Deadlock Avoidance Using Null Messages

Chandy-Misra-Bryant Algorithm

Richard M. Fujimoto
Professor

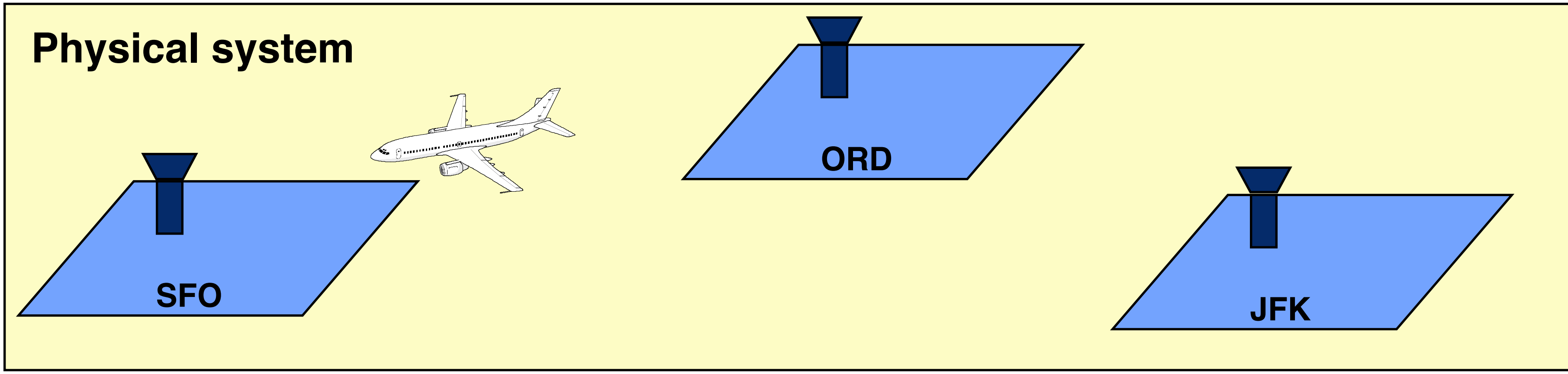
Computational Science and Engineering Division
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0765, USA

http://www.cc.gatech.edu/~fujimoto/

Outline

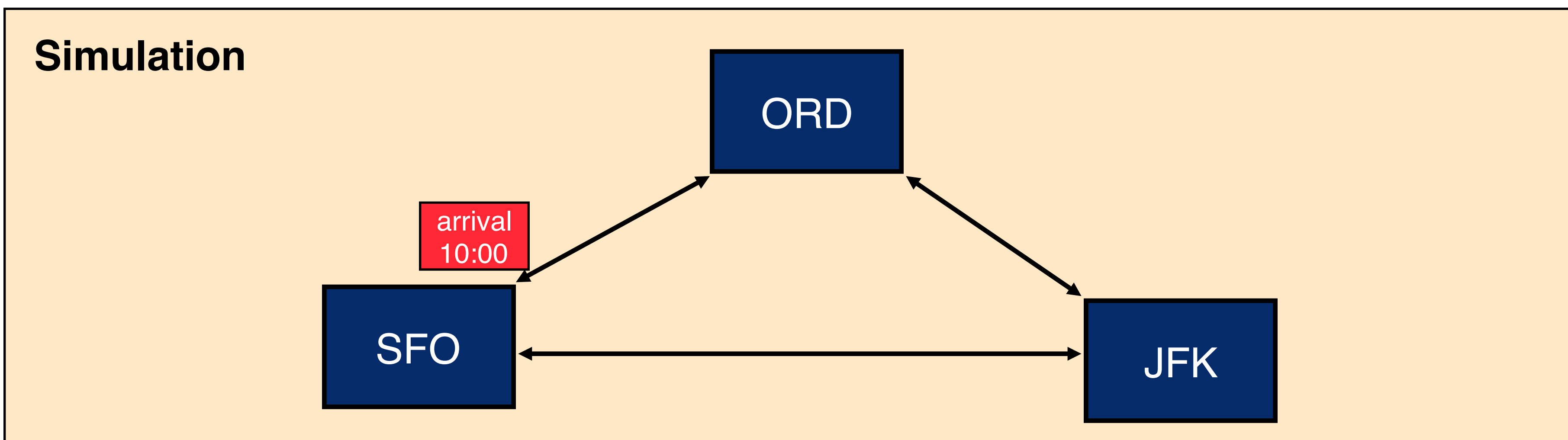
- Example program
- Chandy/Misra/Bryant Null Message Algorithm
  - Ground rules
  - An algorithm that doesn’t work
  - Deadlock avoidance using null messages

