P;1ai. <(x := 1), s> -> <skip, s[x -> 1]>

ii. <while true do skip, s> -> <if true then (skip; while true do skip) else skip, s>
-> <skip; while true do skip, s> -> <while true do skip, s>
Loops this forever

iii. <(x := 1) or (while true do skip), s>
Case 1: LHS chosen
<(x := 1) or (while true do skip), s> -> <x := 1, s> -> <skip, s[x -> 1]>

Case 2: RHS chosen
<(x := 1) or (while true do skip), s> -> <while true do skip, s> -> …
Infinite loop as in part 1aii

b. To diverge, must be infinite loop

As shown in part ii, we get back to another configuration with <while true do skip, s’> for some state s’. However, the command skip does not change the original state s and, by the semantics of While-Or, no other command is introduced to the loop that will change the state.

Therefore s’ = s and <while true do skip, s> ->* <while true do skip, s>

Alternative approach:
For <while true do skip, s> to converge, there exists some k in the positive natural numbers (clearly zero not applicable) such that <(while true do skip)> ->k <skip, s’>.

Assume <while true do skip, s> converges.

Repeating steps in (1aii), it then means that:
<(while true do skip), s> ->k <skip, s’>
<if true then (skip; while true do skip) else skip, s> ->(k-1) <skip, s’>.
<skip; while true do skip, s> ->(k-2) <skip, s’>
<while true do skip, s> ->(k-3) <skip, s’>

<(while true do skip), s> therefore only converges if k = k - 3, which is never true. Hence must diverge.

c. <(x := 1), s> and <(x := 1) or (while true do skip), s> can be

As shown in part 1ai, <(x := 1), s> ->* <skip, s[x -> 1]>
As shown in part 1aiii, on one path, <(x := 1) or (while true do skip)> ->* <skip, s[x -> 1]>

Therefore, <(x := 1), s> ~ <(x := 1) or (while true do skip), s>

No others are as <while true do skip, s> ↛* <skip, s’> as shown in part 1aii.

Nd

1di. **NOTE**: Mistake in the exam question, should be ‘C’ in the place of “skip” (see end of this document)

Proof 1: <C, s[x -> 0, z -> 0]> ->* <C, s[x -> n, z -> 0]> ∀ n $\in N^+$

Base Case: n = 1

<C, s[x -> 0, z -> 0]> -> <if (z = 0) then (x := x + 1; (z := 0) or (z := 1); C) else skip, s>
-> <x := x + 1; (z := 0) or (z := 1); C, s>
-><skip; (z := 0) or (z := 1); C, s[x -> 1]>
-><(z := 0) or (z := 1); C, s[x -> 1]>
-> <(z := 0; C, s[x -> 1]>
-> <skip; C, s[x -> 1, z -> 0]>
-> <C, s[x -> 1, z -> 0]>

Therefore, holds for base case

Assume true for n = k, k $\in N^+$

For n = k + 1:

<C, s[x -> 0, z -> 0]> ->* <C, s[x -> k, z -> 0]>, by IH
By the semantics of While-Or, we must therefore have that:

<C, s[x -> 0, z -> 0]> ->* <(z := 0) or (z := 1); C, s[x -> k, z -> 0]>

<(z := 0) or (z := 1); C, s[x -> k, z -> 0]>
-> <z := 0; C, s[x -> k, z -> 0]>
-> <skip; C, s[x -> k, z -> 0]>
-> <C, s[x -> k, z -> 0]>
-> <if (z=0) then (x := x + 1; (z := 0) or (z : = 1); C) else skip, s[x -> k, z -> 0]>
-> <x := x + 1; (z := 0) or (z : = 1); C, s[x -> k, z -> 0]>
-> <skip; (z := 0) or (z := 1); C, s[x -> k + 1, z -> 0]>
-> <(z := 0) or (z := 1); C, s[x -> k + 1, z -> 0]>
-> <z := 0; C, s[x -> k + 1, z -> 0]>
-> <skip; C, s[x -> k + 1, z -> 0]>
-> <C, s[x -> k + 1, z -> 0]>

Therefore, holds for inductive case.

