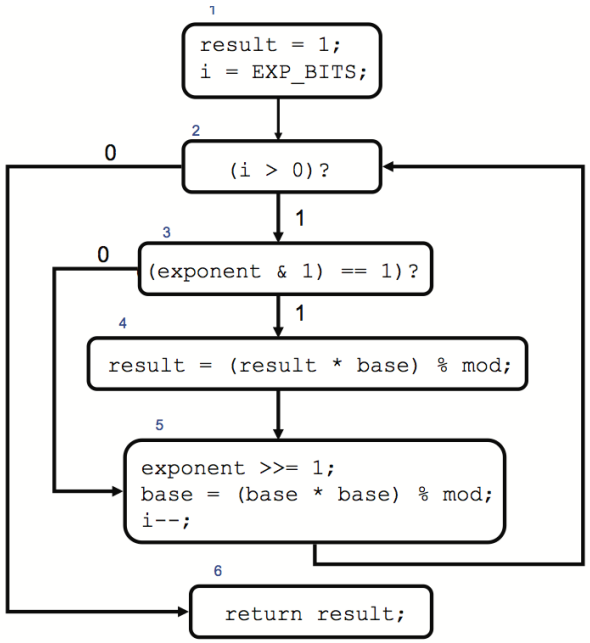
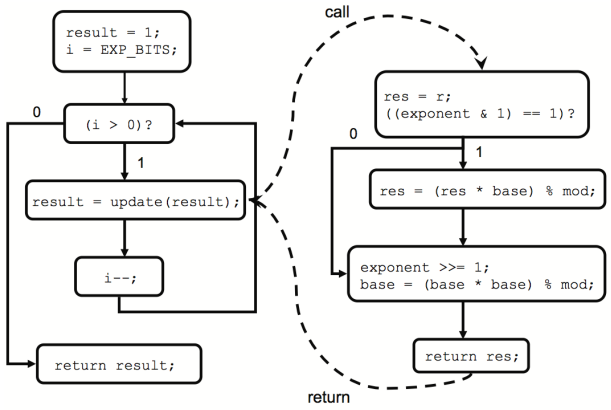
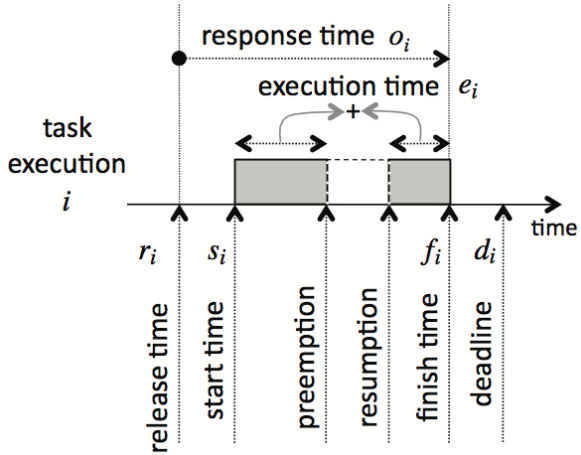
EECS 149 MT2 Note Sheet

```
1  #define EXP_BITS 32
2
3  typedef unsigned int UI;
4
5  UI modexp(UI base, UI exponent, UI mod) {
6    int i;
7    UI result = 1;
8
9    i = EXP_BITS;
10   while(i > 0) {
11     if ((exponent & 1) == 1) {
12       result = (result * base) % mod;
13     }
14     exponent >>= 1;
15     base = (base * base) % mod;
16     i--;
17   }
18   return result;
19 }
```

A **basic block** is a sequence of consecutive program statements in which the flow of control enters only at the beginning of this sequence and leaves only at the end, without halting or the possibility of branching except at the end.







Liu and Layland (1973) showed that a simple preemptive scheduling strategy called **rate monotonic** (**RM**) scheduling is optimal with respect to feasibility among fixed priority schedulers for the above task model. This scheduling strategy gives higher priority to a task with a smaller period.

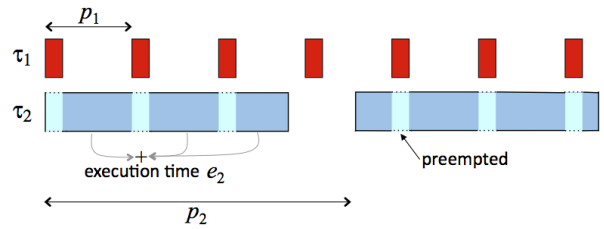
Go to page 499

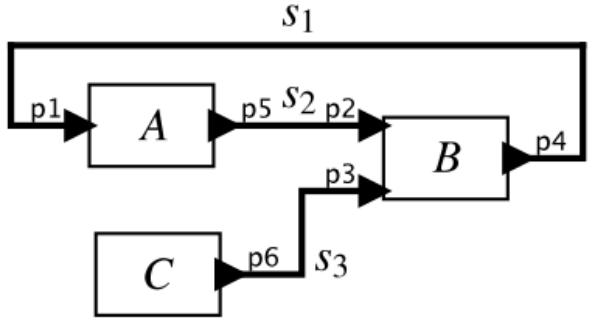
$$\mu = \sum_{i=1}^{n} \frac{e_i}{p_i}.$$

If $\mu = 1$, then the processor is busy 100% of the time. So clearly, if $\mu > 1$ for any task set, then that task set has no feasible schedule. Liu and Layland (1973) show that if μ is less than or equal to a **utilization bound** given by

$$\mu \leq n(2^{1/n} - 1), \tag{12.2}$$

then the RM schedule is feasible.





