Amortised analysis of heap consumption

Olha Shkaravska

Institut of Informatics, LMU
Munich, Germany

Motivation and Structure of this talk

Hofmann-Jost inference system:
inference of linear bounds of heap consumption.

What about non-linear bounds?

The talk:

- Amortized analysis for time, incl. Banker's algorithm
- Hofmann-Jost analysis = a Banker's algorithm with constant credits
- A Banker's algorithm for dependent credits

Amortised Time Analysis

Idea: to distribute the worst-case run time of an entire sequence of operations over the operations.

Given: a sequence of n operations.

Let a_i and t_i be the amortized and actual costs

of i-th operation.

$$\Sigma_{i=1}^{j} a_i \geq \Sigma_{i=1}^{j} t_i,$$

where $1 \leq j \leq n$.

Aggregate method

$$a_i = \frac{T(n)}{n}$$

... like we have payed in the African restaurant.

Banker's (Accounting) method

If

$$a_i \geq t_i$$

then $c_i = a_i - t_i$ is viewed as a credit.

It can be used late to pay for the operations whose amortised cost is less then their actual cost.

Example.

```
while not StackEmpty(S) and k<>0
do {
   Pop(S)
   k:= k-1
   }
```

Banker's (Accounting) method

```
while not StackEmpty(S) and k<>0
do { Pop(S); k:= k-1 }
```

The actual costs, t_i-s:

`Push`	1
`Pop`	1
`Multipop`	$min(s, k)$,

where s is a size of the stack S.

The amortized costs, a_i-s:

`Push`	2
`Pop`	0
`Multipop`	0

Physicist's (Potential) method

One can associate all "prepayment" with the data structure as a whole.

Data structures: $D_0, \ \ldots, \ D_n$:

- D_0 is an initial one,
- D_i is a result of application of i-th operation on D_{i-1}

Find a potential function $\Phi : \ D_i \ \mapsto \ \Phi(D_i)$, a number.

The amortised cost per op.:

$$a_i = t_i + \big(\Phi(D_i) - \Phi(D_{i-1})\big)$$

Physicist's (Potential) method

The amortised cost per op.:

$$a_i = t_i + \Big(\Phi(D_i) - \Phi(D_{i-1})\Big)$$

The total amortized cost is

$$\begin{aligned}\Sigma_{i=1}^{n} a_i &= \Sigma_{i=1}^{n} t_i + \Sigma_{i=1}^{n} \Big(\Phi(D_i) - \Phi(D_{i-1})\Big) \\ &= \Sigma_{i=1}^{n} t_i + \Big(\Phi(D_n) - \Phi(D_0)\Big)\end{aligned}$$

Hofmann-Jost inference system

We can infer linear heap-consumption bounds:

- Given

$$f : \mathtt{L}(\mathtt{Int}) \rightarrow \mathtt{L}(\mathtt{Int})$$

- Obtain a notated, with numbers, signature

$$f : \mathtt{L}(\mathtt{Int},\ k),\ k_0 \rightarrow \mathtt{L}(\mathtt{Int},\ k'),\ k'_0$$

Examples:

$\mathtt{copy} : \mathtt{L}(\mathtt{Int},\ 1),\ 0 \rightarrow \mathtt{L}(\mathtt{Int},\ 0),\ 0$
$\mathtt{cons} : \mathtt{L}(\mathtt{Int},\ 0),\ 1 \rightarrow \mathtt{L}(\mathtt{Int},\ 0),\ 0$

k is a constant credit, $k\,|l|$ is a potential of the list l.

Type system for dependent credits

$$\mathtt{B} = \{0,\ 1\}$$

$$T ::= \mathtt{B} \big| \mathtt{L}_0(T,\ k) \big| \ \ldots\ \big| \ \mathtt{L}_m(T,\ k) \big| \ \ldots\ \big| \mathtt{L}(T,\ k),$$

where

- $k : \mathtt{Nat} \longrightarrow \mathbb{R}^+$,
- $\mathtt{L}_m(T,\ k)$ is a not. list of length m of type T, s. t. i-th element of the list has a credit $k(i)$,
- $\mathtt{L}(T,\ k) = \Sigma_{n=0}^{\infty} \mathtt{L}_n(T,\ k)$.

