Dentro P e oltre NP ?

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

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

Il problema della soddisfacibilità per formule booleane 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.

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, 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 = $\{0^k 1^k \mid k \geq 0\}$ 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 <u>diretto</u> 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=

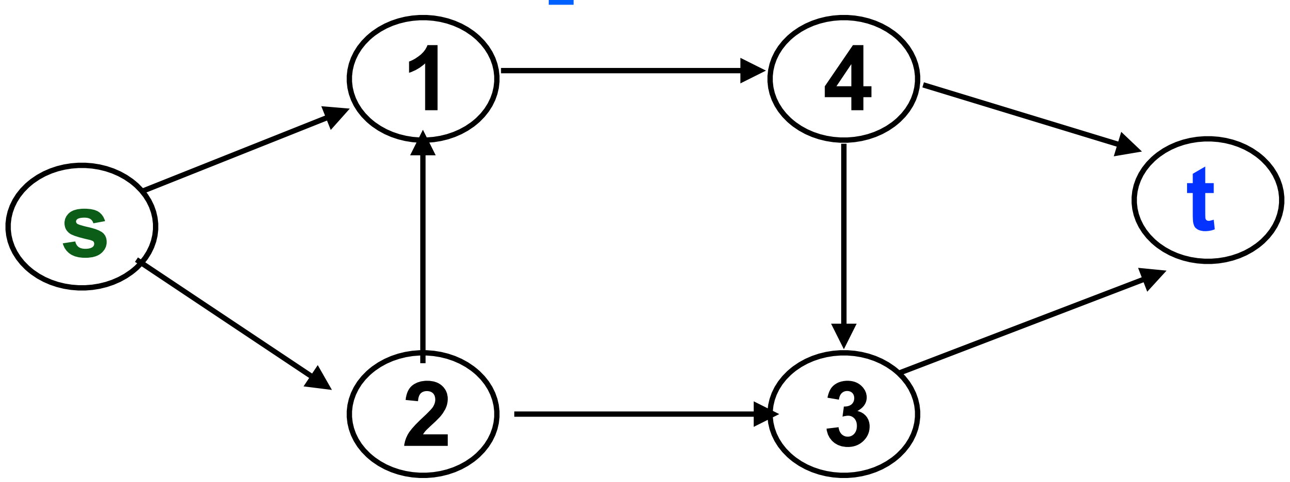
input: <G=(V,E),s,t>

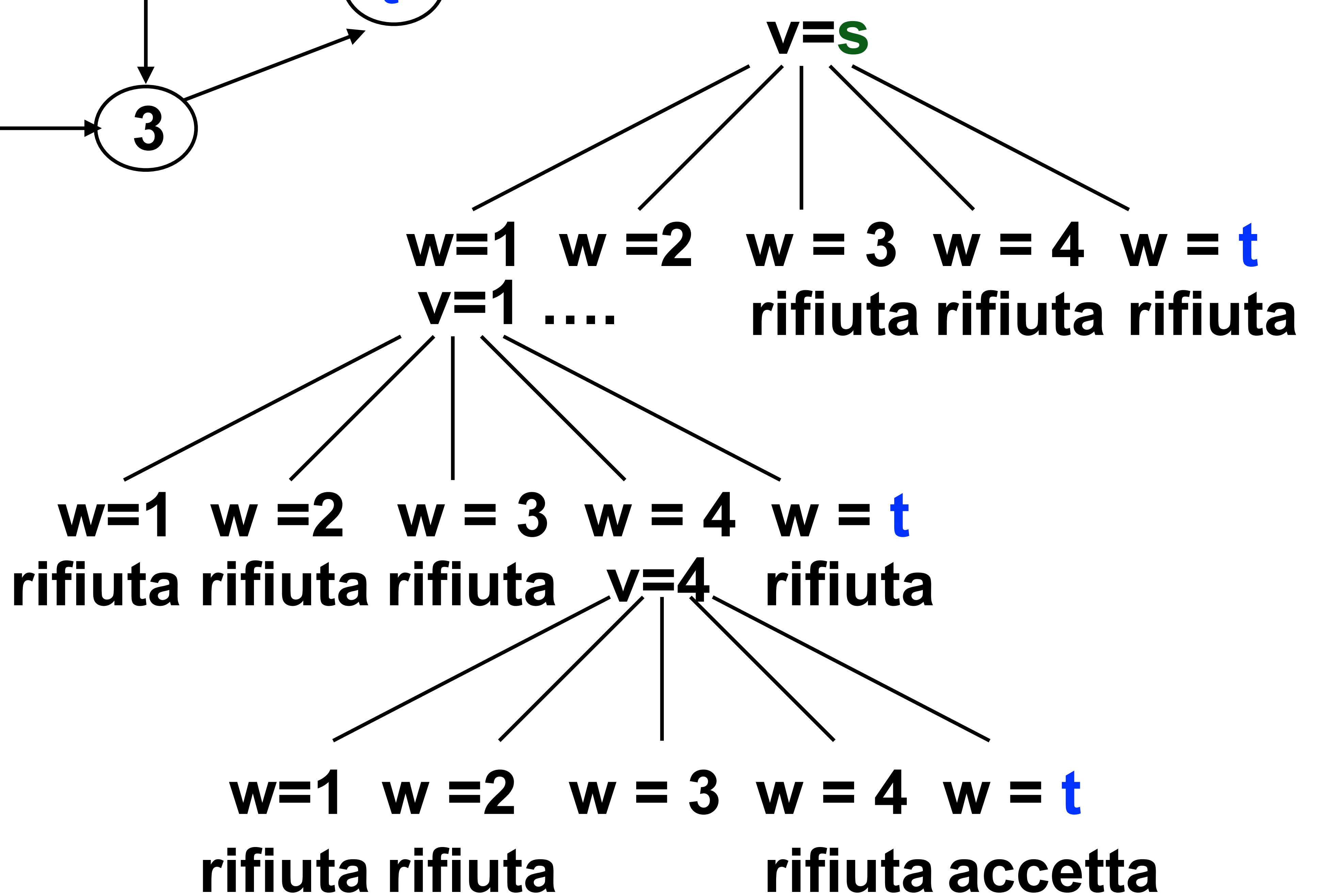
accetta se c'è un cammino da s a t, altrimenti rifiuta

```
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|)

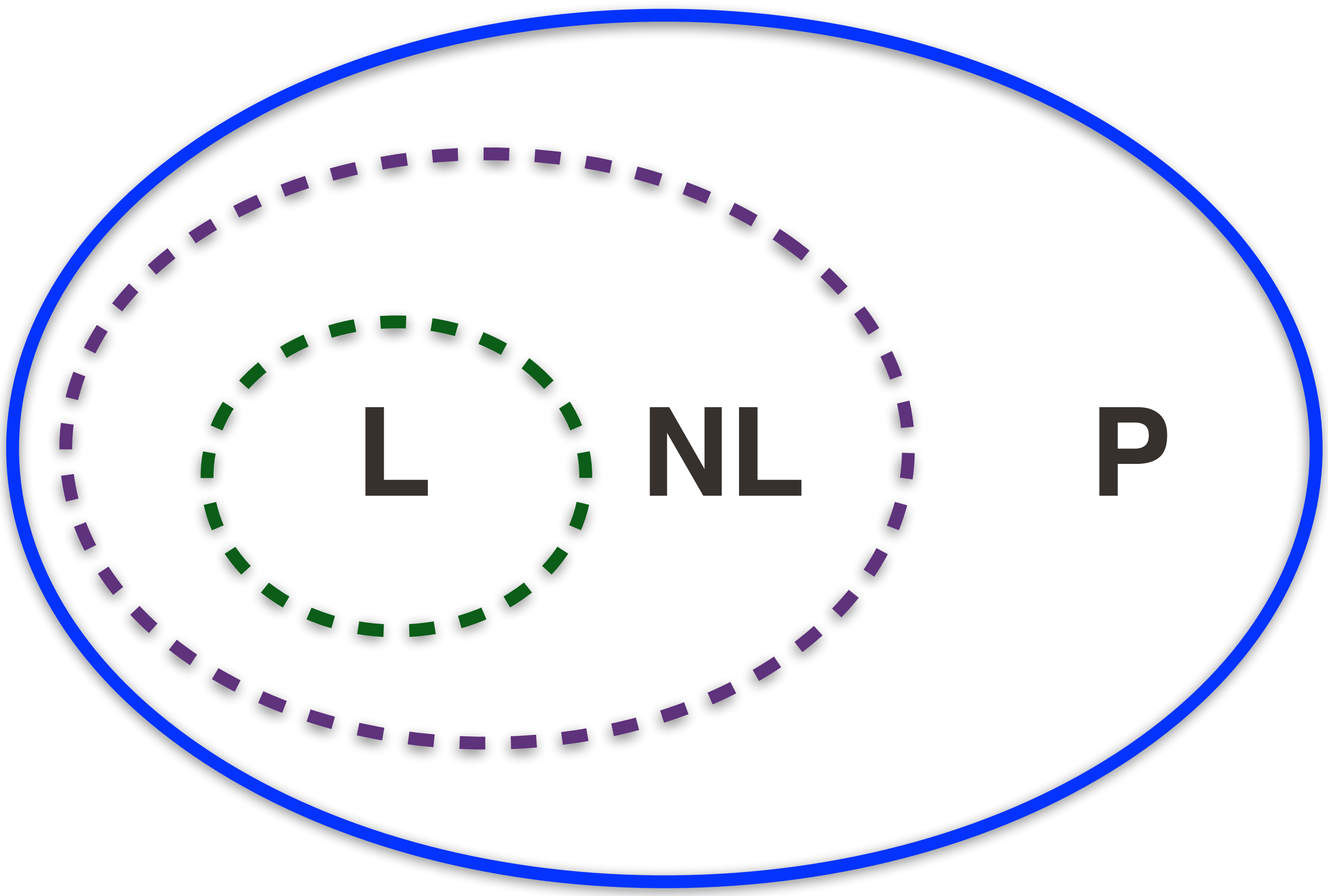
Esempio di esecuzione PATH



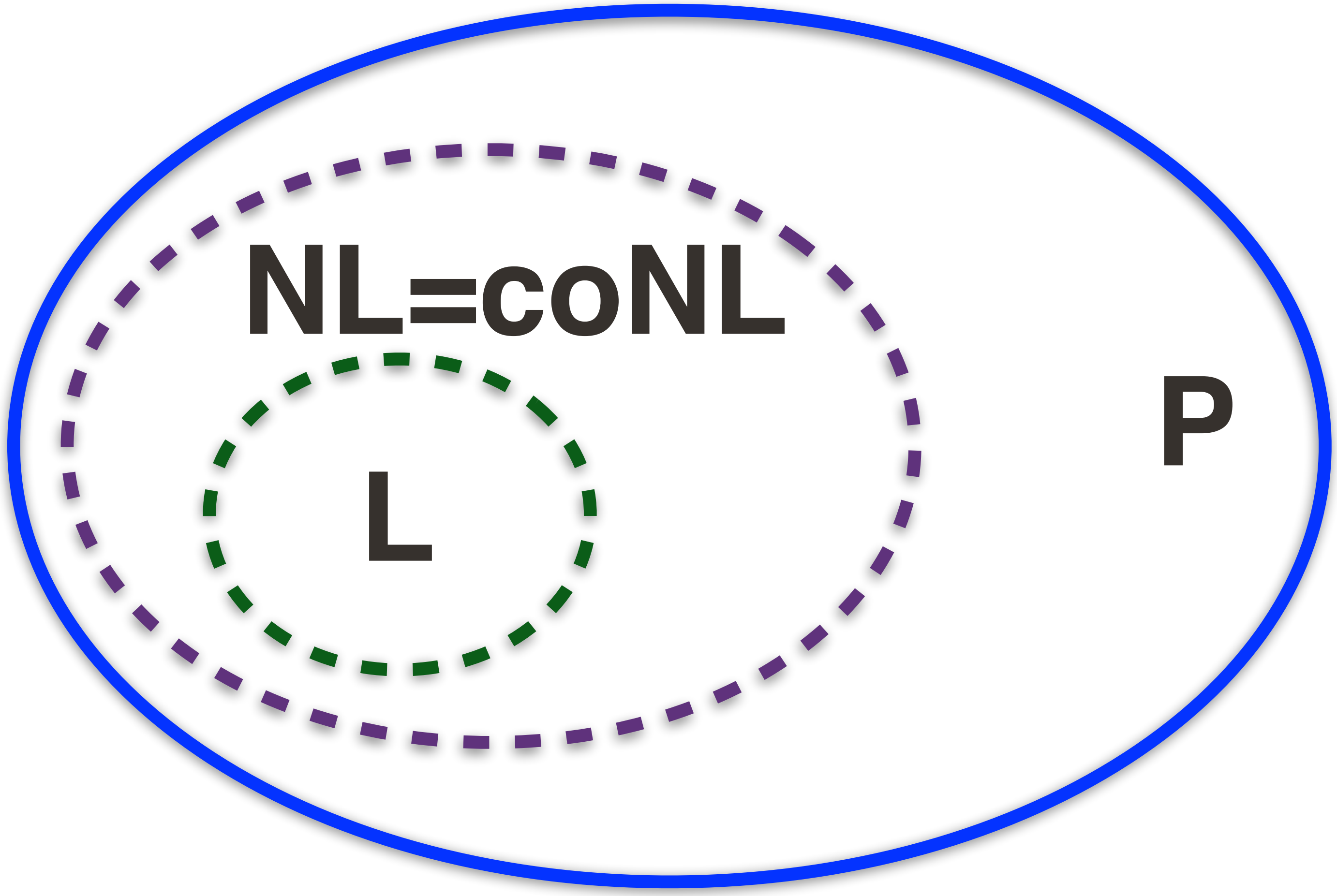


L ⊆ NL ⊆ P

La dimostrazione che NL ⊆ 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

NSPACE(s(n)) = coNSPACE(s(n))

per ogni s(n) ≥ 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 $0 \leq a \leq b$

$$SPACE(n^a) \subsetneq SPACE(n^b).$$

Per Savitch

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

per il teorema della gerarchia di spazio

$$SPACE(\log^2 n) \subsetneq SPACE(n) \subsetneq PSPACE$$

$NL \subsetneq PSPACE$

Visto che $L \subseteq NL \subseteq P$

quindi $NL \subsetneq P$ oppure $P \subsetneq PSPACE$