Assignment Project Exam Help

WASHINGTON STATE UNIVERSITY
VANCOUVER

CS 563

Concurrent Programming

https://tutorcs.com

WeChat: cstutorcs

Lecture 10: Synchronization, Atomic Actions, and Await Statements

Research Project Timeline

- 3/1: Reading list 1 due
- 3/22: Reading list 2 due
- 3/23: Midterm
- 3/30-4/20: Research paper presentations (2 students/lecture time)
- 4/5: Research project proposal due
- 4/22-4/29: Research project presentations (4 students/lecture time)
- 5/8: Research project due (paper and source code)

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

Example

```
y = 0;      z = 0;

P1:     x = y + z;          P2:   y = 1;
                                  z = 2;
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

- What are the final values of x, y, and z?
- Answer depends on execution order and what is atomic

Concepts

- **State:** values of variables at a point in time

Assignment Project Exam Help

- **Atomic action**: indivisible state change

https://tutorcs.com

WeChat: cstutorcs

  - hardware: load, store, add, ...
  - software: critical sections (later...)
- **History:** a trace of ONE execution; an interleaving of atomic actions
- **Property:** an attribute of ALL histories of a program, e.g., correctness

Histories

Assignment Project Exam Help

- How many histories are there in a program with n processes and m atomic actions per process?

https://tutorcs.com

WeChat: cstutorcs

$$(n\ m)!\ /\ (m!)^n$$

- If n = 2 and m=4...

Synchronization

Assignment Project Exam Help

- Synchronization

https://tutorcs.com

WeChat: cstutorcs

  - restricts the number of histories

- Example: given array a[1:n] of positive integers

  - find the maximum value m by examining all elements in parallel

Finding Max of Array

- Goal -- expressed as a predicate

Assignment Project Exam Help

```
(∀ j : 1 <= j <= n : m >= a[j])∧
 (∃ j : 1 <= j <= n : m = a[j])
```

https://tutorcs.com

WeChat: cstutorcs

- Sequential program

```
int m = 0
    for [i = 0 to n-1] {
      if a[i] > m
         m = a[i];
    }
```

Concurrent Program

- Program outline

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

```
int m = 0

  co [i = 0 to n-1] {

  body

  }
```

Program 1

- No synchronization (hence no constraints)

```
if a[i] > m {
m = a[i];
}
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

- What is true before and after this statement?
- What is wrong with this program?
  - interference
- What's the most that we can say?
  - m is SOME a[i]

Program 2

- Make it a single atomic action

Assignment Project Exam Help

https://tutorcs.com

```
⟨ if (a[i] > m) {
    m = a[i]
} ⟩
```

WeChat: cstutorcs

- The program is now correct, but...

Program 3

- Use smaller atomic actions

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

```
if (a[i] > m) {

    ⟨ m = a[i] ⟩

}
```

Program 4

- Combine 2 and 3: Double checking

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

```
if (a[i] > m) {
⟨ if (a[i] > m) {
  m = a[i]
  } ⟩
}
```

Points to Remember

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

- Synchronization is needed when we don't have independence
- Angle brackets specify atomic actions
- Double-checking technique -- especially when it is possible that the first check is false

Synchronization

Assignment Project Exam Help

- Prevents undesirable interleavings by:

https://tutorcs.com

  - Combining fine-grained atomic actions into coarse-grained atomic actions

WeChat: cstutorcs

    - **Mutual exclusion**

  - Delaying process execution until some predicate is satisfied

    - **Condition synchronization**

Atomic Actions

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

- Fine grained -- reads and writes of simple variables (loads/stores)
- Coarse grained -- programmed using <...> or real code (critical section)

Example: Fine-grained Atomicity

```
int y = 0, z = 0;
co x = y + z;
// y = 1; z = 2;
co;
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

co statement:
a simple way
to represent
concurrency

- What is the final value of x
- How to achieve expression atomicity?

Definitions

- A critical reference in an expression is a reference to a variable that is changed by other processes

Assignment Project Exam Help
https://tutorcs.com

- An assignment x=e satisfies the At-Most-Once Property if one of the following is true:

WeChat: cstutorcs

  - e contains at most one critical reference and x is not referenced by another process
  - e contains no critical references, in which case x may be read by other processes

Appearance of Atomicity

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

- If an assignment meets the requirements of the At-Most-Once property
  - execution of the assignment statement will appear to be atomic

Examples

```
int x = 0, y = 0;
co x = x + 1; // y = y + 1; oc;
```

```
int x = 0, y = 0;
co x = y + 1; // y = y + 1; oc;
```

```
int x = 0, y = 0;
co x = y + 1; // y = x + 1; oc;
```

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

Examples

Assignment Project Exam Help

```
int y = 0, z = 0;
co x = y + z;
// y = 1; z = 2;
co;
```

https://tutorcs.com

WeChat: cstutorcs

Coarse-grained Atomicity

Assignment Project Exam Help
https://tutorcs.com

- Atomicity may be required when At-Most-Once Property is not held

WeChat: cstutorcs

- Need mechanism for constructing coarse-grained atomic actions
  - Sequence of fine-grained atomic actions
  - Appearance

Specifying Atomic Actions

- `< S; >`
  - execute statement list S indivisibly (mutual exclusion)
- `< await(B); >`
  - wait for B to be true (conditional synchronization)
- `< await(B) S; >`
  - wait for B to be true, then execute S ALL AS A SINGLE ATOMIC ACTION

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

Example of use of await

Assignment Project Exam Help

```
⟨ s = s+1; ⟩
```

https://tutorcs.com

```
⟨ await (s>0) s = s-1; ⟩
```

WeChat: cstutorcs

At-Most-Once Property

- <S;> == S;
  - if either:
    - S is a single assignment and meets At-Most-Once Property requirement, OR
    - S is implemented by a single indivisible machine instruction
- <await (B);> == while (not B)
  - if B meets At-Most-Once Property requirement (i.e., contains only one critical reference)

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

Producer/Consumer Synchronization

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

- Problem:
  - copy a[n] in Producer to b[n] in Consumer using a single shared buffer
  - How? Use synchronization to alternate access to the buffer

Program

```
int buf, p = 0, c = 0;

process Producer {

   int a[n];

   while (p < n) {

         ⟨ await (p == c); ⟩

     buf = a[p];

     p=p+1;

   }

}
```

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

```
process Consumer {

   int b[n];

   while (c < n) {

         ⟨ await (p > c); ⟩

         b[c] = buf;

         c=c+1;

   }

}
```