

SISTEMI OPERATIVI

Gestione del Processore
Sincronizzazione dei Processi

Lezione 6 – Transazioni atomiche

Vincenzo Piuri

Università degli Studi di Milano

Sommario

- Concetto di transazione atomica
- Transazioni atomiche individuali
 - Logging
 - Check pointing
- Transazioni atomiche concorrenti
 - Serializzazione
 - Protocolli basati su locking
 - Protocolli basati su timestamp

Definizione di transazione

**Un insieme di istruzioni
che eseguono un'unica funzione logica**

- Esempio:

read
read
manipolazione dei dati
read
manipolazione dei dati
write
manipolazione dei dati
read
manipolazione dei dati
write
read
manipolazione dei dati
read
manipolazione dei dati
read
manipolazione dei dati
write
commit o abort

atomicità

Atomicità della transazione

L'effetto della transazione sulle informazioni memorizzate deve essere permanente solo se **tutte** le operazioni sono state completate correttamente senza interferenze da parte di altri processi

La sequenza di operazioni di una transazione deve essere **atomica** come un'unica operazione indivisibile

Terminazione:

corretta → **commit** effetti permanenti
errata → **abort** nessun effetto (roll back)

Tipologie di archivi

Archivio volatile

le informazioni non sopravvivono allo spegnimento del sistema

- memoria cache
- memoria centrale

Archivio non volatile

le informazioni sopravvivono allo spegnimento del sistema

- dischi magnetici e ottici
- nastri magnetici

Archivio stabile

le informazioni non vengono mai perse

- replicazione in molti archivi non volatili

Transazioni atomiche individuali

Gestione basata su

- Logging
 - Write-ahead logging
- Check pointing

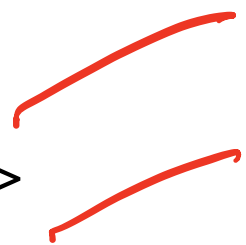
Write-ahead logging

Log (registro) **delle transazioni:**

registra in un archivio stabile le transazioni e il loro stato di esecuzione

- nome della transazione
- nome dell'oggetto dei dati
- vecchio valore dei dati
- nuovo valore dei dati

Meccanismo di write-ahead logging

- Inizio transazione → $\langle T_i \text{ starts} \rangle$
 - Fine transazione → $\langle T_i \text{ commits} \rangle$
- 

Recupero basato sul log (1)

undo(T_i)

- riporta i dati modificati dalla transazione T_i ai vecchi valori

redo(T_i)

- assegna ai dati modificati dalla transazione T_i il nuovo valore

Funzioni idempotenti

Ripristino basato sul log (2)

- Abort della transazione
 - **undo**(T_i)
- Fallimento del sistema di elaborazione
 - Per ogni transazione del log,
 - se il log contiene $\langle T_i \text{ starts} \rangle$ ma non $\langle T_i \text{ commits} \rangle$, esegue **undo**(T_i)
 - se il log contiene sia $\langle T_i \text{ starts} \rangle$ sia $\langle T_i \text{ commits} \rangle$, esegue **redo**(T_i)

Check Pointing

Problema del logging:

Tempo lungo di ripristino per lunghi log

Soluzione:

Check pointing (punti di verifica)



Periodicamente si eseguono:

- scrittura su archivio stabile dei record del log memorizzati su archivio volatile
- scrittura dei dati modificati su archivio stabile
- scrittura del record <checkpoint> su archivio stabile

Ripristino basato su check pointing

- Fallimento del sistema di elaborazione
 - Per ogni transazione del log **a partire dal check point più recente**,
 - se il log contiene $\langle T_i \text{ starts} \rangle$ ma non $\langle T_i \text{ commits} \rangle$, esegue **undo**(T_i)
 - se il log contiene sia $\langle T_i \text{ starts} \rangle$ sia $\langle T_i \text{ commits} \rangle$, esegue **redo**(T_i)

