計算理論　第 1 回
計算とは何か？

岡本 吉央
okamotoy@uec.ac.jp

電気通信大学

2020 年 10 月 1 日

最終更新：2020 年 9 月 30 日　16:44

概要

この講義の主題

計算理論 (Theory of Computation)
- 計算可能性理論 (Computability Theory)
- 計算複雑性理論 (計算量理論) (Complexity Theory)

講義の進め方

- 前半：計算可能性理論 (担当：岡本)
- 後半：計算複雑性理論 (担当：垂井先生)

スケジュール 前半 (予定)

1	計算とは何か？	(10/1)
2	計算モデル	(10/8)
3	チャーチ・チューリングの定立	(10/15)
⋆	休み (体育祭)	(10/22)
4	コード化	(10/29)
5	計算可能性	(11/5)
6	停止性問題	(11/12)
7	再帰定理	(11/19)
8	前半のまとめ	(11/26)

注意：予定の変更もありうる

情報

教員
- 岡本 吉央 (おかもと よしお)
- 居室：西 4 号館 2 階 206 号室
- E-mail：okamotoy@uec.ac.jp
- Web：http://dopal.cs.uec.ac.jp/okamotoy/

講義資料
- Web：http://dopal.cs.uec.ac.jp/okamotoy/lect/2020/comp/
- 注意：資料の入手等は各学生が自ら行う
- 講義前日の夕方 18 時までに，ここに置かれる

講義資料

http://dopal.cs.uec.ac.jp/okamotoy/lect/2020/comp/
- スライド
- 印刷用スライド：8 枚のスライドを 1 ページに収めたもの
- 演習問題
- 用語集

授業の進め方

講義 (85 分)
- スライドと板書で進める
- スライドのコピーに重要事項のメモを取る

退室 (5 分) ←重要
- コメント (授業の感想，質問など) を Google Forms で提出する
- コメントとそれに対する回答は (匿名で) 講義ページに掲載される

オフィスアワー (アポイントメントによる)
- 質問など

演習問題

演習問題の種類
- 復習問題：講義で取り上げた内容を反復
- 補足問題：講義で省略した内容を補足
- 追加問題：講義の内容に追加
- 発展問題：少し難しい (かもしれない)

演習問題 (続)

答案の提出
- 演習問題の答案をレポートとして提出してもよい
- レポートには提出締切がある (各回にて指定)
- レポートは採点されない (成績に勘案されない)
- レポートにはコメントがつけられて，返却される
  - 返却された内容については，再提出ができる (再提出締切は原則なし)

評価

全体の評価

- 前半 (岡本) と後半 (垂井先生) の担当分に対して，別々に評価が行われる
- 別々の評価を合算して，全体の評価とする

前半の評価

1 回のレポート提出のみによる

- 出題形式
  - 4 問の出題が行なわれる
  - 全問に解答する
- 配点：1 題 25 点満点，計 100 点満点

前半の教科書・参考書

前半の教科書

- 指定しない

前半に関する全般的な参考書

- 鹿島亮,『C 言語による計算の理論』，サイエンス社，2008.
- 渡辺治,『計算可能性・計算の複雑さ入門』，近代科学社，1992.
- Uwe Schöning, Theoretische Informatik – kurz gefasst, 5. Auflage, Spektrum Akademischer Verlag, 2008.
- Michael Sipser, Introduction to the Theory of Computation, 3rd Edition, Course Technology Ptr. 2012. (第 2 版の日本語訳あり)

など

今日の目標

- 「計算理論の考え方」を概観し，この講義の目標を明確にする
- 部分関数について復習する

ポイント

- 計算という「現象」がある
- 計算は現象なので，科学の対象となる

$\rightsquigarrow$ コンピュータ・サイエンス ＝ 計算という現象に対する科学

注意：計算科学 (computational science) $\neq$ コンピュータ・サイエンス

目次

計算理論 $\neq$ 計算機理論

重要ポイント

「計算理論」は「計算機の理論」ではない

注：「Computer Science」の訳語として「計算機科学」，「Theoretical Computer Science」の訳語として「理論計算機科学」を充てることがあるけれども，それらが「計算機の科学」であると誤解されることがあるため，この訳語を嫌う専門家もいる

