On Measurement and Computation

by

ZHENG HUIMIN

supervised by Prof. Song Fangmin

A dissertation submitted to
the Graduate School of Nanjing University
for the degree of Master of Engineering

Department of Computer Science & Technology
Nanjing University

May 2012

南京大學

研 究 生 毕 业 论 文
（申请硕士学位）

论文题目 ： 论测量与计算
作　　者 ： 郑惠民
院　　系 ： 计算机科学与技术
专　　业 ： 计算机软件与理论
研究方向 ： 量子计算、数理逻辑
指导老师 ： 宋方敏 教授

二〇一二年五月

Abstract

Inspired by the work of R.Feynman, D.Deutsch, We formally propose the theory of physical computation and accordingly, the physical complexity theory. To achieve this, a framework that could be used to evaluate almost all forms of computation making use of various physical mechanisms is established. Here, we focus on applying this framework to Quantum Computation. As a preliminary study on more general problems, some examples of other physical mechanisms are also discussed in this paper.

Key Words: Quantum Computation, Physical Computation, computational complexity

摘要

受 R.Feynman，D.Deutsch 等人工作的启示，我们形式地建立了物理可计算理论以及相应的物理复杂度理论。文章中建立了一个评估框架，它可以用来评估几乎所有利用物理机制进行的计算。这里我们特别关注了如何将该框架应用在量子计算中。作为对更一般问题的初步探索，一些利用物理机制的其他算例也在本文中进行了详细的论述。

关键词： 量子计算，物理计算，计算复杂度

To My Parents.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Quantum Computation

The research of quantum computation has been lasting for about 30 years since R.Feynman proposed the concept of so-called 'quantum computer' in 1982[2]. Founding out that there do exist some quantum systems which are suspected cannot be efficiently simulated by classical computers, early researchers naturally speculated that quantum mechanism itself may provide stunning power of computation. In order to strictly define what is 'quantum computation', researchers introduced various new computational models, including Quantum Turing machine [1] and Quantum Circuit Model[5]. However, at that time, no convincing evidence was discovered to support the conjecture that quantum mechanism can really be used to speed up the computation of some hard problems greatly.

D.Deutsch found the first evidence that quantum computers may surpass the Turing machine[9] in query. In fact, he constructed a special scene in which any DTM(deterministic Turing machine)has to query the oracle for $O(2^n)$ times to find the correct answer(for certain) in worst cases while QTM(quantum Turing machine) need just once query in all cases.

One of the most remarkable results is quantum factorization, which is due to Peter Shor[10, 11]. The best classical algorithm for factorization so far has to run for $O(\exp(n^{1/3}\log^{2/3} n))$ steps. However, Shor showed that one can use a family of quantum circuit, which contain $O(\log^3 L)$ gates and needs only $O(L^2 \log L \log\log L)$ operations(where $L \equiv \lceil \log(N) \rceil$) to get the right answer.

Grover's Algorithm [13] is another successful example of quantum algorithms. This algorithm can be used to search a database without structure. It is easy to prove that the time complexity of this problem with respect to Turing machine is $O(n)$. However, there does exist a quantum algorithm whose time complexity is $O(\sqrt{n})$. Since it has been showed that this is the optimal algorithm for all algorithms that considering quantum mechanics [18] , so the complexity of Grover's algorithm can be looked as the quantum complexity of this problem.

One of the most important reason that why Quantum algorithms(especially

Shor's algorithm) seem so interesting to many computer scientists is that their existence indicate a huge challenge to extended Church-Turing thesis[18], which states that:

Any model of computation can be simulated on a probabilistic Turing machine with at most a polynomial increase in the number of elementary operations required.

1.2 'Physical Computation'

On the other hand, with the exciting research in quantum computation as well as other new paradigms of computations(e.g.DNA computation), the idea that we may just look physical processes as computations(not just the Turing machine)was also developed. The seeds of this idea can be traced back to Feynman[2], Deuthsch[3] and Pitowsky[7] et al.

It is not very hard to understand and appreciate this idea, for at first glance, adopting this point of view has at least three benefits:

- It covers the concept of classical algorithms naturally, for an algorithm on Turing machines(its physical implementation) can also be looked as a family of physical processes and the corresponding measurement.
- With smart-designed physical oracle, it is possible for us to solve some problems more efficient than any Turing machines.
- Being the ones which could be directly simulated, some physical methods can also enlighten us to design smart algorithms on Turing machines.

What's more, currently, it seems that we cannot exclude the possibility that there may exists a family of physical processes which can help us to calculate some problems which cannot be solved by a universal Turing machine in principle.

However, because of vagueness and extraordinary generality, the theory of so-called 'physical computation' has a significant defect yet.

- In many cases, people cannot decide how to define the resource for a 'physical algorithm'. And as a result they cannot proof or even formally conjecture whether a 'physical algorithm' is really superior to any algorithms on Turing machines.

Note that the theory of quantum computation is almost free from such defect, for researchers have completed the formal definition of the computational model for quantum computation in the early years. Roughly speaking, things tend to go wrong when:

- People adopt a design on which the physical postulates it depends is just an empirical one.
- More than one different systems of physical postulates are used.

1.3 The structure of the article

In chapter-II we shall introduce two well-known models of quantum computation and the definition of complexity respectively. And after that we will formally establish the theoretical foundation of physical computation and propose the theory of physical computability in chapter-III. In chapter-IV, we discuss some famous examples which claimed that even universal Turing machine may not be the most powerful computational model in our physical world. In the beginning of chapter-V,we try to use the theory of physical computation to reanalyze the quantum algorithms. In the end of chapter-V, we focus on the topic about how to construct problems which take advantage of quantum simulations.

Chapter 2

Models of Quantum Computation

2.1 Quantum Turing machine

Quatnum Turing machine was first introduced by Benioff[1] in 1980 and was developed by Deustch and Yao. The modern definitioin was given by Bernstein and Vazirani in 1997[4].

Definition 2.1.1. (Quantum Turing machine, Bernstein 1997) Let $\tilde{C}$ be a set of complex nmber α satisfying: For each α, there exists a polynomial time algorithm to compute the value of $Im(\alpha)$ and $Re(\alpha)$ close to 2^{-n} within the true value.

A Quantum Turing machine M is defined as the triple (Σ, Q, δ), where Σ is a finite alphabet with an identified symbol # , Q is a finite set of states with an identified initial state q_0 and final state $q_f \neq q_0$; δ, the quantum transform function $\delta : Q \times \Sigma \to \tilde{C}^{\Sigma \times Q \times \{L,R\}}$. The QTM has a two-way infinite tape of cells indexed by Z, and a single read/write tape head that moves along the tape. We define configurations initial configurations and final configurations exactly as for DTMs. Let S be the inner-product space of finite complex linear combinations of configurations of M with the Euclidian norm. We call each element $phi \in S$ a superposition of M. The QTM M defines a linear operator $U_M : S \to S$, called the time evolution operator of M as follows: If M starts in configurations c with current state p and scanned symbol σ. The after one step M will be in superposition of configurations $\psi = \sum_i \alpha_i c_i$, where each non-zero α_i corresponds to a $\delta(p, \sigma, \tau, q, d)$, and c_i is the new configuration that results from applying this transition to c. Extending this map to the entire space S through linearity gives the linear time evolution operator U_M.

Definition 2.1.2. If U_M can keep Euclidian norm, then we say M is well deformed.

Theorem 2.1.3. If QTM is in the superposition $\psi = \sum_i \alpha_i c_i$ and is observed, the probability of the observer gets the configuration c_i is $|\alpha_i|^2$, and then M is in the state $\psi' = c_i$.

Theorem 2.1.4. A QTM is well-deformed if and only if its time evolution operator is unitary.

In QTM, the number of the read/write tape head moves during a computation is the cost of time.

Theorem 2.1.5. There exists a universal QTM, which is polynomially equivalent to any QTMs.

2.2 Quantum Circuit Model

The first quantum circuit model was due to Deutsch. Then quantum circuit model was improved by Yao[5], who also proved that for any QTM, there exists a uniform family of quantum circuit which is polynomially equivalent to that QTM.

Not like QTM, quantum circuit model tends to describe an algorithm by using universal quantum gates and circuits without loops. Quantum circuit model does not need infinite many quantum gates, but finite many quantum gates which called the universal quantum gates. It has been proved that Hadamard Gate, phase gate, C-NOT Gate and $\pi/8$ Gate are universal. For any finite dimensional U operators, we can always approach it effectively by means of a universal family of circuits $\mathscr{U}$, which only consists 4 gates above, i.e.

$$\forall\varepsilon(\exists n \in \mathscr{U}), E(U, \tilde{U}_n) \equiv \max_{|\psi\rangle} \|(U - \tilde{U}_n)|\psi\rangle\| < \varepsilon$$

The scale of a quantum circuit is defined as the number of the universal gates and the depth is defined as the longest path from input to output, if the gates is looked as vertices.

Both Quantum circuit model and QTM are important models of quantum computation. But we do not know whether they are the most natural models of quantum computation or do they fully take the advantage of quantum mechanics, no matter in the theory of quantum computability and quantum complexity.

Chapter 3

The Theory of Physical Computation

3.1 Observer

Measurement is in terms of observer. Though there are many differences among people's opinions about the exact definition of human beings, we prudently assume that an observer is classical, that is, the observer will never get incompatible results during one measurement.

In this article, we will never use the terminology such like 'a observer of the observer', or in other words, by 'observer' we always mean the last one outside the whole experiment.

In order to unify various forms of results, we require that the observer only accept the symbols on a tape(just something like the one of Turing machine) and also only use this to initialize an experiment.

So we define the legal inputs and outputs as the elements in set Σ^+, where

$$\Sigma = \{\ 0, 1, *, .\ \}$$

and Σ^+ the finite string composed by elements in Σ.

The concept of observer is crucial to our theory.

3.2 Physical States

We use (usually finite) attributes which may contribute to the computations to label the physical states. In addition, though may not be actually concerned in every computation, three fundamental quantities, namely, *space*, *energy* and *mass* are always included in a state for the sake of analysis of resource and complexity.

We have:

$$\Omega \subset \{x_1\}^{A_1} \times \{x_2\}^{A_2} \times \cdots \times \{x_n\}^{A_n} \times \{m\}^{\mathfrak{M}} \times \{s\}^{\mathfrak{S}} \times \{e\}^{\mathfrak{E}}$$

Or more generally(Quantum),

$$\Omega \subset \{x_1\}^{A_1} \times \{x_2\}^{A_2} \times \cdots \times \{x_n\}^{A_n} \times \{\mathbb{C}^m\}^{\mathfrak{M}} \times \{\mathbb{C}^s\}^{\mathfrak{S}} \times \{\mathbb{C}^e\}^{\mathfrak{E}}$$

For simplicity, fundamental attributes are usually omitted, i.e.

$$\Omega \subset \{x_1\}^{A_1} \times \{x_2\}^{A_2} \times \cdots \times \{x_n\}^{A_n}$$

For a certain attribute A_i, what really matters is its type which is constrained by its dimension. Note that dimensionless quantity(e.g.friction coefficient) can also be assigned to a **null** type. When a quantity is expressed by combination of other quantities, it's dimension type should be preserved, or rather, any equations should be dimensional balanced.

For example:

$$\begin{aligned}
E^{[D:ML^2T^{-2}]} &::= m^{[D:M]} g^{[D:LT^{-2}]} h^{[D:L]} = mgh^{[D:ML^2T^{-2}]} \\
E^{[D:ML^2T^{-2}]} &::= \frac{1}{2}(m)^{[D:M]}(v^2)^{[D:L^2T^{-2}]} = \frac{1}{2}(mv^2)^{[D:ML^2T^{-2}]} \\
E^{[D:ML^2T^{-2}]} &::= (m)^{[D:M]}(c^2)^{[D:L^2T^{-2}]} = (mc^2)^{[D:ML^2T^{-2}]}
\end{aligned}$$

are all dimensional balanced.

3.3 Physical processes and the operation $\circ$

Physical process on a state space Ω is a set of physical state whose elements are labeled by moment $t(t \in [0, T], T \in \mathbb{R}^+)$.

$$P \in \{\ (T, \tilde{P}) |\ T \in \mathbb{R}^+, \tilde{P} : [\ 0, T\ ] \to \Omega\ \} \equiv \mathcal{P}$$

If two physical processes on Ω satisfies

$$(\pi_2 P_1)(\pi_1 P_1) = (\pi_2 P_2)(0)$$

we can define operation $\circ : \mathcal{P} \times \mathcal{P} \to \mathcal{P}$ i.e.

$$P_2 \circ P_1 = P_3$$

satisfies:

1. $\pi_1 P_3 = \pi_1 P_1 + \pi_1 P_2$
2. if $0 \le t \le \pi_1 P_1$,$(\pi_2 P_3)(t) = (\pi_2 P_1)(t)$
3. if $\pi_1 P_1 \le t \le \pi_1 P_1 + \pi_1 P_2$,$(\pi_2 P_3)(t) = (\pi_2 P_2)(t - \pi_1 P_1)$

For convenience, we introduce $\triangleright P \triangleleft$ as the initial state of P, and $\triangleleft P \triangleright$ the final state of P, i.e.

$$\triangleright P \triangleleft \equiv (\pi_2 P)(0), \triangleleft P \triangleright \equiv (\pi_2 P)(\pi_1 P)$$

3.4 Physical Operator and the operation of operator

Physical operator is a tuple whose first component is a state x in Ω and the second component is a physical process whose initial state is x, i.e.

$$O \subset \{(x, P) | x \in \Omega, \triangleright P \triangleleft = x\}$$

In particular, a *deterministic physical operator* O means: O is a physical operator, and

$$\text{if } O(x_1) \neq O(x_2), \text{then } x_1 \neq x_2$$

if we only care about the effect the operator do to the initial state, we can look operator as a mapping in Ω, i.e. $O : \Omega \to \Omega$.

The operation between two deterministic physical operator is defined as follows(if O_1, O_2 are productive):

$$O_3 = O_2 \circ O_1$$

which satisfies

$$\forall x(O_3(x) \equiv O_2(\triangleleft O_1(x) \triangleright) \circ O_1(x))$$

In some more general cases, it is useful to talk about non-deterministic physical operators or random physical operators. A random physical operator $\tilde{O}$ contains the tuples which has the same initial states but different physical processes, i.e.

$$\tilde{O} \subset \{(x, P) | x \in \Omega, \triangleright P \triangleleft = x\}.$$

and

$$\text{if } \tilde{O}(x_1) \neq \tilde{O}(x_2), \text{it is still possible that } x_1 = x_2$$

People cannot decide the output $\tilde{O}(x)$ just by the initial state $x \in \Omega$.

Similarly, if just care about extensionality, we can look operator as a relationship on Ω i.e.$\tilde{O} : \Omega \times \Omega$

We can also define operations between two non-deterministic operators, if some preconditions are satisfied. To do so, we first expand the definition of some symbols.

$$O(x) \equiv \{P | \triangleright P \triangleleft = x\}$$

$$O(X) \equiv \{P | \triangleright P \triangleleft \in X \subset \Omega\}$$

$$\triangleleft O(x) \triangleright \equiv \{y | y \in \Omega, \exists P \in O(x) s.t. \triangleleft P \triangleright = y\}$$

So $O_2 \circ O_1$ (if they are productive) can be defined as:

$$O_2 \circ O_1(x) \equiv \{P_2 \circ P_1 | P_1 \in O_1(x), P_2 \in O_2(\triangleleft O_1(x) \triangleright)\}$$

Note that of all the processes created by random physical operators, in the end their last states shall be exposed to outside world, or rather, the observer.

For instance, if the operator

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix},$$

usually called Hadamard gate, is not measured in the end, it will not be thought as non-deterministic or random operator.

3.5 Physical Computability

3.5.1 Deterministic Physical Computation

Definition 3.5.1. (Deterministic Physical System)Deterministic Physical System $\mathscr{P}$ is a Five-Tuple

$$\mathscr{P} \equiv (\Omega, \Sigma, \nabla, \mathcal{H}, \Delta)$$

where:

- $\Sigma = \{0, 1, *, .\}$, Σ^+ is the collection of finite string formed by elements in Σ , Ω_{Σ^+} is the the set of physical implementation of Σ^+.
- $\Omega = \{\psi_i, i \in \Lambda\}$, $\Omega \neq \Omega_{\Sigma^+}$,$\Lambda$ is an index set, Ω is a set of distinguishable physical states(labeled by their attributes).
- $\nabla : \Omega_{\Sigma^+} \to \Omega$, **initialization operator**
- $\mathcal{H} : \Omega \to \Omega$, **evolution operator**
- $\Delta : \Omega \to \Omega_{\Sigma^+}$, **Measurement operator**

Definition 3.5.2. (Partial Physical Computable Arithmetic Functions)For any partial arithmetic function $f : \mathbb{N} \to \mathbb{N}$ is said to be *partial physical computable*,if and only if there exists a Deterministic Physical System

$$\mathscr{P} \equiv (\Omega, \Sigma, \nabla, \mathcal{H}, \Delta)$$

which satisfies:

If $x \in dom(f)$, then,

$$\lhd\big((\Delta \circ \mathcal{H} \circ \nabla)^{\ulcorner} x^{\urcorner}\big)\rhd \doteq {}^{\ulcorner} f(x)^{\urcorner}$$

Similarly, we can define *Total Physically Computable Arithmetic Functions*

Definition 3.5.3. (Total Physically Computable Arithmetic Functions) For any total arithmetic function $f : \mathbb{N} \to \mathbb{N}$ is said to be *Total Physically Computable*, if and only if there exists a Deterministic Physical System $(\Omega, \Sigma, \nabla, \mathcal{H}, \Delta)$ which satisfies:

$\forall x \in \mathbb{N}$,we have:

$$\lhd\big((\Delta \circ \mathcal{H} \circ \nabla)\ulcorner x \urcorner\big)\rhd \doteq \ulcorner f(x) \urcorner$$

In order to extend the definition of physical computability to non-arithmetic functions, we should take into consideration the precision of the measurement and computation. Therefore, we need a distance function to measure the precision of two values and define the computability as the ability of computing in any desired precision.

Definition 3.5.4. (Partial Physically Computable Functions)Given a partial function $f : A \to B$,and a metric $\mathcal{D} : B \times B \to \mathbb{R}$, f is said to be partial physically computable with respect to the metric $\mathcal{D}$, if and only if for any $\epsilon > 0$ there exists $(\Omega, \Sigma, \nabla, \mathcal{H}, \Delta)$,s.t. for any $x \in A$,we have:

if $x \in dom(f)$,

$$\mathcal{D}\Big(\llcorner\lhd\big((\Delta \circ \mathcal{H} \circ \nabla)\ulcorner x \urcorner\big)\rhd\lrcorner, f(x)\Big) < \epsilon$$

Similarly, we can also define *Total Physical Computable Functions*.

Definition 3.5.5. (Total Physically Computable Functions)Given a total function $f : A \to B$,and a metric $\mathcal{D} : B \times B \to \mathbb{R}$,$f$ is said to be partial physically computable with respect to the metric $\mathcal{D}$ if and only if for any $\epsilon > 0$ there exists $(\Omega, \Sigma, \nabla, \mathcal{H}, \Delta)$, such that,

$\forall x \in A$

$$\mathcal{D}\Big(\llcorner\lhd\big((\Delta \circ \mathcal{H} \circ \nabla)\ulcorner x \urcorner\big)\rhd\lrcorner, f(x)\Big) < \epsilon$$

3.5.2 Non-deterministic Physical Computation

On the other hand, many physical processes are considered to be non-deterministic, which enable us to implement so-called 'randomized algorithms' and 'quantum algorithms'. Our *Probabilistic Physical System* is defined as follows.

Definition 3.5.6. (Probabilistic Physical System)Probabilistic Physical System $\mathscr{P}^*$ is a five-tuple:

$$\mathscr{P}^* \equiv (\Omega, \Sigma, \nabla, \mathcal{H}^*, \Delta)$$

. where,

- $\Sigma = \{0, 1, *, .\}$.
- $\Omega = \{\psi_i, i \in \Lambda\}$.
- $\nabla : \Omega_{\Sigma^+} \to \Omega$,which is also called **initialization operator**
- $\mathcal{H}^* : \Omega \times \Omega$,which also called **evolution operator**, which is non-deterministic.
- $\Delta : \Omega \to \Omega_{\Sigma^+}$, which is also called **measurement operator**.

