1

Capitolo C20
algoritmi e computabilità

Contenuti delle sezioni

a. problemi, istanze, procedimenti risolutivi e risultati p. 2
b. nozione di algoritmo p. 7
c. soluzione di problemi come elaborazione di stringhe p. 10

16 pagine

C20 0.01 Nelle pagine che seguono vengono riprese le nozioni di problema, di procedimento risolutivo e di algoritmo per avviare il riesame, in una prospettiva di generalità, delle attività computazionali, ossia dei processi che abbiamo assunti come base motivazionale della matematica e dell'informatica.

C20 a. problemi, istanze, procedimenti risolutivi e risultati

C20 a.01 Iniziamo con una definizione programmatica.

Con il termine **problema ben definito**, o semplificando, con **problema**, intendiamo una questione di portata tendenzialmente ampia, formulata in termini che possano essere ampiamente giudicati precisi e per la quale si ritiene possibile cercare di ottenere una risposta attraverso un **procedimento** chiaramente definito portato avanti da esecutori umani o artificiali affidabili atti a garantire la riutilizzabilità del procedimento stesso in condizioni simili.

Naturalmente la frase precedente richiede vari chiarimenti e per cominciare presentiamo degli esempi riguardanti nozioni piuttosto comuni che possono essere considerati problemi preliminari per affrontare vari problemi più elaborati.

I primi problemi ai quali si pensa solitamente sono quelli riguardanti oggetti numerici da elaborare al fine di ottenere un risultato ancora numerico.
Ad esempio: dati due interi positivi m ed n, determinare il loro massimo comun denominatore [B25].

Vicini a questi si collocano numerosi problemi geometrici. Per esempio: localizzare il centro della circonferenza passante per tre punti del piano P_1, P_2 e P_3 [G31].

Tra i primi problemi da prendere in esame si devono segnalare anche vari problemi che riguardano configurazioni discrete; ad esempio la determinazione di un cammino euleriano sopra un grafo G [D26].

C20 a.02 Ad un livello un po' più astratto si collocano i problemi riguardanti espressioni matematiche. Per esempio: date due espressioni algebriche usuali (cioè riguardanti operandi numerici, variabili numeriche e operatori aritmetici) $\mathcal{E}_1$ ed $\mathcal{E}_2$, determinare se sono equivalenti o meno.

Altri problemi si incontrano nello studio dell'informatica e nella pratica di programmazione; si pensi per esempio al disporre nell'ordine lessicografico una sequenza di stringhe molto estesa $w_1, ..., w_s$.

Vengono ampiamente studiati problemi più articolati che si pongono in discipline scientifiche come la fisica o la chimica e in vari settori tecnologici. Dopo di questi andrebbero segnalati problemi di grande impatto sulla società che si pongono in settori produttivi, organizzativi e amministrativi; si tratta sempre di problemi compositi che in genere si devono servire dei procedimenti risolutivi adottati per molteplici problemi affrontati nelle discipline fondamentali e negli settori tecnologici di ampia valenza.

Si deve però aggiungere che in questa sede ci occupiamo di problemi solo quando possono essere riformulati in termini strettamente matematici e informatici; queste riformulazioni richiedono di precisare modellizzazioni anche molto impegnative che esulano dagli scopi delle presenti pagine.

C20 a.03 I problemi affrontati in ambito informatico, come il problema di ordinamento sopra accennato, possano invece essere formulati in termini matematici senza che sia necessario ricorrere ad approfondite conoscenze specialistiche dei settori applicativi.

Questo è consentito anche all'alto livello di affidabilità degli odierni strumenti per l'elaborazione automatica dei dati, qualità che consente di affrontare gran parte delle elaborazioni automatiche come evoluzioni di sistemi formali.

Un problema di elaborazione automatica dei dati che abbia una impostazione sufficientemente precisata, a nostro parere, non presenta caratteri di sostanziale diversità rispetto ai problemi discreti che, come quelli sopra accennati, sono normalmente dibattuti negli ambienti dei matematici.

Conviene tuttavia segnalare una differenza rilevante tra i problemi affrontati nella matematica discreta e i problemi più tipici dell'informatica applicata: tendenzialmente un problema studiato in ambito informatico riguarda situazioni o processi caratterizzati da una elevata quantità di dati i quali presentano poche simmetrie che possano influenzare i procedimenti risolutivi. JP Riteniamo comunque che in queste pagine non sia opportuno stabilire una linea di demarcazione tra problemi della matematica discreta e i problemi dell'informatica.

Le minori simmetrie e le maggiori estensioni delle formulazioni dei problemi informatici, oltre alla pressante richiesta di soluzioni effettive che in genere li accompagna, inducono ad affrontarli con un atteggiamento più empirico.

Occorre peraltro osservare che, attualmente, l'interesse per le indagini sperimentali vada crescendo anche nell'ambito matematico, di pari passo con il crescere dell'efficienza, dell'efficacia e della versatilità dei sistemi per le elaborazioni autonmatiche, unite a quelle delle infrastrutture telematiche.

C20 a.04 Occorre aggiungere che nelle prossime pagine dedicheremo poca attenzione ai problemi nei quali intervengono questioni di approssimazione, non certo in quanto giudicati poco importanti, ma per il fatto che il loro studio richiede l'analisi degli errori dei calcoli e conseguentemente varie nozioni di statistica e probabilità che in questa *esposizione*, purtroppo, finora sono sostanzialmente assenti.

C20 a.05 Torniamo alla definizione iniziale e cerchiamo di chiarire cosa si intende con i termini "portata ampia", "formulazione precisa" e "procedimento ben definito".

L'ampiezza della portata dei problemi è stata richiesta in quanto ogni studio scientifico o tecnologico intende contribuire, più o meno direttamente allo sviluppo di strumenti di ampia portata e a promuovere conoscenze che servano a realizzare una buona *economia di pensiero*.

Interessano relativamente poco i procedimenti che consentono di trattare solo situazioni molto particolari; si vogliono invece individuare ed analizzare procedimenti di validità potenzialmente generale, applicabili a gamme significativamente estese di situazioni.

Per esempio interessa stabilire come calcolare il MCD di due interi positivi, quali essi siano. Un procedimento che valesse solo per poche coppie di numeri sarebbe di scarso interesse matematico e qui risulterebbe dispersivo.

Occorre però osservare che, in una ottica costruttiva che vuole essere aperta, al criterio della portata generale dei procedimenti non si può assegnare una prevalenza esclusiva.

In linea di massima molti problemi si possono trattare sia con procedimenti generali poco efficienti che con procedimenti più efficienti, ma di portata limitata.

Quando si affronta uno di questi problemi, le esigenze applicative specifiche e i mezzi disponibili influiscono su una scelta tra generalità ed efficienza, e spesso inducono a cercare un compromesso tra le due.
Va aggiunto che la stessa determinazione di buoni compromessi in molti ambiti costituisce una problematica di rilevante interesse generale.

