

Gestione di Processi

❖ III) Concorrenza

Concorrenza

Un sistema operativo consiste in un gran numero di *attività* che vengono eseguite più o meno *contemporaneamente* dai processore e dai dispositivi presenti in un elaboratore.

Senza un modello adeguato, la coesistenza delle diverse attività sarebbe difficile da descrivere e realizzare.

Il modello che è stato realizzato a questo scopo prende il nome di *modello concorrente* ed è basato sul concetto astratto di *processo*

Tema centrale nella progettazione dei S.O. riguarda la gestione di processi *multipli*

- *Multiprogramming*
 - più processi su un solo processore
 - parallelismo *apparente*
- *Multiprocessing*
 - più processi su una macchina con processori multipli
 - parallelismo *reale*
- *Distributed processing*
 - più processi su un insieme di computer distribuiti e indipendenti
 - parallelismo *reale*

Concorrenza

□ Concorrenza

- ❖ Concetto fondamentale per la progettazione dei sistemi operativi
- ❖ Comprende diversi aspetti
 - **Comunicazione fra processi**
 - **Condivisione e competizione per le risorse**
 - **Sincronizzazione delle attività di processi multipli**
 - **Allocazione di tempo di processore ai processi**

□ Processi concorrenti

- ❖ Insieme di processi la cui esecuzione si sovrappone nel tempo
- ❖ Benefici dell' esecuzione concorrente dei processi
 - Aumento dell'utilizzo della CPU
 - Nei sistemi a partizione di tempo, ove si eseguono lavori quasi parallelamente
 - Suddivisione del carico computazionale
 - Su più processi in esecuzione contemporanea su più CPU di uno stesso sistema o, analogamente, sulle CPU di diversi calcolatori collegati in rete
 - Condivisione della stessa risorsa fisica
 - Fra diversi utenti in modo trasparente e controllato
 - Accesso contemporaneo
 - Da parte di diversi utenti ad una base dati comune e centralizzata

Concorrenza

□ Processi concorrenti

1. Processi indipendenti

- Processi che non condividono dati, temporanei o permanenti, con altri processi
- Processi che non possono influire su altri processi o subirne l'influsso
- Il S.O. deve gestire solo la **competizione** per le risorse

2. Processi cooperanti

- Processi che condividono dati con altri processi
- Processi che possono influire su altri processi o subirne l'influsso
- Il S.O. deve gestire la cooperazione tra i processi tramite **condivisione** o **comunicazione**

Concorrenza

□ Interazione fra processi

1. Competizione fra processi per le risorse

- Interazioni indesiderate e (spesso) impreviste
- Processi che vogliono accedere ad una risorsa durante la loro esecuzione
 - Ogni processo non sa della esistenza degli altri
- I processi **competono** per le risorse comuni che non possono essere utilizzate contemporaneamente
 - Fra i processi in competizione non c'è scambio di informazioni ma l'esecuzione di un processo può influenzare gli altri causando condizioni di attesa ed eventualmente problemi di blocco critico
- Il S.O. deve arbitrare questa competizione, fornendo **meccanismi di controllo e sincronizzazione**
- Possibili problemi di controllo
 - Mutua esclusione
 - Deadlock
 - Starvation

Concorrenza

2. Cooperazione fra processi tramite condivisione

- Interazioni previste
- Processi che hanno accesso a variabili condivise, file o basi di dati
 - I processi possono usare e modificare i dati condivisi senza fare riferimento agli altri
 - Ogni processo non vede esplicitamente gli altri
- I processi devono **cooperare** in modo che i dati che condividono siano gestiti correttamente
- Il S.O. deve favorire questa cooperazione, fornendo **meccanismi di controllo e sincronizzazione** che garantiscano l'integrità dei dati condivisi
- Possibili problemi di controllo
 - Mutua esclusione
 - Deadlock
 - Starvation
 - Coerenza dei dati

