Static Program Analysis Selected Solutions

Lingming Zhang

September 28, 2025

1 Introduction

Exercise 1.1 There are several pointer-related errors in the illustrated program:

- Null pointer dereference at `printf("%s", p)`;
- Use after free at `*p = 'x'`;
- Double free at `free(p)`;
- Memory leak introduced by two consecutive `(char*)malloc(100)`;
- Undefined behavior at `strcat(p, q)`;
- Assertion failure at `assert(argc > 87)`;

2 A Tiny Imperative Programming Language

Exercise 2.1 The under-specified parts of the TIP language may include the production rule for variable definition, and the type of `null` value (treating it as an *integer* or a *pointer* type?).

Exercise 2.3 The normalized form is as follows:

```
t0 = *f;
t1 = *t0;
t2 = g();
t3 = h();
t4 = t2 + t3;
t5 = t1(t4);
x = t5;
```

Exercise 2.4 The statement `**x = **y;` can be normalized to:

```
t0 = *y;
t1 = *t0;
t2 = *x;
*t2 = t1;
```

Exercise 2.6 The AST and CFG of the `rec` program are presented in Figure 1 and Figure 2.

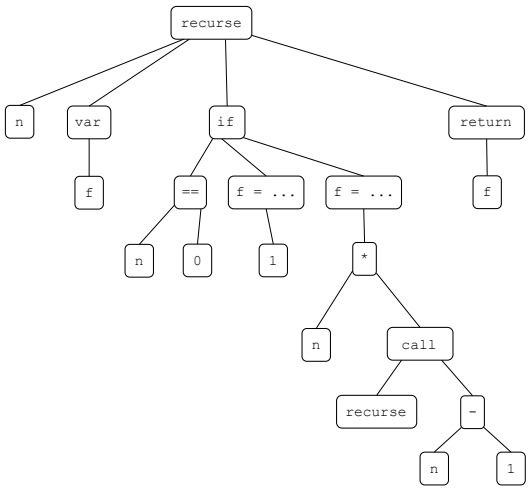

Figure 1: The Abstract Syntax Tree of `rec`

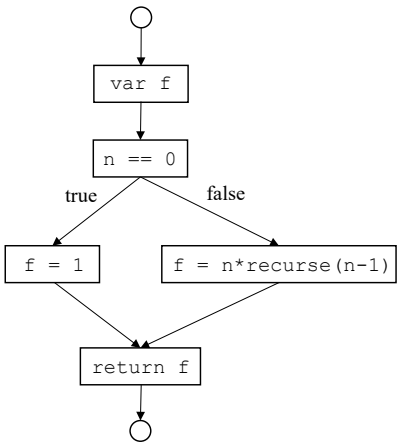

Figure 2: The Control Flow Graph of `rec`

Exercise 2.7 The corresponding CFG should be like Figure 3.

3 Type Analysis

Exercise 3.1 Here is an example where javac cannot recognize `ob` is actually an object of type `B` and reports `error:cannot find symbol`.

```
class A {
  void methodA() {
    System.out.println("Method of Class A.");
  }
}
class B extends A {
  void methodB() {
    System.out.println("Method of Class B.");
  }
  public static void main(String[] args) {
    A ob = new B();
    ob.methodB();
  }
}
```

Exercise 3.2 Java's covariant typing of arrays may allow the following program to be compiled, which raises an `ArrayStoreException` at runtime:

```
String strings[] = {"Broken","Type","system"};
Object objects[] = strings;
objects[0] = 5;
```

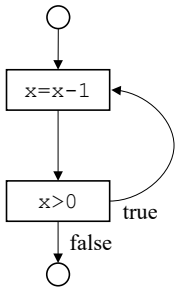

Figure 3: The Control Flow Graph of `do-while`

Exercise 3.4 `recurse` has type (`Int`) $\rightarrow$ `Int`. Both `f` and `n` are of type `Int`.

Exercise 3.6 We explain some of the type constraint rules here:

- $E(E_1, ..., E_n)$: Each parameter type of the function type $[[E]]$ should be equal to the corresponding $[[E_i]]$ and the return type of $[[E]]$ is assigned to current node $[[E(E_1, ..., E_n)]]$;
- `&`X: $[[\texttt{\&}X]]$ should be a pointer type pointing to whichever $[[X]]$ represents, so its type should be $\uparrow [[X]]$;
- $*E_1 = E_2$: A store operation requires two sides having the same type, so $[[E_1]]$ should be of type $\uparrow [[E_2]]$;

Exercise 3.7 The `y = 42` produces two type constraints $[[\texttt{42}]] = \texttt{int}$ and $[[\texttt{y}]] = [[\texttt{42}]]$, and as per the transitivity rule we have $[[\texttt{y}]] = \texttt{int}$. However, the `*y = x`

gives $[[\texttt{y}]] =\uparrow [[\texttt{x}]] =\uparrow \texttt{int}$. Since we have two distinct types assigned to a single variable `y`, the resulting constraints are unsolvable.

Exercise 3.9 The type constraints are as follows:

$[[\texttt{1}]] = \texttt{int}$
$[[\texttt{alloc 1}]] =\uparrow [[\texttt{1}]]$
$[[\texttt{x}]] = [[\texttt{alloc 1}]]$
$[[\texttt{2}]] = \texttt{int}$
$[[\texttt{alloc 2}]] =\uparrow [[\texttt{2}]]$
$[[\texttt{alloc (alloc 2)}]] =\uparrow [[\texttt{alloc 2}]]$
$[[\texttt{y}]] = [[\texttt{alloc (alloc 2)}]]$
$[[\texttt{x}]] = [[\texttt{y}]]$

First, we can deduce that $[[\texttt{x}]] =\uparrow \texttt{int}$ and $[[\texttt{y}]] =\uparrow\uparrow \texttt{int}$. However, for the last type constraint, we have $\uparrow \texttt{int} =\uparrow\uparrow \texttt{int}$. The term equality axiom simplifies the constraint to $\texttt{int} =\uparrow \texttt{int}$, which is clearly unsolvable.