Bisogna inoltre tenere presente che si possono adottare diversi criteri per valutare l'efficienza di un procedimento. Questa tematica sarà affrontata nel capitolo sulla complessità computazionale [C47].

C20 a.06 La richiesta di chiarezza per la questione che sta all'origine di un problema riguarda la natura degli oggetti da esaminare o da costruire (natura che nei semplici esempi inizialmente riportati è sicuramente palese) e riguarda gli insiemi nei quali degli oggetti ricercati possono essere reperiti.

Per i parametri che contraddistinguono una delle situazioni che si possono esaminare useremo il termine **dati del problema**.

Una scelta particolare per i valori di questi parametri determina una cosiddetta **istanza del problema**. Per esempio una istanza del problema del MCD è data da una particolare coppia di valori per m ed n, per esempio da $\langle 18, 42\rangle$.

Per avere i dati di un problema sufficientemente chiari, in linea di principio, è necessario esprimerli con un formalismo preciso. Per questa esigenza le istanze di ogni problema andrebbero individuate mediante le stringhe di un opportuno linguaggio formale.

La precedente è una richiesta di principio; nella pratica dello studio dei problemi ad un livello di generalità poco elevato, se si devono trattare oggetti piuttosto familiari, conviene presentarli in modo discorsivo senza complicare inutilmente i discorsi.
L'esigenza di un preciso linguaggio formale risulta invece importante quando si vuole studiare o implementare un sistema che consenta di risolvere mediante automatismi una gamma di problemi ampia, soprattutto se non facile da circoscrivere.

C20 a.07 Osserviamo che la richiesta precedente comporta che i problemi riguardino oggetti esprimibili finitamente (a questo proposito si veda anche C16a .

Ci si può allora chiedere se non sia possibile considerare oggetti appartenenti alla matematica del continuo, in particolare oggetti esprimibili con numeri reali, come i punti P_1, P_2 e P_3 del piano per i quali si vuol far passare la circonferenza dell'esempio di problema geometrico in a01 .

In effetti gli strumenti governati dei quali disponiamo consentono di affrontare tangibilmente, cioè con la prospettiva di giungere a risposte effettive, esplicitamente formulabili, solo per problemi esprimibili finitamente con oggetti matematici discreti.
Una istanza di un problema di calcolo coinvolgente numeri reali può essere affrontato con successo solo se questi vengono espressi costruttivamente, mediante scritture finite.

Queste scritture possono riguardare approssimazioni dei numeri reali: in tal caso si avrebbe un problema di calcolo approssimato, accompagnato dalle complicazioni cui abbiamo già accennato.

Un'altra possibilità è quella di trattare i numeri mediante espressioni costruttive: questo, oltre che per i numeri interi ed i razionali, è possibile per i numeri algebrici come $\sqrt{p}$ con p intero positivo che non sia quadrato di un altro intero positivo, e per numeri come π ed e individuati da serie o da altri meccanismi che consentono di approssimarli, in linea di principio, quanto si vuole.
Più operativamente si è indotti a cercare approssimazioni precise quanto lo richiedono le soluzioni che vengono richieste dalle varie esigenze applicative che hanno motivato le attività risolutive.
I problemi numerici non si possono risolvere tangibilmente su numeri reali individuati soltanto mediante loro proprietà generali (derivate da assiomi), ma solo su **numeri reali effettivamente costruibili**, ossia con numeri che si dimostrano essere illimitatamente approssimabili (si usa dire "approssimabili quanto si vuole").

Un problema presentato mediante espressioni su variabili reali e un algoritmo che consente di trovarne soluzioni generali sono da considerare, risp., un problema di valutazione simbolica e un procedimento riguardante una elaborazione simbolica: in altri termini si tratta di un problema di elaborazione di stringhe con caratteristiche specifiche e della sua soluzione: anche in questo caso ci si muove nell'ambito delle entità matematiche discrete.

C20 a.08 Veniamo ora ai "procedimenti ben definiti" che consentono di risolvere i problemi computazionali.

Per i semplici problemi presentati i procedimenti risolutivi sono ben noti.
Lo stesso accade per molti altri problemi: nella pratica matematica e informatica si sono posti con chiarezza molti problemi e spesso si sono trovati procedimenti ben definiti che consentano di trovar loro una risposta.

Volendo svolgere considerazioni generali sulle attività computazionali dobbiamo cercare di chiarire meglio quali sono i confini di quella che abbiamo chiamato collezione dei procedimenti risolutivi.
Per questo è necessario chiarire che cosa non possa considerarsi procedimento risolutivo, ovvero che cosa non è lecito pretendere da ogni ipotetico procedimento risolutivo.

Osserviamo che si tratta di un'intenzione piuttosto ambiziosa. La famiglia dei problemi computazionali che oggi la matematica e l'informatica (nonché tutte le altre discipline che si servono di metodi quantitativi) sanno affrontare è estremamente vasta e variegata.

Questa famiglia inoltre si trova in una fase di una grande espansione iniziata intorno al 1950, soprattutto in relazione allo sviluppo degli elaboratori elettronici, ed alla crescita delle esigenze applicative e delle metodologie di calcolo che l'hanno accompagnata.

C20 a.09 In effetti, come vedremo nelle pagine che seguono [v.o.], una definizione dei procedimenti risolutivi non si sa dare in modo pienamente soddisfacente, come accade invece a tante classiche entità della matematica pura.

D'altro lato [C21] si sono individuati vari tipi di procedimenti o di strumenti governati per i quali si hanno buone ragioni per sostenere che abbiano portata generale, cioè che siano in grado di risolvere uno qualsiasi dei problemi che è lecito chiamare "computazionali".
Questi strumenti governati, tra l'altro si sono rivelati sostanzialmente equivalenti, in particolare equivalenti-L.

Questo induce a pensare e la grande varietà dei problemi computazionali può essere affrontata con modalità aventi caratteristiche in comune, tra le quali si possono almeno cercare collegamenti che consentano di definirle secondo una visione in qualche modo unitaria.

Questa unitarietà riguarda sia il piano matematico che quello informatico: infatti bisogna tenere presente che oggi la tecnologia rende ampiamente disponibili strumenti sostanzialmente equivalenti e non tanto diversi da quelli sopra accennati i quali consentono di rendere operative estese famiglie di procedimenti computazionali.

Occorre tuttavia aggiungere che la possibilità di affrontare in modo unitario i problemi computazionali deve essere ridimensionata quando si dà peso alle differenze tra i procedimenti risolutivi in termini di precisione, velocità, portata e versatilità.

In effetti risulta necessario farsi idee chiare anche sui limiti della nostra capacità di risolvere in modo soddisfacente i problemi. A questo contribuisce in particolare lo studio della complessità computazionale [C47].

C20 a.10 Complessivamente negli ultimi decenni la consapevolezza del carattere unitario delle attività computazionali e dei loro limiti ha contribuito notevolmente allo sviluppo dei procedimenti disponibili.