Inference for dependent credits

Typing judgment is almost the same as for HJ typing:
$\Gamma,\ n\ \vdash\ e\ :\ T,\ n'$

The context is mixed: with non-sized and sized types.

$$\frac{n \geq n' + 1 + k(m+1)}{h : T,\ t : \mathtt{L}_m(T,\ k),\ n \vdash}\ \mathtt{Cons}$$

$$\mathtt{cons}(h,\ t) : \mathtt{L}_{m+1}(T,\ k),\ n'$$

Inference for dependent credits

$$\frac{\begin{array}{c}\Gamma,\ n\ \vdash\ e_1 :\ A,\ n' \\ \Gamma,\ h : T, \\ t :\ \mathtt{L}_{m-1}(T,\ k),\ n+1+k(m) \vdash\ e_2 : A,\ n'\end{array}}{\Gamma,\ l : \mathtt{L}_m(T,\ k),\ n\ \vdash}\ \mathtt{DM}$$

$$\begin{array}{ll}\mathtt{match}\ l\ \mathtt{with} & \\ \mathtt{Nil}\ \Rightarrow\ e_1 : & A,\ n' \\ \mid \mathtt{Cons}@(h,\ t)\ \Rightarrow\ e_2 : & \end{array}$$

Inference for dependent credits

The rule

$$\frac{\Gamma,\ n\ \vdash\ e_1 : A,\, n' \qquad \Gamma,\ n\ \vdash\ e_2 : A,\, n'}{\Gamma,\ x : \mathtt{B},\ n\ \vdash\ \mathtt{if}\ \mathtt{x}\ \mathtt{then}\ e_1\ \mathtt{else}\ e_2 : A,\, n'}\ \mathtt{If}$$

perhaps, is not that restrictive if we have

$$\frac{\Gamma,\ n\ \vdash\ e : \mathtt{L}_m(T,\, k),\, n'}{\Gamma,\ n\ \vdash\ e : \mathtt{L}(T,\, k),\, n'}\ \mathtt{Sum}$$

Inference for dependent credits

$$\frac{\begin{array}{c}\Sigma(P) = \mathtt{L}(T,\ k),\ k_0\ \rightarrow^p \\ \mathtt{L}\big(T',\ k'\big),\ k_0' \\ n \geq k_0 \\ n - k_0 \geq n' - k_0'\end{array}}{\begin{array}{c}\Gamma,\ l:\ \mathtt{L}_m(T,\ k),\ n\ \vdash \\ P(l):\ \mathtt{L}_{p(m)}\big(T',\ k'\big),\ n'\end{array}}\ \mathtt{Fun}$$

Inference for dependent credits

$$\frac{\Sigma(P) = \mathtt{L}(T,\, k),\ k_0 \ \longrightarrow^{p} \ \mathtt{L}\big(T',\, k'\big),\ k'_0}{l\,:\ \mathtt{L}_m(T,\, k),\ k_0 \ \vdash \ e_P\,:\ \mathtt{L}_{p(m)}\big(T',\, k'\big),\ k'_0}\ \mathsf{Spec}$$

Checking heap bounds

Given a program of $\mathtt{L}(T) \;\longrightarrow\; \mathtt{L}(T')$.

How to check, if its heap consumption does not exceed $O(f(x))$, where x is a length of an input list, and $f(x)$ is smooth?

Notate the signature with functions of $k, k' : \mathtt{Nat} \rightarrow \mathbb{R}^+$ and nat. numbers k_0, k_0':

$\mathtt{L}(T,\ k),\ k_0 \longrightarrow\ \mathtt{L}(T',\ k'),\ k_0'$

Take $k = f'$

Checking heap bounds $f(x)$

Take $k = f'$ in $\mathtt{L}(T,\ k),\ k_0 \longrightarrow\ \mathtt{L}(T',\ k'),\ k'_0$
If type-checking for this k and some nonnegative k_0, k', k'_0 works (a bit of type-inference for k_0, k', k'_0),
then the program consumes up to $O(f(x))$ heap units.

Why?

- $f(x) = \int_0^x f'(v)\, d\, v + f(0)$
- $\sum_{v=1}^{x} k(v)$ is a total amount of free heap units associated with an input list of length x
- approximate the integral by the sum $\sum_{v=1}^{x} k(v)$, rectang. approx. of the square:
  $|\sum_{v=1}^{x} k(v) - f(x)| \leq C$.

Examples

- $f(x) = x$ for `copy`: we have $k(x) = f'(x) \equiv 1$
- $f(x) = a\log(x+b) + c$ and $k(x) = \frac{a}{\ln 2}\frac{1}{x+b}$ for `binary`

```
binary  l =
   match l with
   Nil => Nil
 | Cons(h, t) => let y = binary t
                 in binInc y
```

where ....

