Monday January 25

Logical equivalence Two compound propositions are logically equivalent

means that they have the same truth values for all settings of truth values to their propositional variables.

Tautology A compound proposition that evaluates to true for all

settings of truth values to its propositional variables; it

is abbreviated T.

Contradiction A compound proposition that evaluates to false for all

settings of truth values to its propositional variables; it

is abbreviated F.

Contingency A compound proposition that is neither a tautology nor

a contradiction.

Can replace p and q with any compound proposition

Definition: A collection of compound propositions is called **consistent** if there is an assignment of truth values to the propositional variables that makes each of the compound propositions true.

Consistency:

Whenever the system software is being upgraded, users cannot access the file system. If users can access the file system, then they can save new files. If users cannot save new files, then the system software is not being upgraded.

1. Translate to symbolic compound propositions

2. Look for some truth assignment to the propositional variables for which all the compound propositions output T

Consider the following algorithm to introduce redundancy in a string of 0s and 1s.

Create redundancy by repeating each bit three times

```
procedure redun3(a_{k-1}\cdots a_0): a binary string)

for i:=0 to k-1

c_{3i}:=a_i

c_{3i+1}:=a_i

c_{3i+2}:=a_i

return c_{3k-1}\cdots c_0
```

Decode sequence of bits using majority rule on consecutive three bit sequences

```
procedure decode3(c_{3k-1}\cdots c_0): a binary string whose length is an integer multiple of 3) for i:=0 to k-1

if exactly two or three of c_{3i}, c_{3i+1}, c_{3i+2} are set to 1

a_i:=1

else

a_i:=0

return a_{k-1}\cdots a_0
```

Give a recursive definition of the set of outputs of the redun3 procedure, Out,

Basis step:		
Recursive step:		

Consider the message m = 0001 so that the sender calculates redun3(m) = redun3(0001) = 000000000111.

Introduce ____ errors into the message so that the signal received by the receiver is _____ but the receiver is still able to decode the original message.

Challenge: what is the biggest number of errors you can introduce?

Building a circuit for line 3 in *decode* procedure: given three input bits, we need to determine whether the majority is a 0 or a 1.

c_{3i}	c_{3i+1}	c_{3i+2}	a_i
1	1	1	
1	1	0	
1	0	1	
1	0	0	
0	1	1	
0	1	0	
0	0	1	
0	0	0	

Circuit

Review

1. Real-life representations are often prone to corruptions. Biological codes, like RNA, may mutate naturally and during measurement; cosmic radiation and other ambient noise can flip bits in computer storage. One way to recover from corrupted data is to exploit redundancy. Consider the following algorithm to introduce redundancy in a string of 0s and 1s.

Create redundancy by repeating each bit three times

```
procedure redun3(a_{k-1}\cdots a_0): a binary string)

for i:=0 to k-1

c_{3i}:=a_i

c_{3i+1}:=a_i

c_{3i+2}:=a_i

return c_{3k-1}\cdots c_0
```

Decode sequence of bits using majority rule on consecutive three bit sequences

```
procedure decode3(c_{3k-1}\cdots c_0): a binary string whose length is an integer multiple of 3)

for i:=0 to k-1

if exactly two or three of c_{3i}, c_{3i+1}, c_{3i+2} are set to 1

a_i:=1

else

a_i:=0

return a_{k-1}\cdots a_0
```

For each of the following, type in your answers precisely including all notational punctuation.

- (a) Give the output of redun3(100).
- (b) If the output of running redun3 is 000000111000111, what was its input?
- (c) Give the output of decode3(100).
- (d) How many distinct possible inputs to decode3 give the output 01?

¹Mutations of specific RNA codons have been linked to many disorders and cancers.

²This RadioLab podcast episode goes into more detail on bit flips: https://www.wnycstudios.org/story/bit-flip

Wednesday January 27

Definition: A **predicate** is a function from a given set (domain) to $\{T, F\}$.

A predicate can be applied, or **evaluated** at, an element of the domain.

Two predicates over the same domain are **equivalent** means they evaluate to the same truth values for all possible assignments of domain elements to the input.

