Bilgisayar İşletim Sistemleri, Uygulama 7

Ölümcül Kilitlenme

İstanbul Teknik Üniversitesi
34469 Maslak, İstanbul

12 Nisan 2017

Bugün

Bilgisayar İşletim Sistemleri, Uygulama 7
- Ölümcül Kilitlenme
- Örnekler

Ölümcül Kilitlenme

Joseph Heller'in Catch-22 adlı romanı bir çelişki üzerine kuruludur. Romanda, olayın gerçekleşmesi bir şarta bağlıdır ancak o şartın gerçekleşmesi de aynı olaya bağlıdır.

- Deneyimsiz biri iş bulamaz. İş bulamazsa deneyim kazanamaz.
- İyi bir takım olmak için iyi futbolcular gerekir. İyi futbolcular iyi takımlara transfer olur.

Benzer çelişkiler prosesleri ilgilendiriyorsa, işletim sistemi ölümcül kilitlenmeyle karşılaşabilir.

Basit bir ölümcül kilitlenme örneği

```c
// mutex variable declarations
pthread_mutex_t lock_1;
pthread_mutex_t lock_2;
```

```c

void* faulty_functionA(void *arg){
  pthread_mutex_lock(&lock_1); // start of Critical Region 1
  printf("\nA is in Critical Region 1\n");
  fflush(stdout); // to print out buffer contents immediately
  sleep(2); // sleep for 2 seconds
  pthread_mutex_lock(&lock_2); // start of Critical Region 2
  printf("\nA is in Critical Region 2\n");
  fflush(stdout); // to print out buffer contents immediately
  pthread_mutex_unlock(&lock_2); // end of Critical Region 2
  pthread_mutex_unlock(&lock_1); // end of Critical Region 1
}
```

Basit bir ölümcül kilitlenme örneği

```c
void* faulty_functionB(void *arg){
  sleep(1); // sleep for 1 second
  pthread_mutex_lock(&lock_2); // start of Critical Region 2
  printf("\nB is in Critical Region 2\n");
  fflush(stdout); // to print out buffer contents immediately
  sleep(2); // sleep for 2 seconds
  pthread_mutex_lock(&lock_1); // start of Critical Region 1
  printf("\nB is in Critical Region 1\n");
  fflush(stdout); // to print out buffer contents immediately
  pthread_mutex_unlock(&lock_1); // end of Critical Region 1
  pthread_mutex_unlock(&lock_2); // end of Critical Region 2
}
```

Basit bir ölümcül kilitlenme örneği

```
int main(){
  pthread_t threadA, threadB; // declaring two threads
  pthread_mutex_init(&lock_1,NULL); // initializing mutex variables
  pthread_mutex_init(&lock_2,NULL); // initially unlocked
  if( pthread_create(&threadA,NULL,faulty_functionA,NULL)){ // creating threadA
    printf("Thread creation error");
    exit(1);
  }
  if( pthread_create(&threadB,NULL,faulty_functionB,NULL)){ // creating threadB
    printf("Thread creation error");
    exit(1);
  }
  if( pthread_join(threadA,NULL)){ // waiting for threadA to terminate
    printf("Thread join error");
    exit(1);
  }
  if( pthread_join(threadB,NULL)){ // waiting for threadB to terminate
    printf("Thread join error");
    exit(1);
  }
  pthread_mutex_destroy(&lock_1); // destroying mutex variables
  pthread_mutex_destroy(&lock_2);
  return 0;
}
```

Basit bir ölümcül kilitlenme örneğ: Çıktı

```
A is in Critical Region 1
B is in Critical Region 2
```

