La Matematica nella Società e nella Cultura

Rivista dell'Unione Matematica Italiana

Guido Gherardi

Paradigmi di computazione per i numeri reali

La Matematica nella Società e nella Cultura. Rivista dell'Unione Matematica Italiana, Serie 1, Vol. **1** (2008), n.3, p. 525–554.

Unione Matematica Italiana

<http://www.bdim.eu/item?id=RIUMI_2008_1_1_3_525_0>

La Matematica nella Società e nella Cultura. Rivista dell'Unione Matematica Italiana, Unione Matematica Italiana, 2008.

La Matematica nella Società e nella Cultura
Rivista dell'Unione Matematica Italiana
Serie I, Vol. I, Dicembre 2008, 525-554

Paradigmi di computazione per i numeri reali

GUIDO GHERARDI

Negli anni '30 del secolo appena trascorso alcuni logici e matematici hanno schiuso le porte all'era informatica ponendo le basi teoriche della teoria della computazione. Gli strumenti concettuali sviluppati da questi ricercatori sono stati tramutati con grande successo in realizzazioni ingegneristiche, dando luogo alla rivoluzione informatica che tutti conosciamo. La teoria della computabilità per i numeri naturali ha ben presto ottenuto una sistemazione largamente accettata, ben prima che i calcolatori elettronici venissero fisicamente realizzati. Tuttavia, sin dalle sue origini, la teoria della computabilità ha rivolto la propria attenzione anche ai numeri reali. Affronteremo ora brevemente la storia di questa seconda avventura concettuale, prendendone in esame alcune delle tappe fondamentali. Tuttavia, nell'affrontare un tema così vasto, dobbiamo necessariamente effettuare una selezione degli argomenti da trattare. Ci sembra che la questione fondamentale da risolvere sia la seguente: quale significato è stato storicamente attribuito all'espressione "*funzione reale computabile*"? A tale domanda non si può in realtà rispondere in modo univoco, giacché sono state suggerite diverse possibili interpretazioni. Di conseguenza, ci limiteremo a passare in rassegna le principali proposte, analizzandone differenze ed analogie.

In particolare, porremo l'accento sulla fondamentale contrapposizione tra due approcci antagonisti, quelli denominati TTE e real-RAM (e modelli a loro affini). Tale disputa è particolarmente interessante dal punto di vista concettuale e filosofico, in quanto dimostra che la discussione sui fondamenti dell'analisi computazionale non è ancora terminata, e anzi è tuttora di viva attualità.

Sebbene infatti si siano storicamente avvicendate diverse scuole, ciascuna con un proprio concetto di computabilità, la maggior parte di esse propone modelli analoghi, anche se non del tutto coincidenti. Tuttavia, possiamo classificare i vari modelli in due gruppi, contrapposti nelle basi concettuali e nei risultati conseguiti, e a tali due gruppi appartengono in modo emblematico, rispettivamente, il paradigma TTE e quello real-RAM. I due manuali "canonici" su cui oggi si apprendono le loro basi tecniche ([1], [14]) contengono entrambi una parte apologetica e al tempo stesso polemica, sulle proprie motivazioni e sulle presunte debolezze degli approcci antitetici.

Il modello TTE si colloca all'interno di una lunga tradizione, i cui epigoni verranno presentati in seguito, ma vede la sua sistemazione definitiva nella teoria delle rappresentazioni di K. Weihrauch. Questa pone l'accento sulla necessità di codificare gli oggetti matematici tramite stringhe di bit, affinché sia possibile la loro manipolazione da parte di un calcolatore. Le stringhe in questione sono sovente di lunghezza infinita, giacché non è possibile in generale compattare in un codice finito la quantità di informazione contenuta in un numero reale. I caratteri (bit) utilizzati per le codifiche, nonché le operazioni eseguite da una macchina che operi su tali successioni simboliche, sono elementi atomici, indivisibili. Gli approcci affini a quello TTE tendono ad estendere in modo diretto il vocabolario, i metodi, ed i risultati della teoria classica della computabilità sui numeri naturali al campo dei numeri reali. Questi sono stati quindi considerati gli approcci "discreti" alla questione. A questi si contrappone l'approccio real-RAM, che dichiara di ispirarsi maggiormente al calcolo effettuato dagli analisti negli ambiti della fisica matematica e dell'analisi numerica. Come vedremo, è lo spirito di alcuni lavori di Newton sulla trattazione del continuo matematico ad essere preso come suo punto di riferimento ideale. Il modello real-RAM viene spesso tuttavia accusato di essere inverosimile e non simulabile a livello pratico da alcun sistema fisico. Di contro, viene ribattuto che l'utilizzo di codifiche digitali nella formulazione di una teoria della computabilità sui numeri reali produca risultati di scarso interesse per la pratica matematica. E tuttavia tale critica è mossa da un teoria che non riconosce pienamente la computabilità di alcune fondamentali operazioni, quotidianamente eseguite

da calcolatrici tascabili, e ancor prima, da matematici di tutte le generazioni, quali l'estrazione di radice e l'elevamento a potenza. Operazioni per le quali, al contrario, l'approccio TTE fornisce una teoria esaustiva.

L'approccio TTE d'altro canto non trova il suo fondamento unicamente nella teoria della computabilità, ma anche nella topologia classica. In particolare, il concetto informatico di *computabilità* e quello topologico di *continuità* sono strettamente interrelati all'interno della teoria, come avremo modo più volte di osservare.

L'ultima parte dell'articolo viene data a titolo di approfondimento e può comunque essere tralasciata dal lettore senza che la visione complessiva del discorso ne sia compromessa. In questa parte ci occuperemo, specularmente, della nozione di "funzione reale non computabile", secondo l'analisi dettagliata e raffinata che l'approccio TTE sta attualmente conducendo. Sulla base di quanto accennato sopra, non c'è da sorprendersi che lo studio delle condizioni di incomputabilità delle funzioni sia strettamente legato all'analisi del loro grado di discontinuità.

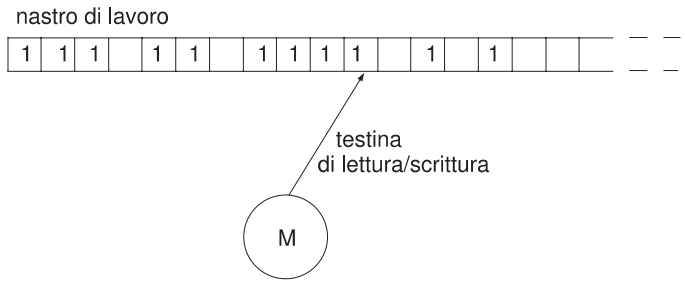
Anche su questo punto constateremo come il dibattito sull'interpretazione di alcuni interessanti fenomeni sia in corso, e sia quindi di stimolante interesse scientifico e filosofico.

1. – Le macchine di Turing.

Sul numero di Novembre 1936 dei "Proceedings della London Mathematical Society" appare lo storico articolo di Alan Turing *On computable numbers, with an application to the "Entscheidungsproblem"*. In questo vengono gettate le basi della moderna teoria della "*ricorsività*", che ha come oggetto di studio le leggi universali della computazione. Sebbene l'articolo fornisca una teoria della computabilità per i numeri naturali divenuta universalmente condivisa dalla comunità scientifica, le effettive intenzioni di Turing sono piuttosto quelle di analizzare il concetto di numero reale computabile. Ciononostante, tale ambito di ricerca viene relegato in secondo piano per diversi decenni dagli addetti ai lavori, periodo nel quale la teoria delle computabilità

sui numeri naturali conosce al contrario sviluppi prodigiosi (per un compendio sull'argomento, vedasi [6] e [7]).

La definizione data da Turing nel 1936 di numero reale computabile è forse la più intuitiva: un numero reale è computabile se il suo sviluppo decimale può essere generato da un processo meccanico ([1]). Certamente, bisogna comprendere cosa si intenda per "processo meccanico", e la fama dell'articolo di Turing è dovuto proprio a tale spiegazione. Turing descrive infatti un tipo di macchina ideale, che oggi prende il suo nome, costituito da un nastro a infinite celle, una testina di lettura e scrittura, ed un insieme finito di istruzioni ([2]).



Una macchina di Turing schematizzata.

