TECNICO SUPERIORE WEB DEVELOPER FULL STACK

Bonus - P vs NP

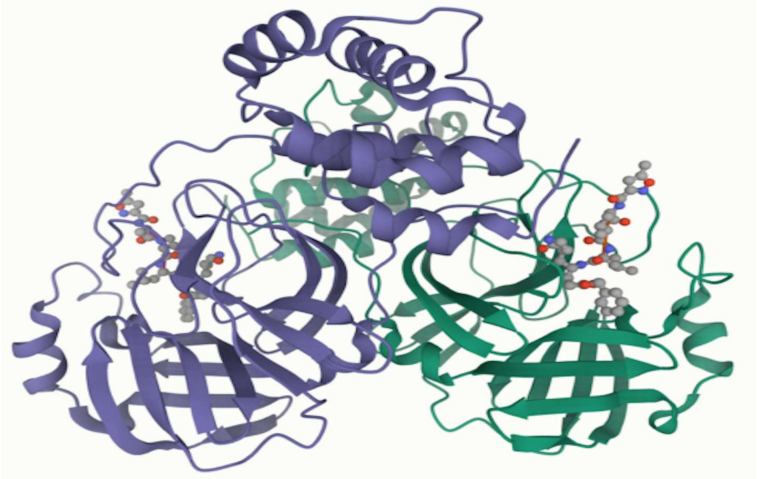
ITS DIGITAL ACADEMY
MARIO VOLPATO

niuko
innovation and knowledge

Protein folding

Il *protein folding* è il processo di ripiegamento molecolare attraverso il quale le proteine ottengono la loro struttura tridimensionale.

Esso è fondamentale per consentire il funzionamento delle proteine stesse, che si “incastrano” tra loro.



Sudoku

Data una griglia 9x9, riempire tutte le caselle con valori da 1 a 9 tali che non vi siano ripetizioni in righe, colonne o quadrati 3x3.

1			2				4	
	2				3	9		
9		7				5		
		4					5	7
			5	4	1			
3	5					1		
		3				7		9
		1	4				8	
	9				2			6

Sudoku - generale

Dato un valore N e una griglia N^2xN^2, riempire tutte le caselle con valori da 1 a N^2 tali che non vi siano ripetizioni in righe, colonne o quadrati NxN.

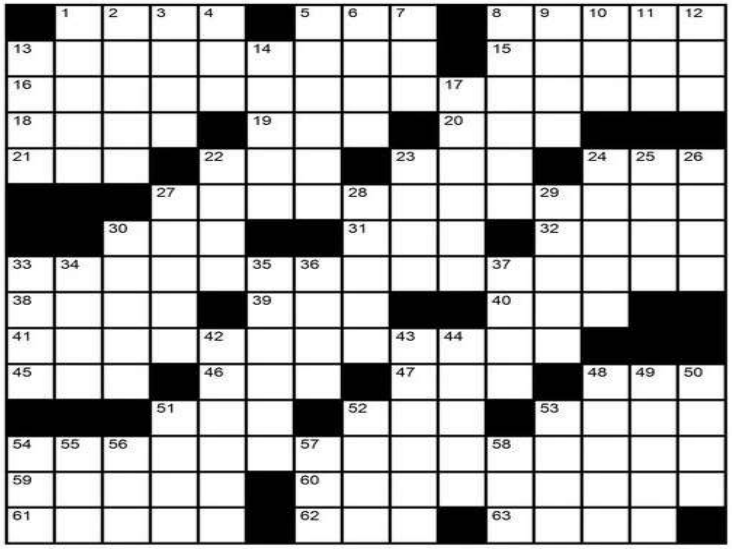
	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16
01	D	9	2						0				4	3		
02	4	C	F						9	D		2		6	E	B
03		E		1					F		8	B	C			A
04	0	5	6						A	3	C					1
05		7	0	5	A	8					6	1				
06	2				C					B	7					
07			A	E		9	6	4			0					
08		F	9		5		7		4							
09						2	9					A		C		
10									6	4	5	D		1		
11					D					1		C		B	7	F
12						E		C	2	0				8	A	9
13	B			9		0	5	2						E	F	6
14		2	5	6			F						D		B	
15	E	1	3		6		D									7
16	A				8	B	C	3					9	5	4	

