CS 70 Discrete Mathematics and Probability Theory
Summer 2020 Course Notes

DIS 2A

1 Unions and Intersections

For each of the following, decide if the expression is "Always Countable", "Always Uncountable", "Sometimes Countable, Sometimes Uncountable".

For the "Always" cases, prove your claim. For the "Sometimes" case, provide two examples – one where the expression is countable, and one where the expression is uncountable.

(a) $A \cap B$, where A is countable, and B is uncountable

(b) $A \cup B$, where A is countable, and B is uncountable

(c) $\bigcap_{i \in A} S_i$ where A is a countable set of indices and each S_i is an uncountable set.

Solution:

(a) Always countable. $A \cap B$ is a subset of A, which is countable.

(b) Always uncountable. $A \cup B$ is a superset of B, which is uncountable.

(c) Sometimes countable, sometimes uncountable.

Countable: When the S_i are disjoint, the intersection is empty, and thus countable. For example, let $A = \mathbb{N}$, let $S_i = \{i\} \times \mathbb{R} = \{(i,x) \mid x \in \mathbb{R}\}$. Then, $\bigcap_{i \in A} S_i = \varnothing$.

Uncountable: When the S_i are identical, the intersection is uncountable. Let $A = \mathbb{N}$, let $S_i = \mathbb{R}$ for all i. $\bigcap_{i \in A} S_i = \mathbb{R}$ is uncountable.

2 Counting Cartesian Products

For two sets A and B, define the cartesian product as $A \times B = \{(a,b) : a \in A, b \in B\}$.

(a) Given two countable sets A and B, prove that $A \times B$ is countable.

(b) Given a finite number of countable sets $A_1, A_2, \ldots, A_n$, prove that

$$A_1 \times A_2 \times \cdots \times A_n$$

is countable.

(c) Consider an infinite number of countable sets: $B_1, B_2, \ldots$. Under what condition(s) is $B_1 \times B_2 \times \cdots$ countable? Prove that if this condition is violated, $B_1 \times B_2 \times \cdots$ is uncountable.

Solution:

(a) As shown in lecture, $\mathbb{N} \times \mathbb{N}$ is countable by creating a zigzag map that enumerates through the pairs: $(0,0),(1,0),(0,1),(2,0),(1,1),\ldots$. Since A and B are both countable, there exists a bijection between each set and a subset of $\mathbb{N}$. Thus we know that $A \times B$ is countable because there is a bijection between a subset of $\mathbb{N} \times \mathbb{N}$ and $A \times B : f(i,j) = (A_i, B_j)$. We can enumerate the pairs (a,b) similarly.

(b) Proceed by induction.
Base Case: $n = 2$. We showed in part (a) that $A_1 \times A_2$ is countable since both A_1 and A_2 are countable.
Induction Hypothesis: Assume that for some $n \in \mathbb{N}$, $A_1 \times A_2 \times \cdots \times A_n$ is countable.
Induction Step: Consider $A_1 \times \cdots \times A_n \times A_{n+1}$. We know from our hypothesis that $A_1 \times \cdots \times A_n$ is countable, call it $C = A_1 \times \cdots \times A_n$. We proved in part (a) that since C is countable and A_{n+1} are countable, $C \times A_{n+1}$ is countable, which proves our claim.

(c) If any of the B_i are empty, then the infinite Cartesian Product is also empty. Hence, we assume that none of the B_i are empty.

A necessary and sufficient condition for the infinite Cartesian Product to be countable is for all but finitely many of the B_i to have exactly one element.

Necessary: Suppose not; then, infinitely many B_i have at least two elements. Notice that for any B_i with only one element, every element of the infinite Cartesian Product must necessarily use the single element of B_i. Therefore, we can ignore each B_i with only one element and assume that every B_i has at least two elements. Proceed with a diagonalization argument by assuming for the sake of contradiction that $B_1 \times B_2 \times \cdots$ is countable and its elements can be enumerated in a list:

$$
\begin{aligned}
&(b_{1,1}, b_{2,1}, b_{3,1}, b_{4,1}, \ldots)\\
&(b_{1,2}, b_{2,2}, b_{3,2}, b_{4,2}, \ldots)\\
&(b_{1,3}, b_{2,3}, b_{3,3}, b_{4,3}, \ldots)\\
&(b_{1,4}, b_{2,4}, b_{3,4}, b_{4,4}, \ldots)\\
&\vdots
\end{aligned}
$$

where $b_{i,j}$ represents the item from set B_i that is included in the jth element of the Cartesian Product. Now consider the element $(\overline{b_{1,1}}, \overline{b_{2,2}}, \overline{b_{3,3}}, \overline{b_{4,4}}, \ldots)$, where $\overline{b_{i,j}}$ represents any item from set B_i that differs from $b_{i,j}$ (i.e. any other element in the set). This is a valid element that should exist in the Cartesian Product $B_1, B_2, \ldots$, yet it is not in the enumerated list. This is a contradiction, so $B_1, B_2, \ldots$ must be uncountable.

Notice that this relies on the fact that each set B_i has some other item we can choose when constructing our diagonal element.

Sufficient: If all but finitely many of the B_i have one element, then there are only a finite number of B_i which have more than one element. Again, we can ignore the B_i with only one element. By the previous part, we showed that the Cartesian Product of a finite number of countable sets is countable, so the infinite Cartesian Product in this case is countable.

3 Hello World!

Determine the computability of the following tasks. If it's not computable, write a reduction or self-reference proof. If it is, write the program.

(a) You want to determine whether a program P on input x prints "Hello World!". Is there a computer program that can perform this task? Justify your answer.

(b) You want to determine whether a program P prints "Hello World!" before running the kth line in the program. Is there a computer program that can perform this task? Justify your answer.

(c) You want to determine whether a program P prints "Hello World!" in the first k steps of its execution. Is there a computer program that can perform this task? Justify your answer.

Solution:

(a) Uncomputable. We will reduce `TestHalt` to `PrintsHW`(P,x).

```
TestHalt(P, x):
  P'(x):
    run P(x) while suppressing print statements
    print("Hello World!")
  if PrintsHW(P', x):
    return true
  else:
    return false
```

If `PrintsHW` exists, `TestHalt` must also exist by this reduction. Since `TestHalt` cannot exist, `PrintsHW` cannot exist.

(b) Uncomputable. Reduce `PrintsHW`(P,x) from part (a) to this program `PrintsHWByK`(P,x,k).

```
PrintsHW(P, x):
  for i in range(len(P)):
    if PrintsHWByK(P, x, i):
      return true
  return false
```

(c) Computable. You can simply run the program until k steps are executed. If P has printed "Hello World!" by then, return true. Else, return false.

The reason that part (b) is uncomputable while part (c) is computable is that it's not possible to determine if we ever execute a specific line because this depends on the logic of the program, but the number of computer instructions can be counted.