

Exploring the Powers and Limitations of Computers

Ming-Yang Kao

4/5/2011

Basic Questions

- 1. Correctness:** Given a problem Q and a computer program P , does P solve Q correctly?
- 2. Computability:** Given a problem Q , is there a computer program that can solve Q ?
- 3. Efficiency:** Given a problem Q and a computer program P , how efficiently can P solve Q ?
- 4. Efficiency:** Given a problem Q , how efficiently can Q be solved by a computer program?

Basic Concepts and Results

1. **Examples of algorithmic problems**
2. **Types of algorithmic problems**
3. **Examples of algorithmic problems which no computer programs can solve**
4. **Concepts of efficiency**

How to Specify an Algorithmic Problem?

Always follow the following format:

Input: ...

Output: ...

Examples of Algorithmic Problems

Input: two Integers J and K .

Output: the numbers $J + K$, $J - K$, $J * K$, and J / K .

Examples of Algorithmic Problems

Input: two Integers J and K.

Output: the number $J^2 + 2K$.

Examples of Algorithmic Problems

Input: a number y .

Output: the number $3y^4 + 2y^3 - 2y^2 + y - 10$.

Examples of Algorithmic Problems

Input: an equation $ax^2 + bx + c = 0$.

Output: the solutions to the equation.

Examples of Algorithmic Problems

Input: a positive integer K .

Output: the sum of all the integers from 1 to K .

Examples of **Decision** Problems

Input: a positive integer K .

Output: 'YES' if K is a prime and 'NO' if K is not a prime.

Question to Think about:

How fast can we solve this problem?

Examples of Algorithmic Problems

Input: a list L of English words.

Output: the list L sorted in alphabetic order.

Examples of Algorithmic Problems

Input: a list L of numbers.

Output: the list L sorted in increasing order.

Question to Think about:

How fast can we solve this problem?

Examples of Algorithmic Problems

Input: two texts in English.

Output: a list of the words that appear in both texts.

Question to Think about:

How fast can we solve this problem?

Examples of **Search** Problems

Input: a road map for cities, with distance attached to road segments.

Output: a trip that starts from Chicago, passes through all cities exactly once, and returns to Chicago.

Question to Think about:

How fast can we solve this problem?

Examples of **Optimization** Problems

Input: a road map for cities, with distance attached to road segments.

Output: a trip that starts from Chicago, passes through all cities, returns to Chicago, and has the smallest length among all such trips.

Question to Think about:

How fast can we solve this problem?

Examples of **Decision** Problems

Input: a road map for cities, with distance attached to road segments, and a number K .

Output: 'YES' if it is possible to take a trip that starts from Chicago, passes through all cities, returns to Chicago, and has a total length no greater than K miles; and 'NO' if such a trip is impossible.

Question to Think about:

How fast can we solve this problem?

Examples of **Decision** Problems

- **Input:** a string P of characters
- **Output:** Is P a Java program which compiles?

Question to Think about:

Can this problem be solved by any computer program at all?

Examples of **Decision** Problems

Input: a string P of characters.

Output: Is P a Java program which compiles and always terminates on every input?

Question to Think about:

Can this problem be solved by any computer program at all?

Examples of **Decision** Problems

Input: two strings P and w of characters.

Output: Is P a Java program which compiles and terminates on w as input?

Question to Think about:

Can this problem be solved by any computer program at all?

A Limitation of Modern Computers

There exists at least one decision problem which no computer program can correctly solve.

A Limitation of Modern Computers

There exist **MANY** decision problems which no computer program can correctly solve.

Three Simple but Useful Ideas

- **We will think of a computer program as an integer.**
- **We will think of an input to a computer program as an integer.**
- **We will think of a set of integers as a decision problem.**

Think of a Computer Program as an Integer

1. Pick any programming language we like. Let B be the alphabet used by this programming language.
2. Suppose that B has k characters. Then each string of characters of B can be viewed as an integer of base k .
3. Each program P is simply a string of characters over B . So, each computer program P can be viewed as an integer.
4. However, not every integer is a valid program.

Think of an Input as an Integer

1. A = the alphabet which we use to specify an input to a computer program.
2. Suppose that A has h characters. Then each string of characters of A can be viewed as an integer of base h .
3. Each input w to a computer program is simply a string of characters over A . So, each input w can be viewed as an integer.
4. However, not every integer is a valid input.

Think of a Set of Integers as a Decision Problem

S = a set of integers.

Decision Problem S :

- **Input:** an integer w
- **Output:** Is w in S ?

We say that a computer program P correctly solves this decision problem if for every integer w , P gives the correct YES/NO answer.

How Many Decision Problems Are There?

- As many as there are subsets of integers,

WHY?

- Which is as many as there are real numbers.

WHY?

How Many Java Programs Are there?

- No more than the number of integers.

WHY?