ITS DIGITAL ACADEMY
MARIO VOLPATO

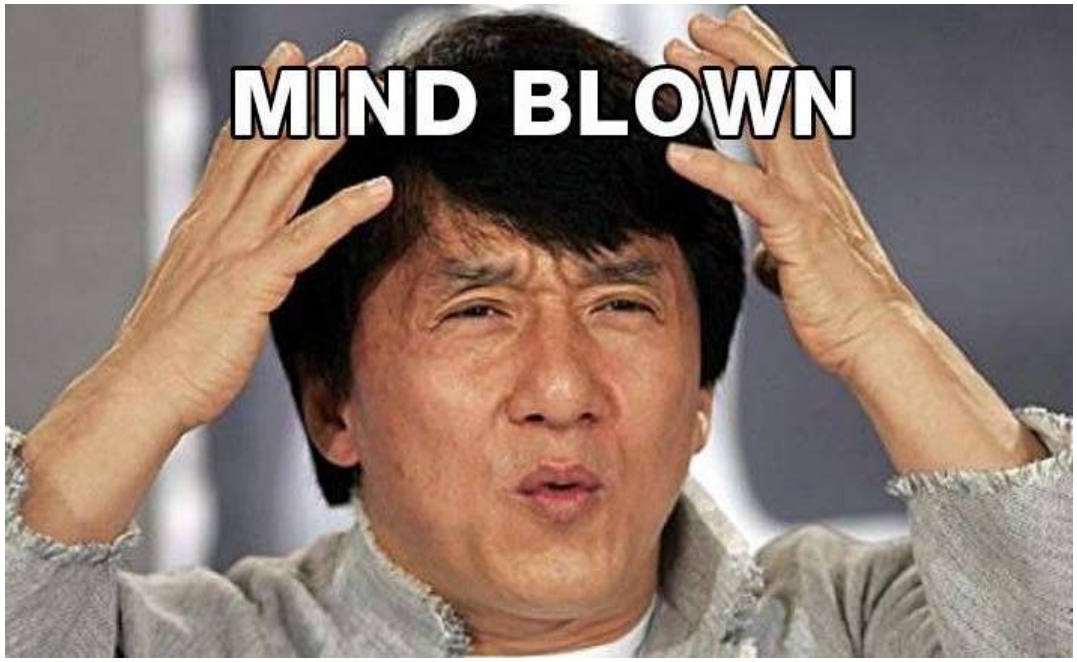
niuk
innovation and knowledge

Cruciverba

Data una griglia e un insieme di definizioni, inserire le parole definite all'interno, riempiendo tutte le caselle bianche e rispettando gli incroci di lettere.



Questi tre problemi sono lo stesso problema.



Millennium Prize Problems

Nel 2000, il Clay Institute ha promosso una challenge chiamata “Millennium Prize Problems”, selezionando 7 problemi “difficili” e offrendo 1.000.000$ per chiunque ne risolva uno.

1. Birch and Swinnerton-Dyer conjecture
2. Navier–Stokes existence and smoothness
3. Yang–Mills existence and mass gap
4. Riemann hypothesis
5. Hodge conjecture
6. P vs NP
7. Poincarè Conjecture

Millennium Prize Problems

Molti di questi sono problemi estremamente specifici in certi campi della matematica o della logica, e di questi solo uno finora è stato risolto.

1. Birch and Swinnerton-Dyer conjecture
2. Navier–Stokes existence and smoothness
3. Yang–Mills existence and mass gap
4. Riemann hypothesis
5. Hodge conjecture
6. P vs NP
7. **Poincarè Conjecture**

Millennium Prize Problems

P vs NP è il più recente tra tutti i problemi della lista (1971), ed anche il più semplice da spiegare.

1. Birch and Swinnerton-Dyer conjecture
2. Navier–Stokes existence and smoothness
3. Yang–Mills existence and mass gap
4. Riemann hypothesis
5. Hodge conjecture
6. **P vs NP**
7. Poincarè Conjecture

Che cos'è P vs NP?

Storicamente, gli informatici hanno incontrato problemi complessi da risolvere con macchine lente.

