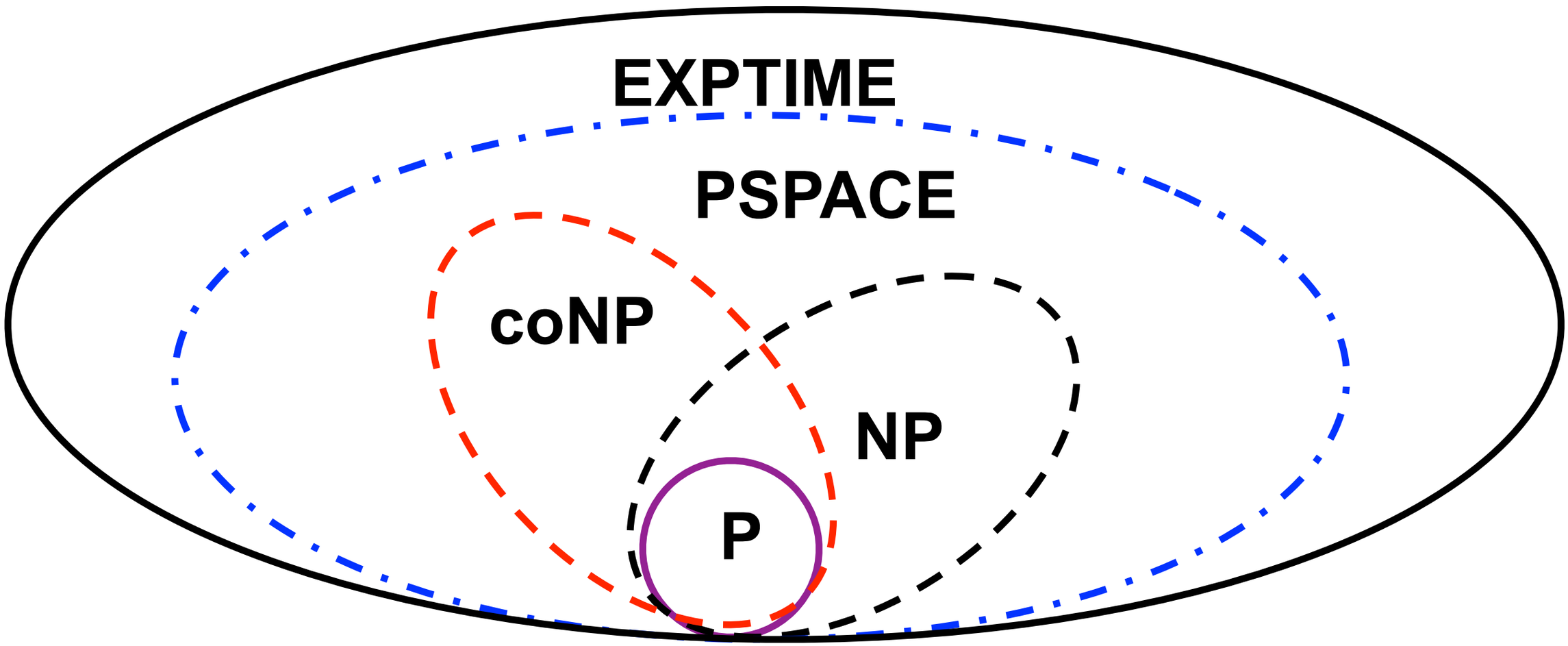
Dentro P e oltre NP ?

Tra P e EXPTIME?



Si sa solo che P ≠ EXPTIME

PSPACE completezza

Un linguaggio A è PSPACE completo se

1. **A è in PSPACE, cioè esiste una TM T che accetta A con complessità di spazio polinomiale.**
2. **ogni linguaggio in PSPACE si riduce in tempo polinomiale ad A**

PSPACE completezza

Va sottolineato che si usa la riduzione polinomiale in tempo, anzichè una riduzione polinomiale in spazio.
Perchè?

Se un problema A si riduce (polinomialmente) a un problema B allora possiamo dire che

la complessità di A ≤ la complessità della riduzione + la complessità di B.

PSPACE COMPLETE

Un problema in PSPACE cui si riducono polinomialmente tutti i problemi in PSPACE è detto completo per PSPACE.