Basit bir ölümcül kilitlenme örneği - düzeltilmiş

```c
void* functionA(void *arg){
  pthread_mutex_lock(&lock_1); // start of Critical Region 1
  printf("\nA is in Critical Region 1\n");
  fflush(stdout); // to print out buffer contents immediately
  sleep(5); // sleep for 5 seconds
  while(pthread_mutex_trylock(&lock_2)){ // try to acquire lock_2
    pthread_mutex_unlock(&lock_1); // release lock_1
    sleep(1); // sleep for 1 second
    printf("\nA is WAITING\n");
    fflush(stdout); // to print out buffer contents immediately
    pthread_mutex_lock(&lock_1); // reacquire lock_1
  }
  // start of Critical Region 2
  printf("\nA is in Critical Region 2\n");
  fflush(stdout); // to print out buffer contents immediately
  pthread_mutex_unlock(&lock_2); // end of Critical Region 2
  pthread_mutex_unlock(&lock_1); // end of Critical Region 1
}
```

Basit bir ölümcül kilitlenme örneği - düzeltilmiş

```
void* functionB(void *arg){
  sleep(1); // sleep for 1 second
  pthread_mutex_lock(&lock_2); // start of Critical Region 2
  printf("\nB is in Critical Region 2\n");
  fflush(stdout); // to print out buffer contents immediately
  sleep(4); // sleep for 4 seconds
  while(pthread_mutex_trylock(&lock_1)){ // try to acquire lock_1
    pthread_mutex_unlock(&lock_2); // release lock_2
    sleep(1); // sleep for 1 second
    printf("\nB is WAITING\n");
    fflush(stdout); // to print out buffer contents immediately
    pthread_mutex_lock(&lock_2); // reacquire lock_2
  }
  // start of Critical Region 1
  printf("\nB is in Critical Region 1\n");
  fflush(stdout); // to print out buffer contents immediately
  pthread_mutex_unlock(&lock_1); // end of Critical Region 1
  pthread_mutex_unlock(&lock_2); // end of Critical Region 2
}
```

Basit bir ölümcül kilitlenme örneği - düzeltilmiş: Çıktı

Çıktı 1:

```
A is in Critical Region 1
B is in Critical Region 2
B is in Critical Region 1
A is WAITING
A is in Critical Region 2
```

Çıktı 2:

```
A is in Critical Region 1
B is in Critical Region 2
A is in Critical Region 2
B is WAITING
B is in Critical Region 1
```

Daha gerçekçi bir örnek (Yarış Durumu)

```
class Pair{ // Pair class declaration (C++)
    int a;
    int b;
    pthread_mutex_t plock; // mutex variable
  public:
    Pair(int,int); // constructors
    Pair(void){};
    ~Pair(); // destructor
    // overloaded operators for comparison
    bool operator<(Pair &);
    bool operator>(Pair &);
    bool operator==(Pair &);
    // methods for setting attributes
    void setA(int);
    void setB(int);
    void setAB(int,int);
    void print(string); // print method
    // methods for mutex operations
    void lock();
    void unlock();
};
```

Daha gerçekçi bir örnek (Yarış Durumu)

```
// constructor
Pair::Pair(int a_in,int b_in){
  a=a_in;
  b=b_in;
}
```

```
// set methods
void Pair::setA(int a_in){ a=a_in;}

void Pair::setB(int b_in){ b=b_in; }

void Pair::setAB(int a_in,int b_in){
  a=a_in;
  b=b_in;
}

// print method
void Pair::print(string name){
  cout << endl << name << ": (" << a <<","<<b<<")"<<endl;
}
```

Daha gerçekçi bir örnek (Yarış Durumu)

```
// overloaded operators
bool Pair::operator<(Pair &other){
  if(a<other.a)
    return true;
  if(a==other.a && b<other.b)
    return true;
  return false;
}
bool Pair::operator>(Pair &other){
  if(a>other.a)
    return true;
  if(a==other.a && b>other.b)
    return true;
  return false;
}
bool Pair::operator==(Pair &other){
  if(a==other.a && b==other.b)
    return true;
  return false;
}
```

