Two kinds of parallel problems

A simulation run takes about a second.

Want to do 1000's of them,
varying a dozen or so parameters.

A simulation of a large network takes hours.

Want to spread the problem over several machines,
each machine handling a subset of the neurons in the network

Serial

```
s = 0
for i = 1, 10 {
    s += f(i)
}
```

Parallel

```
s = 0
for i = 1, 10 {
    pc.submit("f", i)
}
while (pc.working) {
    s += pc.retval
}
```

Goals

Keep all the machines as busy as possible.

If there is only one machine the parallel program should run as fast as the serial program.

Things asked for earlier tend to get done earlier.

Assumptions

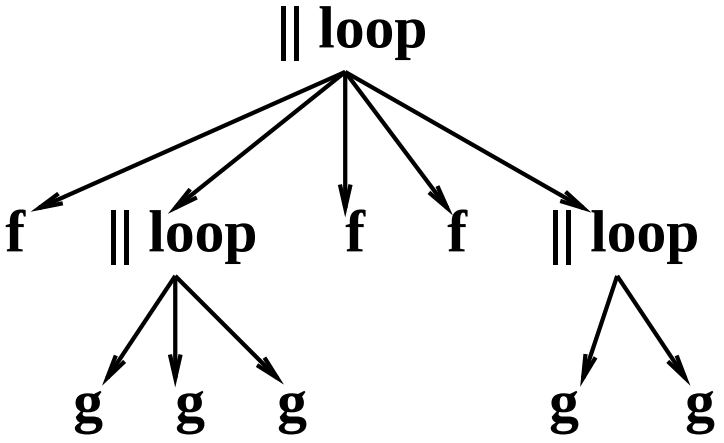
Workstation cluster – 1, 3, 15, 100 machines.

Wide variety of machine speeds.

Sending a byte is 100's (1000's ?) of times slower than executing a hoc statement.

Domain

Very coarse grain parallelization.



NEURON's offering

A bulletin board
. . . on top of PVM.

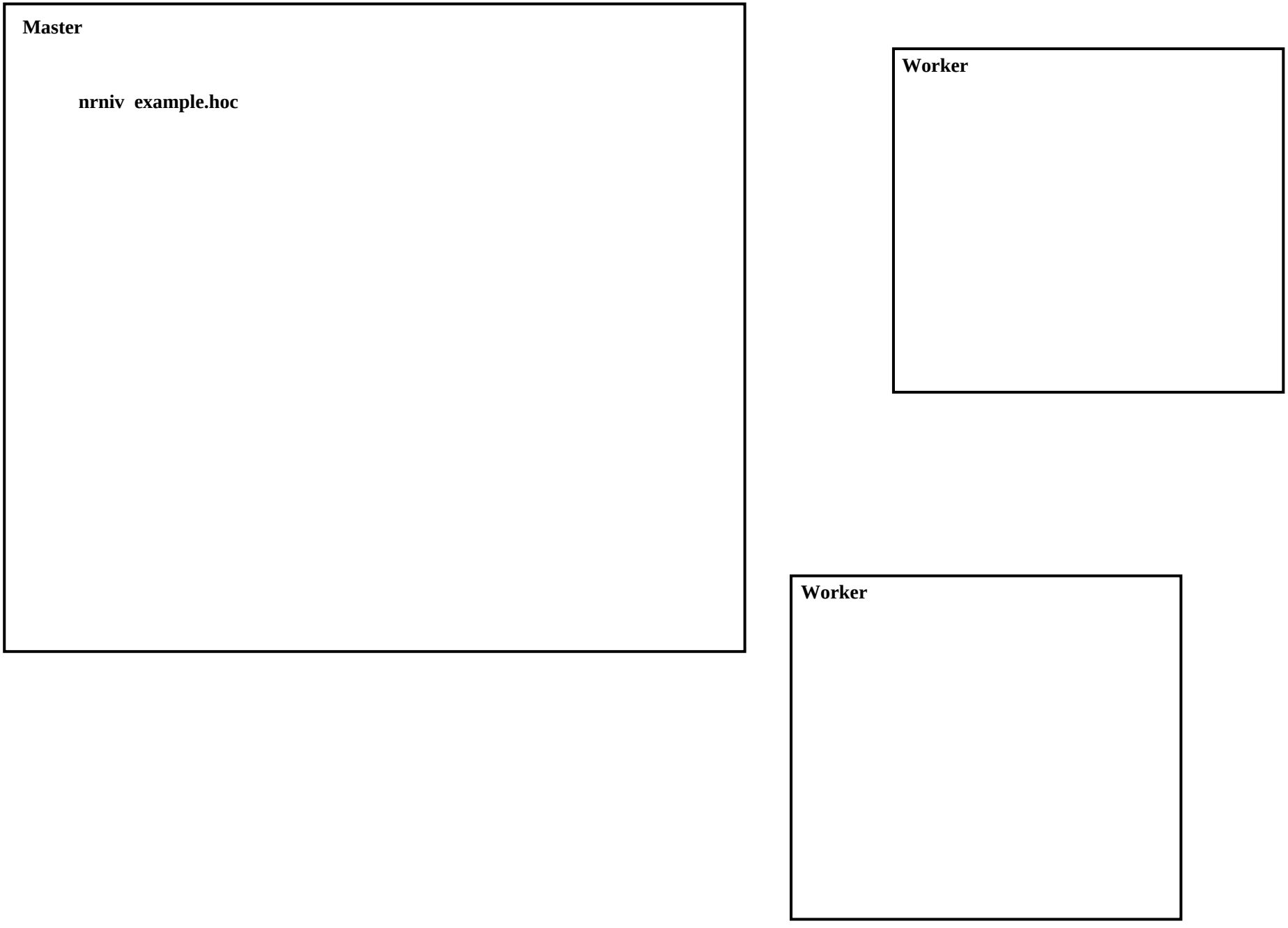
Launching a parallel program

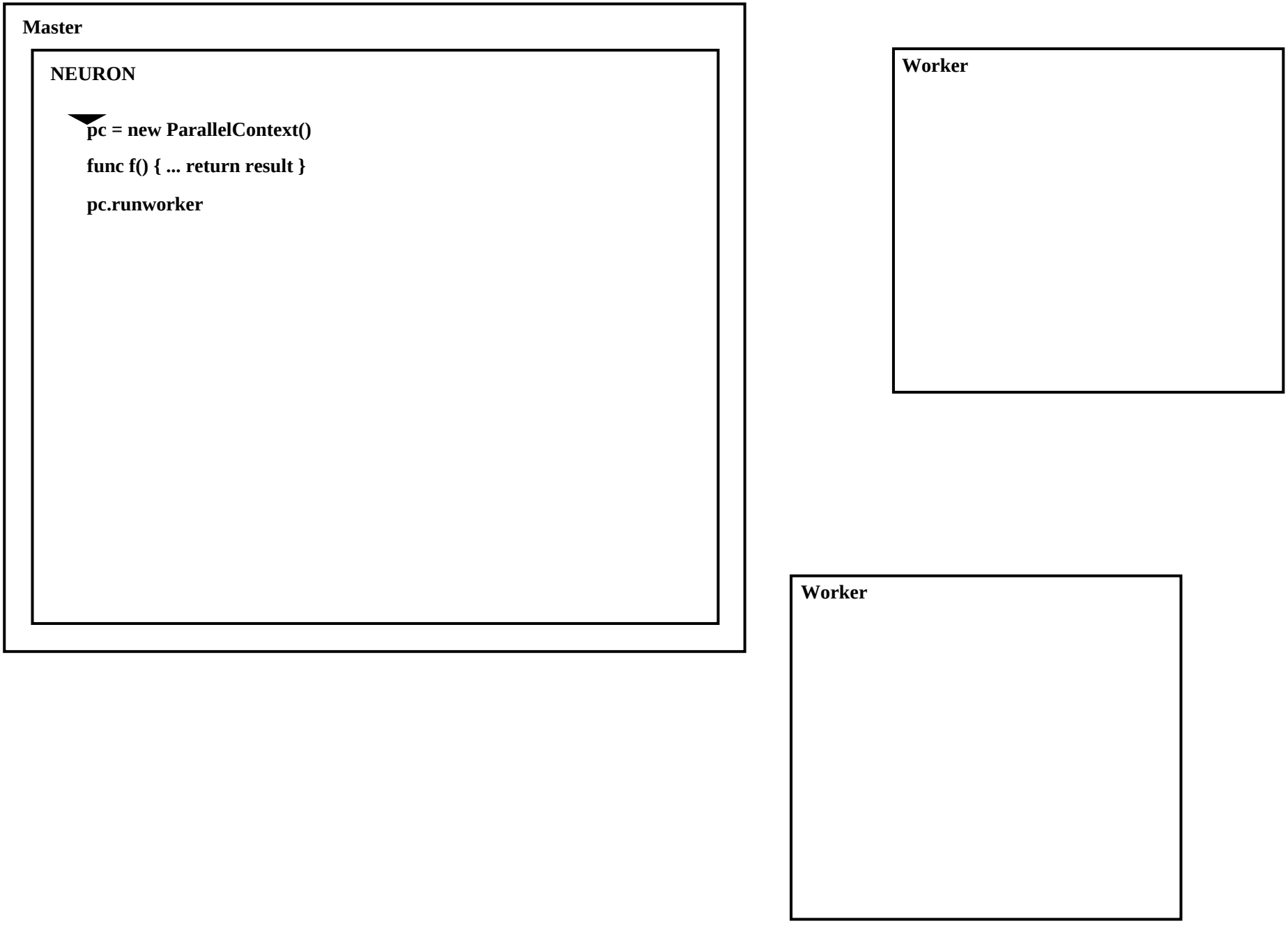
```
objref pc
pc = new ParallelContext()

  // setup which is exactly
  // the same on every machine
  // i.e. declaration of all
  // functions, procedures,
  // setup of neurons

pc.runworker

  // the master scatters tasks
  // onto the bulletin board
  // and gathers results

pc.done
```