Come nel caso di NP-completo sono i problemi più difficili in PSPACE.

PSPACE-completo: esempi

1. Data un'espressione regolare r su un alfabeto Σ il problema di verificare se denota Σ*.

2. Il problema della soddisfacibilità per formule booleane pienamente quantificate.

Esempio:

- **$\forall x \exists y[(x \vee y) \wedge (\neg x \vee (\neg y)]$**

vera perchè per x = 1 si pone y=0 e per x=0, y=1

E' la classe dei giochi.

Formule Booleane Quantificate

Le formule booleane quantificate sono le formule booleane dotate di quantificatori esistenziali e universali:

- $\forall x[x \vee y]$
- $\exists x \exists y[x \vee y]$
- $\forall x[x \vee x]$
- $\forall x \exists y[(x \vee y) \wedge (\neg x \vee (\neg y)]$
- $\exists x \forall y[(x \vee y) \wedge (\neg x \vee (\neg y)]$

vera perchè per x = 1 si pone y=0 e per x=0, y=1

falsa perchè per x = 1 se y=0 va bene ma per y=1 no! E analogamente se x=0

Una formula booleana pienamente quantificata è un formula dove ogni variabile è quantificata.
Una formula booleana pienamente quantificata è sempre vera o sempre falsa

Una formula booleana quantificata è in forma normale prenessa se tutti i quantificatori appaiono all'inizio della formula.

Problema:

TQBF = {<φ> | φ è una formula pienamente quantificata vera}

TQBF è in PSPACE

Definizione ricorsiva di un decisore T per TQBF.
T=
Input φ
If φ non è quantificata, contiene solo costanti
then valutala e se il risultato è 1 accetta, altrimenti rifiuta
If φ = ∀xψ , chiama ricorsivamente T su ψ prima con x = 0
e poi con x = 1.
Accetta se entrambe le chiamate accettano.
If φ = ∃ψ , chiama ricorsivamente T su ψ prima con x = 0
e poi con x = 1.
Accetta se una delle due chiamate accetta.

Per ogni chiamata lo spazio necessario è quello per il valore attribuito a una variabile, la profondità dello stack delle chiamate è il numero delle variabili, m.
Quindi $s_T(n)$ è in O(n), dove n è la lunghezza della formula e n≥m.

TQBF è PSPACE- completo

Si deve costruire una riduzione polinomiale per ogni linguaggio A in PSPACE, $A \leq_p TQBF$.
Detta M una TM polinomiale in spazio tale che L(M)=A e w un input per M, si associa a w, in tempo polinomiale, una formula booleana pienamente quantificata φ per la quale vale che

w è in A sse φ è in TQBF

L'idea della prova è simile a quella di Cook-Levin per SAT, ma usa l'idea di Savitch nella costruzione della formula, per mantenere la complessità di spazio polinomiale.

Ricordiamo che $t(n) \leq 2^{ks(n)}$ e quindi la formula "alla Cook-Levin" avrebbe lunghezza esponenziale.

GIOCHI

Qui un gioco è una competizione tra due contendenti che, seguendo delle regole date, devono raggiungere un determinato obiettivo.

I giochi sono strettamente legati alle formule booleane pienamente quantificate: a ogni formula possiamo associare un gioco e spesso a un gioco possiamo associare una formula.

Già il problema della soddisfacibilità per formule booleane pienamente quantificate può essere visto come un gioco.

Il gioco delle formule

Sia $\varphi = \exists x_1 \forall x_2 \exists x_3 \forall x_4 \ldots Q x_k \psi$ dove Q può essere $\exists$ o $\forall$

Siano All e Ex i contendenti che devono scegliere i valori di verità per le variabili, All per quelle quantificate universalmente ed Ex per quelle quantificate esistenzialmente.

L'ordine della scelta è determinato dall'ordine dei quantificatori all'inizio della formula, al temine delle scelte si valuta la formula e se la formula risulta vera vince Ex, altrimenti vince All.

