Дискретна математика

доц. д-р Тодорка Глушкова,
Катедра „Компютърни технологии“, ФМИ

10

Теория на изчислимостта

Увод

- В течение на годините свикнахме с това, компютрите да са в състояние да решават все по-сложни задачи
  - Управляват самолети, играят шах, симулират военни действия, …
  - В ИИ: самообучаващи се машини, машини с интелект, роботи, …
- Как тогава идваме до идеята за определяне границите на възможностите на компютрите?

Увод

- Хората имат интуитивно усещане за това, какво е нещо да се изчисли
  - Неформалните представи, обаче, са крайно недостатъчни за правене на твърдения, свързани с решимостта на проблемите
  - Те не достигат до формално прецизиране на решимостта
- За тази цел е възникнала теорията за изчислимостта
  - Тя доставя необходимата теоретична основа във формата на формални изчислителни модели

Увод

- Съществува голямо разнообразие на изчислителни модели
  - Въпреки външните им различия, повечето модели обосновават едно и също понятие за изчислимост (тезис на Чърч)
  - В този смисъл изчислимостта се разглежда като едно базово свойство, което съществува независимо от формата и не се влияе от нея

Изчислими функции

- За да може да сравняваме производителността на различни типове машини, искаме да ги сравняваме на основата на обща базова задача:
  - От зададен вход да се изчисли стойност
  - Начинът за въвеждане на данни е без значение

Изчислими функции

- Понеже всеки вход и изход трябва да бъде кодиран по някакъв начин, можем да редуцираме базовата компетентност на една машина до изчисляване на функции от вида
  - f: $\Sigma^* \rightarrow \Gamma^*$, където Σ , Γ - входна и изходна азбука

Изчислими функции

- Функцията f се нарича изчислима, ако съществува алгоритъм, който изчислява тази функция
- Посредством просто прекодиране функции от вида f: $\Sigma^* \rightarrow \Gamma^*$ могат да бъдат трансформирани във функции от вида f: $\mathbb{N} \rightarrow \mathbb{N}$
- Обобщение: една частична функция f: $\mathbb{N}^k \rightarrow \mathbb{N}$ се нарича изчислима, ако съществува алгоритъм, който изчислява тази функция

Тюринг машина

- Представен за първи път през 1936 год. от Алан Тюринг
- Най-старият и най-използваемият модел за формално представяне на понятието „изчислимост“
- Изпълнява във всеки аспект изискванията на един формален модел, като
  - От една страна, позволява математически прецизни твърдения за изчислимостта
  - От друга страна, сравнително прост и ясен, което позволява интуитивно разбиране на тази сложна материя

Алън Матисън Тюринг

- **Алън Матисън Тюринг (1912-1954):** британски математик, логик, криптоаналитик, информатик и философ.
- Има голям принос в развитието на компютърните науки с формализирането на концепциите за „алгоритъм “ и „изчислимост“ чрез машината на Тюринг, абстрактен модел на компютър с общо предназначение. Смятан е за бащата на теоретичната информатика и теорията на изкуствения интелект.
- По време на 2-рата световна война Тюринг работи за Правителствената школа по кодиране и шифроване в британския център за криптоанализ. За известно време оглавява група, чиято задача е криптоанализа на германските военноморски сили.
- След войната Алън Тюринг работи в Националната физическа лаборатория, където създава ACE - един от първите компютри със съхранявана в паметта компютърна програма. През 1948 година постъпва в лабораторията в Манчестърския университет, където участва в създаването на т.нар. Манчестърски компютри.

Мотивация

> *“Computing is normally done by writing certain symbols on paper. We may suppose this paper is divided into squares like a child’s arithmetic book.”*

В оригиналната работа Тюринг мотивира концепцията за машинния модел както следва:

1. Започва от това, което е използвал за смятане в детството: празен кариран лист

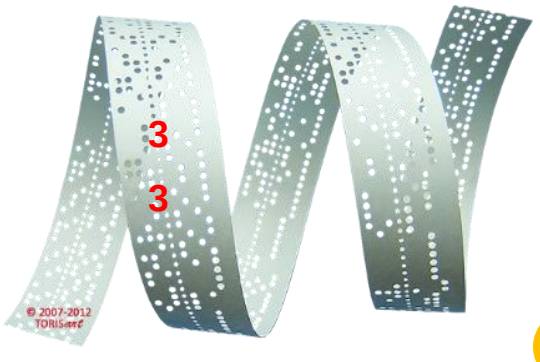
Мотивация

"… I think that it is agreed that the two-dimensional character of paper is no essential of computing. I assume then that the computing is carried out on one-dimensional paper, i.e. on tape divided into squares."

Прави първата абстракция: двуразмерността на карирания лист няма никакво значение за изчислението.

