Complexity Theory — Part 1: Motivation

Dusko Pavlovic

RHUL
Spring 2012

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms

Outline

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms

Outline

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms

Contact

- Dusko Pavlovic
- email: dusko.pavlovic@rhul.ac.uk
- phone: (01784 44 30 81)
- office: 227
- office hours: W 15-17, Θ 16-17, email appointment

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms

Books

- Sanjeev Arora and Boaz Barak,
  Computational Complexity.
  Cambridge University Press (2009)
- Christos Papadimitriou,
  Computational Complexity.
  Addison-Wesley (1994)
- Steven Rudich and Avi Vidgerson (eds.),
  Computational Complexity Theory.
  American Mathematical Society (2000)

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms

Course

- What do we expect from the course?

Course

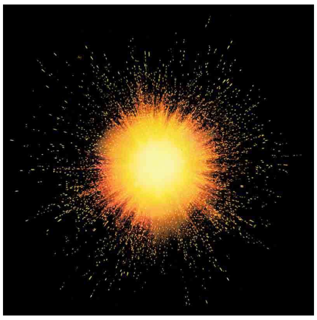
- What do we expect from the course?
- Why are we interested in complexity?

Course

- What do we expect from the course?
- Why are we interested in complexity?
- What is complexity?

Outline

Why are there complex phenomena?



If the initial conditions are simple. . .

Why are there complex phenomena?

. . . where does the complexity come from?

Why are there complex phenomena?

Why does complexity increase and spread?

Why are there complex phenomena?

Why are we so complex?

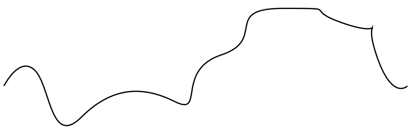
What is complexity?

Complexity is easily recognized

A straight line is simple.

What is complexity?

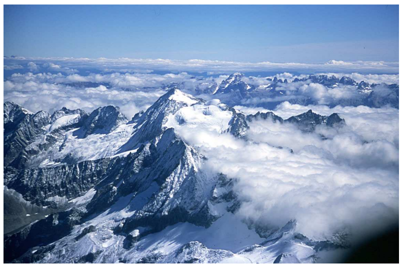
Complexity is easily recognized



A curved line is less simple.

What is complexity?

Complexity is easily recognized



A line of the horizon is complex.

What is complexity?

Which strings are complex?

- $\underbrace{0101010101010101010101010101010101\cdots01}_{100 \text{ digits}}$
- $\underbrace{010011000111000011110000011111 0\cdots1101}_{100 \text{ digits}}$
- $\underbrace{110100010011010100101011101010 0\cdots1101}_{100 \text{ digits}}$

What is complexity?

How do you know which strings are complex?

- $\underbrace{0101010101010101010101010101010101\cdots01}_{100 \text{ digits}}$
- $\underbrace{010011000111000011110000011111 0\cdots1101}_{100 \text{ digits}}$
- $\underbrace{110100010011010100101011101010 0\cdots1101}_{100 \text{ digits}}$

What is complexity?

- $\underbrace{0101010101010101010\cdots01}_{100}$ can be written as
  - $(01)^{50}$

What is complexity?

- $\underbrace{0101010101010101010\cdots01}_{100}$ can be written as
  - $(01)^{50}$
  - do i=1..50 write 01 od

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Why is there complexity?
What is complexity?
Algorithms

What is complexity?

- $\underbrace{0101010101010101010\cdots01}_{100}$ can be written as
  - $(01)^{50}$
  - do i=1..50 write 01 od
- $\underbrace{0100110001110000111 10\cdots11}_{100}$ can be written as
  - $\underbrace{0^1 1^1 0^2 1^2 \cdots 0^i 1^i \cdots}_{100}$
  - i=1; do until length=100 write $0^i 1^i$; i = i+1 od

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Why is there complexity?
What is complexity?
Algorithms

What is complexity?

- $\underbrace{0101010101010101010\cdots01}_{100}$ can be written as
  - $(01)^{50}$
  - do i=1..50 write 01 od
- $\underbrace{0100110001110000111 10\cdots11}_{100}$ can be written as
  - $\underbrace{0^1 1^1 0^2 1^2 \cdots 0^i 1^i \cdots}_{100}$
  - i=1; do until length=100 write $0^i 1^i$; i = i+1 od
- $\underbrace{110100010011010100101\cdots00}_{100}$ can be written as
  - print $110100010011010100101\cdots00$

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Why is there complexity?
What is complexity?
Algorithms

What is complexity?

Idea

Fix a programming language and define

$$\mathcal{CX}(x) \quad = \quad \text{length of the shortest program that outputs } x$$

where x any bitstring.

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Why is there complexity?
What is complexity?
Algorithms

What is complexity?

$$\mathcal{CX}(0101010\ldots) < \mathcal{CX}(010011000\ldots) < \mathcal{CX}(110100010\ldots)$$

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Why is there complexity?
What is complexity?
Algorithms

What is complexity?