Per questo PSPACE è spesso chiamata la classe dei giochi

Esempio del gioco delle formule

Sia $\varphi = \exists x_1 \forall x_2 \exists x_3 \psi$ dove

$\psi = [(x_1 \vee x_2) \wedge (x_2 \vee x_3) \wedge (\neg x_2 \vee \neg x_3)]$

Ex sceglie per esempio 1 per x_1, allora All sceglie 0 per x_2 e allora Ex prende 1 per x_3 e vince.

In realtà Ex può vincere sempre prendendo 1 per x_1 e poi l'opposto del valore che All attribuisce a x_2 per x_3 , cioè Ex ha una strategia vincente. Se prendiamo un altro esempio prendendo

$\psi = [(x_1 \vee x_2) \wedge (x_2 \vee x_3) \wedge (x_2 \vee \neg x_3)]$

ora è All ad avere una strategia vincente perchè se prende 0 per x_2 qualunque sia il valore attribuito a x_3 una delle due clausole è falsa.

Il gioco delle formule

Problema:

GIOCO-FORMULE= {<φ> | il giocatore Ex ha una strategia vincente per φ }

GIOCO-FORMULE è PSPACE-completo

GIOCO-FORMULE non è altro che un diverso modo di vedere TQBF = {<φ> | φ è una formula pienamente quantificata vera}

In un GIOCO un giocatore ha una strategia vincente se vince qualunque siano le mosse dell'avversario

Dentro P?

Classi di complessità di spazio logaritmico

TM modificate

Per poter considerare limiti di spazio sublineari dobbiamo dividere lo spazio necessario per l'input dallo spazio di memoria aggiuntivo.

Consideriamo TM con due nastri:

1) **il nastro di input di sola lettura**
2) **il nastro di lavoro di lettura e scrittura**

Prendiamo in considerazione solo il nastro di lavoro quando consideriamo la complessità di spazio della TM

Relazione tra spazio e tempo

Abbiamo visto che se una TM ha complessità di spazio s(n) allora possiamo limitare superiormente la sua complessità di tempo:

$$t(n) \leq 2^{O(s(n))}$$

Vale ancora questa relazione?

Potremmo avere una TM che lavora in spazio costante ma in tempo lineare.

Relazione tra spazio e tempo

Dobbiamo modificare la definizione di configurazione.

Se T è una TM con un nastro di input di sola lettura una configurazione di T deve dar conto del contenuto del nastro di lavoro, dello stato e delle posizioni delle due testine. Sia d il numero dei simboli di nastro.
Come nel caso standard il numero dei contenuti di nastro, $d^{s(n)}$, moltiplicato per quello delle diverse posizioni che può assumere la testina di lettura/ scrittura, s(n), moltiplicato per quello degli stati, q, contribuisce a costituire il nostro limite superiore:

$$s(n)^* \; q \; * \; d^{s(n)}$$

Relazione tra spazio e tempo

Tenendo conto che la testina di lettura del nastro di input può trovarsi su una cella qualunque, tra gli n simboli in input, otteniamo

$$n * s(n) * q * d^{s(n)}$$

In conclusione $t_T(n) \leq n2^{O(s(n))}$

Se $s(n) \geq \log n$ **allora**

$$n2^{O(s(n))} \leq 2^{\log(n)}2^{O(s(n))} \leq 2^{O(s(n))}$$

Teorema di Savitch per spazi sublineri

Abbiamo dimostrato il teorema per una NTM T con $s(n) \geq n$

Se $s(n) \geq \log n$ possiamo seguire gli stessi passi della prova, infatti abbiamo visto che $t_T(n) \leq 2^{O(s(n))}$, inoltre ogni configurazione (contenuto del nastro, stato e posizione delle due testine) occupa uno spazio

$$O(s(n) + \log n) = O(s(n))$$

se $s(n) \geq \log n$

Classi di complessità di spazio logaritmico

L è la classe dei linguaggi che sono decidibili in spazio logaritmico da una TM :

$$L = SPACE(\log n)$$

NL è la classe dei linguaggi che sono decidibili in spazio logaritmico da una NTM :

$$NL = NSPACE(\log n)$$

Esempio di linguaggio in L

Il linguaggio A = $\{0^k1^k \mid k \geq 0\}$ è in L

Basta contare il numero degli 0 e poi quello degli 1 e confrontare i due numeri.

Questi numeri in binario occupano spazio logaritmico in k.