Fundamental Theorem #1

- The # of real numbers is much larger than the # of integers. (WHY?)
- Therefore, the # of decision problems is much larger than the # of integers, which is no less than the # of computer programs. (WHY?)
- Therefore, there is at least one decision problem which cannot be solved by any computer program. (WHY?)

Fundamental Theorem #2

- The # of real numbers is much larger than the # of integers. (WHY?)
- Therefore, the # of decision problems is much larger than the # of integers, which is no less than the # of computer programs. (WHY?)
- The # of unsolvable decision problems is much larger than the # of solvable decision problems. (WHY?)

Examples of **Unsolvable** Problems

Input: a string P of characters

Output: Is P a Java program which compiles and always terminates on every input?

Question to Think about:

Can this problem be solved by any computer program at all?

Answer:

No!

Examples of **Unsolvable** Problems

Input: two strings P and w of characters

Output: Is P a Java program which compiles and terminates on w as input?

Question to Think about:

Can this problem be solved by any computer program at all?

Answer:

No!

How to Measure Efficiency?

- 1. Specify a resource R .**
- 2. Ask how much of R is used to solve a problem.**
- 3. Specify the input size.**
- 4. Ask how much of R is used to solve a problem for an input of size n .**

Examples of Efficiencies and Resources

Time:

1. How many **seconds** does it take for a computer program P to solve a problem Q ?
2. How many **operations** does it take to solve a problem?
3. How many **comparisons** does a computer program P use to sort a list of n numbers?
4. How many **bit-level operations** does it take to multiply two integers?

Examples of Resources and Efficiencies

Memory Space:

1. How much memory space in the hard disk drive does it take for a program P to solve a problem Q ?

Examples of Efficiencies and Resources

Transistor Count:

1. How many **transistors** are there in a computer chip?

Examples of Efficiencies and Resources

Energy:

1. How many **watts** does a PC consume?

Examples of Efficiencies and Resources

Area:

1. What is the **area of a computer chip?**

Examples of Efficiencies and Resources

Length:

- 1. What is the total wire length of a computer chip?**

Examples of Efficiencies and Resources

Random bits:

1. How many **random bits** are needed to solve a problem?

Examples of Efficiencies and Resources

What else?

More ...

What do you think?

How to Measure Efficiency?

1. Specify a resource R .
2. Ask how much of R is used to solve a problem.
3. Specify the input size.
4. Ask how much of R is used to solve a problem for an input of size n .

Input Size

Count the # of objects in an input.

Examples:

- 1. Count the # of integers in a list.**
- 2. Count the # of bits needed to represent a list of integers.**
- 3. Count the number of words in a text.**
- 4. Count the # of characters in a text.**

Examples of Algorithmic Problems

Input: two Integers J and K .

Output: the numbers $J + K$, $J - K$, $J * K$, and J / K .

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Examples of Algorithmic Problems

Input: two Integers J and K.

Output: the number $J^2 + 2K$.

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Examples of Algorithmic Problems

Input: a number b .

Output: the number $3b^4 + 2b^3 - 2b^2 + b - 10$.

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Examples of Algorithmic Problems

Input: an equation $ax^2 + bx + c = 0$.

Output: the solutions to the equation.

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Examples of Algorithmic Problems

Input: a positive integer K .

Output: the sum of all the integers from 1 to K .

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Examples of **Decision** Problems

Input: a positive integer K

Output: 'YES' if K is a prime and 'NO' if K is not a prime.

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Examples of Algorithmic Problems

Input: a list L of words in English.

Output: the list L sorted in alphabetic order.

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Examples of Algorithmic Problems

Input: a list L of numbers.

Output: the list L sorted in increasing order.

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Examples of Algorithmic Problems

Input: two texts in English.

Output: a list of words that appear in both texts.

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Examples of **Search** Problems

Input: a road map for cities, with distance attached to road segments.

Output: a trip that starts from Chicago, passes through all cities exactly once, and returns to Chicago.

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Examples of **Optimization** Problems

Input: a road map for cities, with distance attached to road segments.

Output: a trip that starts from Chicago, passes through all cities, returns to Chicago, and has the smallest length among all such trips.

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Examples of **Decision** Problems

Input: a road map for cities, with distance attached to road segments, and a number K .

Output: 'YES' if it is possible to take a trip that starts from Chicago, passes through all cities, returns to Chicago, and has a total length no greater than K miles; and 'NO' if such a trip is impossible.

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Examples of **Decision** Problems

Input: a string P of characters

Output: Is P a Java program which compiles?

Questions:

1. What is the input size?
2. How fast can this problem be solved?

Next Time ...

- **NP versus P:** This problem is concerned with efficiency issues. It is probably the most famous open problem in Computer Science.
- **Turing machines:** a mathematical model of computers , computer programs, or computation.
- **the Church-Turing Thesis:** All past, current, and future general-purpose “computers” (including the Turing machines) can solve the same problems.
(Insightful? Or visionless?)