Parallel Discrete Event Simulation Example



physical process → logical process

interactions among physical processes → time stamped event (message)



all interactions between LPs must be via messages (no shared state)

LP Simulation Example

- `Now:` current simulation time
- `InTheAir`: number of aircraft landing or waiting to land
- `OnTheGround`: number of landed aircraft
- `RunwayFree`: Boolean, true if runway available

```
Arrival Event:
InTheAir := InTheAir+1;
If (RunwayFree)
   RunwayFree:=FALSE;
   Schedule Landed event (local) @ Now+R;
```

```
Landed Event:
InTheAir:=InTheAir-1;     OnTheGround:=OnTheGround+1;
Schedule Departure event (local) @ Now + G;
If (InTheAir>0) Schedule Landed event (local) @ Now+R;
Else RunwayFree := TRUE;
```

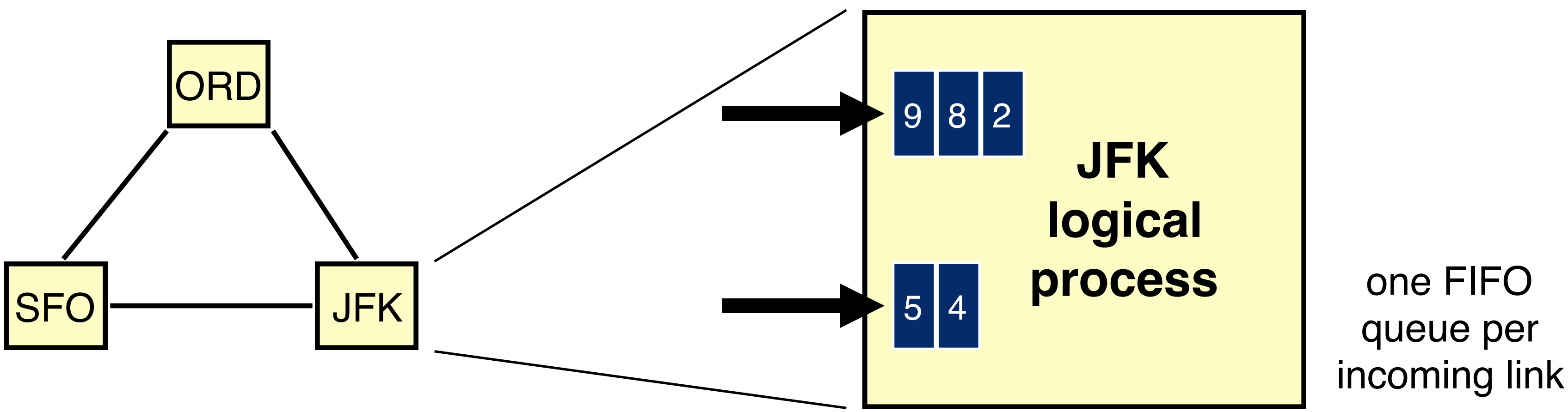
```
Departure Event (D = delay to reach another airport):
OnTheGround := OnTheGround - 1;
Schedule Arrival Event (remote) @ (Now+D) @ another airport
```

Chandy/Misra/Bryant "Null Message" Algorithm

Assumptions

- logical processes (LPs) exchanging time stamped events (messages)
- static network topology, no dynamic creation of LPs
- messages sent on each link are sent in time stamp order
- network provides reliable delivery, preserves order

Observation: The above assumptions imply the time stamp of the last message received on a link is a **lower bound on the time stamp (LBTS)** of subsequent messages received on that link



Goal: Ensure LP processes events in time stamp order

A Simple Conservative Algorithm

Algorithm A (executed by each LP):

Goal: Ensure events are processed in time stamp order:

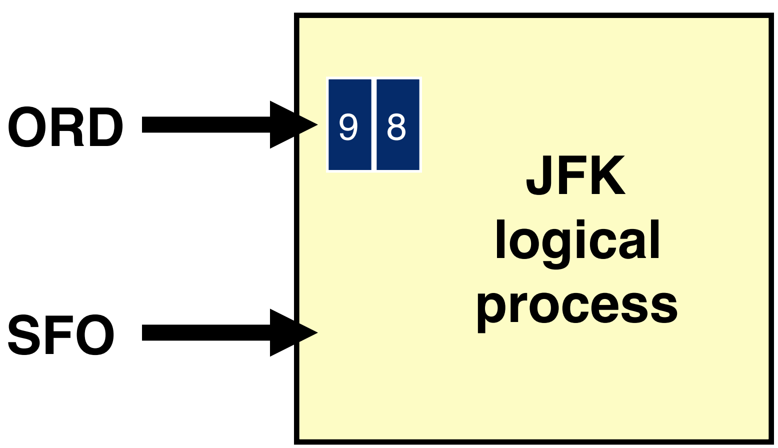
WHILE (simulation is not over)

    wait until each FIFO contains at least one message

    remove smallest time stamped event from its FIFO

    process that event

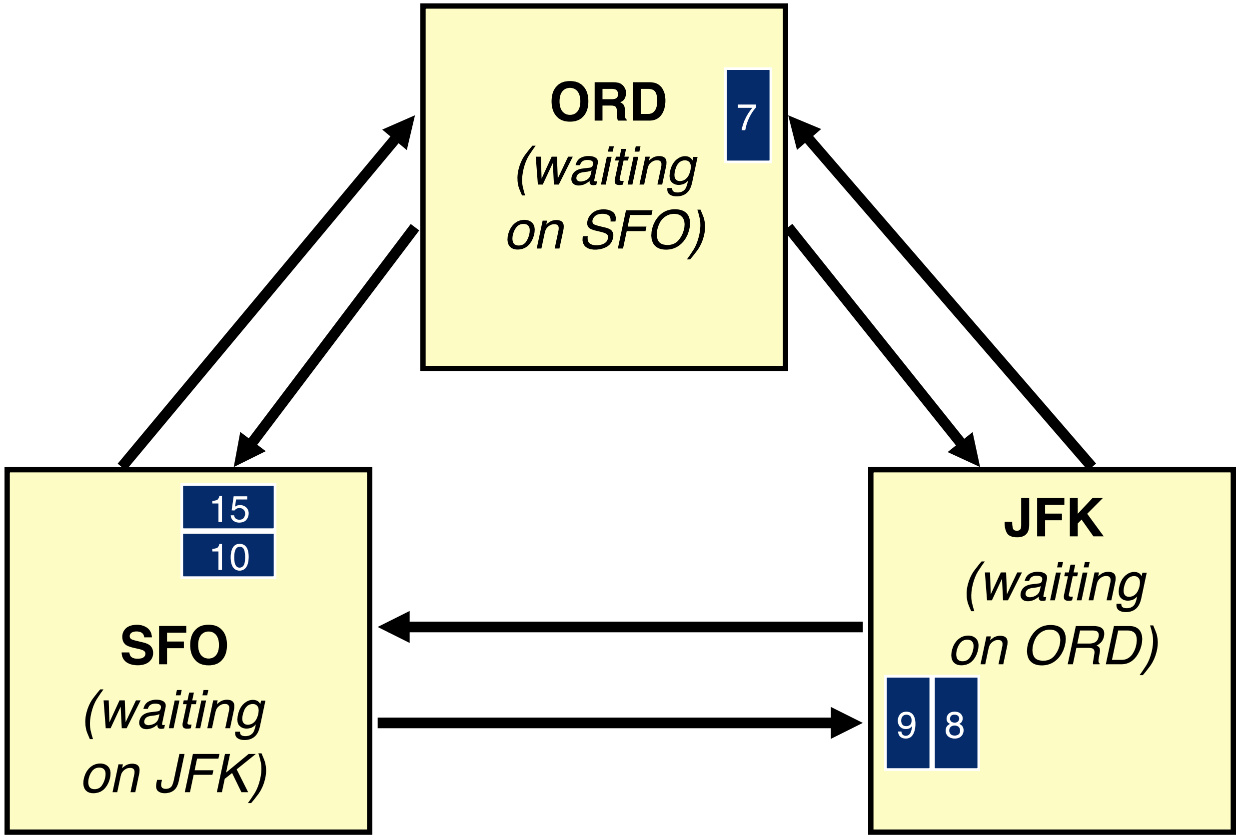
END-LOOP



- process time stamp 2 event
- process time stamp 4 event
- process time stamp 5 event
- wait til message is received from SFO

Observation: Algorithm A is prone to deadlock!