Therefore, holds for all n $\in N^+$

Proof 2: <C, s[x -> 0, z -> 0]> ->* <skip, s[x -> n, z -> 1]> $\forall$ n $\in N^+$

Base Case: n = 1
<C, s[x -> 0, z -> 0]> -> <if (z = 0) then (x := x + 1; (z := 0) or (z := 1); C) else skip, s>
-> <x := x + 1; (z := 0) or (z := 1); C, s>
-><skip; (z := 0) or (z := 1); C, s[x -> 1]>
-><(z := 0) or (z := 1); C, s[x -> 1]>
-> <(z := 1; C, s[x -> 1]>
-> <skip; C, s[x -> 1, z -> 1]>
-> <C, s[x -> 1, z -> 1]>
-> <if (z = 0) then (x := x + 1; (z := 0) or (z := 1); C) else skip, s[x -> 1, z -> 1]>
-> <skip, s[x -> 1, z -> 1]>

Therefore holds for base case

Assume true for n = k, k $\in N^+$

For n = k + 1:

<C, s[x -> 0, z -> 0]> ->* <skip, s[x -> k, z -> 1]>, by IH
By the semantics of While-Or, we must therefore have that:

<C, s[x -> 0, z -> 0]> ->* <(z := 0) or (z := 1); C, s[x -> k, z -> 0]>

<(z := 0) or (z := 1); C, s[x -> k, z -> 0]>
-> <z := 0; C, s[x -> k, z -> 0]>
-> <skip; C, s[x -> k, z -> 0]>
-> <C, s[x -> k, z -> 0]>
-> <if (z=0) then (x := x + 1; (z := 0) or (z : = 1); C) else skip, s[x -> k, z -> 0]>
-> <x := x + 1; (z := 0) or (z : = 1); C, s[x -> k, z -> 0]>
-> <skip; (z := 0) or (z := 1); C, s[x -> k + 1, z -> 0]>
-> <(z := 0) or (z := 1); C, s[x -> k + 1, z -> 0]>
-> <z := 1; C, s[x -> k + 1, z -> 0]>
-> <skip; C, s[x -> k + 1, z -> 1]>
-> <C, s[x -> k + 1, z -> 1]>
-> <if (z = 0) then (x := x + 1; (z := 0) or (z := 1); C) else skip, s[x -> k + 1, z -> 1]>
-> <skip, s[x -> k + 1, z -> 1]>

Therefore holds for inductive case.

Therefore holds for all $n \in N+$

1dii. <x := somenumber; z := 1, s> -> <skip; z := 1, s[x -> n]>
-> <z := 1, s[x -> n]> -> <skip, s[x -> n, z -> 1]>

Therefore, <x := somenumber; z := 1, s> ->* <skip, s[x -> n, z -> 1]

<x := 0; z := 0; C, s> -> <skip; z := 0; C, s[x -> 0]>
-> <z := 0; C, s[x -> 0]>
-> <skip; C, s[x -> 0, z -> 0]>
-> <C, s[x -> 0, z -> 0]>

Therefore, <x := 0; z := 0; C, s> ->* <C, s[x -> 0, z -> 0]>

As <x := 0; z := 0; C, s> ->* <C, s[x -> 0, z -> 0]> and, by part 1di,
<C, s[x -> 0, z -> 0]> ->* <skip, s[x -> n, z -> 1]>:

<x := 0; z := 0; C> ->* <skip, s[x -> n, z -> 1]>

By given *definition*:

The command (x := somenumber; z := 1) is equivalent to the command (x := 0; z := 0; C) as:

<x := somenumber; z := 1, s> ->* <skip, s[x -> n, z -> 1]
and
<x := 0; z := 0; C> ->* <skip, s[x -> n, z -> 1]>

ii) $C_1 ::= x := \text{somenumber} ; z := 1$ $\quad C_2 ::= x := 0 ; z := 0 ; C$

$C_1 \sim C_2$.