Conclusion

- Complexity is relative to our capability
  - to model, predict, *compute*...

What is complexity?

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Why is there complexity?
What is complexity?
Algorithms

Conclusion

- Complexity is relative to our capability
  - to model, predict, *compute*...
- Complexity is studied as *computational* complexity

Outline

Example 1: REACH

Reachability Problem

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Task

Given a directed graph Γ and the nodes a, b in it, determine whether there is a path $a \to \cdots \to b$.

Example 1: REACH

Reachability Problem

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Formalize directed graphs

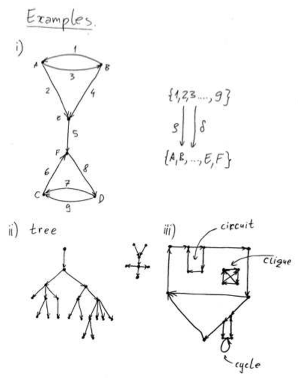
A directed graph is a pair of finite sets and a pair of functions between them:

$$E \overset{\delta}{\underset{\varrho}{\rightrightarrows}} V$$

Example 1: REACH

Reachability Problem

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary



Example 1: REACH

Reachability Problem

Algorithm

Set

$$R_i = \left\{ v \in V \mid \exists j \leq i.\ a \xrightarrow{1}\xrightarrow{2} \cdots \xrightarrow{j} v \right\}$$

$$S_i = \left\{ v \in V \mid a \xrightarrow{1}\xrightarrow{2} \cdots \xrightarrow{i} v \ \wedge \text{ no shorter path} \right\}$$

Example 1: REACH

Reachability Problem

Algorithm

Set

$$R_i = \left\{ v \in V \mid \exists j \leq i.\ a \xrightarrow{1}\xrightarrow{2} \cdots \xrightarrow{j} v \right\}$$

$$S_i = \left\{ v \in V \mid a \xrightarrow{1}\xrightarrow{2} \cdots \xrightarrow{i} v \ \wedge \text{ no shorter path} \right\}$$

Compute

$$S_1 = \varrho\delta^{-1}\{a\} \qquad R_1 = S_1$$

$$S_{i+1} = \varrho\delta^{-1} S_i \setminus R_i \qquad R_{i+1} = R_i \cup S_{i+1}$$

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Example 1: REACH

Reachability Problem

Algorithm

Set

$$R_i = \left\{ v \in V \mid \exists j \leq i.\ a \xrightarrow{1}\xrightarrow{2} \cdots \xrightarrow{j} v \right\}$$

$$S_i = \left\{ v \in V \mid a \xrightarrow{1}\xrightarrow{2} \cdots \xrightarrow{i} v \ \wedge \text{ no shorter path} \right\}$$

Compute

$$S_1 = \varrho\delta^{-1}\{a\} \qquad R_1 = S_1$$

$$S_{i+1} = \varrho\delta^{-1} S_i \setminus R_i \qquad R_{i+1} = R_i \cup S_{i+1}$$

At each step check if $b \in S_i$.

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Example 1: REACH

Reachability Problem

Analysis

- The algorithm spans the maximal subtree rooted in a.

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Example 1: REACH

Reachability Problem

Analysis

- The algorithm spans the maximal subtree rooted in a.
- A tree with n vertices has $n-1$ edges.

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Example 1: REACH

Reachability Problem

Analysis

- The algorithm spans the maximal subtree rooted in a.
- A tree with n vertices has $n-1$ edges.
- If the graph has n vertices, then
  - at each step at most $n-1$ edges may be tested;
  - the search must halt after at most $n-1$ steps.

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Example 1: REACH
Reachability Problem

Complexity

$$\#operations \;=\; \#tests \times \#steps \;\leq\; (n-1)^2 = O(n^2)$$

Digression: $\mathcal{O}$-notation

Definition 1

For $f, g : \mathbb{N} \to \mathbb{N}$ write $f(n) \leq \mathcal{O}(g(n))$ whenever

$$\exists c \exists n_0 \forall n.\ n_0 \leq n \Longrightarrow f(n) \leq c \cdot g(n)$$

Digression: $\mathcal{O}$-notation

Definition 1

For $f, g : \mathbb{N} \to \mathbb{N}$ write $f(n) \leq \mathcal{O}(g(n))$ whenever

$$\exists c \exists n_0 \forall n.\ n_0 \leq n \Longrightarrow f(n) \leq c \cdot g(n)$$

Examples

- $(n-1)^2 = O(n^2)$
- $n^2 = O\big((n-1)^2\big)$
  - $5 \leq n \Longrightarrow n^2 \leq 2(n-1)^2$

Example 1: REACH
Reachability Problem

Complexity

$$\mathcal{CX}(REACH) \;\leq\; O(n^2)$$

Example 2: TSP
Travelling Salesman Problem

Task

Given a road map, find the shortest tour through all cities.

