Kevin Chau
cs70-zh
section 6
23816929

# Homework 1

Partners: Unis Barakat, Ahdil Hameed, Sam Drake

Problem $\boxed{1.}$ I have verified that the registration information for my CS 70 instructional account cs-70 is complete and correct

Kevin Chau

Problem $\boxed{2.}$ Professor Sahai's second favorite number is $\boxed{\ln 2}$ because it is used to translate between the natural logarithmic base of e and binary (base 2) information.

Problem $\boxed{3.}$ Basic implications:

part 1 1) All that is gold does not glitter
All that glitters is not gold

explanation: the first statement says that if something is gold, it cannot glitter; that is, none of the gold can glitter. The second statement says that if something glitters, it cannot be gold; nothing that glitters is gold. The statements are logically equivalent, they are contrapositives, so there is no real difference

2) Every dragon is either fire-breathing or plant-eating
Every dragon that is not fire-breathing is plant-eating

explanation: The first statement is a logical inclusive "or", meaning a dragon can be firebreathing only, plant-eating only, or both firebreathing and plant eating. The second sentence is an implication, saying that a dragon that doesn't breathe fire must eat plants, but a dragon that is fire breathing could be that exclusively or also plant eating. They are logically equivalent, so there is no real difference.

Part 2 1) $\forall x\ Gold(x) \Rightarrow \neg Glitter(x)$ ("All that is gold does not Glitter")
$\forall x\ Glitter(x) \Rightarrow \neg Gold(x)$ ("All that glitters is not Gold")

$$(Gold(x) \Rightarrow \neg Glitter(x)) \equiv (\neg\neg Glitter(x) \Rightarrow \neg Gold(x)) \equiv (Glitter(x) \Rightarrow \neg Gold(x))$$

2) $\forall x\ FireBreathing(x) \vee PlantEating(x)$ ("Every dragon is either ... or ... ")
$\forall x\ \neg FireBreathing(x) \Rightarrow PlantEating(x)$ ("...is not Firebreathing is plant eating")

$$\neg F(x) \Rightarrow P(x) \equiv \neg\neg F(x) \vee P(x) = F(x) \vee P(x)$$ [equivalence of sentences]

Problem 4

Part 1

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

From the truth tables below, we see that the sum of products expression x and the product of sums expression z are both equivalent and identical to the truth table for the output Y.

1. $(\neg A \wedge \neg B \wedge C) \vee (\neg A \wedge B \wedge \neg C) \vee (A \wedge \neg B \wedge C) \vee (A \wedge B \wedge C) = X$

| A B C | $(\neg A \wedge \neg B \wedge C)$ | $(\neg A \wedge B \wedge \neg C)$ | $(A \wedge \neg B \wedge C)$ | $(A \wedge B \wedge C)$ | X |
|---|---|---|---|---|---|
| 0 0 0 | 0 | 0 | 0 | 0 | 0 |
| 0 0 1 | 1 | 0 | 0 | 0 | 1 |
| 0 1 0 | 0 | 1 | 0 | 0 | 1 |
| 0 1 1 | 0 | 0 | 0 | 0 | 0 |
| 1 0 0 | 0 | 0 | 0 | 0 | 0 |
| 1 0 1 | 0 | 0 | 1 | 0 | 1 |
| 1 1 0 | 0 | 0 | 0 | 0 | 0 |
| 1 1 1 | 0 | 0 | 0 | 1 | 1 |

2. $(A \vee B \vee C) \wedge (A \vee \neg B \vee \neg C) \wedge (\neg A \vee B \vee C) \wedge (\neg A \vee \neg B \vee C) = Z$

| A B C | $(A \vee B \vee C)$ | $(A \vee \neg B \vee \neg C)$ | $(\neg A \vee B \vee C)$ | $(\neg A \vee \neg B \vee C)$ | Z |
|---|---|---|---|---|---|
| 0 0 0 | 0 | 1 | 1 | 1 | 0 |
| 0 0 1 | 1 | 1 | 1 | 1 | 1 |
| 0 1 0 | 1 | 1 | 1 | 1 | 1 |
| 0 1 1 | 1 | 0 | 1 | 1 | 0 |
| 1 0 0 | 1 | 1 | 0 | 1 | 0 |
| 1 0 1 | 1 | 1 | 1 | 1 | 1 |
| 1 1 0 | 1 | 1 | 1 | 0 | 0 |
| 1 1 1 | 1 | 1 | 1 | 1 | 1 |

part 2 Any truth table can be represented by a sum of products or a product of sums expression because of the fact that an "and" can only be True (1) when all the parts are True and an "or" is true when at least 1 subexpression is true.