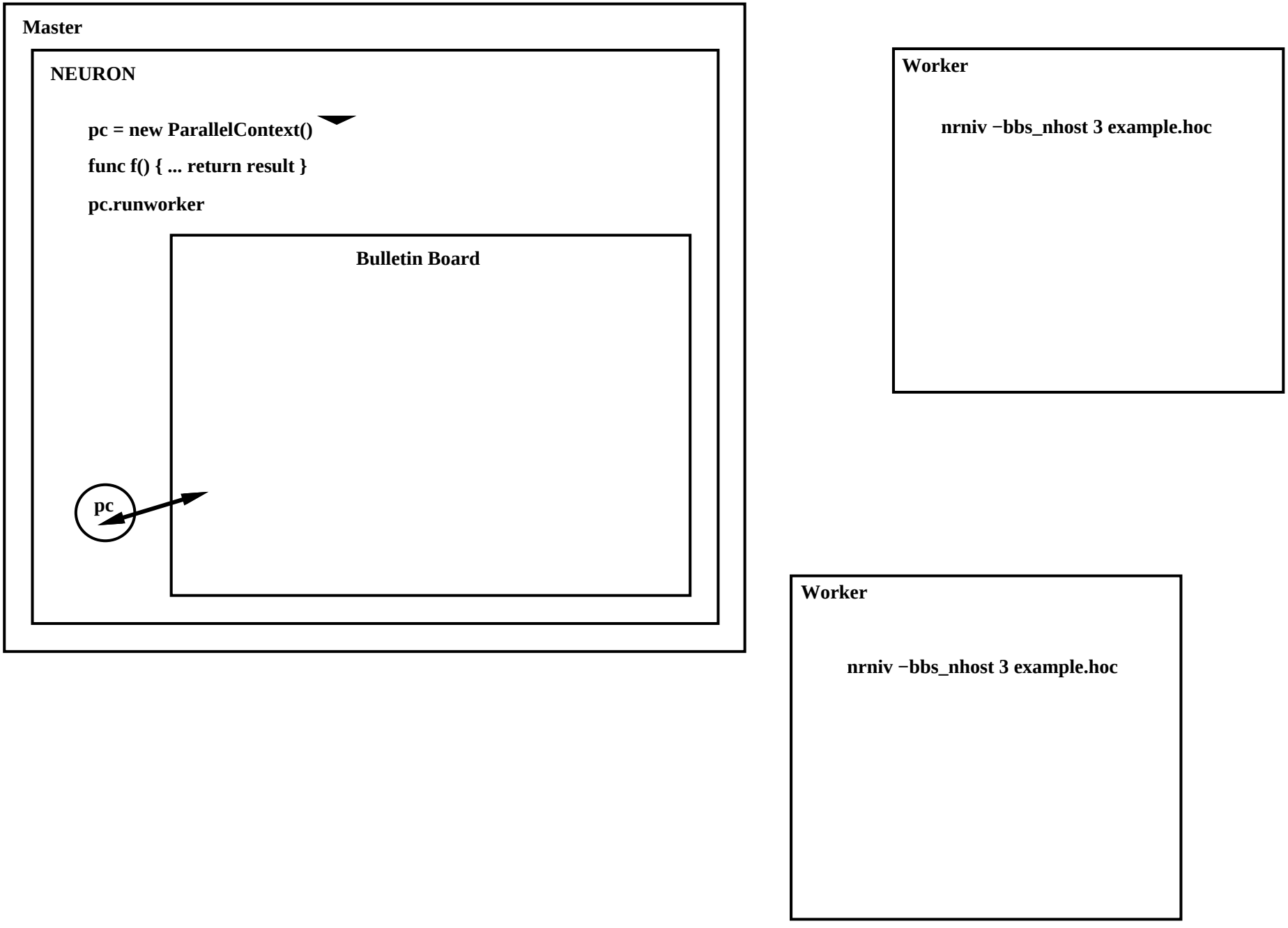
example.hoc

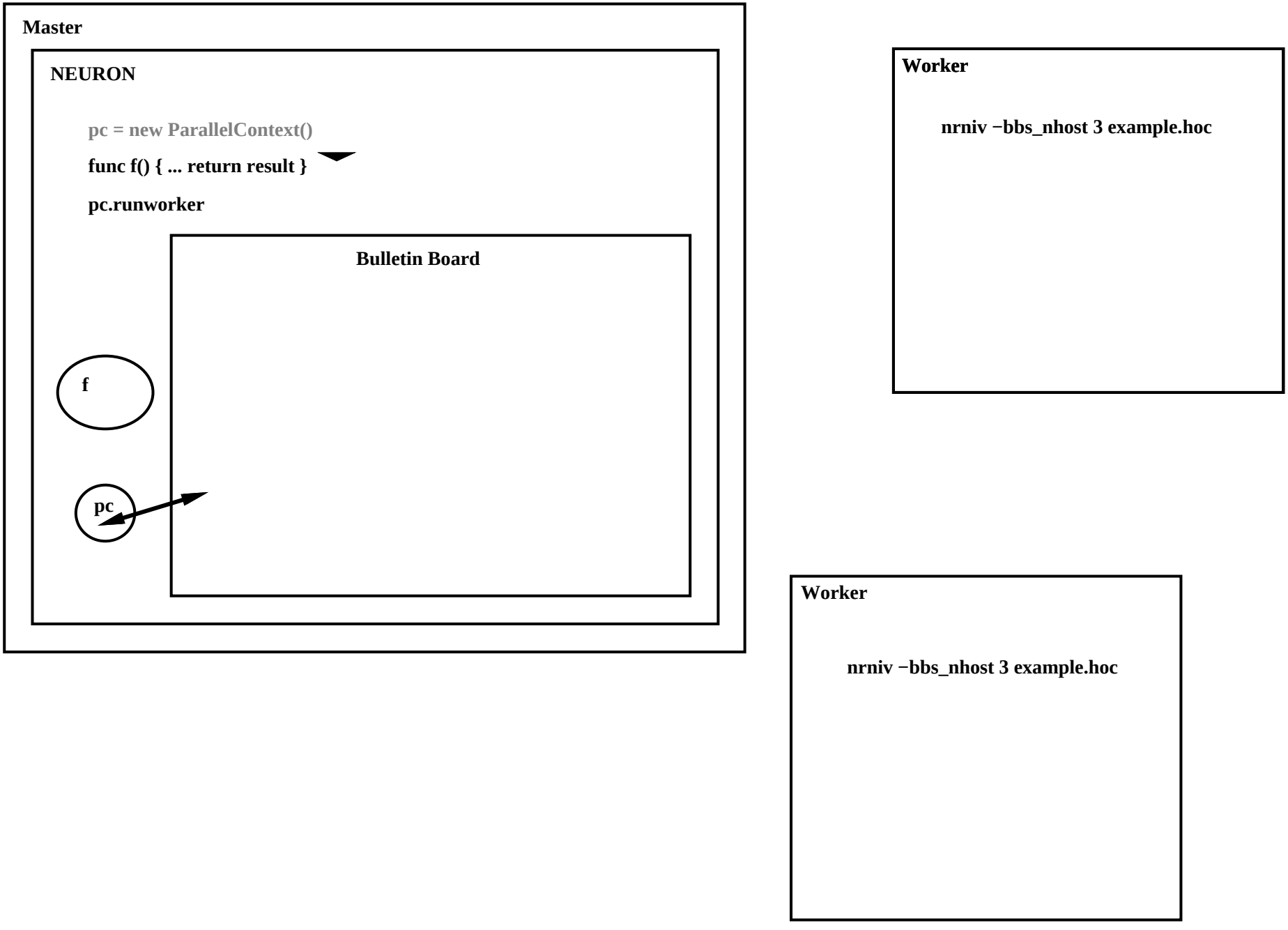
```
objref pc
pc = new ParallelContext()

load_file("nrngui.hoc")
func f(
    s = 0
    for i=1,100000 {
        s += $1
    }
    return s
}

pc.runworker

s = 0
for i = 1, 10 {
    pc.submit("f", i)
}
while (pc.working) {
    print pc.upkscalar, pc.retval
    s += pc.retval
}
print s

pc.done
```