Non-deterministic does not necessarily cause probability, but let's convention that in this article we always discussed the randomness which has a probabilistic distribution.

Definition of the computable functions by means of $\mathscr{P}^*$ is an analog to that of $\mathscr{P}$. As an example, we define Total Non-deterministic Physical Computable Functions.

Definition 3.5.7. (Total Non-deterministic Physical Computable Functions(Las Vegas)) For any total function $f : \mathbb{N} \to \mathbb{N}$ is said to be total non-deterministic physical computable function, if and only if there exists a five-tuple

$$(\Omega, \Sigma, \nabla, \mathcal{H}^*, \Delta)$$

s.t.

$$\forall x \left(\lhd\big(\Delta \circ \mathcal{H}^* \circ \nabla\big)^\ulcorner x^\urcorner\big)\rhd \doteq {}^\ulcorner f(x)^\urcorner\right)$$

Because of randomness, for any identical inputs x, the system may call different process to compute. The above definition is the counterpart of the definition of the so called Las Vegas algorithm.

Definition 3.5.8. (Total Non-deterministic Physical Computable Functions(Monte Carlo)) For any total function $f : \mathbb{N} \to \mathbb{N}$ is said to be total non-deterministic physical computable, if and only if there exists

$$(\Omega, \Sigma, \nabla, \mathcal{H}^*, \Delta)$$

s.t.

$\forall x \in \mathbb{N}$,

$$\Pr\{Event(x)\text{occurs}\} > 2/3$$

where

$$Event(x) \equiv \left(\lhd\big(\Delta \circ \mathcal{H}^* \circ \nabla\big)^\ulcorner x^\urcorner\big)\rhd \doteq {}^\ulcorner f(x)^\urcorner\right)$$

Note that before being initialized, the states should be some 'trivial' ones in Ω. The word 'trivial' means very easy or cheap to get or create. For the sake of clarity, we assume there are infinite many such trivial states. People can get any finite of them and only those being used will be discussed in the analysis of complexity. This is actually quite similar to the cases in Turing machine: Turing machine have infinite many checks and memories, but this does not imply every Turing machine will consume infinite many resource.

Readers may ask that why we do not talk about something like'evolution environments'. Well, because here 'environments' are also treated as states. In fact, 'environments' are usually constructed by some sorts of 'elements' and the 'blueprints' which instruct how to build a proper 'environment' when scale

changes. The 'blueprint' are actually the counterpart of the programmes in traditional computing. In addition, to make the whole physical computation a uniform method, the length of 'blueprints' are required to be finite. Generally speaking, infinite long 'blueprint' can produce physical experiments which can compute uncountable infinite many function. For example, non-uniform circuit model can exceed any Turing machines, because they can compute Turing's halting function. However, we don't discussed computational models which is not uniform. For simplicity, states denote 'environments' are not always written in this article, especially for some very easy environments. Readers may always think there is a Ω_{env} beside the main states space Ω_{main}, i.e.

$$\Omega \equiv \Omega_{env} \times \Omega_{main}.$$

Another fact we need to know is that for each physical system, $\mathscr{P}$ is not unique. People can use different way to explain and understand the same object. For example, if $\Delta \circ \mathcal{H} \circ \nabla$ computes a function, then $\Delta' \circ \mathcal{I} \circ \nabla'$ also computes the same function, where

$$\begin{aligned}\nabla' =& \mathcal{H} \circ \nabla \\ \Delta' =& \Delta\end{aligned}$$

$\mathcal{I}$ satisfies

$$\forall x(\mathcal{I}x = x).$$

$\mathcal{I}$ acts as a unit.

However, there may exist some 'well formed' decomposition. We may require the operator ∇ is a *functor* from Ω_Σ to Ω (or $\mathrm{Ran}(\nabla)$)(Of course, before doing this, it is trivial that Ω_Σ and Ω could be actually looked as categories or even cartesian closed categories), which satisfies the following axioms for functors:

$$\bar{x} \to \nabla\bar{x}$$

and

$$f\bar{x} \to \nabla f\bar{x}$$

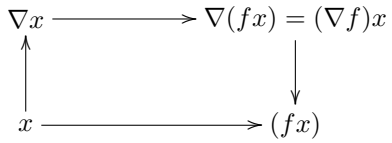
s.t.

$$\mathrm{id}_{\bar{x}} \to \nabla \mathrm{id}_{\bar{x}} = \mathrm{id}_{\nabla\bar{x}}$$

and

$$g \circ f \to \nabla g \circ f = \nabla g \circ \nabla f$$

for all morphism $f : \Omega_\Sigma \to \Omega_\Sigma$ and $g : \Omega_\Sigma \to \Omega_\Sigma$. i.e.



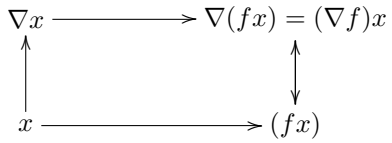
In addition, if ∇ is not only a functor, but also satisfies

$$\forall x_1 \forall x_2 (\nabla x_1 \neq \nabla x_2)$$

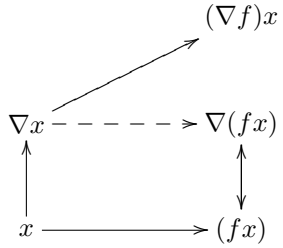
which means ∇ is $1-1$ on Ω and thus reversible or we can say ∇ is an *isomorphism* between Ω_Σ and $\mathrm{Ran}(\nabla)$. So in this case, Δ could be defined as

$$\Delta \equiv \nabla^{-1}.$$

Therefore, we have:



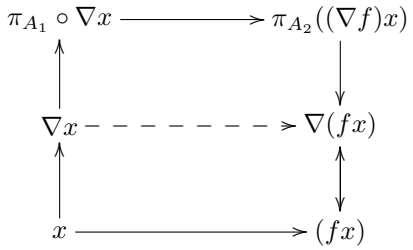
Unfortunately, in most cases, ∇ cannot be a non-trivial functor. $\nabla(f\bar{x})$ may not equals $(\nabla f)(\bar{x})$.



This may happen when the 'input' is a position with dimension L and the 'output' is velocity with dimension LT^{-1}. However, if there exist some project operators π_{A_i}, $i \in 1, 2, 3, \ldots$, s.t.

$$\exists A_1 \exists A_2 (\pi_{A_1}(\nabla(fx)) = \pi_{A_2}((\nabla f)(x)))$$

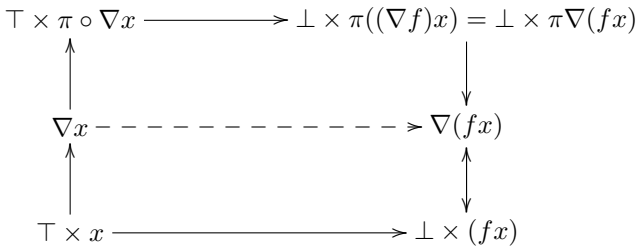
which indicate



s.t. Ω' is isomorphic to Ω_Σ under some isomorphism, but the functor may not be constructed by ∇ or π. Another definition may lead the both categories are not **ccc**. Suppose the initial states and the final states are always marked explicitly. That is, substitute $\Omega_{\text{bool}} \times \Omega_\Sigma$ for Ω_Σ. Thus π_{A_i} $i = 1, 2$ could be replaced as π, where π is defined as

$$\pi = \begin{cases} \pi_{A_1}, & \text{if b=0;} \\ \pi_{A_2}, & \text{o.w.} \end{cases}$$

and $\pi \circ \nabla$ and $\Delta \circ \pi^{-1}$ satisfies the axioms of functor, where π^{-1} is the original image of π. This method actually look $\top \times \nabla x$ and $\bot \times (\nabla f)(x)$ as elements in a certain equivalence class.



However, generally speaking, the extended space $\Omega_{\text{bool}} \times \Omega$ is not cartesian closed, for there may exist two objects without morphism between them.

Since Hilbert's 6th problem has not been solved yet, i.e. the whole theory of physics has not been axiomatized, we do not know that whether there exist some additional fundamental mathematical constraints should be included in this theory, though Beggs et al. have discussed the axioms of measurement based on Hempel's axioms[19]. Now, maybe the only restrictions here are the finiteness of the resource cost by a physical process and the finiteness of the attributes used to label a physical state set.

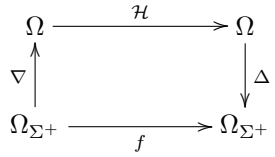
As a result this system may looks looser than many classical computational models and may contains the ability to surpass all these models.

Well, our framework is actually not a specific model, but acts as a template(or a meta-model) of some possible computational models. We would like to let physicists to add more necessary restrictions into the system. On the other hand, properties hold in this framework are universal and independent of any instances(that is, the models) of it. For example, any inputs or outputs of infinite length is forbidden in this framework, though some middle states of infinite details may be permitted.

Of course, when it comes to a specific branch of the physics, we can always know what is a legal states and processes. However, we wish to keep some freedom: to let the experimenter combine various axioms in physics so as to optimize the computations.

3.6 Complexity

For the physical systems defined above, we can even ignore that whether there exists a physical mechanism in reality to implement it. Any functions which could be written as the composition of the three operators would be considered as computable(deterministic version).



But any experiments which implement a certain system will cost resource. We will focus on four kinds of resource, namely, time, space, energy and mass.

One can also include 7 fundamental sorts of attributes considering 7 dimensions in SI(Systéme international d'unités). However, actually our decision is not a totally empirical one. In fact, many differential equations which are established to describe various phenomena involving these attributes are always related to energy, time, and space. After properly choosing unit to make all the constant into 1, we can just rewrite these dimensions in the forms of the combinations of 4 fundamental dimensions, which enable us to continue to use the 4 attributes to represent the increase of the resource. The counterexample may occur only when m or more than m new attributes appear in n equations and $m > n$, however these cases tend to be unlikely to happen, if these equations(theory) we discussed are assumed to be 'enough' for some phenomena in nature.

Definition 3.6.1. (**Resource**) The resource of a physical process $\mathfrak{R}$ includes:

- $\mathfrak{T}$: The (expectation of the)total time the whole process consumed;
- $\mathfrak{S}$: The maximum of (the expectation of)the space the whole process consumed;
- $\mathfrak{M}$: The maximum of (the expectation of)the mass the whole process consumed;
- $\mathfrak{E}$: The maximum of (the expectation of)the energy the whole process consumed.

and $\mathfrak{R} \equiv (\mathfrak{T}, \mathfrak{S}, \mathfrak{M}, \mathfrak{E})$

In the above definitions, the metric of them could be selected as the common ones. Today, most physicists tends to believe that mass and energy are not independent, neither do time and space. But for convenience, we still focus the primitive forms of resource, for actually we don't care about the independence here.

In the above definitions, we don't talk about the potential possibility that even time could be reused.

We convention that the resource is with respect to an inertial system, i.e. the observers obtain their results when they are in an inertial system to the system running the 'algorithms', so as to rule out the paradoxes because of the theory of relativity.

In many cases, we just cannot get a infinite precise estimation about the resource, but for our purpose, we actually do not need such things. Of course, there may exist some cases when we could not get an estimation without any promise of any precision, however, we will not use such processes to construct our implementation.

Suppose the projections of the fundamental attributes(resource) are $\pi_{\mathfrak{M}}(S)$, $\pi_{\mathfrak{S}}(S)$ and $\pi_{\mathfrak{E}}(S)$ Then the resource a physical process consumed is:

$$\begin{aligned}
\mathfrak{T}P &\equiv \pi_1 P \\
\mathfrak{M}P &\equiv \max\{\pi_{\mathfrak{M}}(S), S \in \mathrm{Ran}(\pi_2 P)\} \\
\mathfrak{S}P &\equiv \max\{\pi_{\mathfrak{S}}(S), S \in \mathrm{Ran}(\pi_2 P)\} \\
\mathfrak{E}P &\equiv \max\{\pi_{\mathfrak{E}}(S), S \in \mathrm{Ran}(\pi_2 P)\}
\end{aligned}$$

In general cases, when we have to discuss the process of superposition, the resource can be defined as:

$$\begin{aligned}
\mathfrak{T}P &\equiv \pi_1 P \\
\mathfrak{M}P &\equiv \max\{\mathrm{E}[\![\pi_{\mathfrak{M}}(S)]\!], S \in \mathrm{Ran}(\pi_2 P)\} \\
\mathfrak{S}P &\equiv \max\{\mathrm{E}[\![\pi_{\mathfrak{S}}(S)]\!], S \in \mathrm{Ran}(\pi_2 P)\} \\
\mathfrak{E}P &\equiv \max\{\mathrm{E}[\![\pi_{\mathfrak{E}}(S)]\!], S \in \mathrm{Ran}(\pi_2 P)\}
\end{aligned}$$

So it is easy to see that

$$\begin{aligned}
\mathfrak{T}P_2 \circ P_1 &= \pi_1 P_1 + \pi_1 P_2 \\
\mathfrak{M}P_2 \circ P_1 &= \max\{\mathfrak{M}P_1, \mathfrak{M}P_2\} \\
\mathfrak{S}P_2 \circ P_1 &= \max\{\mathfrak{M}P_1, \mathfrak{M}P_2\} \\
\mathfrak{E}P_2 \circ P_1 &= \max\{\mathfrak{M}P_1, \mathfrak{M}P_2\}
\end{aligned}$$

According to the definition above, the resource of a non-deterministic physical operator O which is initialized by $x \in \Omega$ should be defined as:

$$\begin{aligned}
\mathfrak{T}O(x) &\equiv \mathrm{E}[\![\mathfrak{T}O_i(x)]\!], O_i(x) \in O(x) \\
\mathfrak{M}O(x) &\equiv \mathrm{E}[\![\mathfrak{M}O_i(x)]\!], O_i(x) \in O(x) \\
\mathfrak{S}O(x) &\equiv \mathrm{E}[\![\mathfrak{S}O_i(x)]\!], O_i(x) \in O(x) \\
\mathfrak{E}O(x) &\equiv \mathrm{E}[\![\mathfrak{E}O_i(x)]\!], O_i(x) \in O(x)
\end{aligned}$$

So for operators' operation, we have:

$$\begin{aligned}
\mathfrak{T}O_2 \circ O_1(x) &= \mathfrak{T}O_1(x) + \mathfrak{T}O_2(\lhd O_1(x) \rhd) \\
\mathfrak{M}O_2 \circ O_1(x) &= \max\{\mathfrak{M}O_1(x), \mathfrak{M}O_2(\lhd O_1(x) \rhd)\} \\
\mathfrak{S}O_2 \circ O_1(x) &= \max\{\mathfrak{S}O_1(x), \mathfrak{S}O_2(\lhd O_1(x) \rhd)\} \\
\mathfrak{E}O_2 \circ O_1(x) &= \max\{\mathfrak{E}O_1(x), \mathfrak{E}O_2(\lhd O_1(x) \rhd)\}
\end{aligned}$$

3.6.0.1 Resource and Complexity

Definition 3.6.2. (Resource(deterministic))A resource the physical process which complete the whole computation consumed $\mathfrak{R}_{\mathscr{P}}$ including:

$$\begin{aligned}
\mathfrak{T}_{\mathscr{P}(x)} &\equiv \mathfrak{T}((\Delta \circ \mathcal{H} \circ \nabla)^{\ulcorner} x^{\urcorner}) \\
\mathfrak{S}_{\mathscr{P}(x)} &\equiv \mathfrak{S}((\Delta \circ \mathcal{H} \circ \nabla)^{\ulcorner} x^{\urcorner}) \\
\mathfrak{M}_{\mathscr{P}(x)} &\equiv \mathfrak{M}((\Delta \circ \mathcal{H} \circ \nabla)^{\ulcorner} x^{\urcorner}) \\
\mathfrak{E}_{\mathscr{P}(x)} &\equiv \mathfrak{E}((\Delta \circ \mathcal{H} \circ \nabla)^{\ulcorner} x^{\urcorner})
\end{aligned}$$

i.e. $\mathfrak{R}_{\mathscr{P}(x)} \equiv (\mathfrak{T}_{\mathscr{P}(x)}, \mathfrak{S}_{\mathscr{P}(x)}, \mathfrak{M}_{\mathscr{P}(x)}, \mathfrak{E}_{\mathscr{P}(x)})$

Definition 3.6.3. (Resource(Las Vegas))A resource the physical process which complete the whole computation consumed $\mathfrak{R}_{\mathscr{P}}$ including:

$$\begin{aligned}
\mathfrak{T}_{\mathscr{P}(x)} &\equiv \mathfrak{T}((\Delta \circ \mathcal{H}^* \circ \nabla)^{\ulcorner} x^{\urcorner}) \\
\mathfrak{S}_{\mathscr{P}(x)} &\equiv \mathfrak{S}((\Delta \circ \mathcal{H}^* \circ \nabla)^{\ulcorner} x^{\urcorner}) \\
\mathfrak{M}_{\mathscr{P}(x)} &\equiv \mathfrak{M}((\Delta \circ \mathcal{H}^* \circ \nabla)^{\ulcorner} x^{\urcorner}) \\
\mathfrak{E}_{\mathscr{P}(x)} &\equiv \mathfrak{E}((\Delta \circ \mathcal{H}^* \circ \nabla)^{\ulcorner} x^{\urcorner})
\end{aligned}$$

i.e. $\mathfrak{R}_{\mathscr{P}(x)} \equiv (\mathfrak{T}_{\mathscr{P}(x)}, \mathfrak{S}_{\mathscr{P}(x)}, \mathfrak{M}_{\mathscr{P}(x)}, \mathfrak{E}_{\mathscr{P}(x)})$

Definition 3.6.4. (Resource(Monte Carlo))A resource the physical process which complete the whole computation consumed $\mathfrak{R}_{\mathscr{P}}$ including:

$$\begin{aligned}
\mathfrak{T}_{\mathscr{P}(x)} &\equiv \mathfrak{T}((\Delta \circ \mathcal{H}^* \circ \nabla)^{\ulcorner} x^{\urcorner}) \\
\mathfrak{S}_{\mathscr{P}(x)} &\equiv \mathfrak{S}((\Delta \circ \mathcal{H}^* \circ \nabla)^{\ulcorner} x^{\urcorner}) \\
\mathfrak{M}_{\mathscr{P}(x)} &\equiv \mathfrak{M}((\Delta \circ \mathcal{H}^* \circ \nabla)^{\ulcorner} x^{\urcorner}) \\
\mathfrak{E}_{\mathscr{P}(x)} &\equiv \mathfrak{E}((\Delta \circ \mathcal{H}^* \circ \nabla)^{\ulcorner} x^{\urcorner})
\end{aligned}$$

i.e. $\mathfrak{R}_{\mathscr{P}(x)} \equiv (\mathfrak{T}_{\mathscr{P}(x)}, \mathfrak{S}_{\mathscr{P}(x)}, \mathfrak{M}_{\mathscr{P}(x)}, \mathfrak{E}_{\mathscr{P}(x)})$

The corresponding concept of complexity should be defined as the resource consumed with respect to the length of the input.

Definition 3.6.5. Complexity The complexity of a kind of resource is a function of the length of the input x,

$$\begin{aligned}
\mathbf{C}_{\mathfrak{T}}(n) &= \max\{\mathfrak{T}_{\mathscr{P}(x)} | n-1 \leq \log x \leq n, n = 1, 2, 3 \dots\} \\
\mathbf{C}_{\mathfrak{M}}(n) &= \max\{\mathfrak{M}_{\mathscr{P}(x)} | n-1 \leq \log x \leq n, n = 1, 2, 3 \dots\} \\
\mathbf{C}_{\mathfrak{S}}(n) &= \max\{\mathfrak{S}_{\mathscr{P}(x)} | n-1 \leq \log x \leq n, n = 1, 2, 3 \dots\} \\
\mathbf{C}_{\mathfrak{E}}(n) &= \max\{\mathfrak{E}_{\mathscr{P}(x)} | n-1 \leq \log x \leq n, n = 1, 2, 3 \dots\}
\end{aligned}$$

However, for simplicity, usually we'll continue to use asymptotic notations, such as big O notation, and the like.