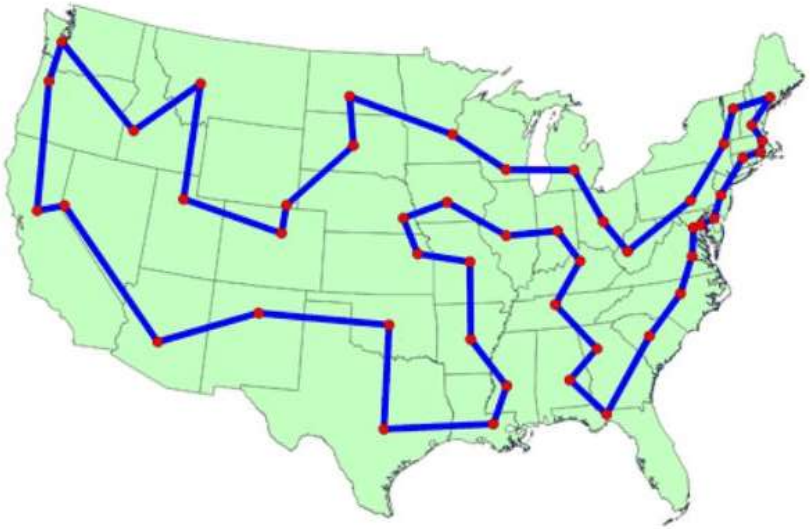
Si cercava quindi quanto più possibile di passare da algoritmi poco efficienti a versioni migliori, in modo da ottenere gli stessi risultati in meno tempo.

Su alcuni problemi questo è stato possibile (pensate a Dijkstra vs Johnson).

Su altri… è più complicato.

TSP, o il problema del commesso viaggiatore

Date `n` città, le distanze tra esse, ed un intero `k`, è possibile partire da una città, attraversare ogni città esattamente una volta tornando alla città di partenza, percorrendo una distanza non superiore a `k`?



TSP, o il problema del commesso viaggiatore

La soluzione più immediata è calcolare tutte le permutazioni per vedere quale sia la più economica (usando un metodo brute-force). Questo approccio, tuttavia, ha costo O(n!), e diventa quindi impraticabile già con 20 città.

Pur non essendo stato trovato un algoritmo che funzioni in tempo efficiente, per qualche caso particolare si è trovata una soluzione (un tour delle città della Svezia, ad esempio).

Perché vi sto annoiando con questo?

Perché il problema del commesso viaggiatore è un esempio di problema che si trova in realtà anche in altre situazioni:

- Problemi di pianificazione e logistica: trasporto merci, organizzazione di eventi...
- Problemi di manifattura di microchips: se devo saldare un certo numero di punti e conosco la distanza tra essi, posso fare una pista lunga al massimo k?
- Problemi di DNA sequencing: quanto sono simili (qual è la distanza) due sequenze di DNA? Posso "andare" da una sequenza ad un'altra entro una certa "distanza"?

Quindi, back to the ‘70s...

Ci sono problemi difficili per i quali abbiamo un algoritmo che li risolve “velocemente” (i.e. in modo efficiente), ma ci sono problemi molto difficili per i quali sappiamo che non esiste una soluzione efficiente.

E i problemi nel mezzo?

