Lecture 7 In Class Message Passing Concurrency Ch. 5

Note Title 1/29/2008

MP models:

- Agents
- Distributed System

Also: MP designs can be made robust rather easily.

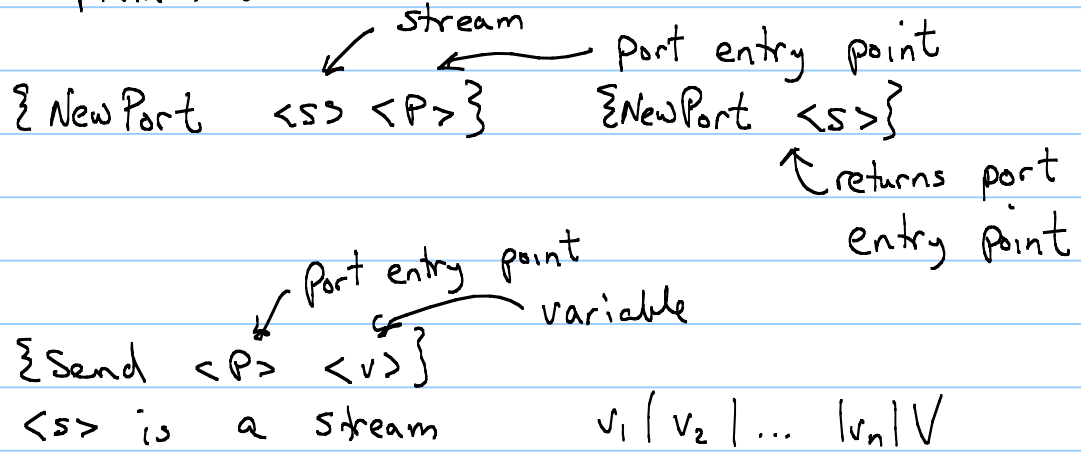
Difference to Decl. Model:

Any # of processes can send to another process and it can read those messages in the order they were sent.

Many different mechanisms called message passing

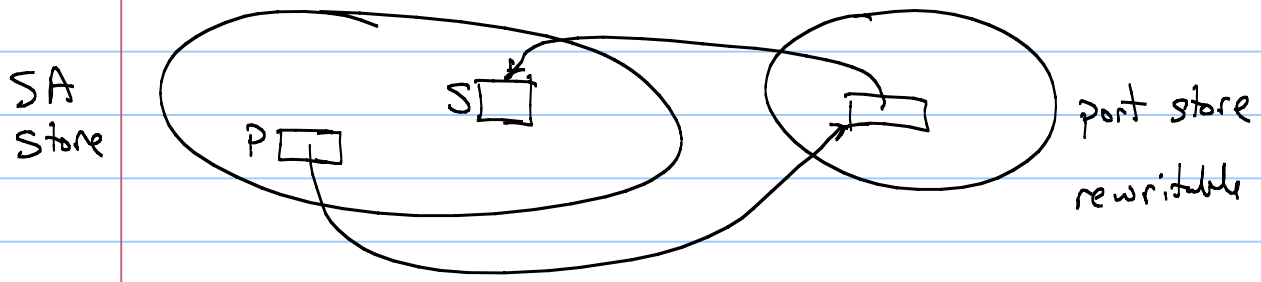
- are messages sent to processes or is there an abstraction called a channel. Channel allows (any of) | (all of) a set of processes to read each message
- are messages passed synchronously or asynchronously.
- Is there a choice operation for receiving or is all non-determinism due to order of sending

Oz primitives

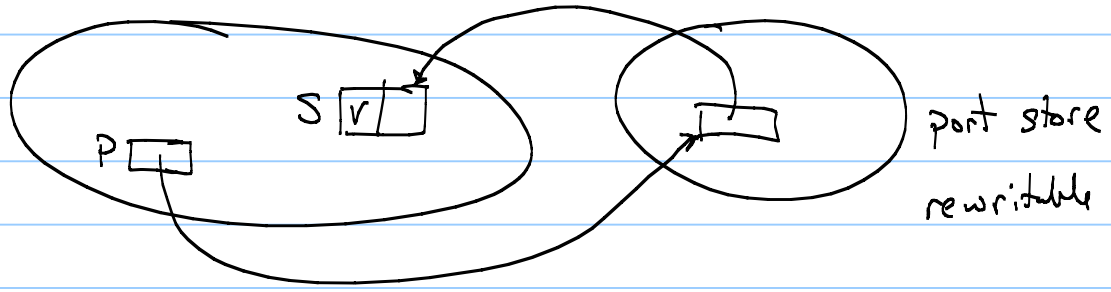


How are ports implemented?

