

Exam for Programming Language Principals, Design and Implementation (Extended)

ID 1803086

After inserting your student ID and the module title in the preamble, write your answers below.

Question 1

(a)

$$M = \lambda f : \mathbb{B} \rightarrow \mathbb{B}. \lambda g : \mathbb{B} \rightarrow \mathbb{B}. \lambda x : \mathbb{B}. \lambda y : \mathbb{B}. \text{if } x \text{ then } fy \text{ else } gy$$

$$\begin{array}{c}
 \text{(i)} \\
 \frac{\frac{\frac{\Gamma \vdash x : \mathbb{B}}{\Gamma \vdash x : \mathbb{B}} \text{VAR} \quad \frac{\frac{\frac{\Gamma \vdash f : \mathbb{B} \rightarrow \mathbb{B}}{\Gamma \vdash f : \mathbb{B} \rightarrow \mathbb{B}} \text{VAR} \quad \frac{\Gamma \vdash y : \mathbb{B}}{\Gamma \vdash y : \mathbb{B}} \text{VAR}}{\Gamma \vdash fy : \mathbb{B}} \text{APP} \quad \Pi_1}{\Gamma, y : \mathbb{B} \vdash \text{if } x \text{ then } fy \text{ else } gy : \mathbb{B}} \text{ABS}}{\Gamma, x : \mathbb{B} \vdash \lambda y : \mathbb{B}. \text{if } x \text{ then } fy \text{ else } gy : \mathbb{B} \rightarrow \mathbb{B}} \text{ABS} \\
 \frac{\Gamma, g : \mathbb{B} \rightarrow \mathbb{B} \vdash \lambda x : \mathbb{B}. \lambda y : \mathbb{B}. \text{if } x \text{ then } fy \text{ else } gy : \mathbb{B} \rightarrow \mathbb{B} \rightarrow \mathbb{B}}{\Gamma \vdash \lambda x : \mathbb{B}. \lambda y : \mathbb{B}. \text{if } x \text{ then } fy \text{ else } gy : \mathbb{B} \rightarrow \mathbb{B} \rightarrow \mathbb{B}} \text{ABS} \\
 \frac{f : \mathbb{B} \rightarrow \mathbb{B} \vdash M_1 : (\mathbb{B} \rightarrow \mathbb{B}) \rightarrow \mathbb{B} \rightarrow \mathbb{B} \rightarrow \mathbb{B}}{\{\} \vdash M : (\mathbb{B} \rightarrow \mathbb{B}) \rightarrow (\mathbb{B} \rightarrow \mathbb{B}) \rightarrow \mathbb{B} \rightarrow \mathbb{B} \rightarrow \mathbb{B}} \text{ABS}
 \end{array}$$

Where:

- $M_1 = \lambda g : \mathbb{B} \rightarrow \mathbb{B}. \lambda x : \mathbb{B}. \lambda y : \mathbb{B}. \text{if } x \text{ then } fy \text{ else } gy$
- $\Pi_1 =$

$$\frac{\frac{\Gamma \vdash g : \mathbb{B} \rightarrow \mathbb{B}}{\Gamma \vdash g : \mathbb{B} \rightarrow \mathbb{B}} \text{VAR} \quad \frac{\Gamma \vdash y : \mathbb{B}}{\Gamma \vdash y : \mathbb{B}} \text{VAR}}{\Gamma \vdash gy : \mathbb{B}} \text{APP}$$

- (ii) To produce the exclusive function from M we can define the first order parameters F and G as follows:

$$F = \lambda y : \mathbb{B}. \text{if } y \text{ then false else true} : \mathbb{B} \rightarrow \mathbb{B}$$

$$G = \lambda y : \mathbb{B}. \text{if } y \text{ then true else false} : \mathbb{B} \rightarrow \mathbb{B}$$

Alternatively, G can simply be defined as the boolean identity function $\lambda y : \mathbb{B}. y : \mathbb{B} \rightarrow \mathbb{B}$. This is the definition I will use in later parts of the question.