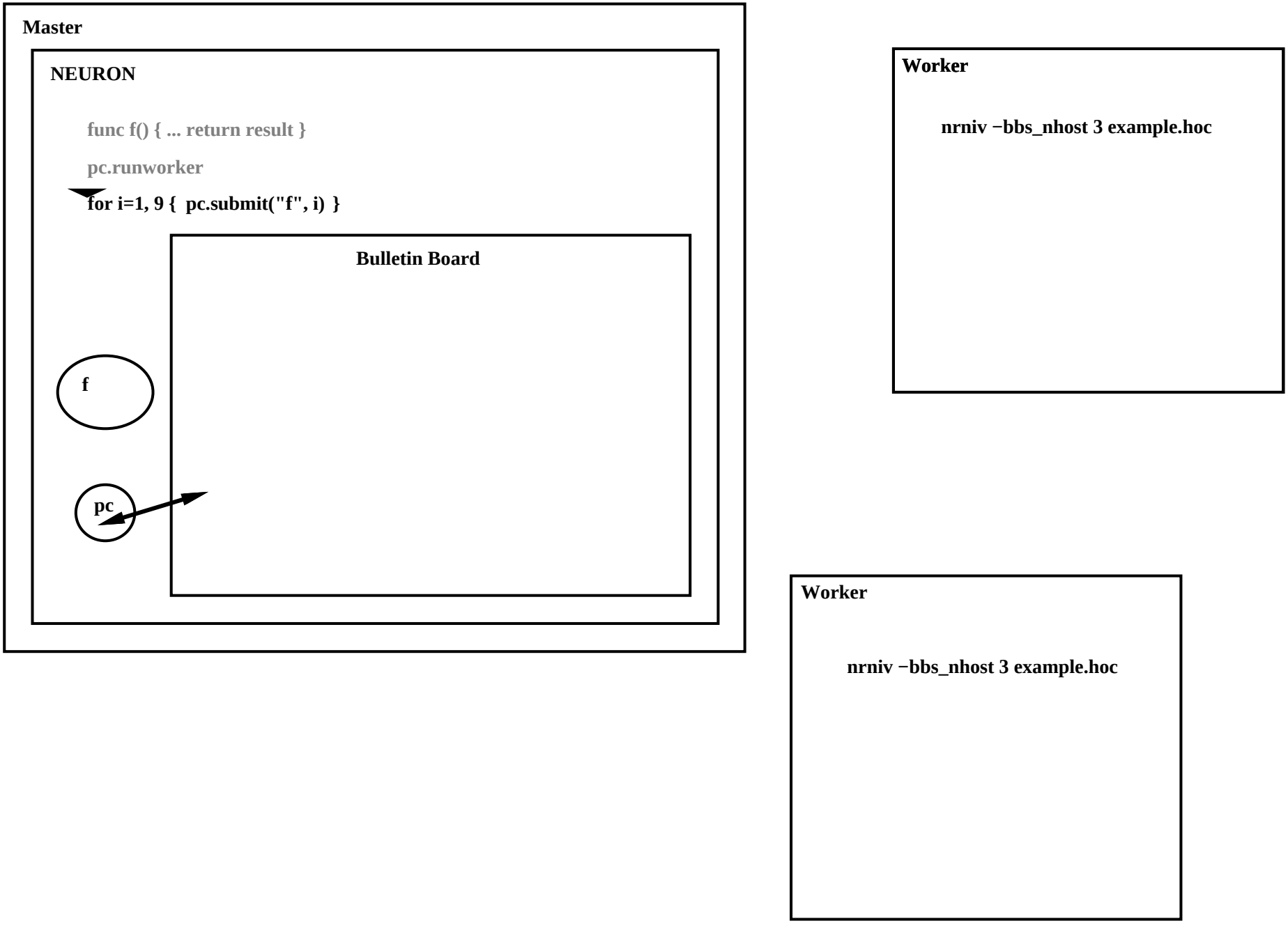
nrniv example.hoc

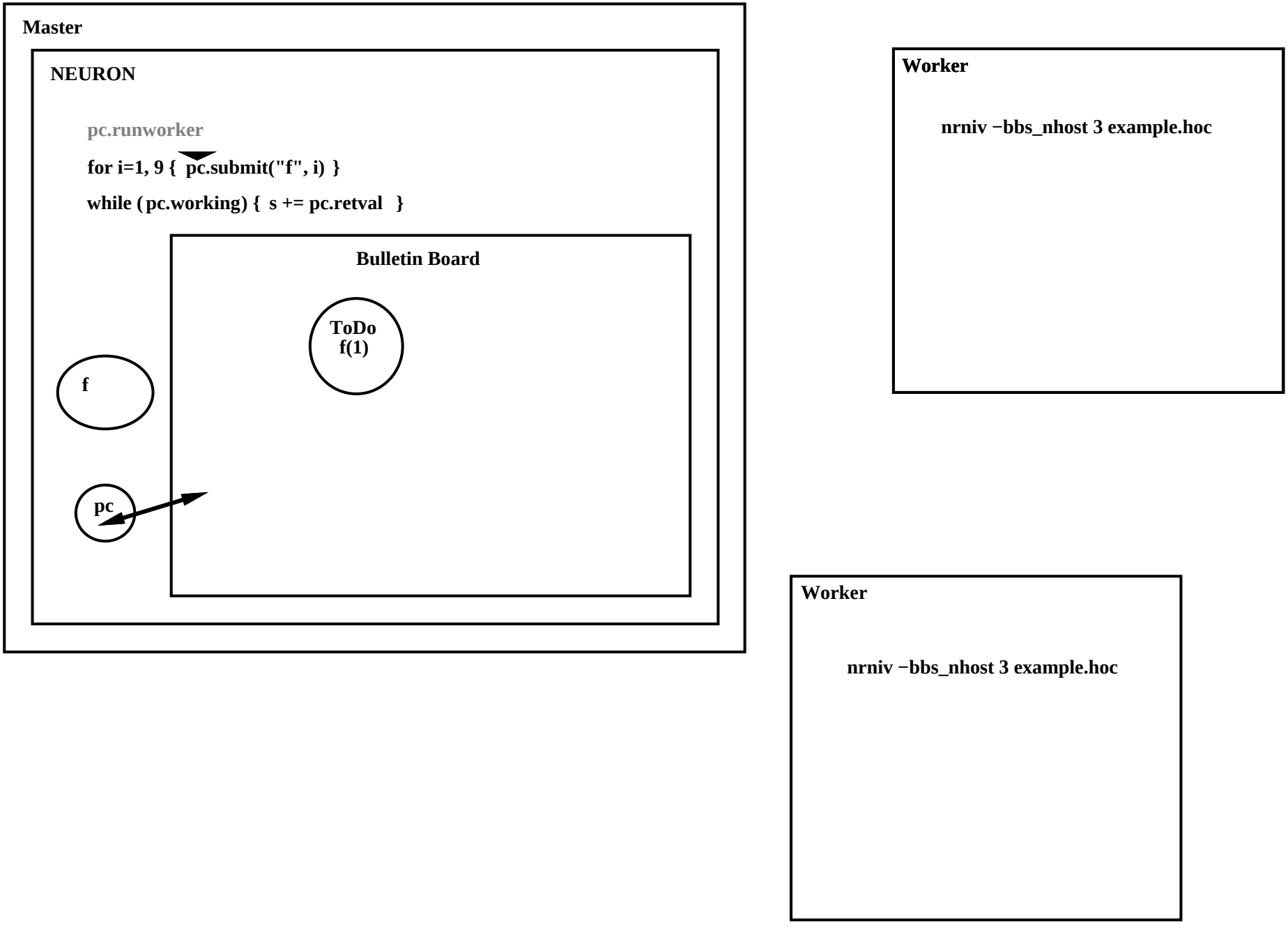

Master
nrniv example.hoc
Worker
Worker

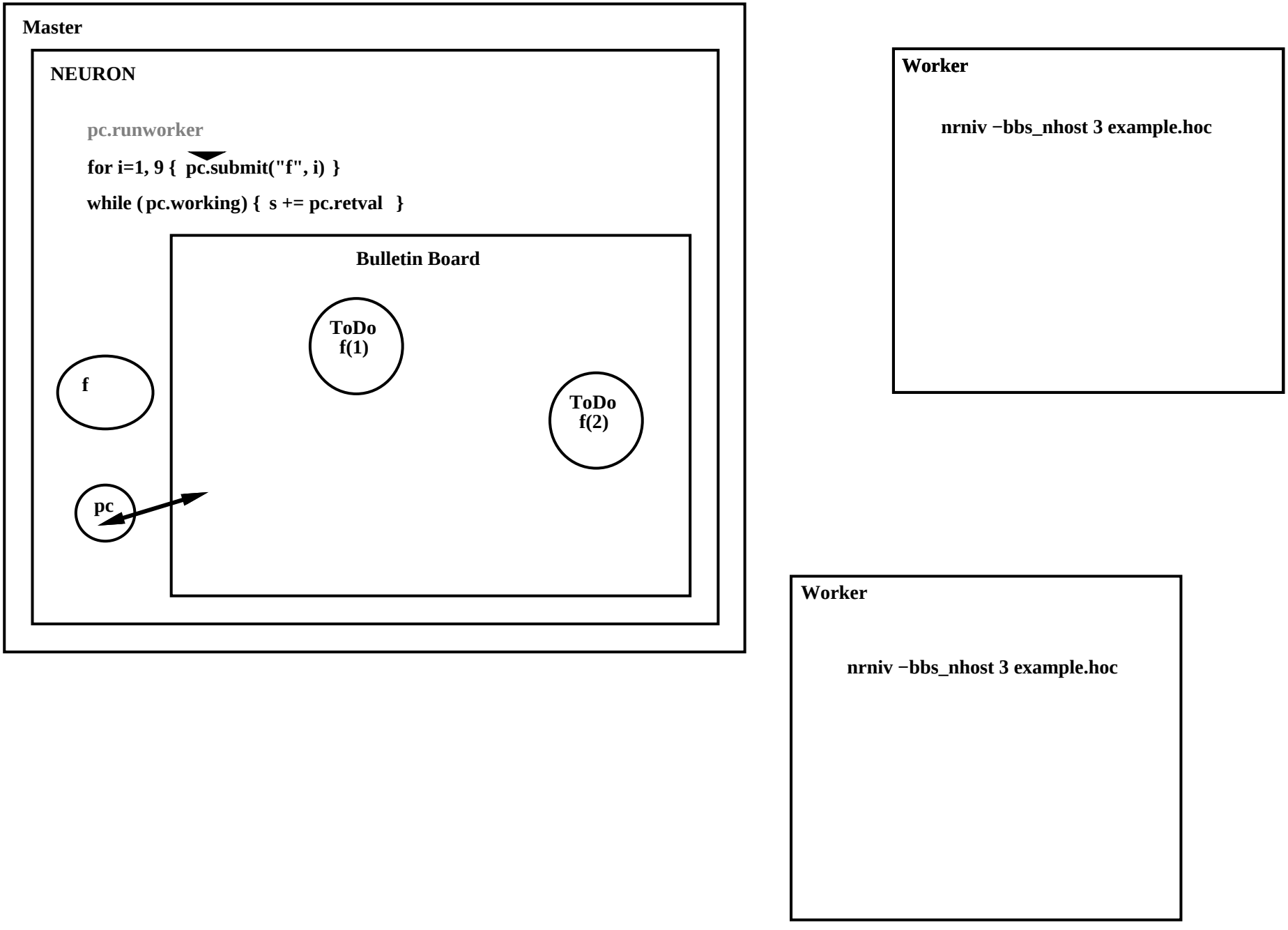

Master
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