Deadlock Example

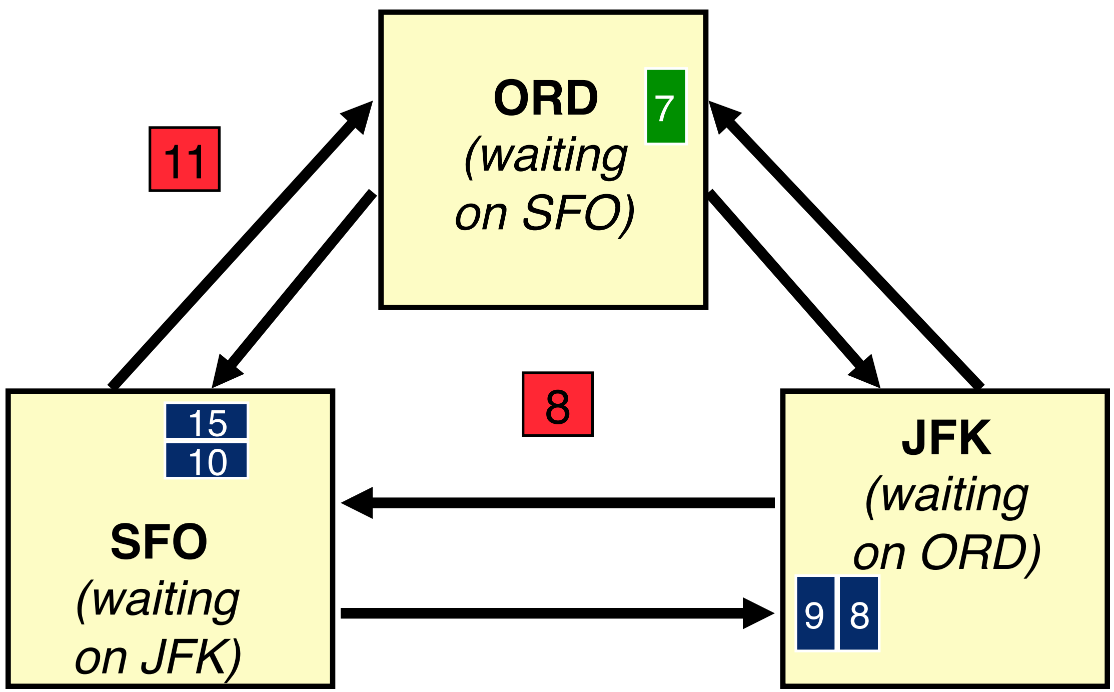


A cycle of LPs forms where each is waiting on the next LP in the cycle.

No LP can advance; the simulation is deadlocked.

Deadlock Avoidance Using Null Messages

Break deadlock: each LP send "null" messages indicating a lower bound on the time stamp of future messages.



Assume minimum delay between airports is 3 units of time

- JFK initially at time 5
- JFK sends null message to SFO with time stamp 8
- SFO sends null message to ORD with time stamp 11
- ORD may now process message with time stamp 7

Deadlock Avoidance Using Null Messages

Null Message Algorithm (executed by each LP):

Goal: Ensure events are processed in time stamp order and avoid deadlock

WHILE (simulation is not over)

wait until each FIFO contains at least one message

remove smallest time stamped event from its FIFO

process that event

send null messages to neighboring LPs with time stamp indicating a lower bound on future messages sent to that LP (current time plus lookahead)

END-LOOP

The null message algorithm relies on a “lookahead” ability.

Summary

- Parallel Discrete Event Simulation
  - Collection of sequential simulators (LPs) possibly running on different processors
  - Logical processes communicating exclusively by exchanging messages
- Chandy/Misra/Bryant Null Message Algorithm
  - Null messages: Lower bound on the time stamp of future messages the LP will send
  - Null messages avoid deadlock