I risultati scientifici e tecnici che questi metodi a loro volta hanno consentito ulteriori sviluppi che hanno favorito l'attuale ampia e crescente disponibilità di strumenti edi metodi computazionali di elevata efficacia e versatilità.

Si deve anche aggiungere che le prospettive di sviluppo delle tecnologie informatiche aprono per gli strumenti e i metodi computazionali la possibilità di andare ben oltre ai risultati finora conseguiti.

Una chiara percezione della loro portata e dei loro limiti costituisce un grande aiuto per i futuri sviluppi e, nei nostri giorni, dovrebbe costituire una indicazione culturale e sociale di primaria importanza.

A nostro parere, tuttavia, le potenzialità culturali e sociali della strumentazione computazionale, pur essendo in continua crescita, nel loro complesso non vengono sufficientemente percepite e apprezzate, sia da parte di chi è incaricato di scelte di vasta portata, sia dall'opinione pubblica.

C20 a.11 La precedente affermazione richiede qualche precisazione: infatti si possono segnalare alcuni organismi che al fine di perseguire loro obiettivi generali investono notevoli risorse per lo sviluppo di strumenti computazionali che stanno portando a successi molto notevoli, almeno in termini di efficienza complessiva.

Per questo proponiamo solo segnalazioni schematiche.
Controllo delle transazioni finanziarie (attività finanziarie internazionali).
Controllo esplicitamente autoritario della cittadinanza (in particolare la Cina, sistematicamente, e vari stati totalitari, settorialmente).
Imprese ad alta spettacolarità (conquista di spazi extraterrestri come Luna e Marte, soprattutto per motivi di prestigio).
Cyberattacchi volti a destabilizzare organismi con elevata influenza.
Imperialismo nella logistica (Amazon), della pubblicità (Google), dell'intrattenimento basato sullo streaming (Sky, Netflix, Amazon, Walt Disney e altre multinazionali).
Armamenti (gran parte degli stati tecnologicamente attrezzati).

Una caratteristica prevalente di queste iniziative sta nel fatto che gli obiettivi si concentrano su vantaggi settoriali e ignorano ampiamente le conseguenze svantaggiose per grandi masse di persone.

C20 b. nozione di algoritmo

C20 b.01 Cerchiamo ora di precisare il termine "procedimento" usato nell'enunciato di apertura del capitolo [a01]. in modo di avvicinarci alla nozione di algoritmo.

Fin dai primi sviluppi del pensiero matematico si è avuta la spinta a precisare "indicazioni operative" che potessero essere seguite meccanicamente da operatori umani sufficientemente affidabili per risolvere dei problemi di elevato interesse applicativo (calcoli commerciali e finanziari, misurazioni del terreno, costruzioni di edifici e di macchine, valutazioni astronomiche, ...).

In effetti queste indicazioni operative fanno sì che numerose istanze di un problema possano essere risolte da persone alle quali non si richiedono profonde competenze né grande inventiva, ma solo che devono avere la capacità di eseguire elaborazioni su numeri, su figure o su altre informazioni singolarmente semplici seguendo attentamente regole computazionali chiaramente espresse.

Così facendo si riducono i rischi delle attività lasciate alla iniziativa di esecutori ingenui, che agiscono non sorrette da concezioni e istruzioni adeguatamente meditate.

C20 b.02 I progressivi sviluppi del pensiero matematico hanno visto la crescita di importanza delle astrazioni, in parte collegata alla opportunità di sviluppare indagini matematiche ad alto livello di generalità al fine (piŏ meno espresso) di realizzare rilevanti economie di pensiero, in parte sulla spinta del bisogno di individuare verità nascoste indagando platonicamente al di là di quanto immanentemente percepibile.

L'attenzione dei cultori della matematica si è quindi rivolta più spesso, specialmente nella prima metà del '900, alle ampie costruzioni teoriche e spesso le difficoltà superate e il fascino delle soluzioni trovate hanno posto in ombra le originarie motivazioni computazionali.

Nello sviluppo dei metodi risolutivi spesso si è attribuita importanza molto maggiore all'ampliamento della portata, piuttosto che alla efficienza e alla concreta applicabilità.

Prima dello sviluppo delle tecnologie computazionali digitali la concreta applicabilità presentava prospettive pratiche limitate e attrattive intellettuali modeste.

Dopo i primi successi della tecnologia dell'informazione, molte persone interessate alla matematica sono state indotte a considerare di scarso interesse le attività di computazione effettiva, proprio in quanto i calcoli tradizionali sono stati resi routinari e demandabili a strumenti ampiamente disponibili.

C20 b.03 Con i successivi sviluppi delle apparecchiature elettroniche e con l'ampliarsi delle applicazioni dei metodi quantitativi le attività di calcolo effettivo sono diventate capillarmente diffuse e molto differenziate.

La loro importanza è cresciuta in tutti i settori della produzione, della gestione, della amministrazione e, naturalmente, nella maggior parte delle settori della ricerca.

Dopo facili successi iniziali (indicativamente tra 1960 e 1970), si è osservato che per una buona soluzione di molti problemi importanti risulta cruciale riuscire spingere molto avanti le capacità di calcolo effettivo, nelle direzioni talora conflittuali della portata, della velocità, della precisione e della versatilità.

È quindi tornata pressante l'esigenza di chiarire indicazioni generali per i procedimenti di calcolo, insieme all'esigenza di allargare le conoscenze teoriche che possono suggerire nuovi metodi e rendere affrontabili nuove problematiche.

C20 b.04 Le indicazioni di procedimenti risolutivi in grado di giungere a risposte definite le quali siano espresse in una forma coerente e priva di ambiguità e siano delegabili a numerosi operatori umani o meccanici incaricati di svolgere concretamente le elaborazioni, vengono dette **algoritmi**.

Questo termine deriva dalla distorsione, in epoca medioevale di "al Khwarizmi", il soprannome significante "originario della regione in occidente nota come Corasmia" del matematico, noto anche come Muhammad Ibn Musa, che operò a Baghdad nel nono secolo e scrisse un trattato di matematica a carattere computazionale che nel mondo arabo-bizantino-europeo fu un *best seller* per vari secoli.

Per chiarire le caratteristiche degli algoritmi può essere utile riprendere in esame alcune indicazioni di procedimenti risolutivi elementari.
Come esempi si possono tenere presenti le regole per lo svolgimento delle operazioni aritmetiche sulle scritture decimali degli operandi, le loro varianti binarie, la determinazione del MCD di interi e di polinomi mediante l'algoritmo euclideo, le regole per la decomposizione di un intero in fattori primi, i procedimenti per la soluzione di un sistema di equazioni lineari con coefficienti razionali, le regole per le costruzioni geometriche classiche effettuabili con riga e compasso, l'individuazione di un percorso minimo sopra un grafo.

