rightarrow \ wp(IF, R))$$

Here, we have of course (1) and (2) as above. With this formulation, we can see that the alternative command theorem does not speak about $wp(IF, R)$ directly, but rather its relation with Q.

To sum up, if you want to simplify an expression that includes the alternative command, use its definition of the weakest precondition. If you want to prove a program correct, and the program has *exactly* the form of IF above, use the alternative command theorem.