Note: *actually, the finiteness of resource cost is a necessary precondition for all "uniform" physical computation models. Another necessary precondition is that all the physical attributes should be at some trivial states(very easily to be constructed, e.g.* $0°C$, $0m/s$*) before the experiments. If these requirement are not satisfied, the model will be a non-uniform one. This case is also discussed in details by Beggs et al.[19],[22]*

3.7 Some Common Examples

It is interesting to find some new methods to compute functions without the help of universal Turing machine. Through the ages, people have found a lot of such examples, the most famous of them are:

- Measure the volume of an object by putting it into the water;
- Obtain the centroid of an object by two-suspension method;
- Compute function sine by analog circuit;
- Decide the path of minimum cost using Fermat's Principle;
- Calculate the mean of numbers by the second Law of Thermodynamics.

Actually, we can give even more similar examples:

- By making use of resonance, we can easily find the desired tuning fork from a heap of them. Otherwise, we have to look up the label of them one by one and even have to compute the frequency one by one if there is no labels on them.
- We can compute the square root of an given number x by the law of free fall. Prepare a vacuum tube T of length x and let it stand vertically, then let an object o which is small enough fall. Get the time t when it touch the bottom, and we have $\sqrt{x} = t/c$, where $c = (2/g)^{1/2}$.

- We can sort a series of numbers through dangling poises by strings, where the strings satisfies Hooke's law. Given an array of numbers $\{x_i\}$ construct or find poises whose mass is just x_i, then dangle them by strings with the same stiffness coefficient. When the system is stable, the position of the poises with respect to their weight just indicate the relationship desired.

However, it is hard for us to estimate the cost of the methods above just after we describe them informally. So we select a part of them to analyze next.

Conventions: x is the representation of number in digits, $[x]$ is the value of x, $[x]^A$ means the attribute A has the value x. $[x]^{\Sigma^*}$ means the representation of quantity x though not on the tape.

3.7.1 Mean of Three Numbers

Given three numbers, compute the mean of them making use of law of thermodynamics. This idea comes from Pitowsky [7].

The strict description of the problem: Given three numbers x_1, x_2, $x_3 \in [\ 0, 100\ ]$, compute

$$\bar{x} = \frac{(x_1 + x_2 + x_3)}{3} \quad \text{(Accurate to two decimal places).}$$

Pitowsky suggests that since all of the three numbers less than 100 and bigger than zero, note that the freezing point of water is 0 C° and the boiling point of water is 100 C° under the one standard air pressure, So for each number x_i, we can prepare the corresponding water of volume V and temperature of $x_i C^\circ$. And then pour the water of three vessels into a bigger one, whose volume is $V'(V' > 3V)$, and wait. After the water arrived at the balance point, measure the temperature. Of course, we assume that during the whole procedure, no calory is lose.

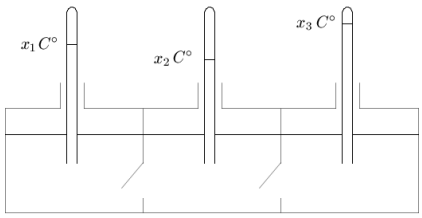


Figure 3.1: Mean of Three Numbers

Apparently, the physical state the method above deal with is the temperature of water, so we have

$$\Omega = \{\vec{t}\,|t_i \in [0, 100], i = 1, 2, 3\}^T.$$

on the other hand, we suppose the water is heat up from 0 C°, i.e. the initial state of the experiment is $([0]^T, [0]^T, [0]^T)$.

Therefore, the process could be depicted as following:

$\nabla : \Sigma^+ \to \Omega$, heat up the water to the desired temperature

$$\nabla(\ulcorner x_1 \urcorner, \ulcorner x_2 \urcorner, \ulcorner x_3 \urcorner) = (x_1^T, x_2^T, x_3^T)$$

$\mathcal{H} : \Omega \to \Omega$, admixture the water of different temperature, the second law of thermodynamics is used

$$\mathcal{H}(x_1^T, x_2^T, x_3^T) = (\bar{x}^T, \bar{x}^T, \bar{x}^T)$$

$\Delta : \Omega \to \Sigma^+$, measure the temperature of the water

$$\Delta(\bar{x}^T, \bar{x}^T, \bar{x}^T) = \ulcorner \bar{x} \urcorner$$

For this problem, since the precision is finite, and there are only constant (three) numbers and the numbers are bounded, we can easily deduct that $\mathfrak{R}_{\not\mathscr{P}}$ is a constant. As a matter of fact, for Turing machine, we can also find a constant resource costing algorithm which is just looking up a finite list to solve the problem.

3.7.2 Compass and straightedge constructions

We look 'compass and straightedge constructions'(CSC) as a sort of physical computation just because of the fact that physical laws permit us to create compasses and rulers to help us to do some special computation. Another implicit postulate is that people can find a 'ideal' or good enough physical plane where Euclid's axioms hold.

Note that though CSC is come from real rulers and compasses, there are some difference between them. The ruler here is infinitely long, and only has one side. Further more, there is no markings on the ruler(this is why the ruler is called 'straightedge'). The compass is assumed to collapse when lifted from the page, so may not be directly used to transfer distances. (This is an unimportant restriction, as this may be achieved via the compass equivalence theorem.)

All compass and straightedge constructions consist of repeated application of five basic constructions using the points, lines and circles that have already been constructed. These are:

- Creating the line through two existing points
- Creating the circle through one point with centre another point
- Creating the point which is the intersection of two existing, non-parallel lines

- Creating the one or two points in the intersection of a line and a circle (if they intersect)
- Creating the one or two points in the intersection of two circles (if they intersect).

People have proved that there many problems which is impossible to solve just by CSC. For example, Carl Friedrich Gauss in 1796 showed that a regular n-sided polygon can be constructed with ruler and compass if the odd prime factors of n are distinct Fermat primes. Gauss conjectured that this condition was also necessary, but he offered no proof of this fact, which was provided by Pierre Wantzel in 1837.

One of the most famous and interesting examples is to 'compute' function $\sqrt{x}$ within system CSC. In fact, a Turing machine could be called to compute $\lceil\sqrt{x}\rceil$(of course, this one is not necessarily the fastest one).

$\lceil\sqrt{x}\rceil$:

	0	1
1	0R19	0R2
2	0R3	1R2
3	1R4	
4	1L5	1R4
5		0L6
6	0L13	1L7
7	0L8	1L7
8	0R9	1L8
9	0R16	0R10
10	0R16	0R11
11	0R12	1R11
12		1R4
13	0R14	1L13
14	0R19	0R15
15	0R19	0R1
16	0R17	1R16
17	0R18	1L17
18		0R19

If people choose to use CSC, he'll get following series of operations:

- Given the segment L_0 of length x
- $\mathcal{L}_1$:Create a segment l of length 1 beside the segment L_0, suppose the intersection of two segments is point p.
- $\mathcal{C}$:Create a circle c_0 whose diameter is $1 + x$
- $\mathcal{L}_2$:Create a line l' pass p and is perpendicular to the segment L_0, suppose the intersection of l' and c_0 is p'.

A few of computation will show that the length of $\overline{p'p}$ is $\sqrt{x}$.

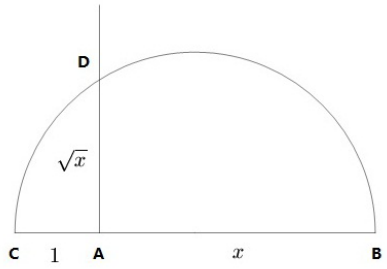


Figure 3.2: Compass and straightedge construction

The whole procedure may be expressed as

$$\Delta \circ \mathcal{L}_2 \circ \mathcal{C} \circ \mathcal{L}_1 \circ \nabla$$

Where

$$\begin{array}{rccc}
\nabla: & \ulcorner x \urcorner & \rightarrow & \{\overline{\mathrm{AB}}\} \\
\mathcal{L}_1: & \{\overline{\mathrm{AB}}\} & \rightarrow & \{\overline{\mathrm{AC}}, \overline{\mathrm{AB}}\} \\
\mathcal{C}: & \{\overline{\mathrm{AC}}, \overline{\mathrm{AB}}\} & \rightarrow & \{\overline{\mathrm{AC}}, \overline{\mathrm{AB}}, \widetilde{\mathrm{BC}}\} \\
\mathcal{L}_2: & \{\overline{\mathrm{AC}}, \overline{\mathrm{AB}}, \widetilde{\mathrm{BC}}\} & \rightarrow & \{\overline{\mathrm{AC}}, \overline{\mathrm{AB}}, \widetilde{\mathrm{BC}}, \overline{\mathrm{AD}}\} \\
\Delta: & \{\overline{\mathrm{AC}}, \overline{\mathrm{AB}}, \widetilde{\mathrm{BC}}, \overline{\mathrm{AD}}\} & \rightarrow & \ulcorner |\overline{\mathrm{AD}}| \urcorner
\end{array}$$

Compass and ruler construction actually take advantage of human's eyes which could find a desired point quickly, though it is still not infinitely precise. However, even assume there exists an 'ideal' machine which could be used to replace human's eyes, CSC system is not very efficient. Considering the length of the input x which is $\lceil \log(x) \rceil$ and the length of $\overline{\mathrm{AB}}$ which is x, we know that the space complexity of the operator ∇ is exponential, thus the complexity of the whole method is also exponential. On the other hand, the time complexity of the method is exponential either.

3.7.3 Sorting Without Repeat

Description of the Problem:

Input: Finite number series of length n:

$$A = \{x_i | x_i \in \mathbb{Z}^+ \cap [0, M](0 \leq i \leq n)\};$$

Output: Finite number series of length m:

$$B = \{x_j | x_j \in A(0 \leq j \leq m)\},$$

s.t. if $j_1 < j_2$ then $x_{j_1} < x_{j_2}$.

Our plan is: for the given series, select a series of poises of length n, s.t. the mass of the ith poise is equivalent to the ith number. Dangling the poises from right to left by strings, whose restoring coefficient are k. Wait until the system is stationary, open the parallel light source and measure the projection onto the vertical ruler at the right end. The measurement could be done by machines and present the results onto the tape for observer. For Example, we can embed some photoconductive diodes in the ruler by graduations, diodes who is not triggered should be read.

The physical state the method is primarily concerned with is the mass of poise M, the horizontal positions of the poises X and the vertical ones Y, the projections Y' and the boole value B indicating which diodes is triggered, i.e.

$$\Omega \equiv \oplus_i^n \{m_i\}^M \times \{x_i\}^X \times \{y_i\}^Y \times \oplus_j \{(j, B_j)\}^{Y' \times B}$$

So we have

$\nabla : \Sigma^+ \to \Omega$ (Select poises)

$$(\oplus_i^n x_i) \to (\oplus_i [x_i]^M [i]^X [0]^Y \oplus_j^M [(j,0)]^{Y' \times B})$$

$\mathcal{H} : \Omega \to \Omega$ (Dangle poises)

$$\begin{pmatrix} \oplus_i^n [x_i]^M \\ [i]^X \\ [0]^Y \\ \oplus_j^M [(j,0)]^{Y' \times B} \end{pmatrix} \to \begin{pmatrix} \oplus_i^n [x_i]^M \\ [i]^X \\ [[x_i] g/k]^Y \\ \oplus_j^M [(j,0)]^{Y' \times B} \end{pmatrix}$$

$\mathcal{H}' : \Omega \to \Omega$(Open the parallel light)

$$\begin{pmatrix} \oplus_i^n [x_i]^M \\ [i]^X \\ [[x_i] g/k]^Y \\ \oplus_j [(j,0)]^{Y' \times B} \end{pmatrix} \to \begin{pmatrix} \oplus_i^n [x_j]^M \\ [j]^X \\ [[x_j] g/k]^Y \\ \oplus_j^M [(j, \epsilon_A(j))]^{Y' \times B} \end{pmatrix}$$

$\Delta : \Omega \to \Sigma^+$(read the projection)

$$\begin{pmatrix} \oplus_i^n ([x_j]^M \\ [j]^X \\ [[x_j] g/k]^Y \\ \oplus_j^M [(j, \epsilon_A(j))]^{Y' \times B}) \end{pmatrix} \to \oplus_{j'} (x_{j'})$$

satisfies if $j_1 < j_2$ then

$$[x_{j_1}] < [x_{j_2}]$$

Considering the ideal implementation, we conclude that the $\mathfrak{R}_{\mathscr{P}}$ is linear, which is superior to Turing machines using comparisons, for the complexity for them was proved to be $O(n \log n)$. However, there does exist Turing machine,

which is not based on comparisons, also has a linear time cost.

Note that if the number series is boundless, the complexity of the method above will be exponential. This is the common defect of most analog computers.

3.7.4 Volume of irregular shape

For this issue, we shall restrict the range of the saying 'irregular' so as to rule out the objects with infinite length of description. So actually, we tend to discuss a subset of the set of all cases.

Description of the problem:

Inputs: point series of length n:$(x_i, y_i)(1 < i < n)$, satisfies $c + r \leq x_i \leq a - c - r, c + r \leq y_i \leq b - c - r$

Outputs: The volume of the box of length a and width b and height h_0, not including the series of cylinders(radius: r height: h_0) which are induced by the series of points. In addition, the parts which are isolated from the side because of the cylinders are also excluded.

Our plan is simple. Assume we have a box of material of dense ρ, and a punch to extract circles from it. Then we measure the mass of the rest then divide it by its dense or just put it into water.

$\nabla : \Sigma^+ \to \Omega$

$$\nabla(\oplus_{i=1}^{n} a(x_i, y_i)) = [\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))]^{M\ulcorner}0^{\urcorner}$$

$\mathcal{H}_1 : \Omega \to \Omega$

$$\mathcal{H}_1[\rho h_0(A - m(\cup_{i=1}^{n} c_i))]^{M\ulcorner}0^{\urcorner}$$

$$= [\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))]^{M\ulcorner}\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))^{\urcorner}$$

$\mathcal{H}_2 : \Omega \to \Omega$

$$\mathcal{H}_2[\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))]^{M\ulcorner}\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))^{\urcorner}$$

$$= [\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))]^{M\ulcorner}h_0(A - m(\bigcup_{i=1}^{n} c_i))^{\urcorner}$$

$\Delta : \Omega \to \Sigma^+$

$$\Delta[\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))]^{M\ulcorner}h_0(A - m(\bigcup_{i=1}^{n} c_i))^{\urcorner} = {\ulcorner}h_0(A - m(\bigcup_{i=1}^{n} c_i))^{\urcorner}$$

Apparently the resource complexity for this method is linear with respect to the number of the points. However, because most people think that we cannot do infinitely measurement during one experiment, this method can only provide

the result of finite precision. This is a good news to Turing machines because this implies there exists a Turing machine which is almost equivalently efficient.

This may be astonish to someone, who may thought that a TM should at least solve the equations first. However, because of the finite precision, Turing machine can just split the object into lattice and use the so-called scan-line algorithm to find the answer.

3.7.5 The centroid of Irregular Shape

Just as the last example, we restrict our topic into the same subsets of all cases.

Description of Problem:

Inputs: point series of length n:$(x_i, y_i)(1 < i < n)$, satisfies $c + r \leq x_i \leq a - c - r, c + r \leq y_i \leq b - c - r$

Outputs: The centroid of the box of length a and width b and height h_0, not including the series of cylinders which is induced by the series of points. In addition, the parts which are isolated from the side because of the cylinders are also excluded.

The method we suggest is similar to the last one, the difference of them is that this time we will record some points.

$\nabla : \Sigma^+ \to \Omega$

$$\nabla(\oplus_{i=1}^{n} a(x_i, y_i)) = [\rho h_0(A - m(\cup_{i=1}^{n} c_i))]^{M\ulcorner}0^{\urcorner\ulcorner}0^{\urcorner\ulcorner}0^{\urcorner}$$

$\mathcal{H}_1 : \Omega \to \Omega$ (Suspend the box by V_0)

$$[\rho h_0(A - m(\cup_{i=1}^{n} c_i))]^{M\ulcorner}0^{\urcorner\ulcorner}0^{\urcorner\ulcorner}0^{\urcorner}$$

$$\to [\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))]^{M\ulcorner}\left[\frac{c - V_0}{|c - V_0|} + V_0\right]^{\urcorner\ulcorner}0^{\urcorner\ulcorner}0^{\urcorner}$$

$\mathcal{H}_2 : \Omega \to \Omega$ (Suspend the box by V_0')

$$[\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))]^{M\ulcorner}\left[\frac{c - V_0}{|c - V_0|} + V_0\right]^{\urcorner\ulcorner}0^{\urcorner\ulcorner}0^{\urcorner}$$

$$\to [\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))]^{M\ulcorner}\left[\frac{c - V_0}{|c - V_0|} + V_0\right]^{\urcorner\ulcorner}\left[\frac{c - V_0'}{|c - V_0'|} + V_0'\right]^{\urcorner\ulcorner}0^{\urcorner}$$

$\mathcal{H}_3 : \Omega \to \Omega$(Extend the unit vectors: Get the point of intersection)

$$[\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))]^{M\ulcorner}\left[\frac{c - V_0}{|c - V_0|} + V_0\right]^{\urcorner\ulcorner}\left[\frac{c - V_0'}{|c - V_0'|} + V_0'\right]^{\urcorner\ulcorner}0^{\urcorner}$$

$$\to [\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))]^{M\ulcorner}\left[\frac{c - V_0}{|c - V_0|} + V_0\right]^{\urcorner\ulcorner}\left[\frac{c - V_0'}{|c - V_0'|} + V_0'\right]^{\urcorner\ulcorner}c^{\urcorner}$$

$\Delta : \Omega \to \Sigma^+$

$$[\rho h_0(A - m(\bigcup_{i=1}^{n} c_i))]^M \ulcorner \left[\frac{c - V_0}{|c - V_0|} + V_0\right] \urcorner \ulcorner \left[\frac{c - V_0'}{|c - V_0'|} + V_0'\right] \urcorner \ulcorner c \urcorner$$

$$\to \ulcorner c \urcorner = \ulcorner c_x \urcorner \ulcorner c_y \urcorner$$

The time the system cost from oscillating to stillness can be bounded by a constant. Because of the same reason this method does not break up the lower bound of Turing machine. But for some other things, we tend to pay more attention to it. Some relevant issues will be discussed in Sec-VI.

3.8 Graph Isomorphism, Graph Spectrum and Oscillators

In this part of the section, we shall talk about a complex example in detail. We do not mean to show that the method we designed here is superior to all of the Turing machines constructed by the people of the same aim. We just want to demonstrate a new style of computation.

3.8.1 Spectrum of Graph

Suppose $X = (V, E)$ is a graph, A is it's adjacent matrix. We say $f_A(\lambda)$ is the characteristic polynomial of X, also denoted by $f_X(\lambda)$. $(\lambda_1, \ldots, \lambda_n)$, the whole root of $f(\lambda)$, is called the spectrum of graph X.

Actually two different adjacent matrices may represent two isomorphic graphs. If we alter the permutation of the number of the vertices, A will become $P^{-1}AP$, where P is the corresponding permutation matrix. However, the characteristic polynomials of them are the same. Therefore, $f_X(\lambda)$ and the spectrum $\text{spec}(X) = (\lambda_1, \ldots, \lambda_n)$ are uniquely determined by X.

For the relationship between spectrum and graph, people conjectured that graph can be uniquely determined by spectrum, i.e. suppose

$$\text{spec}(A) = \text{spec}(B),$$

can we conclude that

$$A \backsimeq B?$$

Unfortunately, the different graphs of the same spectrum were found soon[63].

Nonetheless, calculating the spectrum is also important. Because we can know a lot of crucial properties[38][53], such as the extensionality, rapid mixing time of Markov chains on the graph, by the spectrum of the graph. What's more, when two graph have same spectrum, and spectrum is never repeated, we have a polynomial time algorithm to check whether they are isomorphic.