Esempio di linguaggio in NL

Il linguaggio PATH = {<G,s,t> | G è un grafo <u>diretto</u> con un cammino da s a t} è in NL.

Non è noto se c'è un algoritmo deterministico che lavora in spazio logaritmico per PATH.

L'algoritmo polinomiale in tempo è lineare in spazio.

PATH è in P

Il linguaggio PATH = {<G,s,t> | G è un grafo diretto con un cammino da s a t} è in P.

M = input <G,s,t> G è un grafo diretto e s e t sono vertici di G

1. Marca il nodo s.

2. Ripeti fino a quando non ci sono più nodi da marcare:

esamina tutti gli archi di G.

Se c'è un arco (a,b) in cui a è marcato e b no, marca b

4. Se t è marcato, accetta. Altrimenti, rifiuta.

PATH è in NL

Il linguaggio PATH = {<G,s,t> | G è un grafo diretto con un cammino da s a t} è in P.

Un algoritmo non deterministico per PATH, che lavora in spazio logaritmico, procede scegliendo non deterministicamente il prossimo nodo tra tutti quelli adiacenti al nodo corrente, che è l'unico memorizzato sul nastro di lavoro.
Quindi lo spazio necessario è O(log |V|).

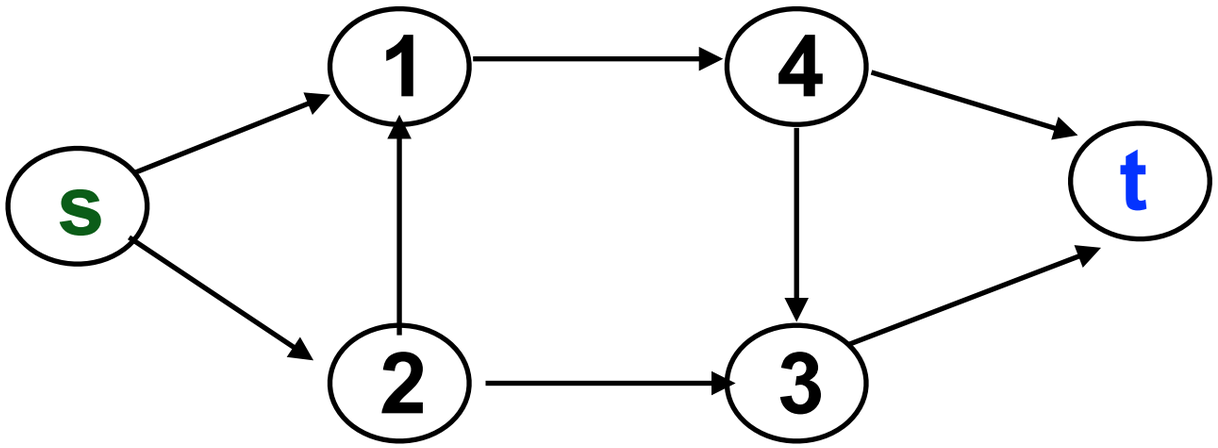
La NTM PATH

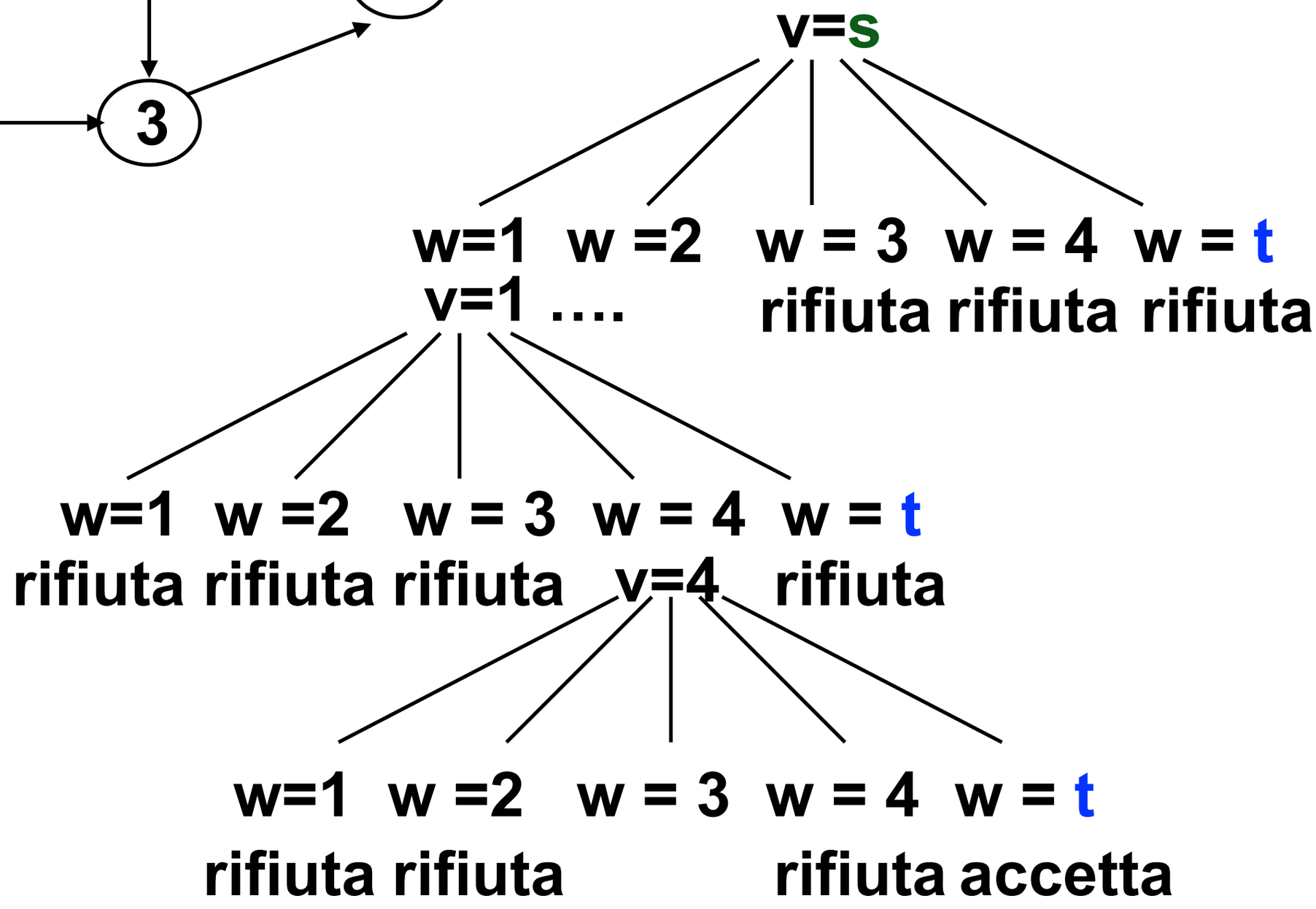
PATH=
input: <G=(V,E),s,t>
accetta se c'è un cammino da s a t, altrimenti rifiuta

```
if s = t return "sì"
v = s
for i = 1 to |V| do
      {scegli non deterministicamente un vertice w≠s in V
      if non c'è l' arco da v a w  return "no"
       altrimenti
            if w = t, return sì
       v = w}
 if i = |V|, return no
```

Lo spazio necessario è quello per i valori delle variabili v,w i cui valori sono al più |V| e quindi rappresentabili in log (|V|)

