

# PITFALLS

## WHAT CAN GO WRONG

Joseph Kehoe<sup>1</sup>

<sup>1</sup>Department of Computing and Networking  
Institute of Technology Carlow

CDD101, 2017

# RACE CONDITIONS

- Two or more threads perform operations on the same location at the same time
- Sequential Consistency may not be guaranteed!
- **System may reorder operations!**

TaskA:

X=1

a=Y

TaskB:

Y=1

b=X

# MUTUAL EXCLUSION AND LOCKS

- Locks are a last resort
- Use Locks to protect logical invariants not memory locations

- Avoid Mutexes where possible
- Hold at most one lock at a time
- Never call someone else's code while holding a lock unless you are sure they never acquire a lock
- Always acquire multiple locks in the same order
  - Stratify the mutexes
  - Sort mutexes
  - Backoff

# STRANGLER SCALING

- Each Mutex is a potential bottleneck
- More threads means more contention
- Fine grained locking helps
- Atomic Operations can also be helpful

# LACK OF LOCALITY

Two types of locality:

- Temporal Locality
  - The core is likely to access the same location again in the near future
- Spatial Locality
  - The core is likely to access nearby locations in the near future
- Once the cache is full use everything in it before using anything else!
- Try use Cache Oblivious algorithms

- Uneven distribution of work accross workers
- Over-decomposition
  - Divide the work into more tasks than there are workers

- If tasks are too small then overhead per task is too large
- Watch arithmetic intensity!
  - Ratio of arithmetic operations per memory access