

计算机科学中的数学教育问题

苏运霖

(暨南大学计算机科学系 广州510632)

G434

摘 要

本文回顾了计算机科学中的数学教育的发展历史,论述今日加强数学教育对于培养计算机科学家及发展计算机应用的重要性,由此进一步探讨计算机科学教育中数学教育的新课程设置问题。

一、引 言

今天的绝大多数计算机科学系,特别是那些历史较久的,当年都是从数学系、无线电系或物理系分离出来的。对于计算机科学系的软件专业,则几乎都是从数学系分离出来的。所以,当年这些专业或系对于数学教育的态度,在很大程度上受到其母体的影响。它们强调数学教育,尤其是逻辑训练的重要性。但随着计算机科学技术的发展,计算机科学教育也形成了自己的体系,建立了一系列的计算机科学的课程并得到逐步完善,于是数学在计算机科学教育中的份量有所减少,一度也被有意识地削弱。然而今天当大量的困难问题被提出来时,计算机科学家发现,它们的解决,离开了数学几乎寸步难行。计算机科学教育忽视数学教育的后果不堪设想,要求人们重新考虑这个问题。于是人们又一次把注意力集中到计算机科学中的数学教育问题上,正确地维持数学在计算机科学中的地位,并且为下一代的计算机科学家提供坚实而又适当的数学基础,作为他们攀登的锐利武器。我们感到这个问题对于中国的计算机科学教育尤为重要,因为我们要在计算机科学的发展中走出自己的新路来。本文的目的,也正在于对这个问题提出个人见解,希望能够引起国内同行的注意。

二、计算机科学中早期的数学教育

在国际上,主要在美国,最早的计算机科学系建立于六十年代初。那时的计算机科学教育本身,自然还不可能形成完整的体系,因为当时的各种系统软件都很少,应用领域也相当有限。在这种情况下,对于数学教育的要求自然也还不很明确。然而

由于当时的计算机科学工作者大部分是从数学的母体脱胎的;计算机的基础又是二进逻辑,所以当年的计算机科学教育,对于数学教育的要求很强烈,不但要求接收较为系统的数理逻辑、集合论、图论等逻辑性较强的课程的训练,也要求接收数学分析、实变函数、复变函数等分析课程的训练。当时的许多计算机科学家,都十分自信地宣布自己是数学家。很自然地,他们要求学生也成为既是数学家,也是计算机科学家。当然从一开始,计算机科学与计算机工程就是两个不同的学科分枝,后者以研究计算机的硬件和组成为主要目标。从事计算机工程教学的师资,大部分来自于电子工程(无线电)系或物理系。他们所受的数学教育,自然要比数学家少一些,不会要求在计算机工程教育中偏重数学教育。下边我们将主要讨论计算机科学的数学教育问题,适当时也顺便提及计算机工程的数学教育问题。

我们这里所说的早期,是指ACM课程表68发表以前的阶段。那时,由于大多数今天的计算机科学系都还未成立起来,计算机科学教育的课程体系也还未形成。在这种情况下,数学教育的比重就比较大。这一方面是由于计算机科学家本身的认识使然,但另一方面也是由于外部压力所致。当时,计算机科学作为一门独立的学科,还未被学术界所公认。人们只承认它不过是数学的一个分枝而已。所以十分自然,它的课程的主体部分必须是数学。如果归纳当时的数学教育的特点,可以说有三:1.数量大。由于它是作为数学系的一部分而存在的(即使已经从数学系分出来的那些计算机科学系也一

样),因此自然必须以数学作为主旋律。2.面广。不但要学习离散性的数学课程(人们一开始就感觉到它对计算机科学的重要性),而且还要求学习连续性的数学(当时人们并不清楚它在计算机科学中的应用价值或理论价值)。3.相当大的盲目性。如上述,人们并不了解许多数学分枝到底在计算机科学中有什么应用,然而由于出身于斯,自然要继承此道。

三、计算机科学中数学教育的规范化

六十年代中期,在计算机问世20年之后,计算机的发展已使世人刮目相看,计算机得以跻身于科学之林的条件是:1.计算机的飞速发展,使得无论在硬件、软件和应用方面,计算机都有令人叹为观止的成就。2.在计算机科学领域内,如同数学、物理学等等自然科学领域一样,有了一系列的课程或分枝,如操作系统、数据库、编译技术、数据处理等等,它们已经超越了数学的领域,以致没有人敢再说计算机科学不是一个科学领域了,因为它的研究内容、方法等等,都同数学迥异。3.计算机科学领域内许多杰出的科学研究成果,更加勾勒出计算机科学的特征,展现了计算机科学无限广阔和深奥的研究范畴。面对如此广阔无垠而又深邃无底的无知世界,谁敢狂妄地宣称计算机科学不是堂堂正正的一个学科领域,而只是一个附属于某一学科的小老弟呢?

