Sequential program: determinism and termination

Concurrent program: single instruction multiple data.

Distributed program: partial failure, concurrent program where one or more thread can fail but algorithm still succeeds

Byzantine program: Can survive failures that occur at the worst time and the worst way

4 comonents of executing program:

code, data, stack, heap

coroutine: data structure for saved

state vector: hardware registers

context block: OS info to define its virtual processor

context switch: process of storing and restoring the state of a process or a thread so that execution can be resumed from the same point at later time

process: a program in execution

thread: instructions that can be scheduled as part of a process

create a thread from a procedure: int pthread\_create(pthread\_t\* threaded, const pthead\_attr\_t\* attributes, void\*(\*start)(void\*), void\*args);

pass arg to thread:

wait for thread completion: void pthread\_exit(void \*value\_ptr);

return result from thread computation:

Memorize Amdahl's formula and be able to apply it to a parallel algorithm to calculate its potential speedup: speedup(f,c) = 1/(1-f) + (f/c)

f = parallel portion, (1-f) = sequential portion of execution, c = # of cores

Memorize different forms of inter process communication:

Pipes, message queus, shared mem, shared files, signals, sockets, mpi, semaphores, remote var access, remote procedure call, remote process invocation

Dis/Advantages of FIFO: like a file, can be opened for read/write or both intermittently

message queues: control of # of messages buffered and max suze of a message. Discrete boundaries, can have 0 length. Non-negative priority values.

shared mem segs: too general pruprose, no high level interfaces. For private or selective sharing use. Different permissions. Created 32 bit keys must be distributed by participants

mem mapped files: map a file or part of a file to a memoery range in one or more processes. Desirable to a single process to avoid overhead of lots of system io calls. Best used when the same data range is being accesed by multiple processes

Explain why openmp is applicable for shared mem arch but not distributed mem:

Shared – all processors share a global pool of memory; simpler to program; bus contention leads to poor scalability

Difference between private and shared openmp vars

private: local in scopre to loops closure. Vars are uninitialized no matter what value they have on the “outside”. Shared: reference declaration for the listed vars that are local in scope to the loops closure. Vars are pointers to the corresponding outside vars’ values. Identifiers not named in any clause are shared by defaut.

Use openmp parallel for and reduction to convert sequential c program to parallel

Explain diff between temporal and spatial locality

Temporal: refers to the reuse of the same address within relatively small time durations. Spatial: refers to the use of data within relatively close storage locations

Apply loop unrolling and software pipelining

Explain how super scalar, pipelines processors speed up program execution

Super scalar- multiple integer or floating point ALUs

Pipeline – executes instructions in steps like an assembly line

Calculate result of simple sse instructions