# Innovative System and Application Curriculum on Multicore Systems

Pangfen Liu [1], Greg C. Lee [2], Jenq Kuen Lee [3], and Cheng-Yen Lin [3]

[1] National Taiwan University, Taipei, Taiwan.
[2] National Taiwan Normal University, Taipei, Taiwan.
[3] National Tsing Hua University, Hsinchu, Taiwan.

MOE Embedded Software Consortium, Taiwan.

# Outline

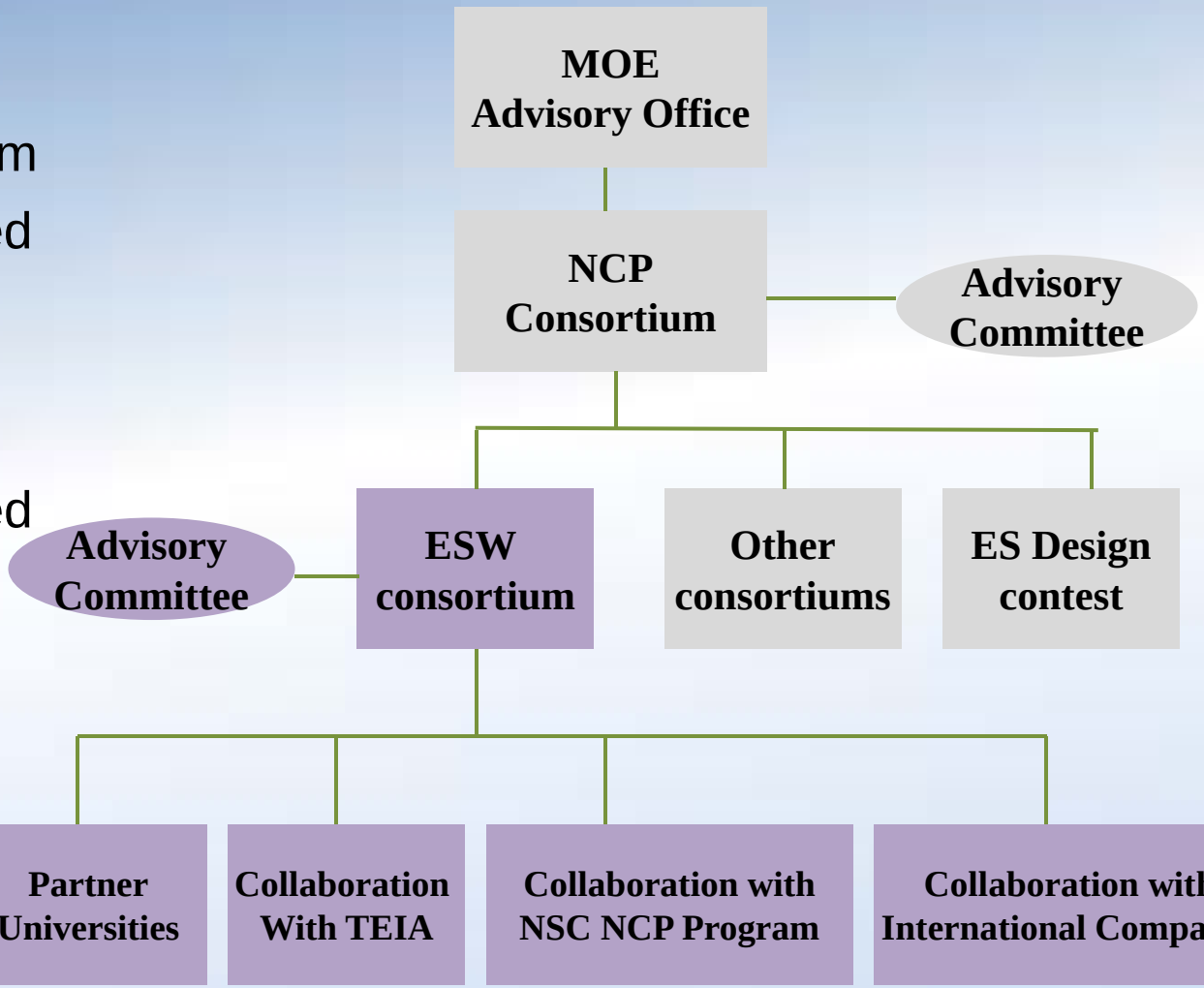
- Taiwan ESW Consortium
- Motivation
- Lab modules with Multicore systems
- Discussions
- Conclusion

# Taiwan Embedded Software Consortium

- The **Embedded Software Consortium**, established in February 2004 in Taiwan, is the consortium funded by the **Ministry of Education**
- The ESW Consortium focuses the development of embedded software curriculum.
- We hope to provide a reference curriculum for universities in Taiwan to develop their embedded program.
- Currently conjunction with National Communications Program(NCP)

# ESW Consortium

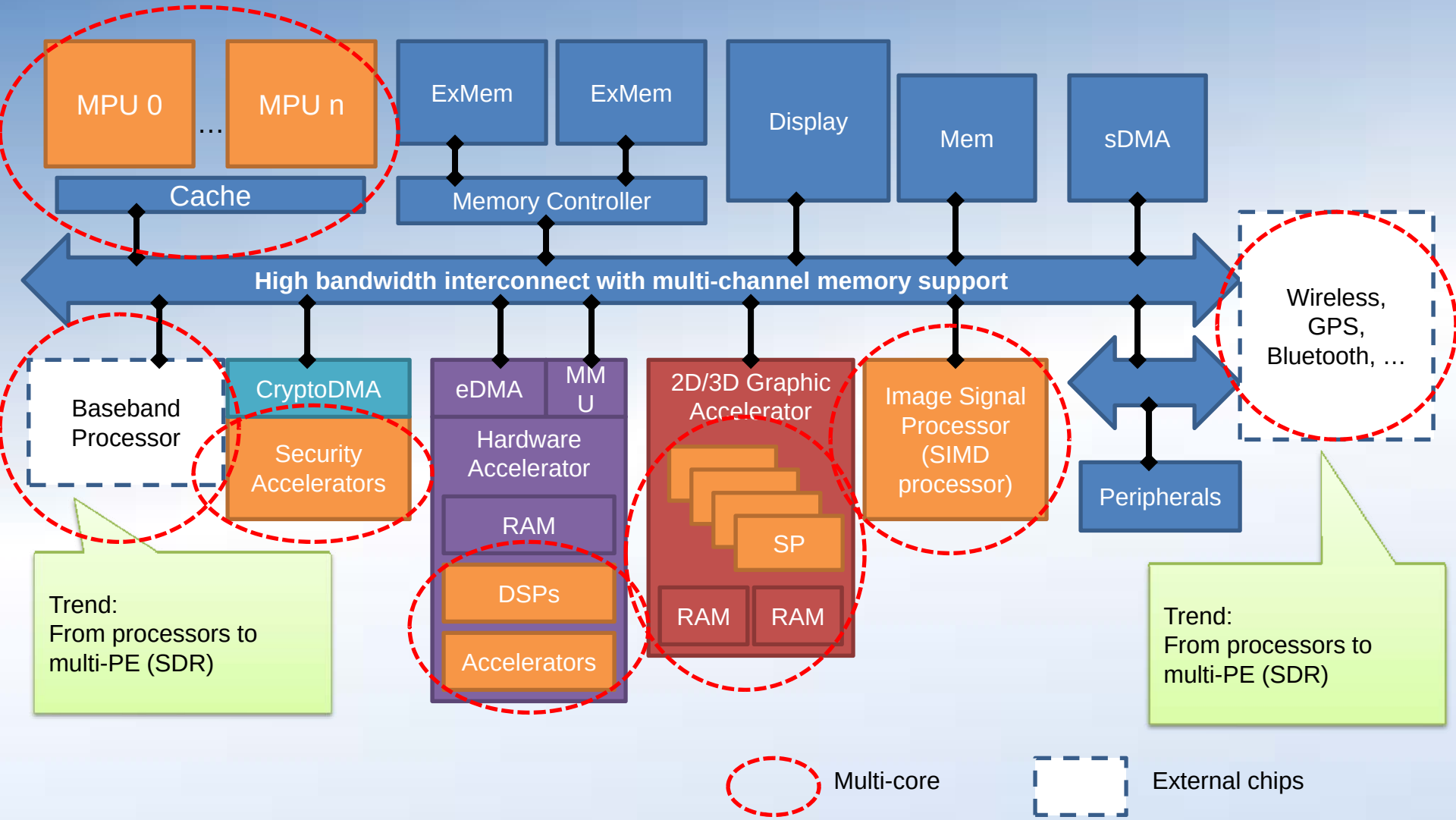
- Hands-on Lab modules
- Focus:
  - •Basic/Advance embedded curriculum
  - •Domestic embedded processor platforms
  - •Open platforms (Android)
  - •Multicore/Embedded Multicore platforms
- •ESW promotion
- •Embedded System Hardware/Software Design Contest