ACM 的课程表 68 标志着计算机科学的课程体系已经成形,计算机科学已成为一门独立的学科领域。虽然它对有的课程的内容的确定还不很准确,甚至对于一些课程的划分也有可商榷之处,但毕竟它搭起了计算机科学课程体系的框架。几乎与此同时,IEEE 教育委员会所制订的课程表,也为计算机工程的课程体系搭起了框架。在此基础上,两个教育委员会定期讨论计算机科学和计算机工程的课程表,使原课程表不断完善和改进。

按照 ACM 课程表 68 所确定的数学教育,不再着眼于数学,而是根据计算机科学的需要,来考虑数学的教育。因此,一门崭新的数学课程——离散数学——问世了。尽管离散数学所包含的内容,都是数学系原有的课程的内容,但这样一门综合性课程本身的存在,就有特殊的意义。因为它不是集合论、数理逻辑、图论等内容的简单重复或合并,而是根据计算机科学的学科需要所作的有意义的组合。除了离散数学外,数学教育还包括了高等数学、线性代数、概率论等课程。

现在来回顾 ACM 课程表 68,它的历史性功绩是巨大而不可磨灭的。不过它在规定数学课程的内

容和要求上,存在一些不足,主要就是对数学的要求不够高,内容上规定得不够广泛,影响所及,使我国的计算机科学系,对数学的要求也普遍降低。学习课程除了离散数学是普遍开设的之外,很少学校开设数学分析课程,而代之以高等数学。在高等代数方面,多数学校限于开设线性代数。开设概率论课程的系在所有计算机系中的比重不会超过 40%。在数学方面要求较高的学校,几乎成了凤毛麟角。

人们似乎产生了这样的误解,由于计算机科学的对象和性质,它同数学的关系确实和它刚从脱离数学时不同。它同数学的关系就同物理等同数学的关系差不多,而且主要是同离散数学的关系。这一点又同物理系与连续数学关系更密切有所不同。由于这种认识,自然就不会在计算机系去开设更多的数学课程了。

所以我们可以说,这个规范化过去是一个让步,但也存在相对地削弱数学的缺点,导致了计算机科学系毕业生的数学素质的降低。

四、对计算机科学中数学教育的再认识

几乎在 ACM 课程表 68 发表的同时,有人就作为程序设计基础的算法进行了研究,并且把它看作整个计算机科学的核心。对于算法的设计,需要进行质量上的比较以确定它的优劣,因此算法的分析就同算法的设计一起被作为一门课程来提出。对算法的分析,最核心的是对它的实现效率的分析,即时间复杂性分析,尽管有时我们也要考虑它对空间数量的要求,或者称为空间复杂性分析,不过从现况而言,这方面的研究明显地居于次要地位,因为空间问题已经不大成为实现算法的一个障碍。对于计算复杂性的分析,十分自然地涉及了概率的应用。比如对于条件语句,就必须考虑从一条路径上走的概率是多少,而从另一条走的概率又是多少,从而确定出整个计算的复杂性。就所涉及的数学工具而言,那就多种多样了,最突出的特点是大量的连续数学工具的使用,因为对于算法的分析,离散的工具显得较为软弱,需要把一些离散的量当成为连续的量来处理较为有利。单纯的一种或几种连续数学往往也不敷应用,要靠“十八般武艺”的熟练使用,“逢山开路、过河搭桥”,对不同的问题采用适合于该情况的工具,才可奏效。

在我国,对于算法设计与分析这门课是 1982 年在福州召开的计算机科学教育会议上才引起人们普遍的重视的。在那次会上,我作了一次发言,根据

本人所译的百科全书式的D.E. Knuth的名著《计算机程序设计技巧》一书在美国及其它国家所引起的强烈反响,根据算法设计及分析这门课程的地位和作用,论述了在我国开设这门课的必要性和迫切性,建议把它作为必修课列入计算机科学的教学大纲中,这一发言,引起了许多人的广泛注意。在那次会上,还只是建议,有条件的学校可以考虑把算法设计与分析课程当作选修课来开设,并未明确提出就把它作为必修课。但是随后不久,许多学校都纷纷开设了这门课程,并且把它列入必修课。