Input	Output		
	P(x)	N(x)	Mystery(x)
x	$[x]_{2c,3} > 0$	$[x]_{2c,3} < 0$	
000	F		T
001	T		T
010	T		T
011	T		F
100	F		F
101	F		T
110	F		F
111	F		T

The domain for each of the predicates $P(x)$, $N(x)$, $Mystery(x)$ is	
Definition : The truth set of a predicate is the collection of all element evaluates to T .	ts in its domain where the predicate
The truth set for the predicate $P(x)$ is	
The truth set for the predicate $N(x)$ is	

The truth set for the predicate Mystery(x) is ______.

Definitions (Rosen 40-45):

The universal quantification of P(x) is the statement "P(x) for all values of x in the domain" and is written $\forall x P(x)$. An element for which P(x) = F is called a **counterexample** of $\forall x P(x)$.

The existential quantification of P(x) is the statement "There exists an element x in the domain such that P(x)" and is written $\exists x P(x)$. An element for which P(x) = T is called a witness of $\exists x P(x)$.

is a true existential quantification. Example:

Statements involving predicates and quantifiers are logically equivalent means they have the same truth value no matter which predicates (domains and functions) are substituted in.

Quantifier version of De Morgan's laws: $|\neg \forall x P(x) \equiv \exists x (\neg P(x))|$

$$|\neg \exists x Q(x) \equiv \forall x (\neg Q(x))$$

Example: is a false universal quantification. It is logically equivalent to

Recall: Each RNA strand is a string whose symbols are elements of the set $B = \{A, C, G, U\}$. The set of all **RNA** strands is called S. The function rnalen that computes the length of RNA strands in S is:

> $rnalen: S \rightarrow \mathbb{Z}^+$ If $b \in B$ then rnalen(b)Basis Step:

Recursive Step: If $s \in S$ and $b \in B$, then rnalen(sb) = 1 + rnalen(s)

Example predicates on S

H(s) = T	Truth set of H is
$L_3(s) = \begin{cases} T & \text{if } rnalen(s) = 3\\ F & \text{otherwise} \end{cases}$	Strand where L_3 evaluates to T is e.g
	Strand where L_3 evaluates to F is e.g
$F_{\mathtt{A}}$ is defined recursively by: Basis step: $F_{\mathtt{A}}(\mathtt{A}) = T, \ F_{\mathtt{A}}(\mathtt{C}) = F_{\mathtt{A}}(\mathtt{G}) = F_{\mathtt{A}}(\mathtt{U}) = F$ Recursive step: If $s \in S$ and $b \in B$, then $F_{\mathtt{A}}(sb) = F_{\mathtt{A}}(s)$	Strand where F_{A} evaluates to T is e.g Strand where F_{A} evaluates to F is e.g
$P_{\texttt{AUC}}$ is defined as the predicate whose truth set is the collection of RNA strands where the string AUC is a substring (appears inside s , in order and consecutively)	Strand where P_{AUC} evaluates to T is e.g Strand where P_{AUC} evaluates to F is e.g

Review

- 1. Consider the following predicates, each of which has as its domain the set of all bitstrings whose leftmost bit is 1
 - E(x) is T exactly when $(x)_2$ is even, and is F otherwise
 - L(x) is T exactly when $(x)_2 < 3$, and is F otherwise
 - M(x) is T exactly when $(x)_2 > 256$ and is F otherwise.
 - (a) What is E(110)?
 - (b) Why is L(00) undefined?
 - i. Because the domain of L is infinite
 - ii. Because 00 does not have 1 in the leftmost position
 - iii. Because 00 has length 2, not length 3
 - iv. Because $(00)_{2,3} = 0$ which is less than 3
 - (c) Is there a bitstring of width (where width is the number of bits) 6 at which M(x) evaluates to T?

