Bounding resource use

To bound a program's use of a limited resource - like memory - have goroutines synchronize their use of that resource using a buffered channel (i.e., use the channel as a semaphore):

```go
const (
    AvailableMemory         = 10 << 20 // 10 MB
    AverageMemoryPerRequest = 10 << 10 // 10 KB
    MaxOutstanding          = AvailableMemory / AverageMemoryPerRequest
)

var sem = make(chan int, MaxOutstanding)

func Serve(queue chan *Request) {
    for {
        sem <- 1 // Block until there's capacity to process a request.
        req := <-queue
        go handle(req) // Don't wait for handle to finish.
    }
}

func handle(r *Request) {
    process(r) // May take a long time & use a lot of memory or CPU
    <-sem      // Done; enable next request to run.
}
```

References

Effective Go's discussion of channels: https://go.dev/doc/effective_go#channels