- (iii) The fact that this expression is well-typed under the Simply-typed λ -Calculus means

Finish this question

(iv)

$$\begin{array}{c}
\frac{\lambda f : \mathbb{B} \rightarrow \mathbb{B}. M_1 F \rightarrow_v \lambda g : \mathbb{B} \rightarrow \mathbb{B} \lambda x : \mathbb{B} \text{ if } x \text{ then } Fy \text{ else } gy}{\lambda f : \mathbb{B} \rightarrow \mathbb{B}. M_1 FG \rightarrow_v} \beta \\
\frac{(\lambda g : \mathbb{B} \rightarrow \mathbb{B} \lambda x : \mathbb{B} \text{ if } x \text{ then } Fy \text{ else } gy)G}{(MFG)\text{false true} \rightarrow_v} \text{CTX}_{\bullet} G \\
\frac{}{((\lambda g : \mathbb{B} \rightarrow \mathbb{B} \lambda x : \mathbb{B} \text{ if } x \text{ then } Fy \text{ else } gy)G)\text{false true}} \text{CTX}_{(\bullet)\text{false true}}
\end{array} \quad (1)$$

$$\begin{array}{c}
\frac{(\lambda g : \mathbb{B} \rightarrow \mathbb{B} \lambda x : \mathbb{B} \text{ if } x \text{ then } Fy \text{ else } gy)G \rightarrow_v}{\lambda x : \mathbb{B}. \lambda y : \mathbb{B}. \text{ if } x \text{ then } Fy \text{ else } Gy} \beta \\
\frac{((\lambda g : \mathbb{B} \rightarrow \mathbb{B} \lambda x : \mathbb{B} \text{ if } x \text{ then } Fy \text{ else } gy)G)\text{false true} \rightarrow_v}{(\lambda x : \mathbb{B}. \lambda y : \mathbb{B}. \text{ if } x \text{ then } Fy \text{ else } Gy)\text{false true}} \text{CTX}_{(\bullet)\text{false true}}
\end{array} \quad (2)$$

$$\begin{array}{c}
\frac{(\lambda x : \mathbb{B}. \lambda y : \mathbb{B}. \text{ if } x \text{ then } Fy \text{ else } Gy)\text{false} \rightarrow_v}{\lambda y : \mathbb{B} \text{ if false then } Fy \text{ else } Gy} \beta \\
\frac{(\lambda x : \mathbb{B}. \lambda y : \mathbb{B}. \text{ if } x \text{ then } Fy \text{ else } gy)\text{false true} \rightarrow_v}{(\lambda y : \mathbb{B} \text{ if false then } Fy \text{ else } Gy)\text{true}} \text{CTX}_{\bullet \text{true}}
\end{array} \quad (3)$$

$$\frac{(\lambda y : \mathbb{B} \text{ if false then } Fy \text{ else } Gy)\text{true} \rightarrow_v}{\text{if false then } F\text{true} \text{ else } G\text{true}} \beta \quad (4)$$

$$\frac{}{\text{if false then } F\text{true} \text{ else } G\text{true} \rightarrow_v G\text{true}} \text{IteF} \quad (5)$$

$$\frac{}{(\lambda y : \mathbb{B}. y)\text{true} \rightarrow_v \text{true}} \beta \quad (6)$$

Where:

- M_1 is as defined the same as above.

$\text{true} \in V, \therefore (MFG)\text{false true}$ computes to a value.

(b)

$$\text{Stack} = \forall \alpha. (\mathbb{N} \rightarrow \alpha \rightarrow \alpha) \rightarrow \alpha \rightarrow \alpha$$

(i)

$$\lambda \alpha. \lambda f : \mathbb{N} \rightarrow \alpha \rightarrow \alpha. \lambda x : \alpha. f0(f0(f1x))$$

(ii)

$$\text{peek} = \lambda d : \mathbb{N}. \lambda s : \text{Stack}. s\{\mathbb{N} \rightarrow \mathbb{N}\} G / d$$

Where:

- $G = \lambda n : \mathbb{N}. \lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}. gn$
- $I = \lambda x : \mathbb{N}. x$

