Service calls

L. minimize work on nested
L. overlapping I/O & computation

L. multiplex CPU

Multiplexed Serves

Class: 21
Date: 11/5
Two approaches to multithreading

- Static # of threads
- Dynamic
Serve for

End

\[ f(c) \]

\[ \text{return (m-1)} \]

\[ \text{main(2/1 service,} \]

\[ \text{stack} \rightarrow u \text{ threads} \]

\[ f(c) \]
- bag - bread
- good - recognition, can express

dynamic

can't believe

bag - recognition is believed

- read - overdue - low, can - fixed

S+gel
Use any of source code/commands

← on Phrase 7

← in meaning

← on second stage

mutliple things ≠ more cond. in

Note
Implementing paths

I machine local paths in kernel

P_i

P_0

 Await

- Probes queues

\text{P}(s_i) \geq c(s_i) \quad V(s_i) = s_i
transmit

\[ \text{Send } \langle \text{IP}, \text{port} \rangle \leadsto \]

@IP

local port

\[ \text{ Manning } \rightarrow \langle \text{IP}, \text{port} \rangle \]
need a server to get incoming msgs.

\[
\text{\texttt{netmsg server}} \\
\text{\texttt{loop}} \\
\text{\texttt{\{ \texttt{rcv (msg);} \}}} \\
\text{\texttt{msg \rightarrow part}} \\
\text{\texttt{\rightarrow \ldots \text{act on incoming msg}}}
\]
if (IP == face) {
    face = new_face(part, msg);
}

if (IP == face) {
    face = send(part, msg);
}

{ msg = { "send", port, msg } } = op_code

{ msg = { "recv", port, msg } } = get_IP(msg)
proxy port & local machine

\{ delegeport (tp) \}

recv (tp, msg) \rightarrow local

not seen (msg, mp:tp, port, msg)

f

\( tp = \text{conceal}(msg) \)

not seen (msg, mp:tp, port, msg)

f

if (mp + local)

IP:port

recv (port, msg)
No remote server used in MacCH
Local

send (port, msg)

receive (msg) loop

recv (port) -> recast (IP, port, msg)

search (IP, port)

TCP/IP
Distributed Mutual Exclusion

Critical Section

- No two processes can be in its point in time.

P1

P2

P3
P2P

L > No Service

 Archiebergs solution

L use 1 Service

Cemahnaa solution