# 1 Complessità del calcolo

### Caso pessimo

$T_M(n) = \max\{T_M(x), |x| = n\}$
$S_M(n) = \max\{T_M(x), |x| = n\}$

### Notazioni

- O-grande: limite asintotico superiore.
  Data $g(n)$, $O(g(n)) = \{f(n) \mid \exists c, n_0\,(c, n_0 > 0 : \forall n \geq n_0 0 \leq f(n) \leq cg(n))\}$
- $\Omega$-grande: limite asintotico inferiore.
  Data $g(n)$, $\Omega(g(n)) = \{f(n) \mid \exists c, n_0\,(c, n_0 > 0 : \forall n \geq n_0 0 \leq cg(n) \leq f(n))\}$
- $\Theta$-grande: limite asintotico sia superiore sia inferiore.
  Data $g(n)$, $\Theta(g(n)) = \{f(n) \mid \exists c_1, c_2, n_0\,(c_1, c_2, n_0 > 0 : \forall n \geq n_0 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n))\}$

# 2 Teoremi di accelererazione lineare

- Se $L$ è accettato da una MT $M$ a $k$ nastri con complessità $S_M(n)$, per ogni $c > 0 (c \in R)$ si può costruire una MT $M'$ a $k$ nastri con complessità $S_{M'}(n) < cS_M(n)$
- Se $L$ è accettato da una MT $M$ a $k$ nastri con complessità $S_M(n)$, si può costruire una MT $M'$ a 1 nastro (*non* a nastro singolo) con complessità $S_{M'}(n) = S_M(n)$
- Se $L$ è accettato da una MT $M$ a $k$ nastri con complessità $S_M(n)$, per ogni $c > 0 (c \in R)$ si può costruire una MT $M'$ a 1 nastro con complessità $S_{M'}(n) < cS_M(n)$
- Se $L$ è accettato da una MT $M$ a $k$ nastri con complessità $T_M(n)$, per ogni $c > 0 (c \in R)$ si può costruire una MT $M'$ (a $k+1$ nastri) con complessità $T_{M'}(n) = \max\{n + 1, cT_M(n)\}$

#### Conseguenze pratiche

- Lo schema di dimostrazione è valido per qualsiasi tipo di modello di calcolo, quindi anche per calcolatori reali (es.: aumentare il parallelismo fisico (16bit $\rightarrow$ 32bit $\rightarrow \ldots$)).
- Aumentando la potenza di calcolo in termini di risorse disponibili si può aumentare la velocità di esecuzione, ma il miglioramento è al più lineare.
- Miglioramenti di grandezza superiore possono essere ottenuti solo cambiando algoritmo e non in modo automatico.

# 3 Macchina RAM

| Comando | Operazione | Complessità |
|---|---|---|
| `LOAD X` | `M[0] = M[X]` | |
| `LOAD= X` | `M[0] = X` | |
| `LOAD* X` | `M[0] = M[M[X]]` | |