$$\begin{array}{c}
\frac{\frac{\Pi_1 \quad \Pi_2}{\Gamma \vdash s\{\mathbb{N} \rightarrow \mathbb{N}\}G : (\mathbb{N} \rightarrow \mathbb{N}) \rightarrow (\mathbb{N} \rightarrow \mathbb{N})} \text{APP} \quad \frac{\frac{\overline{\Gamma, x : \mathbb{N} \vdash x : \mathbb{N}} \text{VAR}}{\Gamma \vdash l : \mathbb{N} \rightarrow \mathbb{N}} \text{ABS}}{\Gamma \vdash ((s\{\mathbb{N} \rightarrow \mathbb{N}\}G)l) : \mathbb{N} \rightarrow \mathbb{N}} \text{APP} \quad \frac{\overline{\Gamma \vdash d : \mathbb{N}} \text{VAR}}{\Gamma, s : \text{Stack} \vdash ((s\{\mathbb{N} \rightarrow \mathbb{N}\}G)l)d : \mathbb{N} \rightarrow \text{Stack} \rightarrow \mathbb{N} : \mathbb{N}} \text{APP} \\
\frac{\Gamma, s : \text{Stack} \vdash ((s\{\mathbb{N} \rightarrow \mathbb{N}\}G)l)d : \mathbb{N} \rightarrow \text{Stack} \rightarrow \mathbb{N} : \mathbb{N}}{d : \mathbb{N} \vdash \lambda s : \text{Stack}. s\{\mathbb{N} \rightarrow \mathbb{N}\}G l d : \mathbb{N} \rightarrow \text{Stack} \rightarrow \mathbb{N} : \text{Stack} \rightarrow \mathbb{N}} \text{ABS} \\
\frac{d : \mathbb{N} \vdash \lambda s : \text{Stack}. s\{\mathbb{N} \rightarrow \mathbb{N}\}G l d : \mathbb{N} \rightarrow \text{Stack} \rightarrow \mathbb{N} : \text{Stack} \rightarrow \mathbb{N}}{\{\} \vdash \lambda d : \mathbb{N}. \lambda s : \text{Stack}. s\{\mathbb{N} \rightarrow \mathbb{N}\}G l d : \mathbb{N} \rightarrow \text{Stack} \rightarrow \mathbb{N}} \text{ABS}
\end{array}$$

Where:

- $\Pi_1 =$

$$\frac{\overline{\Gamma \vdash s : \text{State}} \text{VAR}}{\Gamma \vdash s\{\mathbb{N} \rightarrow \mathbb{N}\} : (\mathbb{N} \rightarrow (\mathbb{N} \rightarrow \mathbb{N}) \rightarrow (\mathbb{N} \rightarrow \mathbb{N})) \rightarrow (\mathbb{N} \rightarrow \mathbb{N}) \rightarrow (\mathbb{N} \rightarrow \mathbb{N})} \text{TAPP}$$

- $\Pi_2 =$

$$\begin{array}{c}
\frac{\overline{\Gamma \vdash g : \mathbb{N} \rightarrow \mathbb{N}} \text{VAR} \quad \overline{\Gamma \vdash n : \mathbb{N}} \text{VAR}}{\Gamma, x : \mathbb{N} \vdash gn : \mathbb{N}} \text{APP} \\
\frac{\Gamma, g : \mathbb{N} \rightarrow \mathbb{N} \vdash \lambda x : \mathbb{N}. gn : \mathbb{N} \rightarrow \mathbb{N}}{\Gamma, n : \mathbb{N} \vdash \lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}. gn : (\mathbb{N} \rightarrow \mathbb{N}) \rightarrow \mathbb{N} \rightarrow \mathbb{N}} \text{ABS} \\
\frac{\Gamma, n : \mathbb{N} \vdash \lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}. gn : (\mathbb{N} \rightarrow \mathbb{N}) \rightarrow \mathbb{N} \rightarrow \mathbb{N}}{\Gamma \vdash G : \mathbb{N} \rightarrow (\mathbb{N} \rightarrow \mathbb{N}) \rightarrow \mathbb{N} \rightarrow \mathbb{N}} \text{ABS}
\end{array}$$

(iii) Prove:

$$\text{peek } ds_2 \rightarrow_v^* m$$

$$\begin{array}{c}
\frac{(\lambda d : \mathbb{N}. \lambda s : \text{Stack}. s\{\mathbb{N} \rightarrow \mathbb{N}\}G l d)d \rightarrow_v \lambda s : \text{Stack}. s\{\mathbb{N} \rightarrow \mathbb{N}\}G l d}{\text{peek } ds_2 \rightarrow_v (\lambda s : \text{Stack}. s\{\mathbb{N} \rightarrow \mathbb{N}\}G l d)s_2} \beta \text{CTX}_{\bullet s_2} \\
\frac{(\lambda s : \text{Stack}. s\{\mathbb{N} \rightarrow \mathbb{N}\}G l d)s_2 \rightarrow_v s_2\{\mathbb{N} \rightarrow \mathbb{N}\}G l d}{(\lambda s : \text{Stack}. s\{\mathbb{N} \rightarrow \mathbb{N}\}G l d)s_2 \rightarrow_v s_2\{\mathbb{N} \rightarrow \mathbb{N}\}G l d} \beta \\
\frac{\frac{\lambda \alpha. \lambda f : \mathbb{N} \rightarrow \alpha \rightarrow \alpha. \lambda x : \alpha. fn(fmx)\{\mathbb{N} \rightarrow \mathbb{N}\} \rightarrow_v}{\lambda f : \mathbb{N} \rightarrow (\mathbb{N} \rightarrow \mathbb{N}) \rightarrow (\mathbb{N} \rightarrow \mathbb{N}). \lambda x : (\mathbb{N} \rightarrow \mathbb{N}). fn(fmx)} T_\beta}{((s_2\{\mathbb{N} \rightarrow \mathbb{N}\}G)l)d \rightarrow_v} \text{CTX}_{(\bullet G)l}d \\
\frac{((\lambda f : \mathbb{N} \rightarrow (\mathbb{N} \rightarrow \mathbb{N}) \rightarrow (\mathbb{N} \rightarrow \mathbb{N}). \lambda x : \mathbb{N} \rightarrow \mathbb{N}. fn(fmx))G)l)d}{((\lambda f : \mathbb{N} \rightarrow (\mathbb{N} \rightarrow \mathbb{N}) \rightarrow (\mathbb{N} \rightarrow \mathbb{N}). \lambda x : \mathbb{N} \rightarrow \mathbb{N}. fn(fmx))G)l)d} \beta \\
\frac{(\lambda f : \mathbb{N} \rightarrow (\mathbb{N} \rightarrow \mathbb{N}) \rightarrow (\mathbb{N} \rightarrow \mathbb{N}). \lambda x : \mathbb{N} \rightarrow \mathbb{N}. fn(fmx))G \rightarrow_v}{\lambda x : \mathbb{N} \rightarrow \mathbb{N}. Gn(Gmx)} \beta \\
\frac{((\lambda f : \mathbb{N} \rightarrow (\mathbb{N} \rightarrow \mathbb{N}) \rightarrow (\mathbb{N} \rightarrow \mathbb{N}). \lambda x : \mathbb{N} \rightarrow \mathbb{N}. fn(fmx))G)l)d \rightarrow_v}{((\lambda x : \mathbb{N} \rightarrow \mathbb{N}. Gn(Gmx))l)d} \text{CTX}_{((\bullet)l)d} \\
\frac{(\lambda x : \mathbb{N} \rightarrow \mathbb{N}. Gn(Gmx))l \rightarrow_v Gn(Gml)}{((\lambda x : \mathbb{N} \rightarrow \mathbb{N}. Gn(Gmx))l)d \rightarrow_v (Gn(Gml)d)} \beta \text{CTX}_{(\bullet)d} \\
\frac{(\lambda n : \mathbb{N}. \lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}. gn)n \rightarrow_v \lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}. gn}{(Gn(Gml)d) \rightarrow_v ((\lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}. gn)(Gml))d} \beta \text{CTX}_{(\bullet(Gml))d}
\end{array}$$

$$\begin{array}{c}
\frac{(\lambda n : \mathbb{N} \lambda g : \mathbb{N} \rightarrow \mathbb{N} \lambda x : \mathbb{N}.gn)m \rightarrow_v \lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}.gm}{((\lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}.gn)(Gml))d \rightarrow_v ((\lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}.gn)((\lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}.gm)l))d} \beta \text{CTX}_{((\lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}.gn)(\bullet l))d} \\
\\
\frac{(\lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}.gm)l \rightarrow_v \lambda x : \mathbb{N}.lm}{((\lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}.gn)((\lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}.gm)l))d \rightarrow_v ((\lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}.gn)(\lambda x : \mathbb{N}.lm))d} \beta \text{CTX}_{((\lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}.gn)(\bullet))d} \\
\\
\frac{(\lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}.gn)(\lambda x : \mathbb{N}.lm) \rightarrow_v \lambda x : \mathbb{N}.(\lambda x : \mathbb{N}.lm)n}{((\lambda g : \mathbb{N} \rightarrow \mathbb{N}. \lambda x : \mathbb{N}.gn)(\lambda x : \mathbb{N}.lm))d \rightarrow_v (\lambda x : \mathbb{N}.(\lambda x : \mathbb{N}.lm)n)d} \beta \text{CTX}_{(\bullet)d} \\
\\
\frac{(\lambda x : \mathbb{N}.(\lambda x : \mathbb{N}.lm)n)d \rightarrow_v (\lambda x : \mathbb{N}.lm)n}{(\lambda x : \mathbb{N}.lm)n \rightarrow_v lm} \beta \\
\\
\frac{lm \equiv (\lambda x : \mathbb{N}.x)n \rightarrow_v m}{lm \equiv (\lambda x : \mathbb{N}.x)n \rightarrow_v m} \beta
\end{array}$$