1. Input graphs G_1,G_2, compute their spectrum, denoted by Λ_1,Λ_2.

2. Compare the spectrums, if $\Lambda_1 \neq \Lambda_2$, then return NOT ISOMORPHIC;else, continue;

3. Check whether the product of the two similar matrices is a permutation matrix, if it is true return ISOMORPHIC, otherwise return NOT ISOMORPHIC;

Notation: Here by $\Lambda_1 \neq \Lambda_2$ we mean after sorting their eigenvalue, the two series are not identical to each other. And accordingly G_1, G_2 should also be altered into $\widetilde{G}_1$, $\widetilde{G}_2$. But for convenience, we do not differeciate G_i and $\widetilde{G}_i$.
Proof:

If $\Lambda_1 \neq \Lambda_2$, then $G_1 \ncong G_2$. So we only consider the case in which $\Lambda_1 = \Lambda_2 = \Lambda$.
i.e.Suppose

$$G_1 = P\Lambda P^T,\ G_2 = Q\Lambda Q^T$$

then we have

$$P^T G_1 P = \Lambda = Q^T G_2 Q$$

thus

$$G_1 = \left(QP^T\right)^T G_2 \left(QP^T\right)$$

by the preconditon,Λ is never repeating, so P,Q is the unique orthganol matrices.
the rest is to show that if G_1, G_2 is isomorphic, then QP^T is the permutation matrix desired.
In fact, if $G_1 \cong G_2$,then there exists a permutation matrix S s.t.

$$G_1 = S^T G_2 S$$

Since $G_2 = Q\Lambda Q^T$, the formula above means

$$G_1 = S^T Q\Lambda Q^T S = \left(Q^T S\right)^T \Lambda \left(Q^T S\right)$$

Because of the uniqueness of P, we can conclude that $Q^T S = P^T$, and by orthgonality of Q, we obtain

$$S = QP^T.$$

□

3.8.2 Harmonic Oscillator of multi-freedom

Suppose s is the number of freedom of the system, $q_{\alpha 0}(\alpha = 1, 2, \ldots, s)$ is the general coordinates[52][51] when the system is in balance. Without lose of generality, we can always assume that $q_{\alpha 0}$ is just zero, i.e. $q_{\alpha 0} = 0(\alpha = 1, 2, \ldots, s)$.

Because we only talk about little vibration, so we only keep several terms in the Taylor series of the Lagrangians L of the system about $q_{\alpha 0}$.

The potential energy:

$$V = V_0 + \sum_{\alpha=1}^{s} \left(\frac{\partial V}{\partial q_\alpha}\right)_0 q_\alpha + \sum_{\alpha=1}^{s}\sum_{\beta=1}^{s} \frac{1}{2}\left(\frac{\partial^2 V}{\partial q_\alpha \partial q_\beta}\right)_0 q_\alpha q_\beta + \cdots .$$

Note that V_0 can be omitted. Introduce the notation $k_{\alpha\beta}$,

$$k_{\alpha\beta} = k_{\beta\alpha} = \left(\frac{\partial^2 V}{\partial q_\alpha \partial q_\beta}\right)_0 ,$$

which is called the strength coefficient. According to the formula $\left(\frac{\partial V}{\partial q_\alpha}\right)_0 = 0$, the second order of the potential energy could be represented as

$$V = \frac{1}{2}\sum_{\alpha=1}^{s}\sum_{\beta=1}^{s} k_{\alpha\beta} q_\alpha q_\beta .$$

Then assume $\boldsymbol{r}_i = \boldsymbol{r}_i(q)$ is not relevant to time, i.e. the obligation is constant, so the kinetic energy is:

$$T = \frac{1}{2}\sum_{t=1}^{n} m_i \dot{\boldsymbol{r}}_i \cdot \dot{\boldsymbol{r}}_i = \frac{1}{2}\sum_{i=1}^{n}\sum_{\alpha=1}^{s}\sum_{\beta=1}^{s} m_i \frac{\partial \boldsymbol{r}_i}{\partial q_\alpha}\cdot\frac{\partial \boldsymbol{r}_i}{\partial q_\beta}\dot{q}_\alpha \dot{q}_\beta .$$

Introduce the symbol $m_{\alpha\beta}$,

$$m_{\alpha\beta} = m_{\beta\alpha} = \sum_{i=1}^{n} m_i \frac{\partial \boldsymbol{r}_i}{\partial q_\alpha}\cdot\frac{\partial \boldsymbol{r}_i}{\partial q_\beta},$$

then the kinetic energy could be represented as

$$T = \frac{1}{2}\sum_{\alpha=1}^{s}\sum_{\beta=1}^{s} m_{\alpha\beta}\dot{q}_\alpha \dot{q}_\beta .$$

Keep the formula above to second order and since $\dot{q}_\alpha \dot{q}_\beta$ is second order $m_{\alpha\beta}$ should be expanded to zeroth order. In other words $m_{\alpha\beta}$ could be looked as constants, we just take the value of them when the system is in balanced point.

So the Lagrangian could be written as

$$L = \frac{1}{2}\sum_{\alpha=1}^{s}\sum_{\beta=1}^{s}(m_{\alpha\beta}\dot{q}_\alpha \dot{q}_\beta - k_{\alpha\beta} q_\alpha q_\beta).$$

Thus the Lagrangian equation is

$$\frac{d}{dt}\frac{\partial}{\partial\dot{q}_\alpha}\left(\frac{1}{2}\sum_{\beta=1}^{s}\sum_{\gamma=1}^{s}m_{\beta\gamma}\dot{q}_\beta\dot{q}_\gamma\right)-\frac{\partial}{\partial q_\alpha}\left(-\frac{1}{2}\sum_{\beta=1}^{s}\sum_{\gamma=1}^{s}k_{\beta\gamma}q_\beta q_\gamma\right)=0.$$

i.e.

$$\frac{d}{dt}\left(\frac{1}{2}\sum_{\gamma=1}^{s}m_{\alpha\gamma}\dot{q}_\gamma+\frac{1}{2}\sum_{\beta=1}^{s}m_{\beta\alpha}\dot{q}_\beta\right)+\left(\frac{1}{2}\sum_{\gamma=1}^{s}k_{\alpha\gamma}q_\gamma+\frac{1}{2}\sum_{\beta=1}^{s}k_{\beta\alpha}q_\beta\right)=0.$$

therefore

$$\sum_{\beta=1}^{s}m_{\alpha\beta}\ddot{q}_\beta+\sum_{\beta=1}^{s}k_{\alpha\beta}q_\beta=0\quad(\alpha=1,2,\ldots,s).$$

Let

$$q_\beta=A_\beta e^{\lambda t}\quad(\beta=1,2,\ldots,s).$$

Take it into the former formula, we get the linear equations for A_β.

$$\sum_{\beta=1}^{s}(m_{\alpha\beta}\lambda^2+k_{\alpha\beta})A_\beta=0\quad(\alpha=1,2,\ldots,s).$$

If the equations have non-trivial solutions, then following conditions should be hold:

$$\begin{vmatrix}m_{11}\lambda^2+k_{11} & m_{12}\lambda^2+k_{12} & \cdots & m_{1s}\lambda^2+k_{1s}\\ m_{21}\lambda^2+k_{21} & m_{22}\lambda^2+k_{22} & \cdots & m_{2s}\lambda^2+k_{2s}\\ \vdots & \vdots & & \vdots\\ m_{s1}\lambda^2+k_{s1} & m_{s2}\lambda^2+k_{s2} & \cdots & m_{ss}\lambda^2+k_{ss}\end{vmatrix}=0$$

This is the equations of times s of λ^2, and we can get s λ^2, denoted by

$$\lambda_l^2\quad(l=1,2,\cdots,s).$$

3.8.3 The characteristic oscillators for a Graph

Making use of the conclusions above, we construct a specific oscillators for any given connected graph.

Denote the vertices of graph by numbers $1\sim n$, according to any order. The mass of a vertex is set 1g. Connect the vertex 1 and n to ends by strings whose k is zero by that direction. For the rest, we connect them according to the adjacent matrix, i.e. if $A_{ij}=1$(Note that $A_{ij}=A_{ji}$), connect vertex i and j by a string whose $k=1$. Let's study the motion of the system: First, if two vertices is not connected by string, we have $k_{\alpha\beta}=k_{\beta\alpha}=0$. Second, the vibration is little, so string is not an obligation. And we take the general coordinates as the usual displacement vectors, so $m_{\alpha\beta}=m_{\beta\alpha}=\delta_{\alpha\beta}$, where $\delta_{\alpha\beta}$ is the well known Kronecker notation.

At last, we obtain the determinant as follows, which is the characteristic polynomial of our system.

$$\begin{vmatrix} \lambda^2+d_1 & -A_{12} & \cdots & -A_{1s} \\ -A_{21} & \lambda^2+d_2 & \cdots & -A_{2s} \\ \vdots & \vdots & & \vdots \\ -A_{s1} & -A_{s2} & \cdots & \lambda^2+d_s \end{vmatrix} = 0$$

It has been proved that $\lambda^2 < 0$. So let $-\Lambda = \lambda^2$,we can see that the determinant above actually compute the spectrum of A' which is converted from A by adding multi-loops(the number of degrees). If a vertex is in the characteristic position, it will take part in the vibrations of all frequencies, if no one is in the characteristic position, then they just vibrate with respective frequency. In both cases, we'll measure the frequency and differentiate them by means of FFT, so as to get the spectrum of A'.

Apparently, adding multi-loops is not harmful to the decision of whether A and B are isomorphic, for if $A' \neq B'$, then $A \ncong B$ If $A \backsimeq B$, then $A' \backsimeq B'$, which will also be checked by the oscillating system.

3.8.4 Comments

The whole procedure could be written as $\Delta \circ \mathcal{F} \circ \mathcal{S} \circ \mathcal{H} \circ \nabla$, where:

$$\nabla : A \to \hat{A} \times \vec{W}|_{W=\vec{0}}$$

$$\mathcal{H} : \hat{A} \times \vec{0} \to \hat{A} \times \vec{W}$$

$$\mathcal{S} : \hat{A} \times \vec{W} \to \hat{A} \times \tilde{\vec{W}}$$

$$\mathcal{F} : \hat{A} \times \tilde{\vec{W}} \to \hat{A} \times \vec{F}$$

$$\Delta : \hat{A} \times \vec{F} \to \tilde{\vec{F}}$$

$\mathcal{S}$ is actually named after the word 'sampling' and $\mathcal{F}$ is named after the 'Fourier'.

Suppose now the problem we want to solve is: try to find the n_0-th(n_0 is a constant) value of such matrices. It is easy to see that the period of the system satisfies

$$\frac{1}{\sqrt{n}f_0} \leq T \leq \frac{\sqrt{n}}{f_0} (\text{or } \frac{1}{\sqrt{n}} f_0 \leq f \leq \sqrt{n} f_0),$$

where f_0 is the eigenfrequency of a single string, considering the minimum case occurs when the corresponding graph is totally parallel connected(two vertices with $n-$multiple edges between them), while the maximum case occurs when it is just a chain.

The total steps of sampling should be

$$N = 2BL = 2\left(\sqrt{n}f_0 - \frac{f_0}{\sqrt{n}}\right)\left(c_{n_0}\frac{\sqrt{n}}{f_0}\right)$$

where the constant c_{n_0} is related to the required precise n_0. So we have: $O(N) = O(n)$ and the corresponding complexity for fast fourier transform should be $O(n \ln n)$.

So we can say that the time complexity of this method should be

$$O(n^2) + O\left(\frac{\sqrt{n}}{f_0}\right) + O(n \ln n) + O(n) = O(n^2)$$

where the left $O(n^2)$ is the cost of constructing the system and $O(n \ln n)$ is the complexity of FFT. On the other hand, if we use the well-known algorithm called QR-method to get the answer, it will cost such Turing machine $O(n^3)$ steps. However, we know little about wether our method is superior to any most efficient Turing machines.

However, such analysis may not be enough. We cannot obtain a reliable result unless some specifications for the materials are considered. The procedure of constructing the system according to an arbitrary graph is actually quite tricky. For this system need n oscillators of the same mass and different length. To achieve this, we have to assume that there exists a kind of 'ideal' material which, at least for a large enough range, can be stretched to a desired length easily(in $O(n)$ time and totally $O(n^2)$) and, at the same time, keep rigid. Another more natural setting may be that there are infinite many such sticks and all of them are located by order.

3.9 Steiner Tree Problem

Steiner Tree Problem is a problem in combinatorics. The general version of Steiner Tree Problem is NP-complete[53], which implies that this problem is unlikely solved in polynomial time.

This problem is similar to the Minimal Spanning Tree Problem in metric space. The difference is that Steiner Tree Problem allow people to add new points $v'(v' \notin V)$ and new edges $e'(e' \notin E)$ into the original graph G, if necessary. When $|G| = 3$, the new point (in this case, at most one point is needed)is called Fermat point.

At a time, some people became to believe that the experiments of soup membrane can be used to solve the Steiner Tree Problem. In fact, when $|G|$ is small, say, less than 5, this method really works. However, when the number of vertices is 10 or more, this experiment just cannot give the right answer. One can attribute the failure to different reasons and derive various explanations. Of all these potential explanations, the one which states that 'it is just the errors during the experiment cause the failure' made many people conjecture faithfully that classical mechanics can be used to solve NP-complete Problems in polynomial time(So they try to proof P=NP).

In fact, the foundation of the experiment is the well-known property that the membrane will stay at a stationary state, where the surface it produces will be

just the minimal surface. Unfortunately, this theory has nothing to do with the fact that the membrane can arrive at the stationary state *fast*. What's more, no one can proof the soundness of such property under the framework of classical mechanics.

3.10 DNA Computation

In 1994, Adleman used a probabilistic DNA algorithm to solve HP problem (Hamilton Path Problem). HP problem is NP-complete, which implies it is difficult to find a polynomial algorithm to solve HP[53][37][14].

In order to understand Adleman's method, the following knowledge seems necessary.

(1) DNA contains chains consisted by four types of nucleotides, denoted by A, C, G and T.

(2) These nucleotides forms complementary couples, i.e. A and T are complementary, C and G are complementary. If the corresponding positions of two DNA chains are complementary, they will patch up as the twin-helix structure.

(3) PCR, which proposed by Kary Mullis, is method to reproduce the specific chain we need.

(4) There is a machine called 'sequencer' which can be used to read out the series of a DNA chain.

Adleman's Algorithm contains five procedures(Suppose $|G| = n$):

(1) Randomly produce the paths in the Graph, encoded by DNA chains.

(2) Keep only those paths which begin with v_{in} and end with v_{out}.

(3) Keep only the paths whose length is n

(4) Keep only those paths which enter all vertices in G at least once.

(5) If any paths remain, return 'True', else return 'False'.

Note that the first step of Adleman's Algorithm, which is usually thought to be work as an initialization operator ∇, is not polynomial with respect to the resource mass $\mathfrak{M}$ and space $\mathfrak{S}$ at least. Considering asymptotically we can only sequentially get the mass the algorithm need, so actually $O(n!)$ mass can cause $O(n!)$ time $\mathfrak{T}$. As a matter of fact the other steps of this algorithm, which require exponentially molecules fully blend by polynomially increasing contacting facades, also cost a lot of resource $\mathfrak{T}$.

It is not very hard to appreciate the conclusion that we can obtain great power of computation suppose we are provided with corresponding quantity of

mass, and do not take the cost of preparing such equipment at all. For one thing, let's consider the following ideal model.

Suppose we have enough universal Turing machines, each of them are denoted by their footnotes. What's more, by some altering in the definition, these UTMs have the ability to transmit their results to others. And the condition of two UTMs $U_i, U_j (i \neq j)$ could communicate to each other is that they are adjacent to each other, denoted by $Adj(U_i, U_j)$.

So the computational model constructed following, called 'Turing Tree', can exponentially speed up the computation of any NP-complete problems.

Definition 3.10.1. *(Turing Tree) Suppose we have infinite many UTMs, each of them denoted by unique footnotes, and*

$$Adj(U_i, U_j) \Leftrightarrow j = 2i + 1 \vee i = 2j + 1 \vee j = 2i + 2 \vee i = 2j + 2,$$

then we call this Turing Tree.

It is easy to see that the following relation holds:

$$Adj(U_0, U_1), Adj(U_0, U_2)$$
$$Adj(U_1, U_3), Adj(U_1, U_4), Adj(U_2, U_5), Adj(U_2, U_6)$$
$$\cdots\cdots\cdots$$

For example, a TSP problem can be solved as following:

a The Observer input the weighted complete graph G to the U_0, U_0 decode 0 to a permutation and compute the sum of the weight, and then transmit G and flag $F = 0$ to U_1, U_2.

b For index i,After U_i get $F = 0$ and G, it check whether $i < \lceil \log_2 n! \rceil$,if the answer if 'yes' then decode i to a permutation and get the sum, and transmit G and $F = 0$ to U_{2i+1} and U_{2i+2};else check whether $i = \lceil \log_2 n! \rceil$, if it is true, decode i to a permutation and get the sum, then submit the weight sum to the $U_{\lceil i/2 \rceil - 1}$. Else, do nothing.

c For index i, after U_i get $F = 1$ and two sum(come from U_{2i+1}, U_{2i+2}), if it's index is not zero, then submit $\min\{S_i, S_{2i+1}, S_{2i+2}\}$ and $F = 1$ to $U_{\lceil i/2 \rceil - 1}$. Else return $\min\{S_0, S_1, S_2\}$ and write it on to the tape.

It is easy to check that the subprocedure of the algorithm which is used to decode a natural number to a permutation is polynomial. So the cost of time the Turing Tree consumed should be $O(2 \log_2 n!) \leq O(2n \log_2 n)$ (Including once sharing the task and once championship for the minimum). So it is the time to answer how can we 'easily' construct a big enough Turing Tree.

3.11 N-body System and ECT

Let's recall the statement of ECT(by YAO): *if a function is computable by some hardware device in time $T(n)$ for input of size n, then it is computable by a Turing machine in time $(T(n))^k$ for some fixed k(dependent on the problem).*

Actually, there are lots of notorious examples in classical mechanics that the results cannot be easily calculated by usual computing devices. In fact, we can say people have not found a Turing machine, by which the mathematical counterpart of these physical problems could not be solved in polynomial time with respect to the length of input.

So naturally people can ask: can these examples provide evidences or even a proof indicating that the ECT is wrong?

Again, we should formalized such ideas.

3.11.1 Chaotic systems

Firstly, we consider chaotic systems. When systems are chaotic, their behavior are very difficult to predict. This, is due to one of the most crucial properties of chaotic systems: being extremely sensitive to errors. Generally speaking, even a most tiny change of the initial state can cause huge difference to the final result. Usually, Ljapunov exponent can help us to estimate the average increment of infinitesimal error of the initial state.

In order to understand the dilemma, it may be sufficient to know the fact that many universal arithmetic algorithms used to solve ODEs or PDEs are based on the opposite belief: using polynomial steps of iteration, the error should be polynomial. So it is almost impossible for these sort of algorithms to solve chaotic systems, though in principle, they are computable(Adding more space and waiting longer time(both exponentially)).

So it seems we can construct difficult problems taking advance of the hardness of chaotic physical system.

For instance, we could ask this question as follows:
Given an ODE(or PDE), which will become chaotic, and the initial condition, the duration t(binary representation), precision ϵ(binary representation), what is the configuration Ξ after t(satisfying $D(\tilde{\Xi}, \Xi) < \epsilon$, where $\tilde{\Xi}$ is what we get from any computing devices)?

Intuitively, this is a very hard question, as we have explained before.

On the one hand, from the point of view of complexity theory, at least no one can show it belong to the class PSPACE. Any computing devices with polynomial memory will fail to get the sufficient precision as long as they continue to use 'old' style algorithms(such as Euler's algorithm or RC and so on).

On the other hand, if there do exist **real** physical systems which evolve according to the differential equation, it seems we could construct the system and feed the system with desired initial condition, and after t, we shall naturally get the state. For example, may be the famous Lorenz equation[33] will play

the role.

$$\begin{cases} \dot{x} = -\sigma(x-y), \\ \dot{y} = x(r-z) - y, \\ \dot{z} = xy - bz, \end{cases}$$

Thus, physical computational system $\mathcal{P} = \langle \Delta, \mathfrak{L}, \nabla \rangle$ will be very efficient.