Worker
Worker

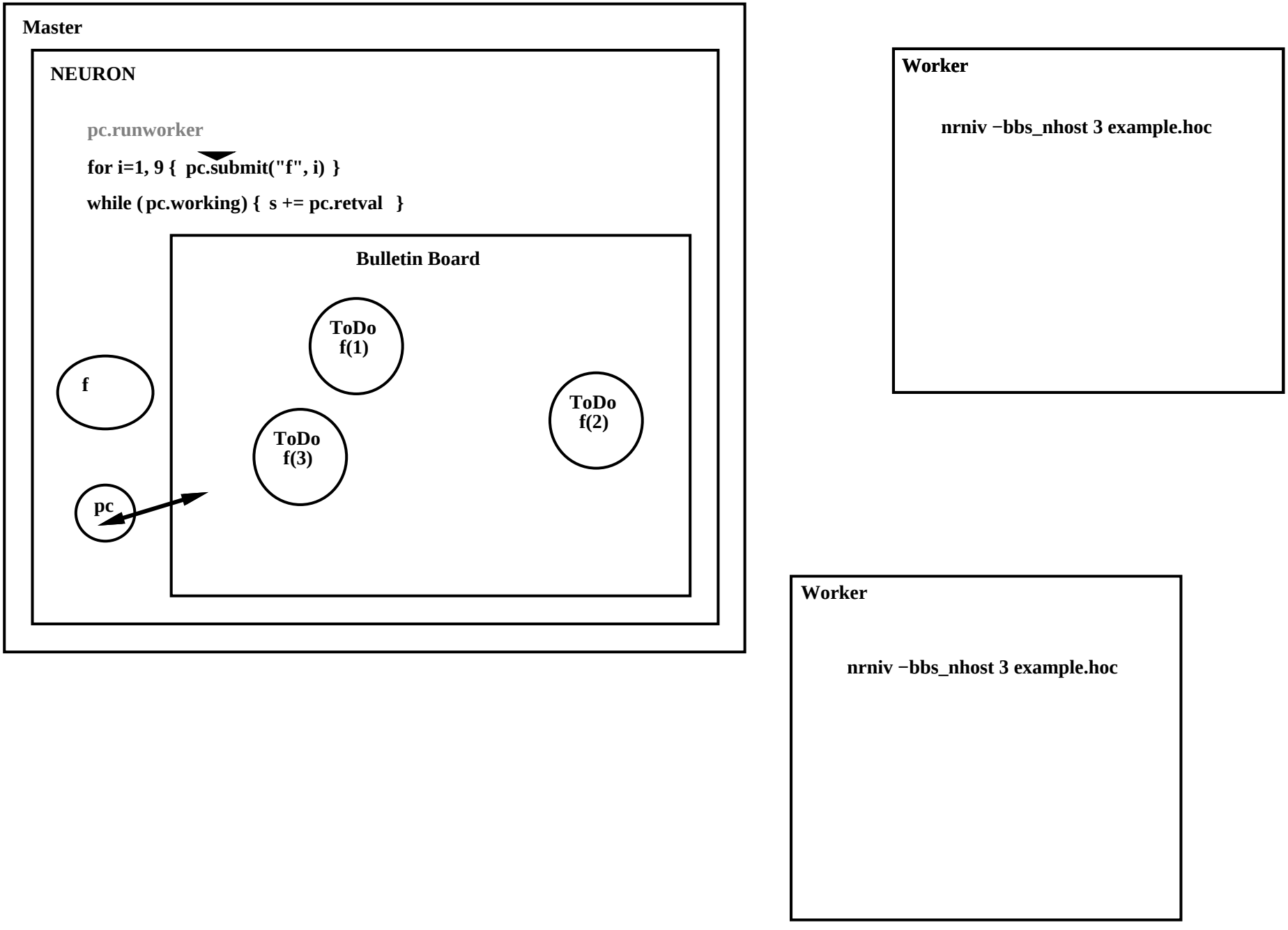

Master
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
Bulletin Board
pc
Worker
nrniv −bbs_nhost 3 example.hoc
Worker
nrniv −bbs_nhost 3 example.hoc

