Compiler Construction

Dr. Michael Petter, Raphaela Palenta SS 2019

Exercise Sheet 9

Assignment 9.1 Type Checking

This exercise is about checking the types of expressions given in our C-like language. Use a deduction tree to check whether the statements are well-typed. Make sure to only use the rules given in the lecture and specify the rule for each step.

1. Given the declarations $\Gamma \coloneqq \{\mathsf{int}\ x, \mathsf{int}\ a[]\}$, check whether the statement $\mathsf{int}\ y = x + a[42];$ is well-typed.

2. Given the declarations $\Gamma \coloneqq \{\mathsf{int}\ y,\ \mathsf{double}\ a[],\ \mathit{struct}\ \{\mathsf{double}\ a[];\}\ g,\ \mathsf{int}\ (*f)(\mathsf{double})\}$, check whether the statement $\mathsf{int}\ x = f(g.a[y+2]);$ is well-typed.

Suggested Solution 9.1

1.

$$\dfrac{\textsc{Var}\ \dfrac{}{\Gamma \vdash x : \mathsf{int}} \qquad \textsc{Array}\ \dfrac{\textsc{Var}\ \dfrac{}{\Gamma \vdash a : \mathsf{int}[]} \qquad \textsc{Const}\ \dfrac{}{\Gamma \vdash 42 : \mathsf{int}}}{\Gamma \vdash a[42] : \mathsf{int}}}{\Gamma \vdash x + a[42] : \mathsf{int}}\ \textsc{Op}$$

2.

$$\textsc{Array}\ \dfrac{\textsc{struct}\ \dfrac{\textsc{Var}\ \dfrac{}{\Gamma \vdash g : struct\{\mathsf{double}[]\ a;\}}}{\Gamma \vdash g.a : \mathsf{double}[]} \qquad \textsc{Op}\ \dfrac{\textsc{Var}\ \dfrac{}{\Gamma \vdash y : int} \qquad \textsc{Const}\ \dfrac{}{\Gamma \vdash 2 : int}}{\Gamma \vdash y + 2 : \mathsf{int}}}{\Gamma \vdash g.a[y+2] : \mathsf{double}}$$

$$\lightning\ \dfrac{\textsc{Var}\ \dfrac{}{\Gamma \vdash f : \mathsf{int(*)(double)}} \qquad \Gamma \vdash g.a[y+2] : \mathsf{double}}{\Gamma \vdash f(g.a[y+2]) :?}$$

There is no rule in our type system that can be applied to `f: int(*)(double)` and `g.a[y+2]: double`.

Assignment 9.2 Subtyping

Consider the following C structs:

```
struct A {
        A f(B, C);
        C g(C);
}

struct B {
        B f(A, D);
        A g(D);
}

struct C {
        C f(B, B);
        D g(A);
}

struct D {
        D f(B, B);
        D g(B);
        int a;
}
```

We are going to use the non-standard subtyping rules for C structures which have been introduced in the lecture. Let $\leq$ be the type comparison operator, that is, for two types A and B the following holds:

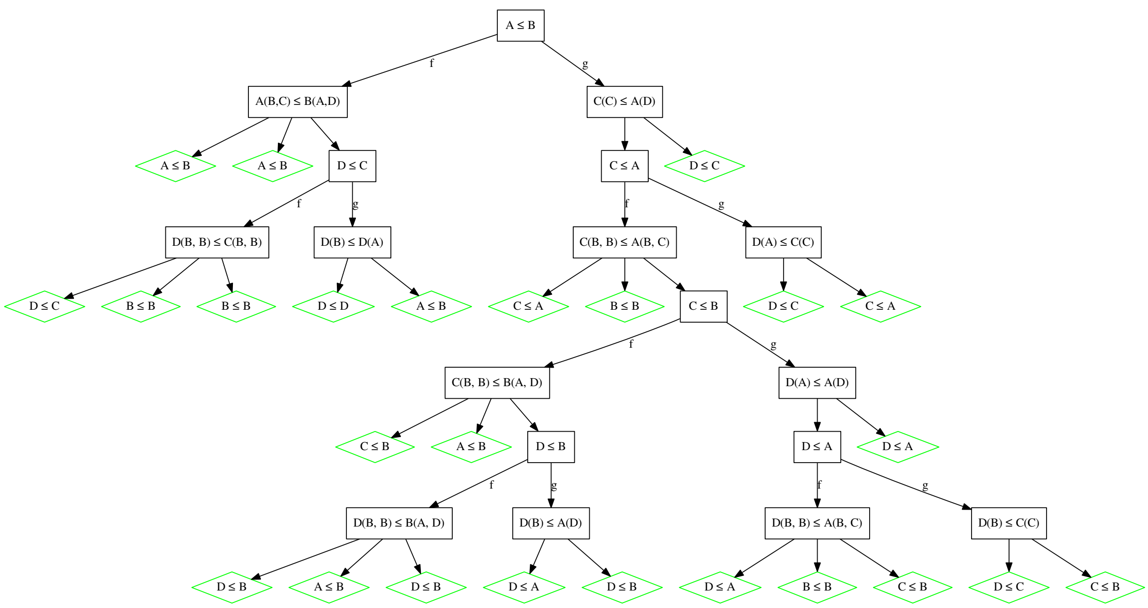
$$A \leq B \Leftrightarrow A \text{ is a subtype of } B \tag{1}$$

Now, proof the assertions below either right or wrong:

1. $A \leq B$
2. $A \leq C$

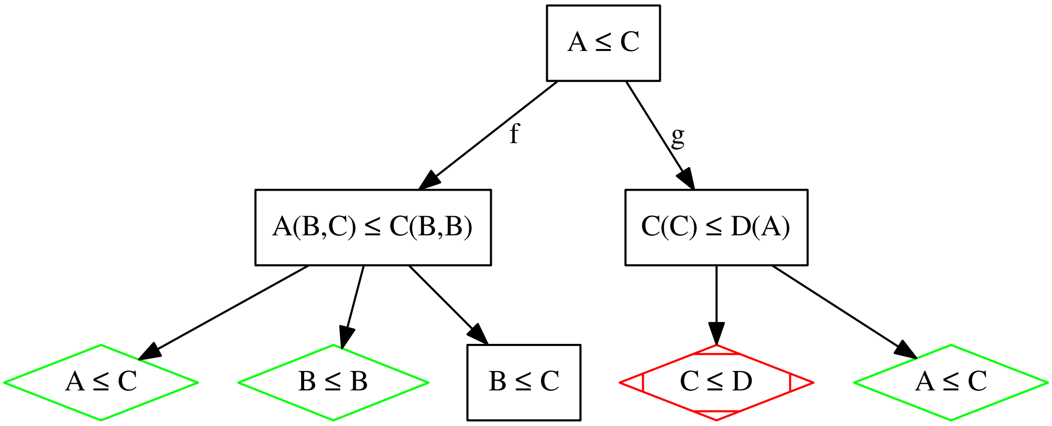
Suggested Solution 9.2

1.



Since no contradictions can be found, it follows that $A \leq B$ holds.

2.



Since D contains a field $(D.a)$ that is not contained in C, $C \leq D$ cannot hold. Therefore, $A \leq C$ does not hold.