C20 b.05 Da questi esempi si ricavano quelle che, introdotte da Kolmogorov e Andrey Markov Jr, si considerano le caratteristiche degli algoritmi.

[a] Un algoritmo è costituito da un complesso di istruzioni espresse con enunciati finiti rivolte a un esecutore reale o ragionevolmente realizzabile che può avere natura umana, meccanica, elettronica, ... (**finitezza della descrizione di un algoritmo** ed **eseguibilità di un algoritmo da parte di suoi esecutori**).

[b] Un algoritmo descrive un processo per la costruzione, a partire da alcune entità date, di nuove entità obiettivo. Queste entità possono essere di varia natura (numerica, geometrica, simbolica, procedurale, ...); si chiede solo che esse possano avere una precisa definizione matematica o comunque siano riconoscibili sulla base di precise informazioni (**finitezza e definitezza dei dati e degli obiettivi di un algoritmo**).

[c] Il processo eseguito dall'esecutore al quale si affida l'effettuazione dell'algoritmo si constata che si realizzi o si dimostra che possa realizzarsi attraverso successivi **passi** che collegano successive **stadi**, questi riferibili a una sequenza finita o a una successione di **istanti di riferimento**.

Nello stadio iniziale, e quindi nell'istante iniziale, all'esecutore è noto solo il sistema (finito) di informazioni che rappresenta il complesso delle condizioni delle entità interessate.

In ogni passo successivo l'esecutore è in grado di ottenere un nuovo sistema di informazioni che può dipendere da tutte le informazioni che sono a sua disposizione nello stadio precedente (**discretezza, automatismo e capacità di memoria di un algoritmo**).

[d] Le informazioni ottenute in ciascun passo sono determinate univocamente dalle informazioni possedute nello stadio precedente (**determinismo di un algoritmo**).

[e] Le regole che consentono di effettuare un passo esecutivo sono tendenzialmente semplici e locali, cioè coinvolgono un insieme circoscritto e completamente controllabile di oggetti (**elementarità delle istruzioni e dei passi di un algoritmo**).

[f] Il complesso delle entità date in partenza può essere scelto in un insieme tendenzialmente esteso chiaramente definibile (**massività o ampiezza della portata di un algoritmo**).

C20 b.06 Queste indicazioni non costituiscono una definizione pienamente soddisfacente sul piano matematico; infatti esse contengono termini (processo, norma operativa, semplice, locale, estesa) di significato piuttosto vago.

Si può però dire che la precedente costituisce una **definizione programmatica e consensuale** della nozione di algoritmo, tesa principalmente a delineare prospettive.

Per ogni procedimento computazionale si tratta di stabilire se si trova consenso nel valutare che le precedenti caratteristiche [a],...,[f] sono soddisfatte. Per i procedimenti accennati all'inizio e in b04 e per molti altri questo consenso si raccoglie facilmente.

La nozione di algoritmo esposta in b05 nel passato veniva adottata, più o meno consapevolmente, ogni qual volta si prendeva in considerazione un problema del quale si venivano a conoscere procedimenti risolutivi soddisfacenti.

Alle soglie del XX secolo, però, si sono presentate questioni per le quali risultava dubbia l'esistenza di una soluzione e per le quali non si aveva accordo su cosa si dovesse considerare algoritmo e cosa no.
Quindi si è reso necessario definire meglio la nozione di algoritmo, in particolare per avere la possibilità di affermare la non esistenza di algoritmi in grado di risolvere problemi che risultavano piuttosto spinosi.

La questione della definizione degli algoritmi è stata tra quelle centrali per la matematica dall'inizio del '900 agli anni '30, in stretta relazione con i vari problemi dei fondamenti della matematica discussi, in particolare, da Hilbert.

Questa questione venne portato a chiarimento dalle ricerche di personalità come Axel Thue, Emil Post, Kurt Goedel, Alonzo Church, Stephen Kleene e Alan Turing, oltre ai già citati Andrey Markov Jr. e Andrey Kolmogorov. Su questi temi torneremo in C21.

C20 b.07 In quel periodo lo studio della natura degli algoritmi e le altre questioni della logica matematica apparivano come questioni generali e astratte, piuttosto lontane dai settori applicativi, impegnati soprattutto in questioni relative ai metodi in grado di dare informazioni per calcoli analitici e approssimati.

Successivamente, con la diffusione degli elaboratori elettronici e con lo sviluppo delle metodologie e delle tecniche del calcolo intensivo, la situazione è mutata profondamente.
Accanto alla teoria degli algoritmi si sono sviluppate teorie come quelle dei linguaggi formali, dei linguaggi di programmazione, degli automi, della complessità, dei codici, dei VLSI (circuiti a larghissima integrazione) che occupano posizioni basilari per lo studio della progettazione e delle applicazioni dei sistemi per l'elaborazione dei dati, per il controllo di processi e di impianti, per la gestione della rete globale, per la fornitura di servizi e per l'organizzazione sociale.

Queste teorie e le metodologie collegate, inoltre, contribuiscono ormai in misura rilevante alla adozione dei metodi matematici, computazionali e di gestione delle conoscenze in discipline consolidate come la biologia, la fisiologia, l'economia, l'organizzazione aziendale, la linguistica e in settori più ambiziosi ma ancora agli inizi, come la comprensione del funzioamento del cervello.

C20 c. soluzione di problemi come elaborazione di stringhe

C20 c.01 Un buon modo per collocare gli algoritmi in un quadro unitario si serve dei linguaggi formali.

Questo modo di fare inizia con la constatazione che la soluzione effettiva di ogni problema si può ricondurre ad una elaborazione delle stringhe che codificano i dati che è finalizzata a ottenere le stringhe che esprimono i risultati.
L'esempio più evidente è fornito dalle operazioni aritmetiche elementari eseguite sulle scritture decimali dei numeri dati.
Altri esempi da tenere presenti riguardano elaborazioni che forniscono una risposta positiva o negativa ad una domanda dicotomica.

Cominciamo con il ricordare che un linguaggio sull'alfabeto T, cioè un sottoinsieme L di un monoide libero T^*, si può definire costruttivamente utilizzando strumenti di diversa natura.

Si possono prendere in considerazione diverse collezioni di strumenti governati [C16a01], cioè di coppie $\langle_{tlpr}\mathbf{S}, \mathcal{M}\rangle$ con $\mathbf{S}$ collezione di strumenti in grado di agire su stringhe su qualsiasi alfabeto T e con $\mathcal{M}$ modalità per il loro utilizzo o meccanismo formale per il loro governo capace di finalizzare ogni $S \in \mathbf{S}$ alla individuazione (identificazione, riconoscimento) di un linguaggio L sul suddetto T.

Questo linguaggio si può denotare con la scrittura $\mathcal{M}(S)$, nella quale $\mathcal{M}$ assume il ruolo di una funzione definita sulla collezione $\mathbf{S}$ e che assume come valori dei determinati sottoinsiemi di T^*.