- Изчисления могат да се правят върху едноразмерна лента

Мотивация

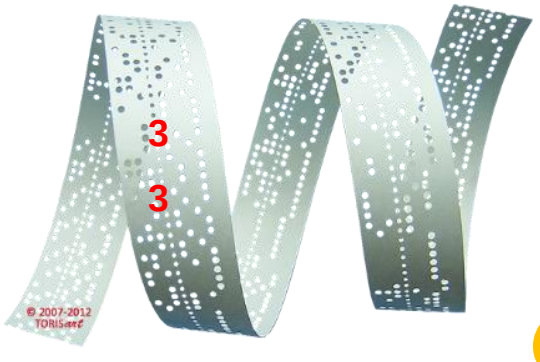


“We may suppose that there is a bound B to the number of symbols or squares which the computer can observe at one moment. … We will also suppose that the number of states of mind which will be taken into account is finite.”

3 Приема следващи допускания:

- Съществуват само краен брой символи, които могат да бъдат записвани върху лентата
- Във всеки момент от процеса на изчисление, човешкият мозък се намира в едно (от крайно многото) състояние

Мотивация



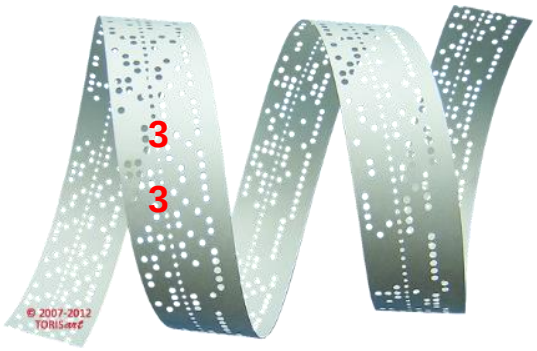
“The simple operations must therefore include: (a) Changes of the symbol on one of the observed square. (b) Changes of one of the square observed to another square within L squares of one of the previously observed squares.”

4

След това, Тюринг дефинира множество от елементарни операции, от които могат да се образуват комплексни изчисления:

- Чрез операции може да се промени символа (съдържанието) на актуално разглежданата клетка от лентата
- Чрез операции може да се пренасочва вниманието към съседните клетки

Мотивация



“It may be that some of these changes necessarily involve a change of state of mind. The most general single operation must therefore be taken to be one of the following: (A) A possible change (a) of symbol together with a possible change of state of mind. (B) A possible change (b) of observed squares, together with a possible change of state of mind.”

5 Двете операции са придружени от възможна промяна на състоянието

История на решимостта

- Изчислимостта като проблем
  - **Георг Кантор** - въведен е методът на диагонализацията.
  - **Хилберт** търси обща алгоритмична процедура за решаване на математически проблеми. - 1900 - 1928 г.
  - **Бъртранд Ръсел** оповестява парадокса на множествата.
  - **Курт Гьодел** предлага своя теорема - всяка математическа система от аксиоми съдържа твърдения, които не са нито доказуеми, нито недоказуеми - 1931 г. Опитът на Хилберт се проваля.
  - **Алонсо Чърч** предлага твърдението, че всяка интуитивно изчислима f е рекурсивна функция. Чърч и Тюринг, независимо един от друг, достигат до сходни резултати: Функцията се счита за изчислима, ако може да бъде изчислена от Тюринг-машина.
  - **Алън Тюринг** прави опит за решение на проблема за изчислимостта и създава машината на Тюринг (1937 г.). Веднага възникава и проблема за спирането на машината.

Неразрешими алгоритмични проблеми

- В практиката често се налага да се определи дали произволен обект от даден клас обекти притежава или не дадено свойство.
- *Например: Дали едно произволно число е просто; дали произволна машина на Тюринг спира след краен брой тактове и т.н.*
- Ако отговорът се търси чрез общ за всеки отделен случай алгоритъм, казваме че проблемът е алгоритмичен. Чрез подходящо кодиране и трансформации проблемът може да се сведе до това дали дадена дума над някаква азбука принадлежи или не на даден формален език, над тази азбука. Ако думата принадлежи на този формален език, казваме, че проблемът е решим, в противен случай – е нерешим.

Неразрешими алгоритмични проблеми

- С други думи един алгоритмичен проблем е нерешим, ако не съществува машина на Тюринг, която за всеки код на отделен случай чрез краен брой тактове да определи дали думата е от фиксирания език или не.
- Казахме, че всеки рекурсивен език е рекурсивно номерируем. Вярно ли е обратното?