Where:

$$\mathfrak{L} \equiv \mathscr{L} \otimes \mathrm{T}$$

Where $\mathscr{L}$ implement the Lorenz system,$(x_0, y_0, z_0; \dot{x}_0, \dot{y}_0, \dot{z}_0)$is the input, T act as a timer and a trigger, when $\mathrm{T}.t = t$ Δ will be asked to serve its function.

Well, there are about 3 problems in this plan.

- Though there is little good algorithms for chaotic system, it does not mean we cannot find one forever.
- Chaotic systems can hardly be replayed. Because we cannot get perfect implementation of operator ∇, the experiment will not tell us a good result too. And the most crucial problem is:
- This is actually a pseudo-polynomial physical algorithm. Pay attention to the specification of the input. We are told to input the non-unary representation of the duration of the time t, which means the length of the input should be $\lceil \log t \rceil$. However, according to our design, we will wait t which is exponential to the length $\lceil \log t \rceil$.

3.11.2 Chaotic systems with *singularities*

In order to overcome the third problem, we can put singularities into a chaotic system. Singularities is something can help us to compress the infinite time of the evolution to a finite one and at the same time, it can preserve the topology of the trajectories. This idea is due to Smith[29].

Suppose there are N point-like particles, the i-th$(1 \leq i \leq N)$ particle has masses m_i, position $\mathbf{r}_i(t)$ at time t, which satisfy:

$$\ddot{\boldsymbol{r}}_i(t) = G \sum_{j \neq i} m_j \frac{\boldsymbol{r}_j(t) - \boldsymbol{r}_i(t)}{|\boldsymbol{r}_j(t) - \boldsymbol{r}_i(t)|^3}$$

Usually, we call the systems above N-body systems. And the corresponding computational task(often called N-body Problems) is: given the initial conditions(that is $(r_1(0), \ldots, r_N(0), r_1'(0), \ldots, r_N'(0),)$), and $t > 0$, calculate the N positions $r_i(t)$.

We say the solution runs into a singularity at time t_0, if the solution is not analytic at $t = t_0$. The singularities due to collisions seem to be the most usual and natural ones among all sorts of singularities, though, there do exist other types of singularities which are non-trivial.

In 1895, Painlevé[26] proved that for $N = 3$ the singularities have to be due to collisions and made the following conjecture.

Painlevé's Conjecture: For $N > 3$, there exist non-collision singularities.

When $N > 4$ Xia[25] and Gerver[24](independently) both proved that Painlevé's Conjecture is true. However the case when $N = 4$ still remain open. Their results are both constructive, which implies that they explicitly found an initial conditions but not reduction to absurdity. In their construction, *the system can exchange its gravitational potential energy for kinetic energy at a geometrically faster rate. The speed of the particles increases so rapidly that at some finite time t_0, the particles go to (real)infinity.*

Inspired by this 'supernatural' properties of the singularity, Smith get a design of his own. For any initial condition β, define the predicate $P(\beta)$ to be 1 if it leads to a non-collision singularity, and 0 otherwise(However, this can also need infinite precision, for any finite one of β will cause P being not robust).

So we assume that the set of singularities is of non-zero measure. Let $Q(\beta)$ denote the probability for $P(\beta')$ to be 1, if the input β' is randomly chosen from a ball of some fixed radius centered at β. Then $Q(\beta)$ is a continuous function that can be probabilistically computed by a gravitational system. Then, if the system is chaotic(hard to simulate), then it is usually hard for a Turing machine to decide.

For the first defect, we may try to construct a set of chaotic systems, which is recursively enumerable. If a property of this set $P(x)$ is undecidable, then we can conclude that there is no general method to avoid iterative style algorithms.

Chapter 4

Computability

In 1936, in order to study D.Hilbert's Entscheidungsproblem[57, 56](in fact, this problem can be traced back to one of Leibniz's great ideals to logic)which asks: *if there was a mechanical procedure for separating mathematical truths from mathematical falsehoods*, A.Church and Alan.Turing introduced their computational models respectively, namely, $\lambda-$ calculus[56] and Turing machine[31].

Definition 4.0.1. The alphabet of $\lambda-$calculus

- $\nabla = \{v, v', v'', v''' \dots\}$, ∇ is infinite.
- λ
- (,).

Definition 4.0.2. The set of $\lambda-$terms Λ is defined recursively as the minimal set which satisfies following condition:

- $x \in \nabla \Rightarrow x \in \Lambda$
- $M, N \in \Lambda \Rightarrow MN \in \Lambda$
- $M \in \Lambda, x \in \nabla \Rightarrow (\lambda xMN \in \Lambda)$.

Definition 4.0.3. ($\lambda-$definable) Suppose $k \in \mathbb{N}^+$, $f : \mathbb{N}^k \to \mathbb{N}$ is a k-variate arithmetic function. We say f is $\lambda-$definable if there exists $F \in \Lambda^\circ$(closed term, no free variables), s.t.

$$\forall n_1, \dots, n_k \in \mathbb{N}. F \ulcorner n_1 \urcorner \cdots \ulcorner n_k \urcorner =_\beta \ulcorner f(n_1, \dots, n_k) \urcorner,$$

where $\ulcorner n \urcorner \equiv \lambda fx.f^n x$, the meaning of $=_\beta$ could be found in many textbooks of $\lambda-$ calculi.

Definition 4.0.4. (Turing machine)Suppose set $D \subset \{0, 1\} \times \mathbb{N}$ is finite. Function $d : D \to \{0, 1\}$, $p : D \to \{-1, 0, 1\}$, $s : D \to \mathbb{N}^+$. A 3-tuple (d, p, s) is called a Turing machine, denoted as $M = (d, p, s)$.

Definition 4.0.5. (Turing computable)Suppose M is a Turing machine. $f : \mathbb{N}^n \to \mathbb{N}$ is a k-variate function. We say M computes f if for any $(x_1, \dots, x_n) \in \mathbb{N}^n$, with input $\overline{(x_1, \dots, x_n)}$ M outputs $\overline{f(x_1, \dots, x_n)}$. And we say f is *Turing computable*.

Their models are both mathematical definition of 'computation'(or 'algorithm'). Turing proved that $\lambda-$ calculus and Turing machines are equivalent in power of computation[61], that is, if a function could be computed by a Turing machine, it is definable on $\lambda-$ terms; and vise versa.

On the other hand, they proved that both these two computational models are not 'omnipotent', which could be understood as : of all arithmetic functions, there do exist functions which is not computable under their definitions[49][31]. Therefore, the answer to Entscheidungsproblem should be negative, if mathematicians accept Church-Turing Thesis.

In many years, people have invented various kinds of models to depict the intuitive idea 'computation' from different point of view and motivation(more natural, more concise, more safe, more effective,etc.)[60][59]. Unfortunately, none of them could compute the functions which have been proved to be not Turing computable. This actually makes more and more people believe that CT is correct, though CT is not a mathematical theorem[56] but rather a conjecture or postulate to a fact of the real world.

4.1 Turing computable is physical computable

The topic about the existence of a theoretical physical system which can provide an implementation of universal Turing machine has been studied by many scholars. In addition to the current implementation of computers, scholars have constructed many other wonderful designs on various axiom systems of physics(e.g. Classical Mechanics[18], Quantum Mechanics[5][3]).

Of course, the results above only imply that it is the ideal mathematic model for a family of physical phenomenons can be look as equivalent to UTM in terms of computability. After all, we cannot know for sure that some theory of physics is completely correct. Because of this, when we talk about the ability of computation for a certain family of physical system, we always assume either of the two preconditions:

- The ideal mathematic model of some branch of physics is believed to be absolutely right.
- At least in a very large scale, the theory works.

4.2 PLATO Machine

For several decades after the Church-Turing Thesis was proposed, people failed to find a counter-example. This kind of counter-example, if they really exist, should satisfies the property that most people think they can be effectively computed in principle, and no Turing machine can compute them.

However, many physicists tend to make efforts in another direction, that is, they want to find a family of processes in nature, whose functional expression

may not be intuitively computable, nor Turing Computable, but actually it can be used to 'compute' a nonrecursive function by measurement[29][30][6][8].

Suppose the problem we attempt to deal with now may cost infinite many steps for some computational model(e.g. Turing machine), does it necessarily mean that we have to wait infinitely long time to get the results? This is not always the case, PLATO Machine, which was proposed by H.Weyl[7], is just a counter-example. Though it is named after Plato, the designer's main inspiration comes from one of Zeno's Paradoxes.

Specifically, PLATO Machines use $(1/2)^n$ seconds to execute the $n-$th step. For instance, suppose the decision problem we want to solve is $\exists n P(n)$, where P is a predicate and $P(x)$ is used to describe some properties of x. Then PLATO machine $\mathbb{P}$ will check whether $P(1) = 1$ holds in $1/2$ seconds, and check whether $P(2) = 1$ in $1/4$ seconds,..., and check whether $P(n)$ holds in 2^{-n} seconds, and so on. It is easy to conclude that if $\mathbb{P}$ find an answer, it will return the answer in one second, otherwise it will return false after a second. Considering the sum of geometric series, the proof is trivial. So the upperbound of the time for $\mathbb{P}$ to solve any question is

$$T = \left(\frac{1}{2}\right)^1 + \left(\frac{1}{2}\right)^2 + \cdots + \left(\frac{1}{2}\right)^n + \cdots = \frac{\frac{1}{2}}{1-\frac{1}{2}} = 1s$$

Apparently, if $\mathbb{P}$ does exist, its power is extraordinarily great, for it can even solve Turing's Halting Problem in one second.

So far we have seen two ideas to implement the PLATO machine $\mathbb{P}$ in the real world. However, unfortunately, neither of them are successful. The first one is to construct the machine according to the definitions of H.Weyl. Apparently, it is difficult, for people do not believe that time is infinitely divisible. The second one is to make use of the theory of general relativity. However, the computing system will also exhaust the resource of the universe which make the observer cannot get the answer.

4.2.1 N-body System and PHCT

Making use of Gerver's results[24], Smith try to disprove CT just under the framework of classical mechanics[29]. However, strictly speaking the phrase 'Church Turing Thesis' here are not very appropriate.

4.2.1.1 classical mechanics

By the phrase 'classical mechanics' we mean the mechanical system evolve under some postulates[52][51] like:

- Newton's law:

$$\begin{cases} \Sigma F = 0 \Rightarrow \frac{dv}{dt}, & \text{the first law;} \\ F = m\frac{dv}{dt} = ma, & \text{the second law;} \\ \Sigma F_{a,b} = -\Sigma F_{b,a}, & \text{the third law.} \end{cases}$$

- or Lagrange equation:

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{q}_\alpha} - \frac{\partial T}{\partial q_\alpha} = Q_\alpha \quad (\alpha = 1, 2, \ldots, s)$$

- or Hamilton equation:

$$\begin{cases} \frac{\partial H}{\partial q_\alpha} = -\dot{p}_\alpha, \\ \frac{\partial H}{\partial p_\alpha} = \dot{q}_\alpha, (\alpha = 1, 2, \ldots, s) \end{cases}$$

Besides, law of universal gravitation is independent of above dynamic equations and should be looked as a fundamental postulate.

Also, classical mechanics should include other more fundamental postulates like Galilean transformation[52][51]:

$$\begin{aligned} x' &= x - vt \\ y' &= y \\ z' &= z \\ t' &= t \end{aligned}$$

Note that the last equation expresses the assumption of a universal time independent of the relative motion of different observers.

However, for our purpose even Galilean transformation is not the most fundamental postulate. Actually, first, one has to admit that the Euclidean spaces exist, where the desired topological properties hold and thus the concepts of 'limitation', 'derivatives' and 'integral'(calculus) can be legally defined and applied. For example, we can thus define velocity as

$$\lim_{\Delta t \to 0} \frac{\Delta \boldsymbol{r}}{\Delta t} = \frac{d\boldsymbol{r}}{dt} = \dot{r}$$

of course, when the limitation above is exist.

On the other hand, postulates for the ideal objects or behavior, should also be discussed. In classical mechanics, 'rigid body', 'point-like particle', 'light string', 'perfect elastic collision', etc are all legal. In fact, from the point of view of classical mechanics, objects with more complicated properties are constructed by means of such ideal objects.

4.2.1.2 Smith's idea

The variation of *Church-Turing Thesis* (or also called physical Church-Turing Thesis)is the belief that *if a function can be computed by any conceivable hardware system, then it can be computed by a Turing machine[29][62].*

Though it may seems to be very complicated, Smith's design can also be looked as a special construction of PLATO machine.

Note that the set of *computable reals* **CR** has a property that

$$\text{card}(\mathbf{CR}) \sim \text{card}(\text{TM}).$$

And in fact, even for a subset of **CR**, say, $\mathbf{CR}_2$, we also have

$$\text{card}(\mathbf{CR}_2) \sim \text{card}(\text{TM}).$$

where the $\mathbf{CR}_2$ is binary representation for **CR**, in which the elements are consist of '1's and '0's. Of course, any other polynomially equivalent types of representation will not affect the results.

In addition, the bijection between the two set is constructable. In fact, we can just map the whole transition of a Turing machine(with some finite input, of course) to a computable reals. Under this mapping, it is easy to see that the following property holds:

TM(x) ***halting*** *iff the corresponding computable real is a* ***finite sequence****.* By finite sequence, apparently we mean after some finite bit there are just '0's(for some real implementation, may be a fresh symbol denoting 'ending' is needed), and we do not try to combine some elements like

$$0.1\underbrace{00\ldots0\ldots}_{\text{infinite}} \text{ and } 0.0\underbrace{11\ldots1\ldots}_{\text{infinite}}$$

which means we just look them as different elements.

In his paper, Smith constructed a N-body system satisfying following statement:

- For any initial condition, if the system will go to singularity, the singularity should be non-collision singularity, which implies when go to singularity, all particles will go infinity at some finite time (say, t_0).
- For any initial condition, if the system will not go to singularity, then there must be some particles, say, $m(m > 0)$ which will pass the neighborhood of original point whose measure is not zero (also in finite time)(say, $t_1(t_1 < t_0)$).
- The types of the topology of the trajectories of the particles will arise by turns during $[0, t_0]$ and all of them can be looked as a sequence of the symbols of the types.

The first two properties indicate that we can physically construct a predicate Singular($*$), where

$$\text{Singular}(O(x)) = \begin{cases} 1, & \text{the system } O \text{ with initial condition } x \\ & \text{will go to singularity;} \\ 0, & \text{o.w.} \end{cases}$$

The third property actually inspired us to encode the sequence of the types of the interaction ϕ to real numbers such that the sequence of the interaction is finite iff the corresponding real number has finite non-zero bit.

Thus, we may appreciate the whole idea of Smith's construction as following:

$$\text{TM} \leftrightarrow \text{CR}_2 \leftrightarrow \Phi$$

And following relation holds:

$$(\forall x)(\forall m \in \text{TM})(\forall r \in \text{CR}_2)(\forall \phi \in \Phi)\Big(\text{Halt}(m(\bar{x}))\ \textbf{\textit{iff}}\ \text{Fin(r)}\ \textbf{\textit{iff}}\ \text{Fin}(\phi)\Big)$$

Furthermore, Smith explained that given a sequence ϕ, the initial condition x_0 can be recursively computed. This means that any finite prefix of ϕ's corresponding initial condition could be recursively computed. We suppose a Turing machine M_1 can do this work.

Therefore, for any $M_0 \in TM$, and any input $\bar{x}$, the formula $M_0(\bar{x})$ can be naturally looked as a finite description of a computable reals, which can be continuously be looked as a sequence of the types of the interaction. So $M_0(\bar{x})$ is the finite description ot the sequence ϕ. Thus the output of $\begin{pmatrix} M_1 \\ \downarrow \\ M_0 \end{pmatrix}(\bar{x})$ is the desired initial condition.

Note that the problem: does there exist a Turing machine M_2 satisfying following property,

$$\begin{pmatrix} M_2 \\ \downarrow \\ M_1 \\ \downarrow \\ M_0 \end{pmatrix}(\bar{x}) = \begin{cases} 1, & \text{if } M_0 \text{ halts;} \\ 0, & \text{o.w.} \end{cases}$$

is exactly the equivalent one to the famous Turing's halting problem, as there is no further constraint on machine M_0. We know for sure that the answer is negative: $\begin{pmatrix} M_2 \\ \downarrow \\ M_1 \end{pmatrix}$ does not exist; now that M_1 exists, thus M_2 shouldn't exist.

On the other hand, however, the system can used as an oracle. We have said that Smith's N-body system can provide a physical implementation of the

predicate Singular($*$). Therefore, with input

$$\llcorner \begin{pmatrix} M_1 \\ \downarrow \\ M_0 \end{pmatrix} (\bar{x}) \lrcorner,$$

which is a quantity depicted by a computable real number.

$$\text{Singular}(\llcorner \begin{pmatrix} M_1 \\ \downarrow \\ M_0 \end{pmatrix} (\bar{x}) \lrcorner) = \begin{cases} 1, & \text{if the system go to singularity;} \\ 0, & \text{o.w.} \end{cases}$$

Note that the correspondence between Turing machine, Computable reals and sequence talked before, we can immediately conclude that if the system go to singularity then the sequence ϕ is infinite, thus the computable reals only have finite non-zero bits, and in the end we are able to decide that $\text{Halt}(M_0) = 0$.

Now we talk about some potential problem of Smith's design. Smith's design could be written as

$$\lhd(\Delta \circ \mathcal{H} \circ \nabla)\ulcorner \#M \urcorner \ulcorner x \urcorner \rhd = \ulcorner \text{HALT}(\#M, x) \urcorner = \begin{cases} \ulcorner 1 \urcorner, & \text{if } M(x) \text{ halts;} \\ \ulcorner 0 \urcorner, & \text{o.w.} \end{cases}$$

Where

$$\begin{aligned} \nabla &\equiv \nabla_2 \circ \nabla_1 \\ \mathcal{H} &\equiv \ddot{\boldsymbol{r}}_i(t) = G \sum_{j \neq i} m_j \frac{\boldsymbol{r}_j(t) - \boldsymbol{r}_i(t)}{|\boldsymbol{r}_j(t) - \boldsymbol{r}_i(t)|^3} \\ \Delta &\equiv \text{Singular}() \end{aligned}$$

Where

$$\begin{aligned} \nabla_1 : &\quad \ulcorner \#M \urcorner \ulcorner x \urcorner \xrightarrow{\nabla_1} \llcorner M(\bar{x}) \lrcorner \\ \nabla_2 : &\quad \llcorner M(\bar{x}) \lrcorner \xrightarrow{\nabla_2} \llcorner \begin{pmatrix} M_1 \\ \downarrow \\ M \end{pmatrix} (\bar{x}) \lrcorner \end{aligned}$$

For the resource $\mathfrak{T}$, we have:

$$\begin{aligned} \mathfrak{T}(\Delta \circ \mathcal{H} \circ \nabla) &= \mathfrak{T}(\nabla) + \mathfrak{T}(\mathcal{H}) + \mathfrak{T}(\Delta) \\ &= \mathfrak{T}(\nabla_1 \circ \nabla_2) + \mathfrak{T}(\mathcal{H}) + \mathfrak{T}(\Delta) \\ &= \mathfrak{T}(\nabla_1) + \mathfrak{T}(\nabla_2) + \mathfrak{T}(\mathcal{H}) + \mathfrak{T}(\Delta) \\ &= \bowtie + \bowtie + \max(t_1, t_0) + \mathbb{C} \\ &= \bowtie \end{aligned}$$

where '$\bowtie$' means "Infimum unknown, infinity so far"and $\mathbb{C}$ means the term of constant.