然而,数学教育并未随着这一课程的开设而相应地进行调整和充实。普通的做法是,在算法分析中需要用到什么数学知识,就讲些什么以补充,或者以承认该数学结论为前提,来进行分析。因此,这样做的结果,必然是:1.所学的数学内容不够系统,而且显得有些支离破碎。一碰到问题,学生就有一种畏俱感,觉得神秘,难以应付。2.由于没有得到系统的严格训练,因此学生感到,即使掌握了所学过的那些东西,但对于分析的要领仍未很好掌握。

所以,我认为,既然算法设计和分析是如此重要,(现在有人甚至建议把这门课程称为算法学,以强调其重要性),它不仅仅是计算机科学的一个分枝,而且乃是计算机科学所有领域,公平地说,乃至于大多数科学,商业和技术的核心。因此,为确保这门课的学习质量,更为了使学生在学这门课的基础上,在数学上打下良好的功底,有必要学习算法分析的数学基础。这门课所包括的数学内容应包括:

- 1) 有关二项式的一些结果。其中包括:
 - 二项式恒等式的推导;
 - 逆关系;
 - 算子演算;
 - 关于调和数的恒等式。
- 2) 递归关系
 - 线性递归关系,包括有限历史和完全历史两方面;
 - 非线性递归关系,包括有关极大和极小函数的关系、连分式、双重指数序列等。
- 3) 其它方法,包括所谓抓小饼怪物方法,级联散列方法以及开放地址散列方法等。
- 4) 渐近分析
 - 基本概念,包括自举、解剖、极限的极限;
 - 斯蒂尔捷积分;
 - 生成函数的渐近式,包括达尔鲍方法、残数演算等。

对于上述各项内容的介绍,不要求包罗万象,而只要求突出重点,抓住要领,使学生懂得它们如何同算法的设计与分析有关,从而掌握进行算法设计与分析时的数学工具。

顺便指出,鉴于算法设计与分析这门课的重要性,在计算机工程中,也应讲授此门课,但所包括的算法可以有所不同。在计算机工程中的算法设计与分析课中,可以讲授一些CAD的算法、VLSI的设计算法,等等。对于这些,讲授有关的数学,也是有益的。当然,份量上可以略为减少一些。

五、全面加强数学教育,适应计算机科学的发展

今天计算机科学本身的发展,以及它的应用,要求我们对数学教育进行一番全面的审核,根据计算机科学本身发展和应用的需要,对它作全面的增强。

先从计算机科学本身的发展来看,今天作为计算机科学核心的算法学,要考虑的不仅仅是作为原来学科分枝的一些基本算法,而且还要考虑新的学科分枝一如分布式系统和并行计算中的算法。对于这些算法的研究——无论设计还是分析,都要求更高的数学素质和知识。这些数学知识包括图论、排队论、规划论、博弈论等等。计算机科学本身提出了一系列崭新的数学问题,要求人们去解决。这也就是说,计算机科学反过来成为一些数学分枝发展的源泉。因此,我们上边提出的一些数学分枝,仅仅是其数学范畴,但它们未必就能提供对于各种问题的现成的答案和方法。我们加强数学教育,旨在为处理这些问题作些准备或打下基础。

从计算机的应用来说,计算机通向外界的渠道是模拟,因为作为使用计算机解决实际问题的头一个步骤,是建立起该问题的数学模型。大千世界,现实问题五花八门,因此它的数学模型所涉及的数学也就不会是很有限的。它们既涉及离散数学,也涉及连续数学;既涉及确定性的数学,也涉及不确定性的数学;既涉及线性的、也涉及非线性的。如当前人们所关注的混沌学,是指数组或属性的一种毫无秩序、顺序作用的全盘混乱状态,这种状态的数学表达即是非线性的。因此为了推广计算机的应用,为了使具有使用计算机解决实际问题的潜在能力,就需要使今天的数学教育覆盖上述几方面的内容。