Изчислимост и алгоритми

- През 30-те години на XX век били предложени няколко математически определения на понятието ефективен метод (алгоритъм, формална процедура, изчисляващо устройство, компютър).
- Било доказано, че всички такива определения са еквивалентни, т.е. класа задачи, които можем да решим с някоя от предложените схеми (примерно с машини на Тюринг) точно съвпада с класа задачи, решими с всяка друга разумна схема за правене на изчисления (примерно най-мощните съвременни компютри, с допълнителна възможност да им добавяме памет по време на изчислението).

Нерешими задачи

- През 30-те години на XX век бил разрешен и следният важен въпрос: Съществуват ли задачи, които могат да се изразят като задачи за изчисляване, но няма алгоритъм, който да ги решава?
- Оказва се, че има много такива задачи. Ето една нерешима задача:

  Има ли алгоритъм, който да определя дали подадена на входа му компютърна програма решава задачата за удвояване на число, представено като поредица от единици?

Нерешими задачи

- Т.е. има ли машина на Тюринг, която да решава дали описана върху лентата ѝ последователност от символи е дефиниция на алгоритъм, еквивалентен на машината **Double**?
- Горната задача е нерешима и в по-обща формулировка, известна като теорема на Райс, т.е. няма алгоритъм, който да установява еквивалентност на компютърни програми. Такъв тип нерешимост има важно практическо значение

***Не можем да създадем автоматизирани средства за проверка на коректността на компютърните програми*.**

Нерешими задачи

- Още Тюринг през 1936 г. доказва, че няма алгоритъм, който да определя дали дадена програма ще завърши изчислението си или ще се зацикли.
- Нерешимите задачи имат пряка връзка с основите на математиката и математическата логика.

Твърдения

- **Теорема:** Съществува рекурсивно номерируем език, който не е рекурсивен.
- **Следствие:** Съществува формален език, който не е рекурсивно номерируем. Следователно не се разпознава от машината на Тюринг.
- **Теорема:** Допълнението на всеки рекурсивен език е рекурсивен.
- **Теорема на Райс**: Алгоритмичен проблем за разпознаване на кое да е нетривиално свойство на рекурсивно номерируемите езици е ***неразрешим.***

Твърдения

- **Теорема:** Един формален език е от тип 0 (т.е. от общ вид) тогава и само тогава, когато е рекурсивно номерируем, т.е. когато се разпознава от машината на Тюринг.
- Следователно по всяка граматика от тип 0 може да се построи недетерминирана машина на Тюринг, която да моделира възможните изводи в тази граматика и да следи дали входната дума се получава в резултат на някой извод.

Твърдения

- Обратно, по всяка недетерминирана машина на Тюринг, може да се построи граматика от тип 0, която чрез изводите си моделира възможните преходи от една конфигурация към друга.

Използвана литература в курса

- D. W. Hoffmann, Theoretische Informatik, Hansen Verlag, 2009
- H. P. Gumm, M. Sommer, Einfuehrung in die Informatik, Oldenbourg Wissenschaftsverlag, 2004
- J. W. Grossman, Discrete Mathematics, Macmillan Pub. Co., 1990
- К. Манев, Увод в дискретната математика, КЛМН, 2005
- Й. Денев, Р. Павлов, Я. Демирович. *Дискретна математика*. Наука и изкуство, София, 1984.

Използвана литература в курса

- Д. Байнов, С. Костадинов, Р. Павлов, Л. Луканова. *Ръководство за решаване на задачи по дискретна математика.* Университетско издателство "Паисий Хилендарски", Пловдив, 1990.
- В.А. Успенский, *Машина Поста*, Москва, Наука, 1988, ISBN 5-02-013735-9.
- L. Lovasz, J. Pelikan, K. Vesztergombi, *Discrete Mathematics – Elementary and Beyond*, Springer Verlag, New York, 2003, ISBN 0-387-95584-4.

Използвана литература в курса

- E. Bender, S. Williamson, *A Short Course in Discrete Mathematics*, Dover, 2006, ISBN 0-486-43946-1.
- P. Linz, An *Introduction to Formal Languages and Automata,* Jones and Bartlett Publishers, 6-th edition, Jones & Bartlett Publishers, ISBN-13: 9781284077247, 2016
- Kenneth H. Rosen, Kamala Krithivasan, Discrete mathematics and its application, McGraw-Hill Companies, 7-th edition, ISBN 978-0-07-338309-5, 2012

Използвана литература в курса

- Owen D. Byer, Deirdre L. Smeltzer, Kenneth L. Wantz, Journey into Discrete Mathematics, AMS, MAA Press, Providence Rhode Island, ISBN 9781470446963, 2018
- Christopher Rhoades, Introductory Discrete Mathematics, Willford Press, ISBN 1682854922, 9781682854921, 2018
- David Liben-Nowell, Discrete Mathematics for Computer Science, Wiley, 2017, ISBN 1119397197, 9781119397199, 2017.
- ***http://www.jflap.org/*** - софтуерна среда