Solutions to **CSC240 Winter 2022 Homework Assignment 2**

My name and student number:
Sebastien Psarianos 1008596119
The list of people with whom I discussed this homework assignment:

NO OUTSIDE DISCUSSION

1. Original Statement:

$$(\forall i \in \mathbb{Z}.g(i,n)) \text{ IMPLIES } [(\exists x \in \mathbb{Z}.e(i,x)) \text{ AND } \forall j \in \mathbb{Z}.(g(j,i) \text{ IMPLIES } \forall y \in \mathbb{Z}.e(j,y))].$$

Since y does not occur in g(j,i)
E IMPLIES $\forall x \in D.q(x)$ can be transformed to $\forall x \in D.(E \text{ IMPLIES } q(x))$

$$(\forall i \in \mathbb{Z}.g(i,n)) \text{ IMPLIES } [(\exists x \in \mathbb{Z}.e(i,x)) \text{ AND } \forall j \in \mathbb{Z}.\forall y \in \mathbb{Z}.(g(j,i) \text{ IMPLIES } e(j,y))].$$

Since x does not occur in $\forall j \in \mathbb{Z}.\forall y \in \mathbb{Z}.(g(j,i) \text{ IMPLIES } e(j,y))$:
$(\exists x \in D.p(x))$ AND E can be transformed to $\exists x \in D.(p(x) \text{ AND } E)$

$$(\forall i \in \mathbb{Z}.g(i,n)) \text{ IMPLIES } \exists x \in \mathbb{Z}.[e(i,x) \text{ AND } \forall j \in \mathbb{Z}.\forall y \in \mathbb{Z}.(g(j,i) \text{ IMPLIES } e(j,y))].$$

Since j and y don't occur in $e(i,x)$):
E AND $\forall x \in D.q(x)$ can be transformed to $\forall x \in D(E \text{ AND } q(x))$ twice:

$$(\forall i \in \mathbb{Z}.g(i,n)) \text{ IMPLIES } \exists x \in \mathbb{Z}.\forall j \in \mathbb{Z}.\forall y \in \mathbb{Z}.[e(i,x) \text{ AND } (g(j,i) \text{ IMPLIES } e(j,y))].$$

Since i exists in three places with two of them outside of the scope of the $\forall i \in \mathbb{Z}$ quantification, there are two not-quantified instances of i. Therefore let the quantified $i = i_1$ and the free variable $i = i_2$, giving:

$$(\forall i_1 \in \mathbb{Z}.g(i_1,n)) \text{ IMPLIES } \exists x \in \mathbb{Z}.\forall j \in \mathbb{Z}.\forall y \in \mathbb{Z}.[e(i_2,x) \text{ AND } (g(j,i_2) \text{ IMPLIES } e(j,y))].$$

Since i_1 does not occur in $\exists x \in \mathbb{Z}.\forall j \in \mathbb{Z}.\forall y \in \mathbb{Z}[e(i_2,x) \text{ AND } (g(j,i_2) \text{ IMPLIES } e(j,y))]$ anymore:
$(\forall x \in D.p(x))$ IMPLIES E can be transformed to $\exists x \in D.(p(x) \text{ IMPLIES } E)$

$$\exists i_1 \in \mathbb{Z}.(g(i_1,n) \text{ IMPLIES } \exists x \in \mathbb{Z}.\forall j \in \mathbb{Z}.\forall y \in \mathbb{Z}.[e(i_2,x) \text{ AND } (g(j,i_2) \text{ IMPLIES } e(j,y))]).$$

Since x does not occur in $g(i_1,n)$:
E IMPLIES $(\exists x \in D.q(x))$ can be transformed to $\exists x \in D.(E \text{ IMPLIES } q(x))$

$$\exists i_1 \in \mathbb{Z}.\exists x \in \mathbb{Z}.(g(i_1,n) \text{ IMPLIES } \forall j \in \mathbb{Z}.\forall y \in \mathbb{Z}.[e(i_2,x) \text{ AND } (g(j,i_2) \text{ IMPLIES } e(j,y))]).$$

Since j and y do not occur in $g(i_1,n)$:
E IMPLIES $(\forall x \in D.q(x))$ can be transformed to $\forall x \in D.(E \text{ IMPLIES } q(x))$

$$\exists i_1 \in \mathbb{Z}.\exists x \in \mathbb{Z}.\forall j \in \mathbb{Z}.\forall y \in \mathbb{Z}.(g(i_1,n) \text{ IMPLIES } [e(i_2,x) \text{ AND } (g(j,i_2) \text{ IMPLIES } e(j,y))]).$$

2. (a) $\forall i \in \{1,2,3,4,5,6\}.(f_i : \{\mathrm{T},\mathrm{F},\mathrm{N}\}, \rightarrow \{\mathrm{T},\mathrm{F},\mathrm{N}\})$