Tale macchina manipola successioni di simboli da un alfabeto (che per quel che ora ci riguarda può contenere anche un solo simbolo, diciamo 1), e precisamente trasforma l'input scritto inizialmente sul nastro infinito sulla base delle istruzioni codificate nel proprio programma. Le istruzioni sono combinazioni, conformi a certe regole, di comandi elementari quali: "Scrivi 1", "Cancella", "Vai a destra", "Vai a sinistra", "Fermati" (ulteriori particolari inerenti gli "stati" della macchina devono essere considerati, ma non ci soffermiamo su questo dettaglio).

([1]) Ad essere rigorosi, dobbiamo ricordare che vi sono molti numeri che hanno due sviluppi decimali, e questi sono banalmente generabili meccanicamente, in quanto costituiti da una sequenza finita di cifre distinte, seguita da un'unica cifra ripetuta all'infinito.

([2]) In realtà, si possono definire diverse varianti di tali macchine, ma di fatto i differenti modelli forniscono le medesime prestazioni.

Ciascuna macchina di Turing può essere considerata un "rudimentale calcolatore" per computare una certa funzione numerica, sebbene in realtà essa sia un dispositivo trasformatore di stringhe di bit, ovvero di *parole*. Non sono i numeri in se stessi ad essere considerati, bensì i loro nomi, i *numerali*. Ad esempio, la funzione "successore di un numero naturale" potrebbe essere concretamente simulata da un'apposita macchina di Turing M nel seguente modo. Supponiamo di volere calcolare il successore del numero 4. Forniamo quindi ad M in input una successione di quattro simboli 1 (il numerale di 4 nel linguaggio utilizzato), scrivendola sul nastro di lavoro della macchina (ciascun simbolo occupa una cella). M legge tutto l'input e aggiunge un'occorrenza del simbolo 1 sulla prima cella vuota incontrata dalla testina di lettura/scrittura. Il risultato (output) è la successione 11111, costituita da cinque occorrenze del simbolo 1. Nel dettaglio, scrivere un programma nel linguaggio macchina per effettuare questo calcolo è molto semplice. Inizialmente si suppone che la testina sia posizionata sulla prima cella, e si suppone anche che l'input sia scritto (senza interruzioni) sul nastro a partire da tale cella. E' sufficiente quindi chiedere alla macchina di muovere la testina verso destra fintanto che questa trova sul nastro occorrenze della cifra 1, e di scriverne un'ulteriore occorrerza, per poi fermarsi, nella prima cella vuota incontrata.

Fintantoché la computazione è ristretta all'ambito dei numeri naturali, ci si può limitare a considerare solo parole finite. Dato un certo input (finito), se la macchina si arresta dopo aver eseguito alcune operazioni, producendo un risultato (ovvero una stringa finita sul nastro di lavoro), si dice che la computazione *converge* sul dato input. Tuttavia, se su tale input la macchina si arresta senza fornire alcun risultato, oppure non termina mai il suo processo di calcolo, perpetuandolo all'infinito, allora diciamo che la computazione *diverge* (nuovamente, si possono formulare varianti più sofisticate del concetto di convergenza e divergenza, ma si tratta di variazioni che lasciano inalterata la sostanza del discorso). Consideriamo ad esempio il programma che associa ad un dato numero naturale la sua radice quadrata, ammesso che questa sia un numero naturale. Una macchina che esegue quel programma si arresterà sull'input 1111 producendo

Un programma simulante una macchina di Turing al Deutsches Museum di Monaco di Baviera (riproduzione per gentile concessione del museo).

l'output 11, mentre sull'input 11 la computazione dovrà necessariamente divergere.

Se vogliamo estendere il nostro dominio di calcolo all'insieme dei numeri reali, dobbiamo considerare computazioni su stringhe infinite di simboli, in modo da poter codificare, ad esempio, gli sviluppi decimali infiniti (non periodici) dei numeri irrazionali. Tradizionalmente un numero viene codificato tramite un'unica sequenza di simboli non interrotta da celle vuote, per cui per esprimere uno sviluppo decimale l'alfabeto deve contenere almeno due caratteri (per esempio 0 e 1).

L'impiego di stringhe infinite richiede l'estensione del concetto di convergenza a computazioni che necessitano di un tempo illimitato per produrre l'output nella sua interezza: tale ad esempio è il tempo indispensabile per la scrittura "completa" dello sviluppo decimale di $\sqrt{2}$.

$\sqrt{2}$ è un numero reale computabile, nel senso che è possibile mettere in atto un meccanismo per il calcolo del suo sviluppo decimale, con precisione assoluta arbitraria, senza bisogno di fornire alcun input. Lo stesso dicasi di tutti gli altri i reali algebrici. In generale, i numeri reali dei quali possiamo conoscere (o approssimare indefinitamente tramite una regola) il preciso valore sono computabili. Un esempio di numero

reale computabile trascendente è π. Come tutti sanno, il suo sviluppo decimale può essere infatti approssimato indefinitamente, e ad oggi se ne conoscono le cifre fino a diversi miliardi di decimali. Il computo di numeri quali $\sqrt{2}$ o π ha avuto origine a partire dall'antichità presso diverse civiltà, decisamente millenni prima che la teoria della computazione nascesse!

2. – Numeri reali computabili.

L'analisi di Turing del concetto di computazione non ha subito sostanziali revisioni per quanto riguarda l'insieme dei numeri naturali. E' noto come i diversi approcci alla computabilità su tali numeri suggeriti nello stesso periodo da altri autori (quali A. Church, E. Post e S. C. Kleene) siano di fatto equivalenti alla Turing-computabilità in quanto a potenza espressiva, sebbene formulati in differenti formalismi. Questa equivalenza è alla base della celebre *Tesi di Church*: le funzioni effettivamente calcolabili sono tutte e sole le funzioni calcolabili con macchine di Turing. Tale tesi è oggi il fondamento della teoria "ortodossa" della computabilità, nonostante non se ne conosca alcuna dimostrazione formale. Sebbene ripetutamente criticata dal punto di vista teorico, questa tesi non è mai stata smentita nella pratica.

Al contrario, i concetti di "numero reale computabile" e (soprattutto) quello di "funzione reale computabile" condividono una storia ben più travagliata.

A seguito della pubblicazione del suo celebre articolo, lo stesso Turing rivede la sua originale definizione di numero reale computabile, secondo la quale, come abbiamo detto, un numero reale è computabile se lo è il suo sviluppo decimale. Manda quindi in stampa nel 1937 una "correzione" al precedente lavoro. Egli si accorge che la prima caratterizzazione comporta infatti alcuni problemi di uniformità. Consideriamo in proposito le successioni computabili di numeri razionali (ovvero le successioni di parole (finite) che sono computate da una qualche macchina di Turing senza bisogno di alcun input e che codificano, secondo una qualche regola determinata, una successione di numeri razionali). Ebbene, Turing osserva ([11]) che talvolta due successioni computabili di numeri razionali possono convergere, ri-

spettivamente da destra e da sinistra, ad un certo numero reale, senza che sia possibile calcolarne lo sviluppo decimale sulla base della sola informazione contenuta nella due successioni stesse. Per dimostrare questo, Turing seleziona una determinata proprietà *non decidibile*, tale cioè che non sia possibile sapere se essa non è valida (per esempio la comparsa di una prefissata sequenza di cifre nello sviluppo decimale di π) e definisce una macchina che, qualora questa condizione sia soddisfatta, produce due successioni di razionali che convergono ad un certo numero a il cui sviluppo decimale comincia con 1; in caso contrario, le due successioni generate dal programma convergono ad un numero a', il cui sviluppo decimale comincia con 0. Qui sorge il problema: ogni cifra dello sviluppo decimale da computare deve essere scritta sul nastro in un determinato istante, ma noi non possiamo sapere in un tempo finito se la la proprietà vale oppure no (in particolare, se essa non vale!) E' quindi impossibile stabilire come cominci lo sviluppo decimale (se con 0 o con 1).

Per rimuovere l'ostacolo, Turing propone una nuova definizione, "*[...] modificando il modo in cui i numeri computabili sono associati a successioni computabili [...].*"

Nel dettaglio, Turing considera ogni stringa del tipo

$$c_0 1^n 0 c_1 c_2 c_3 \ldots,$$

dove ciascun c_i è un'occorrenza del simbolo 0 o 1, e "1^n" indica n occorrenze consecutive della cifra 1. A tale stringa associa il numero reale

$$(2c_0 - 1)n + \sum_{r=1}^{\infty} (2c_r - 1)\left(\frac{2}{3}\right)^r.$$

In questo modo, la stringa diventa un nome del numero reale corrispondente.