Friday January 29

Definition (Rosen p123): The **Cartesian product** of the sets A and B, $A \times B$, is the set of all ordered pairs (a, b), where $a \in A$ and $b \in B$. That is: $A \times B = \{(a, b) \mid (a \in A) \land (b \in B)\}$. The Cartesian product of the sets A_1, A_2, \ldots, A_n , denoted by $A_1 \times A_2 \times \cdots \times A_n$, is the set of ordered n-tuples (a_1, a_2, \ldots, a_n) , where a_i belongs to A_i for $i = 1, 2, \ldots, n$. That is, $A_1 \times A_2 \times \cdots \times A_n = \{(a_1, a_2, \ldots, a_n) \mid a_i \in A_i \text{ for } i = 1, 2, \ldots, n\}$

Recall: Each RNA strand is a string whose symbols are elements of the set $B = \{A, C, G, U\}$. The **set of all RNA strands** is called S. The function *rnalen* that computes the length of RNA strands in S is:

Basis Step: If $b \in B$ then $rnalen: S \to \mathbb{Z}^+$ Recursive Step: If $s \in S$ and $b \in B$, then rnalen(sb) = 1 + rnalen(s)

A function basecount that computes the number of a given base b appearing in a RNA strand s is:

 $basecount: S \times B \rightarrow \mathbb{N}$ Basis Step: If $b_1 \in B, b_2 \in B$ $basecount(b_1, b_2) = \begin{cases} 1 & \text{when } b_1 = b_2 \\ 0 & \text{when } b_1 \neq b_2 \end{cases}$ Recursive Step: If $s \in S, b_1 \in B, b_2 \in B$ $basecount(sb_1, b_2) = \begin{cases} 1 + basecount(s, b_2) & \text{when } b_1 = b_2 \\ basecount(s, b_2) & \text{when } b_1 \neq b_2 \end{cases}$

L with domain $S \times \mathbb{Z}^+$ is defined by, for $s \in S$ and $n \in \mathbb{Z}^+$,

$$L(s,n) = \begin{cases} T & \text{if } rnalen(s) = n \\ F & \text{otherwise} \end{cases}$$

Element where L evaluates to T:

Element where L evaluates to F:

BC with domain _____ is defined by, for $s \in S$ and $b \in B$ and $n \in \mathbb{N}$,

$$BC(s, b, n) = \begin{cases} T & \text{if } basecount(s, b) = n \\ F & \text{otherwise} \end{cases}$$

Element where BC evaluates to T:

Element where BC evaluates to F:

Notation: for a predicate P with domain $X_1 \times \cdots \times X_n$ and a n-tuple (x_1, \ldots, x_n) with each $x_i \in X$, we write $P(x_1, \ldots, x_n)$ to mean $P((x_1, \ldots, x_n))$.

$\exists t \ BC(t)$ In E	English:			
Witness that prove		ification is true:		
$\forall (s,b,n) \ (\ BC(s,b))$	(n,n)) In English:			
Counterexample th	nat proves this universa	l quantification is false	:	
New predicates from old $BC(s, b, n)$ means $basecount(s, b) = n$.				
Predicat	e	Domain	Example domain element where predicate is T	
base count	(s,b) = 3			
base count	(.)			
$\exists n \in \mathbb{N} \ (b$	asecount(s,b) = n)			
$\forall b \in B \ (be$	asecount(s,b) = 1)			
Alternating qua	ntifiers	$\forall s \exists n \ BC(s, \mathtt{A}, n)$		
In English:				
		$\exists n \forall s \ BC(s, \mathtt{U}, n)$		
In English:				
Evaluate each qua	ntified statement as T	or F .		
	$\forall s \ \forall b \ \exists n \ BC(s,b,n)$	$\forall s \ \forall n \ \exists b \ BC(s,b,n)$	$\forall b \ \forall n \ \exists s \ BC(s,b,n)$	
	$\exists s \ \forall b \ \exists n \ BC(s,b,n)$	$\forall s \; \exists n \; \forall b \; BC(s,b,n)$	$\exists b \; \exists n \; \forall s \; BC(s,b,n)$	

Extra example: Write the negation of each of the statements above, and use De Morgan's law to find a logically equivalent version where the negation is applied only to the BC predicate (not next to a quantifier).