Sum of Products Method:

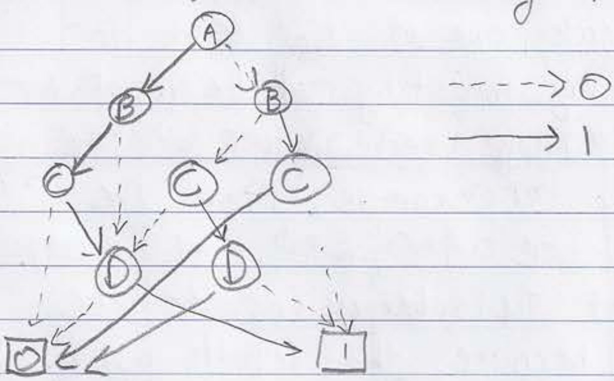
Look at the truth table and find where the output is 1. Since the subexpressions in the sum (or's) are all products (and's), we can create one of the terms for each of the output 1's by negating all the input values that are 0 and "anding" those inputs with the inputs that already have value 1. For example, from the given truth table, we see $A=0$, $B=0$, $C=1$ outputs 1. Then we know that one of the subexpressions to be "ORed" (summed will be $\neg A \land \neg B \land C$ because this results in a 1, so the total sum (OR's) will be 1 no matter the value of the other subexpressions. We also know that any other combinations of inputs ~~for~~ this subexpression will result in a 0 because of the properties of "and", so we are guarenteed that this will not create an incorrect 1 output for an SOP that should be 0.

Product of sums Method:

The entire product is 0 when at least one subexpression sum is a 0, so we just need to find which inputs output a 0 and make the appropriate sum by negating the inputs with value 1 and leaving unchanged those with value 0. For example, $A=0$, $B=1$, $C=1$ gives $Y=0$, so we know that a subexpression sum $A \lor \neg B \lor \neg C$ can be used to create a POS. These sums generated will make a 1 for every other input combination, so the inputs that output 1 will be guaranteed to have a value of 1 for all subexpressions, thus giving us the correct truth table part 3

- In the Sum of Products Form, the number of product terms is equal to the number of output 1's because each product that creates a 1 will automatically make the total sum have a value of 1.
- In the Product of Sums Form, the number of Sum terms is equal to the number of output 0s, because each sum that makes a 0 will automatically make the total product 0.

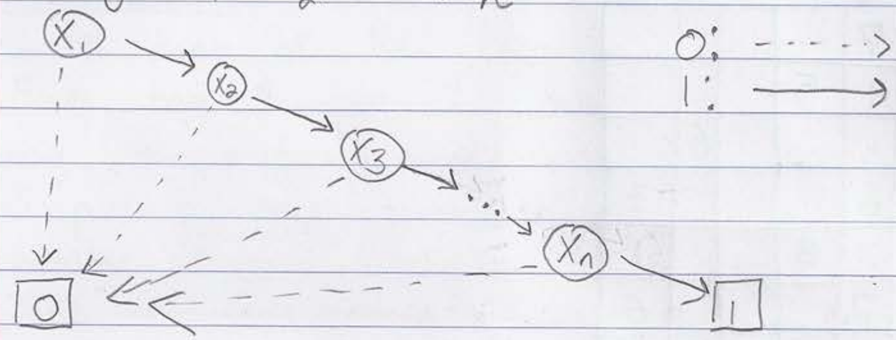
Problem 5




1.

| A | B | C | D | output |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

$y = x_1 \wedge x_2 \wedge \ldots \wedge x_n$

2. 