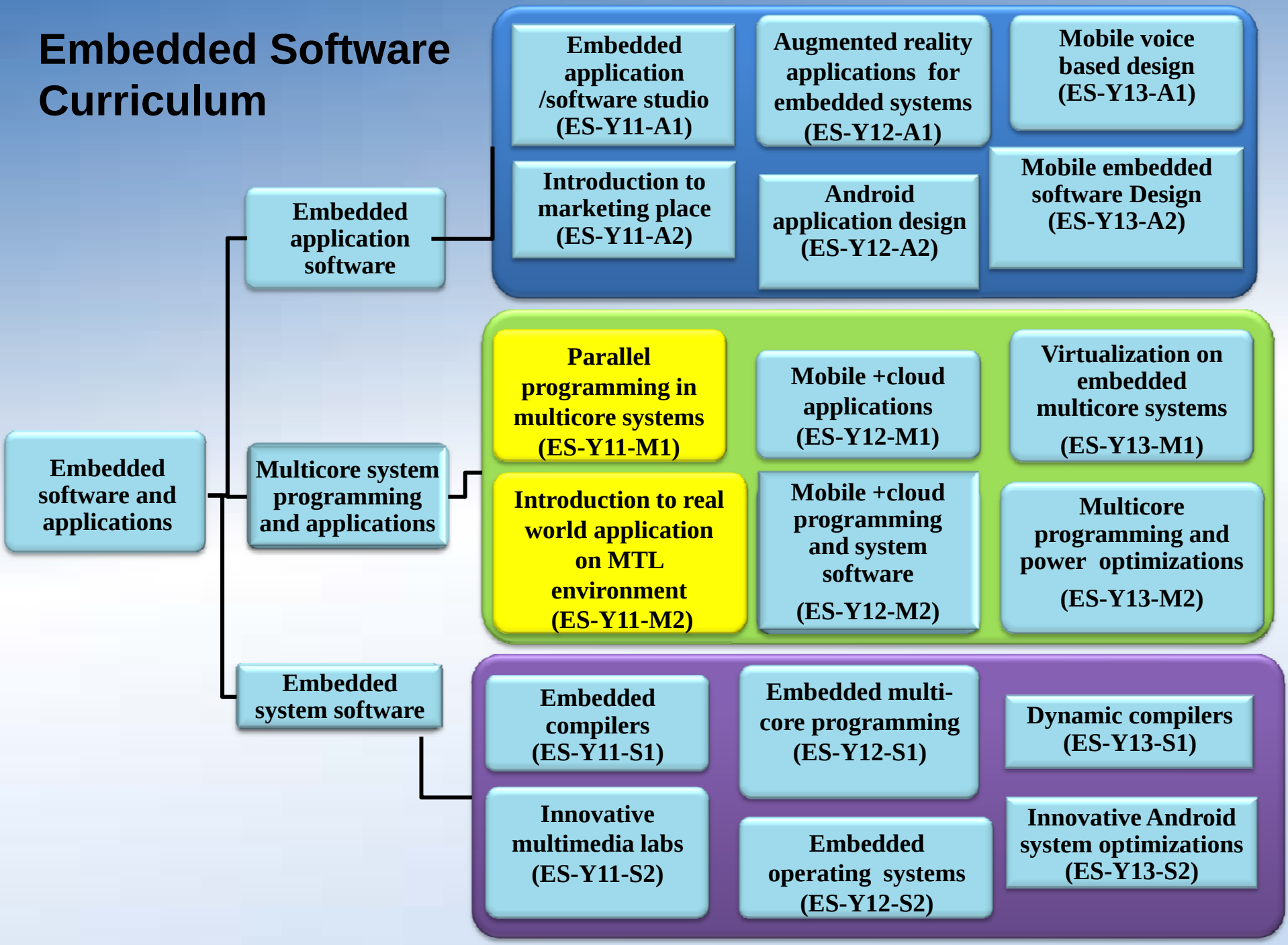
# Motivation

- The multicore architecture are increasingly important in system design
  - Amend traditional content of system education for multicore software development
- Shared ground between embedded multicore systems and high performance parallel systems
- Help students to learn the **parallel design patterns** of parallel programming to lay the foundations for advanced multicore system research.
- Government Support and **Intel** Collaboration

# Multicore/Embedded Multicore Trends



# Embedded Software Curriculum



# Course Introduction

| Number | Name | Introduction |
|---|---|---|
| ES-Y11-A1 | Embedded application/software studio | Introduction to Android programming |
| ES-Y11-A2 | Introduction to marketing place | Developing and business trend of embedded application will be introduced |
| ES-Y12-A1 | Augmented reality applications for embedded systems | Advanced embedded software for augmented reality application and lab modules based on embedded hardware with sensors support |
| ES-Y12-A2 | Android application design | Mobile application design and implementation in the Android environment |
| ES-Y13-A1 | Mobile voice based application design | The introduction of voice recognition and UI control techniques based on mobile devices |
| ES-Y13-A2 | Mobile embedded software design | Mobile applications design and lab modules on based domestic embedded platforms |

# Course Introduction

| Number | Name | Introduction |
|---|---|---|
| ES-Y11-M1 | Parallel programming in multicore systems | Principles and practice for parallel programming. Software studio are evaluated on Intel MTL |
| ES-Y11-M2 | Introduction to real world application in MTL environment | Wide range of real world multicore applications will be introduced |
| ES-Y12-M1 | Mobile + Cloud applications | The design and implementation of mobile cloud applications |
| ES-Y12-M2 | Mobile + Cloud programming and system software | Understand the system software and programming tools for mobile cloud applications |
| ES-Y13-M1 | Virtualization on embedded multicore systems | The advanced virtualization techniques for embedded system will be introduced |
| ES-Y13-M2 | Multicore programming and power optimization | The power optimization techniques for multicore platform |

# Course Introduction

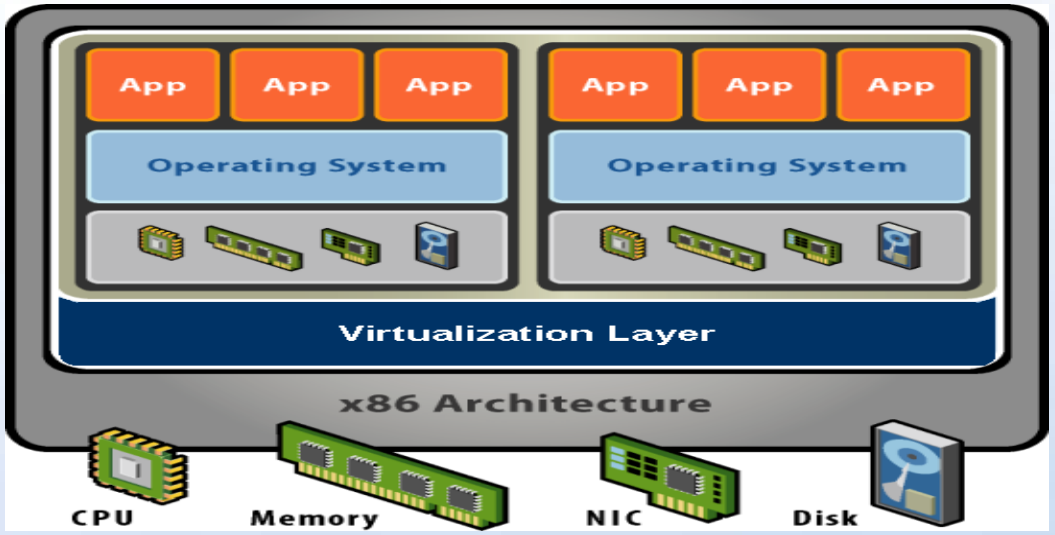
| Number | Name | Introduction |
|---|---|---|
| ES-Y11-S1 | Embedded compilers | Compiler techniques and system software development flow for embedded systems will be introduced |
| ES-Y11-S2 | Innovative multimedia labs | The design and implementation of portable multimedia applications |
| ES-Y12-S1 | Embedded multicore programming | Introduction to embedded multicore platforms and programming techniques |
| ES-Y12-S2 | Embedded operating systems | Introduction to embedded operating systems |
| ES-Y13-S1 | Dynamic compilers | Advanced dynamic compilation techniques will be introduced |
| ES-Y13-S2 | Innovative Android system optimizations | Mobile system optimization techniques based on Android will be introduced |

# This semester

- **Virtualization techniques on embedded systems**
  - Process virtualization: language level, OS level, Cross-ISA
  - Device virtualization
  - System virtualization
  - **Facilities**
    - Prof. Wei-Chung Hsu(NCTU), Prof. Chi-Sheng Shih(NTU),... etc.
  - **Labs**
    - QEMU/LnQ
    - JVM/CVM/KVM
    - Xen/Opennebula
    - OpenStack/Ubuntu UEC
    - GPU+VMGL
    - Hadoop+HDFS