Therefore, the complexity$_t$ $=\bowtie$.
For the resource $\mathfrak{S}$, we have:

$$\begin{aligned}\mathfrak{S}(\Delta \circ \mathcal{H} \circ \nabla) &= \mathfrak{S}(\nabla) + \mathfrak{S}(\mathcal{H}) + \mathfrak{S}(\Delta) \\ &= \mathfrak{S}(\nabla_1 \circ \nabla_2) + \mathfrak{S}(\mathcal{H}) + \mathfrak{S}(\Delta) \\ &= \mathfrak{S}(\nabla_1) + \mathfrak{S}(\nabla_2) + \mathfrak{S}(\mathcal{H}) + \mathfrak{S}(\Delta) \\ &= \bowtie + \bowtie + \infty + \mathbb{C} \\ &= \infty\end{aligned}$$

Therefore the complexity$_s$ $= \infty$.
For the resource $\mathfrak{E}$, we have:

$$\begin{aligned}\mathfrak{E}(\Delta \circ \mathcal{H} \circ \nabla) &= \mathfrak{E}(\nabla) + \mathfrak{E}(\mathcal{H}) + \mathfrak{E}(\Delta) \\ &= \mathfrak{E}(\nabla_1 \circ \nabla_2) + \mathfrak{E}(\mathcal{H}) + \mathfrak{E}(\Delta) \\ &= \mathfrak{E}(\nabla_1) + \mathfrak{E}(\nabla_2) + \mathfrak{E}(\mathcal{H}) + \mathfrak{E}(\Delta) \\ &= \bowtie + \bowtie + \mathbb{C} + \mathbb{C} \\ &= \bowtie\end{aligned}$$

Therefore the complexity$_e$ $=\bowtie$.
For the resource $\mathfrak{M}$, we have:

$$\begin{aligned}\mathfrak{M}(\Delta \circ \mathcal{H} \circ \nabla) &= \mathfrak{M}(\nabla) + \mathfrak{M}(\mathcal{H}) + \mathfrak{M}(\Delta) \\ &= \mathfrak{M}(\nabla_1 \circ \nabla_2) + \mathfrak{M}(\mathcal{H}) + \mathfrak{M}(\Delta) \\ &= \mathfrak{M}(\nabla_1) + \mathfrak{M}(\nabla_2) + \mathfrak{M}(\mathcal{H}) + \mathfrak{M}(\Delta) \\ &= \bowtie + \bowtie + \mathbb{C} + \mathbb{C} \\ &= \bowtie\end{aligned}$$

Therefore the complexity$_m$ $=\bowtie$.
We conclude that $\mathcal{C}(n) = (\bowtie, \bowtie, \infty, \bowtie)$.

4.2.2 Is N-body system too complex?

In fact, Smith's construction based on lots of beliefs which we do not know their correctness so far. For example, his construction ask the particle be an 'ideal' particle, which should be a strict point without any non-zero length or area, or rather, we could say that the diameter of the particle should be zero, if we look the particle as a point set(or $\delta(x_0, x)$).

Another belief which is quite crucial to Smith's construction is that computable reals can be somehow effectively prepared(with infinite precision). More exactly, we can map any computable real to a specific structure or state in Ω, we may call them computable objects(e.g. velocity, volume and so on). However, though computable reals have finite description, most of them have infinite de-

tails. Just being computable only enable us to recursively get a more refined approximation of them. Unfortunately, any finite precision would fail according to Smith's design. This tell us that as long as the precision is finite, there is no promise for the correctness of the results(not like other cases, more precision usually can get more correct bits or more probability for correctness).

Therefore, we can see that these postulates are very powerful. Completing the mapping above using finite resource usually means we have known many non-recursively decidable properties of those recursively enumerable object. To make it more clear, let's study a more simple design of our own.

Convention:

1. It is possible for us to produce strictly point like particle $\delta(x_0, x)$, using finite resource. Furthermore, we can manipulate it and detect it.

2. It is possible for us to map any computable reals to any computable object or state in Ω with finite precision, costing finite resource for each such reals.

The computable reals(binary) between 0 and 1 could be written as

$$\sum_{i=1}^{\infty} 2^{-i} b_i \quad (b_i \in \{0, 1\})$$

There are another set of structure called S, a tuple (B, F), where B is a sequence of the structure $b_i (i \in N)$ and F is a boolean indicating whether the sequence B is finite or not. The structure b is defined as follows.

$$b \in \{(l, r, color, h) | l \in \mathbb{R}^+, r \in \mathbb{R}^{+2}, color \in \text{Color} \equiv \{\text{black}, \text{white}\}, h \in \{T, F\}\}$$

where l indicate the length of a board(geometrically, the board only has length), r indicate the position of a board, the set Color has only two elements: black and white, it indicate the color of a board. The last boolean variable is used to indicate whether the board has a hole.

Technically, we can also let the hole as a diameter $d \in \mathbb{R}^+, d < l$.And if $h = F$, $d = 0$. it's position which means its center is the same as the board. So we can modify the definition b as:

$$b \in \{(l, r, color, h) | l \in \mathbb{R}^+, r \in \mathbb{R}^{+2}, color \in \text{Color}, h \in \{T, F\}, d \in (0, l)\}$$

For every $b_i \in B$, we require the following properties hold:

1. If $F = T$, then $\{b_i\}$ is finite. If the most footnote is j, then $h_j = T$, $d_j = (1/2) l_j$.

2. b_0 satisfies $l_0 = l, r_0 = (0, 0)$; b_i satisfies

$$l_i = \left(\frac{1}{2}\right)^i l$$

$$r_i = \left(\mathrm{rs}(i+1,2) r_{ix}^{odd} + \mathrm{rs}(i,2) r_{ix}^{even}, \mathrm{rs}(i,2) h \left(\frac{1}{2}\right)^{\frac{i+1}{2}} \right)$$

where

$$\theta_0 \in (\tfrac{\pi}{4}, \tfrac{\pi}{2}), \quad h \cot\theta_0 > l/2$$

$$\mathrm{rs}(x,y) \triangleq x \dot{-} y \left[\frac{x}{y}\right]$$

and

$$r_{ix}^{even} = 2hl \cot\theta_0 \left(1 - \left(\frac{1}{2}\right)^{\frac{i}{2}}\right)$$

$$r_{ix}^{odd} = r_{ix}^{even} - h \cot\theta_0 \left(\frac{1}{2}\right)^{\frac{i+1}{2}}$$

Then we design the mapping which send every $s \in S$ to $\mathbf{CR}_2$. For each $a \in \mathbf{CR}_2$, a could be written as

$$a = \sum_{i=1}^{\infty} 2^{-i} a_i, a_i \in \{0,1\}$$

We require that

$$color_0 = white; color_i = \begin{cases} black, & \text{if } a_i = 1; \\ white, & \text{o.w.} \end{cases}$$

However, if the computable number is finite, then $\{b_i\}$ is also finite.

Note that according to the convention, the mapping could be built in finite time and cost finite resource just need a description(description is always finite) of the number(e.g. Turing machine). Now let a point like particle (mass $m_0 > 0$), with velocity

$$v = v_0 \frac{(\cot\theta_0, \tan\theta_0)}{|\cot\theta_0, \tan\theta_0|}$$

leaves from the origin $(0,0)$. Suppose all collision are perfect elastic collision. In each case, if the given number has finite non-zero bits the particle will travel through the hole. Assume there are some field outside the s, then the particle will be push to the left most screen(according to the convention, anyway we can detect the particle).

So it's easy to find out that in both cases, we will know the 'answer' in finite time through the facility. In fact, if no particle hit the screen after a time t_0(note that t_0 is not just finite, it is constant with different given reals), then we can conclude that the given number has infinite non-zero bits(reals with infinite number bits will never enter the field, thus never come back). On the other hand, if the real has finite non-zero bits, it must hit the screen within t_0.

t_0 is consisted of two parts: the time used to travel through the structure s

t_s and the time used to travel through the field t_f. For t_s, we have

$$\begin{aligned} t_s &< \left(\frac{2h\cot\theta_0}{v_0\cos\theta_0}\right)\cdot\left(1+\left(\frac{1}{2}\right)+\left(\frac{1}{2}\right)^2+\cdots\left(\frac{1}{2}\right)^n+\cdots\right) \\ &= \left(\frac{4h\cot\theta_0}{v_0\cos\theta_0}\right) \end{aligned}$$

t_f satisfies the following inequality:

$$t_f < \sqrt{\frac{8h\cot\theta_0}{a}+\frac{v_0^2\cos^2\theta_0}{a^2}}$$

Totally, we have:

$$t_0 < \left(\frac{4h\cot\theta_0}{v_0\cos\theta_0}\right)+\sqrt{\frac{8h\cot\theta_0}{a}+\frac{v_0^2\cos^2\theta_0}{a^2}}$$

which means any t_0 larger than this bound should be large enough.

The whole experiment could be written as

$$\Delta\circ\mathcal{S}\circ\nabla$$

Where

$$\nabla : \overline{\#M(x)} \to s$$

Recalling the convention before, this operator only cost finite resource for any possible inputs. Thus we can suppose it is a general arithmetic function f. Note that f may not be a computable function, but it's adequate for the destination. Therefore, the whole progress will cost finite resource for any inputs.

$$\begin{aligned} &\mathfrak{T}(\Delta\circ\mathcal{S}\circ\nabla)(x) \\ =&\mathfrak{T}\Big(\Delta(\lhd\Big(S\big(\lhd\nabla(x)\rhd\big)\Big)\rhd\Big)+\mathfrak{T}\Big(S\big(\lhd\nabla(x)\rhd\big)\Big)+\mathfrak{T}(\nabla(x)) \\ =&\mathbb{C}+t_0+f \end{aligned}$$

The resource $\mathfrak{E}$ and $\mathfrak{M}$ have the similar results. It is easy to see that the total space we used is bounded by a constant, i.e.

$$\begin{aligned} &\mathfrak{S}(\Delta\circ\mathcal{S}\circ\nabla)(x) \\ =&\mathfrak{S}\Big(\Delta(\lhd\Big(S\big(\lhd\nabla(x)\rhd\big)\Big)\rhd\Big)+\mathfrak{S}\Big(S\big(\lhd\nabla(x)\rhd\big)\Big)+\mathfrak{S}(\nabla(x)) \\ =&\mathbb{C}+\mathbb{C}+\mathbb{C} \\ =&\mathbb{C} \end{aligned}$$

We construct a 'supercomputer' or a real PLATO machine, which can solve any problem in finite time, just under the framework of classical mechanics even

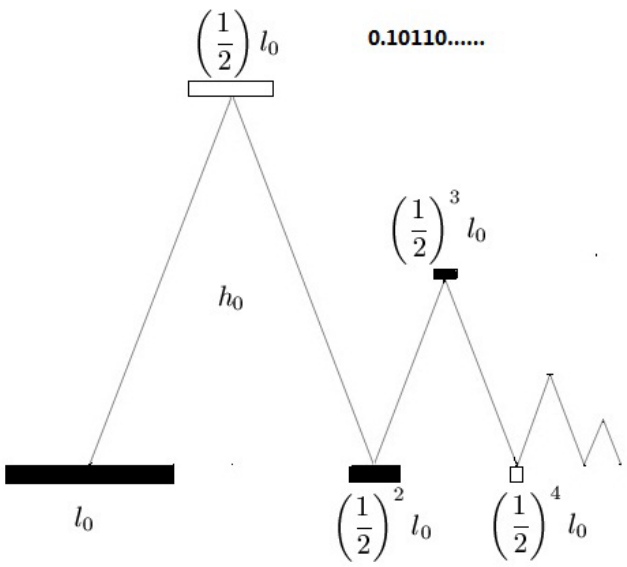


Figure 4.1: A physical implementation of Plato Machine

without the introduce of non-collision singularity.

With the finite input, classical mechanics and universal Turing machine may be equivalent in computational power. If any infinite inputs are thought to be legal, classical mechanics and universal Turing machine are also equivalent. Our results actually show that given family of recursively enumerable oracles with finite descriptions(may be infinite in details), respectively legal or existent , then classical mechanics is superior to universal Turing machine. Given infinite tapes of all transitions of each Turing machine, whose description is finite, universal Turing machine still cannot decide undecidable problem, while just as we discussed above, classical mechanics can use such oracles to construct PLATO machine and thus surpass the universal Turing machine. In fact, infinite tapes with finite description means nothing to universal Turing machine, it cannot visit infinite checks in finite time. So at this time, we can say, let $\mathcal{TM}(\mathcal{CM})$ denote the set of Turing computable functions(classical mechanics computable functions):

$$\left(\{[\mathcal{O}_\infty^\infty]\}_{\mathrm{CM}} \cup \mathcal{CM}\right) = \left(\{[\mathcal{O}_\infty^\infty]\}_{\mathrm{TM}} \cup \mathcal{TM}\right)$$

Where the superscript of $\mathcal{O}$ is the cardinality of $\mathcal{O}$'s extension, the subscript of $\mathcal{O}$ is the cardinality of $\mathcal{O}$'s intension. $[x]_{CM}$ ($[x]_{TM}$) indicates that x is legal in classical mechanics(is in Turing's tape form)

And there exist a family of $[\mathcal{O}_m^\infty]_{\mathcal{CM}}(m < \infty)$, s.t. for all $\{[\mathcal{O}_n^\infty]\}_{\mathrm{TM}}$, $n < \infty$

$$\left(\{[\mathcal{O}_m^\infty]\}_{\mathrm{CM}} \cup \mathcal{CM}\right) \supset \left(\{[\mathcal{O}_n^\infty]\}_{\mathrm{TM}} \cup \mathcal{TM}\right)$$

However, in fact we could combine infinite such oracles into one, which can also be described in finite symbol and cost finite resource, the conclusion may be written as a more concise one:

$$\exists m \in \mathbb{N} \forall n \in \mathbb{N} \big([\mathcal{O}_m^\infty]_{\mathrm{CM}} \cup \mathcal{CM} \big) \supset \big([\mathcal{O}_n^\infty]_{\mathrm{TM}} \cup \mathcal{TM} \big)$$

while the more fundamental question about whether we have

$$\mathcal{CM} = \mathcal{TM}$$

is still open.

By introducing magnetic field and accordingly it's basic postulates, one can actually construct another physical oracle with finite description in classical electromagnetic theory without introduce of collision(because real collision cost time after all). Though the magnetic intensity seems is divergent, but cleverly selected shape(mainly, the volume) will still cost only finite energy.

4.3 Recursive function whose derivative is not recursive

April 1970, J.Myhill published his astonishing result[6]: There exists a recursive function, whose derivative is not recursive. In order to understand the principles of the construction, knowing the following fact about the recursive functions(whose domain is $\mathbb{R}$) should be helpful.

Theorem 4.3.1. *Suppose f is a real-valued function, $\{f_n\}$ is a series of recursive functions, if there exists a recursive function $e : \mathbb{N} \to \mathbb{N}$ s.t. $\forall x \in I, k \geq e(n)$ $|f_k(x) - f(x)| \leq \frac{1}{2^n}$, then f is recursively computable.*

J.Myhill's idea is to build a non-trivial structure(slope or bump) in the neighborhood of 2^{-n} in interval $[0, 1]$, where $n \in \mathscr{A}$, and $\mathscr{A}$ is a recursively enumerable, nonrecursive set. Otherwise $f(x) = 0$. However, in order to make the function computable, the scale of the structure should shrink as the n is enumerated recursively, or rather, should be smaller than the bound in the theorem above. As a result, the derivative of the function is intuitively hard to compute, and on the other hand we can proof that it is not recursive, because if we could compute it we can use the result to decide whether $\lceil x \rceil$ is an element of $\mathscr{A}$ generally, contradicting the nonrecursiveness of $\mathscr{A}$.

Specifically, suppose

$$\theta(x) \equiv \begin{cases} x(x^2-1)^2, & \text{if } -1 \leq x \leq 1; \\ 0, & \text{if } |x| > 1. \end{cases}$$

It is easy to verify that $\theta(-1) = \theta(0) = \theta(1) = 0, \theta'(-1) = \theta'(1) = 0, \theta'(0) = 1$,and $\theta_{min} = \theta(-1/\sqrt{5}) \equiv -\lambda, \theta_{max} = \theta(+1\sqrt{5}) \equiv +\lambda$. We call θ a *bump* of

length 2 and height λ. Then the function $\theta_{\alpha\beta}(x) \equiv (\beta/\lambda)\theta(x/\alpha)$ satisfies the following conditions:

$$\theta_{\alpha\beta}(-\alpha) = \theta_{\alpha\beta}(0) = \theta_{\alpha\beta}(\alpha) = 0, \quad \theta'_{\alpha\beta}(-\alpha) = \theta'_{\alpha\beta}(\alpha) = 0, \quad \theta'_{\alpha\beta}(0) = \theta/\lambda\alpha,$$

$$-\beta \leq \theta_{\alpha\beta}(x) \leq \beta \quad (\alpha \leq x \leq \alpha.)$$

For each $n \in \mathscr{A}$,we shall construct a*bump*: $\theta_{\alpha_n\beta_n}$ at 2^{-n} i.e.

if $n \in \mathscr{A}$, $\delta \in [-\alpha_n, +\alpha_n]$, $f(2^{-n} + \delta) \equiv \theta_{\alpha_n\beta_n}(\delta)$, otherwise $f(x) \equiv 0$. To make f well-defined, parameters $\alpha_n, \beta_n, n \in \mathscr{A}$ is defined as

$$\alpha \equiv 2^{-k-2n-2}, \quad \beta_n \equiv 2^{-k-n-2},$$

where $n = h(k)$ and h is a function enumerating $\mathscr{A}$ without repetitions(It is easy to proof that if there exists a recursive function enumerating $\mathscr{A}$, then there exists such function with no repetitions).

For physicists, does J.Myhill's results imply that if an object move under the condition that the displacement and the time satisfies the following relations

$$\boldsymbol{r}(t) = f(t) = \begin{cases} \theta_{\alpha_n\beta_n}(\delta)(n \in \mathscr{A}), & \text{if } t = 2^{-n} + \delta, \quad \delta \in [\alpha_n, +\alpha_n]; \\ 0, & \text{o.w.} \end{cases}$$

The speed $\boldsymbol{v} \equiv \boldsymbol{r}'(t)$ will be a physical quantity which is not computable?

It is easy to see that the operator ∇ for the desired construction is very difficult to implement. Again, any finite approximation here is useless. However, suppose anyway we realize the operator ∇, the operator Δ also seems impossible: A bound for finding a non-zero derivative is not Turing computable, therefore, at least a naïve brute force method will fail. According to Hempel's axiom[19] (which means brute force is almost the only method for us)for measure, our problem is actually equivalent to the original Turing's Halting Problem. Now we don't know which physical will help us this time.

4.4 Physical States which is not computable

Pour-El et al published their results in 1997: for a differential equation, one can design a specific initial state to make the solution after t (t could be take some computable value)seconds is nowhere computable[8].

Consider the IVP of the following wave equation:

$$\begin{cases} \frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 x}{\partial x^2} + \frac{\partial^2 y}{\partial y^2} + \frac{\partial^2 z}{\partial z^2}, \\ u(x, y, z, 0) = f(x, y, z), \frac{\partial u}{\partial t}(x, y, z, 0) = 0 \quad . \end{cases}$$

where $(x, y, z) \in \mathbb{R}^3$, $\quad t \in [\ 0, +\infty\ )$ for all $f \in \mathscr{C}^1$ this IVP has a form of

solution known as Kirchhoff's formula:

$$u(\overrightarrow{x},t)=\iint_{S^2}[f(\overrightarrow{x}+tn)+t\nabla f(\overrightarrow{x}+t\overrightarrow{n})\cdot\overrightarrow{n}]d\sigma(\overrightarrow{n})$$

The conclusion Pour-El get is the following theorem:

Theorem 4.4.1. *For all compact set $D\subset\mathbb{R}^3\times[\ 0,\infty\ )$, there exists a computable function $f(x,y,z)\in\mathscr{C}^1$, s.t. the corresponding solution $u(\overrightarrow{x},t)$ is not computable in the neighborhood of any point in D.*

Pour-El et al construct the initial value through the non-computable real number: $\sum_{i=0}^{\infty}\frac{1}{2^{a(i)}},\quad a(i)\in\mathscr{A}$.

Apparently, one can conclude that in this wave equation, the initial state is computable but the state $u(0,0,0,1)$ is a state which can not be compute.

For us, can we safely conclude that

$$\{\text{Turing Computable}\}\subset\{\text{Physical Computable}\}$$

but

$$\{\text{Turing Computable}\}\not\supset\{\text{Physical Computable}\}?$$

Just like the design of Myhill, Pour-EL's construction also need a design of ∇, the initial value is far more trivial. A good news for us is that suppose anyway we implement the ∇, the final state is measurable in principle! That is to say: the value of that quantity could be measured progressively digit by digit. The method is actually based on the idea of amplification which we will discussed in details later.