- a stream — SA variable
- Port entry point — SA. variable



{Send P v}



Invariant: port store always points to unbound variable in SA S.

Technicality: this stream has to be special in that only Send can append a new value.

Recall
Stream Objects:
Port Object:

declare $P_1 \ldots P_n$ in ← entry points
local $S_1 \ldots S_n$ in ← These are the streams
{NewPort S_i P_i} ... ← creates n ports
thread {RP $S_1 \ldots S_n$} end
end

Port Object factories

```
fun {NewPortObject Init Fun}
   Sin in
      thread {Fun Sin Init} end
      {NewPort Sin}
end
```

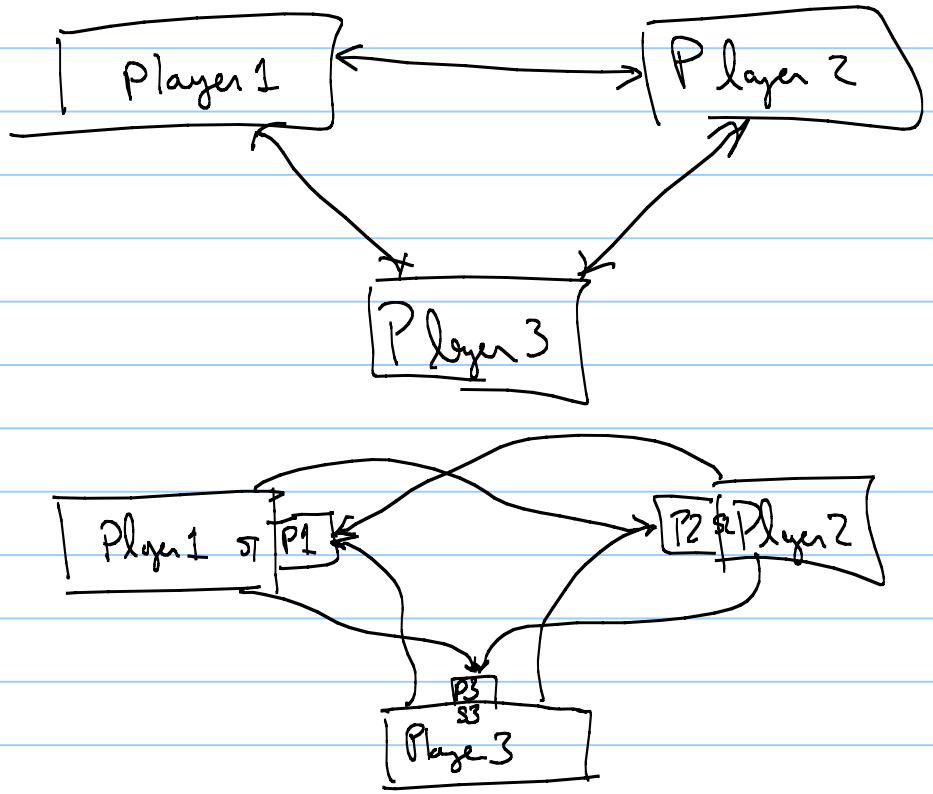
```
fun {F Sin St}
   case Sin of
   S|Sr ....
      St' = ....
      {F Sr St'}
end
```

Sr: remaining input

St': new state

Examples in § 5.2

```
{ForAll S Browse}
for M in S do {Browse M}
for Msg in S do {Proc Msg}
```


Player 1
Player 2
Player 3
Player 1
P1
P2
Player 2
P3
S3
Player 3