Concorrenza

3. Cooperazione fra processi tramite comunicazione

- Interazioni desiderate e previste
- I processi **cooperano tramite comunicazione** per eseguire un'attività comune mediante scambio di informazioni, al fine di giungere alla soluzione di un problema complesso
 - La comunicazione è un meccanismo per sincronizzare le attività dei processi
- Il S.O. deve favorire questa cooperazione, fornendo **meccanismi di comunicazione**
 - Architetture client-server, in cui un processo richiede un servizio ad un altro e attende da esso la risposta
- Possibili problemi di controllo
 - Deadlock
 - Starvation

Concorrenza

□ Esecuzione concorrente di processi

- ❖ Il S.O. gestisce l'interazione fra processi cooperanti mediante
 1. **meccanismi di sincronizzazione** fra le attività che ogni processo deve svolgere in modo parallelo rispetto agli altri
 - Controllo degli eventi mediante **"semafori logici" e primitive "wait e signal"**
 2. **meccanismi di comunicazione** ossia modalità di scambio di dati fra processi
 - a. Spazio di **memoria comune** in cui i processi possono scambiarsi i dati, nel caso di processi all'interno dello stesso calcolatore (**Modello a memoria condivisa**)
 - b. Scambio di **messaggi** e primitive **"send e receive"** fra i processi, sia all'interno dello stesso calcolatore sia in una rete di comunicazione (**Modello a scambio di messaggi**)

Concorrenza

a) Modello a memoria condivisa

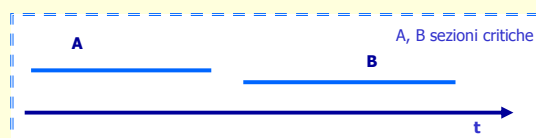
- Sistema formato da un certo numero di processi cooperanti, tutti in esecuzione asincrona e con la possibilità di condividere dati
- L'accesso concorrente a dati condivisi può causare situazioni di **incoerenza** degli stessi dati
- **Processi produttore e consumatore**
 - Un processo produttore produce informazioni che sono consumate da un processo consumatore
 - Un programma per la stampante produce caratteri che sono consumati dal driver della stampante
 - Per permettere una esecuzione concorrente di processi consumatore e produttore occorre disporre di un **vettore di n elementi** che possono essere inseriti dal produttore e prelevati dal consumatore
 - Poiché il vettore ha dimensione fissata, il consumatore deve attendere se il vettore è vuoto, viceversa il produttore deve attendere se il vettore è pieno
 - Si utilizza una variabile contatore che si incrementa ogni volta che si inserisce un elemento e si decrementa quando si preleva un elemento dal vettore

Concorrenza

- Per evitare situazioni in cui più processi accedono e modificano gli stessi dati in modo concorrente e i risultati dipendono dall'ordine di accesso (**race condition**) occorre assicurare che un solo processo alla volta possa modificare la variabile contatore
- Il segmento di codice nel quale un processo può modificare variabili comuni, aggiornare una tabella, scrivere in un file è chiamato **sezione critica**
 - Quando un processo è in esecuzione nella propria sezione critica non si deve consentire a nessun altro processo di essere in esecuzione nella propria sezione critica
 - L'esecuzione delle sezioni critiche da parte dei processi deve essere **mutuamente esclusiva**

❖ Problema della mutua esclusione

- Le operazioni con le quali i processi accedono alla risorsa comune (sezioni critiche) devono escludersi mutuamente nel tempo
- Nessun vincolo sull'ordine con il quale le sezioni critiche vengono eseguite



Concorrenza

□ Soluzioni al problema della mutua esclusione

❖ Approcci software

- Nel caso di processi concorrenti eseguiti su un solo processore o su una macchina multiprocessore a memoria condivisa, si definiscono algoritmi che consentono gli accessi alla stessa locazione di memoria in sequenza tramite un meccanismo di arbitraggio
- Algoritmo di Dekker, algoritmo di Petterson