$f_1(x) = x$
$f_1(\mathrm{T}) = \mathrm{T},\ f_1(\mathrm{N}) = \mathrm{N},\ f_1(\mathrm{F}) = \mathrm{F}$

$f_2(x) = \mathrm{GNOT}\ x$
$f_2(\mathrm{T}) = \mathrm{F},\ f_2(\mathrm{N}) = \mathrm{N},\ f_2(\mathrm{F}) = \mathrm{T}$

$f_3(x) = \mathrm{GROT}\ x$
$f_3(\mathrm{T}) = \mathrm{N},\ f_3(\mathrm{N}) = \mathrm{F},\ f_3(\mathrm{F}) = \mathrm{T}$

$f_4(x) = \mathrm{GROT}\ (\mathrm{GNOT}\ x)$
$f_4(\mathrm{T}) = \mathrm{T},\ f_4(\mathrm{N}) = \mathrm{F},\ f_4(\mathrm{F}) = \mathrm{N}$

$f_5(x) = \mathrm{GNOT}\ (\mathrm{GROT}\ x)$
$f_5(\mathrm{T}) = \mathrm{N},\ f_5(\mathrm{N}) = \mathrm{T},\ f_5(\mathrm{F}) = \mathrm{F}$

$f_6(x) = \mathrm{GNOT}\ (\mathrm{GROT}\ (\mathrm{GNOT}\ x))$
$f_6(\mathrm{T}) = \mathrm{F},\ f_6(\mathrm{N}) = \mathrm{T},\ f_6(\mathrm{F}) = \mathrm{N}$

There are $P(3,3) = 6$ possible bijective functions with an domain and range sets of size of 3.

f_1: Every element mapped to itself
f_2 and f_2: GNOT and GROT are bijective functions themselves of domain and range $\{\mathrm{T},\mathrm{F},\mathrm{N}\}, \rightarrow \{\mathrm{T},\mathrm{F},\mathrm{N}\}$.
f_4,f_5 and f_6: The other 3 possible bijective functions of domain and range $\{\mathrm{T},\mathrm{F},\mathrm{N}\}, \rightarrow \{\mathrm{T},\mathrm{F},\mathrm{N}\}$, they can be represented by combinations of GNOT and GROT .

GNOT provides the ability to map a T input to a F input and vice versa. GROT provides the ability to map T input to N and by extension, in conjunction with GNOT , provides the ability to map F input to N. Once the appropriate input is mapped to N, GNOT can be used if needed to reverse the order of which elements are mapped to T and F.
With these two functions any bijective function can be created for domain and range $\{\mathrm{T},\mathrm{F},\mathrm{N}\}, \rightarrow \{\mathrm{T},\mathrm{F},\mathrm{N}\}$ as shown above in. $f_{1\rightarrow 6}$

(b) $\forall i \in \{7,8,9\}.(f_i : \{\mathrm{T},\mathrm{F},\mathrm{N}\}, \rightarrow \{\mathrm{T},\mathrm{F},\mathrm{N}\})$

$f_7(x) = x\ \mathrm{GOR}\ [(\mathrm{GNOT}\ x)\ \mathrm{GOR}\ (\mathrm{GNOT}\ (\mathrm{GROT}\ x)]$
$f_7(\mathrm{T}) = \mathrm{T},\ f_7(\mathrm{F}) = \mathrm{T},\ f_7(\mathrm{N}) = \mathrm{T}$

$f_8(x) = \mathrm{GNOT}\ (x\ \mathrm{GOR}\ [(\mathrm{GNOT}\ x)\ \mathrm{GOR}\ (\mathrm{GNOT}\ (\mathrm{GROT}\ x))])$
$f_8(\mathrm{T}) = \mathrm{F},\ f_8(\mathrm{F}) = \mathrm{F},\ f_8(\mathrm{N}) = \mathrm{F}$

$f_9(x) = \text{GROT } (x \text{ GOR } [(\text{GNOT } x) \text{ GOR } (\text{GNOT } (\text{GROT } x))])$
$f_9(\text{T}) = \text{N},\ f_9(\text{F}) = \text{N},\ f_9(\text{N}) = \text{N}$

There are three possible constant functions with domain and range of $\{\text{T}, \text{F}, \text{N}\}, \rightarrow \{\text{T}, \text{F}, \text{N}\}$.

f_7: The function f_7 always outputs T, this is because the GOR function will always output T if the either of the inputs are true. If the input is T then the x part will be T. If the Input is F then the GNOT x part will be T. If the input is N, the GNOT (GROT x) part will be true (as shown in f_4)

f_8: The GNOT function is applied to the output of f_7. Since GNOT maps T to F, the always T output of f_7 is converted to F.

f_9: The GROT function is applied to the output of f_7. Since GROT maps T to N, the always T output of f_7 is converted to N.