Tale "complicata" definizione rappresenta storicamente il primo esempio di *rappresentazione ammissibile*, secondo la terminologia introdotta in seguito da K. Weihrauch. Una *rappresentazione* di un dato insieme non è altro che un modo di denotare (codificare) gli elementi di quell'insieme per mezzo di stringhe infinite di bit (nel nostro caso la successione $c_0 1^n 0 c_1 c_2 c_3 \ldots$). Torneremo in seguito su tale con-

cetto, limitandoci per ora ad osservare che Turing identifica effettivamente un buon sistema di codifica dei numeri reali, per questo detto appunto "*ammissibile*".

Definizioni formalmente differenti di numero reale computabile, ma equivalenti alla seconda formulazione data da Turing, vengono fornite in seguito da altri scienziati, quali S. Mazur e H. G. Rice.

Stanislav Mazur pubblica nel 1963 un lavoro di analisi computazionale che contiene una definizione di funzione reale computabile maturata durante la precedente collaborazione con Hans Banach. Questa coppia di eminenti matematici, oltre ad aver fornito significativi contributi all'analisi funzionale e alla teoria dei giochi, ha infatti legato il proprio nome anche ad una delle prime formulazioni del concetto di funzione reale computabile. L'opera di Banach-Mazur sull'argomento ha in realtà notevole influenza già a partire dagli anni '50, molto tempo prima quindi della pubblicazione. Per introdurre il concetto, Mazur propone innanzitutto alcune definizioni di numero reale computabile, tutte equivalenti tra loro, come osservato da R. M. Robinson. Tra queste, la più nota può essere formulata nel seguente modo: un numero reale a è detto computabile se possiamo computare una successione di numeri razionali che tende a quel numero, ed in modo che la velocità di approssimazione sia ben determinata. Formalmente, Mazur richiede l'esistenza di una funzione f computabile sui numeri naturali ([3]) tale che

$$\forall n \in \mathbb{N} : \left| a - \frac{f(n)}{n+1} \right| < \frac{1}{n+1}.$$

In pratica Mazur identifica un numero reale a con successioni di Cauchy $r_0, r_1, r_2, \ldots$ di numeri razionali soddisfacenti il seguente criterio di convergenza:

$$\forall i : |a - r_i| < \frac{1}{i+1}.$$

([3]) Ricordiamo che per la tesi di Church le funzioni computabili sui numeri naturali sono esattamente le funzioni Turing-computabili.

Intuitivamente si può comprendere l'importanza di selezionare solo le successioni di Cauchy che soddisfano determinati moduli di convergenza. La semplice convergenza non è dal punto di vista computazionale molto soddisfacente: giacché in generale non possiamo conoscere il valore esatto di un numero reale, è per lo meno desiderabile sapere quanto siano affidabili le approssimazioni razionali che via via riusciamo a calcolare. In questo modo otteniamo rappresentazioni *ammissibili* dei numeri reali.

Generalizzando il concetto di numero reale computabile, Mazur introduce poi il concetto di successione computabile di numeri reali $(a_k)_{k\in\mathbb{N}}$, definendola tramite una funzione f computabile su coppie di numeri naturali tale che:

$$\forall n, k \in \mathbb{N} : \left| a_k - \frac{f(k,n)}{n+1} \right| < \frac{1}{n+1}.$$

3. – Nascita del concetto di funzione reale computabile.

Abbiamo accennato a come l'umanità abbia da millenni manipolato numeri reali computabili, ben prima della nascita della teoria della computazione. Anzi, prima di questa, gli unici numeri reali effettivamente conosciuti erano computabili, e uno dei meriti della teoria è stato appunto quello di mostrare esempi di numeri reali concretamente definibili ma non computabili. Similmente, ci si dovrebbe aspettare che una qualsiasi teoria soddisfacente della computabilità sui numeri reali definisca come calcolabili le operazioni fondamentali dell'analisi (somma, prodotto, radice, ...). Osserviamo innanzitutto che se lavoriamo con gli sviluppi decimali andiamo tuttavia incontro a nuovi problemi di uniformità. Come calcolare la prima cifra del prodotto $0,3333... \times 3$? Se il primo fattore è $0,\overline{3}$, allora tale cifra potrebbe essere 1. Questo è tuttavia un errore se lo sviluppo decimale del primo fattore contiene in realtà un 2. Possiamo allora scrivere 0, ma nuovamente questo si rivela essere un errore se nel suo sviluppo non vi sono occorrenze di 2 ma di 4. Al contrario, se i numeri reali vengono identificati con successioni di Cauchy al modo suggerito da Mazur (o in un secondo tempo da Turing) allora questi

problemi scompaiono ed otteniamo i risultati desiderati. Proprio su tale base Mazur fornisce una definizione di funzione reale computabile (o *Banach-Mazur-computabile*, diremo noi, per distinguerla dagli altri approcci che analizzeremo in seguito).

Innanzitutto, egli considera le funzioni reali computabili come entità definite solamente all'interno dell'insieme dei numeri computabili $\mathbb{R}_c$.

Ebbene, secondo Mazur, una funzione reale φ è computabile se per ogni successione computabile $(a_k)_{k\in\mathbb{N}}$ di numeri reali nel dominio di φ, la successione dei valori $(\varphi(a_k))_{k\in\mathbb{N}}$ è a sua volta computabile.

Chiamiamo *computabilità sequenziale* tale proprietà, seguendo Pour-El e Richards.

Come rileveremo ripetutamente, il concetto di computabilità e quello di continuità (nel suo significato topologico) sono strettamente correlati. Un primo esempio è costituito dal principale risultato di Mazur, che riguarda la continuità sequenziale sull'insieme delle successioni computabili di numeri reali: data una funzione Banach-Mazur-computabile $\varphi : A \to \mathbb{R}_c$, con $A \subseteq \mathbb{R}_c$, e data $x_0, x_1, x_2, \ldots$ compatibile in $A = \mathrm{dom}(\varphi)$, se

$$\lim_{i\to\infty} x_i = x \in \mathrm{dom}\,(\varphi)$$

allora

$$\lim_{i\to\infty} \varphi(x_i) = \varphi(x).$$

Negli anni '50 D. Lacombe utilizza direttamente il concetto di continuità per definire quello di computabilità. Possiamo infatti dire che la nozione che egli propone altro non sia che il concetto di continuità topologica corretto con un criterio di *effettività*. L'insieme degli intervalli aperti in $\mathbb{R}$ con estremi razionali può essere *enumerato*, ovvero messo in corrispondenza biunivoca con l'insieme dei numeri naturali (i due insiemi hanno infatti la medesima cardinalità), in modo tale che questa associazione non sia eseguita in modo arbitrario bensì effettuata in modo meccanico (diciamo quindi "*computabilmente enumerato*"). Denotato con s_n l'n-simo intervallo in questa enumerazione, una funzione reale to-

tale φ è computabile ([4]) se esiste una funzione computabile f sui numeri naturali tale che per ogni numero reale $a \in \mathbb{R}$:

- se $a \in s_n$ allora $\varphi(a) \in s_{f(n)}$;
- se $\varphi(a) \in s_m$ allora esiste un $n \in \mathbb{N}$ tale che $a \in s_n$ e $s_{f(n)} \subseteq s_m$;
- se $s_n \subseteq s_m$ allora $s_{f(n)} \subseteq s_{f(m)}$.

Nello stesso anno di pubblicazione del lavoro di Lacombe, A. Grzegorczyk fornisce a sua volta una definizione di funzione reale computabile. Riconsideriamo il metodo di identificare i numeri reali con successioni di Cauchy di numeri razionali a modulo di convergenza precisato. Una funzione computabile che associa ad un numero reale a un numero reale b può essere quindi considerata come un processo meccanico che trasforma le successioni convergenti ad a (secondo il modulo stabilito) in successioni convergenti a b (secondo lo stesso modulo).