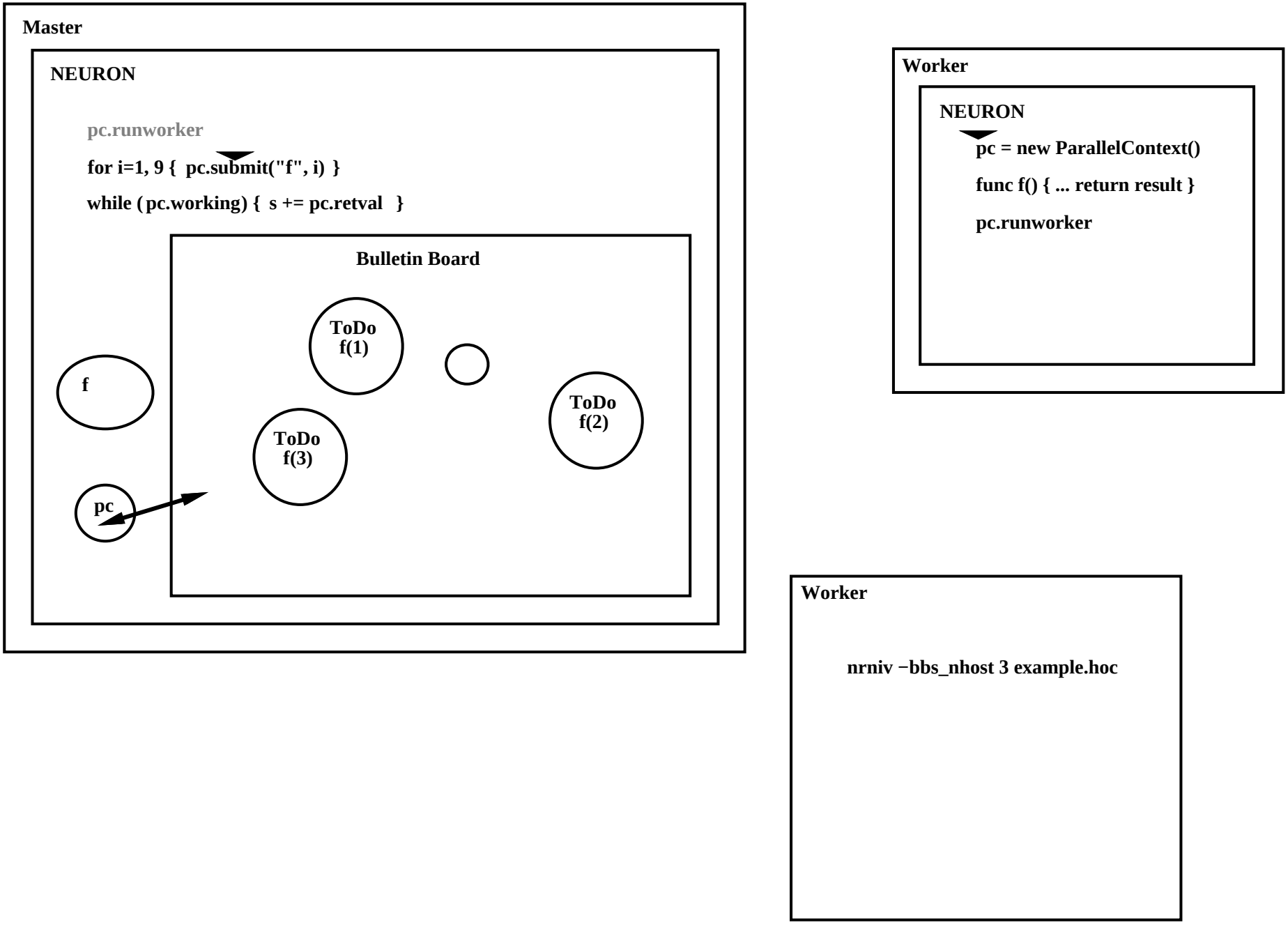

Master
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
Bulletin Board
f
pc
Worker
nrniv −bbs_nhost 3 example.hoc
Worker
nrniv −bbs_nhost 3 example.hoc

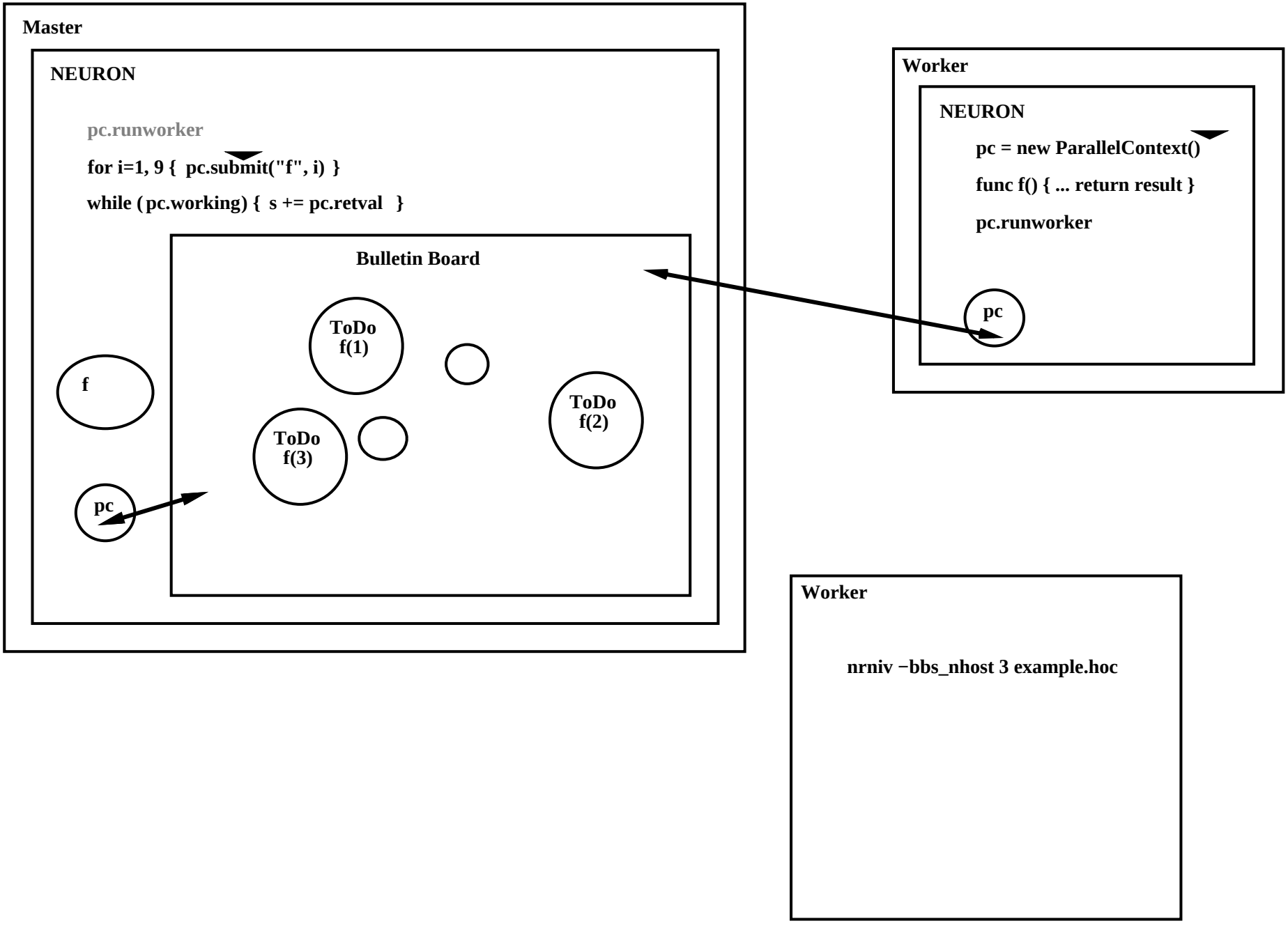

Master
NEURON
func f() { ... return result }
pc.runworker
for i=1, 9 { pc.submit("f", i) }
Bulletin Board
f
pc
Worker
nrniv −bbs_nhost 3 example.hoc
Worker
nrniv −bbs_nhost 3 example.hoc