Example 2: TSP
Travelling Salesman Problem

Formalize road maps

A road map is a directed $\mathbb{N}$-labelled graph

$$\mathbb{N} \xleftarrow{\lambda} E \underset{\varrho}{\overset{\delta}{\rightrightarrows}} V$$

where the labels $\lambda(e)$ represent distances.

Example 2: TSP
Travelling Salesman Problem

Algorithm

List all possible tours.

Example 2: TSP
Travelling Salesman Problem

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Analysis

- If there is a road between every two cities, then there are $n!$ tours between n cities.
- If all distances are different, then all $n!$ tours must be computed and compared.

Example 2: TSP
Travelling Salesman Problem

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Complexity

$$\mathcal{CX}(TSP) \quad \leq \quad \mathcal{O}(n!)$$

Example 2: TSP
Travelling Salesman Problem

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Remark.

- We don't know a better way to do this with a computer.
- Ants do it in $\mathcal{O}(n)$ operations, using pheromones and parallel search.

Example 2: TSP
Travelling Salesman Problem

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Remark.

- We don't know a better way to do this with a computer.
- Ants do it in $\mathcal{O}(n)$ operations, using pheromones and parallel search.
- Computational complexity is relative to computational resources.

Example 3: HALT
Halting Problem

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Fact

- Some programs always halt

Example 3: HALT

Halting Problem

Fact

- Some programs always halt:
  - (i) $m(x,y) = x * y$
  - (ii) $s(x) = m(x,x)$
  - (iii) $t(x) = s(x) + 1$

Example 3: HALT

Halting Problem

Fact

- Some programs always halt:
  - (i) $m(x,y) = x * y$
  - (ii) $s(x) = m(x,x)$
  - (iii) $t(x) = s(x) + 1$
- Some programs do not always halt:
  - (1) $u(x,y) = x(y)$
  - (2) $v(x) = u(x,x)$
  - (3) $w(x) = v(x) + 1$

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Example 3: HALT

Halting Problem

Fact

- Some programs always halt:
  - (i) $m(x,y) = x * y$
  - (ii) $s(x) = m(x,x)$
  - (iii) $t(x) = s(x) + 1$
- Some programs do not always halt:
  - (1) $u(x,y) = x(y)$
  - (2) $v(x) = u(x,x)$
  - (3) $w(x) = v(x) + 1$

  because

$$v(w) \stackrel{(2)}{=} u(w,w) \stackrel{(1)}{=} w(w) \stackrel{(3)}{=} v(w) + 1$$

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Example 3: HALT

Halting Problem

Task

Determine which programs halt for which inputs.

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Example 3: HALT

Halting Problem

Claim

There is no algorithm solving HALT.

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Example 3: HALT

Halting Problem

Proof.

Suppose that HALT has a solution, and set

(1) $h(x,y) \iff \exists z.\ x(y) = z$

(2) $k(x) =$ "[B] if h(x,x) then go to B else 0"

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Example 3: HALT

Halting Problem

Proof.

Suppose that HALT has a solution, and set

$$(1)\quad h(x,y) \iff \exists z.\ x(y) = z$$
$$(2)\quad k(x) = \text{"[B] if h(x,x) then go to B else 0"}$$

Then for any x holds

$$h(k,x) \overset{(1)}{\iff} \exists z.k(x) = z \overset{(2)}{\iff} \neg h(x,x)$$

Example 3: HALT

Halting Problem

Proof.

Suppose that HALT has a solution, and set

$$(1)\quad h(x,y) \iff \exists z.\ x(y) = z$$
$$(2)\quad k(x) = \text{"[B] if h(x,x) then go to B else 0"}$$

Then for any x holds

$$h(k,x) \overset{(1)}{\iff} \exists z.k(x) = z \overset{(2)}{\iff} \neg h(x,x)$$

Hence for $x = k$ holds

$$h(k,k) \iff \neg h(k,k)$$

□

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Example 3: HALT

Halting Problem

Complexity

$$\mathcal{CX}(HALT) = \infty$$

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Summary

There are problems of different complexities

$$\begin{aligned} \mathcal{CX}(REACH) &\leq O(n^2) \\ \mathcal{CX}(TSP) &\leq O(n!) \\ \mathcal{CX}(HALT) &= \infty \end{aligned}$$

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary

Summary

There are problems of different complexities

$$\begin{aligned} \mathcal{CX}(REACH) &\leq O(n^2) \\ \mathcal{CX}(TSP) &\leq O(n!) \\ \mathcal{CX}(HALT) &= \infty \end{aligned}$$

Can we tell more?

Complexity 1: Motivation
Dusko Pavlovic
Introductions
What?
Algorithms
REACH
TSP
HALT
Summary