Un discorso a parte merita l'approccio sviluppato negli anni '40 dal russo A. A. Markov e dalla sua scuola. Le sue indagini si collocano nell'ambito del programma *costruttivista*, in analogia con la tradizione intuizionista di L. E. J. Brouwer. Il nome di Markov è associato ad una versione debole del principio logico del *terzo escluso* aristotelico, secondo il quale qualsiasi enunciato o la sua negazione sono necessariamente veri. In realtà, già Aristotele ne aveva riconosciuto dei limiti di applicabilità (per esempio, nelle previsioni di avvenimenti futuri), e il principio di Markov afferma più semplicemente che nel caso in cui sia contraddittorio affermare che una certa successione infinita di bit 0 o 1

([4]) Come abbiamo visto, in analisi computazionale si considerano spesso funzioni *parziali*, il cui dominio cioè è un sottoinsieme dei numeri reali. Alcuni paradigmi, come quello di Banach-Mazur o di Markov, ammettono nell'universo del discorso solamente funzioni parziali (poichè considerano solamente i numeri computabili). Al contrario gli altri approcci, come quello che stiamo analizzando, consentono la trattazione anche di funzioni reali totali (definite quindi anche sui numeri non computabili), accanto a quelle parziali (che possono comunque essere definite in ogni caso anche per punti non computabili). Per semplicità di descrizione, nei casi in cui l'estensione del dominio non sia di particolare rilevanza ai fini del discorso, formuleremo le nostre definizioni in termini di funzioni totali, confidando che i concetti introdotti possano essere tradotti in modo del tutto ovvio per le funzioni parziali.

abbia costantemente valore 0, allora è corretto assumere che tale successione contenga almeno un 1.

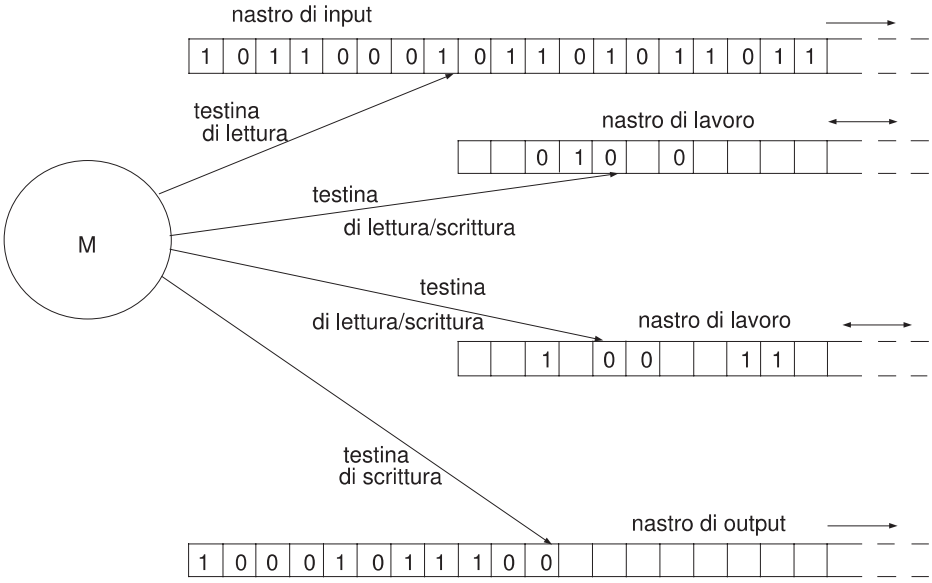
Una caratteristica che accomuna l'indagine di Markov al paradigma di Banach-Mazur è la restrizione dell'universo matematico ad oggetti "costruibili", che nel nostro caso si deve intendere come "computabili". Secondo Markov, un numero reale (computabile) è da identificarsi con i programmi che ne generano rappresentazioni ammissibili (usando un termine che abbiamo già incontrato, ma estraneo a Markov). A loro volta, solo le funzioni reali "costruibili" sono ammesse, e queste sono interpretate come programmi per trasformare numeri (computabili) in numeri (computabili), ovvero, programmi in programmi.

Il rapporto tra Banach-Mazur-computabilità e Markov-computablità è stato chiarito in modo esaustivo solo di recente. Ogni funzione Markov-computabile è anche Banach-Mazur-computabile. Da Pour-El e Richards ([9]) sappiamo che l'inverso non vale nel caso di funzioni definite su particolari sottoinsiemi di $\mathbb{R}_c$. La domanda tuttavia se si diano controesempi significativi (come funzioni definite sull'intero insieme dei numeri reali computabili) è rimasta insoluta per molto tempo. Solo recentemente P. Hertling, riprendendo un'idea del grande ricorsivista R.M. Friedberg, è riuscito a darvi risposta, esibendo un effettivo controesempio di tale natura.

4. – La teoria delle rappresentazioni.

Tutti i precedenti approcci sono strettamente correlati alla cosiddetta "Teoria di Tipo 2 dell'Effettività" (*Type Two Theory of Effectivity, TTE*). L'appellativo "di Tipo 2" è stato apposto per distinguerla dalla teoria della computabilità usuale (per parole finite e numeri naturali).

La TTE considera macchine di Turing aventi uno o più nastri di input di sola lettura a senso unico, alcuni nastri di lavoro di lettura/scrittura a doppio senso, ed infine un nastro di output di sola scrittura a senso unico. Ogni nastro è dotato di una testina, che esegue operazioni di lettura o scrittura a seconda delle specificazioni che abbiamo dato per il nastro corrispondente. Per "nastri a senso unico" si intende nastri che possono essere percorsi dalle testine corrispondenti solo in un verso (conven-



Un esempio di macchina di Turing di tipo 2, utilizzata in TTE.

zionalmente da sinistra a destra), senza che queste possano retrocedere di posizione. Ciascuna cella del nastro viene quindi considerata una sola volta. Al contrario, se il nastro è a doppio senso, può essere percorso in entrambi i versi, e su una stessa cella è possibile operare ripetutamente. Queste macchine possono operare su successioni infinite di simboli. Tali stringhe di bit servono per codificare oggetti matematici, similmente a come abbiamo già visto nei paragrafi precedenti. L'utilizzo di nastri a senso unico corrisponde in primo luogo alla necessità di avere approssimazioni affidabili dell'output infinito: giacché è concretamente impossibile consultare istantaneamente l'output infinito nella sua interezza, si richiede almeno che i risultati parziali della computazione siano affidabili, e perciò si impedisce alla testina di scrittura di retrocedere e correggere. Questa imposizione consente in ultima analisi un'elegante caratterizzazione topologica, che si riflette nello stretto legame che va così (nuovamente) ad instaurarsi tra computabilità e continuità. In TTE si utilizzano solitamente o un alfabeto finito, $\{0,1\}$, o uno infinito, $\mathbb{N}$ (la scelta solitamente non comporta sostanziali differenze teoretiche). Il linguaggio consiste in successioni finite o infinite di simboli dell'alfabeto (per semplicità, limitiamoci ora al solo caso infinito). A seconda del-

l'alfabeto utilizzato, questi linguaggi coincidono con lo spazio $\{0,1\}^\omega$ o con lo spazio $\mathbb{N}^\omega$, ossia con i prodotti topologici numerabili degli spazi discreti $\{0,1\}$ e $\mathbb{N}$, rispettivamente. In topologia, il primo è chiamato "spazio di Cantor" ($\mathcal{C}$), e il secondo "spazio di Baire" ($\mathbb{B}$). Ebbene, un principio della TTE afferma che ogni funzione computabile in tali linguaggi (da una macchina di Turing del tipo previsto dalla TTE) è una funzione continua sugli spazi topologici corrispondenti. Detto grossolanamente, una funzione è continua su tali spazi se segmenti iniziali uguali di due diverse successioni infinite sono proiettati su segmenti uguali.

La "teoria delle rappresentazioni" è centrale nella TTE ed affronta in modo sistematico il problema della codifica del mondo matematico tramite il linguaggio. Se i casi che abbiamo visto precedentemente riguardavano solamente i numeri reali, la teoria delle rappresentazioni si occupa dei principi generali di codificazione, soprattutto di quelli soddisfacenti, di varie strutture matematiche astratte: non solo insiemi di numeri quindi, ma anche ad esempio spazi di funzioni, o spazi topologici.