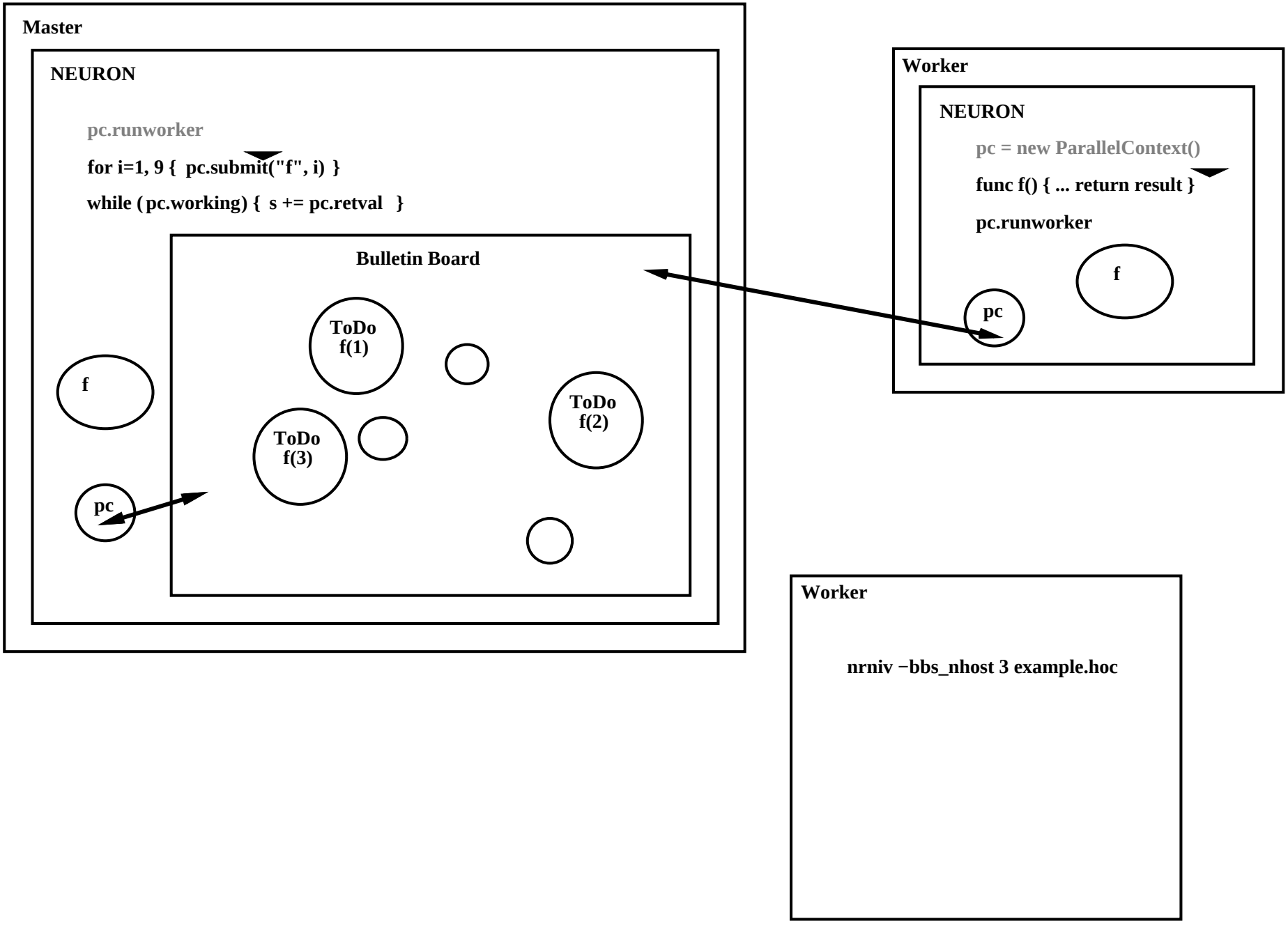

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
ToDo
f(1)
f
pc
Worker
nrniv −bbs_nhost 3 example.hoc
Worker
nrniv −bbs_nhost 3 example.hoc

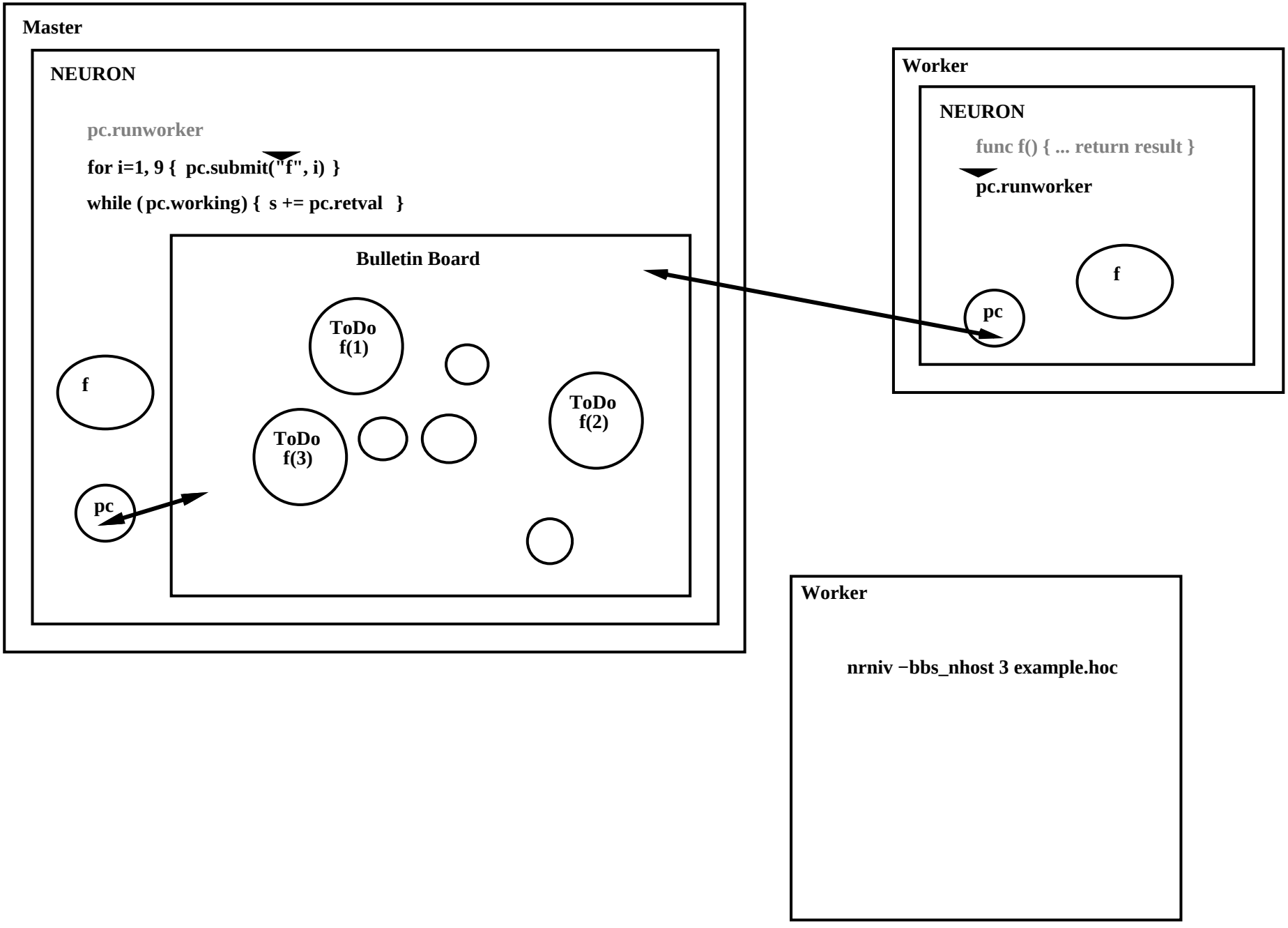

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
ToDo
f(1)
ToDo
f(2)
f
pc
Worker
nrniv −bbs_nhost 3 example.hoc
Worker
nrniv −bbs_nhost 3 example.hoc