Exercise 3.13 The type constraints are as follows:

$[[\texttt{input}]] = \texttt{int}$
$[[\texttt{x}]] = [[\texttt{input}]]$
$[[\texttt{y}]] =\uparrow [[\texttt{x}]]$
$[[\texttt{y}]] =\uparrow [[*\texttt{y}]]$
$[[\texttt{z}]] = [[*\texttt{y}]]$

The unification algorithm proceeds as follows:

- $[[\texttt{input}]] = \texttt{int}$: $\texttt{Set}(\texttt{int}) = \{\texttt{int}, [[\texttt{input}]]\}$;
- $[[\texttt{x}]] = [[\texttt{input}]]$: $\texttt{Set}(\texttt{int}) = \{\texttt{int}, [[\texttt{input}]], [[\texttt{x}]]\}$;
- $[[\texttt{y}]] =\uparrow [[\texttt{x}]]$: $\texttt{Set}(\uparrow [[\texttt{x}]]) = \{\uparrow [[\texttt{x}]], [[\texttt{y}]]\}$;
- $[[\texttt{y}]] =\uparrow [[*\texttt{y}]]$: $\texttt{Set}(\uparrow [[*\texttt{y}]]) = \{\uparrow [[\texttt{x}]], [[\texttt{y}]], \uparrow [[*\texttt{y}]]\}$, and according to the term equality axiom, $\texttt{Set}(\texttt{int}) = \{\texttt{int}, [[\texttt{input}]], [[\texttt{x}]], [[*\texttt{y}]]\}$;
- $[[\texttt{z}]] = [[*\texttt{y}]]$: $\texttt{Set}(\texttt{int}) = \{\texttt{int}, [[\texttt{input}]], [[\texttt{x}]], [[*\texttt{y}]], [[\texttt{z}]]\}$;

Exercise 3.14 The unification algorithm proceeds as follows:

- $[[\texttt{null}]] =\uparrow \texttt{t}$: $\texttt{Set}(\uparrow \texttt{t}) = \{\uparrow \texttt{t}, [[\texttt{null}]]\}$;
- $[[\texttt{alloc null}]] =\uparrow [[\texttt{null}]]$: $\texttt{Set}(\uparrow [[\texttt{null}]]) = \{\uparrow [[\texttt{null}]], [[\texttt{alloc null}]]\}$;
- $[[\texttt{p}]] = [[\texttt{alloc null}]]$: $\texttt{Set}(\uparrow [[\texttt{null}]]) = \{\uparrow [[\texttt{null}]], [[\texttt{alloc null}]], [[\texttt{p}]]\}$;
- $[[\texttt{p}]] =\uparrow [[\texttt{p}]]$:
  1. $\texttt{Set}(\uparrow [[\texttt{p}]]) = \{\uparrow [[\texttt{p}]], \uparrow [[\texttt{null}]], [[\texttt{alloc null}]], [[\texttt{p}]]\}$;
  2. $\texttt{Set}(\uparrow [[\texttt{p}]]) = \{\uparrow [[\texttt{p}]], \uparrow [[\texttt{null}]], [[\texttt{alloc null}]], [[\texttt{p}]], [[\texttt{null}]], \uparrow \texttt{t}\}$;
  3. $\texttt{Set}(\uparrow [[\texttt{p}]]) = \{\uparrow [[\texttt{p}]], \uparrow [[\texttt{null}]], [[\texttt{alloc null}]], [[\texttt{p}]], [[\texttt{null}]], \uparrow \texttt{t}, \texttt{t}\}$;

Exercise 3.16 The type constraints are as follows:

$[[\texttt{map}]] = ([[\texttt{l}]], [[\texttt{f}]], [[\texttt{z}]]) \to [[\texttt{r}]]$
$[[\texttt{l} == \texttt{null}]] = \texttt{int}$
$[[\texttt{null}]] =\uparrow \alpha$
$[[\texttt{l}]] = [[\texttt{null}]]$
$[[\texttt{r}]] = [[\texttt{z}]]$
$[[\texttt{l}]] =\uparrow [[*\texttt{l}]]$
$[[\texttt{map}]] = ([[*\texttt{l}]], [[\texttt{f}]], [[\texttt{z}]]) \to [[\texttt{map}(*\texttt{l}, \texttt{f}, \texttt{z})]]$
$[[\texttt{f}]] = ([[\texttt{map}(*\texttt{l}, \texttt{f}, \texttt{z})]]) \to [[\texttt{f}(\texttt{map}(*\texttt{l}, \texttt{f}, \texttt{z}))]]$
$[[\texttt{r}]] = [[\texttt{f}(\texttt{map}(*\texttt{l}, \texttt{f}, \texttt{z}))]]$
$[[\texttt{foo}]] = ([[\texttt{i}]]) \to [[\texttt{i} + 1]]$
$[[1]] = \texttt{int}$
$[[\texttt{i}]] = \texttt{int}$
$[[\texttt{i} + 1]] = \texttt{int}$
$[[\texttt{main}]] = () \to [[\texttt{map}(\texttt{h}, \texttt{foo}, 0)]]$
$[[\texttt{t}]] = [[\texttt{null}]]$
$[[42]] = \texttt{int}$
$[[\texttt{n}]] = [[42]]$
$[[0]] = \texttt{int}$
$[[\texttt{n}]] = \texttt{int}$
$[[\texttt{n} > 0]] = \texttt{int}$
$[[\texttt{n} - 1]] = \texttt{int}$
$[[\texttt{n}]] = [[\texttt{n} - 1]]$
$[[\texttt{alloc null}]] =\uparrow [[\texttt{null}]]$
$[[\texttt{h}]] = [[\texttt{alloc null}]]$
$[[\texttt{h}]] =\uparrow [[*\texttt{h}]]$
$[[\texttt{h}]] =\uparrow [[\texttt{t}]]$
$[[\texttt{t}]] = [[\texttt{h}]]$
$[[\texttt{map}]] = ([[\texttt{h}]], [[\texttt{foo}]], [[0]]) \to [[\texttt{map}(\texttt{h}, \texttt{foo}, 0)]]$
$[[\texttt{map}(\texttt{h}, \texttt{foo}, 0)]] = \texttt{int}$