Transazioni atomiche concorrenti

Esecuzione concorrente di transazioni atomiche

- Esecuzione delle transazioni
in modo seriale
in un ordine arbitrario



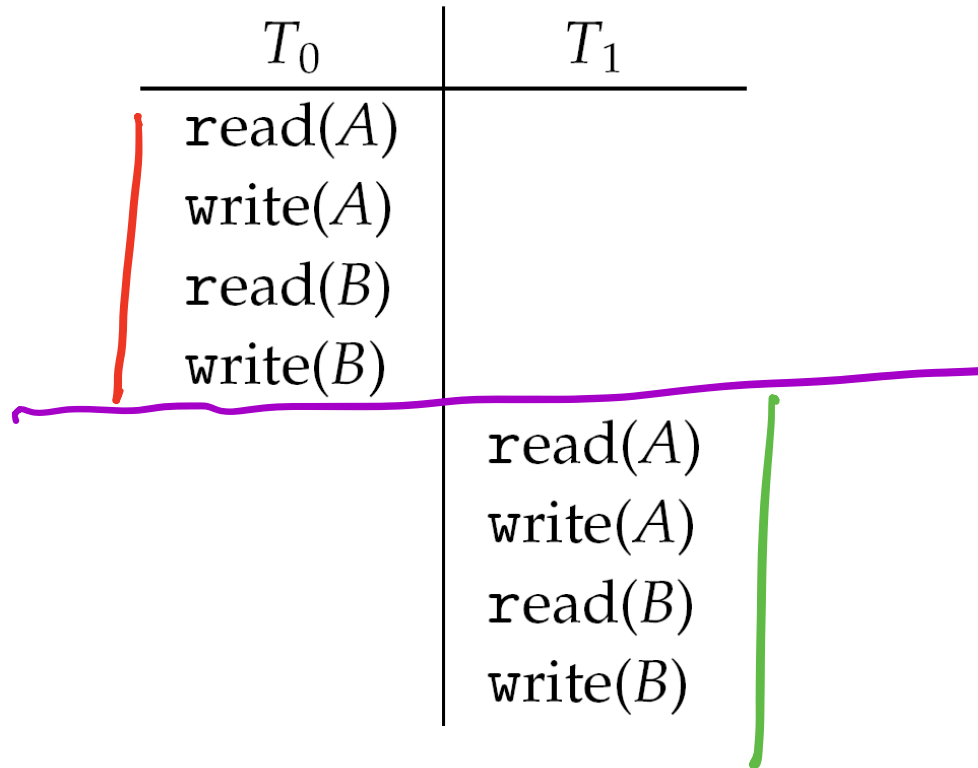
serializzabilità

Tecniche per la serializzabilità

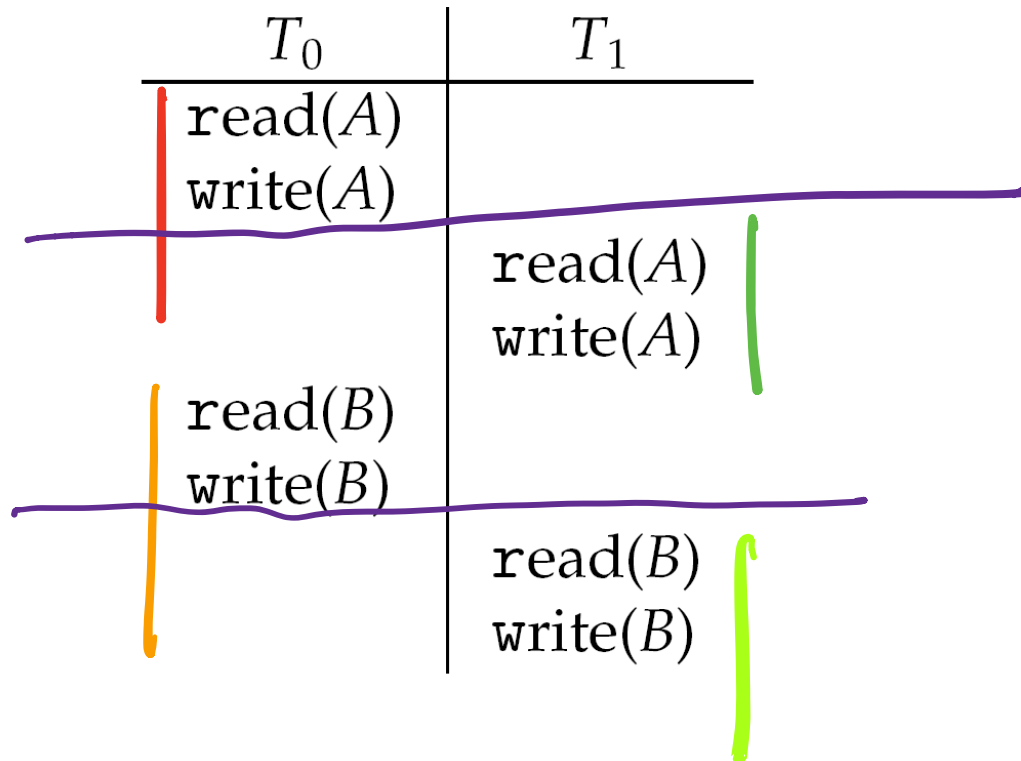
- A livello di transazione
 - Transazioni eseguite in sezioni critiche
 - Condivisione di un semaforo *mutex* comune tra le transazioni
- A livello di operazioni nelle transazioni
 - Algoritmi di controllo della concorrenza delle operazioni
 - schedulazione concorrente seriale
 - schedulazione concorrente serializzabile
 - » protocollo di lock
 - » protocolli basati su timestamp