(iv) an abstract *stack* datatype can be defined as follows:

```

pack <Stack, <s0, <push, <peek, pop>>>>
  as
  ∃stack.stack × (ℕ → stack → stack ×
    (stack → ℕ → ℕ × stack → stack))

```

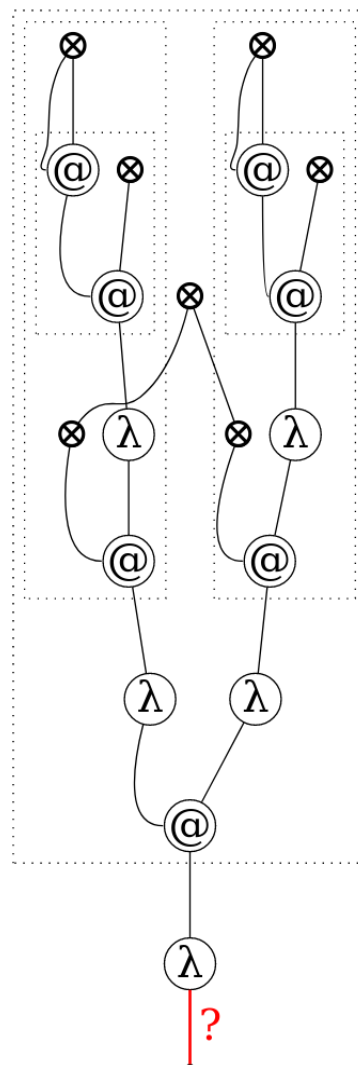
Where:

- $s_0 = \lambda \alpha. \lambda f : \mathbb{N} \rightarrow \alpha \rightarrow \alpha. \lambda x : \alpha. x$
- $\text{Stack} = \forall \alpha. (\mathbb{N} \rightarrow \alpha \rightarrow \alpha) \rightarrow \alpha \rightarrow \alpha$
- `push`, `peek` and `pop` are defined the same as in the exam booklet, over (concrete) the `Stack` type.

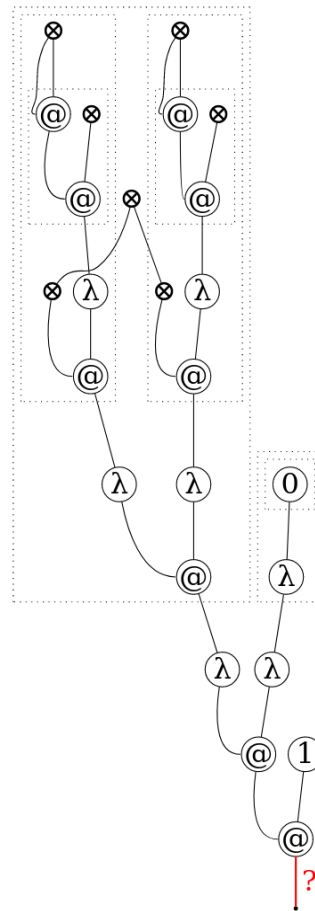
Check
type
of
"pop"

(a) The following ASG was generated using SPARTAN and the following code:

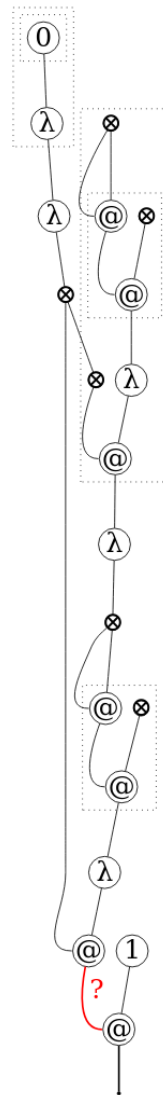
))