The unification algorithm proceeds as follows:

- $\texttt{Set}(([[\texttt{l}]], [[\texttt{f}]], [[\texttt{z}]]) \to [[\texttt{r}]]) = \{([[\texttt{l}]], [[\texttt{f}]], [[\texttt{z}]]) \to [[\texttt{r}]], [[\texttt{map}]]\}$;
- $\texttt{Set}(\texttt{int}) = \{\texttt{int}, [[\texttt{l} == \texttt{null}]]\}$;
- $\texttt{Set}(\uparrow \alpha) = \{\uparrow \alpha, [[\texttt{null}]]\}$;
- $\texttt{Set}(\uparrow \alpha) = \{\uparrow \alpha, [[\texttt{null}]], [[\texttt{l}]]\}$;
- $\texttt{Set}([[\texttt{z}]]) = \{[[\texttt{z}]], [[\texttt{r}]]\}$;
- $\texttt{Set}(\uparrow [[*\texttt{l}]]) = \{\uparrow [[*\texttt{l}]], \uparrow \alpha, [[\texttt{null}]], [[\texttt{l}]]\}$;
    1. $\texttt{Set}([[*\texttt{l}]]) = \{[[*\texttt{l}]], \alpha\}$;
- $\texttt{Set}(([[*\texttt{l}]], [[\texttt{f}]], [[\texttt{z}]]) \to [[\texttt{map}(*\texttt{l}, \texttt{f}, \texttt{z})]]) =$
  $\{([[*\texttt{l}]], [[\texttt{f}]], [[\texttt{z}]]) \to [[\texttt{map}(*\texttt{l}, \texttt{f}, \texttt{z})]], ([[\texttt{l}]], [[\texttt{f}]], [[\texttt{z}]]) \to [[\texttt{r}]], [[\texttt{map}]]\}$
    1. $\texttt{Set}(\uparrow [[*\texttt{l}]]) = \{\uparrow [[*\texttt{l}]], \uparrow \alpha, [[\texttt{null}]], [[\texttt{l}]], [[*\texttt{l}]], \alpha\}$;
    2. $\texttt{Set}([[\texttt{map}(*\texttt{l}, \texttt{f}, \texttt{z})]]) = \{[[\texttt{map}(*\texttt{l}, \texttt{f}, \texttt{z})]], [[\texttt{r}]], [[\texttt{z}]]\}$;

- $\texttt{Set}((\![\texttt{map}(*\texttt{l},\texttt{f},\texttt{z})]\!]) \to [\![\texttt{f}(\texttt{map}(*\texttt{l},\texttt{f},\texttt{z}))]\!]) =$
  $\{(\![\texttt{map}(*\texttt{l},\texttt{f},\texttt{z})]\!]) \to [\![\texttt{f}(\texttt{map}(*\texttt{l},\texttt{f},\texttt{z}))]\!], [\![\texttt{f}]\!]\}$;
- $\texttt{Set}([\![\texttt{f}(\texttt{map}(*\texttt{l},\texttt{f},\texttt{z}))]\!]) = \{[\![\texttt{f}(\texttt{map}(*\texttt{l},\texttt{f},\texttt{z}))]\!], [\![\texttt{map}(*\texttt{l},\texttt{f},\texttt{z})]\!], [\![\texttt{r}]\!], [\![\texttt{z}]\!]\}$;
- $\texttt{Set}((\![\texttt{i}]\!]) \to [\![\texttt{i}+1]\!]) = \{(\![\texttt{i}]\!]) \to [\![\texttt{i}+1]\!], [\![\texttt{foo}]\!]\}$;
- $\texttt{Set}(\texttt{int}) = \{\texttt{int}, [\![\texttt{l} == \texttt{null}]\!], [\![1]\!], [\![\texttt{i}]\!], [\![\texttt{i}+1]\!]\}$;
- $\texttt{Set}(() \to [\![\texttt{map}(\texttt{h},\texttt{foo},0)]\!]) = \{() \to [\![\texttt{map}(\texttt{h},\texttt{foo},0)]\!], [\![\texttt{main}]\!]\}$;
- $\texttt{Set}(\uparrow [\![*\texttt{l}]\!]) = \{\uparrow [\![*\texttt{l}]\!], \uparrow \alpha, [\![\texttt{null}]\!], [\![\texttt{l}]\!], [\![*\texttt{l}]\!], \alpha, [\![\texttt{t}]\!]\}$;
- $\texttt{Set}(\texttt{int}) =$
  $\{\texttt{int}, [\![\texttt{l} == \texttt{null}]\!], [\![1]\!], [\![\texttt{i}]\!], [\![\texttt{i}+1]\!], [\![42]\!], [\![\texttt{n}]\!], [\![0]\!], [\![\texttt{n} > 0]\!], [\![\texttt{n}-1]\!]\}$;
- $\texttt{Set}(\uparrow [\![\texttt{null}]\!]) = \{\uparrow [\![\texttt{null}]\!], [\![\texttt{alloc null}]\!], [\![\texttt{h}]\!]\}$;
- $\texttt{Set}(\uparrow [\![*\texttt{h}]\!]) = \{\uparrow [\![*\texttt{h}]\!], \uparrow [\![\texttt{null}]\!], [\![\texttt{alloc null}]\!], [\![\texttt{h}]\!]\}$;

  1. $\texttt{Set}(\uparrow [\![*\texttt{l}]\!]) = \{\uparrow [\![*\texttt{l}]\!], \uparrow \alpha, [\![\texttt{null}]\!], [\![\texttt{l}]\!], [\![*\texttt{l}]\!], \alpha, [\![\texttt{t}]\!], [\![*\texttt{h}]\!]\}$;