Gli oggetti matematici e le funzioni tra di essi non vengono più classificati come computabili o non computabili in sè, ma in relazione alle loro codifiche, al punto che secondo tale prospettiva questi possono essere computabili rispetto ad alcune codifiche ma non rispetto ad altre. Infatti, rigorosamente parlando, non sono gli oggetti o le funzioni ad essere computabili in sé, bensì alcuni dei loro nomi. Per tale motivo l'attenzione della teoria delle rappresentazioni è posta sulle "simulazioni linguistiche" delle funzioni matematiche, piú che sulle funzioni stesse. Che una funzione matematica sia classificata quindi come computabile o non, dipende da come si decide di *rappresentare* (codificare) le strutture tra cui essa è definita. Rispetto ad alcune codifiche, può effettivamente esistere una corrispondente simulazione computabile (in $\mathcal{C}$ o in $\mathbb{B}$), rispetto ad altre potrebbe non esistere.

Tuttavia si può dimostrare che due rappresentazioni "computabilmente equivalenti" ([5]) di una stessa struttura matematica determi-

([5]) Tali cioè che esiste una funzione computabile di traduzione dell'una nell'altra e viceversa.

nano per tale struttura una medesima *nozione di computabilità*: definiscono classi eguali di oggetti e funzioni computabili. Le nozioni di computabilità possibili sono molte (addirittura non numerabili), e tuttavia solo poche sono realmente interessanti (in relazione ad una qualche struttura matematica). Come ci si può aspettare, interessanti sono quelle rappresentazioni che non denotano arbitrariamente gli oggetti, ma li descrivono in modo significativo. Ad esempio, un tipo semplice di rappresentazione, ma di grande importanza, consiste nel descrivere gli oggetti elencando le loro proprietà elementari. Ovviamente, affinché non sorgano ambiguità nel riferimento, non dovranno mai esserci due oggetti soddisfacenti esattamente le stesse proprietà. L'idea riprende quindi indirettamente la concezione di Leibniz, secondo cui ciascun oggetto è univocamente determinato dall'insieme delle sue proprietà. Poiché lavoriamo con *successioni* di bit, si richiede tuttavia che una quantità numerabile di proprietà, dette "*atomiche*", sia sufficiente a caratterizzare un individuo. Tecnicamente, stiamo considerando uno spazio topologico T_0 con una sottobase numerabile ([6]) (si osservi che $\mathbb{R}$ è uno spazio di questo tipo). La rappresentazione definita in tal modo è chiamata *rappresentazione standard* dello spazio considerato.

Le rappresentazioni degli spazi T_0 con sottobasi numerabili vengono trattate in un articolo del 1985 di C. Kreitz e K. Weihrauch, "*Theory of Representations*". In particolare, gli autori focalizzano il concetto di *rappresentazione ammissibile* per tali spazi: una rappresentazione è ammissibile se esiste una funzione continua (nello spazio del linguaggio) che traduce tale rappresentazione nella rappresentazione standard, e viceversa.

Come dicevamo, la TTE è strettamente correlata agli approcci precedentemente citati, ma in generale non si tratta di una completa equivalenza. L'approccio di Grzegorczyk può esserne considerato un sottocaso, considerando di fatto funzioni computabili rispetto a rappresentazioni ammissibili dei numeri reali. La situazione è invece più

([6]) Uno spazio topologico è T_0 quando dati due qualsiasi punti in esso esiste un insieme aperto che contiene uno dei due punti ma non l'altro.

complessa per quanto riguarda i paradigmi di Banach-Mazur e di Markov.

Ogni funzione reale TTE-computabile ([7]) è Banach-Mazur-computabile (o meglio, la sua restrizione a $\mathbb{R}_c$ lo è), mentre l'inverso vale se il dominio della funzione è denso ([8]) in $\mathbb{R}$. In caso contrario, sono noti in letteratura alcuni controesempi.

Il discorso è simile per quanto riguarda il paradigma di Markov: ciascuna funzione reale TTE-computabile è Markov computabile in $\mathbb{R}_c$, mentre l'inverso non vale, come dimostrato da J. Myhill. Tuttavia, sulla base di un risultato di G.S. Ceitin del 1959, sappiamo che se il dominio della funzione Markov-computabile contiene una successione computabile densa di numeri reali, allora essa è anche TTE-computabile.

5. – Strutture di computabilità e teoria dei domini.

Come abbiamo visto, la TTE si occupa di computabilità per varie strutture matematiche oltre che per i numeri reali. Accanto ad essa, anche altre teorie si sono mosse su prospettive più vaste di quella della semplice definizione di funzione reale computabile, con risultati spesso convergenti, ma con impostazioni teoriche differenti. Tra questi, vi è il paradigma di M. Pour-El e J. I. Richards, esposto in modo sistematico nel loro volume *Computability in Analysis and Physics* del 1989 ([9]). In questo, gli autori analizzano il concetto di computazione sulla base di determinate *strutture di computabilità*, definite come coppie $(X, \mathscr{S})$, con X spazio di Banach ([9]) e $\mathscr{S}$ insieme delle successioni in X soddisfacenti precisi assiomi di calcolabilità da loro formulati. Come caso particolare, un oggetto x dello spazio di Banach è detto computabile se la successione costante $n \mapsto x$ per ogni $n \in \mathbb{N}$ è in $\mathscr{S}$.

([7]) Quando parliamo di funzioni reali TTE-computabili senza ulteriori specificazioni ci riferiamo alla rappresentazione standard dello spazio euclideo $\mathbb{R}$.

([8]) Dato uno spazio X dotato di *metrica*, ovvero uno spazio su cui siano definite le distanze per le coppie dei suoi punti, un insieme A è *denso* in X se preso un qualsiasi punto x di X, vi sono punti di A a distanze arbitrariamente piccole da x.

([9]) Si consideri uno spazio X dotato di norma $\|.\|$. Si definisca la distanza tra due punti x, y in X come $\|x - y\|$. Supponiamo che X sia completo rispetto a tale metrica. Allora X è uno *spazio di Banach*.

Nei casi "interessanti", l'approccio tramite le strutture di computabilità diviene un caso particolare di quello TTE. Per "caso interessante" intendiamo un qualsiasi spazio di Banach per cui possiamo computabilmente generare un insieme denso nello spazio stesso. La struttura di computabilità che ne risulta può essere infatti considerata come uno spazio metrico computabile, oggetto di fondamentale importanza in TTE. ([10])

Ciononostante, non è detto che i due approcci siano teoricamente del tutto equivalenti, a causa dell'esistenza di spazi di Banach non (effettivamente) separabili.

Nel libro vi sono diverse definizioni di funzione reale computabile. Tra queste, ricordiamo quella che essi considerano la "*versione effettiva del teorema dell'approssimazione di Weierstrass*", e che riguarda funzioni parziali definite su intervalli compatti con estremi computabili (per semplicità, prendiamo l'intervallo [0,1]). Una funzione $\varphi : [0,1] \to \mathbb{R}$ è computabile se esiste una sequenza computabile di polinomi razionali $\{p_m(x)\}$ che converge a φ "in modo effettivo", ovvero vi è una funzione computabile $e : \mathbb{N} \to \mathbb{N}$ tale che per ogni $x \in [0,1]$ e ogni $N \in \mathbb{N}$:

$$m \geq e(N) \Rightarrow |\varphi(x) - p_m(x)| \leq 2^{-N}.$$

Intuitivamente, la funzione viene approssimata tramite successioni di linee "spezzate" che tendono al grafico della funzione stessa, ed il livello di approssimazione è computabilmente determinato.

Nella nostra breve trattazione dei paradigmi di computabilità non possiamo soffermarci sulla *Domain Theory*, in quanto il suo astratto apparato concettuale rende difficile una sua presentazione in termini intuitivi, e tuttavia la sua importanza è tale da richiedere, per lo meno, un rapido accenno, restando inteso che l'argomento meriterebbe un maggiore approfondimento. La Domain Theory è stata sviluppata come strumento per la semantica matematica di

([10]) Uno spazio metrico è detto "computabile" se possiamo enumerare meccanicamente una successione A densa in esso (spazio effettivamente separabile) e riusciamo a calcolare le distanze tra coppie di punti in A (come esempio, si consideri $\mathbb{R}$, per il quale l'insieme dei numeri razionali soddisfa le condizioni richieste).

linguaggi di programmazione (per maggiori informazioni rimandiamo a [13]). Ci limitiamo ad osservare che le funzioni reali TTE-computabili sono definibili all'interno della Domain Theory, e viceversa, i domini utilizzati per tale traduzione sono topologie di tipo T_0 e con base numerabile, e quindi per essi la nozione di computabilità può essere caratterizzata tramite la teoria delle rappresentazioni.