(c) Let a mapper be a function $g_i : \{\text{T}, \text{F}, \text{N}\} \rightarrow \{\text{T}, \text{F}, \text{N}\}$ of the form:

$$f_a \text{ GAND } f_b$$

Where the functions f_a and f_b are two functions from the bijectives defined above ($f_{1\rightarrow 6}$)

A mapper is a function that outputs one specific output (**programmed-output**) if a specific input is provided (**programmed-input**), and output F for all other input values. A mapper can take any input from the domain of the function $\{\text{T}, \text{F}, \text{N}\}$ and can "convert" it to T or N.

Both functions in a mapper will be from the bijective functions $f_{1\rightarrow 6}$ defined above. The two bijective functions in the mapper must both output the **programmed-output** when the **programmed-input** is supplied. Since there are always two bijective functions that satisfy this criteria for any **programmed-input/programmed-output** combo, both of these functions will be on either side of the GAND .

If the mapper's input is not the **programmed-input**, it will always output F. This is due to the fact that both possible inputs that are not the **programmed-input** will output F in one of the two bijective functions within the mapper. If either bijective function is F, the output of the mapper will be F.

Using these mappers, you can create a generalized propositional formula of the form:

$$(f_a \text{ GAND } f_b) \text{ GOR } (f_c \text{ GAND } f_d) \text{ GOR } (f_e \text{ GAND } f_f)$$

Where the functions $f_{a\rightarrow f}$ are various functions from the bijectives defined above ($f_{1\rightarrow 6}$)

For any input to this formula, every mapper that isn't provided with its **programmed-input** will return F. Since each mapper is separated by a GOR , the value that the overall formula will output will be either the **programmed-output** of the one mapper that was not F (the one with that **programmed-input** or the entire formula will be false. Any input can be mapped to false by simply not including a mapper with that **programmed-input**.

All other functions of this domain and range can be made from 1-3 of these mappers.

(d) The function defined above, f_9 is an example of a function of domain and range $\{\mathrm{T}, \mathrm{F}, \mathrm{N}\}, \rightarrow \{\mathrm{T}, \mathrm{F}, \mathrm{N}\}$ that cannot be defined without GROT . This is because without GROT , there is no way to map T to N. GNOT only allows the swapping of T and F values. The only way that GAND and GOR output N is if they have at least one input that is N. Therefore there is no way to create a function that maps T to N without GROT .

(e) The cartesian product $\{\mathrm{T}, \mathrm{F}, \mathrm{N}\} \times \{\mathrm{T}, \mathrm{F}, \mathrm{N}\}$ results in a set with $3 \times 3 = 9$ elements. Therefore the domain set has size 9 and the range set has size 3. Since there are 3 possible outputs for each input, there are $3^9 = 19683$ possible functions.

(f) Let $g_i : \{\mathrm{T}, \mathrm{F}, \mathrm{N}\} \rightarrow \{\mathrm{T}, \mathrm{F}, \mathrm{N}\}$ represent the mapper defined in part C
Given inputs x and y all 9 individual pairs of inputs can be mapped to an output as follows:

$$g_1(x) \text{ GAND } g_2(y)$$

The pair of mappers g_1 and g_2 must have the same **programmed-output**, however they can have different **programmed-inputs**. For example, if x and y had values T and N respectively and the desired function should output N with this combination of inputs:

The mapper g_1 would have **programmed-input**:T and **programmed-output**:N
The mapper g_2 would have **programmed-input**:N and **programmed-output**:N.

As shown previously if either x or y are not each mapper's **programmed-input** ($x \neq \mathrm{T}$ or $y \neq \mathrm{N}$), then that respective mapper would be F, making the entire statement F.

Otherwise, the above combination of mappers allows any combination of inputs to be mapped to any output. This can be done by connecting each pair of mappers with a GOR statement. Only the pair of mappers that has the correct **programmed-input** for both inputs can possibly output its **programmed-output**. As previously, if the desired output for a specific x and y combination is F, the pair of mappers for that input can be omitted. eg:

$$(g_1(x) \text{ GAND } g_2(y)) \text{ GOR } (g_3(x) \text{ GAND } g_4(y)) \text{ GOR } (g_5(x) \text{ GAND } g_6(y))$$

By including 1-9 of these when required to output T or N for a specific pair of inputs, any function from $\{\mathrm{T}, \mathrm{F}, \mathrm{N}\} \times \{\mathrm{T}, \mathrm{F}, \mathrm{N}\}$ to $\{\mathrm{T}, \mathrm{F}, \mathrm{N}\}$ can be defined.