- $\texttt{Set}(\uparrow [\![\texttt{t}]\!]) = \{\uparrow [\![*\texttt{h}]\!], \uparrow [\![\texttt{null}]\!], [\![\texttt{alloc null}]\!], [\![\texttt{h}]\!], \uparrow [\![\texttt{t}]\!]\}$;
- $\texttt{Set}(\uparrow [\![\texttt{t}]\!]) =$
  $\{\uparrow [\![*\texttt{h}]\!], \uparrow [\![\texttt{null}]\!], [\![\texttt{alloc null}]\!], [\![\texttt{h}]\!], \uparrow [\![\texttt{t}]\!], \uparrow [\![*\texttt{l}]\!], \uparrow \alpha, [\![\texttt{null}]\!],$
  $[\![\texttt{l}]\!], [\![*\texttt{l}]\!], \alpha, [\![\texttt{t}]\!], [\![*\texttt{h}]\!]\}$;
- $\texttt{Set}((\![\texttt{h}]\!], [\![\texttt{foo}]\!], [\![0]\!]) \to [\![\texttt{map}(\texttt{h},\texttt{foo},0)]\!]) =$
  $\{(\![*\texttt{l}]\!], [\![\texttt{f}]\!], [\![\texttt{z}]\!]) \to [\![\texttt{map}(*\texttt{l},\texttt{f},\texttt{z})]\!], (\![\texttt{l}]\!], [\![\texttt{f}]\!], [\![\texttt{z}]\!]) \to [\![\texttt{r}]\!], [\![\texttt{map}]\!],$
  $(\![\texttt{h}]\!], [\![\texttt{foo}]\!], [\![0]\!]) \to [\![\texttt{map}(\texttt{h},\texttt{foo},0)]\!]\}$;

  1. $\texttt{Set}((\![\texttt{i}]\!]) \to [\![\texttt{i}+1]\!]) =$
     $\{(\![\texttt{i}]\!]) \to [\![\texttt{i}+1]\!], [\![\texttt{foo}]\!],$
     $(\![\texttt{map}(*\texttt{l},\texttt{f},\texttt{z})]\!]) \to [\![\texttt{f}(\texttt{map}(*\texttt{l},\texttt{f},\texttt{z}))]\!], [\![\texttt{f}]\!]\}$;
     (a) $\texttt{Set}(\texttt{int}) =$
     $\{\texttt{int}, [\![\texttt{l} == \texttt{null}]\!], [\![1]\!], [\![\texttt{i}]\!], [\![\texttt{i}+1]\!], [\![42]\!], [\![\texttt{n}]\!], [\![0]\!],$
     $[\![\texttt{n} > 0]\!], [\![\texttt{n}-1]\!], [\![\texttt{f}(\texttt{map}(*\texttt{l},\texttt{f},\texttt{z}))]\!], [\![\texttt{map}(*\texttt{l},\texttt{f},\texttt{z})]\!], [\![\texttt{r}]\!], [\![\texttt{z}]\!]\}$;
  2. $\texttt{Set}(\texttt{int}) =$
     $\{\texttt{int}, [\![\texttt{l} == \texttt{null}]\!], [\![1]\!], [\![\texttt{i}]\!], [\![\texttt{i}+1]\!], [\![42]\!], [\![0]\!], [\![\texttt{map}(\texttt{h},\texttt{foo},0)]\!],$
     $[\![\texttt{n} > 0]\!], [\![\texttt{n}-1]\!], [\![\texttt{f}(\texttt{map}(*\texttt{l},\texttt{f},\texttt{z}))]\!], [\![\texttt{map}(*\texttt{l},\texttt{f},\texttt{z})]\!], [\![\texttt{r}]\!], [\![\texttt{z}]\!], [\![\texttt{n}]\!]\}$;

Based on the above calculation, the types of critical variables are:

- map: $(\mu\texttt{t}.\uparrow \texttt{t}, (\texttt{int}) \to \texttt{int}, \texttt{int}) \to \texttt{int}$;
- foo: $(\texttt{int}) \to \texttt{int}$;
- main: $() \to \texttt{int}$;
- h, t: $\mu\texttt{t}.\uparrow \texttt{t}$;

The output from running this program should be 42.

Exercise 3.17 The unification algorithm is dedicated to solving equality constraints instead of inequality ones.

Exercise 3.18 The type constraints are as follows:

$[[\mathtt{a}]] = [[\{\mathtt{f} : 3, \mathtt{g} : 17\}]]$
$[[\{\mathtt{f} : 3, \mathtt{g} : 17\}]] = \{\mathtt{f} : [[3]], \mathtt{g} : [[17]], \mathtt{h} : \diamond\}$
$[[3]] = \mathtt{int}$
$[[17]] = \mathtt{int}$
$[[\mathtt{b}]] = [[\mathtt{a.f}]]$
$[[\mathtt{a}]] = \{\mathtt{f} : [[\mathtt{a.f}]], \mathtt{g} : \alpha_1, \mathtt{h} : \alpha_2\}$
$[[\mathtt{c}]] = [[\{\mathtt{f} : \mathtt{alloc}\ 5, \mathtt{h} : 15\}]]$
$[[\{\mathtt{f} : \mathtt{alloc}\ 5, \mathtt{h} : 15\}]] = \{\mathtt{f} : [[\mathtt{alloc}\ 5]], \mathtt{g} : \diamond, \mathtt{h} : [[15]]\}$
$[[5]] = \mathtt{int}$
$[[15]] = \mathtt{int}$
$[[\mathtt{alloc}\ 5]] = \uparrow [[5]]$
$[[\mathtt{d}]] = [[\mathtt{c.f}]]$
$[[\mathtt{c}]] = \{\mathtt{f} : [[\mathtt{c.f}]], \mathtt{g} : \alpha_3, \mathtt{h} : \alpha_4\}$
$[[\mathtt{a.f}]] \neq \diamond$
$[[\mathtt{c.f}]] \neq \diamond$

The unification algorithm proceeds as follows:

- $\mathtt{Set}([[\{\mathtt{f} : 3, \mathtt{g} : 17\}]]) = \{[[\{\mathtt{f} : 3, \mathtt{g} : 17\}]], [[\mathtt{a}]]\}$;
- $\mathtt{Set}(\{\mathtt{f} : [[3]], \mathtt{g} : [[17]], \mathtt{h} : \diamond\}) =$
  $\{\{\mathtt{f} : [[3]], \mathtt{g} : [[17]], \mathtt{h} : \diamond\}, [[\{\mathtt{f} : 3, \mathtt{g} : 17\}]], [[\mathtt{a}]]\}$;
- $\mathtt{Set}(\mathtt{int}) = \{[[3]], [[17]]\}$;
- $\mathtt{Set}([[\mathtt{a.f}]]) = \{[[\mathtt{a.f}]], [[\mathtt{b}]]\}$;
- $\mathtt{Set}(\{\mathtt{f} : [[\mathtt{a.f}]], \mathtt{g} : \alpha_1, \mathtt{h} : \alpha_2\}) =$
  $\{\{\mathtt{f} : [[\mathtt{a.f}]], \mathtt{g} : \alpha_1, \mathtt{h} : \alpha_2\}, \{\mathtt{f} : [[3]], \mathtt{g} : [[17]], \mathtt{h} : \diamond\}, [[\{\mathtt{f} : 3, \mathtt{g} : 17\}]], [[\mathtt{a}]]\}$;
  1. $\mathtt{Set}(\mathtt{int}) = \{[[3]], [[17]], [[\mathtt{a.f}]], [[\mathtt{b}]], \alpha_1\}$;
  2. $\mathtt{Set}(\diamond) = \{\diamond, \alpha_2\}$;
- $\mathtt{Set}([[\{\mathtt{f} : \mathtt{alloc}\ 5, \mathtt{h} : 15\}]]) = \{[[\{\mathtt{f} : \mathtt{alloc}\ 5, \mathtt{h} : 15\}]], [[\mathtt{c}]]\}$;
- $\mathtt{Set}(\{\mathtt{f} : [[\mathtt{alloc}\ 5]], \mathtt{g} : \diamond, \mathtt{h} : [[15]]\}) =$
  $\{\{\mathtt{f} : [[\mathtt{alloc}\ 5]], \mathtt{g} : \diamond, \mathtt{h} : [[15]]\}, [[\{\mathtt{f} : \mathtt{alloc}\ 5, \mathtt{h} : 15\}]], [[\mathtt{c}]]\}$;
- $\mathtt{Set}(\mathtt{int}) = \{[[3]], [[17]], [[\mathtt{a.f}]], [[\mathtt{b}]], \alpha_1, [[5]], [[15]]\}$;
- $\mathtt{Set}(\uparrow [[5]]) = \{\uparrow [[5]], [[\mathtt{alloc}\ 5]]\}$;
- $\mathtt{Set}([[\mathtt{c.f}]]) = \{[[\mathtt{c.f}]], [[\mathtt{d}]]\}$;
- $\mathtt{Set}(\{\mathtt{f} : [[\mathtt{c.f}]], \mathtt{g} : \alpha_3, \mathtt{h} : \alpha_4\}) =$
  $\{\{\mathtt{f} : [[\mathtt{c.f}]], \mathtt{g} : \alpha_3, \mathtt{h} : \alpha_4\}, \{\mathtt{f} : [[\mathtt{alloc}\ 5]], \mathtt{g} : \diamond, \mathtt{h} : [[15]]\},$
  $[[\{\mathtt{f} : \mathtt{alloc}\ 5, \mathtt{h} : 15\}]], [[\mathtt{c}]]\}$;
  1. $\mathtt{Set}(\uparrow [[5]]) = \{\uparrow [[5]], [[\mathtt{alloc}\ 5]], [[\mathtt{c.f}]], [[\mathtt{d}]]\}$;
  2. $\mathtt{Set}(\diamond) = \{\diamond, \alpha_2, \alpha_3\}$;
  3. $\mathtt{Set}(\mathtt{int}) = \{[[3]], [[17]], [[\mathtt{a.f}]], [[\mathtt{b}]], \alpha_1, [[5]], [[15]], \alpha_4\}$;

Exercise 3.19 The type constraints are as follows:

$[[\mathtt{a}]] = [[\{\mathtt{f} : \mathtt{null}, \mathtt{g} : 17\}]]$
$[[\{\mathtt{f} : \mathtt{null}, \mathtt{g} : 17\}]] = \{\mathtt{f} : [[\mathtt{null}]], \mathtt{g} : [[17]], \mathtt{h} : \diamond\}$
$[[\mathtt{null}]] =\uparrow \beta$
$[[17]] = \mathtt{int}$
$[[\mathtt{b}]] = [[\mathtt{alloc}\ \{\mathtt{g} : 42, \mathtt{h} : 87\}]]$
$[[\mathtt{alloc}\ \{\mathtt{g} : 42, \mathtt{h} : 87\}]] =\uparrow [[\{\mathtt{g} : 42, \mathtt{h} : 87\}]]$
$[[\{\mathtt{g} : 42, \mathtt{h} : 87\}]] = \{\mathtt{f} : \diamond, \mathtt{g} : [[42]], \mathtt{h} : [[87]]\}$
$[[42]] = \mathtt{int}$
$[[87]] = \mathtt{int}$
$[[\mathtt{a.f}]] = [[\mathtt{b}]]$
$[[\mathtt{a}]] = \{\mathtt{f} : [[\mathtt{a.f}]], \mathtt{g} : \alpha_1, \mathtt{h} : \alpha_2\}$
$[[117]] = \mathtt{int}$
$[[(*\mathtt{b}).\mathtt{g}]] = [[117]]$
$[[*\mathtt{b}]] = \{\mathtt{f} : \alpha_3, \mathtt{g} : [[(*\mathtt{b}).\mathtt{g}]], \mathtt{h} : \alpha_4\}$
$[[\mathtt{b}]] =\uparrow [[*\mathtt{b}]]$
$[[\mathtt{c}]] = [[(*(\mathtt{a.f})).\mathtt{g}]]$
$[[*(\mathtt{a.f})]] = \{\mathtt{f} : \alpha_5, \mathtt{g} : [[(*(\mathtt{a.f})).\mathtt{g}]], \mathtt{h} : \alpha_6\}$
$[[\mathtt{a.f}]] =\uparrow [[*(\mathtt{a.f})]]$
$[[\mathtt{a.f}]] \neq \diamond$
$[[(*\mathtt{b}).\mathtt{g}]] \neq \diamond$
$[[(*(\mathtt{a.f})).\mathtt{g}]] \neq \diamond$