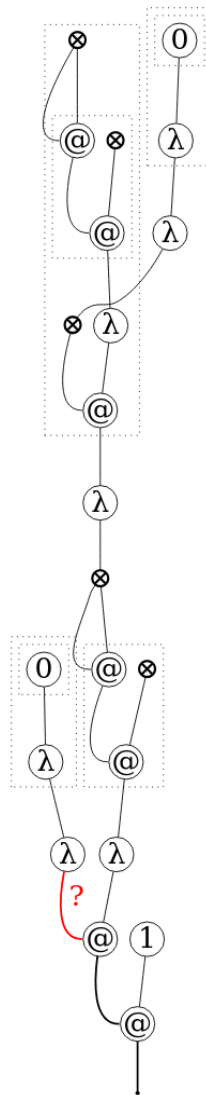


(b) (i)

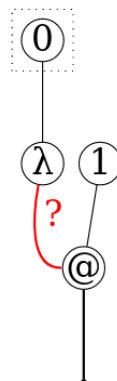
 $Y(\lambda f.\lambda x.0)1$ 

Firstly, the ASG evaluate the LHS, finding values on either side of the first application, it performs a reduction, replacing $\lambda t.$ with $\lambda f.\lambda x.0$:





Again finding two values on the LHS application, a reduction is performed, stripping the outer λ from $\lambda f.\lambda x.0$, discarding the Y combinator:



Finally, finding 2 values either side of our application the system removes the next λ , leaving just 0, our final result from this computation:

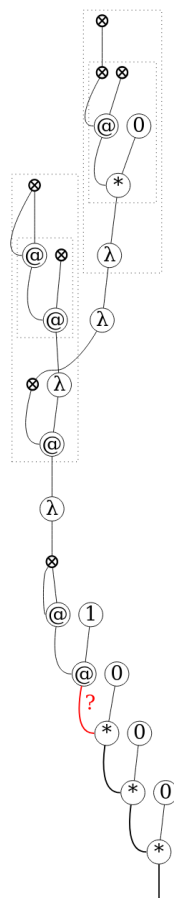


(ii)

$$Y(\lambda f. \lambda x. f(x) * 0)1$$

This expression will diverge. This occurs as the second operand of the $*$ operator (1) is never evaluated as our Y combinator will infinitely expand over the first function argument $\lambda f. \lambda x. f(x) * 0$

After 91 steps of execution the ASG abstract machine will be in the following state:



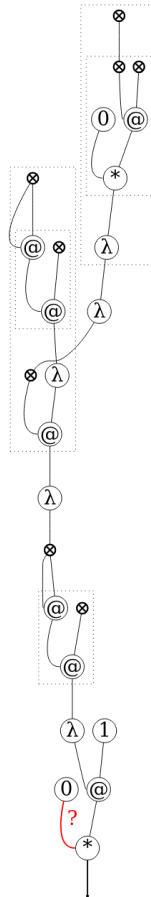
Intuitively, you can see that the second operand (1) is being pushed up as the Y combinator duplicates the operation and first operand (0) infinitely.

(iii) If we were to swap our operands to form an expression:

$$Y(\lambda f. \lambda x. 0 * f(x))1$$

This would terminate, due to the nature of our $*$ *shortcut* operator which does not need to evaluate a second argument in the case where the first operand is 0.

After 26 steps of execution we reach the following state:



Here you can see that we have a state where we are attempting to evaluate $0 * @$, with a traditional (eager) multiplication operator, we would find a value on the left and an expression on the right, forcing us to evaluate the RHS until we reach a value. However, our *shortcut* operator allows us to instead skip this evaluate and return 0, eliminating the hang the Y combinator would otherwise cause. Our final result would be:



- (c) (i) My answer to b)i) would not be affected. Nor would my answer to b)ii) as the cause of the infinite expansion was the first argument, not the second.

For b)iii) however, our answer would change. The program would no longer terminate as when it reached the state shown above, where $*$ is applied to a value (0) and an expression ($@$), it would be forced to expand the RHS application and create a similar infinite expansion as in b)ii) due to the use of the Y combinator without a terminating condition.