"⇒" take some arbitrary state s and assume $\langle C_1, s\rangle \to^* \langle skip, s'\rangle$

$\langle C_1, s\rangle \to_c \langle x := n; z := 1, s[x \mapsto n]\rangle$ for some $n \in \mathbb{N}^+$

$\to_c \langle skip; z := 1, s[x \mapsto n]\rangle$

$\to_c \langle z := 1, s[x \mapsto n]\rangle$

$\to_c \langle skip, s[x \mapsto n, z \mapsto 1]\rangle$

$\Rightarrow s' = s[x \mapsto n, z \mapsto 1]$ by determinism of While (no Or involved)

$\langle C_2, s\rangle \to_c \langle skip; z := 0; C, s[x \mapsto 0]\rangle$

$\to_c \langle z := 0; C, s[x \mapsto 0]\rangle$

$\to_c \langle skip; C, s[x \mapsto 0, z \mapsto 0]\rangle$

$\to_c \langle C, s[x \mapsto 0, z \mapsto 0]\rangle$

$\to_c^* \langle skip, s[x \mapsto n, z \mapsto 1]\rangle$ by part (i) for generated n.

$\Rightarrow [\forall s, s' \; \langle C_1, s\rangle \to^* \langle skip, s'\rangle \Rightarrow \langle C_2, s\rangle \to_c^* \langle skip, s'\rangle]$

where we note the choice of s restricts s'.

"⇐" take some arbitrary state s and assume $\langle C_2, s\rangle \to^* \langle skip, s'\rangle$

"$\Leftarrow$" take some arbitrary state s and assume $\langle C_2, s\rangle \xrightarrow{*} \langle skip, s'\rangle$

$$\begin{aligned}\langle C_2, s\rangle &\rightarrow_c \langle skip; z:=0; C, s[x\mapsto 0]\rangle \\ &\rightarrow_c \langle z:=0; C, s[x\mapsto 0]\rangle \\ &\rightarrow_c \langle skip; C, s[x\mapsto 0, z\mapsto 0]\rangle \\ &\rightarrow_c \langle C, s[x\mapsto 0, z\mapsto 0]\rangle \\ &\rightarrow_c^* \langle skip, s[x\mapsto n, z\mapsto 1]\rangle \quad n\in\mathbb{N}^+ \quad \text{from (i)}\end{aligned}$$

$\Rightarrow \quad s' = s[x\mapsto n, z\mapsto 1]$ can't use determinism here but can show in inductive proof that all other derivations of C diverge.

$\langle C_1, s\rangle \rightarrow_c^* \langle skip, s[x\mapsto n, z\mapsto 1]\rangle$ from above, where somenumber generates matching n