The unification algorithm proceeds as follows:

- $\mathtt{Set}([[\{\mathtt{f} : \mathtt{null}, \mathtt{g} : 17\}]]) = \{[[\{\mathtt{f} : \mathtt{null}, \mathtt{g} : 17\}]], [[\mathtt{a}]]\}$;
- $\mathtt{Set}(\{\mathtt{f} : [[\mathtt{null}]], \mathtt{g} : [[17]], \mathtt{h} : \diamond\}) =$
  $\{\{\mathtt{f} : [[\mathtt{null}]], \mathtt{g} : [[17]], \mathtt{h} : \diamond\}, [[\{\mathtt{f} : \mathtt{null}, \mathtt{g} : 17\}]], [[\mathtt{a}]]\}$;
- $\mathtt{Set}(\uparrow \beta) = \{\uparrow \beta, [[\mathtt{null}]]\}$;
- $\mathtt{Set}(\mathtt{int}) = \{\mathtt{int}, [[17]]\}$;
- $\mathtt{Set}([[\mathtt{alloc}\ \{\mathtt{g} : 42, \mathtt{h} : 87\}]]) = \{[[\mathtt{alloc}\ \{\mathtt{g} : 42, \mathtt{h} : 87\}]], [[\mathtt{b}]]\}$;
- $\mathtt{Set}(\uparrow [[\{\mathtt{g} : 42, \mathtt{h} : 87\}]]) =$
  $\{\uparrow [[\{\mathtt{g} : 42, \mathtt{h} : 87\}]], [[\mathtt{alloc}\ \{\mathtt{g} : 42, \mathtt{h} : 87\}]], [[\mathtt{b}]]\}$;
- $\mathtt{Set}(\{\mathtt{f} : \diamond, \mathtt{g} : [[42]], \mathtt{h} : [[87]]\}) =$
  $\{\{\mathtt{f} : \diamond, \mathtt{g} : [[42]], \mathtt{h} : [[87]]\}, [[\{\mathtt{g} : 42, \mathtt{h} : 87\}]]\}$;
- $\mathtt{Set}(\mathtt{int}) = \{\mathtt{int}, [[17]], [[42]], [[87]]\}$;
- $\mathtt{Set}(\uparrow [[\{\mathtt{g} : 42, \mathtt{h} : 87\}]]) =$
  $\{\uparrow [[\{\mathtt{g} : 42, \mathtt{h} : 87\}]], [[\mathtt{alloc}\ \{\mathtt{g} : 42, \mathtt{h} : 87\}]], [[\mathtt{b}]], [[\mathtt{a.f}]]\}$;
- $\mathtt{Set}(\{\mathtt{f} : [[\mathtt{a.f}]], \mathtt{g} : \alpha_1, \mathtt{h} : \alpha_2\}) =$
  $\{\{\mathtt{f} : [[\mathtt{null}]], \mathtt{g} : [[17]], \mathtt{h} : \diamond\}, \{\mathtt{f} : [[\mathtt{a.f}]], \mathtt{g} : \alpha_1, \mathtt{h} : \alpha_2\},$
  $[[\{\mathtt{f} : \mathtt{null}, \mathtt{g} : 17\}]], [[\mathtt{a}]]\}$;

  1. $\mathtt{Set}(\uparrow [[\{\mathtt{g} : 42, \mathtt{h} : 87\}]]) =$
     $\{\uparrow [[\{\mathtt{g} : 42, \mathtt{h} : 87\}]], [[\mathtt{alloc}\ \{\mathtt{g} : 42, \mathtt{h} : 87\}]], [[\mathtt{b}]], [[\mathtt{a.f}]],$
     $[[\mathtt{null}]], \uparrow \beta\}$;

(a) $\texttt{Set}(\{\texttt{f}:\diamond, \texttt{g}:[[42]], \texttt{h}:[[87]]\}) =$
$\{\{\texttt{f}:\diamond, \texttt{g}:[[42]], \texttt{h}:[[87]]\}, [[\{\texttt{g}:42, \texttt{h}:87\}]], \beta\}$;

2. $\texttt{Set}(\texttt{int}) = \{\texttt{int}, [[17]], [[42]], [[87]], \alpha_1\}$;
3. $\texttt{Set}(\diamond) = \{\diamond, \alpha_2\}$;

- $\texttt{Set}(\texttt{int}) = \{\texttt{int}, [[17]], [[42]], [[87]], \alpha_1, [[117]], [[(*\texttt{b}).\texttt{g}]]\}$;
- $\texttt{Set}(\{\texttt{f}:\alpha_3, \texttt{g}:[[(*\texttt{b}).\texttt{g}]], \texttt{h}:\alpha_4\}) = \{\{\texttt{f}:\alpha_3, \texttt{g}:[[(*\texttt{b}).\texttt{g}]], \texttt{h}:\alpha_4\}, [[*\texttt{b}]]\}$;
- $\texttt{Set}(\uparrow [[*\texttt{b}]]) =$
$\{\uparrow [[\{\texttt{g}:42, \texttt{h}:87\}]], [[\texttt{alloc}\ \{\texttt{g}:42, \texttt{h}:87\}]], [[\texttt{b}]], [[\texttt{a.f}]],$
$[[\texttt{null}]], \uparrow \beta, \uparrow [[*\texttt{b}]]\}$;