C20 c.02 Anche i primi risultati della teoria dei linguaggi formali ci forniscono una buona gamma di esempi con i quali si riesce a dare corpo alla definizione programmatica consensuale in b05. Vediamo alcuni casi particolari.

Possono essere considerati come strumenti le espressioni di un certo genere e come modalità d'uso la interpretazione di queste scritture come insiemi di stringhe su un preciso sottoalfabeto dell'alfabeto sul quale sono costruite le espressioni.
In particolare le espressioni razionali attraverso la loro interpretazione forniscono i linguaggi razionali [C12].

Un secondo genere di strumenti è dato dalle macchine formali di determinati tipi (riconoscitori di Rabin-Scott, riconoscitori a pila, accettatori di Turing, ...): da una macchina di un dato genere il corrispondente meccanismo dell'accettazione ricava un linguaggio.

Un altro tipo di strumento è costituito dalle grammatiche: da ciascuna di esse il meccanismo operativo della derivazione delle stringhe di caratteri terminali ottiene un linguaggio (come ad esempio in C14]).

Un meccanismo di generazione si applica anche alle macchine di Turing generatrici [C21], ai vari sistemi sviluppamentali di Lindenmayer [C26], ai sistemi di riscrittura regolamentati [C23] e a varie altri strumenti governati.

Un po' più astrattamente possiamo considerare presentazioni finite ottenute da sistemi di equazioni per linguaggi e da corrispondenti meccanismo risolutivi che risultino concretamente attuabili.

C20 c.03 A questo punto si può essere ragionevolmente sicuri che numerose presentazioni effettive di linguaggi ricadano sotto lo schema precedente, cioè che una estesa gamma di linguaggi costruibili (e quindi utilizzabili) si possano presentare con espressioni della forma $\mathcal{M}(S)$.

Numerosi linguaggi costruibili, quindi, si possono presentare fornendo le descrizioni di uno strumento S e di una modalità per il suo utilizzo $\mathcal{M}$ attraverso espressioni finite o descrizioni operative formalizzate con opportuni alfabeti e rispettando meticolose regole sintattiche.

Ne segue che l'insieme dei linguaggi su un dato alfabeto effettivamente definibili, come le stringhe di un alfabeto T, hanno il cardinale del numerabile, $\aleph_0$, nettamente inferiore al cardinale della totalità dei linguaggi su T che è il cardinale del continuo, $\aleph_1$.

C20 c.04 La possibilità di utilizzare fattivamente le presentazioni dei linguaggi pone vari problemi che si possono organizzare in problematiche piuttosto generali. Vediamone alcune.

Facendo riferimento ad un alfabeto T, si dice **problema dell'appartenenza** per lo strumento S che agisce su stringhe che possono coinvolgere T e per la sua modalità di governo $\mathcal{M}$, ossia il problema dell'appartenenza per il linguaggio $\mathcal{M}(S)$ e ancor più concisamente il problema $\mathsf{PRBmember}[\mathcal{M}(s), \mathsf{T}]$ il problema dello stabilire per ogni stringa $w \in \mathsf{T}^*$ se $w \in L := \mathcal{M}(s)$ o se viceversa $w \notin L$ [v.a. B18e01].

Si dice che il problema dell'appartenenza per $\mathcal{M}(S)$ è un **problema decidibile** sse si conosce un algoritmo $\mathcal{A}$ che per ogni $w \in \mathsf{T}^*$ sia in grado di decidere, in un numero finito di passi, se $w \in \mathcal{M}(S)$ o meno.

Questo algoritmo in linea di principio dipende da S, $\mathcal{M}$ e T e si può individuare con la scrittura $\mathcal{A}_{S,\mathcal{M},\mathsf{T}}$.
Esso si può interpretare come funzione che, applicata a una generica $w \in \mathsf{T}^*$, fornisce una risposta positiva, ovvero un valore 1, se w viene riconosciuta appartenere al linguaggio, mentre fornisce una risposta negativa, ovvero 0, nel caso contrario.
In altri termini l'algoritmo da conoscere è un procedimento per il calcolo effettivo della funzione indicatrice $\mathcal{I}_{\mathsf{T}^*,\mathcal{M}(S)}$.

Si può anche affermare che si ha una buona padronanza del suddetto problema di appartenenza quando per un ampio sottoinsieme $\mathbf{S}' \subseteq \mathbf{S}$ (al meglio per $\mathbf{S}' = \mathbf{S}$) si conosce uno schema di algoritmo, che possiamo individuare con $\mathcal{D}_{\mathcal{M},\mathbf{S}'}$, dal quale si ricava facilmente $\mathcal{D}_{\mathcal{M},S,\mathsf{T}}$ per ogni $S \in \mathbf{S}'$.

In questi casi alla scrittura insiemistica con la quale si definisce il linguaggio, $L := \{w \in \mathsf{T}^* \,[\!]\, w \in \mathcal{M}(S)\}$, tendenzialmente semplice, si può affiancare l'espressione $L = \{w \in \mathsf{T}^* \,[\!]\, \mathcal{D}_{\mathcal{M},S}(w) = 1\}$ che coinvolge un algoritmo effettivamente conosciuto e quindi che si può pensare di utilizzare concretamente.

C20 c.05 In generale diciamo **linguaggio ricorsivo** su T un $\mathsf{L} \in \mathsf{T}^*$ determinato da una presentazione finita $\mathsf{L} = \mathcal{M}(S)$ per la quale sia noto un algoritmo $\mathcal{A}_{\mathcal{M},S,\mathsf{T}}$ che permette di decidere il relativo problema dell'appartenenza. Denotiamo con $\mathbf{LngRcr}_{\mathsf{T}}$ la classe di tali linguaggi.

Ovviamente ogni linguaggio finito individuato dall'elenco delle sue stringhe è ricorsivo. Questo elenco costituisce una sua presentazione effettiva e data una w qualsiasi basta verificare se compare nell'elenco per decidere se appartiene o meno al linguaggio.

Sono note altre ampie collezioni di linguaggi ricorsivi: in particolare tutti quelli individuati utilizzando con il meccanismo della accettazione da parte di riconoscitori deterministici.
Per essi l'algoritmo per l'appartenenza è fornito dal funzionamento dello stesso riconoscitore. Sono quindi ricorsivi i linguaggi razionali e tutti i linguaggi accettati da un automa deterministico capace di concludere in un numero finito di passi la sua evoluzione conseguente alla presa in esame di una qualsiasi stringa del suo alfabeto di ingresso.

C20 c.06 Dei problemi di appartenenza fanno parte problemi che si incontrano in vari campi.

È un problema di appartenenza il problema della decidibilità delle proposizioni ben formate in una qualsiasi teoria formalizzata [B60, B61].

