CMPSCI 377 Operating Systems **Fall 2009**

Lecture 6

Lecturer: Emery Berger *Scribe: Sandeep P, Li Mi*

6.1 Amdahl's Law for multi-processors

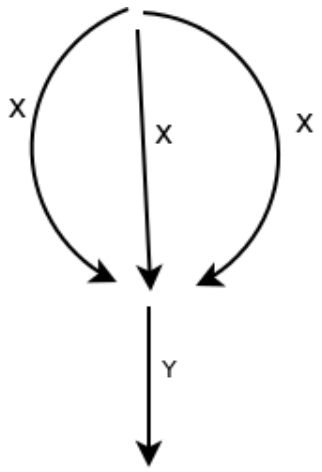

Figure 6.1: Paths in a program

Let x be the time taken by the parallel part of the program .
Let y be the time taken by the sequential part of the program .
Time taken on a uniprocessor system = 3x+y
Time taken on a $\infty processor system = x + y$
$\frac{T_1}{T_\infty} = \frac{3x+y}{x+y}$

The speedup achieved by parallel processing is bounded by the sequential part of the program .

$$T_p = Time\ on\ p\ processors \tag{6.1}$$
$$T_1 = Time\ on\ 1\ processors \tag{6.2}$$
$$T_\infty = Time\ on\ p\ processors \tag{6.3}$$
$$T_p = \frac{T_1}{p} + \mathcal{O}(T_\infty) \tag{6.4}$$

As the serial part increases speed up is not achieved due to contention

6.2 Pure Private Heap

If the entire heap is controlled by a single lock . Memory allocation becomes sequential as all threads contend for the same lock .An alternative to this is every thread having its own private heap . If the objects are

moved between threads . The private of one thread keeps running out of memory and the thread which recieves memory from this thread keeps gaining memory . This results in an unbounded worst case .

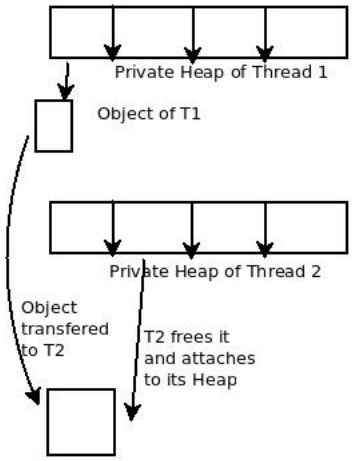

Figure 6.2: Worst Case Private Heap

6.3 Fragmentation

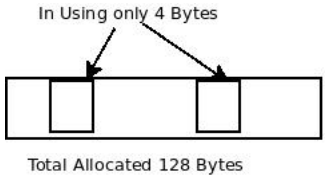

Figure 6.3: Fragmentation

Allocated memory is 128 bytes while the actual memory in use is only 8 bytes , therefore fragmentation is 128/8 .
If all objects are of the same size , then it is guaranteed perfect fit.

$$Fragmentation = (\frac{A}{U}) \tag{6.5}$$
$$A = D\ memory\ in\ bytes \tag{6.6}$$
$$U = memory\ in\ use \tag{6.7}$$
$$\tag{6.8}$$

The best memory allocator will suffer a fragmentation of

$$\mathcal{O}(log(\frac{M}{m})) \tag{6.9}$$
$$M = max\ size\ of\ allocated\ object \tag{6.10}$$
$$m = max\ size\ osf\ allocated\ object \tag{6.11}$$
$$\tag{6.12}$$

6.4 parallel Allocator

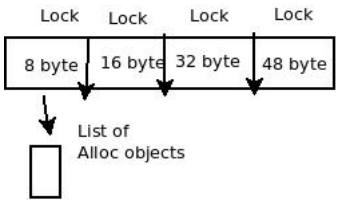

Figure 6.4: Parallel Allocator

Other alternatives to prevent fragmentation.
Divide memory into chunks of 8,16,32.. bytes.
Each chunk of memory has a lock .