❖ Supporto hardware

- In una macchina a singolo processore, una primitiva del kernel del sistema operativo per disabilitare gli **interrupt** quando un processo è nella sezione critica e riabilitarli alla fine della sezione critica
- In una macchina multiprocessore, **istruzioni macchina** che effettuano azioni in modo atomico, come lettura e scrittura di una locazione di memoria in un solo ciclo di istruzione e quindi non soggette ad interferenze con altre istruzioni

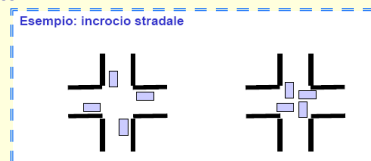
❖ Semafori

- Segnali per consentire la cooperazione fra processi

Concorrenza

□ Inconvenienti dell'esecuzione concorrente dei processi

- ❖ La mutua esclusione risolve il problema della interferenza, ma può causare il blocco di processi



❖ Situazioni di Blocco

- La **virtualizzazione** delle risorse, al fine di ottimizzare lo sfruttamento delle stesse da parte dei vari processi utente, può generare alcuni inconvenienti nel sistema di elaborazione
 1. **Deadlock**
 2. **Starvation**

Concorrenza

❑ **Deadlock (stallo)**

❖ **Blocco critico**

- Si verifica quando, in un insieme di processi, ciascun processo dell'insieme attende un evento che può essere causato solo da un altro processo dell'insieme
- I processi rimangono tutti permanentemente bloccati, senza che la risorsa contesa venga utilizzata da alcuno di essi

• Due processi condividono la stessa stampante: il processo A che inizialmente detiene la stampante, richiede i risultati del processo B per poter terminare le proprie elaborazioni e ugualmente il processo B necessita dei risultati di A

• Entrambi i processi si bloccano in attesa della terminazione dell'altro, senza che nessuno possa evolvere rilasciando la stampante

Esempio:

- siano R_1 e R_2 due risorse
- siano P_1 e P_2 due processi che devono accedere a R_1 e R_2 contemporaneamente, prima di poter terminare il programma
- supponiamo che il S.O. assegni R_1 a P_1 , e R_2 a P_2
- i due processi sono bloccati in attesa circolare

Si dice che P_1 e P_2 sono in deadlock

- è una condizione da evitare
- è definitiva
- nei sistemi reali, se ne può uscire solo con metodi "distruttivi", ovvero uccidendo i processi, riavviando la macchina, etc.

Concorrenza

❑ **Starvation (inedia)**

❖ **Blocco non critico**

- Si manifesta quando più processi potrebbero accedere alla stessa risorsa ma solo alcuni di essi vi riescono per la politica di gestione adottata
- Se a ogni processo è assegnata una priorità legata al tipo di utente e il S.O. assegna il processore ai processi con priorità più alta, si può verificare una situazione in cui un processo non ha mai accesso al processore poiché è sempre presente un processo con priorità più alta

Esempio

- sia R una risorsa
- siano P_1, P_2, P_3 tre processi che accedono periodicamente a R
- supponiamo che P_1 e P_2 si alternino nell'uso della risorsa
- P_3 non può accedere alla risorsa, perché utilizzata in modo esclusivo da P_1 e P_2

Si dice che P_3 è in **starvation**

- a differenza del deadlock, non è una condizione definitiva
- è possibile uscirne, basta adottare un'opportuna politica di assegnamento
- è comunque una situazione da evitare

Concorrenza

□ Tecniche per evitare o risolvere situazioni di blocco

- ❖ Politica Round-Robin di assegnamento circolare del processore consente di evitare fenomeni di starvation fra processi concorrenti
- ❖ Tecnica di eliminazione dei processi coinvolti sino alla completa rimozione del blocco
- ❖ Tecnica di assegnare a priori una aliquota delle risorse a tutti i processi coinvolti nel sistema oppure, ad ogni richiesta di risorse, verificare preventivamente l'assenza di un blocco critico
 - Applicazioni in ambito bancario o controllo di impianti industriali