肯定地说,我们提出这样的主张,会受到一些人的非议,他们会提出反对的种种理由。我们设想,

他们反对的理由不外乎是下面这些。

没有必要。他们无非认为“计算机科学今天的数学教育已经够用了，没有必要去自找麻烦。针对这个观点，我觉得引用一下Dijkstra的一句话是有用的。“关键的问题在于，大学应当为社会提供什么样的领袖呢，还是仅仅按照社会的要求来培训人才？如果是前者，那所大学就应当根据发展的情况，去调整、更新和充实自己的教学内容。大学甚至应该去开创人类新的知识，即先于客观的需要而开拓、创造人类新的精神文明。如果是这样，那些没有必要的理由就站不住脚了。”

办不到。这种理由似乎也赞成作些改进、调整或充实。但他们认为要按我们上边的观点那样进行全面的充实和加强，则太多了，在有限的大学教育期间办不到。确实，如果要把我们提到的所有数学分枝，把各个分枝的所有内容都安排到大学四年期间内，无疑会大大加重学生的负担。这样当然办不到。

没有区别，就没有政策。我们提出全面加强计算机科学中的数学教育，只能是根据大学四年中学生的实际负担所能承受的程度，来考虑我们的安排。完全作为独立的一门一门课程来把需要的各个数学分枝加到计算机科学的课程表中去，显然负担太重。因此我建议新加一门课程，叫做连续数学，

它可以像离散数学那样，集各种连续性的数学于一体，内容可以取自复变函数、实变函数、泛函分析、概率论、排队论、规划论等。这作为对计算机科学数学教育加强的具体措施之一。然后在研究生教育中，则可根据所攻读的方向，再进一步加大有关数学内容的份量。这样，就可以解决增加份量过大，超过学生承受能力的问题。

太数学化了。这又是一种似是而非的反对意见。我们无意使计算机科学又回到数学去，也无意和不可能去把今天或明天的学生培养成既是数学家又是计算机科学家，或者既是后者又是前者；我们的目标却是，使计算机系的学生具有良好的数学功底。如果这就是所谓的太数学化，那我们认为绝对需要。为了实现这样一个目标，我们不仅要在计算机系里开设上边提到的课程，还应该把数学和逻辑贯穿于一些计算机的课程中。如果说Dijkstra的一些观点有些偏颇之处，他对于在程序设计课程中就加强对学生的逻辑思维的训练这一观点是很有见地和正确的。要向学生们强调，程序员的任务不仅是把程序写出来，而且应尽力形式地证明，所写程序满足形式的功能说明。一旦他们接受了这一训练，他们就为能力的飞跃打下良好的基础。这也就是我们探索计算机科学的数学教育所要追求的目标。

(参考文献略)

(接第29页)

附录 启发式信息的定义

启发式信息的定义是基于统计学中线性相关系数的概念的，作者做了一点改动。

1°. 单变元情况下启发式信息 r 定义如下：

$$r = \frac{\sum_{i=1}^m (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^m (x_i - \bar{x})^2 \sum_{i=1}^m (y_i - \bar{y})^2}}, \text{其中:}$$

$$\bar{x} = \frac{1}{m} \sum_{i=1}^m x_i, \quad \bar{y} = \frac{1}{m} \sum_{i=1}^m y_i$$

2°. 多变元情况下启发式信息 r 定义如下：

$$r = \sqrt{\sum_{i=1}^n l_{i0} b_i} / \sqrt{l_{00}}$$

其中：

$$l_{ij} = \sum_{k=1}^m (x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j), (i, j = 1, \dots, n)$$

$$l_{i0} = \sum_{k=1}^m (x_{ik} - \bar{x}_i)(y_k - \bar{y}), (i = 1, \dots, n)$$

$$l_{00} = \sum_{k=1}^m (y_k - \bar{y})^2$$

$$\bar{y} = \frac{1}{m} \sum_{k=1}^m y_k$$

$$\bar{x}_i = \frac{1}{m} \sum_{k=1}^m x_{ik}$$

$$l_{11} \dots l_{1n}$$

$$L = \dots \dots \dots$$

$$l_{n1} \dots l_{nn}$$

$$\Sigma = L^{-1} L_0$$

$$L_0 = (l_{10}, \dots, l_{n0})^*$$

$$L^{-1} = \begin{bmatrix} c_{11} & \dots & c_{1n} \\ \vdots & & \vdots \\ c_{n1} & \dots & c_{nn} \end{bmatrix}$$

$$B = (b_1, \dots, b_n)^*$$