"Ruolo della Matematica (e della Ricerca Operativa) nella formazione Informatica"

Antonio Sassano (Università di Roma "La Sapienza")
Intervento alla Tavola Rotonda in occasione dell'Inaugurazione della nuova sede del DIS – Pisa
12 Giugno 2003

"*The idea behind teaching is to expect students to learn why things are true, rather than have them memorize ways of solving a few problems*...(Israel Gel'fand)

La tendenza alla specializzazione è certamente la nostra principale "malattia professionale". Questa tendenza ci porta spesso a dichiarare di "non saper nulla" di tematiche che sono adiacenti a quelle alle quali abbiamo dedicato la nostra intera vita scientifica.

Questa tendenza raggiunge il suo culmine quando si riceve un invito a partecipare ad una tavola rotonda. Leggiamo il tema e ci domandiamo immediatamente quale sia la nostra specifica competenza a rispondere alle questioni poste dagli organizzatori. Il riflesso condizionato della specializzazione scatta e la prima tendenza è quella di declinare l'invito, dichiarandosi incompetenti a trattare il tema proposto.

Responsabilità e cortesia intervengono però (quasi sempre) e ci spingono ad accettare il gentile invito e a porci il problema di quali contributi concreti la nostra esperienza possa apportare alla discussione. Qui interviene la seconda "malattia professionale" (più grave tra gli ingegneri!): quella della ricerca della documentazione. Poiché riteniamo di essere "non qualificati" a rispondere, rifuggiamo dall'idea di dire le cose che, sull'argomento, affiorano per prime alla nostra mente (magari quelle stesse cose che diciamo abitualmente con convinzione e veemenza ai nostri colleghi). Al contrario, ci sforziamo di "ancorare" ogni nostra affermazione al solido e oggettivo porto dei "dati". Inizia così la febbrile ricerca dei dati statistici o dei documenti di posizione messi a punto dalle organizzazioni scientifiche internazionali.

Spesso, oltre a seguire con puntuale regolarità questa "via crucis" da tavola rotonda, aggiungiamo una terza deformazione (più grave tra i matematici) che ci spinge alla maniacale ricerca di una o due "questioni fondamentali" alle quali dedicare l'intervento. Va da sé che, immancabilmente, le "questioni fondamentali" alle quali la nostra ricerca (corroborata da statistiche e documenti internazionali) giunge, non sono altro che la razionalizzazione di quelle stesse idee che avremmo espresso se provocati sulla questione durante un "coffee break".

Quel che è grave è che, quasi sempre, questo risultato finale ci tranquillizza.

Il tema di oggi: "Informatica e Formazione: Fondamenti, Tecnologia e Applicazioni" ha ovviamente provocato in me la serie di reazioni che ho appena descritto.

L'argomento è certamente parte integrante della nostra esperienza quotidiana di docenti e ricercatori. In particolare, in una fase di convulsa trasformazione degli ordinamenti didattici come quella che attraversiamo (e soffriamo), la questione dell'adeguamento della formazione (universitaria e non) all'evoluzione della tecnologia e dell'organizzazione del lavoro costituisce uno dei temi principali della nostra riflessione.

Tuttavia, il mio "non essere" informatico in senso stretto e il timore di avere posizioni personali "eccentriche" sulle questioni dell'interazione tra Informatica, Ricerca Operativa e Matematica Applicata mi ha spinto immediatamente a ricercare l'oggettività di dati e documenti.

Dati statistici e documenti internazionali sulla formazione informatica e sulla sua evoluzione sono disponibili in quantità industriali e pongono più un problema di selezione che uno di carenza di informazioni. La mia scelta è caduta su quello che ritengo sia lo sforzo più autorevole e documentato oltre che, a mia conoscenza, più recente. Si tratta del documento ***"CC-2001 Computing Curricula -***

Computer Science " messo a punto dall'**ACM** e dall'**IEEE-CS.** Questo rapporto appartiene ad una serie realizzata dalle due maggiori organizzazioni scientifiche e professionali nel campo della Computer Science. Tra i vari rapporti dedicati ad aree specifiche dell'Informatica (Computer Engineering, Software Engineering, Information Systems) quello preso in considerazione è, a mio parere, il più generale e il più adatto a fornire un quadro di tendenza della formazione informatica a livello internazionale.