**Outline**

- **Introduction to virtual machines**
- **Fast emulation: dynamic binary translation**
- **Full virtualization and paravirtualization**
- **Architecture supports for virtualization**
- **KVM-based virtualization for ARM-based embedded systems**
- **Optimizations**
- **Hypervisor for embedded real-time systems**
- **Emerging applications of embedded system virtual machines**

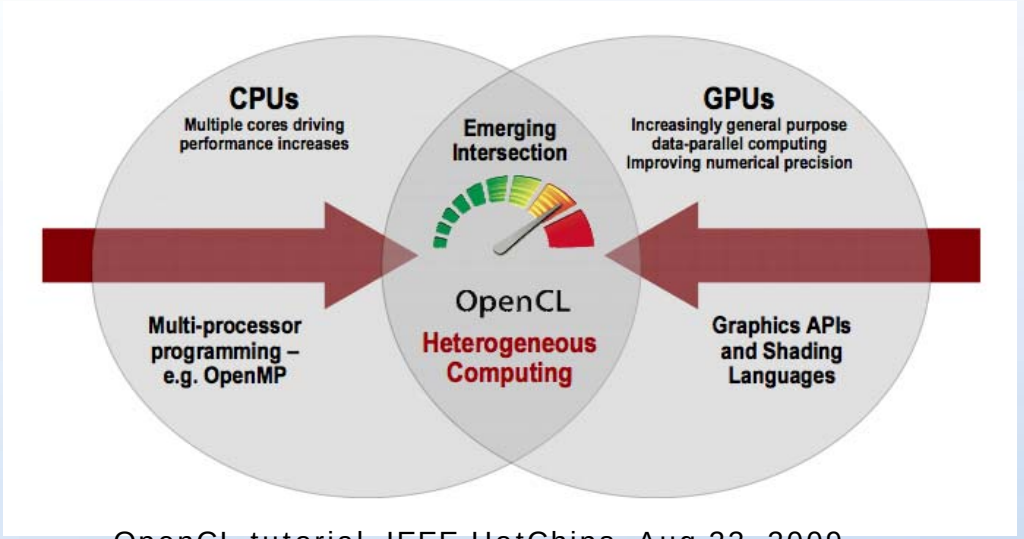


# This semester

- **Embedded multicore programming**
  - Embedded multicore architectures
  - GPU/GP-GPU
  - OpenCL
  - **Facilities**
    - Prof. Ching-Hsien Hsu(CHU), Prof. Kuan-Ching Li(PU), Prof.Yuan-Shin Hwang(NTUST) ...etc.
  - **Labs**
    - OpenCL programming on X86/ATI/NVIDIA

# Outline

- Introduction to parallel programming
- GPU Architecture
- Introduction to OpenCL
- The programming model of OpenCL
- Synchronization
- Debugging techniques
- Case study for optimizations



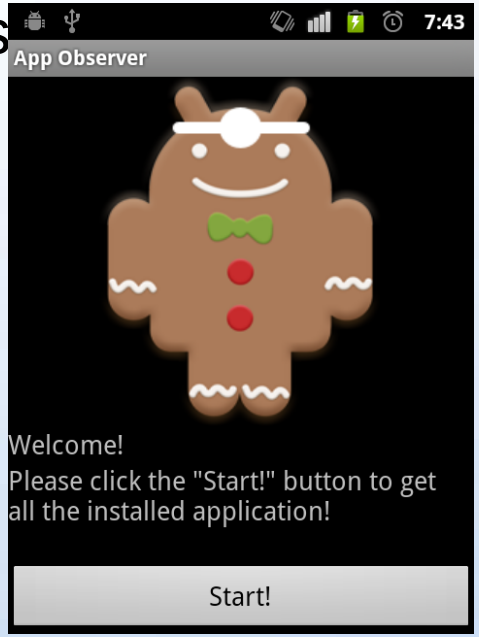

OpenCL tutorial, IEEE HotChips, Aug.23, 2009

# This semester

- **Android programming**
  - Understand the Android system
  - Mobile applications
  - Sensor applications
  - **Facilities**
    - Prof. Gwan-Hwan Hwang(NTNU), Prof. Shih-Hao Hung(NTU), Prof.Shang-Hung Wu Hwang(NTHU) ...etc.
  - **Labs**
    - Android app widget
    - GPS applications
    - Android graphic applications

# Outline

- Introduction to Android system
- Understand the development of Android app
- The app widget
- The Android UI design
- GPS/Sensor Applications
- Optimization techniques
- Advanced graphic applications
- Cloud and Devices



# This semester

- **Embedded system applications**
  - The development of embedded applications
  - Domestic embedded platforms
  - Network & multimedia applications
  - **Facilities**
    - Prof. Jyh-Cheng Chen(NCTU), Prof. Jing Chen(NCKU), Prof.Jyh-Shing Jang(NTHU) ...etc.
  - **Labs**
    - Wireless applications
    - Speech and Audio applications

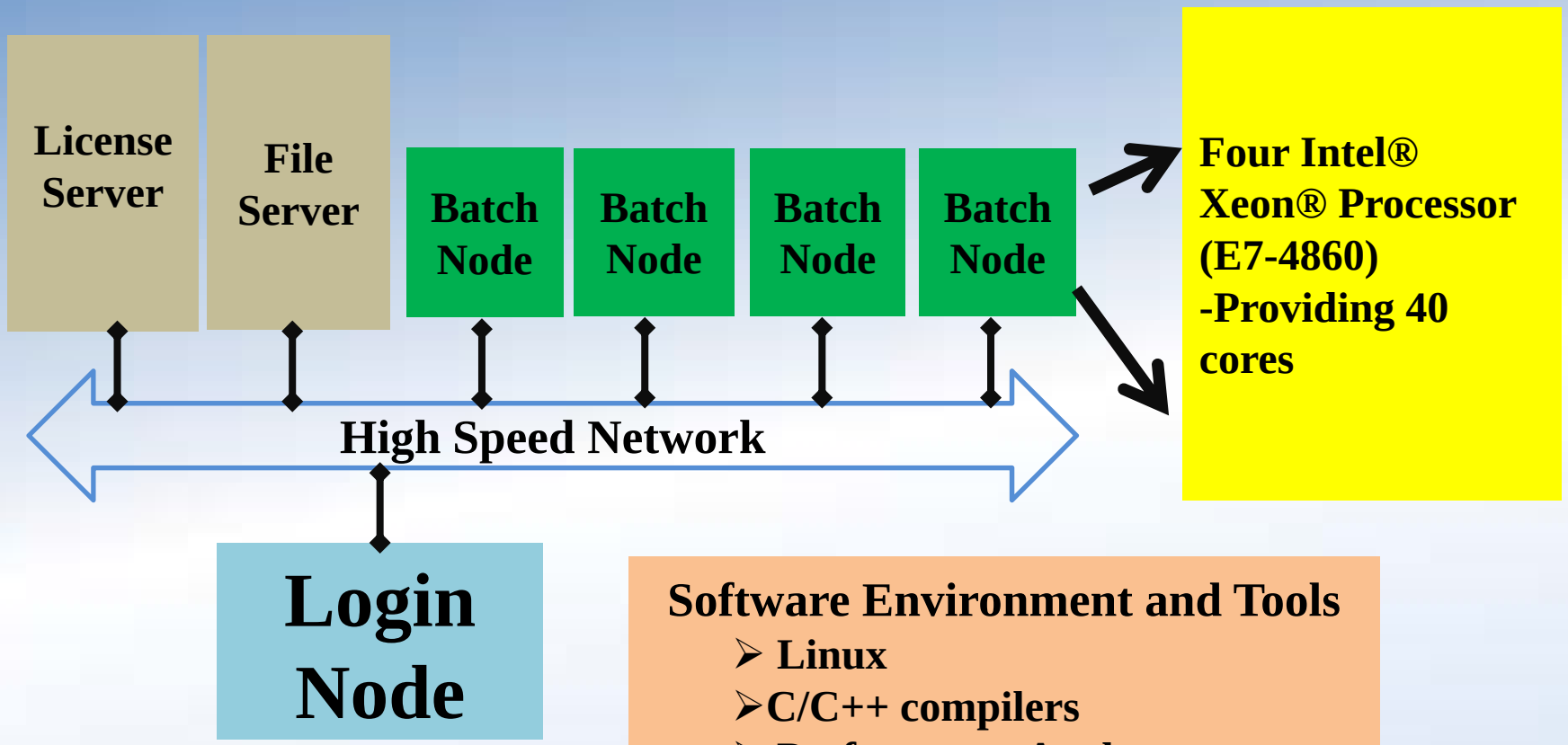
# Outline

- Embedded software design and practice
  - ✓Cross compilation
  - ✓Embedded OS
  - ✓Device driver
- Wireless applications
  - ✓802.1X
  - ✓Socket programming
  - ✓Security standards
- Speech and Audio processing
  - ✓Audio signal processing
  - ✓Speech features
  - ✓Recognition techniques
- PAC/PAC-DUO Introduction
  - ✓Toolchian
  - ✓Media codecs
  - ✓DVFS