4.5 A few Comments

In the above scenario, the use of (actual) infinity is their common theme. They ask the system to run for infinite steps or just encode the solutions into real numbers. It is easy to find out that adding either of these two presumption into a physical system will make the original system extraordinarily powerful.

4.5.1 Examples in Quantum Mechanics

For example, we can throw a particle onto a plane $[0,1]\times[0,1]$ at random(obey the uniform distribution), then we can proof that with high probability, the x-coordinate(or y-coordinate) of the center of the particle will indicate a non-recursive real number. In fact, in cell $[0,1]\times[0,1]$, the Lebesgue measurement for the recursive real numbers is 0, while the rest is 1, i.e.

$$m([0,1]\times[0,1]\cap\mathbb{R}_r)=0, m([0,1]\times[0,1]\cap\mathbb{R}_r^c)=1$$

This is geometric probability and consider the uniform distribution, the probability of the either event of the two are just their measurement. Therefore we can look the $x-$coordinate as a function with respect to the digits. According to Beggs et al, a theoretical machine called SME may help us to get the value of the position coordinates[20][21][22].

4.5.2 Measure Reals

Here, we give another stepwise method to measure a true real number, our method is more realistic than Begg's.

Perhaps the most commonly used thoughts of measuring physical quantity(expressed by real numbers) is try to enlarge the effects related to them. Note that a point could be determined by two lines, if we randomly shoot two segment(instead of just a point-like particle) towards a screen we could compute the point determined by this two segment. Also, of course, the physical implementation of segment is not perfect, it may have the dimension 'width' in addition to 'length'. Without loss of generality, we assume that each segment is actually a set of points. By sampling the point in each segment, people can use various numerical algorithms(e.g.least squares method) to acquire the equation whose image is an approximation of the line. To acquire a better approximation, that is, to get more and more digits of the number, we should amplify the whole image. Though the precision of the sampling is the same, amplification enable the experimenter to sample more points. Thus the results could be refined progressively.

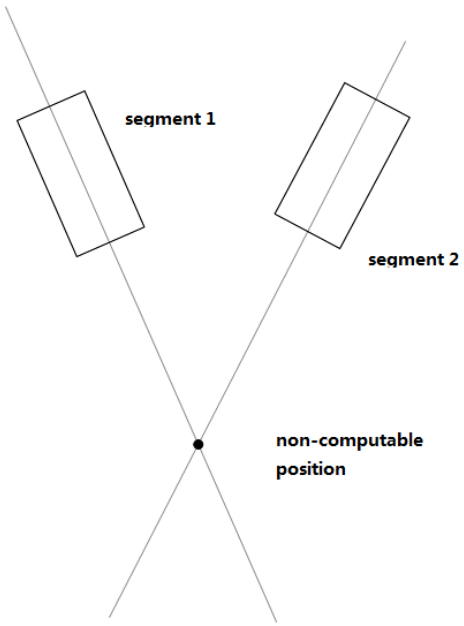


Figure 4.2:

People might argue that the rectangle shape is hard to find in microscopic scale, well, we could use another plan which need 4 times shooting quantum particles with round profile.

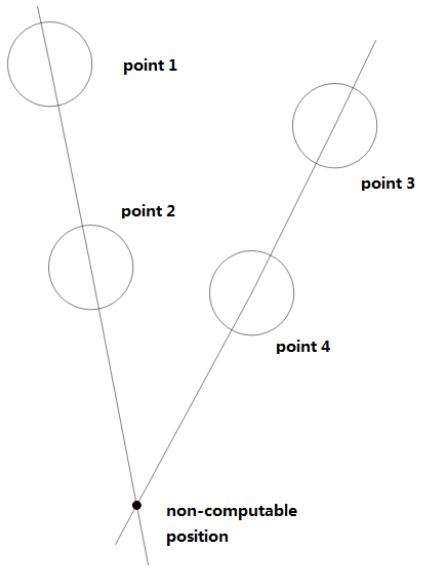


Figure 4.3:

Fortunately, amplifying an image that is thought to be easily to implement. There are many devices which can complete this job. For simplicity, suppose there is a point-like light source behind the screen and the screen is replaced by a transparent one. On the other side of the transparent screen, we put an opaque screen. Assume that the distance between the light source and the transparent screen is d, the distance between the transparent screen and the opaque screen is $-9d$ at first time. After turning on the light source, we could get the 10 times shadow of the original image. If we move the opaque screen to the position $-99d$, a 100 times shadow of the original image will be displayed on the opaque screen. And so on, if the distance is $(10^i - 1)d$, the opaque screen will show the shadow which is 10^i times the size of original image.

However, does the strict plane really exist in the physical world? We just do not know.

4.5.3 Existence

Now we give some levels for f which is not computable. Suppose o is an operator, $\hat{o}$ is a physical implementation of o and operators Δ , ∇ always mathematically exist. For $f \in F$ where all elements in F are non-recursive functions, we have the level of existence as follows (Assume here the representations are all well-formed or para well-formed).

- **Existence-I**

$\exists\hat{p}\forall x(\lhd(\Delta\circ\hat{p}\circ\nabla)(\ulcorner x\urcorner)\rhd\doteq\ulcorner f(x)\urcorner)$

- **Existence-I***
  $\exists\hat{p}\exists\hat{\nabla}\exists\hat{\Delta}\forall x((f\in F)\wedge(\Pr\{\lhd(\hat{\Delta}\circ\hat{p}\circ\hat{\nabla})(\ulcorner x\urcorner)\rhd\doteq\ulcorner f'(x)\urcorner|f'\in F\}>0))$
- **Existence-II**
  $\exists\hat{\Delta}\exists\hat{p}\forall x(\lhd(\hat{\Delta}\circ\hat{p}\circ\nabla)(\ulcorner x\urcorner)\rhd\doteq\ulcorner f(x)\urcorner)$
- **Existence-II***
  $\exists\hat{p}\exists\hat{\nabla}\forall x(\lhd(\Delta\circ\hat{p}\circ\hat{\nabla})(\ulcorner x\urcorner)\rhd\doteq\ulcorner f(x)\urcorner)$
- **Existence-III**
  $\exists\hat{p}\exists\hat{\Delta}\exists\hat{\nabla}\forall x(\lhd(\hat{\Delta}\circ\hat{p}\circ\hat{\nabla})(\ulcorner x\urcorner)\rhd\doteq\ulcorner f(x)\urcorner)$

Some more weak class are omitted here. Apparently, we have

$$\textbf{Existence}-\textbf{I}\supset\textbf{Existence}-\textbf{I}^*\supset\textbf{Existence}-\textbf{II}\supset\textbf{Existence}-\textbf{III}$$

$$\textbf{Existence}-\textbf{I}\supset\textbf{Existence}-\textbf{I}^*\supset\textbf{Existence}-\textbf{II}^*\supset\textbf{Existence}-\textbf{III}$$

According to the levels we proposed above, assuming the space is continuous, we can find out that $\mathbb{P}\in$ Existence-II, J.Myhill's function $f\in$ Existence-I, Pour-El's construction $\phi\in$ Existence-II, our example is in Existence-I*. Of course, we wish to find an example in Existence-III.

Chapter 5

Quantum Computation

5.1 Quantum Algorithms

For general quantum computation, we only need to explain the definition of the physical state set and the required evolution operators. More over, we only talk about Monte Carlo styled quantum algorithms.

According to von Neumann's four postulates[18] for quantum mechanics, we require that the state of any representation should be vectors in Hilbert space,i.e.

$$\Omega \subset \mathbb{H}$$

and the evolution operators should be unitary, i.e.

$$\mathcal{H}^{\dagger}\mathcal{H} = \mathcal{H}\mathcal{H}^{\dagger} = I.$$

Without loss of generality, we can assume that the measurement operators is projection operators(POVM could be substituted by projection operators through adding more auxiliary qubits)

Our framework for quantum mechanics is more extensive than quantum Turing machine and quantum circuit model. This quantum computational model consider uncountable or countable unitary operators while other models only talk about finite Hilbert space and the unitary operators could be represented as matrix. These differences may be insignificant in quantum complexity theory. However, in computability theory, our model is quite different from the ones before. The example in chapter IV implies that our model is a little more powerful than Turing machine and quantum Turing machine(or quantum circuit model), while quantum Turing machine(or quantum circuit model) have been proved equivalent to Turing machine.

5.1.1 Quantum Computability and Quantum Complexity

The resource cost by a computation is

$$\begin{aligned}\mathfrak{R}_{\mathscr{P}}((\Delta\circ\mathcal{H}\circ\nabla)(\ulcorner x\urcorner)) &= (\mathfrak{T}((\Delta\circ\mathcal{H}\circ\nabla)(\ulcorner x\urcorner)),\\ &\quad \mathfrak{S}((\Delta\circ\mathcal{H}\circ\nabla)(\ulcorner x\urcorner)),\\ &\quad \mathfrak{E}((\Delta\circ\mathcal{H}\circ\nabla)(\ulcorner x\urcorner)),\\ &\quad \mathfrak{G}((\Delta\circ\mathcal{H}\circ\nabla)(\ulcorner x\urcorner))\end{aligned}$$

Our definition here is special a case of the one in chapter-III. Suppose our discussion is restricted to QCM, i.e. we have finite kinds of universal quantum operators, then the number of gates used and the depth of the whole circuit will be the main parameter which should be took into account. It is easy to find out that this definition is similar to that of quantum circuit model. One of the difference between them is that we will also take the cost of design(usually this costs time) of a new circuit into account. Though in most cases, this will not cause great difference from the result given by QCM, however, we don't think we can safely ignore the potential exceptions just because it is usually easy to expand the scale of some circuits.

So far, people always assume that qubit is relatively easy to prepared. At least in the asymptotic sense, no matter how difficult to prepared a quantum bit, the cost should be bounded by a constant. We will also do this.

5.1.2 Deutsch-Josza Algorithm

Deutsch-Josza algorithm is one of the most successful algorithms in the early years. The corresponding problem of the algorithm is: consider two sets of functions:

A: $\left\{\varphi|\varphi:\{0,\ldots,2^n-1\}\to\{0,1\},\text{ and }\forall x(\varphi(x)=0)\right\}$

B: $\left\{\varphi|\varphi:\{0,\ldots,2^n-1\}\to\{0,1\},\text{and }\left|\{x|\varphi(x)=0\}\right|=\left|\{x|\varphi(x)=1\}\right|\right\}$

Apparently we have $A\cap B=\emptyset$, now suppose $f\in A\cup B$ and there is an oracle to compute f. We are required to decide whether $f\in A$ or not. It is no doubt that people wish to reduce the times of query the oracle as much as possible.

Note that the cost of constructing the oracle is not taken into account, because we assume some others have implemented it.

The algorithm needs a trivial input $\psi_0=|0\rangle^{\otimes n}|1\rangle$, and used the gate $H^{\otimes n}\otimes H$ onto the stateψ_0 a nd get ψ_1, i.e.

$$\psi_1=\left(H^{\otimes n}\otimes H\right)\left(|0\rangle^{\otimes n}|1\rangle\right)$$

Note that $H = \frac{1}{\sqrt{2}}\Big((|0\rangle + |1\rangle)\langle 0| + (|0\rangle - |1\rangle)\langle 1|\Big)$. By induction we have

$$H^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{x,y} (-1)^{x \cdot y} |x\rangle\langle y|$$

where i.e. $x \cdot y \equiv \bigoplus_i x_i \wedge y_i$. So we get:

$$\begin{aligned}
\psi_1 &= \Big(H^{\otimes n} \otimes H\Big)\left(|0\rangle^{\otimes n}|1\rangle\right) \\
&= \tfrac{1}{\sqrt{2^n}}\Big(\sum_{x,y}(-1)^{x\cdot y}|x\rangle\langle y|\Big)|0\rangle^{\otimes n}\left[\tfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right] \text{(by orthogonality)} \\
&= \tfrac{1}{\sqrt{2^n}}\sum_{x}(-1)^{0}|x\rangle\left[\tfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right] \\
&= \tfrac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle\left[\tfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right]
\end{aligned}$$

Now use the oracle $U_f : |x, y\rangle \to |x, y \oplus f(x)\rangle$ onto the state ψ_1 to get ψ_2

$$\begin{aligned}
\psi_2 &= U_f\left(\tfrac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle\left[\tfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right]\right) \\
&= \tfrac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle\Big(f(x)\oplus\tfrac{|0\rangle-|1\rangle}{\sqrt{2}}\Big) \\
&= \tfrac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}(-1)^{f(x)}|x\rangle\left[\tfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right]
\end{aligned}$$

At last we use $H^{\otimes n} \otimes I$ onto $\psi_2$to getψ_3:

$$\begin{aligned}
\psi_3 &= \Big(H^{\otimes n}\otimes I\Big)\left(\tfrac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}(-1)^{f(x)}|x\rangle\left[\tfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right]\right) \\
&= \sum_z\sum_x \tfrac{(-1)^{x\cdot z+f(x)}|z\rangle}{2^n}\left[\tfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right]
\end{aligned}$$

The observer is supposed to check the first n qubits, note that the amplitude of $|0\rangle^{\otimes n}$ is $\sum_x (-1)^{f(x)}/2^n$. If $f \in A$, $f(x)$ is constant and the amplitude of $|0\rangle^{\otimes n}$ is $+1$ or -1. So the amplitude of another cases should be zero and the observer will get $|0\rangle^{\otimes n}$. On the other hand, if $f \in B$, the amplitude of $|0\rangle^{\otimes n}$ will be zero. So the observer will always get a non-zero vector.

In our opinion, the procedure could be written as follows.

$$\begin{aligned}
\nabla &\equiv \text{Initialize the state}|0\rangle^{\otimes n} \otimes |1\rangle \\
&\quad\ \text{Generate the whole circuit} \\
\mathcal{H}_1 &\equiv H^{\otimes n} \otimes H \\
\mathcal{H}_2 &\equiv U_f \\
\mathcal{H}_3 &\equiv H^{\otimes n} \otimes I \\
\Delta &\equiv \sum_i |P_i\rangle\langle P_i|
\end{aligned}$$

Let $\mathcal{H} = \mathcal{H}_3 \circ \mathcal{H}_2 \circ \mathcal{H}_1$
$(\Delta \circ \mathcal{H} \circ \nabla)(|x\rangle) = \Big(\Delta \circ \mathcal{H}_3 \circ \mathcal{H}_2 \circ \mathcal{H}_1 \circ \nabla\Big)(|x\rangle) = P(f \in B)$

Though Deutsch-Jozsa Algorithm is great, someone still think it is not very useful. In addition to the fact that the problem they studied is not very important, there does exist an efficient classical probabilistic algorithm to solve the problem with high probability.

5.1.3 Grover's Algorithm

Quantum Search Algorithm[13], also known as Grover's Algorithm, is another quite successful quantum algorithm. Though this algorithm is not faster than the fastest classical search algorithms super-polynomially, one can proof it is the fastest one considering quantum mechanics. Therefore, the complexity of the algorithm is the complexity of the problem it deals with.

The crucial subroutine of Grover's Algorithm is the Grover iteration, often denoted by G:

- Apply Oracle $O : |x\rangle|-\rangle \to (-1)^{f(x)}|x\rangle|-\rangle$;
- Apply Hadamard Gates:$H^{\otimes n}$;
- Perform a conditional phase shift$(2|0\rangle\langle 0| - I)$ on the computer, with every non-zero bases receiving a phase shift of -1.
- Perform Hadamard transformation $H^{\otimes n}$.

Note that $H^{\otimes n}(2|0\rangle\langle 0| - I)H^{\otimes n} = 2|\psi\rangle\langle\psi| - I$ One can proof that Grover iteration can be looked as a rotation in the plane spanned vectors which denoted the right answers and the wrong answers.

Let Σ'_x be the sum of all the vectors which indicate a solution to the search problem, Σ''_x the rest. Define normalized states:

$$|\alpha\rangle \equiv \frac{1}{\sqrt{N-M}}\Sigma''_x|x\rangle$$

$$|\beta\rangle \equiv \frac{1}{\sqrt{M}}\Sigma'_x|x\rangle$$

thus the initial state $|\psi\rangle = \frac{1}{N^{1/2}}\Sigma_{x=0}^{N-1}|x\rangle$ could be represented as

$$|\psi\rangle = \sqrt{\frac{N-M}{N}}|\alpha\rangle + \sqrt{\frac{M}{N}}|\beta\rangle$$

The action of Operator O is $O(a|\alpha\rangle + b|\beta\rangle) = a|\alpha\rangle - b|\beta\rangle$, which could be looked as perform a reflection in $\alpha\beta-$ plane. Similarly Operator $2|\psi\rangle\langle\psi| - I$ also performs a reflection in $\alpha\beta-$plane. Thus both two reflections which could be looked as a rotation occur in the $\alpha\beta-$plane. Let $\cos\theta/2 = \sqrt{(N-M)/N}$, s.t.$|\psi\rangle = \cos\theta/2|\alpha\rangle + \sin\theta/2|\beta\rangle$, apply the iteration once makes $|\psi\rangle$ become

$$G|\psi\rangle = \cos\frac{3\theta}{2}|\alpha\rangle + \sin\frac{3\theta}{2}|\beta\rangle$$

k times use of Grover's Iteration will lead to the following result:

$$G^k|\psi\rangle = \cos\left(\frac{2k+1}{2}\theta\right)|\alpha\rangle + \sin\left(\frac{2k+1}{2}\theta\right)|\beta\rangle$$

Since $|\psi\rangle = \sqrt{(N-M)/N}|\alpha\rangle + \sqrt{M/N}|\beta\rangle$, we just need to rotate $|\psi\rangle$ $\arccos\sqrt{M/N}$ radians to the one which is parallel to vector $|\beta\rangle$. So repeating G for $R = [\frac{\arccos\sqrt{M/N}}{\theta}]$ times will get $|\psi\rangle$ to within an angle $\theta/2 \leq \pi/4$ of $|\beta\rangle$. This is a 'good' state, for people only have to repeat the experiment for expected constant times to get the solution to the problem(Consider geometric probability distribution: $\mathrm{E}[\![X]\!] = 1/(1/2) = 2$).

Apparently $R \leq \lceil \pi/2\theta \rceil$, suppose $M \leq N/2$ then we have $\frac{\theta}{2} \geq \sin\frac{\theta}{2} = \sqrt{\frac{M}{N}}$. Thus, we obtain:

$$R \leq \left\lceil \frac{\pi}{4}\sqrt{\frac{N}{M}} \right\rceil$$

in other words we need repeat G for $R = O(\sqrt{N/M})$ times.

In our framework, Grover's Algorithm could be expressed as:

$$\nabla \equiv \text{initialize } |0\rangle^{\otimes n} \otimes |0\rangle^{\otimes o}$$

the number o depends on the implementation of the oracle O.

$$\mathcal{H} \equiv \mathcal{H}'^{O(\sqrt{N})}$$

where

$$\mathcal{H}' \equiv (2|\psi\rangle\langle\psi| - I)O)$$

$$= H^{\otimes n} \otimes I^{\otimes o} \circ (2|0\rangle\langle 0| - I) \otimes I^{\otimes o} \circ H^{\otimes n} \otimes I^{\otimes o} \circ O$$

and

$$\psi = \frac{1}{N^{1/2}} \sum_{x=0}^{N-1} |x\rangle$$

$$\Delta \equiv \sum |P_i\rangle\langle P_i|$$

In Grover's scenes, just like the case in Deutsch's algorithm, we never need to consider any complexity cost by constructing the oracle. $H^{\otimes n}$ costs $n = \log(N)$ operations and $2|0\rangle\langle 0| - I$ costs $O(n)$ operations. Therefore, the whole procedure costs about $O(\sqrt{N}\log(N))$ operations which is superior to the lower bound for Turing machines which is $O(N)$.

5.1.4 Shor's Algorithm

Shor's Algorithms for prime factorization and discrete logarithms[10, 11] is so far the most exciting quantum algorithms. The appearance of Shor's Algorithms is the greatest challenge to strong Church Turing Thesis.

Shor's Algorithms depends on a technique of so called "quantum Fourier Transform". But of course QFT is not enough. Shor's Algorithm is totally non-trivial and marvelous, and few people can produce any algorithms like that easily.