E' importante dire con chiarezza (come d'altra parte fanno, correttamente, gli autori del documento) che le conclusioni del documento CC-2001 sono fortemente condizionate dal panorama universitario e industriale del Nord-America e che le proposte operative sono ispirate a scenari tipicamente statunitensi e canadesi. Tuttavia, per individuare tendenze e ricavare chiavi di lettura della nostra situazione nazionale credo che questo documento sia comunque di grandissimo interesse.

Veniamo ora alle "questioni fondamentali" da approfondire in questo intervento. I miei personali interessi e il mio ruolo di Presidente del Centro Interuniversitario di Ricerca Operativa (CIRO) indirizzano in modo naturale la scelta verso le questioni poste dall'interazione tra Ricerca Operativa e Informatica.

Il motivo che rende interessante tali questioni è, a mio parere, la doppia natura della Ricerca Operativa. Da un lato il suo essere un settore della Matematica Applicata e, quindi, l'avere un ruolo fondamentale e di importanza crescente nella formazione di base degli informatici. Dall'altro, quello di avere un ruolo di "gateway" tra l'informatica e la scienza dell'organizzazione e, quindi, di avere un ruolo di finalizzazione applicativa nei curricula informatici.

Questa duplice natura è particolarmente interessante nell'ottica della "questione fondamentale" che, tra le tante possibili, voglio affrontare in questo intervento: *L'evoluzione dell'insegnamento delle materie matematiche di base e il ruolo particolare svolto, tra queste ultime, dalla Ricerca Operativa.*

L'Autosufficienza culturale dell'Informatica

Il documento CC-2001 ha analizzato i vari problemi della formazione informatica grazie al lavoro di una serie di "focus group" pedagogici. Il "focus group" 2 (PFG2) ha avuto come obiettivo quello di individuare argomenti e corsi "esterni" all'informatica che avessero un ruolo nella formazione "undergraduate".

Due sono, a mio parere, i punti significativi del mandato del PFG2 (che riporto più sotto). Il primo è che la matematica viene considerata una materia integrativa al pari delle materie scientifiche, di quelle dell'ingegneria, dell' economia, della scrittura tecnica, del parlare in pubblico e della gestione dei progetti. In altre parole: a) *la matematica è "esterna" alla "Computer Science"*; b) *il suo è un ruolo di completamento culturale e non di formazione di competenze.*

Il secondo punto interessante del mandato è complementare e correttivo rispetto a quello che abbiamo appena visto. Infatti al punto e) il mandato è quello di sviluppare modelli di curricula che soddisfino gli obiettivi formativi ***integrando*** nei corsi di informatica alcuni dei contenuti di supporto. In altre parole: *la matematica (e le altre materie!) possono avere un ruolo nella formazione delle competenze (e non soltanto nel completamento culturale) ma allora sarà bene assorbirne i contenuti nei corsi di informatica.*

PFG2. Supporting topics and courses

a. Specify a set of educational goals outside of traditional computer science that support undergraduate computer science education, such as mathematics, engineering, science, technical writing, public speaking,

economics, project management, and so forth.

b. Identify a minimal list of supporting topics deemed essential to any undergraduate computer science curriculum regardless of the nature of the institution.

c. Present suggestions for additional supporting topics beyond that minimum that may vary depending on the type of institution, the populations an institution serves, and the number of courses which the institution is allowed to include in a program.

d. Develop specifications for one or more sets of non-CS courses that satisfy these goals.

e. Develop one or more models for satisfying some or all of these goals by integrating them into computing courses.

Fedele al mandato ricevuto, il "focus group" ha individuato negli *strumenti* (si noti il termine) della *matematica discreta* i contenuti matematici indispensabili per tutti gli studenti di informatica. Coerentemente al punto e), la raccomandazione del CC-2001 (che riporto sotto) è quella di inserire nei corsi di informatica gli strumenti della matematica discreta.

