Complexity Theory

Lecture 2: Complexity classes – The Class P

Tom Gur

Active learning

The story so far

- Goal: understand the complexity of computational problems.
- Decidability is necessary, but not enough!
- Upper bound: show one algorithm.
- Lower bounds: argue about all algorithms.
- Towards that, we abstract the notion of an algorithm
- Extended Church-Turing Thesis: the model doesn't matter (perhaps, unless it's quantum...)
- We will use Turing Machines, as they are relatively simple.

Out next goal: characterise efficient computation!

Complexity Classes

We will study the landscape of computational power by group problems into complexity classes.

A complexity class is a collection of languages determined by three things:

- A *model of computation* (such as a deterministic Turing machine, or a nondeterministic TM, or a parallel Random Access Machine).
- A *resource* (such as time, space or number of processors).
- A *set of bounds*. This is a set of functions that are used to bound the amount of resource we can use.

How shall we model efficient computation?

The Big Idea:
Efficient = Polynomial Time

Polynomial Time

$$\mathsf{P} = \bigcup_{k=1}^{\infty} \mathrm{TIME}(n^k)$$

The class of languages decidable in polynomial time.

The complexity class P plays an important role in our theory.

- Concrete enough to rule out unphysical (exponential) complexity.
- Abstract enough to be robust (Extended Church Turing Thesis).
- Group structure: captures sub-procedures.
- It serves as our formal definition of what is *feasibly computable*

However, it is not perfect: Is runtime $\theta(n^{100})$ feasible?

The distinction between polynomial and exponential leads to a useful and elegant theory.

Example 1: Reachability

The Reachability decision problem is, given a *directed* graph $G = (V, E)$ and two nodes $a, b \in V$, to determine whether there is a path from a to b in G.

A simple search algorithm as follows solves it:

1. mark node a, leaving other nodes unmarked, and initialise set S to $\{a\}$;
2. while S is not empty, choose node i in S: remove i from S and for all j such that there is an edge (i, j) and j is unmarked, mark j and add j to S;
3. if b is marked, accept else reject.

What are the time and space complexities?

Analysis

This algorithm requires $O(n^2)$ time and $O(n)$ space.

The description of the algorithm would have to be refined for an implementation on a Turing machine, but it is easy enough to show that:

$$\text{Reachability} \in \mathrm{P}$$

To formally define Reachability as a language, we would have to also choose a way of representing the input (V, E, a, b) as a string.

Example 2: Euclid's Algorithm

Consider the decision problem (or *language*) RelPrime defined by:

$$\{(x, y) \mid \gcd(x, y) = 1\}$$

What is the naive algorithm? Complexity? is it in P?

Example 2: Euclid's Algorithm

Consider the decision problem (or *language*) RelPrime defined by:

$$\{(x, y) \mid \gcd(x, y) = 1\}$$

The standard algorithm for solving it is due to Euclid:

1. Input (x, y).
2. Repeat until $y = 0$: $x \leftarrow x \bmod y$; Swap x and y
3. If $x = 1$ then accept else reject.

Analysis

The number of repetitions at step 2 of the algorithm is at most $O(\log x)$.