疑問

では,「計算理論」は何を扱うのか？

$\rightsquigarrow$ つまり,「計算機」ではない「計算」とは何なのか？

人間が行なう計算

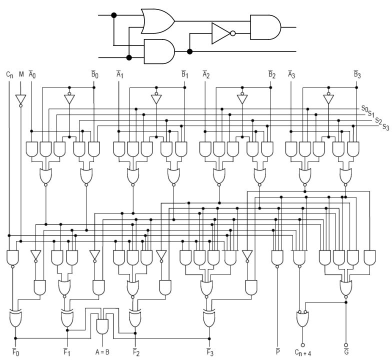
$$
\begin{aligned}
&((1+2)\times(3-4))/((5-6)\times(7+8))\\
&=(3\times(3-4))/((5-6)\times(7+8))\\
&=(3\times(-1))/((5-6)\times(7+8))\\
&=(-3)/((5-6)\times(7+8))\\
&=(-3)/((-1)\times(7+8))\\
&=(-3)/((-1)\times 15)\\
&=(-3)/(-15)\\
&=1/5
\end{aligned}
$$

いま，この計算を人間が行なった

事実

人間も計算を行なう

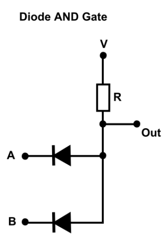
論理回路が行なう計算

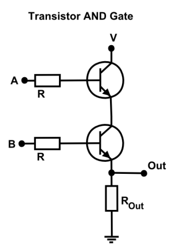


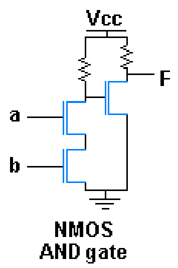
https://commons.wikimedia.org/wiki/File:74181aluschematic.png

論理素子を電気的な方法で実現する

AND 素子の実現



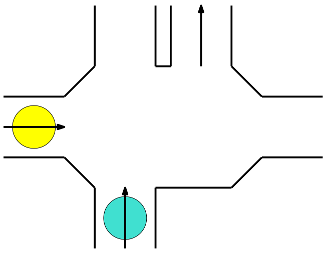




https://commons.wikimedia.org/wiki/File:DiodeANDgate.png
https://commons.wikimedia.org/wiki/File:TransistorANDgate.png
https://commons.wikimedia.org/wiki/File:NMOS_AND_gate.png

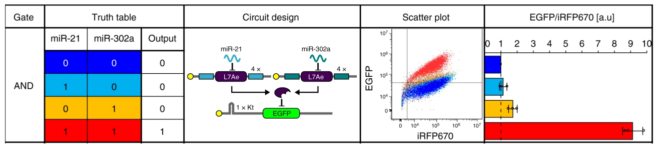
論理素子を力学的な方法で実現する

AND 素子の実現



論理素子を物質的 (生命的) な方法で実現する

AND 素子の実現



S. Matsuura, H. Ono, S. Kawasaki, Y. Kuang, Y. Fujita, H. Saito, Synthetic RNA-based logic computation in mammalian cells. Nature Communications 9 Article No. 4847 (2018)

論理素子をゲームで実現する

https://www.youtube.com/watch?v=VEcmaXwjwuY
https://www.youtube.com/watch?v=jx59oHXdXBU

計算を理解するために必要な理論

ここまでのまとめ
- 計算機だけが計算を行なうわけではない
- 様々な対象が計算を行なえる

求められるのは？
- 結局「計算」とは何なのか？ はっきりさせること
- 「計算」について抽象的に語るための理論体系を作ること

$\rightsquigarrow$ 計算理論

計算が行なっていること

計算が行なっていること (直感的な説明)
- 何か入力 (初期状態) が与えられる
- 入力 (初期状態) が時間とともに処理される (変化する)
- 最終的に，出力 (終了状態) が得られる

時間とともに処理される (変化する) ＝ 計算する

入力 ⟶ $\boxed{\text{処理}}$ ⟶ 出力

アルゴリズムとは？ (直感的な定義)
処理の手順のこと

「どのような手順が許されるのか」ということは重要

「計算」と「計算機」の理解 — カリキュラムにおける位置づけ

抽象度	講義名
高	プログラミング言語論，計算理論
	アルゴリズム論第一，アルゴリズム論第二
	形式言語理論，プログラミング通論
	オペレーティング・システム論，言語処理系論
	計算機通論，コンピュータ設計論
	論理設計学
低	電気・電子回路