....
The CC2001 Task Force makes the following recommendations with respect to the mathematical content of the computer science curriculum:

• *Discrete mathematics. All students need exposure to the tools of discrete mathematics. When possible, it is best for students to take more than one course in this area, but all programs should include enough exposure to this area to cover the core topics in the DS area.* ***Strategies for integrating discrete mathematics into the introductory curriculum are discussed in section 7.4.***

• *Additional mathematics. Students should take additional mathematics to develop their sophistication in this area. That mathematics might consist of courses in any number of areas including statistics, calculus, linear algebra, numerical methods, number theory, geometry, or symbolic logic. The choice should depend on program objectives, institutional requirements, and the needs of the individual student.*

Ci troviamo quindi di fronte all'esplicitazione di un atteggiamento culturale di ***autosufficienza dell'informatica*** le cui tracce sono riscontrabili, a mio parere, anche nella maggioranza dei curricula triennali delle nostre facoltà di ingegneria e di scienze. Schematizzo le caratteristiche di questo atteggiamento:

1. La matematica non deve essere, al contrario della capacità di programmare o delle conoscenze base di "Computer Graphics" o di teoria della Basi Dati, una competenza fondamentale degli studenti "undergraduate" in informatica. Solo alcuni contenuti matematici sono rilevanti (si vedano gli argomenti sottolineati nella figura della pagina seguente).

Figure 5-1. Computer science body of knowledge with core topics underlined

DS. Discrete Structures (43 core hours)
- DS1. Functions, relations, and sets (6)
- DS2. Basic logic (10)
- DS3. Proof techniques (12)
- DS4. Basics of counting (5)
- DS5. Graphs and trees (4)
- DS6. Discrete probability (6)

PF. Programming Fundamentals (38 core hours)
- PF1. Fundamental programming constructs (9)
- PF2. Algorithms and problem-solving (6)
- PF3. Fundamental data structures (14)
- PF4. Recursion (5)
- PF5. Event-driven programming (4)

AL. Algorithms and Complexity (31 core hours)
- AL1. Basic algorithmic analysis (4)
- AL2. Algorithmic strategies (6)
- AL3. Fundamental computing algorithms (12)
- AL4. Distributed algorithms (3)
- AL5. Basic computability (6)
- AL6. The complexity classes P and NP
- AL7. Automata theory
- AL8. Advanced algorithmic analysis
- AL9. Cryptographic algorithms
- AL10. Geometric algorithms
- AL11. Parallel algorithms

AR. Architecture and Organization (36 core hours)
- AR1. Digital logic and digital systems (6)
- AR2. Machine level representation of data (3)
- AR3. Assembly level machine organization (9)
- AR4. Memory system organization and architecture (5)
- AR5. Interfacing and communication (3)
- AR6. Functional organization (7)
- AR7. Multiprocessing and alternative architectures (3)
- AR8. Performance enhancements
- AR9. Architecture for networks and distributed systems

OS. Operating Systems (18 core hours)
- OS1. Overview of operating systems (2)
- OS2. Operating system principles (2)
- OS3. Concurrency (6)
- OS4. Scheduling and dispatch (3)
- OS5. Memory management (5)
- OS6. Device management
- OS7. Security and protection
- OS8. File systems
- OS9. Real-time and embedded systems
- OS10. Fault tolerance
- OS11. System performance evaluation
- OS12. Scripting

NC. Net-Centric Computing (15 core hours)
- NC1. Introduction to net-centric computing (2)
- NC2. Communication and networking (7)
- NC3. Network security (3)
- NC4. The web as an example of client-server computing (3)
- NC5. Building web applications
- NC6. Network management
- NC7. Compression and decompression
- NC8. Multimedia data technologies
- NC9. Wireless and mobile computing

PL. Programming Languages (21 core hours)
- PL1. Overview of programming languages (2)
- PL2. Virtual machines (1)
- PL3. Introduction to language translation (2)
- PL4. Declarations and types (3)
- PL5. Abstraction mechanisms (3)
- PL6. Object-oriented programming (10)
- PL7. Functional programming
- PL8. Language translation systems
- PL9. Type systems
- PL10. Programming language semantics
- PL11. Programming language design

Note: The numbers in parentheses represent the minimum number of hours required to cover this material in a lecture format. It is always appropriate to include more.