# Intel MTL/MOE Curriculum Program

- Intel collaborates with Taiwan MOE
- Intel provides the resources, trainings, tools and community
- Course materials design by Taiwan professors
- Two Lab Modules :
  - Lab Modules for Parallel Programming in Multicore Systems
  - Lab Modules of Parallel Programming on Real-World Applications
- 14Faculties: Pangfeng Liu (NTU), Jen-Wei Hsieh (NTUST), Rong-Guey Chang (CCU), Chao-Tung Yang (THU), Chih-Ping Chu (NCKU), Wuu Yang (NCTU), Greg Lee (NTNU), Tsung-Che Chiang(NTNU), Che-Rung Lee (NTHU), Li-Chun Wang (NCTU), Charles Wen (NCTU), Ren-Guey Lee (NTUT). Chung-Chih Lin (CGU), Che-Lun Hung (PU)

# Intel MTL Environment

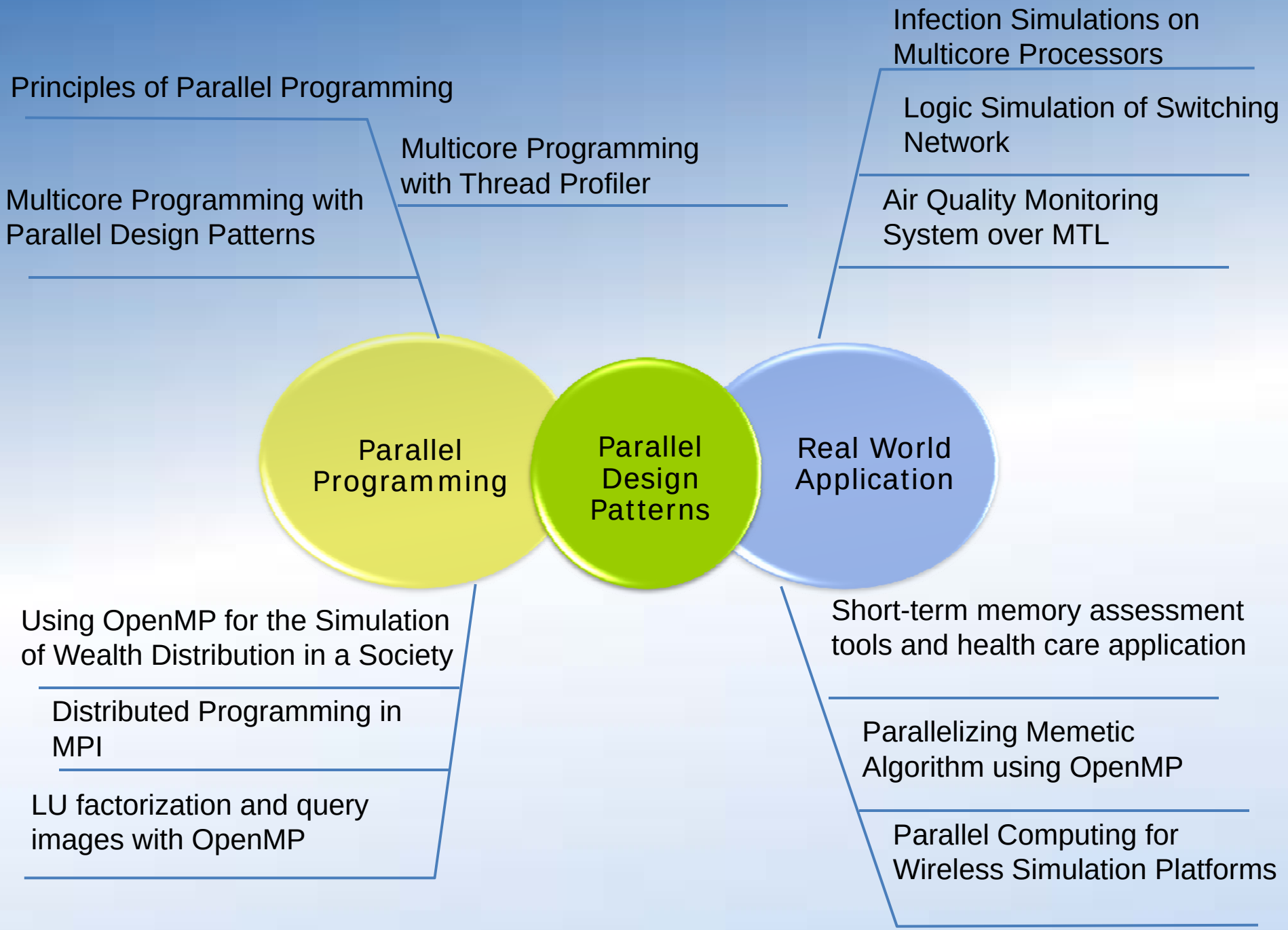



**Software Environment and Tools**

- Linux
- C/C++ compilers
- Performance Analyzers
- Thread builder
- Runtime library support

# Multicore Curriculum

- Software Tool Lab Modules for Parallel Programming in Multicore System
  - Principle of parallel programming
  - Programming models
  - Parallel algorithms
  - Tools for parallel programming
- Lab Modules for Real world Application
  - Real world application
    - Infection simulations, logic simulation, healthcare applications, monitoring applications with air quality…
  - Parallel design patterns


Principles of Parallel Programming
Multicore Programming with Parallel Design Patterns
Multicore Programming with Thread Profiler
Infection Simulations on Multicore Processors
Logic Simulation of Switching Network
Air Quality Monitoring System over MTL
Parallel Programming
Parallel Design Patterns
Real World Application
Using OpenMP for the Simulation of Wealth Distribution in a Society
Distributed Programming in MPI
LU factorization and query images with OpenMP
Short-term memory assessment tools and health care application
Parallelizing Memetic Algorithm using OpenMP
Parallel Computing for Wireless Simulation Platforms

# Principle of Parallel Programming

- Multicore Systems
  - Shared/Distributed memory
- Level of Parallelism
  - Data parallelism
  - Task parallelism
- Parallel Programming Models
  - Pthread
  - OpenMP
  - MPI
  - CUDA
  - MapReduce
- Amdahl's law

# Amdahl's Law

- Speedup = $\frac{T_1(n)}{T_p(n)}$

Assume p→ Infinite

Speedup = 1/ r

$$= \frac{T_{seq}(n) + T_{parallel}(n)}{T_{seq}(n) + T_{parallel}(n)/p}$$

$$= \frac{1}{T_{seq}(n)/T + T_{parallel}(n)/(T*p)}$$

$$r = \frac{T_{seq}}{T}$$

$$= \frac{1}{r + (1-r)/p}$$

# Sandia's Version of Amdahl's Law

$$\text{Speedup}(n) = \frac{T_1(n)}{T_p(n)}$$

N → Infinite

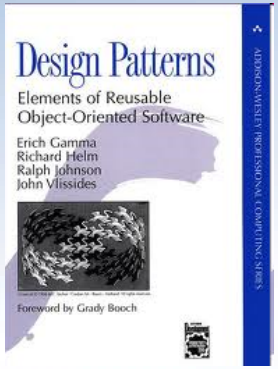
Speedup(n) → p

$$= \frac{T_{seq}(n) + T_{parallel}(n)}{T_{seq}(n) + T_{parallel}(n)/p}$$

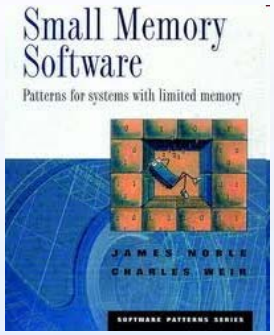
$$= \frac{O(n) + O(n^3)}{O(n) + O(n^2)/p}$$

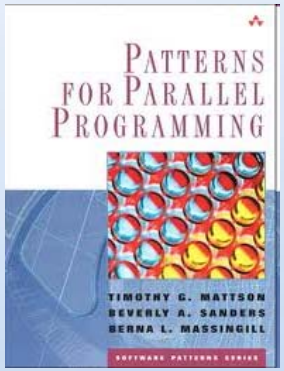
# Design Patterns

- Brief introduction to design patterns
  - Proposed by C. Alexander for city planning and architecture.
  - Introduced to software engineering by Beck and Cunningham.
  - Become prominent in object-oriented programming by GoF.




- Design patterns describe “good solutions” to recurring problems in a particular context.
  - Patterns for **object-oriented programming**
    - Creational patterns, Structural patterns, Behavioral patterns, etc.
  - Patterns for **limited memory systems**
    - Compression, Small data structures, Memory allocation, etc.
  - Patterns for **parallel programming**
    - Finding concurrency, Algorithm structure, Supporting structures and Implementation mechanisms.