位置づけに異論はあるかもしれない

目次

講義『計算理論』の概要

よく議論される計算モデル
- 「万能」な計算モデル　(← 普通の「計算理論」の対象)
- 「具体的」な計算モデル

- 「万能」の定義は，後の講義で紹介
- 「万能 $\neq$ なんでもできる」なので，注意

「万能」な計算モデルに対する 2 つの主な理論

計算可能性理論
- 「万能」な計算モデルで原理的にできることは何か？
- 「万能」な計算モデルで原理的にできないことは何か？

計算複雑性理論 (計算量理論)
- 「万能」な計算モデルで実用的にできることは何か？
- 「万能」な計算モデルで実用的にできないことは何か？

計算可能性理論 (Computability Theory)

計算可能性理論

- 「万能」な計算モデルで原理的にできることは何か？
- 「万能」な計算モデルで原理的にできないことは何か？

ここで「原理的にできる」とは？

- アルゴリズムが存在する

重要な事実

原理的にできないことが存在する

前半の目標

- 何が原理的にできないのか，理解する
- なぜ原理的にできないのか，理解する

計算複雑性理論 (計算量理論，computational complexity theory) (1)

計算複雑性理論 (計算量理論)

- 「万能」な計算モデルで実用的にできることは何か？
- 「万能」な計算モデルで実用的にできないことは何か？

ここで「実用的にできる」とは？

- 文脈に依存する ($\rightsquigarrow$ 計算における資源の概念)
- 多くの場合,「多項式時間で」できる

後半の目標 (1)

- 計算における「資源」の概念を理解する

計算複雑性理論 (計算量理論，computational complexity theory) (2)

計算複雑性理論 (計算量理論)

- 「万能」な計算モデルで実用的にできることは何か？
- 「万能」な計算モデルで実用的にできないことは何か？

重要な事実

- 原理的にできるが，実用的にできないことが存在する
- よく現れる問題が，実用的に解けるかどうか分かっていない
  $\rightsquigarrow$ P vs NP 問題 ($\mathrm{P} \neq \mathrm{NP}$ 問題，$\mathrm{P} =?\ \mathrm{NP}$ 問題)

後半の目標 (2)

- 「P vs NP 問題」とは何なのか，理解する

目次

計算モデル

計算モデルとは？ (直感的な定義)

「計算主体」を数学的に抽象化したもの

大きく分けて 2 つの種類のモデルがある

- 「機械」によるモデル
- 「関数」によるモデル

注意 (重要)

- 計算モデルによって，異なる計算理論が生まれる
  - しかし，計算可能性理論 (の本質) はあまり変わらない
  - しかし，計算複雑性理論 (の本質) は大きく変わりうる

(変わる部分と変わらない部分がある)

「機械」による計算モデル

代表的なもの

- チューリング機械
- ランダム・アクセス機械
- カウンタ機械
- ポインタ機械
- タグ・システム

計算を機械における操作であると見なして，抽象化する

「関数」による計算モデル

代表的なもの

- 帰納的関数 (μ 再帰関数)
- ラムダ計算
- マルコフ・アルゴリズム

計算を関数の適用であると見なして，抽象化する

この講義の前半で扱う計算モデル

この講義 (の前半) では

- ある「単純化したプログラミング言語」を計算モデルとして用いる
  $\rightsquigarrow$ 次回
- 計算というときは,
  自然数の組を自然数にうつす部分関数の計算を考察対象とする
  $\rightsquigarrow$ 今日の残り時間

プログラミング言語も計算モデルであると言えるが
それが「機械」か「関数」か，という分類は難しい

目次

自然数

自然数 (直感的な定義)

0 以上の整数を自然数と呼ぶ

注
- 自然数を「厳密」に定義することは重要であり，情報科学や数学の基礎でもあるが，ここではやらない
- 数学 (特に数論) では，1 以上の整数を自然数と定義することが多い (分野によって用語の定義は異なる可能性がある)