Si consideri un qualsiasi tipo di insieme discreto ben definito (ad esempio l'insieme dei numeri primi o quello dei grafi euleriani) ed una modalità per la codifica dei suoi elementi.
La decidibilità del problema dell'appartenenza per il relativo linguaggio di codifica corrisponde alla possibilità di delimitare effettivamente attraverso un algoritmo $\mathcal{D}$ l'insieme discreto in esame nell'ambito del monoide libero nel quale va collocato.

C20 c.07 Finora abbiamo considerato solo esecuzioni di algoritmi che per ogni istanza del problema si concludono dopo un numero finito di passi. Occorre considerare anche meccanismi con le caratteristiche di definitezza e determinismo degli algoritmi, ma che possono operare indefinitamente, cioè tanto quanto si vuole, senza peraltro escludere la possibilità che si arrestino dopo un numero finito di passi. Questi meccanismi vengono chiamati **procedure**.

A questo tipo di meccanismo appartengono le **procedure generative** o procedure elencative, meccanismi che sono in grado di procedere con successione di passi che potrebbe continuare illimitatamente e che in alcuni di questi passi emettono una stringa su un nastro di uscita, cioè sopra un dispositivo in grado di registrare sequenze di caratteri.

Questa emissione si può configurare come scrittura di una stringa ben formata che costituisce la codifica di una entità significativa.

Una procedura elencativa basilare è quella in grado di "presentare tutte le stringhe" di un monoide libero T^* secondo un ordine sequenziale, per esempio secondo un ordine-llx, un ordinamento che procede ad elencare primariamente le stringhe di lunghezze crescenti e, secondariamente, stringhe di data lunghezza seguendo un ordine lessicografico.

C20 c.08 La precedente espressione "presentare tutte le stringhe", volendo rispettare la realtà, chiaramente, non si può applicare a operazioni giunte ad una conclusione osservabile; ogni operazione per poter essere osservata deve svolgersi e concludersi in un intervallo di tempo finito.

Questa aspressione va considerata una semplificazione dell'espressione "procedere alla emissione di stringhe quanto si vuole (ovvero, quanto viene richiesto dall'applicazione che l'ha motivata) e quanto consentono le risorse (di tempo e di memoria) che le circostanze rendono effettivamente disponibili.

In effetti si conoscono procedure elencative che si concludono dopo un numero finito di passi, altre che si dimostrano in grado di proseguire illimitatamente (ammesso che motivazioni, risorse disponibili ed eventi ambientali lo permettano) ed altre che non si sanno collocare in una dei due precedenti raggruppamenti.

Questa incertezza si riscontra anche di fronte a insiemi definibili con procedure elencative che non si giungano a una conclusione finita o meno.

A questo punto si rende necessario individuare linguaggi diversi dai linguaggi ricorsivi definiti in c05.

Diciamo dunque **linguaggio ricorsivamente enumerabile** su T un sottoinsieme di T^* per il quale si conosce una procedura generativa in grado di emettere le sue stringhe e non altre. La classe di tali linguaggi la denotiamo con $\mathbf{LngREn}_{\mathsf{T}}$.

Occupandoci di insiemi diciamo **insieme ricorsivamente enumerabile** un insieme i cui elementi sono esprimibili con le stringhe di un linguaggio in $\mathbf{LngREn}_{\mathsf{T}}$ per qualche alfabeto T. La collezione di tali insiemi la denotiamo con **SetREn**.

Un **linguaggio ricorsivo** su T si può anche definire come un linguaggio sottoinsieme di T^* per il quale si conosce una procedura elencativa in grado di procedere con l'emissione di tutte le sue stringhe e solo di esse in modo da consentire di risolvere il corrispondente problema di appartenenza.

Denotata con $\mathbf{LngRcr}_{\mathsf{T}}$ la collezione dei linguaggi ricorsivi sull'alfabeto T, definiamo **insieme ricorsivo** un insieme i cui elementi sono esprimibili con le stringhe di un linguaggio in $\mathbf{LngRcr}_{\mathsf{T}}$ per qualche alfabeto T. La collezione di tali insiemi la denotiamo con **SetRcr**.

C20 c.09 **Teorema** Ogni linguaggio ricorsivo è ricorsivamente enumerabile.

Dim.: Consideriamo un linguaggio $L \in \mathbf{LngRcr}_{\mathsf{T}}$, una sua presentazione $\mathcal{M}(S)$ e un algoritmo $Dcl_{\mathcal{M},S}$ che permettere di decidere il problema dell'appartenenza per lo stesso L.
Una procedura elencativa per tale linguaggio si ottiene servendosi della procedura cheprocede a generare di tutte le stringhe di T^* e facendo agire DCl come filtro su ogni stringa generata in modo che siano emesse solo le stringhe accettate come appartenenti ad L ∎

Osserviamo che le stringhe emesse dalla macchina sopra prospettata sono presentate secondo l'ordine seguito dalla procedura generativa di T^*, per esempio secondo un ordine-llx.

Nella definizione generale, invece, non si è chiesto che le stringhe di un linguaggio ricorsivamente enumerabile siano emesse secondo un ordine sequenziale ben definito e quindi senza ripetizioni.
Resta quindi aperta la possibilità di individuare qualche linguaggio ricorsivamente enumerabile che non sia ricorsivo.

C20 c.10 Se volessimo servirci di una procedura elencativa $\mathcal{E}$ per decidere l'appartenenza a un linguaggio $L \in \mathbf{LngREn}_{\mathsf{T}}$ di una certa $w \in \mathsf{T}^*$ si possono incontrare due situazioni.

Nel caso "migliore" dopo un numero finito di passi si ha l'emissione di w e quindi risulta decisa la sua appartenenza ad L.

Nel caso "peggiore" dopo un certo numero di passi, che potrebbe essere giudicato impegnativo per le risorse disponibili, la w non è stata emessa e quindi l'appartenenza rimane indecisa.

A questo punto il responsabile dell'esecuzione dovrebbe stabilire se far proseguire la procedura: forse se operasse ancora potrebbe garantire la appartenenza di w ad L; per contro farla operare ulteriormente risulta costoso, forse eccessivamente, e potrebbe rivelarsi inutile.

Una tale situazione di incertezza può effettivamente verificarsi nella pratica del trattamento effettivo mediante apparecchiature informatiche di problemi impegnativi, tipicamente nella ricerca di una soluzione di un problema che richiede un oggetto che deve soddisfare requisiti precisi.
È dunque molto importante riuscire a evitare tali situazioni.

Se al contrario si affronta una attività elencativa per la quale si dispone di una procedura in grado di emettere le stringhe dell'ambiente richiesto T^* in un ordine ben determinato, risulta garantito il carattere ricorsivo del linguaggio individuato.

Infatti di fronte alla necessità di sapere se una determinata stringa w appartiene a un $L \in \mathbf{LngRcr}$, dopo un certo numero $n(|w|)$ di passi valutabile con una qualche ragionevolezza, se la w non è stata emessa è sicuro che essa non potrà più essere prodotta, ed è lecito concludere che $w \notin L$.