- (ii) Because of the fact $\forall x, x * 0 = 0 * x = 0$, both $Y(\lambda f. \lambda x. f(x) * 0)1$ and $Y(\lambda f. \lambda x. 0 * f(x))1$ can be optimised to 0. In the first case, $f(x) * 0$ can be replaced with 0 where we let $f(x) = x$ in the above fact. In the second case, $0 * f(x)$ can be replaced with 0 where we let $f(x) = x$ in the above fact.

is this a trick question, check notes on code optimisation

- (iii) Yes $Y(\lambda f. \lambda x. 0)1$ can be optimised to 0. This is the case as the result of evaluating the expression is not dependant on either of the parameters f or x .

Question 3

- (a) (i) a is true whenever b is true. This is an example of an invariant, (a subset of *safety properties*) as it can only be verified by checking $b \rightarrow a$ in each state individually. It can be represented in LTL as:

$$\Box(b \rightarrow a)$$

- (ii) a and b are simultaneously true only a finite number of times.

This is a *liveness property* as only an infinite path could satisfy $\Box(a \wedge b)$ and given such a path, we could easily extend it with a single state which satisfied $\neg(a \wedge b)$ to satisfy this property. In LTL it can be represented as:

Check

$$\neg\Box(a \wedge b)$$

- (iii) every b is immediately followed by an a

This is an example of a *safety property* as any violation of this property would have a state satisfying b (immediately) followed by a state satisfying $\neg a$. Such a trace cannot be extended to satisfy the property. It can be written in LTL as:

check wording

$$\Box(b \rightarrow \bigcirc a)$$

- (iv) exactly one of a or b (but not both) is eventually true. This is a *liveness property* as given a trace in which both a and b appear true at least once, no suffix can be added to extend the trace s.t. it satisfies this property.

check

$$(\Diamond a \vee \Diamond b) \wedge (a \rightarrow \Box \neg b) \wedge (b \rightarrow \Box \neg a)$$

- (v) if a ever becomes true, then it remains true forever, and this is immediately preceded by a state where b was true.

This is a *liveness property* as given a trace that violates this property such as a, b , the trace cannot be extended with any suffix s.t. the property now holds.

$$((b \wedge \bigcirc a) \rightarrow \bigcirc \bigcirc \Box a) \vee \neg(\neg b \wedge \bigcirc a)$$

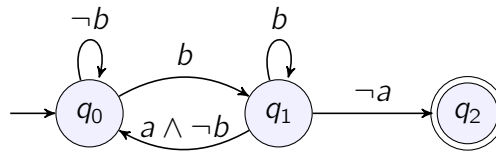
$$((b \wedge \bigcirc a) \rightarrow \bigcirc \bigcirc \Box a) \vee (b \vee \bigcirc \neg a)$$

- (b) The first stage of LTL model checking is to negate the LTL formula $\Psi = \Box(b \rightarrow \bigcirc a)$:

$$\begin{aligned} \neg\Psi &= \neg\Box(b \rightarrow \bigcirc a) \\ &= \Diamond\neg(b \rightarrow \bigcirc a) \\ &= \Diamond\neg(\neg b \vee \bigcirc a) \\ &= \Diamond(b \wedge \bigcirc\neg a) \end{aligned}$$

Our negation makes syntactic sense in this case, to prove that “every b is immediately followed by an a ” does not hold, one must find a trace whereby a b is immediately followed by $\neg a$.

We then construct an NFA, \mathcal{A}_Ψ of this formula:



We then take the product of the LTS given in the question and the NFA \mathcal{A}_Ψ , defined above, to produce:

$\mathcal{A}_\Psi \otimes M$, where M is the LTS given in the question:



check NFA prod

Above you can see that the product of the NFA and LTS has an *accepting* state, meaning there exists a trace that satisfies the negation of the property, i.e. the property does not hold for all possible traces.

Statement of good academic conduct

By submitting this assignment, I understand that I am agreeing to the following statement of good academic conduct.

- I confirm that this assignment is my own work and I have not worked with others in preparing this assignment.
- I confirm this assignment was written by me and is in my own words, except for any materials from published or other sources which are clearly indicated and acknowledged as such by appropriate referencing.
- I confirm that this work is not copied from any other person's work (published or unpublished), web site, book or other source, and has not previously been submitted for assessment either at the University of Birmingham or elsewhere.
- I confirm that I have not asked, or paid, others to prepare any part of this work for me.
- I confirm that I have read and understood the University regulations on plagiarism.