6. – Il modello "real-RAM" e la disputa sui fondamenti.

In una prospettiva radicalmente diversa rispetto a tutte le precedenti si colloca il modello detto "real-RAM", che è trattato in modo sistematico da L. Blum, F. Cucker, M. Schub, S. Smale (medaglia Field 1966) nel volume *Complexity and Real Computation* del 1998 ([1]); per semplicità, ci riferiremo spesso a questo quartetto di autori tramite le iniziali "BCSS".

La parte introduttiva del saggio è dedicata ad una vivace difesa del paradigma, nella consapevolezza della sua radicale diversità rispetto ai modelli basati sull'utilizzo, più o meno esplicito, delle macchine di Turing. Si comprende quindi come mai agli occhi degli autori appaia quanto mai rilevante la necessità di fornire argomentazioni di supporto al loro sistema.

Uno dei principali obiettivi del volume è quello di avvicinare due antitetiche concezioni di "calcolo", storicamente appartenenti a due ambiti ben distinti della matematica, e contrapposti da BCSS persino in termini linguistici: *scienza della computazione* ("computer science") e *computazione scientifica* ("scientific computation"). La prima trae origine dal lavoro dei logici e su di essa si basano gli altri paradigmi precedentemente analizzati; la seconda ha invece come fondamento le equazioni della fisica matematica (e l'analisi numerica). In [1] leggiamo in merito:

> *"Vi è un sostanziale conflitto tra la scienza teorica della computazione e l'analisi numerica. Queste due discipline nonostante gli scopi comuni si sono sviluppate indipendentemente (...) Il conflitto ha origine in un altro secolare antagonismo, quello tra il continuo e il discreto. La scienza della computa-*

zione è caratterizzata dalla natura digitale delle macchine di Turing e dalla sua conseguente fondazione mediante il concetto di discreto. Per l'analisi numerica al contrario, sono i sistemi di equazioni e le equazioni differenziali ad essere fondamentali e questa disciplina dipende quindi fortemente dalla natura continua dei numeri reali."

Il confronto storico suggerito è veramente suggestivo:

"Isaac Newton dovette affrontare un problema analogo all'epoca della stesura dei "Principia". Ai tempi di Newton, gli scienziati supponevano che il mondo fosse atomico, secondo la visione dell'antico filosofo greco Democrito. Newton accettò l'idea secondo la quale la materia sarebbe composta di parti indivisibili, in numero finito in ciascuna regione limitata. Dall'altro lato, la matematica di Newton era continua come quella di Euclide. Per di più, le equazioni differenziali che Newton utilizzava nella sua teoria presupponevano il continuo, in contrasto con la visione corpuscolare dell'universo. Si trattava sostanzialmente di riconciliare il mondo discreto con il continuo matematico".

BCSS sostengono che la risposta di Newton al problema fu la seguente:

"La soluzione fu trovata analizzando i risultati della sostituzione di un singolo oggetto (ad esempio la terra) mediante un numero finito di particelle, proseguendo poi per approssimazioni via via maggiori con numeri sempre più grandi di particelle. Passando al limite, la matematica diviene continua."

Il quesito su come conciliare la natura discreta della computazione meccanica con il continuo dell'analisi dovrebbe quindi essere risolto in modo del tutto analogo:

"Il nostro suggerimento è quello di idealizzare il computer moderno digitale allo stesso modo in cui Newton idealizzò il suo universo discreto. I numeri con cui la macchina opera sono numeri razionali in quantità finita, ma questi sono distribuiti all'interno di un intervallo limitato di numeri reali (...) in modo sufficientemente omogeneo che intendere il computer come una macchina manipolatrice di numeri reali diventa un'astrazione del tutto accettabile."

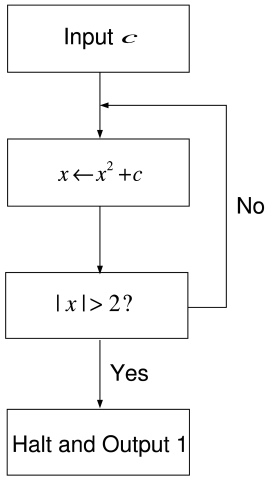
Non a caso, tra gli esempi storici concretamente trattati dagli autori vi è il metodo di Newton per l'individuazione degli zeri dei polinomi.

Tuttavia, nel loro lavoro BCSS non si occupano solamente di cercare un metodo soddisfacente per la trattazione di problemi classici della matematica. La loro ricerca è infatti ispirata anche dai moderni sorprendenti sviluppi dell'informatica, tra cui la grande diffusione, persino a livello popolare e di divulgazione scientifica, della *computer graphics* applicata alla geometria frattale. Nel suo libro "*La Mente nuova dell'imperatore*" ([8]) il fisico Roger Penrose pone, in modo non rigoroso, la questione della *decidibilità* dell'insieme di Mandelbrot. Questo insieme è definito da semplici disequazioni nel dominio dei numeri complessi, e ciononostante è divenuto alquanto popolare per via della sua "misteriosa" ed "accattivante" raffigurazione ottenibile in computer graphic. Sulla base di questi due assunti (semplicità descrittiva e complessità visiva), si può comprendere meglio la domanda di Penrose sull'esistenza di un metodo meccanico per stabilire l'appartenenza o la non appartenenza di un punto qualsiasi all'insieme ("decidibilità"). BCSS riformulano rigorosamente all'interno del loro modello la domanda di Penrose per rispondervi negativamente: secondo il loro paradigma un insieme non può essere decidibile se non è unione numerabile di insiemi semi-algebrici, e altrettanto deve essere il complementare; al contrario, il complementare dell'insieme di Mandelbrot non è di tale tipo. ([11])

Ma vediamo, per quel che ci è possibile in questa sede, qualche maggiore dettaglio sull'approccio di BCSS. Le macchine real-RAM sono definite tramite flowchart, questi ultimi intesi come algoritmi costruiti tramite le istruzioni "if... then.... else", "goto", "stop", e le operazioni di somma, sottrazione, moltiplicazione, divisione, e confronto di due numeri reali. Tali operazioni matematiche sono assunte

([11]) La struttura dei numeri complessi è qui identificata con $\mathbb{R}^2$. Un insieme $S \subseteq \mathbb{R}^n$ è semi-algebrico se è costituito dai punti che soddisfano un determinato sistema finito di equazioni e disequazioni polinomiali.
Facciamo rilevare che il problema della decidibilità dell'insieme di Mandelbrot nel modello TTE è tuttora aperto.

come primitive. A titolo di esempio, riportiamo un ipotetico flowchart "ingenuo" per individuare gli elementi del complementare dell'insieme di Mandelbrot:



Flowchart per individuare i punti che non appartengono all'insieme di Mandelbrot.

Senza addentrarci nella analisi matematica dell'algoritmo, ci limitiamo ad osservare che $x^2 + c$ è una funzione di variabile complessa, e che i numeri complessi possono essere considerati come coppie ordinate di numeri reali. Ciò che occorre rilevare è il fatto che il modello real-RAM si basa sul principio che sia effettivamente possibile confrontare, o sommare, due numeri reali in tempo finito, il che è inammissibile in un approccio con macchine di Turing. Il contrasto tra le due prospettive è quindi evidente, ed infatti esse definiscono due insiemi radicalmente distinti di funzioni candidate al titolo di "computabili". Ad esempio, la funzione "a gradini" $s : \mathbb{R} \to \mathbb{R}$:

$$s(x) = \begin{cases} 0 & \text{se } x < 0 \\ 1 & \text{se } x \geq 0 \end{cases}$$

è real-RAM computabile, mentre non è assolutamente TTE-computabile, in quanto discontinua. Le stesso dicasi per un'altra funzione a "gradini", quella di Gauss, che associa ad ogni numero reale "x" la sua parte intera.

Esistono anche funzioni TTE-computabili non real-RAM-computabili, e tra queste alcune alquanto familiari in matematica e regolarmente "computate" da macchine calcolatrici, quali la radice quadrata $\sqrt{x}$ o la funzione esponenziale e^x. Talvolta queste vengono aggiunte assiomaticamente alle funzioni primitive calcolate da una macchina real-RAM, e tuttavia si può dimostrare che comunque si arricchisca un modello real-RAM con un numero finito di funzioni TTE-computabili, esisteranno sempre delle funzioni TTE-computabili non computabili nel modello real-RAM così arbitrariamente potenziato.

