Formal proofs of asymptotic complexity

Armaël Guéneau
supervised by A. Charguéraud & F. Pottier

The setting

Consider:

- Higher-order imperative programs (OCaml)
- An interactive proof assistant (Coq)

Functional correctness

The *CFML* tool: sound and complete approach based on Separation Logic.

For example, a specification for a sorting function on arrays:

$$\{t \rightsquigarrow \mathrm{Array}\, L\}$$
$$\text{(sort t)}$$
$$\{\lambda().\, \exists L'.\ t \rightsquigarrow \mathrm{Array}\, L' \star [\text{permutation L L'} \wedge \text{sorted L'}]\}$$

Time complexity

Extend the logic with a new resource: time credits.

- "$n" asserts the ownership of *n* time credits
- Function calls and loop steps consume $1

$$\{t \rightsquigarrow \mathrm{Array}\, L \star \$(3|L| \log |L| + 2|L| + 7)\}$$
$$(\text{sort t})$$
$$\{\lambda().\, \exists L'.\ t \rightsquigarrow \mathrm{Array}\, L' \star [\text{permutation L L'} \wedge \text{sorted L'}]\}$$

A. Charguéraud, F. Pottier, *Machine-Checked Verification of the Correctness and Amortized Complexity of an Efficient Union-Find Implementation*

Amortized time complexity & modular specifications

Specifications with explicit credits count are not *modular*.

Just as in paper proofs, use the $O()$ notation to introduce abstraction.

$\exists$ sort_cost,

$\{t \rightsquigarrow \mathrm{Array}\, L \star \$(\mathrm{sort\_cost}\ |L|)\}$
(sort t)
$\{\lambda().\ \exists L'.\ \ t \rightsquigarrow \mathrm{Array}\, L' \star [\text{permutation L L' } \wedge \text{ sorted L'}]\} \wedge$

$\mathrm{sort\_cost} \in O(\lambda n.\, n \log n)$

Formalizing $O()$ in Coq

Standard definition, from e.g. Cormen et al:

$$f(n) \in O(g(n)) \equiv \exists c, \exists n_0, \forall n \geqslant n_0, |f(n)| \leqslant c \times |g(n)|$$

Are we done here?

The case of multivariate $O()$

What does "$f(m,n) \in O(g(m,n))$" mean?

One possible answer:

$$\exists c\, n_0, \forall m\, n, m \geqslant n_0 \wedge n \geqslant n_0 \Rightarrow |f(m,n)| \leqslant c \times |g(m,n)|$$

But then desirable properties do not always hold.

Other possibility: use an alternative definition of $O()$.
Typically has nicer properties, but is harder to prove directly.

R. Howell, *On Asymptotic Notation with Multiple Variables*

Reusing multivariate $O()$ specifications

function R(m,n)
    for $i \leftarrow 0$ **to** m
        for $j \leftarrow 0$ **to** n
            ()
        done
    done
end

$$R(m, n) \in O(mn)$$

What is the asymptotic complexity of R(0,n)?

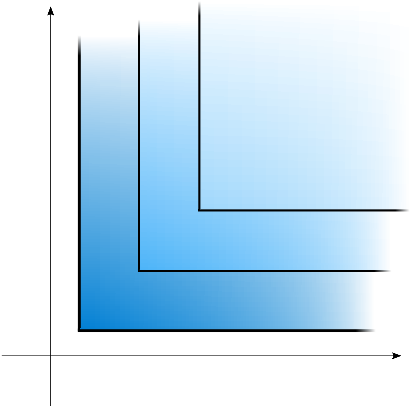
- Not $O(0n)$ (i.e. $O(0)$)
- We cannot deduce it from the previous specification

Can we prove a specification we can instantiate with $m = 0$?

```
function R(m,n)
    for i ← 0 to m
        for j ← 0 to n
            ()
        done
    done
end
```

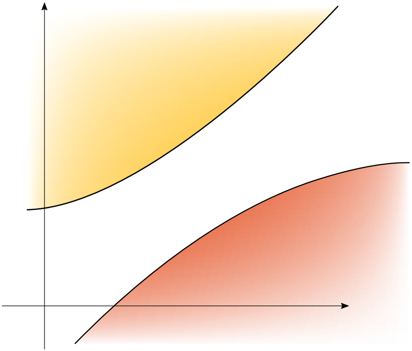
$$\forall m, R(m,n) \in O(mn + n)$$
$$\forall n, R(m,n) \in O(mn + m)$$

Multivariate $O()$ specifications



$$R(m, n) \in O(mn)$$

Multivariate $O()$ specifications



$$\forall m, R(m,n) \in O(mn+n)$$
$$\forall n, R(m,n) \in O(mn+m)$$

Question:

Is there a most general specification for a multivariate function?

Infrastructure for proving asymptotic costs

Proving a program specification with $O()$

$\exists$ sort_cost,

$\{t \rightsquigarrow \text{Array}\, L \star \$(\text{sort\_cost}\ |L|)\}$
(sort t)
$\{\lambda().\, \exists L'.\ \ t \rightsquigarrow \text{Array}\, L' \star [\text{permutation L L' } \wedge \text{ sorted L'}]\} \wedge$

sort_cost $\in O(\lambda n.\, n \log n)$

Provide "sort_cost $= \lambda n.\, 3n \log n + 2n + 7$" upfront?

- Tactics to automatically and interactively infer cost functions
- Future work: for recursive functions, infer recurrence equations, then feed them to a “Master Theorem”

Theoretical foundations

Strengthen the theoretical foundations of time credits

- Time credits count function calls and loop iterations
- Unspecified gap between these and wall-clock time

Future work (long term project):

- In the line of previous work on CakeML
- Prove an end-to-end compiler theorem relating time credits and execution time

A. Guéneau, M. Myreen, R. Kumar, M. Norrish, *Verified characteristic formulae for CakeML*

function F(m,n)
    for $i \leftarrow 0$ **to** $m - 1$
        G(i, n)
    done
end

$$G(i,n) \in O(in)$$
$$\overset{?}{\Rightarrow} F(m,n) \in O(m^2 n)$$

Does not hold e.g. for:

$$G(m,n) = \begin{cases} 2^n & \text{if } m = 0 \\ mn & \text{if } m \geqslant 1 \end{cases}$$