Va anche segnalato che in taluni casi la valutazione di n potrebbe essere tanto elevata da rendere inattuabile l'effettiva prosecuzione a causa della scarsità delle risorse disponibili.

C20 c.11 Osserviamo che sono ricorsivamente enumerabili tutti i linguaggi individuati da uno dei riconoscitori appartenenti al livello-S e a fortiori al livello-R e al livello-F, sia del genere deterministico che del nondeterministico [C12c, C14d, ...].
Ogni linguaggio ricorsivamente enumerabile è per definizione finitamente presentabile: infatti la sua prospettata procedura enumerativa è uno strumento atto a presentarla ed è finitamente individuabile.

Si potrebbe anche ritenere che ogni presentazione finita di un linguaggio si possa ricondurre a una procedura che elenca le sue stringhe e quindi far coincidere i linguaggi ricorsivamente enumerabili con i linguaggi finitamente presentabili.
Peraltro nel discorso che stiamo sviluppando le due nozioni non sono ancora definite in modo sufficiente per poter affermare l'equivalenza suggerita.
Per questo sarà opportuno appoggiarsi a qualche nozione meglio definita: questa questione sarà ripresa dopo l'esame delle macchine di Turing [C21].

C20 c.12 Prop. Anche il complementare $L^{\complement} := \mathsf{T}^* \setminus L$ di un linguaggio ricorsivo L è ricorsivo.

Infatti se $\mathcal{D}$ è un algoritmo in grado di decidere l'appartenenza di una $w \in \mathsf{T}^*$ ad L, basta trasformrlo cambiando l'accettazione conclusiva nella non accettazione e viceversa per avere un algoritmo che decide il problema cell'appartenenza a $L^{\complement}$ ∎

C20 c.13 Prop. Un linguaggio L è ricorsivo sse sia esso che il suo complementare sono ricorsivamente enumerabili.

Dim.: Abbiamo già visto che se L è ricorsivo è anche ricorsivamente enumerabile; inoltre anche $L^{\complement}$ è ricorsivo e quindi ricorsivamente enumerabile.
È invece un po' macchinoso dimostrare che se L ed $\mathsf{L}^{\complement}$ sono ricorsivamente enumerabili, allora L è ricorsivo (e tale à anche $\mathsf{L}^{\complement}$).

Per questo occorre prospettare due procedure enumerative $\boldsymbol{P}$ e $\overline{\boldsymbol{P}}$ in grado di generare, risp., le stringhe di L e di $\mathsf{L}^{\complement}$ e di utilizzarle per individuare un algoritmo $\mathcal{D}$ che consenta di decidere il problema dell'appartenenza ad L.
Questo algoritmo quando è incaricato di decidere l'appartenenza della stringa w ad L procede a eseguire una successioni di fasi operative in ciascuna delle quali demanda alternativamente a $\boldsymbol{P}$ e a $\overline{\boldsymbol{P}}$ l'esecuzione di un loro ulteriore passo.
Quando uno di questi passi non comporta alcuna emissione, $\mathcal{D}$ si limita a passare alla fase successiva.
Se invece un passo di $\boldsymbol{P}$ o di $\overline{\boldsymbol{P}}$ porta all'emissione di una stringa x, $\mathcal{A}$ controlla se essa coincide con la stringa w; in caso positivo è in grado di fornire una decisione conclusiva: se w è stata emessa da $\mathcal{P}$ stabilisce che $w \in L$; se è stata emessa da $\mathcal{P}'$, afferma che $w \notin L$.
Ad una di queste decisioni si deve sempre arrivare in un numero finito di passi, in quanto ogni stringa del monoide T^* deve essere emessa ad un certo passo da $\mathcal{P}$ oppure da $\mathcal{P}'$ ∎

C20 c.14 A questo punto ci proponiamo di individuare un linguaggio ricorsivamente enumerabile che non è ricorsivo, e quindi dimostrare che la collezione dei linguaggi ricorsivi è contenuta propriamente nella collezione dei ricorsivamente enumerabili.

Questa dimostrazione si serve di una cosiddetta costruzione diagonale, del genere introdotto da Cantor per dimostrare che $\aleph_0 < \aleph_1$, ovvero per negare la possibilità di porre in corrispondenza biunivoca l'insieme dei reali compresi tra 0 ed 1 con l'insieme dei naturali (ovvero di avere una biiezione tra i linguaggi sopra un alfabeto T e le stringhe su T).

C20 c.15 Prop. Esiste un linguaggio ricorsivamente enumerabile il cui complementare non è ricorsivamente enumerabile.

Dim.: Dato che tutte le procedure generative di stringhe su un alfabeto T devono potersi esprimere finitamente con un alfabeto $\overline{T} \supset \mathsf{T}$, il loro insieme può essere sequenzializzato in quanto può essere sequenzializzato l'insieme delle loro descrizioni dttagliate.

Facciamo quindi riferimento a una successione delle procedure generative riguardanti tutti i linguaggi ricorsivamente enumerabili sopra T, $\langle \mathcal{P}_0, \mathcal{P}_1, ..., \mathcal{P}_n, ... \rangle$ (ad esempio una successione che segue un ordine-llx per e alla corrispondente successione di tutti i linguaggi in $\mathbf{LngREn}_{\mathsf{T}}$, $\langle \mathsf{L}_0, \mathsf{L}_1, ..., \mathsf{L}_n, ... \rangle$.
Consideriamo inoltre una successione di tutte le stringhe di T^* $\langle w_0, w_1, ..., w_n, ..., \rangle$ (ad esempio una successione che segue un ordine-llx per $\overline{T}^*$) e quindi il linguaggio

$$\overline{L} \ := \ \{ w_n \in \mathsf{T}^* \ [\!] \ w_n \in \mathsf{L}_n \} \ .$$

Mostriamo che $\overline{L} \in \mathbf{LngREn}_{\mathsf{T}}$ individuando una procedura $\overline{\boldsymbol{P}}$ in grado di procedere a generare le sue stringhe.
Dato che ogni procedura $\mathcal{P}_n$ esegue una successione di passi, si può associare a ogni $c_{n,i} := \langle n, i \rangle \in \mathbb{N} \times \mathbb{N}$ il passo i-esimo eseguito della procedura $\mathcal{P}_n$, convenendo di associare la semplice inazione al caso in cui la $\mathcal{P}_n$ si sia arrestata prima del passo i (questo accade sse L_n è un linguaggio finito).
Si può quindi procedere a scorrere tutti i passi eseguiti da tutte le procedure generatrici di stringhe su T secondo l'ordinamento diagonale alla Cantor per $\mathbb{N} \times \mathbb{N}$: $\langle 0,0 \rangle$, $\langle 0,1 \rangle$, $\langle 1,0 \rangle$, $\langle 0,2 \rangle$, $\langle 1,1 \rangle$, $\langle 2,0 \rangle$, $\langle 0,3 \rangle$, ...
La coppia $\langle n, i \rangle$ risulta associata alla fase esecutiva numero $\dfrac{(i+n)^2 + 3\,n + i}{2} + 1$ della $\overline{\boldsymbol{P}}$.
Questa fase consiste nel ricavare dalla simulazione di $\mathcal{P}_n$, informazione per la quale è necessario che la $\overline{\boldsymbol{P}}$ sia in grado di simulare tutte le procedure $\mathcal{P}_0$, $\mathcal{P}_1$, ... , $\mathcal{P}_n$
Si tratta quindi di un lavoro estremamente onerosa che in particolare richiede che $\overline{\boldsymbol{P}}$ disponga di tutte le descrizioni delle procedure in $\{ i = 1, 2, ..., n \ :| \ Pcl_i \}$ e delle configurazioni che esse hanno raggiunte, prestazione che richiede una grandissima quantità di memoria; questa esigenza di memoria ad ogni nuova fase cresce vistosamente, ma in linea di principio continua a rimanere finita e sostenibile da una strumentazione che, almeno in linea di principio, è manutenibile.