Si comprende bene quindi come mai i due approcci siano stati ideologicamente contrapposti tra loro. Secondo BCSS:

> *"Uno degli ostacoli principali per la riconciliazione dei concetti di computazione scientifica e scienza della computazione è l'attuale idea di macchina, ossia di computer digitale. Fino a quando il calcolatore sarà inteso come un oggetto finito e discreto, risulterà alquanto difficile fornire una trattazione sistematica dell'analisi numerica."*

Lo stesso nome del paradigma, "real-RAM", sottolinea la volontà di affrancarsi dall'utilizzo di informazione codificata in successioni. In informatica, "RAM" è acronimo di "Random Access Memory", ed indica il supporto di memoria su cui è possibile leggere e scrivere informazioni con un accesso *casuale*, senza dover cioè rispettare un determinato ordine sequenziale (come avviene invece utilizzando i nastri magnetici).

Un fermo oppositore dell'approccio real-RAM è K. Weihrauch, che ne critica i fondamenti in diverse occasioni. Nel suo *Computable Analysis* ([14]), scrive:

> *"Le macchine real-RAM non possono essere realizzate tramite macchine fisiche, ovvero, sono di fatto irreali, per le seguenti ragioni: in un tempo finito un qualsiasi canale di informazione fisico può trasmettere solo un numero finito di bit di informazione ed inoltre ogni memoria fisica è finita. Dal momento che l'insieme dei numeri reali non è numerabile, è impossibile identificare un numero reale tramite una quantità finita di informazione. Dunque, è impossibile trasferire un numero reale arbitrario da o verso un computer in un tempo finito, oppure memorizzare concretamente un numero reale in un computer."*

Abbiamo già osservato che le obiezioni sollevate in questo passo da K. Weihrauch dovrebbero essere fronteggiate tramite la soluzione "newtoniana", ovvero l'approssimazione del continuo con il discreto. Tuttavia, anche questo metodo è criticato da Weihrauch ([14]), secondo il quale vi sono funzioni che pur essendo dimostrabilmente real-RAM-computabili, non possono di fatto essere simulate in modo soddisfacente tramite il metodo dell'approssimazione del continuo con il discreto ([12]).

Inoltre, seguendo M. Braverman e S. Cook ([3]), le operazioni computate concretamente sin dall'antichità e quotidianamente eseguite da calcolatori elettronici dovrebbero essere classificate come (facilmente) computabili da qualsiasi teoria. Questo tuttavia non è il caso del modello real-RAM, come abbiamo già osservato per quanto riguarda l'estrazione di radice e la funzione esponenziale.

In realtà, tali funzioni possono essere sostituite da loro approssimazioni real-RAM computabili. Esiste infatti ad esempio una funzione real-RAM computabile che associa ad un qualsiasi $x \in \mathbb{R}$ e ad un qualsiasi $\varepsilon \in \mathbb{Q}^+$, un numero reale $y \in \mathbb{R}$ che approssima e^x a meno di ε. Tuttavia una tale traduzione non risulta essere molto soddisfacente per quanto riguarda il principio di *componibilità* delle funzioni, giacchè le funzioni real-RAM computabili richiedono in input valori esatti. Ad esempio, per calcolare (o meglio, approssimare) e^{e^x}, dovremmo applicare la funzione esponenziale "approssimata" al valore esatto di e^x, ma non possiamo conoscere quest'ultimo se precedentemente abbiamo applicato ad x non la funzione esponenziale in sé, ma la sua stessa traduzione approssimata. Al contrario, nei processi di calcolo è del tutto naturale che l'output di una computazione sia fornito in input ad un'altra computazione, sicché l'intero sistema risulta essere un processo computabile. La TTE tratta appunto il basilare principio di componibilità delle funzioni computabili in tutta naturalezza.

([12]) Tra queste, vi è la funzione

$$f(x) = \begin{cases} 1 & \text{se } x \in \mathbb{Q} \\ 0 & \text{se } x \notin \mathbb{Q} \text{ and } x^2 \in \mathbb{Q} \\ \text{indefinito} & \text{altrimenti} \end{cases}$$

Nonostante queste problematiche insite nell'approccio real-RAM, il dibattito è ad oggi ancora aperto. Attualmente non si conosce alcun dispositivo fisico che computi esattamente la classe delle funzioni real-RAM-computabili, ma in compenso vi sono interessanti tentativi di spiegazione del perché talvolta la simulazione delle funzioni real-RAM computabili tramite il metodo delle approssimazioni risulti appropriata.

Per quel che ci concerne, ci congediamo dalla "disputa" con alcune osservazioni di tono finalmente riconciliatorio, che illustrano possibili prospettive di convergenza tra i due approcci.

Innanzitutto, il paradigma real-RAM accetta la Turing-computabilità nella sua interezza per quanto riguarda i numeri naturali, anzi le due teorie coincidono nell'ambito del discreto.

Inoltre, l'apparato concettuale dell'approccio real-RAM è in parte derivato da quello della ricorsività (si pensi ad esempio alle nozioni di "*halting set*" e di decidibilità).

Da un lato, possiamo considerare le funzioni real-RAM-computabili come funzioni calcolate da macchine di Turing capaci di memorizzare numeri reali su singole celle del nastro (anche se queste macchine non hanno, che si sappia, realizzazioni nel mondo fisico).

Dall'altro lato, P. Hertling e V. Brattka hanno elaborato il modello *feasible real-RAM* per il quale la classe delle funzioni computabili coincide esattamente con quella delle funzioni TTE-computabili. Persino la complessità di calcolo rimane dello stesso ordine di grandezza.

Ci rivolgiamo infine al passato, e scopriamo che i due paradigmi contrapposti si ricongiungono parzialmente nello stesso Turing, come viene riconosciuto da F. Cucker. In un articolo pubblicato nel 1948 inerente la valutazione degli errori nella risoluzione di sistemi di equazioni lineari ([12]), Turing suggerisce l'idea che in un calcolo matriciale le operazioni di base sui numeri reali possano essere eseguite in multipli *finiti* dell'unità di tempo, in poche parole in tempo finito. Turing chiama "*misura del lavoro*" la complessità di calcolo valutata come numero di passi effettuati per ottenere il risultato:

> *"È conveniente avere una misura del tempo di lavoro impiegato in un processo di calcolo (...) Potremmo contare il numero di volte che varie operazioni elementari sono applicate nell'intero processo ed assegnare loro vari pesi. Potremmo, per esempio, contare il numero delle addizioni, sottrazioni,*

moltiplicazioni, divisioni, registrazioni di numeri ed estrazioni di cifre dalle tavole. Nel caso della computazione su matrici, la maggior parte del lavoro consiste in moltiplicazioni e registrazioni di numeri, e dovremo perciò preoccuparci solo di contare il numero di moltiplicazioni e registrazioni. A questo proposito, l'operazione di reciproco avrà lo stesso valore di una moltiplicazione (...) Una divisione sarà quindi valutata come due moltiplicazioni."

In merito alla risoluzione di un insieme di equazioni lineari, Turing *misura* le differenti quantità di lavoro richieste rispettivamente per risolvere il problema direttamente, o tramite il calcolo di matrici inverse. Nel secondo caso, rispetto al primo, osserva:

"Ciò comporta, in aggiunta, n^2 moltiplicazioni e n registrazioni per ciascun vettore, di contro a un totale di $\frac{1}{3}n^2$ moltiplicazioni (...)".

Turing analizza poi il grado di errore ottenuto nell'output sulla base dell'approssimazione iniziale dell'input, e conia per questo il termine "*numero di condizione*". Riconosce come per alcuni insiemi di equazioni, che chiama "*mal condizionate*", si assista ad un rilevante fenomeno di *disturbo*, tant'è che:

"piccole percentuali di imprecisione nei coefficienti possono causare grandi errori nella soluzione".

7. – Condizioni di incomputabilità.