Binary Decision Diagrams show the short-circuiting nature of "and" and "or"; that is, for "and", any input 0 will make the output 0 no matter the other inputs. The binary diagram is a quicker way to see this, and it is also a much more compact way to represent a function than a truth table by exploiting the properties of "and"/"or". A function with many inputs would have quite a large truth table, where you would need to write out all the 0s & 1s, whereas a BDD lets you appropriately skip looking at the other inputs when they are irrelevant, getting you straight to an output.

Problem 6 1.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 8 | 6 | 9 | 7 | 2 | 4 | 3 | 5 |
| 9 | 4 | 7 | 6 | 3 | 5 | 2 | 1 | 8 |
| 3 | 5 | 2 | 1 | 8 | 4 | 6 | 9 | 7 |
| 4 | 9 | 5 | 8 | 2 | 1 | 7 | 6 | 3 |
| 6 | 2 | 3 | 7 | 4 | 9 | 8 | 5 | 1 |
| 8 | 7 | 1 | 3 | 5 | 6 | 9 | 2 | 4 |
| 7 | 6 | 8 | 5 | 9 | 3 | 1 | 4 | 2 |
| 2 | 3 | 9 | 4 | 1 | 8 | 5 | 7 | 6 |
| 5 | 1 | 4 | 2 | 6 | 7 | 3 | 8 | 9 |

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 2. A | 5 | 3 | | | 7 | | | | |
| B | 6 | | | 1 | 9 | 5 | | | |
| C | | 9 | 8 | | | | | 6 | |
| D | 8 | | | | 6 | | | | 3 |
| E | 4 | | | 8 | | 3 | | | 1 |
| F | 7 | | | | 2 | | | | 6 |
| G | | 6 | | | | | 2 | 8 | |
| H | | | | 4 | 1 | 9 | 3 | | 5 |
| I | | | | | 8 | | 6 | 7 | 9 |

Neither of these squares can hold a 1, but the 3x3 box needs one. Hence the puzzle is unsolvable.

Proof that the puzzle is unsolvable: (By Contradiction)

P: the puzzle is unsolvable

Start by assuming $\neg P$ is true – the puzzle is solvable. Then, the rules of sodoku will hold (R proposition) and every 3x3 box can only contain one of each digit, as well as every row and column. Following these rules, we see that square I7 must have a 6 because the encompassing 3x3 box (rows G, H, I + columns 7, 8, 9) is missing a 6, 4, and 1, but Row G and columns 8 and 9 already contain a 6. Hence I7 is the only square left in this 3x3 box for the 6. This leaves squares H8 and G9 to be filled with a 4 or a 1. But Row H and Column 9 already have 1s, so for this puzzle to be solvable, we would need to break the rules, therefore $\neg R$ is true (the sudoku rules do not apply). Then we have a contradiction: both R and $\neg R$ are true, but we know $R \wedge \neg R$ is always False. Therefore the original assumption $\neg P$ that the puzzle is solvable is not true. Hence P is true, so the puzzle is in fact $\boxed{\text{unsolvable}}$, by contradiction.

Problem 7

1. Label Each of the bags a number 1 through 10. From bag 1 put 1 coin on the scale, From bag 2 put 2 coins on the scale, from bag 3 put 3 coins, and so on for all the bags (the number of coins from each bag is equal to the bag's number label). Take a measurement from the scale and read the weight. Since each fake coin is a milligram heavier, you can tell which bag the fake coins came from by figuring out the number of milligrams that the collection of coins is heavy compared to if all the coins where genuine. For example, if the scale read something that ended in .006g, you would know that the fake coins are from the 6th bag since we put 6 coins (each .001g too heavy) on the scale from that bag. All of this is equivalent to the calculation:

$$[(\text{Final weight}) \bmod 10] \times 1000 = \#\text{ of bag with fakes}$$

Of course we know that in total we will place 55 coins on the scale, so if all the coins were real, they would weigh 550 grams collectively.