HC. Human-Computer Interaction (8 core hours)
- HC1. Foundations of human-computer interaction (6)
- HC2. Building a simple graphical user interface (2)
- HC3. Human-centered software evaluation
- HC4. Human-centered software development
- HC5. Graphical user-interface design
- HC6. Graphical user-interface programming
- HC7. HCI aspects of multimedia systems
- HC8. HCI aspects of collaboration and communication

GV. Graphics and Visual Computing (3 core hours)
- GV1. Fundamental techniques in graphics (2)
- GV2. Graphic systems (1)
- GV3. Graphic communication
- GV4. Geometric modeling
- GV5. Basic rendering
- GV6. Advanced rendering
- GV7. Advanced techniques
- GV8. Computer animation
- GV9. Visualization
- GV10. Virtual reality
- GV11. Computer vision

IS. Intelligent Systems (10 core hours)
- IS1. Fundamental issues in intelligent systems (1)
- IS2. Search and constraint satisfaction (5)
- IS3. Knowledge representation and reasoning (4)
- IS4. Advanced search
- IS5. Advanced knowledge representation and reasoning
- IS6. Agents
- IS7. Natural language processing
- IS8. Machine learning and neural networks
- IS9. AI planning systems
- IS10. Robotics

IM. Information Management (10 core hours)
- IM1. Information models and systems (3)
- IM2. Database systems (3)
- IM3. Data modeling (4)
- IM4. Relational databases
- IM5. Database query languages
- IM6. Relational database design
- IM7. Transaction processing
- IM8. Distributed databases
- IM9. Physical database design
- IM10. Data mining
- IM11. Information storage and retrieval
- IM12. Hypertext and hypermedia
- IM13. Multimedia information and systems
- IM14. Digital libraries

SP. Social and Professional Issues (16 core hours)
- SP1. History of computing (1)
- SP2. Social context of computing (3)
- SP3. Methods and tools of analysis (2)
- SP4. Professional and ethical responsibilities (3)
- SP5. Risks and liabilities of computer-based systems (2)
- SP6. Intellectual property (3)
- SP7. Privacy and civil liberties (2)
- SP8. Computer crime
- SP9. Economic issues in computing
- SP10. Philosophical frameworks

SE. Software Engineering (31 core hours)
- SE1. Software design (8)
- SE2. Using APIs (5)
- SE3. Software tools and environments (3)
- SE4. Software processes (2)
- SE5. Software requirements and specifications (4)
- SE6. Software validation (3)
- SE7. Software evolution (3)
- SE8. Software project management (3)
- SE9. Component-based computing
- SE10. Formal methods
- SE11. Software reliability
- SE12. Specialized systems development

CN. Computational Science (no core hours)
- CN1. Numerical analysis
- CN2. Operations research
- CN3. Modeling and simulation
- CN4. High-performance computing

2. La matematica rilevante nell'informatica è la *matematica discreta*. Ma non la matematica discreta in tutto il suo dispiegarsi, i suoi rapporti con l'algebra e la geometria, la teoria

dei grafi e la teoria dei numeri (che infatti appartengono alla "matematica addizionale"), l'ottimizzazione combinatoria e la simulazione (che vengono classificate nelle "Computational Science"). Piuttosto, gli *strumenti della matematica discreta*: le tecniche di prova, gli algoritmi fondamentali, le strutture dati, la probabilità discreta.

3. Questa "versione ridotta" della matematica discreta può essere insegnata all'interno dei corsi di informatica e ridotta ad un ruolo strumentale ("tool").
4. Il resto delle competenze matematiche viene relegato ad un ruolo di pura formazione culturale dello studente (magari in un ruolo di supporto alla "cultura scientifica").
5. Il settore della matematica che comprende Modellistica, Simulazione e Ricerca Operativa, e che viene classificato all'interno del "corpus" delle competenze informatiche come "Computational Science", non ha spazio nella formazione delle competenze fondamentali.

Credo che la prima reazione della maggior parte di noi di fronte a questa schematica rappresentazione sia di rifiuto e di orgogliosa ri-affermazione di una specificità italiana ed europea nella formazione informatica e, più in generale, scientifica.