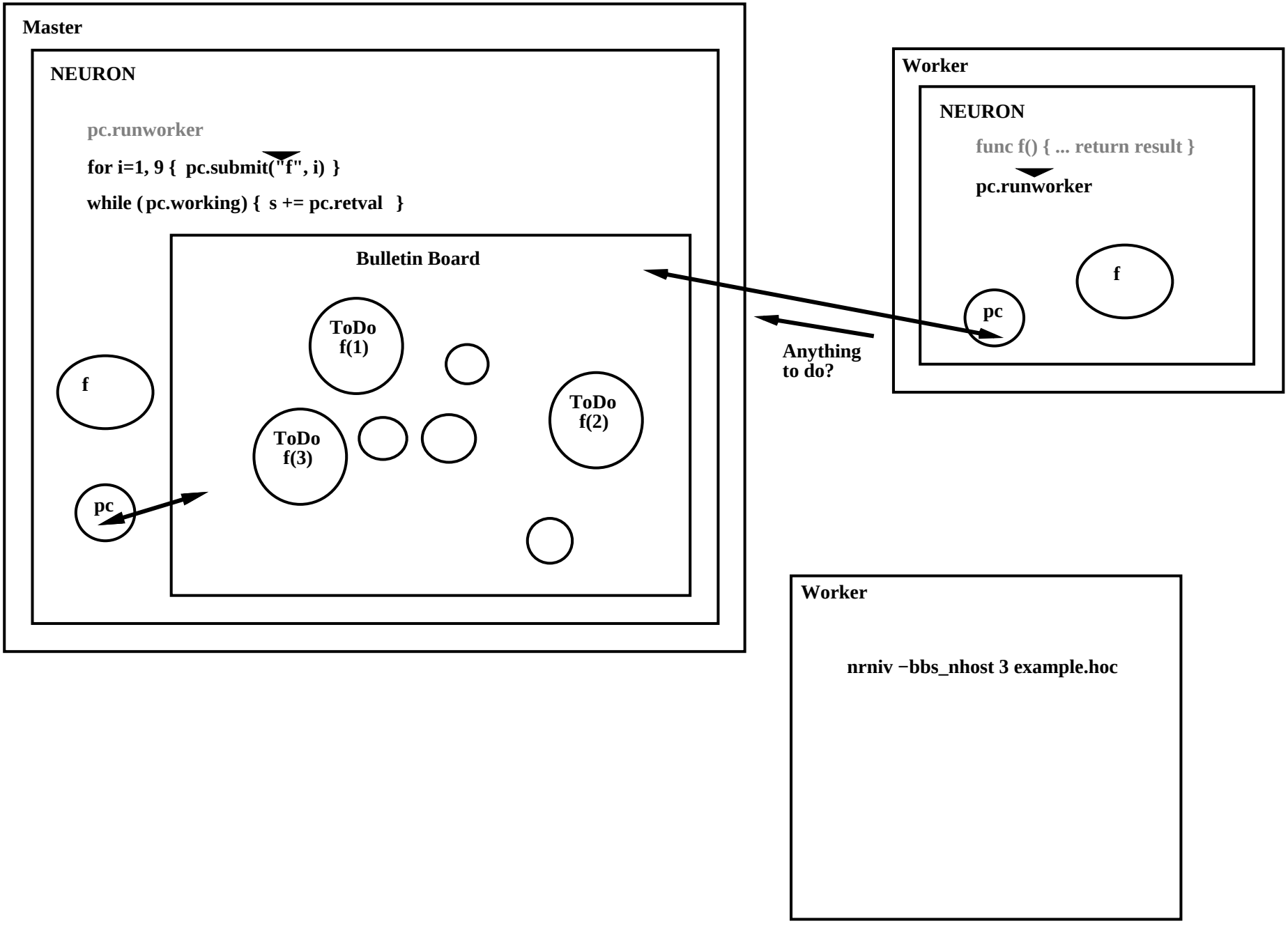

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
ToDo
f(1)
f
ToDo
f(2)
ToDo
f(3)
pc
Worker
nrniv −bbs_nhost 3 example.hoc
Worker
nrniv −bbs_nhost 3 example.hoc

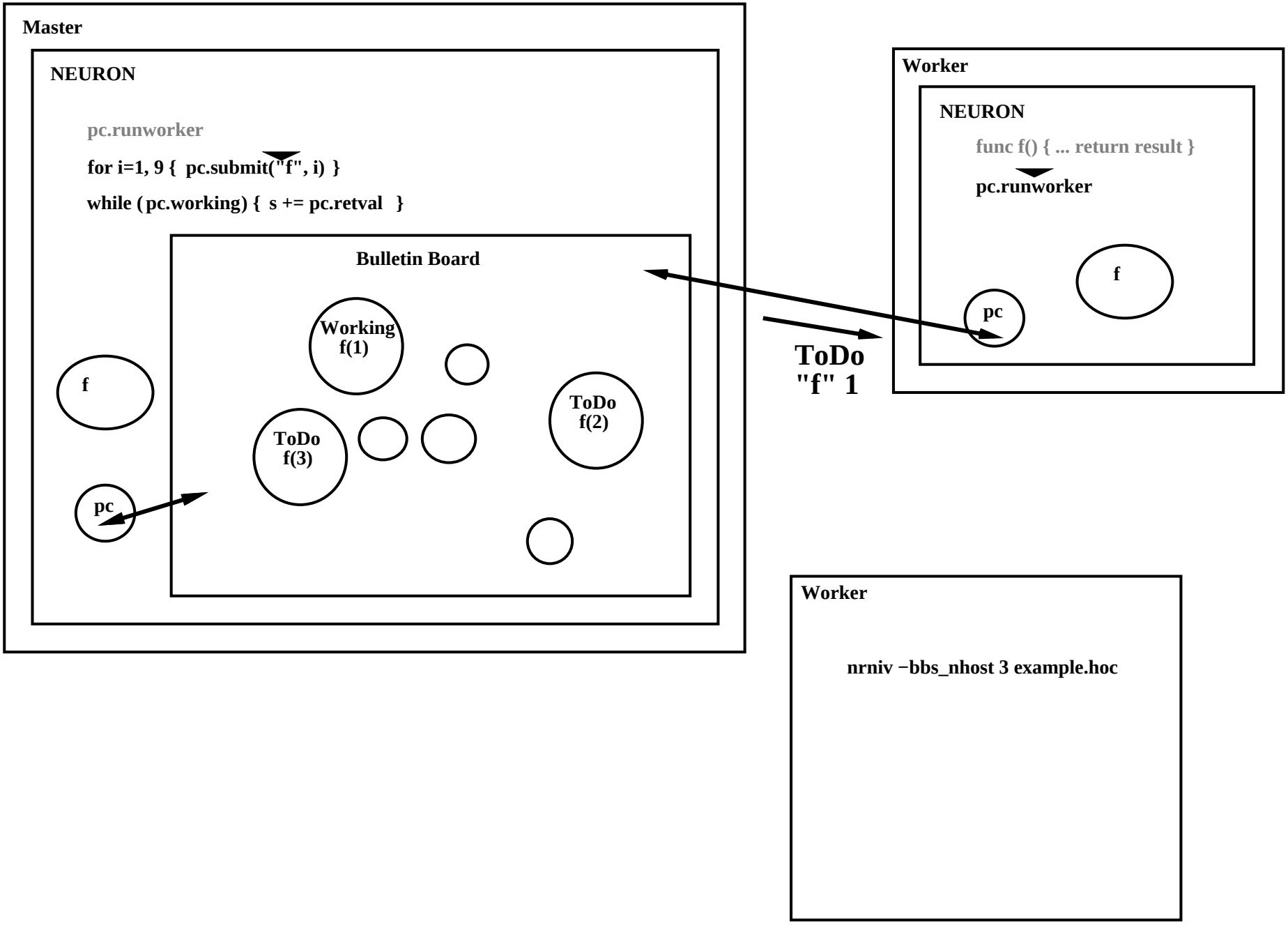

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
ToDo
f(1)
f
ToDo
f(2)
ToDo
f(3)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
Worker
nrniv −bbs_nhost 3 example.hoc

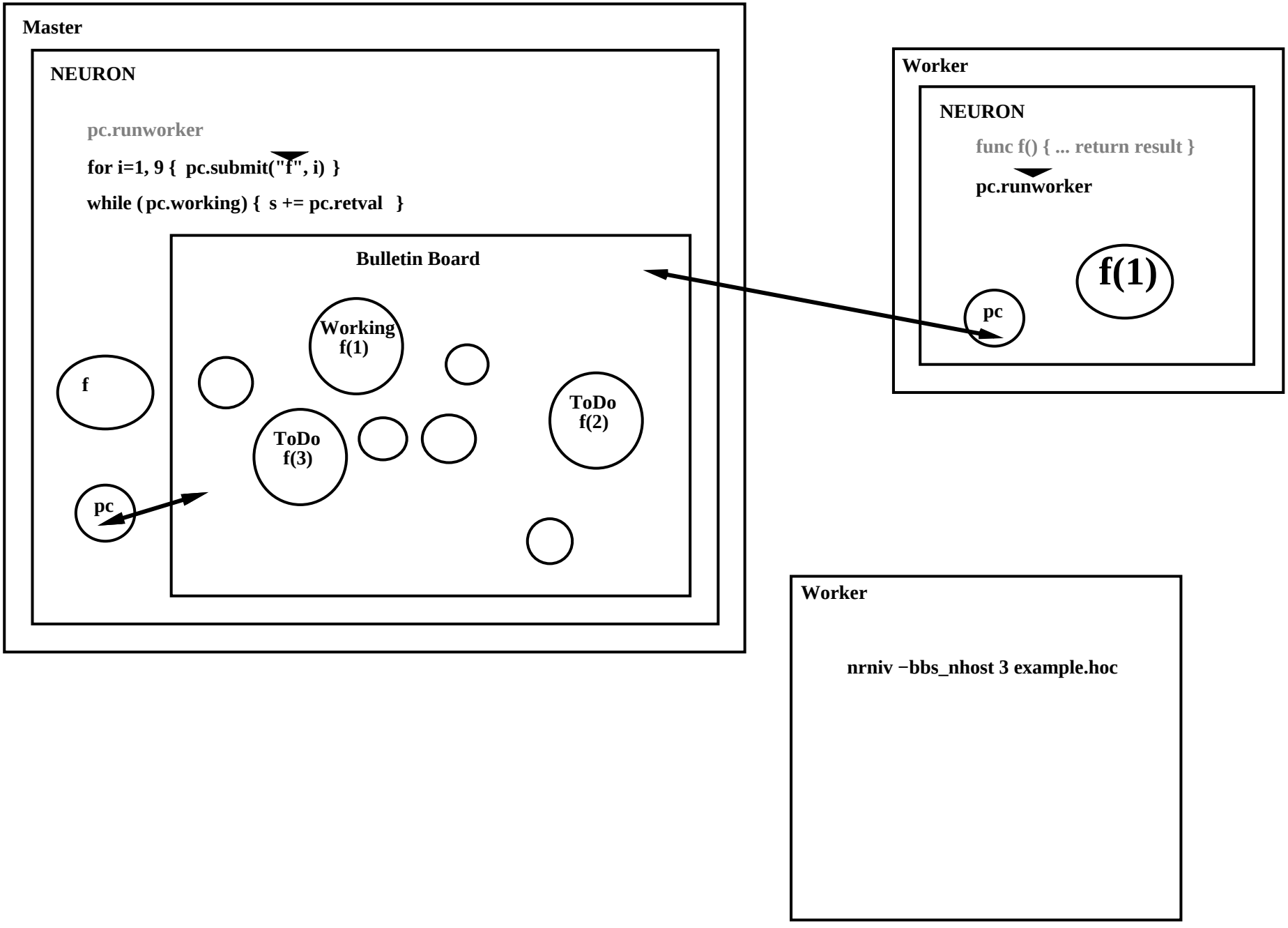

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
ToDo
f(1)
ToDo
f(2)
ToDo
f(3)
f
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
Worker
nrniv −bbs_nhost 3 example.hoc

