CS 4104: Data and Algorithm Analysis

Clifford A. Shaffer

Department of Computer Science
Virginia Tech
Blacksburg, Virginia

Fall 2010

Countable vs. Uncountably Infinite Sets

Two sets have the **same cardinality** if there is a bijection between them.

Notation: $|A| = |B|$.

This concept can also be applied to infinite sets.

Example: Let Odd and Even be the sets of odd and even natural numbers, respectively.
Then, $|\text{Odd}| = |\text{Even}|$ because the function $f : |\text{Odd} \rightarrow \text{Even}|$ defined by $f(x) = x - 1$ is a bijection.

How about $|\text{Even}| = |\mathbb{N}|$?

Counting Infinite Sets

A set C is **countable** if it is finite or if $|C| = |\mathbb{N}|$.

If a set is not countable, then it is **uncountable**.

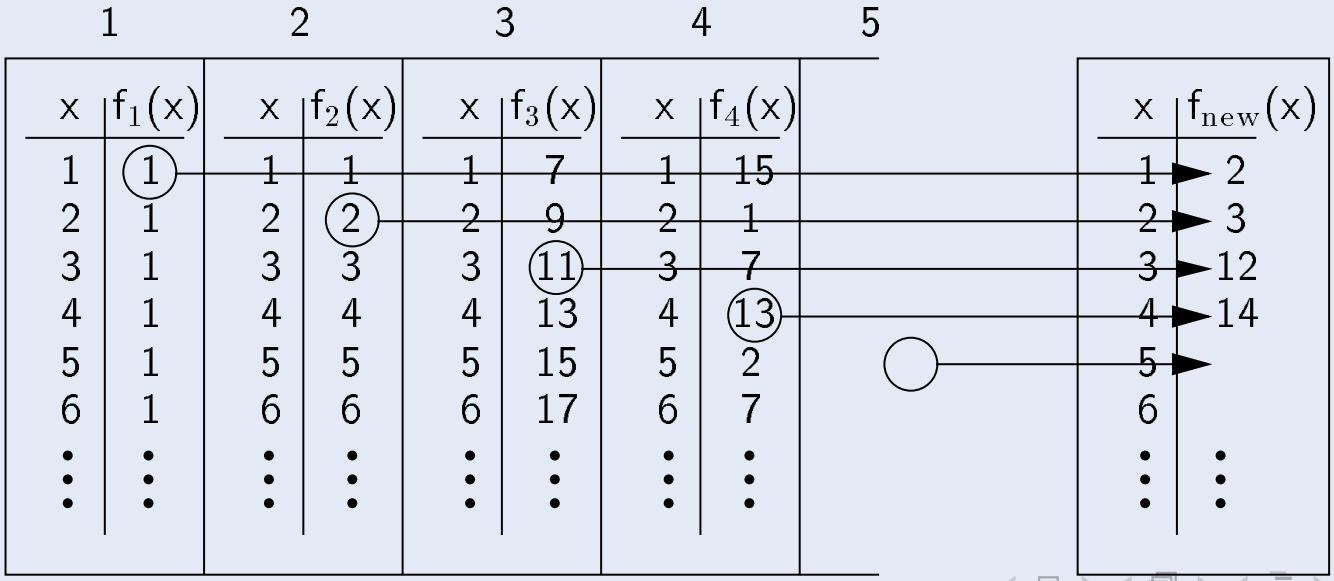
If A is a finite alphabet, then A^* is countably infinite.

Proof: Arrange the strings in order by length, and within a given length by alphabetical order. This provides a bijection.

As a corollary, the set of all computer programs is countable.

More Functions than Programs

- Consider set of functions $f(x) = y$ for x, y natural numbers.
- The set of such functions is uncountable.
- Diagonalization argument
- Not all functions on natural numbers are computable.



Halting Problem for Programs

Does the following terminate?

```
while (n > 1)
  if (ODD(n))
    n = 3 * n + 1;
  else
    n = n / 2;
```

Can a **C**++ program be written to solve the following problem?

Halting Problem:

- Input: A program P and input X.
- Output: “Halts” if P halts when run with X as input. “Does not Halt” otherwise.

Halting Problem Proof

Theorem: There is no program to solve the Halting Problem.

Proof: (by contradiction).

Assumption: There is a **C++** program that solves the Halting Problem.

```
bool halt(char* prog, char* input)
{
  Code to solve halting problem
  if (prog does halt on input) then
    return(TRUE);
  else
    return(FALSE);
}
```

Two More Procedures

```
bool selfhalt(char *prog) {
  // Return TRUE if program halts
  //  when given itself as input.
  if (halt(prog, prog))
    return(TRUE);
  else
    return(FALSE);
}

void contrary(char *prog) {
  if (selfhalt(prog))
    while(TRUE); // Go into an infinite loop
}
```

The Punchline

- What happens when function `contrary` is run on itself?
- Case 1: `selfhalt` returns TRUE.
  - `contrary` will go into an infinite loop.
  - This contradicts the result from `selfhalt`.
- `selfhalt` returns FALSE.
  - `contrary` will halt.
  - This contradicts the result from `selfhalt`.
- Either result is impossible.
- The only flaw in this argument is the assumption that `halt` exists.
- Therefore, `halt` cannot exist.

Computability Reduction Proof

Given arbitrary program M, does it halt on the EMPTY input?

This is uncomputable. Proof:

- Suppose that program M_0 determines if M halts on the EMPTY input.
- Given arbitrary program M and string w, we can create a new program M_w that operates as follows on empty input:
  - Write w into a static variable.
  - Simulate the execution of M.
- So, we can take arbitrary program M and string w, create M_w, and invoke M_0 on M_w (with empty input) to solve the original halting problem.
- Thus, M_0 must not exist.

Another Reduction Proof

Does there exist ANY input for which an arbitrary program halts?

Proof that this is uncomputable:

- Suppose that program M_0 could decide if arbitrary program M halts on ANY input.
- We can take an arbitrary program M and string w, and modify it so that it ignores its input before proceeding.
- Thus, arbitrary program M is modified to be M' that effectively is M operating on the empty input.
- Thus, we can take arbitrary program M and string w, modify it to become M' and feed that to M_0 to solve the problem of deciding if M halts on the empty input.
- We already know that is undecidable.
- Thus, M_0 cannot exist.

Other Noncomputable Functions

Does a program halt on EVERY input?

Do two programs compute the SAME function?

Does a particular line in a program get executed?

Does a program compute a particular function?

Does a program contain a “computer virus”?