module Manager
    op getTask(result double left, right);
    op putResult(double area);
body Manager
    process manager {
        double a, b; # interval to integrate
        int numIntervals; # number of intervals to use
        double width = (b-a)/numIntervals;
        double x = a, totalArea = 0.0;
        int tasksDone = 0;
        while (tasksDone < numIntervals) {
            in getTask(left, right) st x < b ->
                left = x; x += width; right = x;
            [] putResult(area) ->
                totalArea += area;
                tasksDone++;
            ni
            print the result totalArea;
        }
    }
end Manager

double f() { ... } # function to integrate
double quad(...) { ... } # adaptive quad function

process worker[w = 1 to numWorkers] {
    double left, right, area = 0.0;
    double fleft, fright, lrarea;
    while (true) {
        call getTask(left, right);
        fleft = f(left); fright = f(right);
        lrarea = (fleft + fright) * (right - left) / 2;
        # calculate area recursively as shown in Section 1.5
        area = quad(left, right, fleft, fright, lrarea);
        send putResult(area);
    }
}

Figure 9.2  Adaptive quadrature using manager/workers paradigm.