Ma no! La matematica ha un ruolo fondamentale nella formazione degli ingegneri e degli informatici! Abbiamo una serie di esami di analisi, geometria e probabilità che caratterizzano i nostri primi di anni di corso e che hanno esattamente lo scopo di formare e fornire competenze ai nostri studenti della laurea breve!

Voglio brevemente rispondere a queste obiezioni.

Parlerò della realtà di ingegneria che conosco meglio. I corsi di base di matematica sono un retaggio della formazione di base degli ingegneri del secolo scorso (ahimè!). Si trattava di competenze "vive" che gli ingegneri utilizzavano per comprendere gli aspetti formali del "corpus" di conoscenze indispensabili per la loro professione. Ingegneri civili, idraulici, meccanici ma anche elettrotecnici ed elettronici utilizzavano quelle competenze per descrivere e studiare i fenomeni che dovevano governare.

La trasformazione indotta dallo sviluppo delle tecnologie dell'informazione e, in particolare, dell'informatica ha toccato solo marginalmente questa situazione per quasi tutte le specializzazioni dell'ingegneria.

Per gli informatici, invece, la trasformazione è stata drammatica. La maggior parte delle competenze matematiche acquisite nei corsi dei primi anni non sono utilizzate nei corsi successivi e se lo sono, lo sono in corsi "esterni" al flusso principale (come chimica, fisica, automatica ed elettronica). Al contrario, competenze importanti di matematica discreta, di ottimizzazione combinatoria e di logica non sono oggetto di corsi dedicati e, nel tempo, sono state acquisite dagli studenti mediante più o meno lunghi "detours" all'interno dei corsi di informatica propriamente detti.

La conseguenza è che il concetto di relazione è inestricabilmente associato a quello di base dati relazionale e il calcolo dei predicati o la programmazione intera sono entrambi capitoli della "constrained programming".

Come si vede, la situazione era perfettamente matura per il nascere e lo svilupparsi di quell'atteggiamento che ho definito di *autosufficienza dell'informatica*. Conoscenze che non vengono "vivificate" dall'uso divengono, inevitabilmente, patrimonio culturale dello studente e smettono di essere una competenza operativa. Gli strumenti utili vengono invece assorbiti e identificati con le loro applicazioni. Il passo successivo è quello della scomparsa della propedeuticità dei corsi di base e della tendenza dei corsi specifici dell'informatica a divenire auto-contenuti.

Voglio aggiungere che, nello specifico caso italiano, la nascita delle lauree brevi, con lo spostamento

ai primi anni di alcuni corsi professionalizzanti, ha accelerato questo processo. La prospettiva, in assenza di interventi correttivi, è esattamente quella prospettata dal CC-2001:

1 Marginalizzazione della matematica di base, con una riduzione del numero dei corsi.
2 La tendenza dei docenti di matematica, con una scelta anch'essa auto-referenziale, a mantenere invariati i programmi in presenza di una diminuzione del numero di ore disponibili. Questa tendenza implica, coadiuvata dall'effetto sinergico della richiesta di maggior "clemenza", la scelta frequente di dimostrare solo un numero limitato dei teoremi presentati.
3 Un sostanziale disinteresse dei docenti dei corsi professionalizzanti all'effettivo apprendimento da parte degli studenti (frequente è la frase "in una lezione insegnerò io quello che serve sul serio") con la conseguente spinta all'abolizione della propedeuticità.
4 L'assorbimento di alcuni argomenti all'interno dei corsi informatici, con la conseguente riduzione ad un ruolo strumentale delle competenze matematiche.

Tutto questo trasforma, spesso, i corsi matematici di base in una sequenza di definizioni (anche i teoremi divengono tali!) che non vengono più utilizzate nei corsi successivi e contribuiscono, al pari della fisica e della chimica, alla creazione di una generica "forma mentis" scientifica dello studente. Contestualmente, le parti "vive" della matematica vengono assorbite nei corsi professionalizzanti.

Il documento C-2001, con pragmatismo nord-americano, non fa altro che prendere atto di questa evoluzione e definire un percorso formativo coerente ad essa.

