Introduction to CSIM

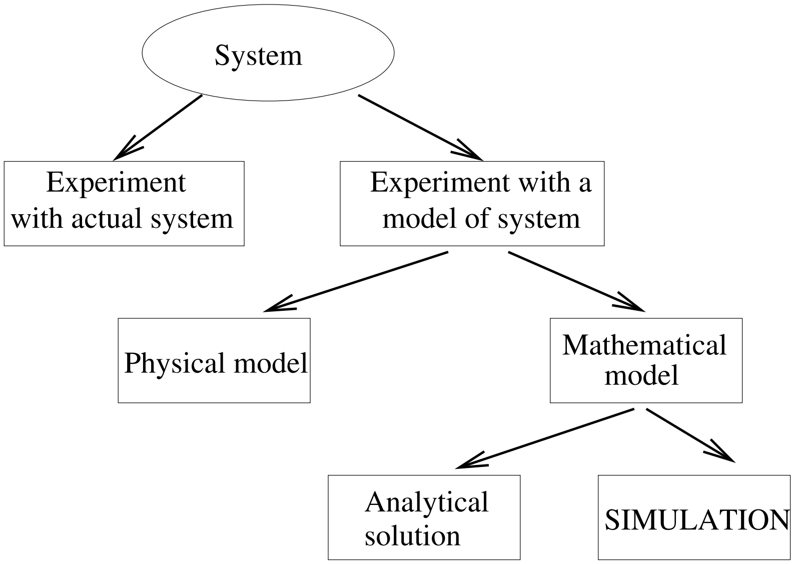
Turgay Korkmaz

Computer Science
University of Texas at San Antonio

Outline

- Simulation (from big picture perspective)
- Steps in Simulation
- World Views in Simulation
- CSIM

Simulation (from big picture perspective)



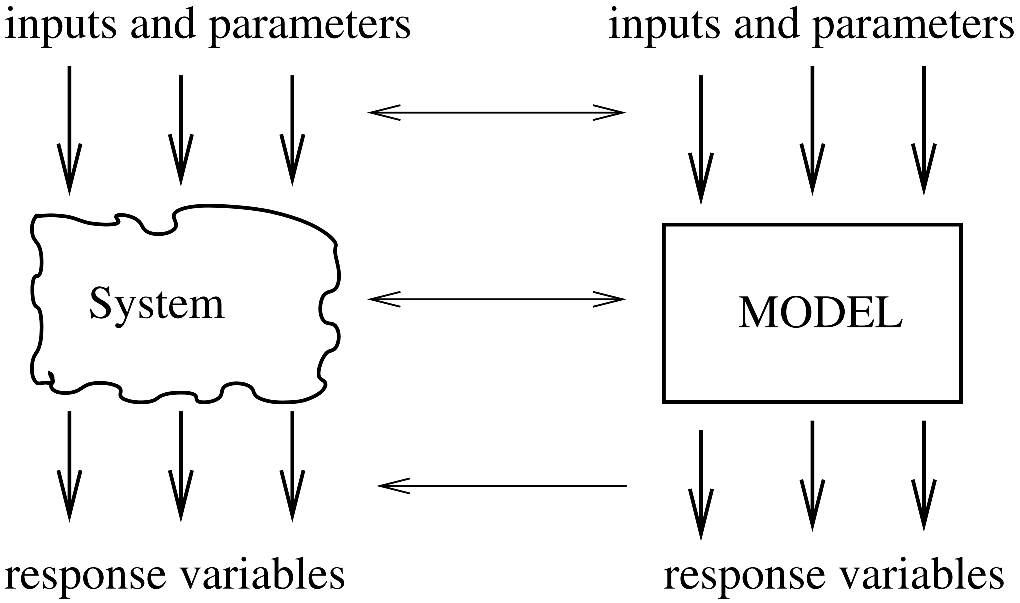
When to simulate?

- Analytical model too complex
- Analytical model cannot be solved
- Validate analytical solutions
- Understand the operation and performance

Simulation models:

- Static vs. Dynamic
- Deterministic vs. Stochastic
- Continues vs. Discrete

Steps in Simulation

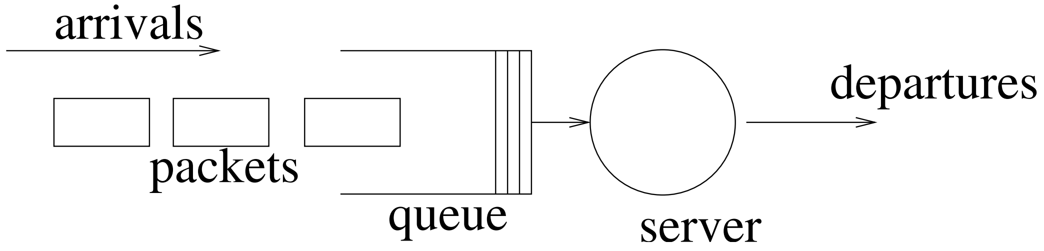


- Problem formulation
- Data collection and Model development
- Computer programming (e.g., CSIM)
- Verification of the program and model
- Design Experiments
- Run simulation (several times)
- Analyze output
- Report results

World View

How we look at the system while modeling it?

