CSC 470F -- Computer System Modelling and Analysis

Advanced MAP Commands

These advanced MAP commands go beyond those included in the documentation in the Course Notes from the bookstore. For full detailed documentation of any of these commands, see the Map Reference Manual (/local/doc/map/MAPRefManual.txt on CDF).

1. Priority Scheduling

Rather than scheduling customers normally (i.e., FCFS or Round-Robin), a center can be made to do (preemptive) priority scheduling, based on a priority level associated with each customer class.

The relevant commands are:

```
schedule normal        - causes normal scheduling at the current center
schedule priority      - causes priority scheduling at the current center
schedule delay         - causes pure delay scheduling at the current center
plevel <expr>          - sets the priority level of the current class
resetpri <expr>        - changes the priority level of the current class to a new value
```

2. Limited Memory Space

When memory space is limited, arriving jobs must first enter a memory queue, and wait there until there is enough memory space available to activate them. In MAP, classes are assigned to memory *domains*, and each domain has a limit on the number of customers that can be active in that domain at a time. All classes can be assigned to a single shared domain, or each class can be assigned to its own domain. Admission to the domain can be either "*fcfs*" or "*priority*".

The relevant commands are:

```
domain <name>          - creates or names a domain
limited_mpl <expr>     - sets the number of customers allowed in the domain
indomain <name>        - places the current class into the named domain
outdomain              - removes the current class from its domain
memsize <expr>         - indicates the size of main memory
admit <discipline>     - chooses fcfs or priority entry to the current domain
```

3. Moving from Validation Phase to Projection Phase

In advanced MAP models, it is necessary to use a special command, called SETSYSTEM, to signify passage from the validation phase to the projection phase of model building and use. This is necessary in order to allow MAP to know whether the purpose of a particular change to the model is (i) to modify the model to achieve better validation, or (ii) to predict performance that would occur in some interesting scenario about the future. Remember that SETSYSTEM must be OFF during model validation, but must be set to ON before attempting projections.

The relevant command is:

```
setsystem on           - invoke automatic loading conversion by MAP
```

4. Type Definitions for Processors and Disks

Using the type definition command, various characteristics of particular models of processors and disks, etc. can be recorded. Then, by giving a particular center a model name, it inherits all the characteristics of that model. Some toy devices are reflected in the file ~kcs/map-files/INITDEVS.mapexec. This exec is automatically invoked when MAP is started, and forms the set of "built-in" models that you see in response to a "model" command in MAP. The "tdefine" command allows you to either see or modify the characteristics of any known model type. (To see type definitions for some real (but very old) devices, look in the file ~kcs/map-files/INITDEVS.old.)

This feature makes it easier to examine the effect of replacing a device by another one with different characteristics. The steps are: (1) associate a model type with each center of interest, (2) use the command "setsystem on" to indicate that the model is fully defined, and (3) change the model type of a center or move some service demand from one center to another. MAP will automatically change the loadings according to the known characteristics of the model types.

The relevant commands are:

```
tdefine <name> <attrs> - examine or modify a type definition
model <type>           - specify or alter the model type of the current center
move <f> <c1> <c2>     - move fraction f of the service demand at c1 to c2
diskmod <model>        - exec to change the model of the disks
                              (presumed to be all centers except the first one).
```

Some Map commands have no effect on model results when SETSYSTEM is OFF, but do have an effect when SETSYSTEM is on. For example, when SETSYSTEM is OFF, the Map command:

```
center PROC model FLAME
```

will be interpretted as a correction to prior information about the actual current model type of the processor. Once SETSYSTEM is ON, however, the same command is interpretted as asking Map to project the performance of the system with the current SPARK processor upgraded to model type FLAME. To do that, Map will modify the PROC service demand for each class to reflect the fact that the service demand will be lower on the faster processor. These changes can be observed using the command, "list loading".

5. I/O Subsystem Contention

I/O operations require not just a disk, but also other components such as channels, controllers, etc. Since all the components are required for use simultaneously, these other components can’t simply be represented as other centers. In a MAP model it is possible to identify all these components and show how they are interconnected. This makes it possible to model situations where the performance bottleneck is a channel or controller rather than a processor or disk. (For I/O subsystem modelling, it is again necessary to "setsystem on" after the definition of the model is complete, but before trying out the modifications to the system.) A full I/O path for a disk center includes a string, a controller, and a channel. Typically, there are multiple controllers per channel, multiple strings per controller, and multiple disks per string.

The relevant commands are:

```
channel <name>         - create or name a channel
onchannel <name>       - the current controller is connected to the named channel
offchannel             - the current controller is disconnected from the named channel
controller <name>      - create or name a controller
oncontroller <name>    - the current string is connected to the named controller
offcontroller          - the current string is disconnected from the named controller
string <name>          - create or name a string
onstring <name>        - the current center is connected to the named string
offstring              - the current center is disconnected from the named string
```