# Parallel Design Patterns

**Parallelization can be a process to transform problems to programs by selecting appropriate patterns.**

- Finding Concurrency

  ⇩ decomposition of problems

- Algorithm Structure

  ⇩ appropriate algorithms

- Supporting Structures

  ⇩ appropriate program constructs

- Implementation Mechanisms

  ⇩ parallelized programs

Design Space of Parallel Patterns

- **Decomposition** patterns: {data, task}
- **Dependency analysis** patterns: {group tasks, order tasks, data sharing}
- **Design evaluation** pattern

How the given problem is organized?

- By **tasks**: {task parallelism, divide & conquer}
- By **data decomposition**: {geometric, recursive}
- By **flow of data**: {pipeline, event-based coord.}

Software constructs to express parallel algorithms

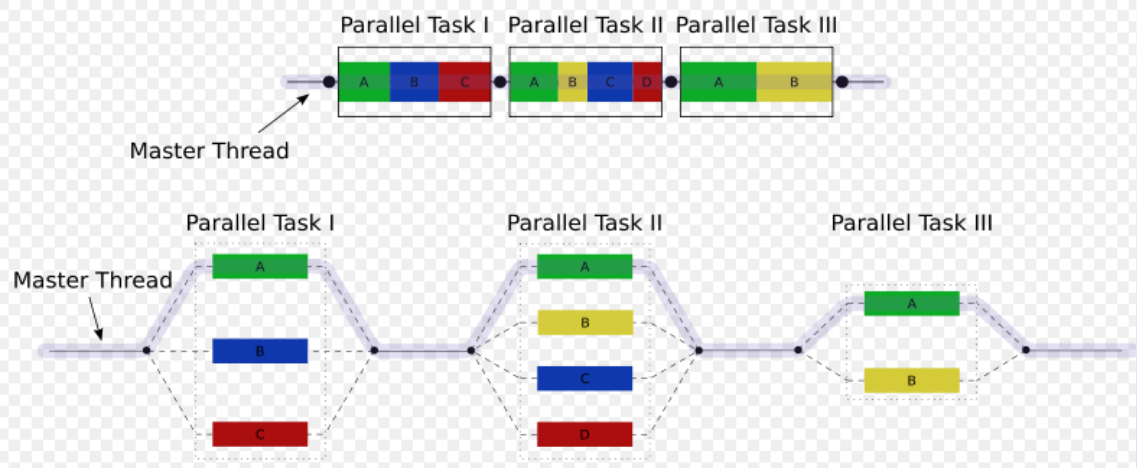
- **Program structures**: {SPMD, master/worker, loop parallelism, fork-join, client-server, SIMD}
- **Data structures**: {shared data, shared queue, distributed array}

- UE management: {thread/process creation/destr.}
- Synchronization: {barrier, mutex, mem fence}
- Communication: {msg passing, collective comm}

These patterns are summarized from the book, "Patterns for Parallel Programming" by Mattson et al

# Lab Module: Principles of Parallel Programming

- Basic of Parallel Programming
  - Programming Models
    - Open MP
  - Runtime library
    - Intel TBB
  - Fork-Join Parallelism




TBB Components

Generic Parallel Algorithms
parallel_for, parallel_for_each*
parallel_reduce
parallel_scan
parallel_do
pipeline
parallel_sort
parallel_invoke*

Concurrent Containers
concurrent_hash_map
concurrent_queue
concurrent_vector

Synchronization Primitives
atomic, mutex, recursive_mutex
spin_mutex, spin_rw_mutex
queuing_mutex, queuing_rw_mutex
null_mutex*, null_rw_mutex*

Task scheduler
task_group*
task
task_scheduler_init
task_scheduler_observer

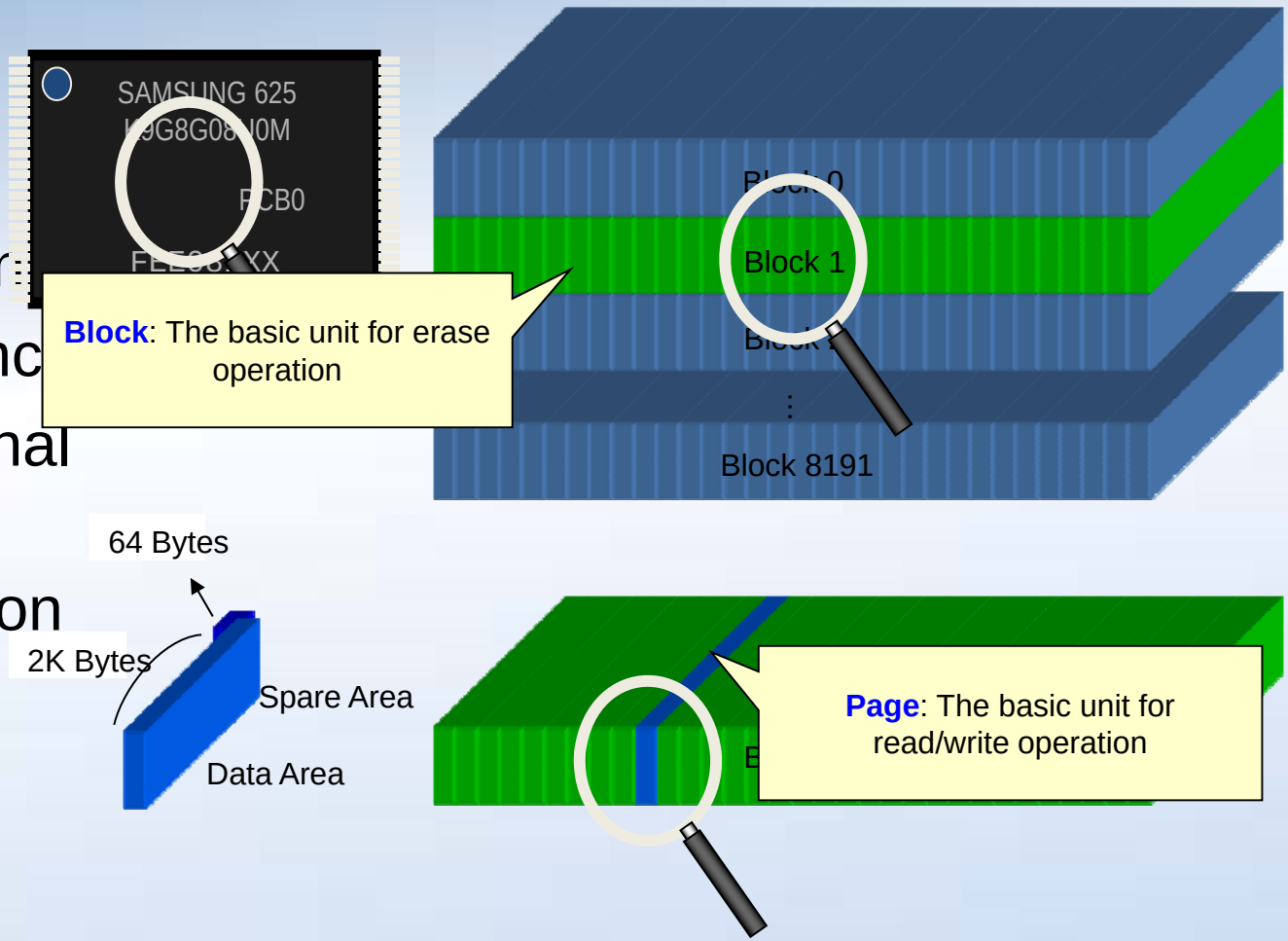
Threads
tbb_thread

Memory Allocation
tbb_allocator
cache_aligned_allocator
scalable_allocator

# Lab Module: Multicore Programming with Thread Profiler

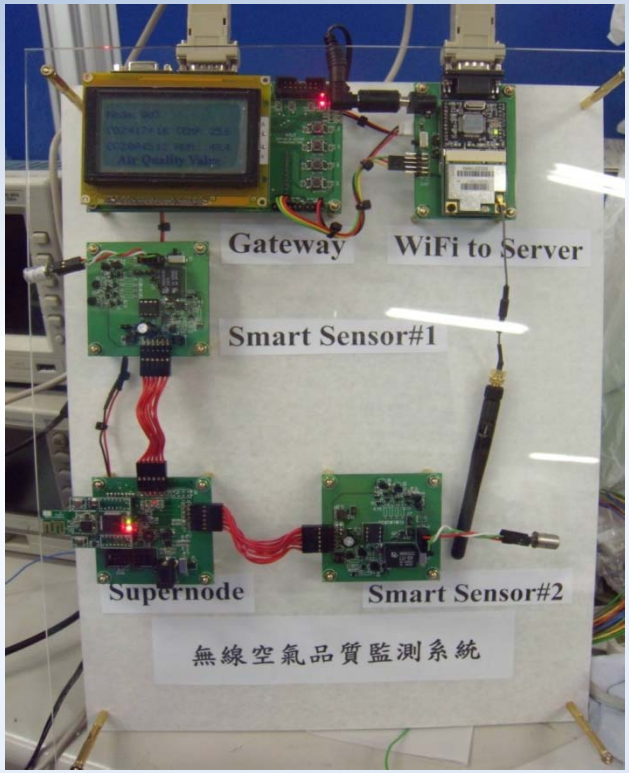
- Performance bottleneck analysis with software tools
  - Critical section
  - Loading balanc
  - Waiting External event
  - Synchronization overhead
- Sequential -> Parallel

