MICHIGAN STATE COMPUTATIONAL MATH, SCIENCE AND ENGINEERING DEPARTMENT

MICHIGAN STATE

A long list

Because concurrency is such an important topic there are a number of different types (in C++ and elsewhere).

I'll list them here, but just briefly and you can investigate more on your own if you are interested.

CMSE 822, FS21, W.F. Punch

MICHIGAN STATE

mutex variety

- timed_mutex: lock to a timepoint
- recursive_mutex: allow the same thread to recursively call itself
 - for every lock called, an unlock must be called before the mutex is released
 - only the same thread can make the multiple calls. Other threads wait
- recursive_timed_mutex: combination of the two above

CMSE 822, FS21, W.F. Punch

MICHIGAN STATE

more

- shared_mutex: two ways to lock:
 - exclusive (using lock, try_lock). Only 1 thread.
 - shared (using lock_shared, try_lock_shared). More than 1 thread can use it.
- shared_timed_mutex
 - combo of shared and timed.

CMSE 822, FS21, W.F. Punch

MICHIGAN STATE UNIVERSITY

semaphores

Semaphore (c++-20) has a count of the number of threads that can share access:

- acquire grants access and decrements the count. When the count hits 0, acquire blocks
- release ups the count, unblocks.
- not tied to one thread. The acquire thread can be different from release.

CMSE 822, FS21, W.F. Punch

MICHIGAN STATE

lock variety

- unique_lock: more general purpose lock_guard
 - can be constructed without locking
 - can be unlocked
- scoped_lock: a lock_guard to grab multiple mutexs before moving forward

CMSE 822, FS21, W.F. Punch

MICHIGAN STATE

barrier

- C++-20, requires g++/clang++ 11 or greater
 - has a count such that all threads reaching the barrier wait until the count reaches 0.

CMSE 822, FS21, W.F. Punch



condition variable

- is a way to have a thread wait and to notify other threads when processing can proceed
 - wait is on a condition (a Boolean function)
 - can notify one or many threads
 - think reader/writer or produce/consumer as natural fits.

