Se il confronto tra $\mathcal{P}$ ed $\mathcal{NP}$ sembra difficile, sorprende che lo stesso confronto tra $\mathcal{PS}$ ed $\mathcal{NPS}$ sia invece facile: queste due classi di linguaggi infatti coincidono. La dimostrazione consiste nel simulare una TM non deterministica con un limite polinomiale sullo spazio $p(n)$ mediante una TM deterministica con limite polinomiale sullo spazio $O\big(p^2(n)\big)$.

Il nocciolo della prova consiste nel verificare, in modo ricorsivo e deterministico, se una NTM N può passare dalla ID I alla ID J in non più di m mosse. Una DTM D tenta sistematicamente tutte le ID intermedie K per controllare se da I può passare a K in $m/2$ mosse e da K a J in altre $m/2$ mosse. Supponiamo cioè che esista una funzione ricorsiva $reach(I, J, m)$ che decide se $I \overset{*}{\vdash} J$ in m mosse al massimo.

Trattiamo il nastro di D come uno stack in cui sono collocati gli argomenti delle chiamate ricorsive di *reach*. Uno *stack frame*[4] D contiene $[I, J, m]$. La Figura 11.3 presenta uno schema dell'algoritmo eseguito da *reach*.

```
BOOLEAN FUNCTION reach(I,J,m)
    ID: I,J; INT: m;
    BEGIN
        IF (m == 1) THEN /* base */ BEGIN
            verifica se I == J o
                     se I può diventare J in una mossa;
            RETURN TRUE in caso positivo,
                   FALSE altrimenti;
        END;
        ELSE /* passo induttivo */ BEGIN
            FOR ogni ID K DO
                IF (reach(I,K,m/2) AND
                    reach(K,J,m/2)) THEN
                    RETURN TRUE;
            RETURN FALSE;
        END;
    END;
```

Figura 11.3 La funzione ricorsiva *reach* verifica se da una ID si può passare a un'altra entro un numero fissato di mosse.

È importante osservare che *reach*, sebbene chiami se stessa due volte, compie le chiamate in sequenza, e dunque in ogni istante solo una di esse è attiva. Così, se partiamo da uno stack frame $[I_1, J_1, m]$, a ogni istante c'è solo una chiamata $[I_2, J_2, m/2]$, una $[I_3, J_3, m/4]$, una $[I_4, J_4, m/8]$, e così via, finché a un certo punto il terzo argomento diventa 1 e *reach* può applicare il passo di base senza ulteriori chiamate ricorsive. *reach* verifica se $I = J$ oppure $I \vdash J$, restituendo TRUE se almeno una delle due è vera e FALSE in caso contrario. Dato un valore iniziale m del numero di mosse, la Figura 11.4 illustra lo stack della DTM D con tutte le possibili chiamate attive a *reach*.
Se può sembrare che siano possibili molte chiamate a *reach* e che il nastro della Figura 11.4 possa diventare molto lungo, dimostreremo che non potrà essere "troppo lungo". Se si parte da un numero di mosse pari a m, sul nastro ci possono essere a ogni istante solo $\log_2 m$ stack frame. Poiché il Teorema 11.4 garantisce che la NTM N non può fare più di $c^{p(n)}$ mosse, m può avere un valore iniziale non maggiore. Di conseguenza il numero di

$I_1\ J_1\ m$	$I_2\ J_2\ m/2$	$I_3\ J_3\ m/4$	$I_4\ J_4\ m/8$	...

Figura 11.4 Il nastro di una DTM che simula una NTM mediante chiamate ricorsive a *reach*.

stack frame è al più $\log_2 c^{p(n)}$, che è $O\big(p(n)\big)$. Disponiamo ora degli elementi essenziali su cui basare la dimostrazione del seguente teorema.