Name: Flori Kusari Problem Sheet # 9

Problem 9.1: triangle display

a

X_2	X_1	X_0	a	b	c	d	e	f
0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0
0	1	0	0	1	1	0	0	0
0	1	1	1	1	1	0	0	0
1	0	0	1	1	1	0	1	0
1	0	1	1	1	1	1	0	1
1	1	0	1	1	1	1	1	1

b

$a = X_2 \vee X_0$

$b = X_2 \vee X_1$

$c = X_2 \vee X_1$

$d = x_2 \wedge (x_1 \vee x_0)$

$e = x_2 \wedge \neg x_0$

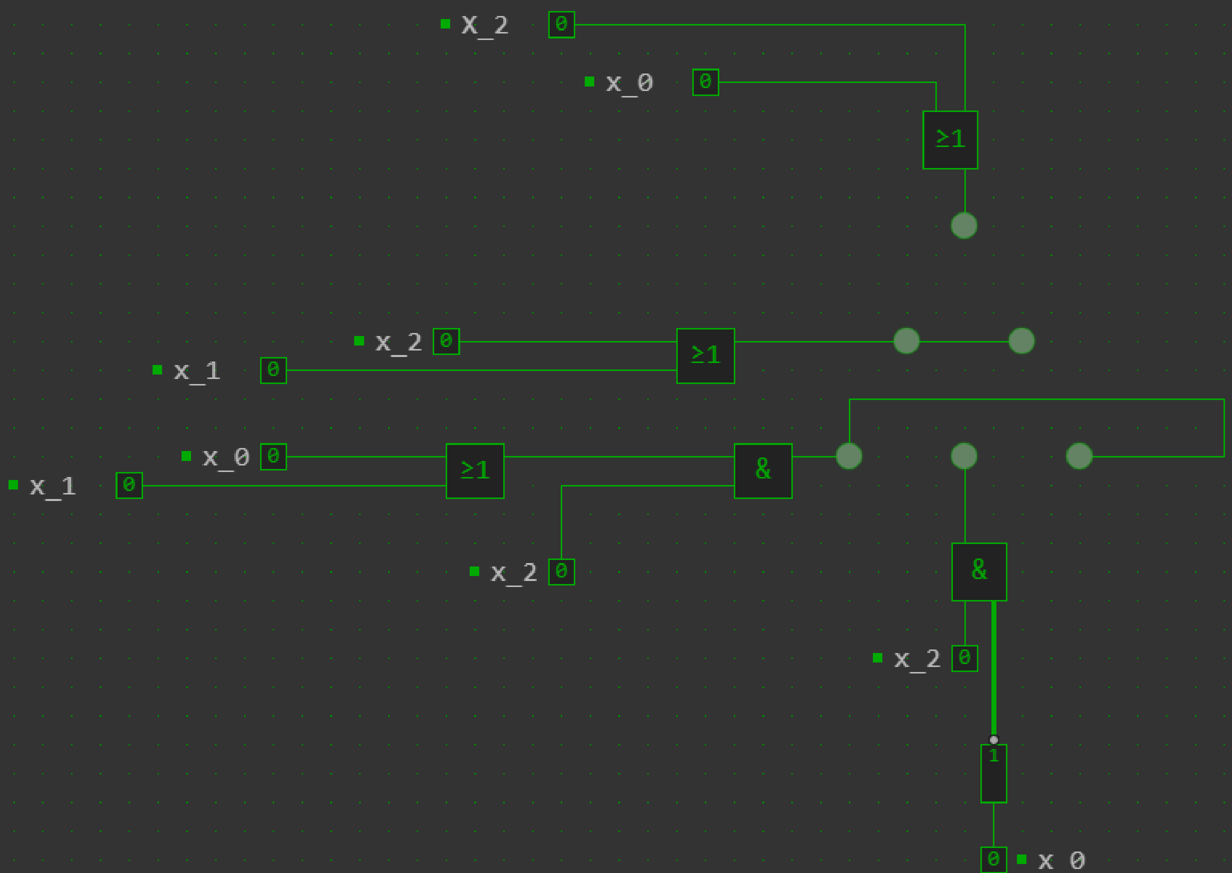
$f = x_2 \wedge (x_1 \vee x_0)$

c Turned in as a link

Problem 9.2: map function equivalence proof in haskell

Base Case: Since "map (f . g) []" is empty it is logically inferred that it is equivalent to "map f . map g []" since it is also empty. So thus the base case is correct by default.

Inductive Step: Assume "map (f.g) []" equivalent to "map f . map g []" for some list "xs". Now to prove we need to prove that the equivalence holds for x: xs for an element x.



x_2
0
x_0
0
≥1
x_2
0
x_1
0
≥1
x_0
0
x_1
0
≥1
&
x_2
0
&
x_2
0
1
0
x_0

So: $map\,(f.g)(x:xs) = (f.g)\,x : map\,(f.g)\,xs$

Now since the expressions are assumed to be equivalent:

$map\,(f.g)\,xs \equiv map\,f\,.\,map\,g\ \$\ xs$

when simplified it can be written as:

$(f.g)\,x : map\,f\,(map\,g\,xs)$

using the general rule of how functions of this type work.

Using the def. of composition "$(f.g)\,x$":

$f(g\,x) : map\,f\,(map\,g\,xs)$

When rewritten:

$f(g\,x) : map\,f\,.\,map\,g\ \$\ x:xs.$

Therefore we have proven that "$map(f.g)xs$" is equivalent to "$map\,f.\,map\,g\ \$\ xs$" we can conclude that it must also hold for the arbitrary element aswell "$x:xs$".

Since we have proved the Base Case and the Inductive Step we have completed the proof. □

Problem 9.3: left and right folds in haskell

a) Step 0: "f" is binary function "/"
"e" is the accumulator value "50"
"[4,2,5]" is the list given.

Step 1. Apply the foldl. Part 1

foldl (/) (50/4) [2,4] →

Step 2. Apply the foldl. Part 2.

foldl (12.5 / 2) [5]

Step 3: Apply the foldl. Part 3

foldl (6.25 / 5) [ ]

Step 4: Calculate

1.25

(b) Step 0: "f" is the binary function "/"

"e" is the accumulator value "50"

"[4, 2, 5]" is the list given.

Step 1: Apply the foldr. Part 1

4 / (foldr (/) 50) [2, 5]

Step 2: Apply the foldr. Part 2

4 / (2 / foldr (/) 50) [5]

Step 3: 4 / (2 / (5 / foldr (/) 50)

Step 4: Calculate

4 / (2 / (5 / 50))

4 / (2 / 0.1)

4 / 20

0.2

THE END