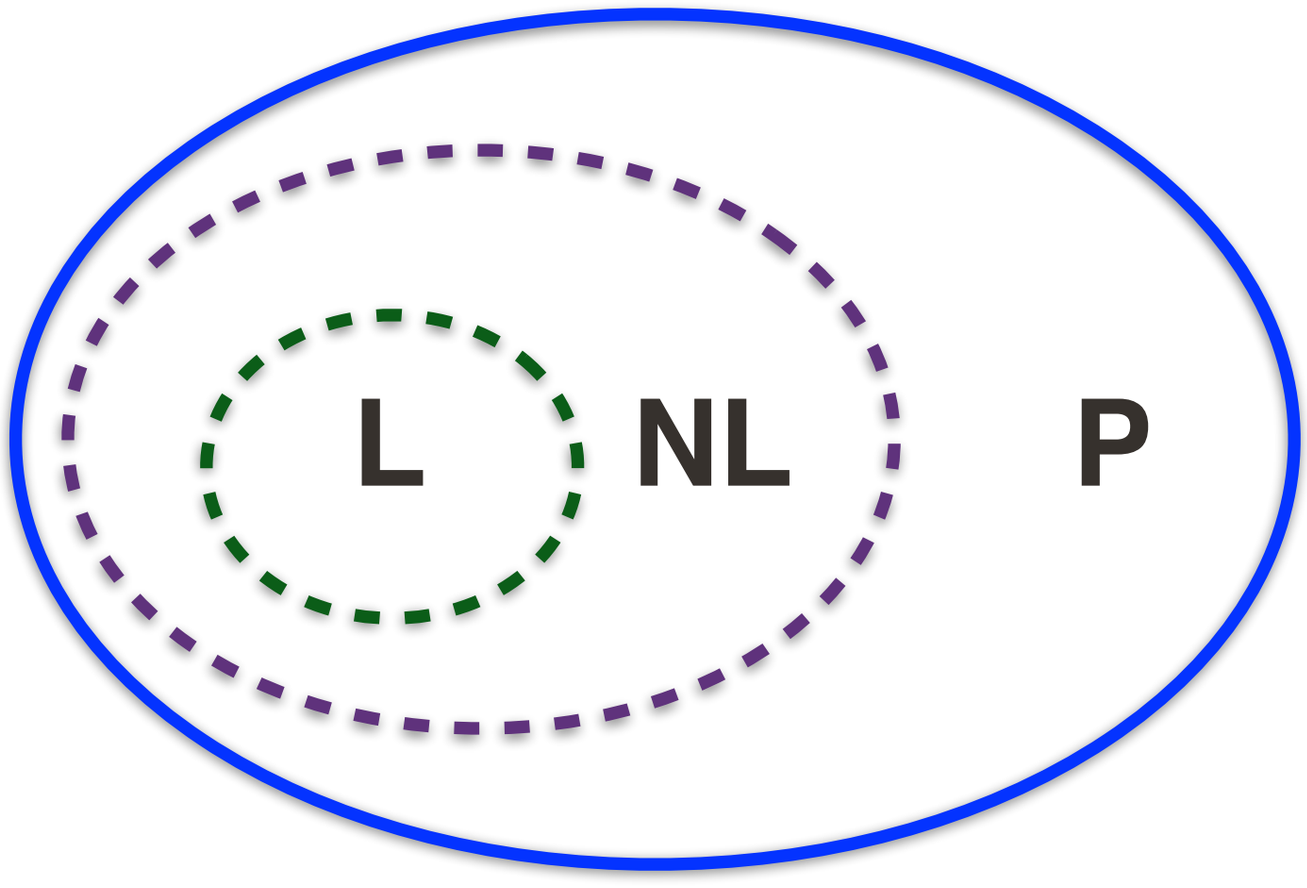
Esempio di esecuzione PATH



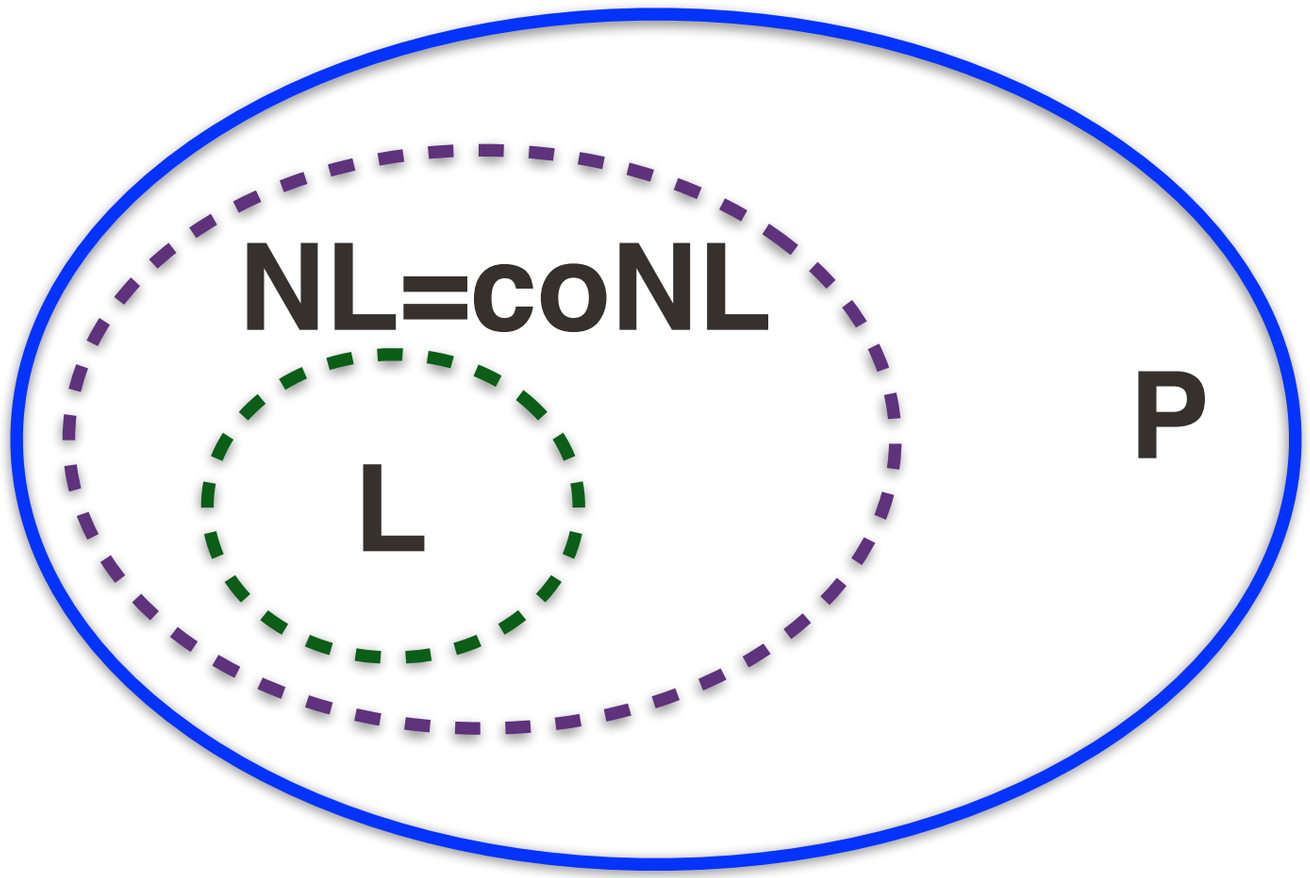


$L \subseteq NL \subseteq P$

La dimostrazione che $NL \subseteq P$ si basa sul fatto che ogni linguaggio in NL si riduce polinomialmente a PATH, che è in P.



NL = coNL



NSPACE e coNSPACE

L'affermazione NL = coNL è un caso particolare di

$$\text{NSPACE}(s(n)) = \text{coNSPACE}(s(n))$$

per ogni $s(n) \geq \log n$

Come Neil Immerman e Róbert Szelepcsényi hanno dimostrato indipendentemente nel 1987.

Cosa che ha loro fruttato il premio Gödel dell' ACM SIGACT (Association for Computing Machinery Special Interest Group on Algorithms and Computation Theory) e dell' EATCS (European Association for Theoretical Computer Science) nel 1995.

NL $\subsetneq$ PSPACE

Si dimostra usando il teorema di Savitch e un teorema che consente di separare le classi di complessità di spazio, il teorema della gerarchia di spazio, che ha come corollario che per ogni coppia di numeri reali $0 \leq a \leq b$

$$\mathbf{SPACE}(n^a) \subsetneq \mathbf{SPACE}(n^b).$$

Per Savitch

$$\mathbf{NL} = \mathbf{NSPACE}(\log n) \subseteq \mathbf{SPACE}(\log^2 n).$$

per il teorema della gerarchia di spazio

$$\mathbf{SPACE}(\log^2 n) \subsetneq \mathbf{SPACE}(n^2) \subsetneq \mathbf{PSPACE}$$

NL ⊊ PSPACE

Visto che L ⊆ NL ⊆ P

quindi NL ⊊ P **oppure P ⊊ PSPACE**