CMSE 822, FS21, W.F. Punch

COMPUTATION	onal Math,	SCIENCE AND E	NGINEERING	DEPARTME	NT			MICHIGAN STATE	E Y
5376 004889C7 E85D505 5440 10050000 488B15 55564 38158A0A 000043 5568 C7E8A004 000043 5662 EB040000 004889 5696 GEC34839 C7E8OA 5760 89C7E8CD 0300004 5824 09000048 89D648 5855 0000488B 15C708 6616 04488B1 5806 6616 04488B1 58080 6614 0C080000 4889D 66208 4839C78 7A0200 6144 0C080000 4883D 6208 4839C48 89C24 6326 88000000 04883B 6627 4839E648 89C24 6336 88000000 04833B 6464 4889E588 00000 6484 4889E588 00000 6529 4839E548 88000 66526 FFES2A06 0000FFE 6720 000FFE5	00488815 000800000000000000000000000000000000	3 4889D648 89C7E827 050000 3 C7E8EC04 0000E8CF 040000 3 C7E8EC04 0000E8CF 040000 3 C7E8EC04 0000E8CF 040000 4889C7E8 6A040000 4889D6 5 K889C7E8 6A040000 4889D6 6 D64889C7 E8A90300 00488D 7 E82D0300 00E8E602 00066 8 EE0200000 E8A10200 006648 7 E82D0300 00E8E602 000048 8 EE0200000 E8A10200 006648 7 C8000048 8D358704 000048 8 S2D64839 C7E8602 200048 8 S2D64839 C7E8602 200048 8 S2D64839 C7E8602 200048 8 S2D64839 C7E86802 200048 8 S2D64839 C7E8602 200048 8 C966000 C7E8600 000048 8 C966000 C7E8600 000048 9 C966000 C7E8600 0000FF 6 C966000 C7E583 000000	E8 10050000 4889C348 48 39C3488D 351E0600 48 89C7E89F 64000048 31 0A000048 89D64889 2 04000008 89D64889 32 04000008 4880561 E8 62030000 4880561 E8 62030000 4880560 F7C24348 35A00400 F7C2548 35A00400 B8 05670800 0048890 B0 05670800 0048890 B0 05670800 0048890 B0 0578000 0048890 B0 0578000 0048890 B0 0004888 0557900 Q0488805A5 0700004 008E4000 T0 FC017532 23TDF8F Q0 00800000 000085D Q0 00800000 000085D Q0 00800000 000085D Q0 00800000 000085D Q0 00800000 00000000	8D354A06 0000488B 0 00488B05 9C0A0000 48D35F795 0000488B 0 C7E85804 0000488B 0 C7E85804 0000488B 0 C3488D35 5B050000 4 09000483 5B051000 0 09000488 8D51A09 0 0000488B0 9D0000048 000488B05 9C080000 0 20860000 4889C7E8 3 00483B05 9C080000 0 20860000 4889C7E8 3 0000488B0 9C00000B 6 000488B0 9C00000B 6 000488B0 9C0000B 6 000488B0 9C0000B 6 000488B0 9C0000B 6 000048B0 9C0000B 6 00000B1 9C0000B 6 00000B1 9C0000B 6 00000B1 9C0000B 6 00FFFFFF FFBAFEF 7 00FFFFFFF FFBAFEF 8 00FFFFFFF FFBAFEF 9 00FFFFFFF FRBAFEF 9 00FFFFFFF FRBAFEF 9 00FFFFFF 6 8280000 0		00 004838DE 4889C7E8 45 89C7E8E1 04000048 40 008E0800 00004839 00 4889C7E8 78040000 89 C7E33A04 0000666 FD 03000066 0F6EC348 E8 9E030000 488B154D 03 00000648 39C7E863 0F 6EC34889C7 E8C70200 88 15530800 00488916 C7 E8690200 00488B15 13 02000048 8815CE07 00 00488B15 302000000 15 04E7FFFF 488035E5 ES B8FF7F0 005DC355 89 E58B0589 01000066 7F 66480F6E C05DC355 FO 66480F6E C05DC355 FO 66580FFFF 684F0000 FF 55206 0000FF25 E9 C8FFFFF 684F0000 FF 66553A00 000000000000000000000000000000000	Hā.GHā+Hā«ĒnHā.SHHā«Ē]HāHā+Hā«Ē'ĒHāHā+Hā«Ē'ĒHāAāHā+Hā«Ē'HāHā+HKē',HāHā-Hā«Ē'HāHāHā.GHā-Hā«Ē'HāHā-Hā«Ē'AäHā-Hā«Ē'HāHā-Hā«Ē'AāHā-Hā«Ē'HāHā-Hā«Ē'HāHā-Hā«Ē'HāHā-Hā«Ē'HāHā-Hā«Ē'HāHā-Hā-Ā-Ā-ĒĒHāHā-Hā-Ā-Ā-Ā-Ā-Ā-Ā-Ā-Ā-Ā-Ā-Ā-Ā-Ā-Ā-Ā-Ā-Ā-	ā÷Hā«Ē\HçSyHāHā«ĒIHā/HçSJHā.0Hā«ĒIHā/HçSJHā.0Hā«ĒI	. HáñHác á«Ē·
								ï	
							/%		
							9		

MICHIGAN STATE

Deadlock?

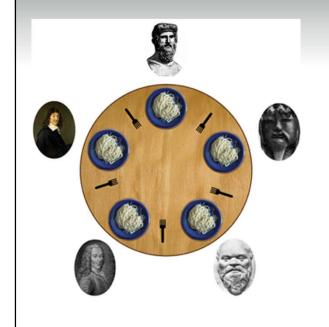
When two (or more) processes are stuck waiting for access to a resource held by a different process.

If true, we have reached an impasse. No deadlocked process can continue unless another process releases a resource.

CMSE 822, FS21, W.F. Punch

MICHIGAN STATE

Dining Philosophers



- you need two forks to eat
- you can only grab a fork to your direct left or right
- you need to alternate eating and thinking. When thinking (not eating) you must put down both forks

