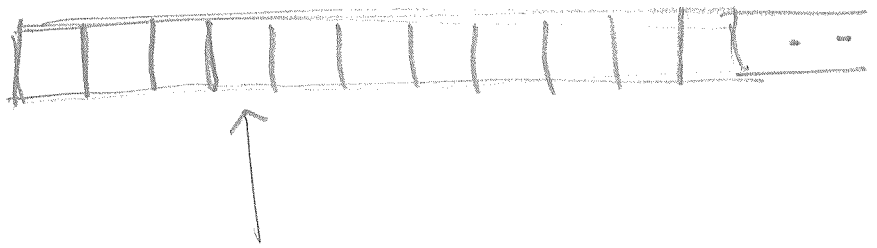
4.6



Definition

A problem is called a decision problem if its answer is "Yes" or "No".

In the study of computational complexity, an optimization problem is usually formulated as a decision problem.

The decision version of the prototype GT problem: Given n items and two integers d and k ($d>0$ and $k<n$), determine whether $M(d,n) \geq k$ or not.

P: A decision problem belongs to P if it can be computed by a polynomial time deterministic TM.

(non-deterministic) TM

NP: ↗

Deterministic TM:

The set of rules prescribes at most one action to be performed for any given situation.

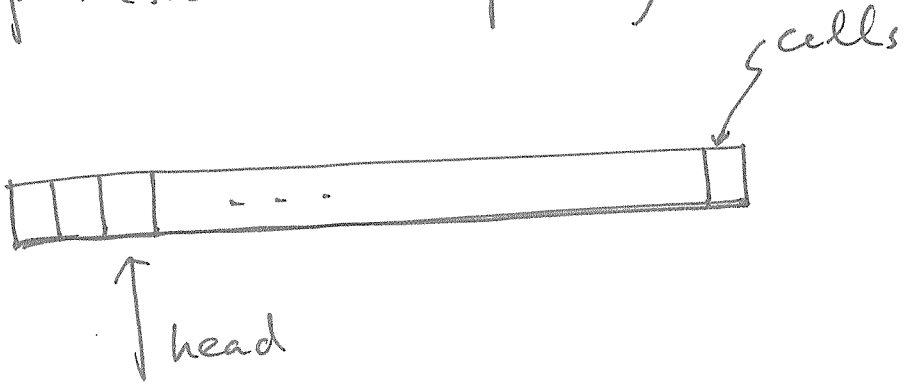
下一步驟唯一確定

Non-deterministic TM:

The set of rules that prescribes more than one action for a given situation.

下一步驟有多种選擇

Computational Complexity



Time and Space

- The number of moves of the Turing Machine (TM)
- The number of cells on the tape which have been visited by the head during the computation

PSPACE

A decision problem belongs to PSPACE if it can be computed by a polynomial-space deterministic (or non-deterministic) TM.

So, $P \subseteq NP \subseteq PSPACE$.

等号成立与否是目前未知的难题。

Definitions

① A decision problem A is poly.-time many-one reducible of a decision problem B, denoted by $A \leq_m^p B$, if there is a poly.-time computable mapping f from instances of A to instances of B such that A has the Yes-answer on x iff B has the Yes-answer on $f(x)$.

② For a complexity class C, a decision problem A is C-complete if

(a) A is in C and

(b) For any decision problem $B \in C$, $B \leq_m^p A$.

C : NP

(*) If B is NP-complete and $B \leq_m^p A$, then A is NP-complete

---

③ A decision problem A is co-NP-complete if the complement of A is NP-complete.

↓

④ A decision problem B is the complement of another decision problem A if they always obtain different answers on the same input. ⑤

最常見的 NP-complete problem：

The decision version of vertex-cover (Vertex-Cover).

Given a graph $G=(V,E)$ and an integer $k \le |V|$, determine whether there is a set $V' \subseteq V$ of size k such that each edge $e \in E$ is incident to at least one vertex $v \in V'$.

(*) {6.} NP-complete 的問題都可以轉換這個問題的 Reduction。

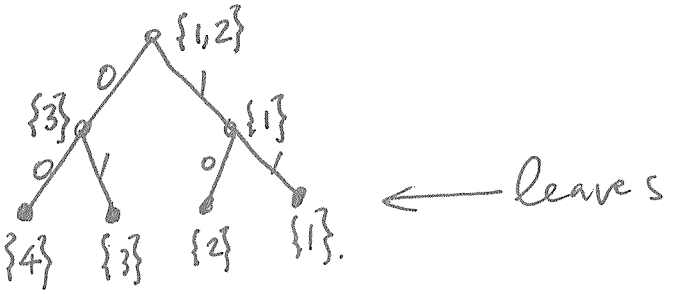
Combinatorial Search Problem (CSP)

Given a domain D and an integer k, determine whether there is a decision tree of height "$\le k$" of which each path uniquely determines an object in D.

Fact

Here, each internal node of the decision tree corresponds to a Yes-No query and the left and the right sons of the internal node are the queries following two answers respectively unless they are leaves.

1 is positive, 2 is positive, ..., 4 is positive 都可能



Theorem CSP $\in$ PSPACE.

Proof:

The algorithm needs only $O(k)$ space to store one path of the decision tree if D (domain) and k is given. (The tree may have 2^k nodes and 2^{k-1} leaves.).

(*) The prototype groups testing problem is a special case of the CSP.

GPP (Generalized Prototype GT Problem)

Given n items, a test history, and two integers $d>0$, $0<k<n$, determine whether or not there exists a decision tree of height $\leq k$ of which each path uniquely determines a sample point in $S(d,n)$ where $S(d,n)$ consists of all sample points each consisting exactly d defectives from n items.

GPP $\in$ PSPACE.

Is GPP $\in$ PSPACE-complete ? (Conjecture)

Learning by Examples

The problem that has been studied in this direction can be described as follows:

Suppose there is an unknown boolean function which belongs to a certain family. When values are assigned to the variables, the function value is returned.

(*) The problem is how to identify the unknown function by using a small number of assignments.

(**) The (典型) prototype group testing problem can be seen as a special case of this problem.

⇓

Let $\mathcal{F} = \left\{ x_{i_1} + \cdots + x_{i_d} \;\middle|\; 1 \le i_1 \le i_2 \le \cdots \le i_d \le n \right\}$.

$$\boxed{f(x) = x_{i_1} + x_{i_2} + \cdots + x_{i_d}}$$ where $i_1, \cdots, i_d$ are defectives

~~Recall~~ | $T(a)$ is contaminated (a is an assignment: $\{j_1, j_2, \cdots, j_t\}$)

$f(a) = 1 \leftrightarrow T(a) = 1$ iff $\{i_1, i_2, \cdots, i_d\} \cap \{j_1, \cdots, j_t\} \neq \emptyset$

Otherwise

$f(a) = 0 \leftrightarrow T(a) = 0$

if $f(x) = ?$

(d, n) prototype GT problem

In learning theory, if the unknown function can be learned in polynomially many assignments w.r.t. n, then this problem is considered to be an "Easy problem".

Since (d, n) Prototype GT problem can be solved with at most $d(\log_2 \frac{n}{d} + 1)$ tests, it is an easy one from the viewpoint of learning.

But, if optimality is concerned, then it is a "hard problem".