記法：自然数全体の集合

自然数全体の集合を $\mathbb{N}$ で表す

「○○全体の集合」とは「○○をすべて集めてできる集合」という意味

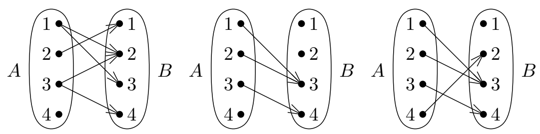
関数 (1)

集合 A, B

定義：関数

$f \subseteq A \times B$ が A から B への関数であるとは

任意の $a \in A$ に対して，ある $b \in B$ が一意に存在して，$(a, b) \in f$ となること



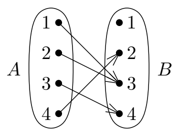
関数 (2)

集合 A, B，A から B への関数 f

定義：関数の値

$(a, b) \in f$ であるとき，b を a における f の値と呼び，b を $f(a)$ と書く

関数の記法：$f \colon A \to B$



- $f(1) = 3$
- $f(2) = 3$
- $f(3) = 4$
- $f(4) = 2$

A を f の始域，B を f の終域と呼ぶ

記法：関数全体の集合

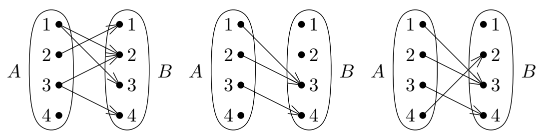
A から B への関数全体の集合を B^A と書く

部分関数 (1)

集合 A, B

定義：部分関数

$f \subseteq A \times B$ が A から B への部分関数であるとは
任意の $a \in A$ に対して，$(a, b) \in f$ となる $b \in B$ が
存在しないか，存在するとしてもただ 1 つであること



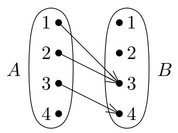
部分関数 (2)

集合 A, B，A から B への部分関数 f

定義：部分関数の値

$(a, b) \in f$ であるとき，b を a における f の値と呼び，b を $f(a)$ と書く

部分関数の記法：$f \colon A \nrightarrow B$ (この講義ではこれを採用)



- $f(1) = 3$
- $f(2) = 3$
- $f(3) = 4$

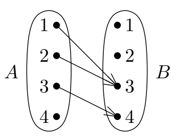
部分関数の記法は他にもいろいろある (統一的な記法が見当たらない)

部分関数 (3)

集合 A, B，部分関数 $f \colon A \nrightarrow B$

記法：定義されているかされていないか

$a \in A$ に対して，
- $b = f(a)$ となる $b \in B$ が存在することを，$f(a)\downarrow$ と書く
- $b = f(a)$ となる $b \in B$ が存在しないことを，$f(a)\uparrow$ と書く (このとき，$f(a)$ は定義されない，と言うことがある)



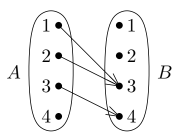
- $f(1)\downarrow$
- $f(2)\downarrow$
- $f(3)\downarrow$
- $f(4)\uparrow$

部分関数 (4)

集合 A, B，部分関数 $f \colon A \nrightarrow B$

定義：定義域

f の定義域とは，$\mathrm{dom}(f) = \{a \in A \mid f(a)\downarrow\}$



- $\mathrm{dom}(f) = \{1, 2, 3\}$

性質 (演習問題)

部分関数 $f \colon A \nrightarrow B$ に対して

$$f \text{ が関数} \iff \mathrm{dom}(f) = A$$

部分関数と対比させて，関数のことを全域関数と呼ぶことがある

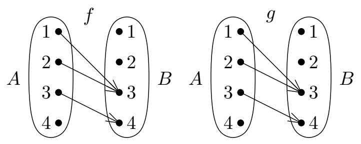
2 つの部分関数が等しいこと

集合 A, B，部分関数 $f, g\colon A \nrightarrow B$

定義：2 つの部分関数が等しいこと

任意の $a \in A$ に対して，次のどちらかが成り立つとき，
f と g は等しいといい，$f = g$ と書く

1. $f(a)\downarrow$，かつ，$g(a)\downarrow$，かつ，$f(a) = g(a)$
2. $f(a)\uparrow$，かつ，$g(a)\uparrow$



目次

今日のまとめ と 次回の予告

今日のまとめ

- 「計算理論の考え方」を概観し，この講義の目標を明確にする
- 部分関数について復習する

次回の予告

- この講義で扱う計算モデルを導入する