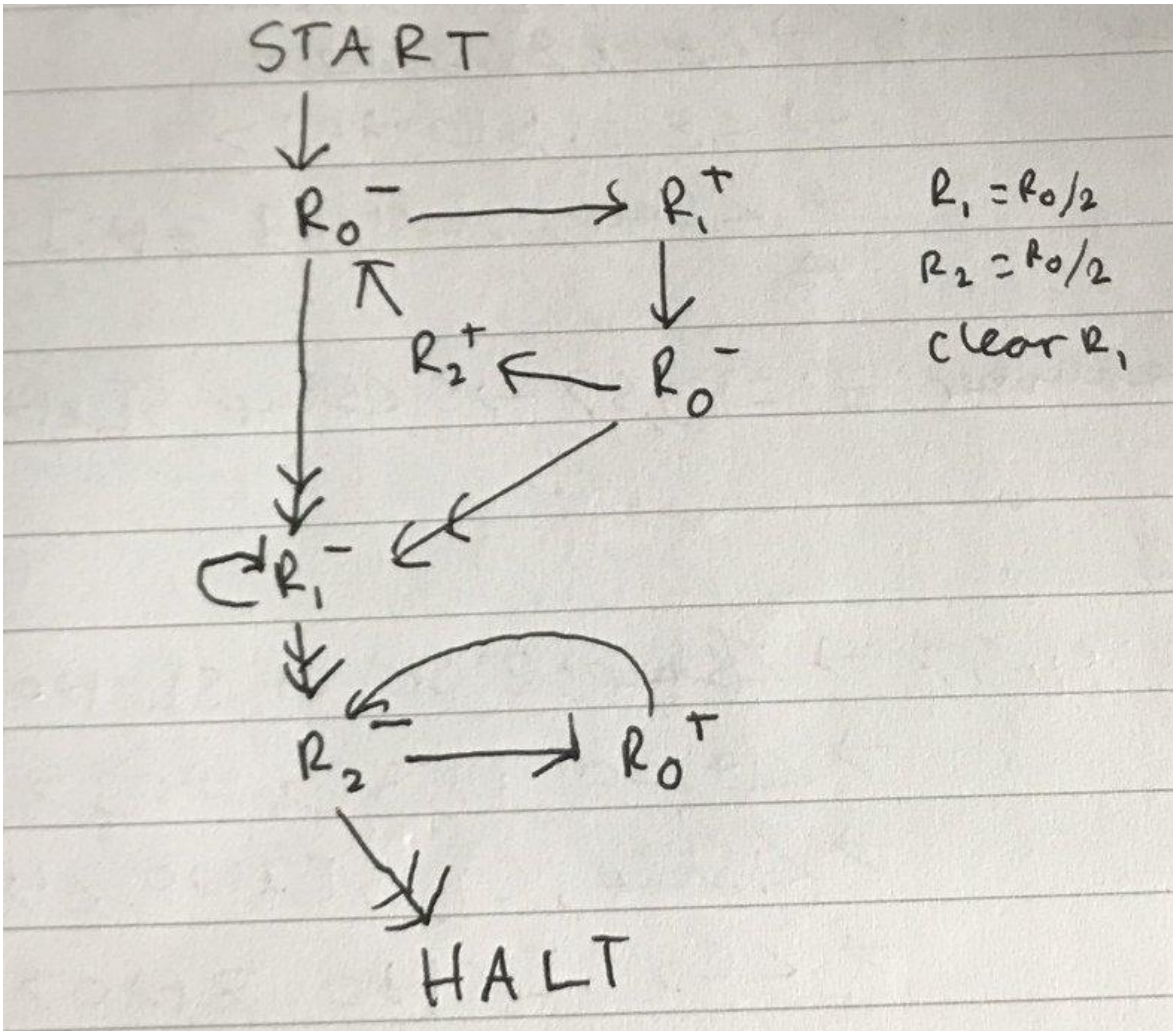
$\Rightarrow [\forall s, s' \; \langle C_2, s\rangle \xrightarrow{*} \langle skip, s'\rangle \Rightarrow \langle C_1, s\rangle \rightarrow_c^* \langle skip, s'\rangle]$

where we note again the choice of s restricts s'.

$\Rightarrow \quad \forall s, s' \; \langle C_1, s\rangle \xrightarrow{*} \langle skip, s'\rangle \Leftrightarrow \langle C_2, s\rangle \rightarrow_c^* \langle skip, s'\rangle$

$\Rightarrow \quad C_1 \sim C_2$

2ai



Performs integer division by 2 on m (DIV 2)

Explanation: With the initial state of R0 being m with R1 = R2 = 0, we repeat the process of taking 1 away from R0 and putting it into R1 or R2 and alternating, until R0 is empty. As R1 is first, it gets the bigger half (if odd). R2 will get the smaller half (m DIV 2). We then empty R1 and move (m DIV 2) into R0.