Concludiamo la nostra breve analisi dei principali paradigmi di computazione sui numeri reali con alcuni risultati inerenti le possibili condizioni di incomputabilità, secondo la concezione della TTE e dei sistemi affini. Come abbiamo visto, secondo tale approccio, ogni funzione computabile è continua. Ne risulta che una qualsiasi funzione discontinua è non computabile. Si consideri tuttavia la funzione:

$$f(x) = \begin{cases} 1 & \text{se } x = 1 \\ 0 & \text{altrimenti} \end{cases}$$

Questa funzione, pur essendo discontinua, e quindi non TTE-computabile, mappa tutti i numeri reali (compresi quelli computabili) su numeri reali computabili (tali sono banalmente 0 e 1). Perciò la sua non computabilità è dovuta essenzialmente a proprietà topologiche (di-

scontinuità), piuttosto che computazionali. Può quindi essere interessante individuare funzioni la cui non computabilità sia maggiormente dovuta a questioni di natura squisitamente computazionale. Tali sono ad esempio le funzioni che, a prescindere dal grado di discontinuità, sono non computabili in quanto mappano qualche elemento computabile su un'immagine che non lo è.

Casi banali sono immediatamente dati: si prenda una funzione costante, il cui unico valore sia un numero reale non computabile. Più interessante è sicuramente trovare condizioni generali per tale fenomeno. Il così detto "First main theorem" di Pour-El e Richards ([9]) individua una classe di tali funzioni nell'ambito degli operatori lineari tra spazi di Banach. Un caso particolare è costituito dal loro celebre esempio sull'equazione d'onda. Si consideri la classe $C(\mathbb{R}^n)$ delle funzioni continue in $\mathbb{R}^n$, e la sua sottoclasse $C^1(\mathbb{R}^n)$ determinata da quelle funzioni nella classe che hanno le derivate parziali a loro volta nella classe (funzioni continue con derivate parziali continue). Si consideri poi il problema di Cauchy dell'equazione d'onda 3-dimensionale:

$$\begin{cases} u_{tt} = \Delta u, \\ u(0,x) = f(x), u_t(0,x) = g(x), t \in \mathbb{R}, x \in \mathbb{R}^3 \end{cases}$$

Classicamente è ben noto che nel caso $f \in C^1(\mathbb{R}^3)$ (f è continua con derivate parziali continue) e $g \in C(\mathbb{R}^3)$ (g è continua) il problema del valore iniziale ha un'unica soluzione $u(t, \bullet)$, per $u \in C(\mathbb{R}^3)$ (u è continua). Pour-El e Richards dimostrano che esiste una funzione computabile $f \in C^1(\mathbb{R}^3)$ tale che la soluzione del problema di Cauchy all'istante $t = 1$ non è computabile.

Questo risultato ha suscitato un interessante dibattito in merito alla questione se tale processo fisico non sia effettivamente simulabile da alcuna macchina di Turing. In particolare, le condizioni iniziali del sistema potrebbero essere fornite ad una macchina di Turing (in modo effettivo, poiché la funzione f e l'istante 1 sono Turing-computabili), ma l'output in uscita non potrebbe essere fornito da nessuna macchina di Turing che abbia f in input. Il sistema fisico in oggetto quindi realizzerebbe una funzione non simulabile da macchine di Turing.

Questa interpretazione non è tuttavia accettata da autori quali K. Weihrauch e N. Zhong ([15]). Secondo questi, infatti, la ragione del fenomeno sarebbe dovuta ad un "grossolano" modo di codificare il problema di Cauchy, verosimilmente non sufficientemente raffinato. Filosoficamente, si può illustrare l'obiezione tramite un sistema fisico ben più semplice. Si consideri un pendolo che all'istante 0 occupa la posizione r, dove r è un numero reale che può essere approssimato computabilmente solo "dal basso", ovvero possiamo conoscere con certezza tutti i numeri razionali minori di esso ma non tutti quelli maggiori (o uguali) ([13]). Supponiamo di considerare un numero reale come computabile esattamente quando può essere approssimato dal basso. Di conseguenza $-r$ non è computabile, essendo simmetricamente approssimabile solo "dall'alto". Il pendolo viene lasciato libero di muoversi ed all'istante 1 si trova in posizione $-r$. Apparentemente il nostro sistema è passato da uno stato computabile ad uno non computabile, mentre probabilmente abbiamo semplicemente accettato inizialmente una nozione di computabilità troppo debole per le nostre finalità: solitamente reputiamo un numero reale computabile quando possiamo approssimarlo da entrambe le direzioni, e questo non avviene nel caso considerato.

Similmente, Weirauch e Zhong ritengono in [15] che Pour-El e Richards abbiano utilizzato una codifica di f inappropriata: dal momento che è fondamentale non solo che f sia continua, ma anche che lo siano le sue derivate parziali, una soddisfacente codifica di f dovrebbe descrivere anche le sue derivate parziali. Weihrauch e Zhong dimostrano che con tale codifica il problema di Cauchy è perfettamente Turing-computabile. Questa obiezione rappresenta probabilmente un punto di forza della teoria delle rappresentazioni, in quanto il concetto di computabilità non è trattato in essa astrattamente, ma sempre in relazione ad una precisa codifica, che viene dichiarata in modo espli-

([13]) È merito della teoria della ricorsività aver mostrato che tali numeri esistono: si pensi ad esempio alla sommatoria di e tutti e soli gli addendi 10^{-i} tali che l'i-esima equazione diofantea ammette soluzioni intere.

cito: in questo modo è più facile trattare i risultati in un rigoroso linguaggio ed individuare chiaramente il loro contesto.

Tramite il First Main Theorem di Pour-El e Richards è anche possibile derivare come corollario l'esistenza di funzioni computabili con derivate continue ma non computabili, risultato originariamente ottenuto da J. Myhill nel 1971.

L'individuazione di condizioni generali sotto le quali una funzione associa ad alcuni oggetti computabili immagini non computabili è stata finalmente affrontata in modo sistematico da V. Brattka in [2]. Ne è risultato il così detto *Invariance Theorem*, che afferma, approssimativamente, che quanto maggiore è il grado di incomputabilità di una funzione matematica, tanto maggiore è il livello di incomputabilità delle immagini degli oggetti computabili nel suo dominio. La teoria di Brattka ripropone a livello più approfondito lo stretto legame tra i concetti topologici e quelli computazionali, costruendo una gerarchia di gradi di incomputabilità che ricalca quella dei gradi di discontinuità tramite gli insiemi boreliani, come è stata formulata in topologia classica ([5]) e teoria descrittiva degli insiemi ([4]).

BIBLIOGRAFIA

[1] L. Blum - F. Cucker - M. Schub - S. Smale, *Complexity and Real Computation*. Springer (1998).

[2] V. Brattka, *Effective Borel measurability and reducibility of functions*. Mathematical Logic Quarterly, **51** (2005), 19-44.

[3] M. Braverman - S. Cook, *Computing over the Reals: Foundations for Scientific Computing*. Notices of the American Mathematical Society, **53** (2005), 318-329.

[4] A. Kechris, *Classical Descriptive Set Theory*. Springer (1995).

[5] K. Kuratowski, *Topology*. Volume I. Academic Press (1966).

[6] P. Odifreddi, *Classical Recursion Theory*. North Holland (1989).

[7] P. Odifreddi, *Classical Recursion Theory.* Volume II. North Holland (1999).

[8] R. Penrose, *The Emperor's New Mind.* Oxford University Press. 1989 (edizione italiana: *La Mente Nuova dell'Imperatore.* Rizzoli, 1992)

[9] M. B. Pour-El - J. I. Richards, *Computability in Analysis and Physics.* Springer (1989).

[10] A. Turing, *On computable real numbers, with an application to the "Entscheidungsproblem"*. Proceedings of the London Mathematical Society, **422** (1936), 230-265.

[11] A. Turing, *On computable real numbers, with an application to the "Entscheidungsproblem". A correction*. Proceedings of the London Mathematical Society, **432** (1937), 544-546.

[12] A. TURING, *Rounding-off Errors in Matrix Processes*. The Quarterly Journal of Mechanics and Applied Mathematics, **1** (1948), 287-308.
[13] K. WEIHRAUCH, *Computability.* Springer (1987)
[14] K. WEIHRAUCH: *Computable Analysis.* Springer (2000).
[15] K. WEIHRAUCH - N. ZHONG, *The Wave Propagator is Turing Computable*, in J. WIEDERMANN P. VAN EMDE BOAS - M. NIELSEN (Eds.), *Automata, Languages and Programming*. Lecture Notes in Computer Science, **1644** (1999), 697-706.

Guido Gherardi, Dipartimento di Scienze Matematiche e Informatiche R. Magari
Università di Siena
e-mail: gherardi3@unisi.it