CMSE 822, FS21, W.F. Punch



Four Conditions for Deadlock

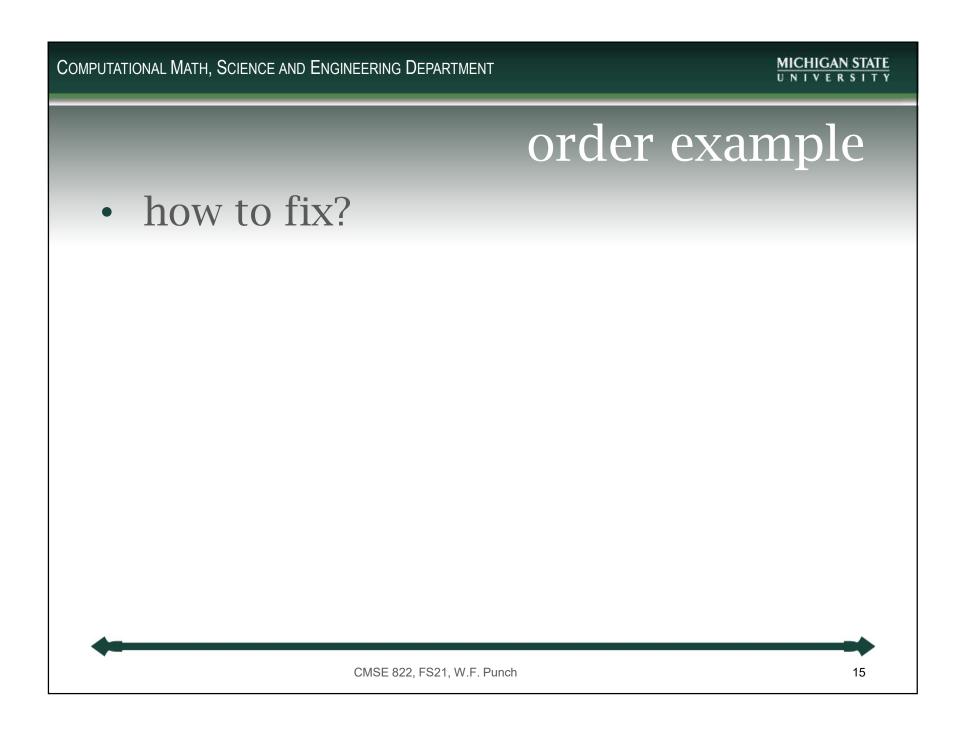
- Coffman et. al. 1971
- Necessary conditions for deadlock to exist:
 - Mutual Exclusion
 - At least one resource must be held is in non-sharable mode
 - Hold and wait
 - There exists a process holding a resource, and waiting for another
 - No preemption
 - Resources cannot be preempted
 - Circular wait
 - There exists a set of processes $\{P_1, P_2, \dots P_N\}$, such that
 - P_1 is waiting for P_2 , P_2 for P_3 , and P_N for P_1

All four conditions must hold for deadlock to occur

CMSE 822, FS21, W.F. Punch

MICHIGAN STATE COMPUTATIONAL MATH, SCIENCE AND ENGINEERING DEPARTMENT Common causes self deadlock lack of mutex ordering starvation CMSE 822, FS21, W.F. Punch

MICHIGAN STATE UNIVERSITY COMPUTATIONAL MATH, SCIENCE AND ENGINEERING DEPARTMENT selflock example how to fix this? CMSE 822, FS21, W.F. Punch



MICHIGAN STATE UNIVERSITY COMPUTATIONAL MATH, SCIENCE AND ENGINEERING DEPARTMENT barrier/starvation how to fix CMSE 822, FS21, W.F. Punch

Operation	Effect			
mutex m	Default constructor; creates an unlocked mutex			
m.~mutex()	Destroys the mutex (must not be locked)			
m.lock()	Locks the mutex (blocks for lock; error if locked and not recursive)			
<pre>m.try_lock()</pre>	Tries to lock the mutex (returns true if lock successful)			
<pre>m.try_lock_for(dur)</pre>	Tries to lock for duration dur (returns true if lock successful)			
<pre>m.try_lock_until(tp)</pre>	Tries to lock until timepoint tp (returns true if lock successful)			
m.unlock()	Unlocks the mutex (undefined behavior if not locked)			
<pre>m.native_handle()</pre>	Returns a platform-specific type native_handle_type for nonportable extensions			

Table 18.7. Operations of Mutex Classes, If Available

Operation	Effect
lock guard lg(m)	Creates a lock guard for the mutex m and locks it
<pre>lock guard lg(m,adopt_lock)</pre>	Creates a lock guard for the already locked mutex m
lg.~lock_guard()	Unlocks the mutex and destroys the lock guard

Table 18.8. Operations of Class lock_guard