Review

1. Recall that S is defined as the set of all RNA strands, strings made of the bases in $B = \{A, U, G, C\}$. Define the functions *mutation*, *insertion*, and *deletion* as described by the pseudocode below:

```
procedure mutation(b_1 \cdots b_n): a RNA strand, k: a positive integer, b: an element of B)
    for i := 1 to n
       if i = k
         c_i \ := \ b
4
       else
5
          c_i := b_i
6
    return c_1 \cdots c_n {The return value is a RNA strand made of the c_i values}
1
    procedure insertion (b_1 \cdots b_n): a RNA strand, k: a positive integer, b: an element of B)
       for i := 1 to n
3
         c_i := b_i
5
       c_{n+1} := b
6
    else
       \mathbf{for} \ i \ := \ 1 \ \mathbf{to} \ k-1
8
         c_i := b_i
       c_k := b
       {\bf for} \ i \ := \ k+1 \ {\bf to} \ n+1
10
11
         c_i := b_{i-1}
    return c_1 \cdots c_{n+1} {The return value is a RNA strand made of the c_i values}
12
    procedure deletion(b_1 \cdots b_n): a RNA strand, k: a positive integer)
1
2
       m := n
       for i := 1 to n
         c_i := b_i
    else
6
       m := n - 1
       for i := 1 to k-1
         c_i := b_i
       for i := k to n-1
10
         c_i := b_{i+1}
11
    return c_1 \cdots c_m {The return value is a RNA strand made of the c_i values}
```

For this question, we will use the following predicates.

 F_{A} with domain S is defined recursively by:

```
Basis step: F_{\mathbf{A}}(\mathbf{A}) = T, F_{\mathbf{A}}(\mathbf{C}) = F_{\mathbf{A}}(\mathbf{G}) = F_{\mathbf{A}}(\mathbf{U}) = F
Recursive step: If s \in S and b \in B, then F_{\mathbf{A}}(sb) = F_{\mathbf{A}}(s)
```

 P_{AUC} with domain S is defined as the predicate whose truth set is the collection of RNA strands where the string AUC is a substring (appears inside s, in order and consecutively)

L with domain $S \times \mathbb{Z}^+$ is defined by, for $s \in S$ and $n \in \mathbb{Z}^+$,

$$L(s,n) = \begin{cases} T & \text{if } rnalen(s) = n \\ F & \text{otherwise} \end{cases}$$

Mut with domain $S \times S$ is defined by, for $s_1 \in S$ and $s_2 \in S$,

$$Mut(s_1, s_2) = \exists k \in \mathbb{Z}^+ \exists b \in B(\ mutation(s_1, k, b) = s_2)$$

Ins with domain $S \times S$ is defined by, for $s_1 \in S$ and $s_2 \in S$,

$$Ins(s_1, s_2) = \exists k \in \mathbb{Z}^+ \exists b \in B(insertion(s_1, k, b) = s_2)$$

Del with domain $S \times S$ is defined by, for $s_1 \in S$ and $s_2 \in S$,

$$Del(s_1, s_2) = \exists k \in \mathbb{Z}^+ (deletion(s_1, k) = s_2)$$

- (a) Which of the following is true? (Select all and only that apply.)
 - i. $F_{A}(AA)$
 - ii. $F_{A}(AC)$
 - iii. $F_{A}(AG)$
 - iv. $F_{A}(AU)$
 - v. $F_{A}(CA)$
 - vi. $F_{A}(CC)$
 - vii. $F_{A}(CG)$
 - viii. $F_{A}(CU)$
- (b) Which of the following is true? (Select all and only that apply.)
 - i. $\exists s \in S \ \exists n \in \mathbb{Z}^+ \ (L(s,n))$
 - ii. $\exists s \in S \ \forall n \in \mathbb{Z}^+ \ (L(s,n))$
 - iii. $\forall n \in \mathbb{Z}^+ \ \exists s \in S \ (L(s,n))$
 - iv. $\forall s \in S \ \exists n \in \mathbb{Z}^+ \ (L(s,n))$
 - v. $\exists n \in \mathbb{Z}^+ \ \forall s \in S \ (L(s,n))$
 - vi. $\forall s \in S \ \forall n \in \mathbb{Z}^+ \ (L(s,n))$
- (c) Which of the following is true? (Select all and only that apply.)
 - i. $\exists s \in S \ Mut(s,s)$
 - ii. $\forall s \in S \ Mut(s,s)$
 - iii. $\exists s \in S \ Ins(s, A)$
 - iv. $\exists s \in S \ Ins(A, s)$
 - v. $\exists s \in S \ Del(s, A)$
 - vi. $\forall s \in S \ Del(s, A)$