P2 Miscellany

Dave Eckhardt
de0u@andrew.cmu.edu

“Style”

- Field names should ***mean something***
  - Even more than variable names
- Single-letter variable names may be ok
  - It's unclear single-letter field names can be ok
    - Unless you're implementing complex numbers

“Style”

- Please don't give us
  - Large chunks of commented-out code
  - Voluminous debugging (enabled)
  - Comments which are ***misleading***
  - 180 copies of singly-linked-list traversal
    - It is quite familiar to us
    - It is like kudzu

“Style”

- Please don't give us
  - Two files called mutex.c
  - One file called mutex.s
  - Containing overlapping sets of entry points
  - In three different directories
- Really

Thead Safety - 1

```
if (!initialized) {
  mutex_init(&m); /*not &mp!!!*/
  initialized = 1;
}
```

- See also
  - The “Double-Checked Locking is Broken” Declaration

Thead Safety – 2

```
mutex_init(&m);
mutex_lock(&m);
cond_wait(cv, &m);
mutex_unlock(&m);
mutex_destroy(&m);
```

Thead Safety – 3

```
/* ATOMIC deschedule */
int zero = 0;
deschedule(&zero);
```

Write Something Down!

- Instead of “I want this to be atomic”...
  - Think about what atomic means
  - ***Enumerate the hazards***
  - Write down how each hazard is avoided
- Writing down analyses helped
  - People missed ***other*** cases
  - ...but typically got the cases they analyzed
- In real life...the ***next person*** will read it too!

Thead Safety – 4

```
void foo(void){
  m_lock(&m1);
  ++counter;
  m_unlock(&m1);
}
```

```
void bar(void){
  m_lock(&m2);
  --counter;
  m_unlock(&m2);
}
```

Thread Safety – 5

- ***Compound objects*** need special care
  - For example, linked list

```
mutex_lock(list_guard);
ptr = l_last(list); /*Atypical on P2*/
mutex_unlock(list_guard);
...
...
ptr->next = 0; /* XCHG is no help */
```

Thread Safety – 5

- ***Compound objects*** need special care
  - For example, linked list
    - Traversal is safe with respect to traversal – maybe
    - Traversal is ***not safe*** with respect to deletion!
      - Make sure you see why
    - Think about other operations too...

Scheduling – 1

```
yield(-1);
```

- You ***want*** your pesky younger brother's process??

```
yield(0);
```

- Hmm....

```
if (error)
  while (1)
    ;
```

Scheduling – 2

`yield(tid);`

- How many instructions will it run?

`sleep(200);`

- This is not likely to be the right amount of time

Synchronization

- “If all you have is a hammer, everything looks like a nail.”
- We have ***multiple*** synchronization problems
  - Not all are solved correctly with mutexes
  - (or deschedule)
- Design involves
  - Issues, implications, ***choices***

Surprises - 1

```
thr_table = realloc(thr_table, nsize);

for (cur = head-> next; ...)
```

Surprises – 2

```
ptr=malloc(size);
ptr->f = 33;
...
if (issue)
  return (-4); /* meaning...? */
```

Summary

- Initial grades
  - Bell curve
  - Centered around 73%
  - Adjustment under consideration
- Summary
  - Many of you know many of the issues
  - If you are uneasy about something, asking is wise!