1. $\texttt{Set}(\{\texttt{f}:\alpha_3, \texttt{g}:[[(*\texttt{b}).\texttt{g}]], \texttt{h}:\alpha_4\}) =$
$\{\{\texttt{f}:\alpha_3, \texttt{g}:[[(*\texttt{b}).\texttt{g}]], \texttt{h}:\alpha_4\}, [[*\texttt{b}]], \{\texttt{f}:\diamond, \texttt{g}:[[42]], \texttt{h}:[[87]]\},$
$[[\{\texttt{g}:42, \texttt{h}:87\}]], \beta\}$;
   (a) $\texttt{Set}(\diamond) = \{\diamond, \alpha_2, \alpha_3\}$;
   (b) $\texttt{Set}(\texttt{int}) = \{\texttt{int}, [[17]], [[42]], [[87]], \alpha_1, [[117]], [[(*\texttt{b}).\texttt{g}]], \alpha_4\}$;

- $\texttt{Set}([[(*(\texttt{a.f})).\texttt{g}]]) = \{[[(*(\texttt{a.f})).\texttt{g}]], [[\texttt{c}]]\}$;
- $\texttt{Set}(\{\texttt{f}:\alpha_5, \texttt{g}:[[(*(\texttt{a.f})).\texttt{g}]], \texttt{h}:\alpha_6\}) =$
$\{\{\texttt{f}:\alpha_5, \texttt{g}:[[(*(\texttt{a.f})).\texttt{g}]], \texttt{h}:\alpha_6\}, [[*(\texttt{a.f})]]\}$;
- $\texttt{Set}(\uparrow [[*(\texttt{a.f})]]) =$
$\{\uparrow [[\{\texttt{g}:42, \texttt{h}:87\}]], [[\texttt{alloc}\ \{\texttt{g}:42, \texttt{h}:87\}]], [[\texttt{b}]], [[\texttt{a.f}]],$
$[[\texttt{null}]], \uparrow \beta, \uparrow [[*\texttt{b}]], \uparrow [[*(\texttt{a.f})]]\}$;

1. $\texttt{Set}(\{\texttt{f}:\alpha_5, \texttt{g}:[[(*(\texttt{a.f})).\texttt{g}]], \texttt{h}:\alpha_6\}) =$
$\{\{\texttt{f}:\alpha_3, \texttt{g}:[[(*\texttt{b}).\texttt{g}]], \texttt{h}:\alpha_4\}, [[*\texttt{b}]], \{\texttt{f}:\diamond, \texttt{g}:[[42]], \texttt{h}:[[87]]\},$
$[[\{\texttt{g}:42, \texttt{h}:87\}]], \beta, \{\texttt{f}:\alpha_5, \texttt{g}:[[(*(\texttt{a.f})).\texttt{g}]], \texttt{h}:\alpha_6\}, [[*(\texttt{a.f})]]\}$;
   (a) $\texttt{Set}(\diamond) = \{\diamond, \alpha_2, \alpha_3, \alpha_5\}$;
   (b) $\texttt{Set}(\texttt{int}) =$
   $\{\texttt{int}, [[17]], [[42]], [[87]], \alpha_1, [[117]], [[(*\texttt{b}).\texttt{g}]], \alpha_4,$
   $[[(*(\texttt{a.f})).\texttt{g}]], [[\texttt{c}]], \alpha_6\}$;

Based on the above calculation, the types of critical variables are:

- a: $\{\texttt{f}:\uparrow \{\texttt{f}:\diamond, \texttt{g}:\texttt{int}, \texttt{h}:\texttt{int}\}, \texttt{g}:\texttt{int}, \texttt{h}:\diamond\}$
- a.f, b: $\uparrow \{\texttt{f}:\diamond, \texttt{g}:\texttt{int}, \texttt{h}:\texttt{int}\}$
- $(*\texttt{b}).\texttt{g}, (*(\texttt{a.f})).\texttt{g}, \texttt{c}$: int

Exercise 3.21 We define the type constraints as follows:

$$\{E_1, E_2, ..., E_n\} : \begin{cases} [[\{E_1, E_2, ..., E_n\}]] = [[E_1]][] \ \wedge\ [[E_1]] = ... = [[E_n]], & n > 0 \\ [[\{E_1, E_2, ..., E_n\}]] = \alpha[], & n = 0 \end{cases}$$

$$E_1[E_2] : [[E_2]] = \texttt{int} \ \wedge\ [[E_1]] = [[E_1[E_2]]][]$$

The type constraints are as follows:

$[[\texttt{x}]] = [[\{2, 4, 8, 16, 32, 64\}]]$
$[[\{2, 4, 8, 16, 32, 64\}]] = [[2]][]$
$[[2]] = [[4]] = [[8]] = [[16]] = [[32]] = [[64]]$
$[[2]] = \texttt{int}$
$[[4]] = \texttt{int}$
$[[8]] = \texttt{int}$
$[[16]] = \texttt{int}$
$[[32]] = \texttt{int}$
$[[64]] = \texttt{int}$
$[[\texttt{y}]] = [[\texttt{x}[\texttt{x}[3]]]]$
$[[\texttt{x}]] = [[\texttt{x}[\texttt{x}[3]]]][]$
$[[\texttt{x}[3]]] = \texttt{int}$
$[[\texttt{x}]] = [[\texttt{x}[3]]][]$
$[[3]] = \texttt{int}$
$[[\texttt{z}]] = [[\{\{\}, \texttt{x}\}]]$
$[[\{\{\}, \texttt{x}\}]] = [[\{\}]][]$
$[[\{\}]] = [[\texttt{x}]]$
$[[\{\}]] = \alpha_1[]$
$[[\texttt{t}]] = [[\texttt{z}[1]]]$
$[[\texttt{z}]] = [[\texttt{z}[1]]][]$
$[[1]] = \texttt{int}$
$[[\texttt{t}[2]]] = [[\texttt{y}]]$
$[[\texttt{t}]] = [[\texttt{t}[2]]][]$

The unification algorithm proceeds as follows:

- $\texttt{Set}([[\{2, 4, 8, 16, 32, 64\}]]) = \{[[\{2, 4, 8, 16, 32, 64\}]], [[\texttt{x}]]\}$;
- $\texttt{Set}([[2]][]) = \{[[2]][], [[\{2, 4, 8, 16, 32, 64\}]], [[\texttt{x}]]\}$;
- $\texttt{Set}([[64]]) = \{[[2]], [[4]], [[8]], [[16]], [[32]], [[64]]\}$;
- $\texttt{Set}(\texttt{int}) = \{\texttt{int}, [[2]], [[4]], [[8]], [[16]], [[32]], [[64]]\}$;
- $\texttt{Set}([[\texttt{x}[\texttt{x}[3]]]]) = \{[[\texttt{x}[\texttt{x}[3]]]], [[\texttt{y}]]\}$;
- $\texttt{Set}([[\texttt{x}[\texttt{x}[3]]]][]) = \{[[\texttt{x}[\texttt{x}[3]]]][], [[2]][], [[\{2, 4, 8, 16, 32, 64\}]], [[\texttt{x}]]\}$;

    1. $\texttt{Set}(\texttt{int}) = \{\texttt{int}, [[2]], [[4]], [[8]], [[16]], [[32]], [[64]], [[\texttt{x}[\texttt{x}[3]]]], [[\texttt{y}]]\}$;

- $\texttt{Set}(\texttt{int}) = \{\texttt{int}, [[2]], [[4]], [[8]], [[16]], [[32]], [[64]], [[\texttt{x}[\texttt{x}[3]]]], [[\texttt{y}]], [[\texttt{x}[3]]]\}$;
- $\texttt{Set}([[\texttt{x}[3]]][]) = \{[[\texttt{x}[3]]][], [[\texttt{x}[\texttt{x}[3]]]][], [[2]][], [[\{2, 4, 8, 16, 32, 64\}]], [[\texttt{x}]]\}$;
- $\texttt{Set}(\texttt{int}) =$
  $\{\texttt{int}, [[2]], [[4]], [[8]], [[16]], [[32]], [[64]], [[\texttt{x}[\texttt{x}[3]]]], [[\texttt{y}]], [[\texttt{x}[3]]], [[3]]\}$;
- $\texttt{Set}([[\{\{\}, \texttt{x}\}]]) = \{[[\{\{\}, \texttt{x}\}]], [[\texttt{z}]]\}$;
- $\texttt{Set}([[\{\}]][]) = \{[[\{\}]][], [[\{\{\}, \texttt{x}\}]], [[\texttt{z}]]\}$;
- $\texttt{Set}([[\texttt{x}[3]]][]) =$
  $\{[[\texttt{x}[3]]][], [[\texttt{x}[\texttt{x}[3]]]][], [[2]][], [[\{2, 4, 8, 16, 32, 64\}]], [[\texttt{x}]], [[\{\}]]\}$;

- $\texttt{Set}(\alpha_1[]) =$
  $\{[[\texttt{x}[3]]][], [[\texttt{x}[\texttt{x}[3]]]][], [[2]][], [[\{2,4,8,16,32,64\}]], [[\texttt{x}]], [[\{\}]], \alpha_1[]\};$

  1. $\texttt{Set}(\texttt{int}) =$
     $\{\texttt{int}, [[2]], [[4]], [[8]], [[16]], [[32]], [[64]], [[\texttt{x}[\texttt{x}[3]]]],$
     $[[\texttt{y}]], [[\texttt{x}[3]]], [[3]], \alpha_1\};$

- $\texttt{Set}([[\texttt{z}[1]]]) = \{[[\texttt{z}[1]]], [[\texttt{t}]]\};$
- $\texttt{Set}([[\texttt{z}[1]]][]) = \{[[\texttt{z}[1]]][], [[\{\}]][], [[\{\{\}, \texttt{x}\}]], [[\texttt{z}]]\};$

  1. $\texttt{Set}([[\texttt{z}[1]]]) =$
     $\{[[\texttt{z}[1]]], [[\texttt{t}]], [[\texttt{x}[3]]][], [[\texttt{x}[\texttt{x}[3]]]][], [[2]][], [[\{2,4,8,16,32,64\}]],$
     $[[\texttt{x}]], [[\{\}]], \alpha_1[]\};$

- $\texttt{Set}(\texttt{int}) =$
  $\{\texttt{int}, [[2]], [[4]], [[8]], [[16]], [[32]], [[64]], [[\texttt{x}[\texttt{x}[3]]]],$
  $[[\texttt{y}]], [[\texttt{x}[3]]], [[3]], \alpha_1, [[1]], [[\texttt{t}[2]]]\};$
- $\texttt{Set}([[\texttt{t}[2]]][]) =$
  $\{[[\texttt{z}[1]]], [[\texttt{t}]], [[\texttt{x}[3]]][], [[\texttt{x}[\texttt{x}[3]]]][], [[2]][], [[\{2,4,8,16,32,64\}]],$
  $[[\texttt{x}]], [[\{\}]], \alpha_1[], [[\texttt{t}[2]]][]\};$

Based on the above calculation, the types of critical variables are:

- $\texttt{x}, \texttt{t}, \texttt{z}[1]$: `int[]`
- $\texttt{y}, \texttt{x}[3], \texttt{x}[\texttt{x}[3]], \texttt{t}[2]$: `int`
- `z`: `int[][]`

Exercise 3.22 The following type rules should be changed:

$$E_1 == E_2 : [[E_1]] = [[E_2]] \ \wedge \ [[E_1 == E_2]] = \texttt{bool}$$

$$E_1 > E_2 : [[E_1]] = [[E_2]] = \texttt{int} \ \wedge \ [[E_1 > E_2]] = \texttt{bool}$$

$$\texttt{if} \ (E) \ S : [[E]] = \texttt{bool}$$

$$\texttt{if} \ (E) \ S_1 \ \texttt{else} \ S_2 : [[E]] = \texttt{bool}$$

$$\texttt{while} \ (E) \ S : [[E]] = \texttt{bool}$$

Exercise 3.24 If `x` equals to `0`, `polyrec` returns `g+1`; otherwise, `polyrec` returns `4`. Our type analysis will produce type constraint $[[\texttt{null}]] = [[2]]$ for this program, which is clearly unsolvable. The let-polymorphism does not help here because we need to know the type of `polyrec` when analyzing `r=polyrec(2,0)` to further determine the polymorphic type of `polyrec`, which yields a chicken-and-egg problem.