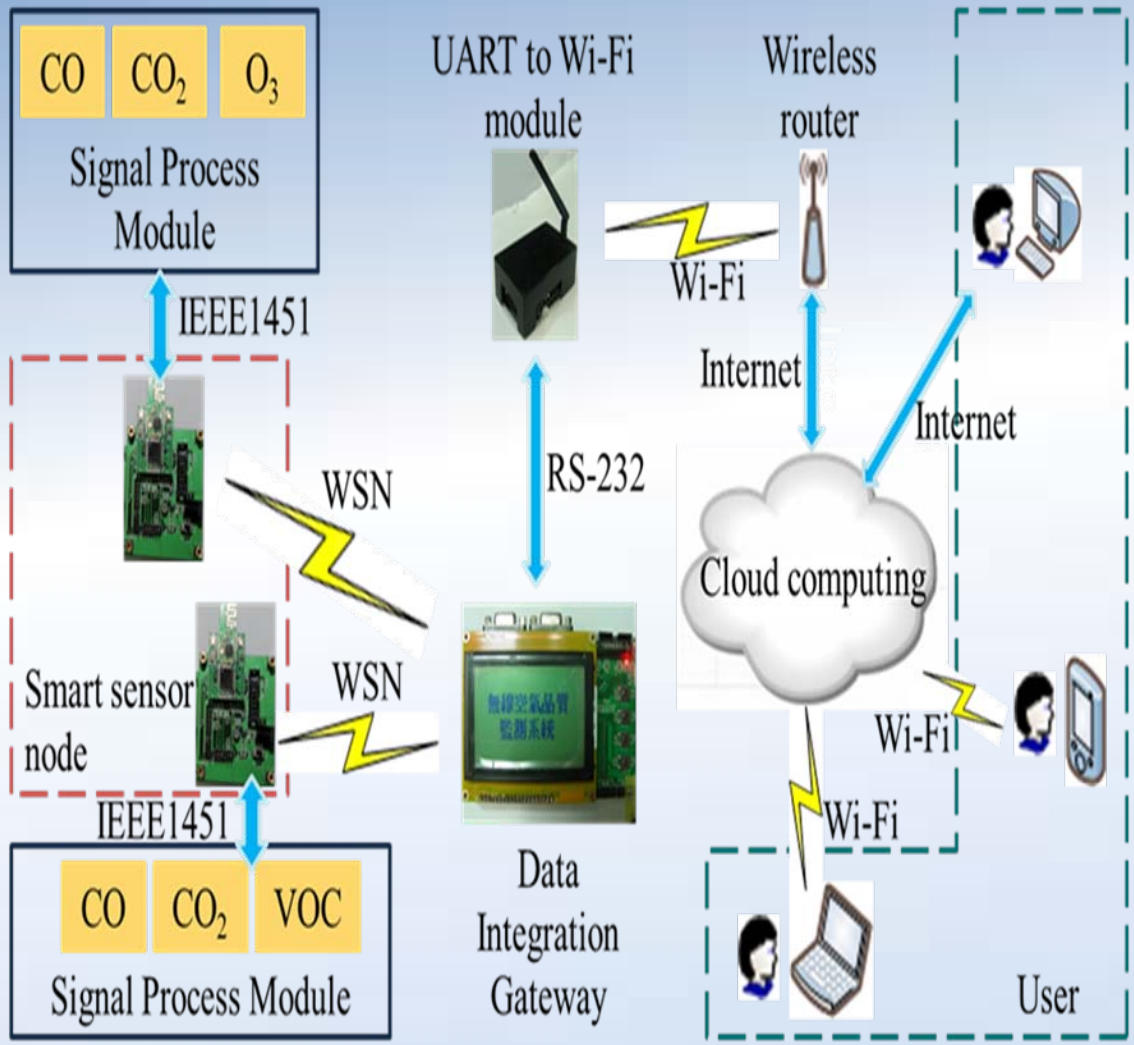
Project: Try the parallel the flash memory simulator FAST



# Lab Module: Air Quality Monitoring System over MTL platform

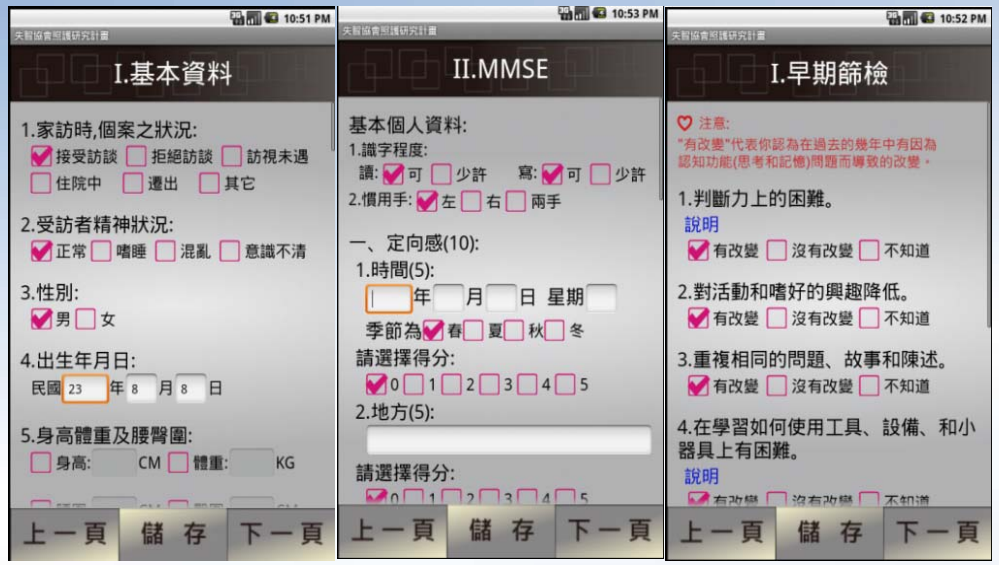
- Air Quality Monitoring with Sensors
  - CO, $CO_2$,$O_3$
- Data Analyzing on Cloud System



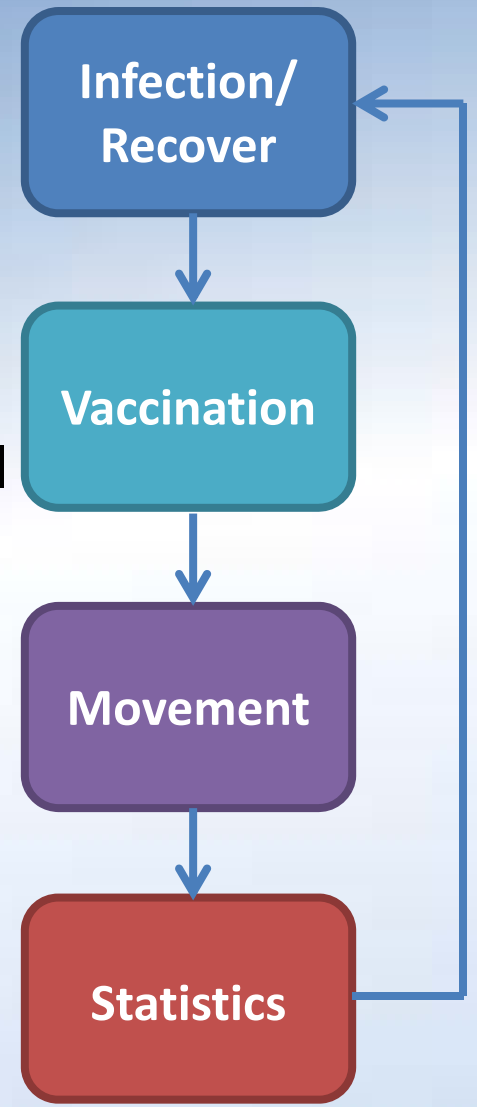
# Lab Module: Development of short-term memory assessment tools and for healthcare applications