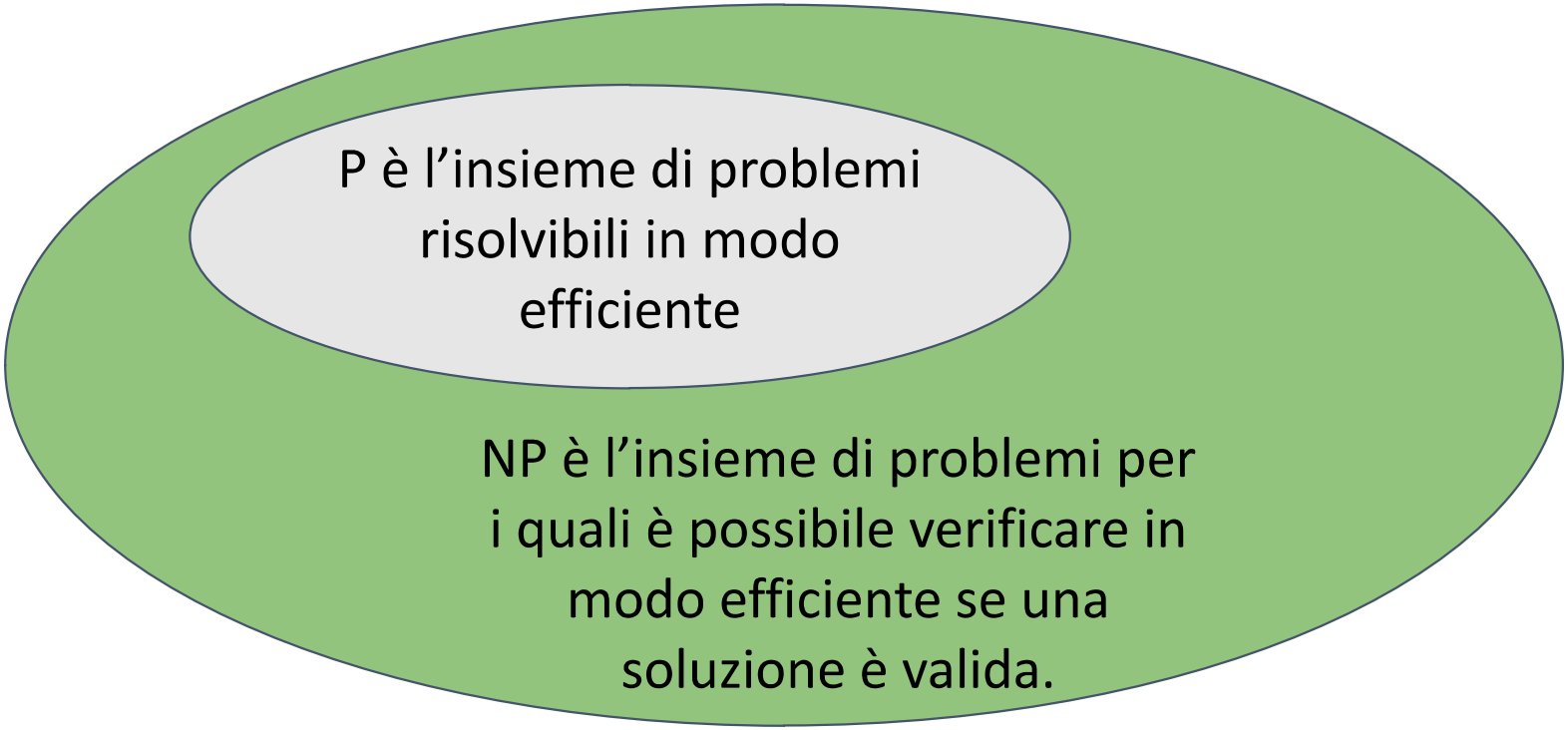
Visivamente

Scacchi	TSP (commesso viaggiatore)	Moltiplicazione Ordinamento
Problemi difficili (lenti)	???	Problemi semplici (veloci)

Introducing P

P è l'insieme di problemi risolvibili in modo efficiente

Introducing NP



Chi sta dove?

P è l'insieme di problemi risolvibili in modo efficiente, come:

- Moltiplicazione
- Minimo
- Ordinamento

NP è l'insieme di problemi per i quali è possibile verificare in modo efficiente se una soluzione è valida, come:

- TSP
- Design di circuiti (SAT)

Da NP a P

Talvolta siamo fortunati, e qualcuno scopre un algoritmo efficiente per risolvere un problema che pensavamo fosse NP.

In quel caso, tutti i problemi equivalenti a quello risolto vengono inseriti nell'insieme P.

La domanda è: tutti i problemi in NP sono anche in P, solo che non sappiamo ancora come risolverli?

Ovvero: **P = NP**?

La domanda quindi è:

Essere in grado di **riconoscere** una soluzione corretta significa che c'è sicuramente un modo "veloce" per **trovarla**?

Perché è importante?

Ci sono un sacco di problemi attualmente in NP che, se si scoprisse un algoritmo per risolverli in modo efficiente, avrebbero potenti ripercussioni sulla vita quotidiana:

- Protein folding
- Analisi dei mercati
- Crittografia a chiave pubblica e privata

Ok, ma perché P e NP?

P sta per "*Polynomial time*", ovvero i problemi in **P** sono quelli risolvibili in tempo polinomiale nella dimensione dell'input.

Non importa se parliamo di N^2, $3N^5$, N^{17}, questi casi, asintoticamente, ad un certo punto saranno migliori di qualsiasi funzione esponenziale 2^N.

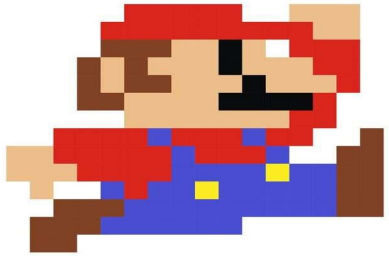
NP sta per "*Non-deterministic polynomial time*", ossia il tempo di verifica di una soluzione è polinomiale.

E' un po' come dire che, se avessimo a disposizione un quantitativo sufficiente di computer, ognuno dedicato a verificare una soluzione, ci metterebbero un tempo polinomiale.