ii. $\ulcorner R_0^-$ - -> $L_1, L_4 \urcorner$ = <<(2 x 0) + 1, <1, 4>>> = <<1, <1, 4>>> = <<1, 17>> = 2(34 + 1) = 70

$\ulcorner R_1^+$ -> $L_2 \urcorner$ = <<(2 x 1), 2>> = <<2, 2>> = 4(4 + 1) = 20

$\ulcorner R_0^-$ -> $L_3, L_4 \urcorner$ = <<(2 x 0) + 1, <3, 4>>> = <<1, <3, 4>>> = <<1, 71>> = 2(142 + 1) = 286

iii. ΓI_1 ⌝ = 1144 = 2^x (2y + 1) = 2^3 (2(71) + 1) = <<3, 71>> = <<(2 x 1) + 1, 71>>
= <<(2 x 1 + 1, <3, 4>>>

So I_1: R_1^- -> L_3, L_4

ΓI_2 ⌝ = 448 = 2^x (2y + 1) = 2^6 (2(3) + 1) = <<6, 3>> = <<(2 x 3), 3>>

So I_1: R_3^+ -> L_3

b. Was not taught for 2021

b) i) $s\ 0\ (b\ (l\ 2)\ 1\ (l\ 3))$

single $\underline{0}$ (branch (leaf 1) 1 (leaf 3))

~~$\lambda bsl.\ s\ \underline{0}\ (b\ (l\ 1)\ 1\ (l\ 3))$~~

$\underline{T_l} = \lambda bsl.\ s\ \underline{0}\ (b\ (l\ \underline{2})\ \underline{1}\ (l\ \underline{3}))$

$\underline{T_r} = \lambda bsl.\ b\ (l\ \underline{1})\ \underline{0}\ (s\ \underline{3}\ (l\ \underline{3}))$

ii) $B \triangleq \lambda t_1 x t_2 bsl.\ b\ (t_1\ b s l)\ x\ (t_2\ b s l)$

$B\ \underline{(l\ 1)}\ \underline{0}\ \underline{(s\ 2\ (l\ 3))}$

$=_B \lambda t_1 x t_2 bsl.\ b\ (t_1\ b s l)\ x\ (t_2\ b s l)\ \underline{(l_1)}\ \underline{0}\ \underline{(s\ 2\ (l\ 3))}$

$\to^3_\beta \lambda bsl.\ b\ ((\underline{l 1})\ b s l)\ \underline{0}\ ((\underline{s\ 2\ (l\ 3)})\ b s l)$

$=_B \lambda bsl.\ b\ ((\lambda bsl.\ l\ \underline{1})\ b s l)\ \underline{0}\ ((\underline{s\ 2\ (l\ 3)})\ b s l)$

$\to^3_\beta \lambda bsl.\ b\ (l\ \underline{1})\ \underline{0}\ ((s\ 2\ (l\ 3))\ b s l)$

$=_B \lambda bsl.\ b\ (l\ \underline{1})\ \underline{0}\ ((\lambda bsl.\ s\ \underline{2}\ (l\ \underline{3}))\ b s l)$

$\to^3_\beta \lambda bsl.\ b\ (l\ \underline{1})\ \underline{0}\ (s\ \underline{2}\ (l\ \underline{3}))$

$=_B \underline{T_r}$

iii) $C \triangleq \lambda t. t\ (\lambda xyz. \underline{2})\ (\lambda xy. \underline{1})\ (\lambda x. \underline{0})$

$C\ \underline{T_l}$

$=_B \lambda t. t\ (\lambda xyz. \underline{2})\ (\lambda xy. \underline{1})\ (\lambda x. \underline{0})\ \underline{T_l}$

$\to_B \underline{T_l}\ (\lambda xyz. \underline{2})\ (\lambda xy. \underline{1})\ (\lambda x. \underline{0})$