Daha gerçekçi bir örnek (Yarış Durumu)

```
int main(){
  pthread_t mythreadA; // declaring mythreadA
  Pair* x=new Pair(1,2);
  Pair* y=new Pair(2,3);
  // creating a list of two Pairs (x and y)
  Pair* pList[]={x,y};
  // creating mythreadA
  if( pthread_create(&mythreadA,NULL,thread_function,(void*) pList)){
    printf("error creating thread");
    abort();
  }
  sleep(1); // to have a race
  // set attribute a of x to 5 and print x
  x->setA(5);
  pList[0]->print("x");
  // wait for mythreadA to terminate
  if( pthread_join(mythreadA,NULL)){
    printf("error joining thread");
    abort();
  }
  delete x;
  delete y;
  return 0;
}
```

Daha gerçekçi bir örnek (Yarış Durumu)

```
// thread handling function
void* thread_function(void *arg){
  Pair** pList=(Pair**) arg;
  // print x and y
  pList[0]->print("x");
  pList[1]->print("y");
  sleep(1); // to have a race
  // compare x and y and print the result
  if((*pList[0])>(*pList[1]))
    cout<<endl<<"x>y"<<endl;
  if((*pList[0])<(*pList[1]))
    cout<<endl<<"x<y"<<endl;
  if((*pList[0])==(*pList[1]))
    cout<<endl<<"x=y"<<endl;
  return NULL;
}
```

Daha gerçekçi bir örnek (Yarış Durumu): Çıktı

```
x:  (1,2)
y:  (2,3)
x<y
x:  (5,2)
```

```
x:  (1,2)
y:  (2,3)
x:  (5,2)
x>y
```

Daha gerçekçi bir örnek (Ölümcül Kilitlenme)

```cpp
// constructor
Pair::Pair(int a_in, int b_in){
  a=a_in;
  b=b_in;
  pthread_mutex_init(&plock,NULL);
}

// destructor
Pair::~Pair(){
  pthread_mutex_destroy(&plock);
}
```

```cpp
// set methods (using mutex)
void Pair::setA(int a_in){
  lock();
  a=a_in;
  unlock();
}
```

`setB` and `setAB` are modified similarly to include mutex.

Daha gerçekçi bir örnek (Ölümcül Kilitlenme)

```
// mutex lock method
void Pair::lock(){
  pthread_mutex_lock(&plock);
}

// mutex unlock method
void Pair::unlock(){
  pthread_mutex_unlock(&plock);
}
```

Daha gerçekçi bir örnek (Ölümcül Kilitlenme)

```
bool Pair::operator<( Pair &other){
  // acquire own lock
  lock();
  sleep(1); // to ensure deadlock
  // acquire other's lock
  other.lock();
  if(a<other.a){
    // release locks
    unlock();
    other.unlock();
    return true;
  }
  if(a==other.a && b<other.b){
    // release locks
    unlock();
    other.unlock();
    return true;
  }
  // release locks
  unlock();
  other.unlock();
  return false;
}
```

operator> and operator== are modified similarly to include mutex.

Daha gerçekçi bir örnek (Ölümcül Kilitlenme)

```
int main(){
  pthread_t mythreadA,mythreadB; // declaring two threads
  Pair* x=new Pair(1,2);
  Pair* y=new Pair(2,3);
  // creating two lists of Pairs (x,y) and (y,x)
  Pair* pList[]={x,y};
  Pair* qList[]={y,x};
  // creating two threads
  if( pthread_create(&mythreadA,NULL,thread_function,(void*) pList)){
    printf("error creating thread");
    abort();
  }
  if( pthread_create(&mythreadB,NULL,thread_function,(void*) qList)){
    printf("error creating thread");
    abort();
  }
  if( pthread_join(mythreadA,NULL)){ // waiting for threadA to terminate
    printf("error joining thread");
    abort();
  }
  if( pthread_join(mythreadB,NULL)){ // waiting for threadB to terminate
    printf("error joining thread");
    abort();
  }
  delete x; delete y;
  return 0;
}
```

Daha gerçekçi bir örnek (Ölümcül Kilitlenme): Çıktı

```
x:  (2,3)
y:  (1,2)
x:  (1,2)
y:  (2,3)
```