2. We can prove that this will always identify the correct bag by using a direct Proof:

Direct Proof:

P: place $n$ coins from the $n$th bag on the scale, $n \in [1, 10]$

Q: the bag# is always correctly determined

- We need to prove that P implies Q ($P \Rightarrow Q$)
- Assume that we place $n$ coins from each $n$th bag, with each bag having a label 1-10.

IF all the coins were real, they would weigh 550 grams togethe

- Since there are fake coins, the scale will read weight $= 550 + (.001)x$
- we know there are fake coins, so $x$ cannot be 0 and therefore is an integer number since each coin is .001 grams heavy. $X$ must be a value 1 through 10.

Since we uniquely identified each bag by placing $n$ coins from the $n$th bag, we can conclude that $x = n =$ the bag # with the fake coins. Because every possibility for a fake bag corresponds to a unique extraneous weight of the form $(.001)x$, we know we can always identify the bag ($x$ is never an +integer above 10). Thus we have proven directly that this method will always work.

Problem 8

1. A set is equisplittable if its elements add up to an even number
<u>Disprove by counterexample:</u>
Let our set be $[2, 4]$. $2+4=6$, so the elements sum to an even number. However, the set is not equisplittable since $2 \neq 6$. Therefore the original statement is False.

2. A set is equisplittable only if its elements add up to an even number.
Equisplittable $\Rightarrow$ even sum (an even sum is necessary for equisplitability)
<u>Direct Proof:</u>
P: A set is equisplittable
Q: that set's elements sum to an even number
Assume P - we have a set that is equisplittable.
- Then we know that some of the elements, say $x_1 \cdots x_k$ sum to some number $a \in \mathbb{Z}$ such that the rest of the numbers $x_{k+1} \cdots x_n$ also sum up to the same number $a \in \mathbb{Z}$.
- Then the total sum of all the elements is

$$(x_1 + \cdots + x_k) + (x_{k+1} + \cdots + x_n) = a + a = 2a$$

By definition, $2a$ is even ($a$ can be odd or even), so it is a necessity that an equisplittable set sums to an even number. Hence $P \Rightarrow Q$ and we have proven the original statement

3. A set is not equisplittable only if its elements add up to an even number.

$\neg$ Equisplittable $\Rightarrow$ EvenSum

Disproof by Counter example:

Consider the set [1, 2, 3, 4, 5]. The set is not equisplittable because the sum of the set is 15, an odd number, which by definition does not result in a integer when divided by 2. Hence this non-splittable set does not have an even sum so the original statement is not true in general.

4. A set is not equisplittable if and only if its elements do not add up to an even number

$\neg$ Equisplittable $\Leftrightarrow$ $\neg$EvenSum

Forward Direction: $\neg$ Equisplittable $\Rightarrow$ $\neg$ EvenSum

Contrapositive: a set that sums to an even number implies an equisplittable set: EvenSum $\Rightarrow$ Equisplittable.

From part I, we saw that this statement is false (counter-ex [0, 4]). Since the contrapositive is equivalent to the original forword implication, we know that that implication was false. So $\neg$EvenSum $\not\Rightarrow$ $\neg$Equisplittable

Backward Direction: $\neg$ EvenSum $\Rightarrow$ $\neg$ Equisplittable

contrapositive: Equisplittable $\Rightarrow$ EvenSum

An equisplittable set has elements that sum to an even number. This was proven in part 2. Assume our set is equisplittable. Then the elements sum to $2a$ for some $a \in \mathbb{Z}$ such that some elements sum to $a$ and all the rest sum to $a$ as wel

By definition $2a$, is an even number. Since the contrapositive of the backward implication is true, the backwards implication itself is true.

The implication does not go both ways — a set that does not even-sum is not equisplittable, but a nonequisplittable set does not necessarily have to be odd summed. Thus the two way implication "if-and-only-if" is $\boxed{\text{False}}$.

$$\neg \text{Equisplittable} \not\Rightarrow \neg \text{EvenSum}$$
$$\neg \text{Even sum} \Longrightarrow \neg \text{Equisplittable}$$
$$\therefore \neg \text{Equisplittable} \not\Leftrightarrow \neg \text{EvenSum}$$