Sempre negli anni ‘70

Si è scoperto che molti dei problemi considerati “molto difficili” (hard) che avevano notevoli applicazioni in vari campi delle scienze erano in realtà lo stesso problema, con minime variazioni.

Problemi come *isomorfismo tra grafi, cammini massimi, fattorizzazione, 3-SAT, Graph coloring, Partition, Job sequencing*, sono tutti problemi cosiddetti **NP-completi**.

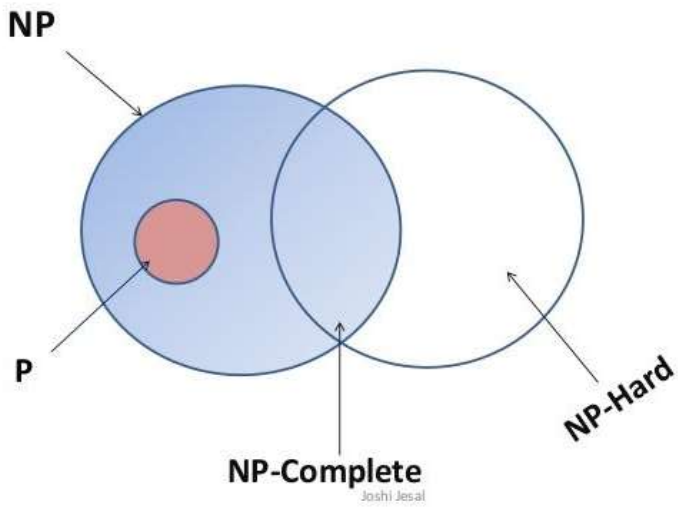


Tra questi, possiamo anche inserire il Sudoku, le parole crociate, i problemi di folding delle proteine, ma anche Battaglia Navale, Super Mario e Metroid.

Schematicamente...

Tutti i problemi in **P** sono in **NP**, e i problemi molto difficili in **NP** sono chiamati **NP-completi**.

Se si riuscisse a trovare un algoritmo polinomiale per risolvere un problema **NP-completo**, tutti i problemi in **NP** sarebbero risolvibili in tempo polinomiale, e quindi **P = NP**.



Perché è così difficile?

Perché “dimostrare” le cose è un problema che sta in **NP**.

Perché è così difficile?

Perché “dimostrare” le cose è un problema che sta in **NP**.

Quindi potrebbe essere difficile.

Perché è così difficile?

Perché “dimostrare” le cose è un problema che sta in **NP**.

Quindi potrebbe essere difficile.

O no.

Perché è così difficile?

Perché “dimostrare” le cose è un problema che sta in **NP**.

Quindi potrebbe essere difficile.

O no.

Non lo sappiamo, perché non sappiamo se **P = NP**.

Ci sono solo P e NP?

No, ovviamente.

Nel corso dei decenni sono state scoperte molte cosiddette "classi di complessità", tra cui:

- **EXP** (i problemi risolvibili in tempo esponenziale)
- **PSPACE** (i problemi risolvibili avendo a disposizione tempo infinito ma spazio polinomiale)
- **BQP** (i problemi risolvibili in tempo polinomiale da un computer quantistico mantenendo una certa soglia probabilistica di errore).

E poi ci sono i problemi non decidibili

Un problema non decidibile è un problema per il quale è stato dimostrato che è impossibile generare un algoritmo in grado di produrre una risposta corretta.

Il problema della terminazione (halting problem) ne è un esempio: è stato dimostrato che non esiste un algoritmo in grado di determinare correttamente se un programma terminerà oppure no.

Quindi?

Questo enorme mare di classi di problemi e di complessità ha implicazioni in fisica, in biologia, in matematica, in astrofisica...

La discussione sul rapporto tra **P** e **NP,** in realtà, è una discussione sullo spazio e il tempo, e sulla natura stessa delle decisioni e della risoluzione di problemi.

Insomma, è una discussione sulla vita.

In conclusione

Se **P = NP**, allora il mondo sarebbe un posto profondamente diverso.

Non ci sarebbe nessun valore nel “salto creativo”.[...]

Chiunque apprezzi una sinfonia sarebbe Mozart, e chiunque sia in grado di seguire un ragionamento sarebbe Gauss.

- Scott Aaronson, MIT -

In conclusione

Mi rallegro. Hai fatto il tuo primo passo in un mondo più vasto.

La Forza sarà con te, sempre.

- Obi Wan Kenobi, Star Wars -