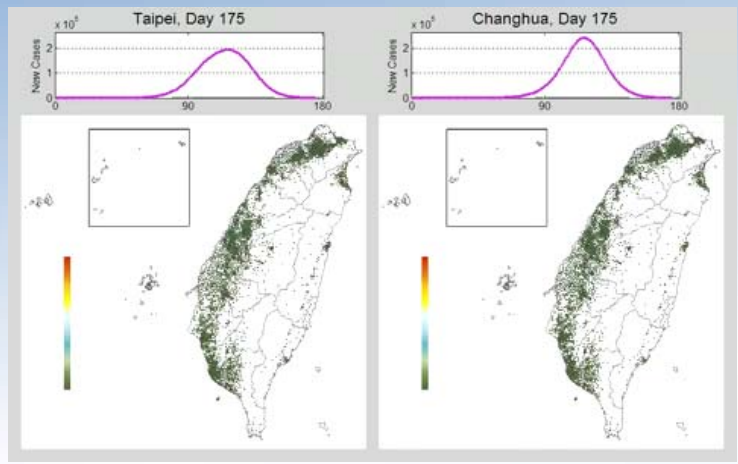
- Dementia Diagnosing System
  - PDA/hand held devices
  - Memory assessment system to help diagnosing dementia disease
- Cloud Database System
  - Global rating analysis for Alzheimer disease
  - Elder degenerative trend analysis



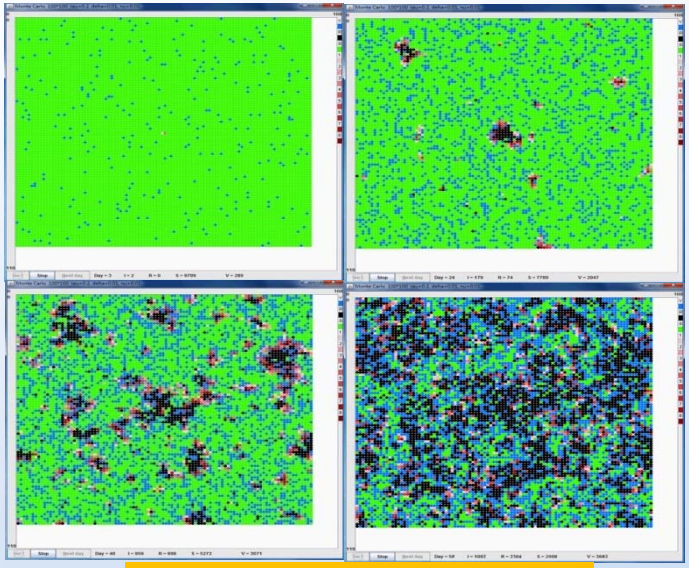

# Lab Module: Infection Simulations on Multicore Processors

- Mote Carlo Simulation
  - Predict the Infection trend
    - Build up the analyical model
  - Sequential -> Parallel
    - OpenMP
    - Pthread
    - Intel TBB
  - Amdahl's Law



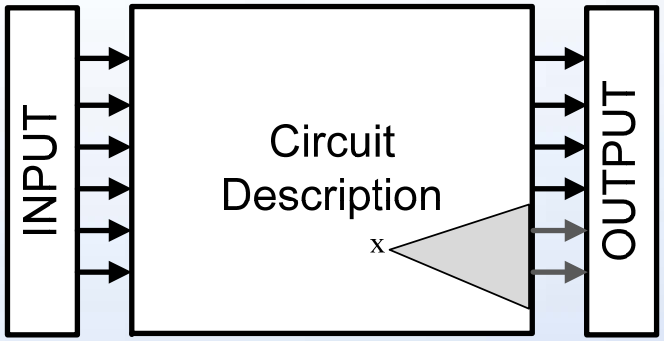

Taiwan Flu Statistics

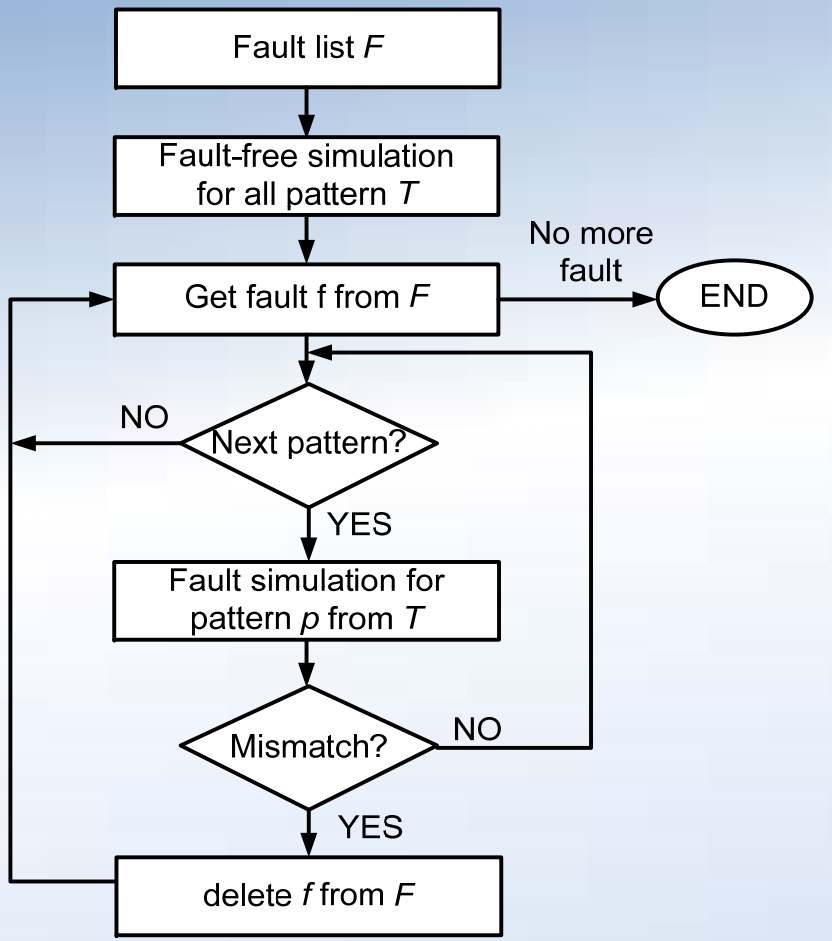


Simulation Results

# Lab Module: Implementation of Parallel Algorithm for Logic Simulation on Switching Network

- HOPE
  - A fault simulator for synchronous sequential circuits
  - Parallel fault simulation techniques
- Performance analysis
  - gprof
  - Identify the bottlenecks