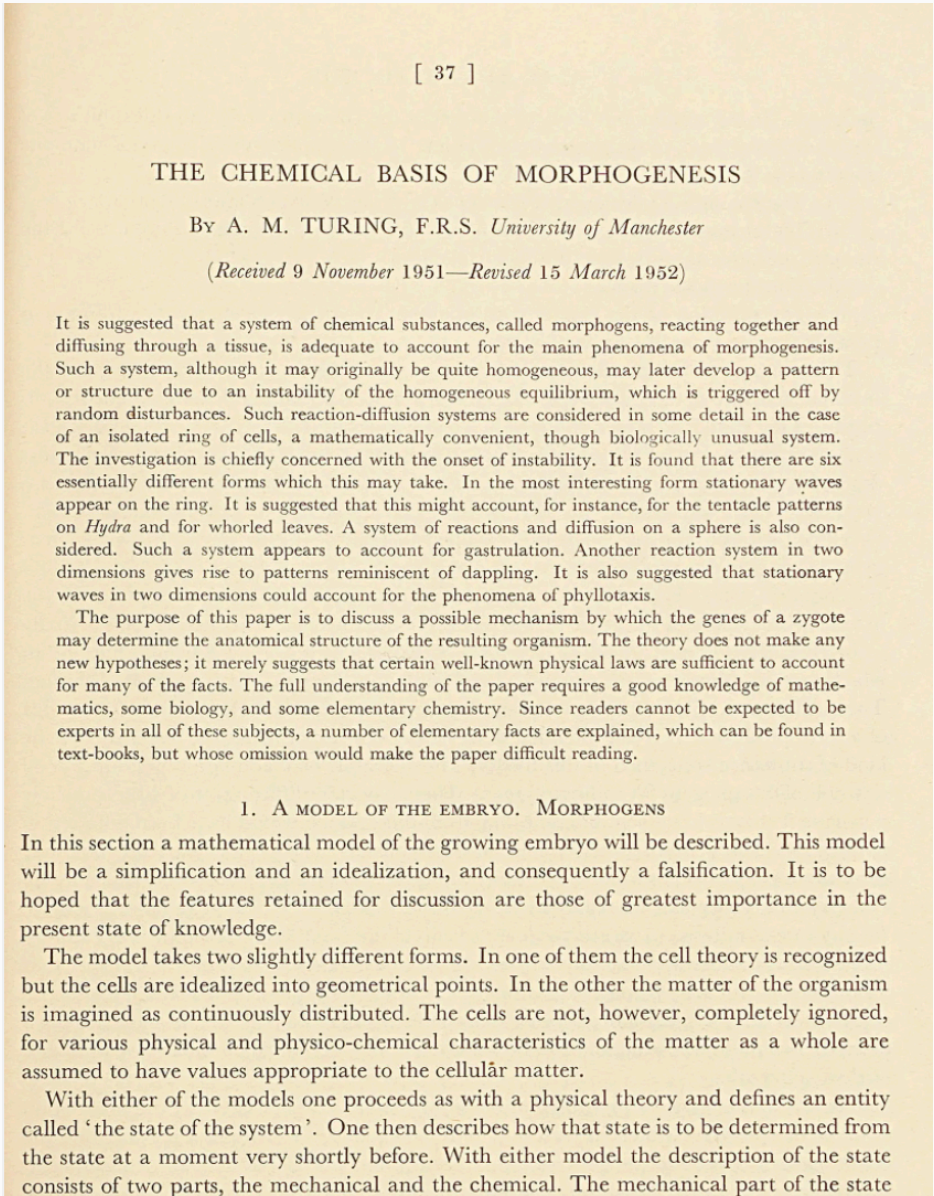
why?

This implies that RelPrime is in P.

If the algorithm took $\theta(x)$ steps to terminate, it would not be a polynomial time algorithm, as x is not polynomial in the *length* of the input.

Turing fact of the day

Turing had authored one of the most cited papers in *biology*.



[37]

THE CHEMICAL BASIS OF MORPHOGENESIS

By A. M. TURING, F.R.S. *University of Manchester*

(*Received* 9 *November* 1951—*Revised* 15 *March* 1952)

It is suggested that a system of chemical substances, called morphogens, reacting together and diffusing through a tissue, is adequate to account for the main phenomena of morphogenesis. Such a system, although it may originally be quite homogeneous, may later develop a pattern or structure due to an instability of the homogeneous equilibrium, which is triggered off by random disturbances. Such reaction-diffusion systems are considered in some detail in the case of an isolated ring of cells, a mathematically convenient, though biologically unusual system. The investigation is chiefly concerned with the onset of instability. It is found that there are six essentially different forms which this may take. In the most interesting form stationary waves appear on the ring. It is suggested that this might account, for instance, for the tentacle patterns on *Hydra* and for whorled leaves. A system of reactions and diffusion on a sphere is also considered. Such a system appears to account for gastrulation. Another reaction system in two dimensions gives rise to patterns reminiscent of dappling. It is also suggested that stationary waves in two dimensions could account for the phenomena of phyllotaxis.