$=_B (\lambda bsl. s\ \underline{0}\ (b\ (l\ \underline{2})\ \underline{1}\ (l\ \underline{3})))\ (\lambda xyz. \underline{2})\ (\lambda xy. \underline{1})\ (\lambda x. \underline{0})$

$\to_B (\lambda sl. s\ \underline{0}\ ((\lambda xyz. \underline{2})\ (l\ \underline{2})\ \underline{1}\ (l\ \underline{3})))\ (\lambda xy. \underline{1})\ (\lambda x. \underline{0})$

$\to_B (\lambda l. (\lambda xy. \underline{1})\ \underline{0}\ ((\lambda xyz. \underline{2})\ (l\ \underline{2})\ \underline{1}\ (l\ \underline{3})))(\lambda x. \underline{0})$

$\to_B (\lambda xy. \underline{1})\ \underline{0}\ ((\lambda xyz. \underline{2})((\lambda x. \underline{0})\ \underline{2})\ \underline{1}\ ((\lambda x. \underline{0})\ \underline{3}))$

$\to_B (\lambda y. \underline{1})\ ((\lambda xyz. \underline{3})\ ((\lambda x. \underline{0})\ \underline{2})\ \underline{1}\ ((\lambda x. \underline{0})\ \underline{3}))$

$\to_B \underline{1}$ //

$C\ \underline{T_r}$

$=_B \lambda t. t\ (\lambda xyz. \underline{2})\ (\lambda xy. \underline{1})\ (\lambda x. \underline{0})\ \underline{T_r}$

$\to_B \underline{T_r}\ (\lambda xyz. \underline{2})\ (\lambda xy. \underline{1})(\lambda x. \underline{0})$

$=_B [\lambda bsl. b\ (l\ \underline{1})\ \underline{0}\ (s\ \underline{3}\ (l\ \underline{3}))](\lambda xyz. \underline{2})(\lambda xy. \underline{1})(\lambda x. \underline{0})$

$\to_B [\lambda sl. (\lambda xyz. \underline{2})\ (l\ \underline{1})\ \underline{0}\ (s\ \underline{3}\ (l\ \underline{3}))]\ (\lambda xy. \underline{1})\ (\lambda x. \underline{0})$

$\to_B [\lambda l. (\lambda xyz. \underline{2})\ (l\ \underline{1})\ \underline{0}\ ((\lambda xy. \underline{0})\ \underline{3}\ (l\ \underline{3}))]\ (\lambda x. \underline{0})$

$\to_B [(\lambda xyz. \underline{2})\ ((\lambda x. \underline{0})\underline{1})\ \underline{0}\ [(\lambda xy. \underline{0})\ \underline{3}\ ((\lambda x. \underline{0})\ \underline{3})]]$

$\to_B [\lambda yz. \underline{2}\ \ \underline{0}\ [(\lambda xy. \underline{0})\ \underline{3}\ ((\lambda x. \underline{0})\ \underline{3})]]$

$\to_B [\lambda z. \underline{2}\ \ [(\lambda xy. \underline{0})\ \underline{3}\ ((\lambda x. \underline{0})\ \underline{3})]]$

$\to_B \underline{2}$ //

1.hi

1. d) yes

F

1 d) <C, s[x -> 0, z -> 0]> ->* <C, s[x -> n, z -> 0]> was the intended meaning

0000000000000000000

question @124

Exam mistake

I didn't realise the exam mistake until later and this had a collateral impact on the rest of my paper due to time wasted on the question. I wasted even more time on this because i knew it was worth 40% of the question and 20% of t

What kind of mitigation is there for this? I appreciate there is going to be some lenient marking for this question but I don't think this will account for the rest of the exam which some of us were unable to complete due to timing.

exam

edit · good question | 14

the students' answer, *where students collectively construct a single answer*

Click to start off the wiki answer

the instructors' answer, *where instructors collectively construct a single answer*

We will understand how to mark appropriately.

thanks! | 0