Cosa fare dunque? La prima domanda da porsi è se una tale evoluzione sia auspicabile.

Per quanto detto, la risposta sembra essere positiva. Come studioso di ottimizzazione combinatoria debbo accogliere con piacere il (tardivo) riconoscimento del ruolo cruciale della matematica discreta nella formazione informatica (e, in generale, ingegneristica). Inoltre, abbiamo visto come l'attuale insegnamento delle materie matematiche di base non sia più adatto all'evoluzione dell'informatica. Molti dei contenuti non sono più ripresi nei corsi successivi, molti risultati sono presentati senza dimostrazione e le materie matematiche di base sembrano trovare maggiori giustificazioni alla propria esistenza nella fisica che nelle materie dell'informatica.

Malgrado tutto questo, ritengo estremamente pericolosa la certificazione, compiuta dal documento CC-2001, che quella discreta sia l'unica matematica "rilevante" nella formazione informatica e che il suo destino debba essere quello di "diluirsi" nei corsi informatici.

Innanzitutto, la ricerca e le più recenti evoluzioni tecnologiche ci dicono che argomenti come, ad esempio, la programmazione non-lineare e la teoria dell'approssimazione hanno un ruolo fondamentale nella teoria delle Reti, nello studio del Web e delle Basi di Dati ("Data Mining"). Quelle competenze matematiche sono, infatti, indispensabili per comprendere e progettare strumenti informatici efficienti e il non possederle esclude lo studente da una reale comprensione del ruolo e delle potenzialità di quegli strumenti.

Inoltre, l'insegnamento della matematica discreta all'interno dei corsi informatici tende inevitabilmente ad esaltare il ruolo strumentale e/o puramente descrittivo di quest'ultima e a minimizzarne le potenzialità conoscitive.

In conclusione, a mio parere una soluzione meno orientata all'autosuficienza e diversa da quella rilevabile nell'evoluzione dei nostri corsi di laurea e nel documento CC-2001 sarebbe auspicabile.

L'obiettivo di questo intervento era quello di fotografare una tendenza e non quello di indicare soluzioni operative. Tuttavia, credo si possa convenire sui punti seguenti:

1. I corsi di base debbono essere certamente rivisti; debbono mantenere i loro spazi

"classici" (o anche aumentarli) ma concentrarsi su un numero limitato di concetti e proprietà fondamentali.

2. La scelta dei concetti e delle proprietà da proporre nel nucleo fondamentale deve essere effettuata, con una procedura "a ritroso", a partire dalle esigenze dei corsi professionalizzanti che ne fanno uso. Questa procedura tende, infatti, a selezionare porzioni della matematica "vive" e utili per la formazione delle professionalità informatiche.
3. I concetti presentati in questo nucleo fondamentale debbono essere illustrati in modo completo e lasciando un tempo adeguato di maturazione agli studenti.
4. I corsi non debbono diventare sequenze di definizioni ma tutti i teoremi debbono essere dimostrati in modo formale, sottolineando le caratteristiche delle tecniche di prova (questo è richiesto anche nel documento CC-2001). Si dovrebbe tentare di mettere in condizione lo studente di derivare in modo autonomo semplici proprietà degli oggetti matematici del corso.
5. Inevitabilmente, la matematica discreta dovrà avere un ruolo importante nella formazione degli informatici. Tuttavia, è indispensabile che i concetti della matematica discreta vengano presentati in corsi appositi e che le interazioni con gli altri settori della matematica (algebra e geometria soprattutto) vengano approfonditi e chiarificati.
6. La matematica discreta non è l'unica "matematica rilevante" per la formazione di base in informatica. La programmazione non-lineare, la teoria dell'approssimazione, la statistica, (assieme ai contenuti di base dell'analisi e della geometria necessari al loro insegnamento) debbono necessariamente far parte della formazione di uno specialista informatico.
7. I corsi di matematica debbono essere ripresi alla fine del triennio (per chi decide di proseguire) o all'inizio del biennio specialistico per integrare le conoscenze con i contenuti eliminati dai corsi dei primi anni.