- **System:** a set of entities interacting with each other
- **Entities:** components of a system
- **Rules:** (Laws & policies) how the entities behave
- **State:** complete description of the system
- **Event:** a point in time that the state changes



Most commonly used world views

- Event-scheduling (...): *focuses on events and describes what to do when an event occurs*
- Process-oriented (CSIM): *focuses on entities and describes their progresses through the model*

CSIM

CSIM (online at http://www.mesquite.com)

- is a library of routines in C/C++
- creates process-oriented, discrete-event simulation

The structures provided in CSIM are as follows:

- **Processes**: the active entities that request service, wait for events, communicate with others
- **Facilities**: passive entities that are reserved/relased or used by processes
- Storages: resources that can be partially allocated to processes (has a counter and a queue for processes waiting to receive the requested allocation)
- Buffers: resources that can be partially allocated to processes (has a counter and **two** queues: one for processes waiting to receive the requested tokens; one for processes to return tokens)

- **Events**: used to synchronize and control process activities
- **Mailboxes**: used for inter-process communications between processes
- **Random Numbers and Streams**: streams of random numbers
- Data collection structures (Tables, Qtables, Meters, Boxes): used to collect data during the execution of a model
- Process classes: used to segregate statistics for reporting purposes
- Other Features: inspector functions, report functions, debug options

An example in CSIM

```
/* simulate an M/M/1 queue */
#include "csim.h"
FACILITY f;          /* pointer for facility (server) */

void sim()           /* 1st process - named sim    */
{
   create("sim");    /* required create statement  */
   f = facility("server");   /* initialize server  */
   while(simtime()<5000.0) {
     hold(exponential(1.0)); /* inter-arrival time */
     packet();               /* a new packet       */
   }
   report();
   terminate();
}
void packet()
{
   create("packet");        /*  a new process   */
   use(f, exponential(0.5));  /*  use server      */
   terminate();
}
```

Processes in CSIM

The active entities of a system (a C/C++ procedure)

```
// see void packet(){...} in previous page
```

Differences from normal C/C++ procedures

- create() creates a new process (unique id, priority) and immediately returns the control to the invoking process
- CSIM execution supervisor controls the operation of processes
- Many instances of the same process can be "active"
- Processes are in one of four process states: Computing, Ready to start, Holding, Waiting
- A process remains in the Computing state (executing) until it voluntarily takes one of the following actions: `hold(1.0)`, `wait(e)`, `terminate()`
- A process cannot return control to its caller (or return a functional value to its caller);

Resources

Passive entities (used or allocated by processes)

- Facilities represent resources used “one-at-a-time”
  - Single server facility

    ```
    FACILITY f;
    f = facility ("fac");
    use (f, expntl(1.0));

    reserve (f);
    hold(expntl(1.0));
    release(f);
    ```

  - Multi-server facility or an array of single server facilities
  - Service disciplines can be specified (fcfs, priority, preempt-resume)
- Storages and buffers represent resources partially allocated

Process Interactions

- Events used to synchronize process activities
  - Two states: OCC and NOT_OCC

    ```
    EVENT e;
    e = event ("arrive");
    wait(e); timed_wait (e, 100.0);
    queue (EVENT e);    timed_queue (e, 100.0);
    set(e);
    state (e);
    wait_cnt(e); queue_cnt(e);
    ```

- Mailboxes used for inter-process communications

  ```
  MBOX m;
  m = mailbox ("requests");
  send (m, (long) buffer);
  receive (m,(long*) &ptr);
  result=timed_receive(m,(long*) &ptr, 100.0);
  if (result ! = TIMED_OUT) ...
  msg_cnt (m)
  ```

- An array of events/mailboxes can be defined

Random Number Generation

- Single Stream

```
reseed (NIL, 13579);
uniform    (min, max)
triangular (min, max, mode)
....
normal     (mean, stddev)
....
geometric  (prob_success)
```

- Multiple Streams

```
STREAM s;
s = create_stream ();
reseed (s, 24680);

stream_uniform   (s, min, max)
stream_triangular(s, min, max, mode)
....
stream_normal    (s, mean, stddev)
....
stream_geometric (s, prob_success)
```