Actor Networks as a System of Equations

In a model, if the actors are determinate, then each actor is a function that maps input signals to output signals. For example, in Figure 6.1(a), actor A may be a function relating signals s_1 and s_2 as follows,

$$s_2 = A(s_1).$$

Similarly, actor B relates three signals by

$$s_1 = B(s_2, s_3).$$

Actor C is a bit more subtle, since it has no input ports. How can it be a function? What is the domain of the function? If the actor is determinate, then its output signal s_3 is a constant signal. The function C needs to be a constant function, one that yields the same output for every input. A simple way to ensure this is to define C so that its domain is a singleton set (a set with only one element). Let $\{\emptyset\}$ be the singleton set, so C can only be applied to $\emptyset$. The function C is then given by

$$C(\emptyset) = s_3.$$

Hence, Figure 6.1(a) gives a system of equations

$$\begin{aligned} s_1 &= B(s_2, s_3) \\ s_2 &= A(s_1) \\ s_3 &= C(\emptyset). \end{aligned}$$

The semantics of such a model, therefore, is a solution to such a system of equations. This can be represented compactly using the function F in Figure 6.1(d), which is

$$F(s_1, s_2, s_3) = (B(s_2, s_3), A(s_1), C(\emptyset)).$$

All actors in Figure 6.1(a) have output ports; if we had an actor with no output port, then we could similarly define it as a function whose codomain is $\{\emptyset\}$. The output of such function is $\emptyset$ for all inputs.

6.2.2 Well-Formed and Ill-Formed Models

There are two potential problems that may occur when seeking a fixed point. First, there may be no fixed point. Second, there may be more than one fixed point. If either case occurs in a reachable state, we call the system **ill formed**. Otherwise, it is **well formed**.

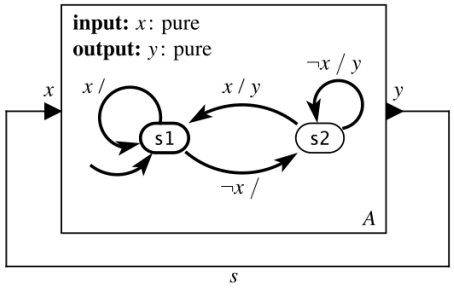


Figure 6.2: A simple well-formed feedback model.

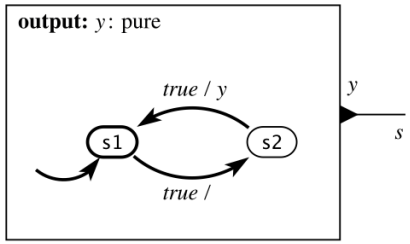


Figure 6.3: The semantics of the model in Figure 6.2.

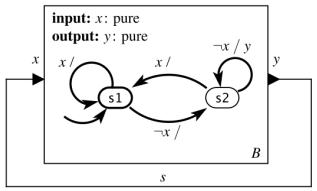


Figure 6.4: An ill-formed feedback model that has no fixed point in state s2.

Example 6.3: For the example in Figure 6.2, if the machine is in state s1, then $a_{s1}(x) = absent$ for all $x \in \{present, absent\}$.

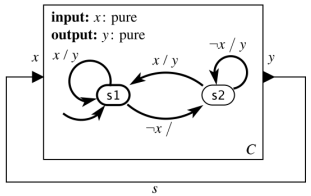


Figure 6.5: An ill-formed feedback model that has more than one fixed point in state s1.

Example 6.4: Consider machine B shown in Figure 6.4. In state s1, we get the unique fixed point $s(n) = absent$. In state s2, however, there is no fixed point. If we attempt to choose $s(n) = present$, then the machine will transition to s1 and its output will be *absent*. But the output has to be the same as the input, and the input is *present*, so we get a contradiction. A similar contradiction occurs if we attempt to choose $s(n) = absent$.

Since state s2 is reachable, this feedback model is ill formed.

Example 6.5: Consider machine C shown in Figure 6.5. In state s1, both $s(n) = absent$ and $s(n) = present$ are fixed points. Either choice is valid. Since state s1 is reachable, this feedback model is ill formed.

1. Start with $s(n)$ *unknown*.
2. Determine as much as you can about $f_i(s(n))$, where f_i is the firing function in state i. Note that in this step, you should use only the firing function, which is given by the state machine; you should not use knowledge of how the state machine is connected on the outside.
3. Repeat step 2 until all values in $s(n)$ become known (whether they are present and what their values are), or until no more progress can be made.
4. If unknown values remain, then reject the model.