Schedulazione concorrente seriale

T_0	T_1
read(A)	
write(A)	
read(B)	
write(B)	
	read(A)
	write(A)
	read(B)
	write(B)



Schedulazione concorrente serializzabile



Lock (blocco) è una variabile associata a un dato che definisce l'accessibilità al dato stesso

Lock = libero	→ accesso consentito
= in uso	→ transazione sospesa in attesa di blocco libero

Tipi di lock

- Lock condiviso
- Lock esclusivo

Protocollo di lock di base

- T_i esegue lock su dato Q
- Se lock disponibile, T_i accede al dato
- Se lock non disponibile,
 - Se il lock richiesto è esclusivo, T_i attende finché il dato viene rilasciato
 - Se il lock richiesto è condiviso,
 - T_i accede al dato se esso è correntemente bloccato con lock condiviso
 - T_i attende se il dato è correntemente bloccato con lock esclusivo

Serializzabilità non garantita

Protocollo di lock a due fasi

Fase di crescita (growing phase)

- una transazione può ottenere dei lock, ma non li può rilasciare

Fase di contrazione (shrinking phase)

- una transazione può rilasciare i lock, ma non ne può ottenere di nuovi

Assicura la serializzabilità

Non previene gli stalli

Serializzazione nei protocolli di lock

L'ordine di serializzazione
di ogni coppia di transazioni in conflitto
è determinato in esecuzione
dal primo lock che richiedono
e che implica incompatibilità

Timestamp

**Timestamp (marca di tempo) $TS(T_i)$
è un attributo
che rappresenta quando la transazione T_i
è entrata nel sistema**

**Timestamp è univocamente associato
alle transazioni dal sistema**

Generazione del timestamp

- Clock di sistema
- Contatore

Protocollo basato su timestamp (1)

Tipi di timestamp

- W-timestamp(Q)
- R-timestamp(Q)

**Ogni operazione read o write in conflitto
è eseguita nell'ordine della marca di tempo**

Protocollo basato su timestamp (2)

read(Q)

- Se $TS(T_i) < W\text{-timestamp}(Q)$,
la lettura è negata e T_i esegue roll-back
- Se $TS(T_i) \geq W\text{-timestamp}(Q)$,
la lettura è eseguita e
 $R\text{-timestamp}(Q) = \max\{R\text{-timestamp}(Q), TS(T_i)\}$

write(Q)

- Se $TS(T_i) < R\text{-timestamp}(Q)$,
la scrittura è negata e T_i esegue roll-back
- Se $TS(T_i) < W\text{-timestamp}(Q)$,
la scrittura è negata e T_i esegue roll-back
- Altrimenti, la scrittura è eseguita

Serializzazione nei protocolli di timestamp

L'ordine di serializzazione
di ogni coppia di transazioni in conflitto
è determinato dal timestamp
associato a ciascuna transazione alla sua attivazione

In sintesi

- Concetto di transazione atomica
- Transazioni atomiche individuali
 - Definizione
 - Gestione
 - Logging
 - Check pointing
- Transazioni atomiche concorrenti
 - Serializzazione
 - Gestione
 - Locking
 - Timestamp