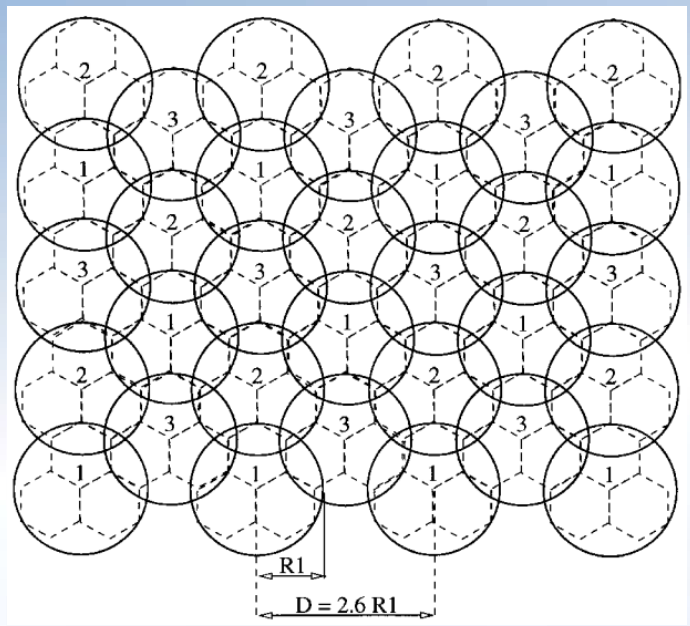



Simple fault simulation flow

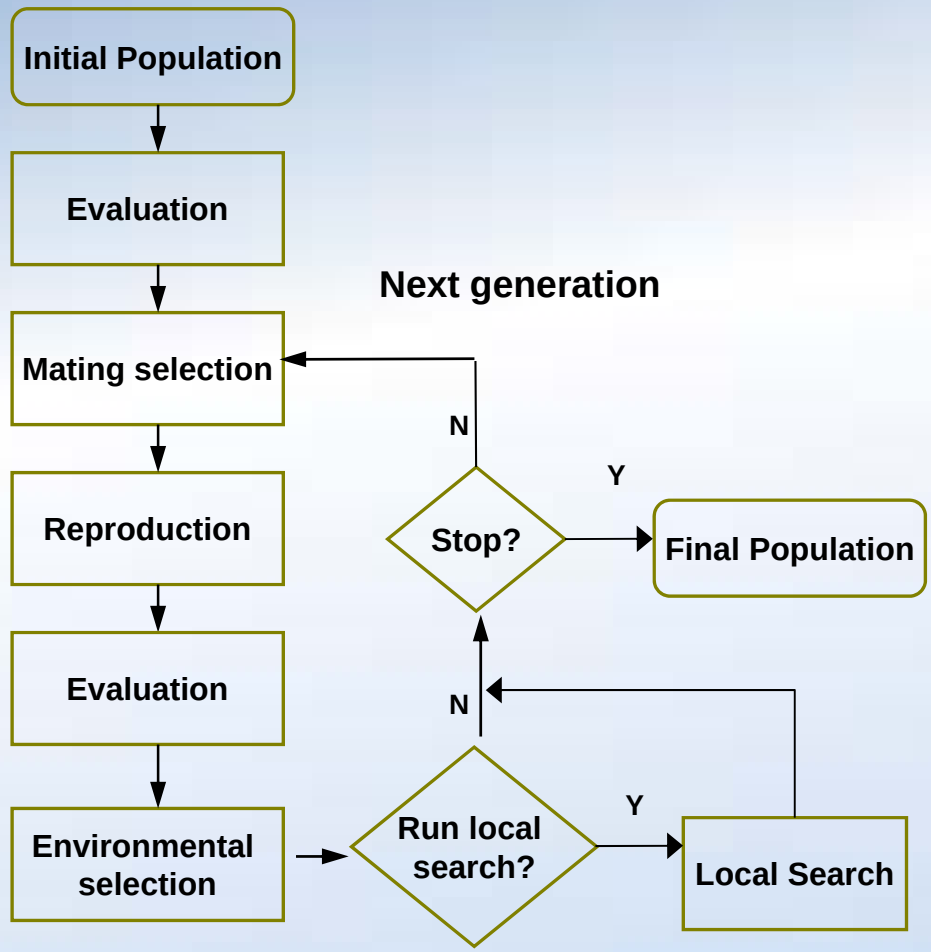
# Lab Module: Parallel Computing for Wireless Simulation Platforms

- Apply the concept of parallel computing to expedite comprehensive wireless system simulations.
- Dynamically changing environments
  - Radio channel characteristics
  - Terminal mobility
- Complex scenarios—Many “multiples”
  - Multiple paths
  - Multiple base stations
  - Multiple users
  - Multiple antenna
  - Multiple networks



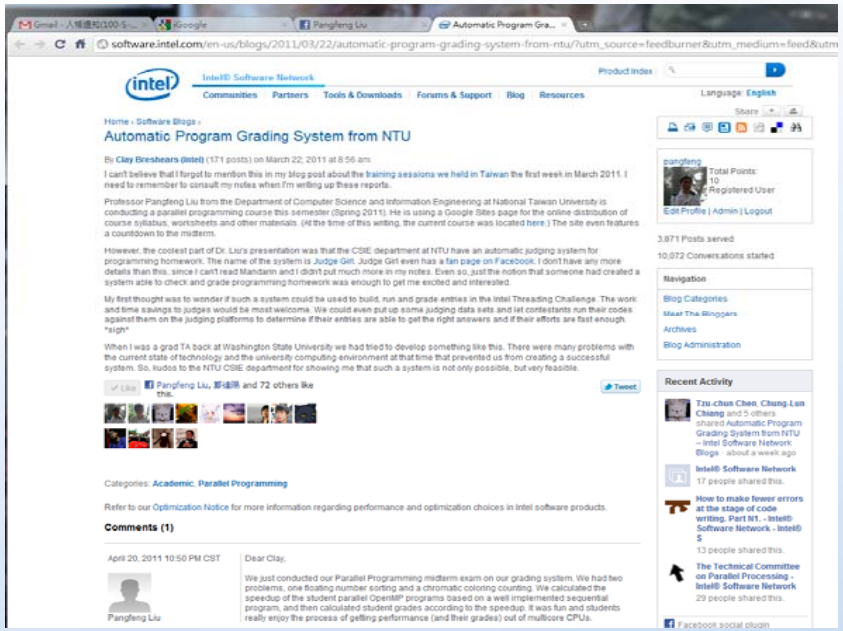
# Lab Module: Parallelizing Memetic Algorithm Using OpenMP

- Metaheuristics
  - Characteristics
    - Category of approximation algorithms for optimization problems.
    - Tool for solving hard optimization problems
      - Scheduling, routing, clustering
    - Iterative processes
    - Memetic Algorithm
    - Genetic Algorithm
  - Components
    - Solution encoding/decoding schemes
    - Neighborhood functions
    - Selection/acceptance criteria
    - Stopping criteria



# Online Judge System

- ACM online judge style system
- Check and grade programming homework
- A platform for in-class worksheets and midterm evaluation
- Parallel processing

# Making Progress

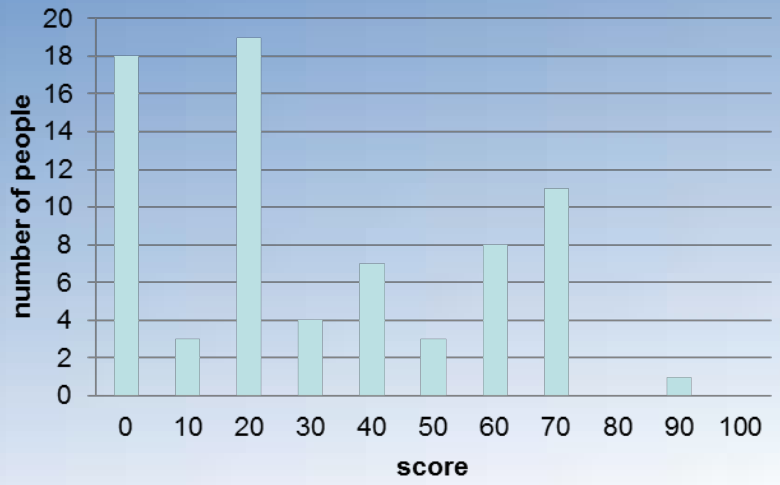
- Understand Students' progress by their submission results of worksheets

| | Worksheet 1 2011/3/26 | Worksheet 2 2011/4/13 | Worksheet3 2011/4/20 |
|---|---|---|---|
| # of students solved the problem | 47/75 | 72/75 | 75/75 |
| Average # of trials till success | 2.2 | 2.8 | 1.7 |

From the course, *principle of parallel programming*

# Midterm Scoring

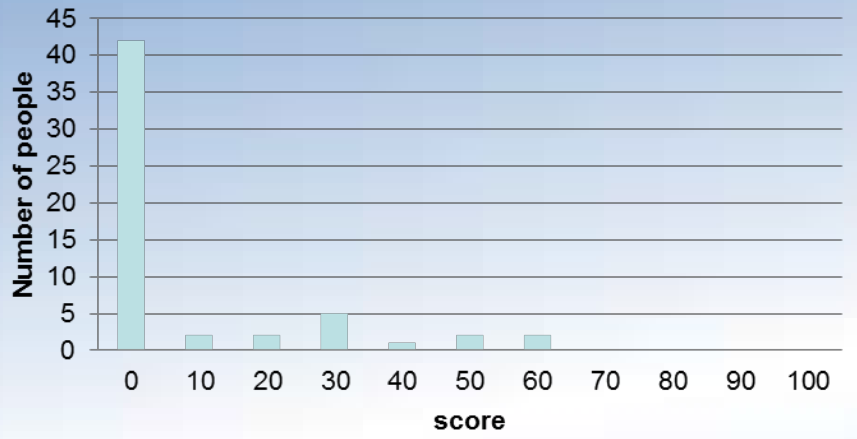
## Floating Number Sorting




| | min | max | average |
|---|---|---|---|
| Time(s) | 0.657065 | 4.767353 | 1.733752 |

| | |
|---|---|
| Number of submission | 330 |
| Number of people | 75 |
| Average number of submissions | 4.4 |

## Chromatic Coloring Counting




| | Min | max | average |
|---|---|---|---|
| Time(s) | 0.264323 | 8.894231 | 1.436465 |

| | |
|---|---|
| Number of submission | 110 |
| Number of people | 75 |
| Average number of submissions | 1.47 |

# Term Projects

- Student propose the proposal for their term project
  - Application
  - Programming Model
- Selected Projects
  - From Neural Networks Implementations to Find out the Advantage of Parallel Programming
  - Automatic Panoramic Image Stitching using Invariant Features
  - Use Cuda to Implement a Path Finding Algorithm
  - Motion Estimation on CUDA
  - Use CUDA to simulate ocean, and render the scene with OpenGL
  - A Social Network Centrality Analyzer
  - Ray-Tracing Parallelization
- Course site
  - https://sites.google.com/site/ntucsiepp2011/home

# Conclusion

- We developed a series of system software and application lab modules on multicore system.
- These lab modules involve various topics of multicore programming techniques and real world multicore applications
- We trained students with capabilities to develop multicore applications.
- The parallel design patterns is introduced to lay the foundations for advanced multicore system research.
- MTL-based lab modules show good promises for further devising innovative system curriculum.