Ogni manovra primaria della $\overline{P}$ consiste nella generazione delle coppie $\langle n, i \rangle$ nell'ordine diagonale.
Generata una $\langle n, 0 \rangle$, $\overline{\boldsymbol{P}}$ si preoccupa di individuare la descrizione della procedura $\mathcal{P}_n$ portando avanti lo scorrimento ordinato del monoide libero al quale appartengono tutte le descrizioni delle procedure che hanno T come alfabeto di uscita; quindi simula il suo passo 0 e si preoccupa di memorizzare procedura e configurazione raggiunta in modo da poterle riprendere.

Generata una $\langle n, i \rangle$ con $i > 0$ la $\overline{\boldsymbol{P}}$ recupera la descrizione della $\mathcal{P}_n$ e la sua configurazione raggiunta al passo $\langle n, i-1 \rangle$, simula la esecuzione del passo successivo e memorizza opportunamente la nuova configurazione raggiunta.
Inoltre nel passo nel quale la $\mathcal{P}_n$ genera una stringa w stabilisce il numero d'ordine di questa stringa, cioè individua lo j tale che $w = w_j$; nel caso sia $j = n$ la $\overline{\boldsymbol{P}}$ effettua anche l'emissione sul suo nastro di uscita della stringa w_n facendola riconoscere come stringa generata da essa stessa.
Dunque la procedura $\overline{\boldsymbol{P}}$ genera esattamente $\overline{L}$.

Evidentemente la procedura individuata a grandi linee deve compiere una ampia gamma di operazioni in quanto deve riuscire a simulare tutte le procedure generative, deve essere in grado di memorizzare una quantità di informazioni articolate e deve compiere una enorme massa di operazioni per riuscire a effettuare ciascuna delle generazioni di una sue stringa.

Occorre anche aggiungere che la gamma delle operazioni svolte dalla $\overline{\boldsymbol{P}}$ è finita e quindi è possibile effettuarla utilizzando un insieme finito di segni ausiliari che consentano di separare o delimitare le registrazioni delle strutture di dati esaminate o aggiunte e che consentano di esprimere le manovre richieste.

Le esigenze della $\overline{\boldsymbol{P}}$ comporterebbero risorse di memoria, tempi esecutivi e complessità di programmazionr insostenibili per una procedura con scopi pratici, non inficiano la validità della dimostrazione di un enunciato generale negativo che si sta conducendo.

Per concludere si tratta di mostrare che il complementare del linguaggio ricorsivamente enumerabile $\overline{L}$ non è ricorsivamente enumerabile dimostrando che è impossibile trovare una procedura generativa in grado di produrre le sue stringhe.
Questo fatto, prevedibilmente, viene dimostrato per assurdo, assumendo che esista una procedura $\mathcal{P}_k$ in grado di generale $\overline{L}^{\complement}$ e derivando una situazione costruttivamente improponibile.
La situazione insostenibile riguarda una stringa w_k che dovrebbe potersi associare alla ipotetica $\mathcal{P}_k$. Essa per la definizione di $\overline{\boldsymbol{L}}$ dovrebbe appartenere a questo linguaggio; ma dovrebbe anche essere generata da $\mathcal{P}_k$ e quindi appartenere a $\overline{L}^{\complement}$; quindi la ipotetica $\mathcal{P}_k$ non si può individuare ∎

C20 c.16 Teorema LngRcr ⊂ LngREn .

Dim.: Il linguaggio ricorsivamente enumerabile $\overline{L}$ non puàppartenere a **LngRcr**, in quanto questa collezione di linguaggi, in forza di c13, è invariante per complementazione ∎

C20 c.17 Osserviamo che la precedente inclusione stretta corrisponde ad affermare che la portata degli algoritmi, procedure per le quali è garantita la possibilità di concludere ogni loro evoluzione in un numero finito di passi è sostanzialmente inferiore rispetto a quella delle procedure, strumenti ai quali viene consentito di operare illimitatamente anche quando operano con fini di accettazione (non solo se finalizzati alla generazione di successioni estese quanto si vuole).
Le procedure nonalgoritmiche sono entità definibili con modalità accettate sul piano matematico (in realtà taluni le contestano) le quali in alcune circostanze, cioè se "stimolate" da talune stringhe di input, concludono la loro evoluzione in tempi finiti e possono fornire risultati utili, in particolare prendere decisioni di appartenenza o di non appartenenza.
Taluni di questi risultati conclusivi di portata limitata e che potrebbe essere poco prevedibile sono giudicati utili anche in quanto potrebbero essere ottenuti anche con procedure algoritmiche, probabilmente non facili da definire e probabilmente richiedendo un dispendio di risorse che opportuni studi secifici potrebbero ridurre significativamente.

Il risultato precedente può vedersi anche nel seguente modo. La matematica consente di definire meccanismi talora utili la cui portata tuttavia può essere difficilmente controllabile o forse incontrollabile e in alcuni momenti storici non controllata.
Si tratta quindi di un risultato che pone dei limiti alla matematica e a varie sue tradizionali presunte certezze.
In questo senso la inclusione stretta enunciata in c16 si colloca accanto ad altri risultati limitativi della matematica, come i teoremi di incompletezza di Goedel.

Osserviamo anche che le costruzioni che hanno consentito di arrivare agli enunciati di questo capitolo sono tendenzilmente vaghe, in quanto fanno riferimento a strutture e manovre individuate da definizioni programmatiche e consensuali, non su definizioni meticolosamente costruttive.

Per avere una situazione più solida è necessario riferirsi a modelli di algoritmi e procedure meglio definiti: questo faremo riferendoci prevalentemente alle macchine di Turing nel capitolo che segue.

Testo fruibile in https://www.mi.imati.cnr.it/alberto/ e https://arm.mi.imati.cnr.it/Matexp/matexp_main.php