The purpose of this paper is to discuss a possible mechanism by which the genes of a zygote may determine the anatomical structure of the resulting organism. The theory does not make any new hypotheses; it merely suggests that certain well-known physical laws are sufficient to account for many of the facts. The full understanding of the paper requires a good knowledge of mathematics, some biology, and some elementary chemistry. Since readers cannot be expected to be experts in all of these subjects, a number of elementary facts are explained, which can be found in text-books, but whose omission would make the paper difficult reading.

1. A model of the embryo. Morphogens

In this section a mathematical model of the growing embryo will be described. This model will be a simplification and an idealization, and consequently a falsification. It is to be hoped that the features retained for discussion are those of greatest importance in the present state of knowledge.

The model takes two slightly different forms. In one of them the cell theory is recognized but the cells are idealized into geometrical points. In the other the matter of the organism is imagined as continuously distributed. The cells are not, however, completely ignored, for various physical and physico-chemical characteristics of the matter as a whole are assumed to have values appropriate to the cellular matter.

With either of the models one proceeds as with a physical theory and defines an entity called 'the state of the system'. One then describes how that state is to be determined from the state at a moment very shortly before. With either model the description of the state consists of two parts, the mechanical and the chemical. The mechanical part of the state

Example 3: Primality

Consider the decision problem (or *language*) Prime defined by:

$$\{x \mid x \text{ is prime}\}$$

The obvious algorithm:

For all y with $1 < y \leq \sqrt{x}$ check whether $y|x$.

requires $\Omega(\sqrt{x})$ steps and is therefore *not* polynomial in the length of the input.

Is Prime $\in$ P?

Example 4: Boolean Formula Evaluation

Boolean expressions are built up from an infinite set of variables

$$X = \{x_1, x_2, \ldots\}$$

and the two constants `true` and `false` by the rules:

- a constant or variable by itself is an expression;
- if ϕ is a Boolean expression, then so is $(\neg\phi)$;
- if ϕ and ψ are both Boolean expressions, then so are $(\phi \wedge \psi)$ and $(\phi \vee \psi)$.

Evaluation

If an expression contains no variables, then it can be evaluated to either `true` or `false`.

Otherwise, it can be evaluated, *given* a truth assignment to its variables.

Examples:

$(\texttt{true} \vee \texttt{false}) \wedge (\neg \texttt{false})$

$(x_1 \vee \texttt{false}) \wedge ((\neg x_1) \vee x_2)$

$(x_1 \vee \texttt{false}) \wedge (\neg x_1)$

$(x_1 \vee (\neg x_1)) \wedge \texttt{true}$

Boolean Evaluation

There is a deterministic Turing machine, which given a Boolean expression *without variables* of length n will determine, in time $O(n^2)$ whether the expression evaluates to `true`.

The algorithm works by scanning the input, rewriting formulas according to the following rules:

Rules

- $(\texttt{true} \vee \phi) \Rightarrow \texttt{true}$
- $(\phi \vee \texttt{true}) \Rightarrow \texttt{true}$
- $(\texttt{false} \vee \phi) \Rightarrow \phi$
- $(\phi \vee \texttt{false}) \Rightarrow \phi$
- $(\texttt{false} \wedge \phi) \Rightarrow \texttt{false}$
- $(\phi \wedge \texttt{false}) \Rightarrow \texttt{false}$
- $(\texttt{true} \wedge \phi) \Rightarrow \phi$
- $(\phi \wedge \texttt{true}) \Rightarrow \phi$
- $(\neg\texttt{true}) \Rightarrow \texttt{false}$
- $(\neg\texttt{false}) \Rightarrow \texttt{true}$

Analysis

Each scan of the input ($O(n)$ steps) must find at least one subexpression matching one of the rule patterns.

Applying a rule always eliminates at least one symbol from the formula.

Thus, there are at most $O(n)$ scans required.

The algorithm works in $O(n^2)$ steps.

Last Problem: Satisfiability

For Boolean expressions ϕ that contain variables, we can ask

> *Is there an assignment of truth values to the variables which would make the formula evaluate to* `true`*?*

The set of Boolean expressions for which this is true is the language SAT of *satisfiable* expressions.

This can be decided by a deterministic Turing machine in time $O(n^2 2^n)$.

An expression of length n can contain at most n variables.

For each of the 2^n possible truth assignments to these variables, we check whether it results in a Boolean expression that evaluates to `true`.

Is SAT $\in$ P?

Questions?