To generalise type inference

```
binInc  l =
   match l with
   Nil => Cons(1, Nil)
 | Cons@(h, t) => if h=0 then
                   Cons(1, t)
                   else
                   Cons(0, binInc t)
```

If $\|l\| = 2^s - 1$, for some natural number s, `binInc` consumes exactly one heap unit, otherwise there is no consumption.

Consider another measure $\mu = \| \cdot \|$.

Consumption on measure

$$\begin{aligned} \texttt{consume}(l) &= f(\mu(l)) \\ &= \int_0^{\mu(l)} f'_\mu(v)\, d\, v + f(0) \\ &\approx \sum_{v=1}^{\mu(l)} f'_\mu(v) + C \end{aligned}$$

Generally: a credit in heap units is payed pro 1 unit of growth of measure.

Consumption on measure: the example to do

`binInc`

- $\mu = \| \cdot \|$,
- $f(x) = \lceil log_2(x+2) \rceil - \lceil log_2(x+1) \rceil$,
- $k(x) = \frac{a}{\ln 2}\frac{1}{x+2} - \frac{b}{\ln 2}\frac{1}{x+1}$
- $\mathtt{L}_d(\mathtt{B},\ k),\ 0 \longrightarrow \mathtt{L}_{d+1}(\mathtt{B},\ 0),\ 0$,

  where d is a measure of an input.

Functions of 2 arguments (back to length)

$k_1,\ k_2 : \mathtt{Nat} \to \mathtt{Nat} \to \mathbb{R}^+$
k_i: length of the partner $\to$ position of the element $\to$ credit

$$\frac{\begin{array}{c}\Sigma(P) = \mathtt{L}(T_1,\ k_1),\ \mathtt{L}(T_1,\ k_2),\ k_0 \ \to^p \\ \mathtt{L}\big(T',\ k'\big),\ k'_0 \\ n \geq k_0 \\ n - k_0 \geq n' - k'_0\end{array}}{\begin{array}{c}\Gamma,\ l_1 : \mathtt{L}_{m_1}(T,\ k_1(m_2)),\ l_2 : \mathtt{L}_{m_2}(T,\ k_2(m_1)), n \vdash \\ P(l_1,\ l_2) : \mathtt{L}_{p(m_1, m_2)}\big(T',\ k'\big),\ n'\end{array}}$$

Functions of 2 arguments

$$
\begin{aligned}
f(x, y) &= \int_0^y f'_y(x,\, u)\, d\, u + f(x,\, 0) \\
&= \int_0^y f'_y(x,\, u)\, d\, u + \int_0^x f'_{0\ x}(v) dv \\
&+ f_0(0) \\
&\approx \sum_{u=1}^y f'_y(x,\, u) + \sum_{v=1}^x f'_{0\ x}(v)
\end{aligned}
$$

where $f_0 := f(x,\, 0)$

Functions of 2 arguments

$$f(x,y) \approx \sum_{u=1}^{y} f'_y(x,\ u) + \sum_{v=1}^{x} f'_{0\ x}(v)$$

with $f_0 := f(x,\ 0)$

Let

$$k_1(y) = f'_{0\ x}$$
$$k_2(x) = f'_y(x,\ y)$$

Example of bounds

$$f(x,\ y) = a\,x\,y + b\,x + c\,y + d\ ?$$

How to answer this question?

$$k_1(y) = f'_{0\,x} = b$$

$$k_2(x) = f'_y(x,\ y) = ax + c$$

Find such $a,\ b,\ c,\ d$ that for some $k_0,\ k',\ k'_0$ type-checking works...

Let for simplicity $k' \equiv 0$, $k'_0 = 0$.

Example: Multiplication

```
mult(l1, l2) =
match l2 with
    Nil => Nil
    | Cons(h, t) => let x=mult(l1, t)
                    in
                    let l=copy l1
                    in
                    cons(l, x)
```

Typechecking works with
$a = 1,\ b = 0,\ c = 1,\ k_0 = 0$

$$\mathtt{L}(\mathtt{B},\ 0),\ \mathtt{L}(\mathtt{B},\ x+1),\ 0 \longrightarrow\ \mathtt{L}((\mathtt{B},\ 0),\ 0),\ 0$$

To Do

- Design an Inference system parametric w.r.t. measures
- The Examples: type-checking revisited
- Soundness of the inference system w.r.t. op.sem of Hofmann-Jost