In order to understand Shor's Algorithm, it may be enough to gain a clear idea of quantum ordering algorithm. This is the only subprogram in the Shor's Algorithm which has to be implemented by quantum computers so far, and it is really the most important subprogram.

First, note that

$$\sum_{s=0}^{r-1} \exp(-2\pi isk/r) = r\delta_{k0}$$

and define $|u_s\rangle$ as follows

$$|u_s\rangle \triangleq \frac{1}{\sqrt{r}} \sum_{k'=0}^{r-1} e^{-2\pi isk'/r} \left| x^{k'} \mathrm{mod} N \right\rangle$$

According to the fact above, we can get

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{2\pi isk/r} |u_s\rangle = \left| x^k \mathrm{mod}\ N \right\rangle$$

In fact

$$\begin{aligned}
\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{2\pi isk/r} |u_s\rangle =& \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} \left(e^{2\pi isk/r} \frac{1}{\sqrt{r}} \sum_{k'=0}^{r-1} e^{-2\pi isk'/r} \left| x^{k'} \mathrm{mod} N \right\rangle \right) \\
=& \frac{1}{r} \sum_{s=0}^{r-1} \left(e^{2\pi isk/r} \sum_{k'=0}^{r-1} e^{-2\pi isk'/r} \left| x^{k'} \mathrm{mod} N \right\rangle \right) \\
=& \frac{1}{r} \sum_{k'=0}^{r-1} \sum_{s=0}^{r-1} \exp\left(\frac{2\pi is(k-k')}{r} \right) \left| x^{k'} \mathrm{mod} N \right\rangle \\
=& \frac{1}{r} \sum_{k'=0}^{r-1} r\delta_{kk'} \left| x^{k'} \mathrm{mod} N \right\rangle \\
=& \left| x^k \mathrm{mod}\ N \right\rangle
\end{aligned}$$

In particular, when $k = 0$, we have

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle = |1\rangle^{\otimes L}$$

where $L \equiv \lceil \log(N) \rceil$.

Suppose $U_{x,N}$ satisfies $U_{x,N}|y\rangle \triangleq |xy(\mathrm{mod} N)\rangle$. Considering $\mathbb{Z}_N^*$ and the fact that the permutation on orthnormal basis can be represented as a unitary operator, one can know for sure that $U_{x,N}$ is unitary. What's more u_s is a

eigenvector of $U_{x,N}$, the corresponding eigenvalue is $e^{\frac{2\pi is}{r}}$ since

$$U_{x,N}|u_s\rangle = \frac{1}{\sqrt{r}}\sum_{k=0}^{r-1} e^{\frac{-2\pi isk}{r}}\left|x^{k+1}\mathrm{mod}N\right\rangle = e^{\frac{2\pi is}{r}}|u_s\rangle$$

Reverse the results above, we get the first half of the quantum ordering Algorithm, which complete the following task:

$$|1\rangle^{\otimes L} = \frac{1}{\sqrt{r}}\sum_{s=0}^{r-1}|u_s\rangle \xrightarrow[\text{Modular exponentiation}]{U_{x,N}^{z_t 2^{t-1}}\cdots U_{x,N}^{z_1 2^0}} \boxed{\frac{1}{\sqrt{r}}\sum_{s=0}^{r-1} e^{2\pi isk/r}|u_s\rangle} = \left|x^k \mathrm{mod}\ N\right\rangle$$

where the state $\frac{1}{\sqrt{r}}\sum_{s=0}^{r-1} e^{2\pi isk/r}|u_s\rangle$ is the one we desire. Apparently the eigenvalue contains the information of r. So as to extract the information, we need a sub-progress named "quantum phase estimation" which based on inverse quantum fourier transformation. One can verify that if t is large enough, such like $t = 2L + 1 + \lceil \log\left(2 + \frac{1}{2\varepsilon}\right)\rceil$, for each $s \in \{0, \ldots, r-1\}$, we will obtain the estimation of $\varphi \approx s/r$ accurate to $2L+1$ bits with probability at least $(1-\varepsilon)/r$. Through the continued fractions algorithm, we will get r with high probability(According to PNT).

In our opinion, the procedure above could be written as:

$$\begin{aligned}
\nabla &\equiv \text{Initialize the state} |0\rangle^{\otimes t} \otimes |1\rangle^{\otimes L} \\
&\quad\ \text{Generate the whole circuit} \\
\mathcal{H}_1 &\equiv H^{\otimes n} \otimes H \\
\mathcal{H}_2 &\equiv CU_{x,N} \\
\mathcal{H}_3 &\equiv FT^{\dagger} \otimes I^{\otimes L} \\
\mathcal{H}_4 &\equiv CF \otimes I^{\otimes L} \\
\Delta &\equiv \sum_i |P_i\rangle\langle P_i|
\end{aligned}$$

It is easy to check that except the $\mathcal{H}_3$, all operators cost polynomial time with respect to $\log N$. The complexity of operator modular exponentiation and continued fraction are both $O(L^3)$, which are two most time-consuming subprocedure of the whole algorithm except the $\mathcal{H}_3$(inverse quantum fourier transform).

Note that $\mathcal{H}_3$ is indeed not an operator which could be implemented by polynomial universal gates. Consider a family of gates used in $\mathcal{H}_3$ which is usually noted by $R_k (k \in \{2, \ldots, L\})$

$$R_k = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{pmatrix}$$

In other words, the original Shor's Algorithm is not a algorithm with superpolynomial acceleration. In order to overcome this, Coppersmith created a new algorithm called the AFFT(Approximate Fast Fourier transform) [12] which can

substitute for the procedure QFT.

5.2 Quantum Simulation and Quantum Algorithm

Quantum Lattice Celluar Automata(QLCA) and Quantum Gas Automata(QGA) are two familiar ideal models in the research of quantum simulation[15] . Meyer, Boghosian[15, 16, 17] have obtained their results respectively by using these models, that is, they construct some quantum algorithms which demonstrate exponentially speedup in such models. For Bohosian, the object they tried to simulated is a QGA which obey lattice Boltzman distribution, where arbitrary fields can be concerned with. They have proofed that the complexity of simulation is only related to the dimension of the lattice, but almost has nothing to do with the number of the particles. However, the number of particle always cause exponentially hardness on a classical computer. In fact, Boghosian's results imply that it is almost impossible for a classical computer to simulate one evolution step of a quantum system including dozens of particles.

We've mentioned that it is the difficulty of quantum simulation that makes people believe quantum mechanics can provide enormous power of computation in the early years.

Note that in this article we do not care about the hardness of simulations. Generally speaking, the hardness of simulation has nothing to do with the one of computation. For instance, people may find it difficult to simulate some classical celluar automaton according to the given regulations, however once the tedious work has been completed there often exists some more simple methods to produce the series. A typical example is that the regulations of an automata actually cause a circle with a finite period in the series. The same thing can happens to quantum simulations too.

However, it is important to know that there must exists some cases in which simulations and computations are equivalent. These extreme cases often appears when the length of regulations is near the Kolmogorov complexity(lower bound of description) of a series. Still, strictly speaking, at present no one can proof that polynomially universal unitary operators really cause exponentially difficulty in classical computation. To understand this, just consider an easy but helpful fact that almost all the problems we want to efficiently solved on a quantum computer are in the class BQP, and we have BQP$\subseteq$PSPACE. Unfortunately PSPACE=P is not totally impossible. Of course most people don't believe this is true, since this would imply that Shor's Algorithms can be polynomially simulated on a classical computers.

Now we discuss how to extract a corresponding quantum algorithm from a method of quantum simulation, which is believed to be exponentially faster than any classical one of the same target.

On a high level, we should do following things:

- Find a family of experiments of quantum mechanics which can be efficiently simulated by quantum computers but are believed to be hardly to simulate and compute by classical computers;
- Design a 'good' problem about some non-trivial properties of the last state of the system, which makes quantum computers able to present the answer to the observer quickly.

Designing the problem is a crucial step. In most cases, though we may have quickly obtained the probabilistic distribution very close to the real experiments, we can not know the whole information in short time. So first we have to ask a question which can be easily verified by any quantum computers containing the whole quantum information of the system.

For example, we can ask a question such like:

- What the number of the n_0-th digit of the probability of a certain system arriving in $\Omega'(\Omega' \subset \Omega)$?

The problem of this method is that in high dimensional spaces, it is very likely that the probability of the set Ω' is exponentially close to zero, which actually enables a classical computers to guess zero without running and get the right answer in most cases.

Now we propose our version: Suppose ϕ is the wave function of the system we've simulated and $|\phi(X)|^2, X \subset \Omega$ is the probability of $\vec{x}$ appear in X. Try to find two subsets $A, B \subset \Omega$ s.t.

$$\frac{3}{7} \leq \frac{|\phi(A)|^2}{|\phi(B)|^2} \leq \frac{4}{7}$$

and determine the value of the n_0-th digit of $\phi(A)$.

For the systems which (probabilistic)Turing machine cannot simulate in polynomial time, the question above is intuitively hard to answer, though up till now no one can proof or disproof it.

On the other hand, if these systems can be efficiently simulated by quantum computers, repeating following procedure will ensure us to find the answer relatively much faster than any probabilistic Turing machine of the same aim.

Definition 5.2.1. (Vector of normal vectors $\vec{x}$)

$$\vec{x} \equiv \left(\begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} 1 \\ \vdots \\ 0 \\ 0 \end{pmatrix} \right)$$

Definition 5.2.2. (Procedure P_Q) P_Q(In pseudo-code):
while(find the answer)

{
. Mid-cut the space Ω by super-plane whose normal vector is x_i.
. Suppose the two spaces is Ω_1 and Ω_2
. if(the condition is satisfied(verifies by testing))
. {
. halt
. }
. else
. {
. $\Omega = \min_{|\phi|}\{\Omega_1, \Omega_2\}$
. $i++$
. }
}

5.3 Conclusions and Future Works

We formally propose the theory of physical computation, define the concepts of resource and complexity. Several examples, including classic mechanics and quantum mechanics, are discussed and analyzed under the framework of physical computation. A technique, which is used to converse a method of quantum simulation into a quantum algorithm, is discussed.

This is an exciting field, we believe that there is more exciting topic to study. A very interesting question is: can we find a physical mechanism as the fastest implementation of an arbitrary functions?[55]

In chapter III, we talk about the question of calculating the centroid of an object. We thought it is the limitation of dimensions(only three dimensions) hide the advance of the method we mentioned. We conjecture that this method has a excellent counterpart in high dimensional cases. We'll have a try in the (quantum)statistics mechanism.

In chapter V, we talked about quantum simulations and how to construct a clever problem to induced a quantum algorithm. Actually, we conjecture that the problem we construct is a hard one in class $\#P$, for these questions have a counting style. However, we are not sure about whether the designed questions could be in $\#P - hard$ under some specific statistical models. We shall try to work on this in the future.

We've mentioned that we assume that polynomial qubits is polynomially hard to prepare. However, it is harder to control the qubits as the number of them increase[18] so far. So one can still conjecture that preparing qubits itself is a "complicated computing", and the results up till now can be explained as someone displace the resource consuming procedure, just like DNA Algorithms.

Bibliography

[1] P.Benioff. *The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines* J. Stat. Phys.,22(5):563-591,1980

[2] Richard P.Feynman. *Simulating Physics with Computers* Int. **J**. Theor. Phys. **21**(1982)467-888

[3] D.Deutsch. *Quantum theory, the Church-Turing Princple and the universal quantum computer.* Proc. R. Soc. Lond. A,400:97,1985

[4] E.Bernstein and U.Vazirani. *Quantum complexity theory.* SIAM J. Comput., 26(5): 1411-1473, 1997.

[5] A.C.Yao. *Quantum circuit complexity.* Proc. of the 34th Ann. IEEE Symp. on Foundations of Computer Science,pages 352-361,1993

[6] J.Myhill. *A Recursive Function,Defined On a Compact Interval and Having a Continuous Derivative that is Not Recursive.* Michigan Math.J.18(1971)

[7] Itamar Pitowsky. *The Physical Church Thesis and Physical Computational Complexity* A Jerusalem Philosophical Quarterly 39(January 1990)

[8] Marian B.Pour-El and Ning Zhong. *The Wave Equation with Computable Initial Data Whose Unique Solution Is Nowhere Computable* Mathematical Logic Quarterly ©Johann Ambrosius Barth 1997

[9] D.Deutsch and R.Jozsa. *Rapid solution of problems by quantum computation.* Proc. R.Soc. London A, 439:553, 1992

[10] P.W.Shor. *Algorithms for quantum computation:discrete logarithms and factoring.* In Prodeedings, 35th Annual Symp. on Foundations of Computer Sciece, IEEE press, Los Alamitos, CA, 1994

[11] P.W.Shor. *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer.*SIAM. J. Comp.,26(5):1484-1509,1997

[12] D.Coppersmith. *An approximate Fourier transform usefulin quantum factoring.* IBM Research Report RC 1994

[13] L.K.Grover. *Quantum mechanics helps in searching for a needle in a haystack.* Phys. Rev. Lett., 79(2):325,1997

[14] L.M.Adleman. *Molecular computation of solutions to combinatorial problems.* Science, 266:1021, 1994

[15] D.A.Meyer, J.Stat.Phys.85,551(1996)

[16] B.M.Boghosian and W. Taylor, Phys. Rev. E 57, 54(1998)

[17] B.M.Boghosian and W.Taylor, Intl. J.Mod. Phys. C8, 705(1997).

[18] M.A.Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press (2000)

[19] Edwin Beggs, José Félix Costa, and John V. Tucker *Computational Models of Measurement and Hempel's Axiomatization*

[20] Edwin Beggs, José Félix Costa, and John V. Tucker *Physical Oracles: The Turing machine and the Wheatstone Bridge*

[21] Edwin Beggs, José Félix Costa, Bruno Loff, and John V. Tucker *Oracles and Advice as Measurements*

[22] Edwin Beggs, José Félix Costa, Bruno Loff, and John V. Tucker *On the Complexity of Measurement in Classical Physics*

[23] Edwin Beggs, José Félix Costa, Bruno Loff, and John V. Tucker *Computational complexity with experiments as oracles. II. Upper bounds*

[24] J.Gerver, *"The existentce of pseudocollisions in the plane"*, *J.Differential Equations*, 89(1991),1-68

[25] Z.Xia *"The existentce of noncollision singularities in the n-body problem"*, *Annals Math*, 135(3)(1992),411-468

[26] P.Painlevé, *"Lecons sur la theorie analytic des equatons differentielles"*, *Hermann*, Paris, 1897.

[27] P.Penrose, *"The emperor's New Mind"*, Oxford University Press, 1989.

[28] P.Penrose, *"Shadows of the Mind"*, Oxford University Press, 1994.

[29] W.Smith, *"Church's thesis meets the N-body problem"*, manuscript, 1993(http://www.neci.nec.com/homepages/wds/works.html).

[30] W.Smith, *"Church's thesis meets quantum mechanics"*, manuscript, 1999(http://www.neci.nec.com/homepages/wds/works.html)

[31] A.M.Turing, *"On computable numbers, with an application to the Entscheidungproblem,"*, *Proc. London Math. Soc., Series 2*, 42(1936-37),230-265.

[32] Mac Lane, *"Categories for the Working Mathematician"*, Springer (Graduate Texts in Mathematics) ISBN 0-387-98403-8, 1998 (1972).

[33] Heinz-Otto Peitgen, Hartmut Jürgens, Dietmar Saupe, *"Chaos and Fractals "*, Springer Press, 1992,2004.

[34] C.D.GODSIL,M.GRÖTSCHEL, D.J.A.WELSH *"Combinatorics in Statistical Physics "*©ELsevier Science B.V.1995.

[35] D.J.A.WELSH *"The Computational Complexity of Some Classical Problems from Statistical Physics "*

[36] Christos H.Papadimitriou *"Computational Complexity "*Addison-Wesley

[37] T.H.Cormen, C.E.Leiserson, and R.L.Rivest *"introduction to Algorithms"*MIT Press, Cambridge, Mass.,1990

[38] R.Motwani and P.Raghavan. *"Randomized Algorithms"*Cambridge University Press, Cambridge,1995

[39] A.Y.Kitaev. *"Quantum measurements and the Abelian stabilizer problem"* arXive e-print quant-ph/9511026,1995

[40] M.Mosca.*"Quantum Computer Algorithms"*Ph.D. thesis, University of Oxford,1999

[41] A.Ekert and R.Jozsa. *"Quantum computation and Shor's factoring algorithm"*Rev. Mod. Phys., 68:1, 1996

[42] G.H.Hardy and E.M.Wright. *"An Introduction to the Theory of Numbers, Fourth Edition"*Oxford University Press, London,1990

[43] D.P.DiVincenzo. *"Quantum computation."* Science, 270:255, 1995, arXive e-print quantu-ph/9503016

[44] D.P.DiVincenzo. *"Two-bit gates are universal for quantum computation."* Phys. Rev. A, 51(2):1015-1022, 1995

[45] D.P.DiVincenzo. *"Quantum Gates and circuits."* Proc. R. Soc. London A, 454:261-276,1998

[46] A.K.Lenstra and H. W. Lenstra Jr. editors. *"The Development of the Number Field Sieve."* Springer-Verlag, New York, 1993

[47] P.O.Boykin, T.mor, M.pulver, V.Roychowdhury, and F.Vatan *"On universal and fault-tolerant quantum computing"* arXiave e-print quant-ph/9906054,1999

[48] A.T.Kitaev *"Quantum Computations: algorithms and error correction."* Russ. Math. Surv. 52(6):1191-1249,1997

[49] A. Church. *"An unsolvable problem of elementary number theory ."* Am. J. Math, 58:345,1936

[50] M.D. Davis *"The Undecidable."* Raven Press, Hewlett, New York, 1965

[51] W.Greiner *"Classical Mechanics Systems Of Particles And Hamiltonian."* Springer-Verlag, New York, 2003

[52] Herbert Goldstein, Charles Poole, John Safko *"Classical Mechanics ."* American Journal of Physics – July 2002 – Volume 70, Issue 7, pp. 782

[53] Sanjeev Arora, Boaz Barak *"Computational Complexity:A Modern Approach."*Draft of a book, 2007

[54] Michael Sipser *"Introduction to the Theory of Computation."*©2006 Thomson Course Technology, a division of Thomson Learning, Inc.

[55] Stephen Wolfram *"Undecidability and Intractability in Theoretical Physics."*1985 Physical Review Letters

[56] Song Fangmin *"Introduction to Models of Computation."* handbook

[57] H.P.Barendregt *"The Lambda Calculus: Its Syntax and Semantics."* 2nd edition, North-Holland, 1984

[58] S.C.Kleene *"Introduction to MetaMathematics."* North-Holland, 1967

[59] George S. Boolos, Richard C.Jeffrey, John Burgess *"Computability and Logic."* Cambridge University Press, 2002

[60] Daniel E. Cohen *"Computability and Logic."* Ellis Horwood, 1987

[61] A.M.Turing *"Computability and $\lambda-$Definability."* The Journal of Symbolic Logic, Volume 2, Number 4, 1937

[62] Andrew Chi-Chih Yao *"Classical Physics and the Church Turing Thesis."* ISSN 1433-8092

[63] F.Harary *"Graph theory."* Addison-Wesley 1969

Publications

Some results of the paper are published in “physical computation theory”, Journal of Wuhan University(Natural Science Edition) vol.58, 2012

Acknowledgements

I am grateful to my supervisor Prof. Dr. Song Fangmin, who introduced me to the marvelous palace of mathematical logic and encouraged me to do research in physical computation. Some interesting examples are also due to him.

Also, without the thoughtful remarks and helpful suggestions of Nan Wu, Haixing Hu, Jiasen Wu and Kun Wang, many valuable things will be missed in this paper.

Finally, great thanks to my dear parents. Your love ensures that I will never be misled by books, someone else or myself during my journey through the great ocean of truth.

Vita

Zheng Huimin was born on Nov. 06^{th} 1985, in Bengbu, a city in An Hui Province. In 2005, he was admitted into Nanjing University, majoring at Computer Science and Technology. In 2009, he was admitted into master program, supervised by Prof. Song Fangmin.

Now, his research interests are in the fields of quantum computation and mathematical logic.