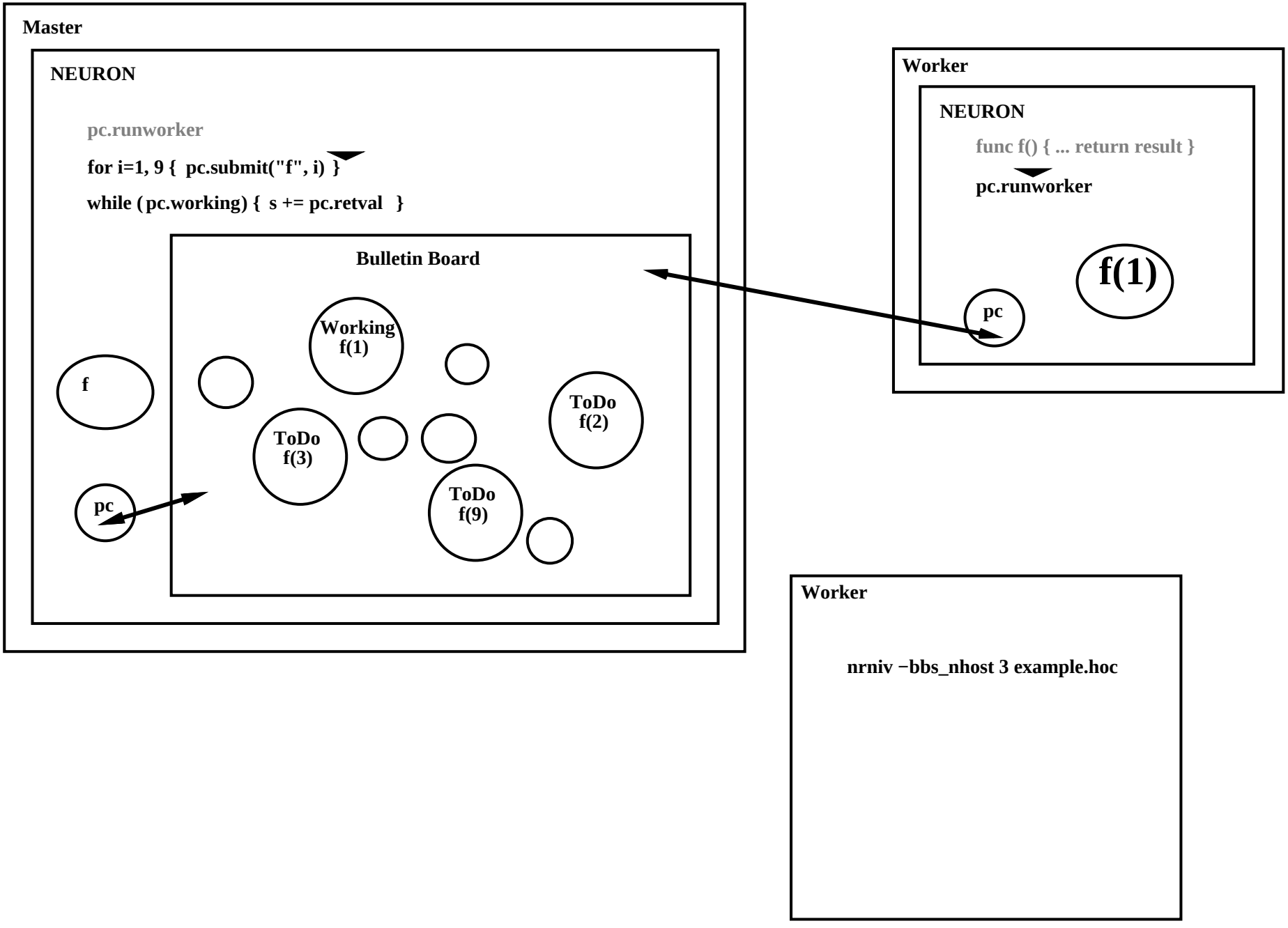

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
ToDo
f(1)
ToDo
f(2)
ToDo
f(3)
f
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
f
pc
Worker
nrniv −bbs_nhost 3 example.hoc

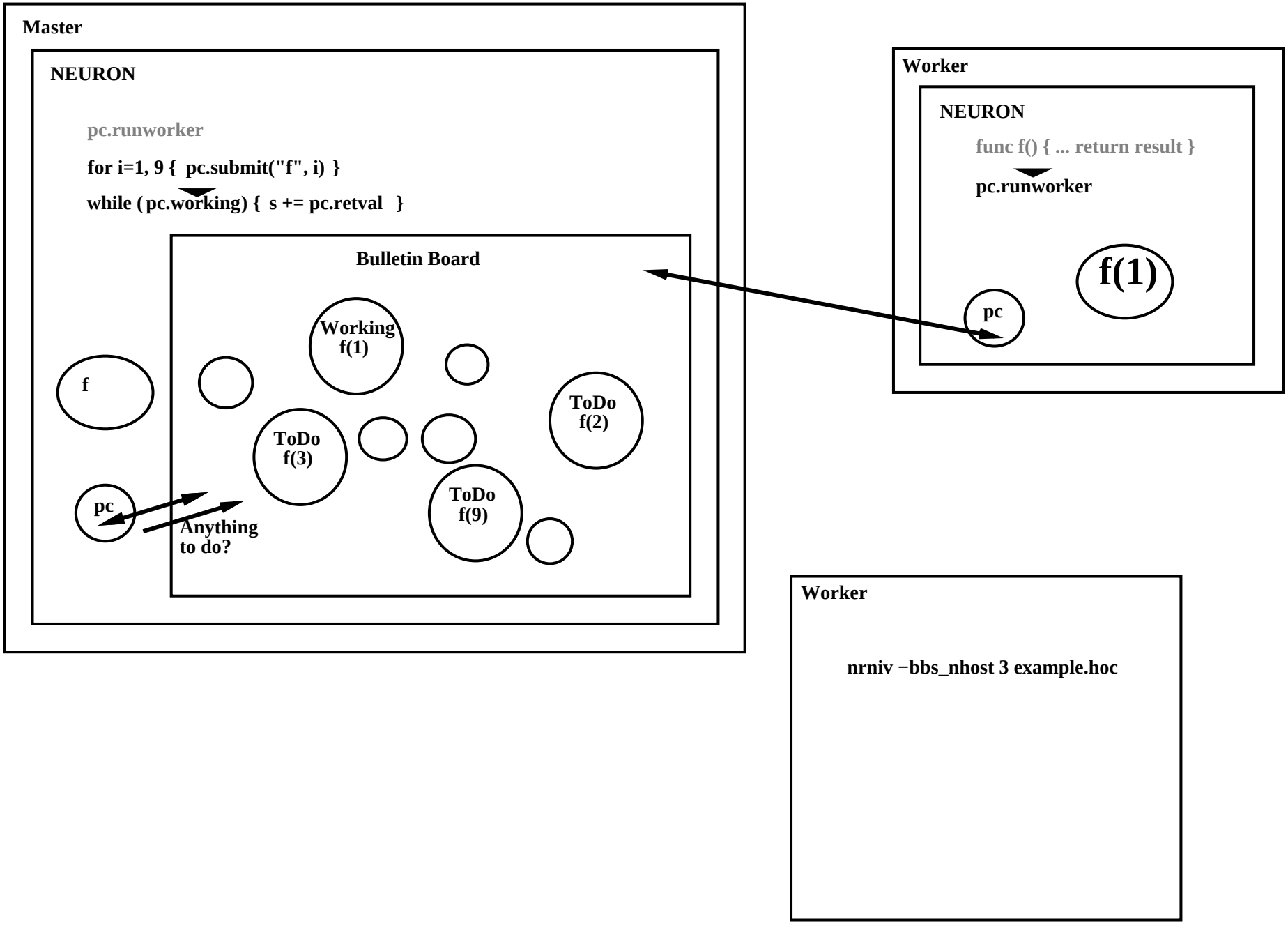

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
ToDo
f(1)
ToDo
f(2)
ToDo
f(3)
f
pc
Worker
NEURON
func f() { ... return result }
pc.runworker
f
pc
Worker
nrniv −bbs_nhost 3 example.hoc

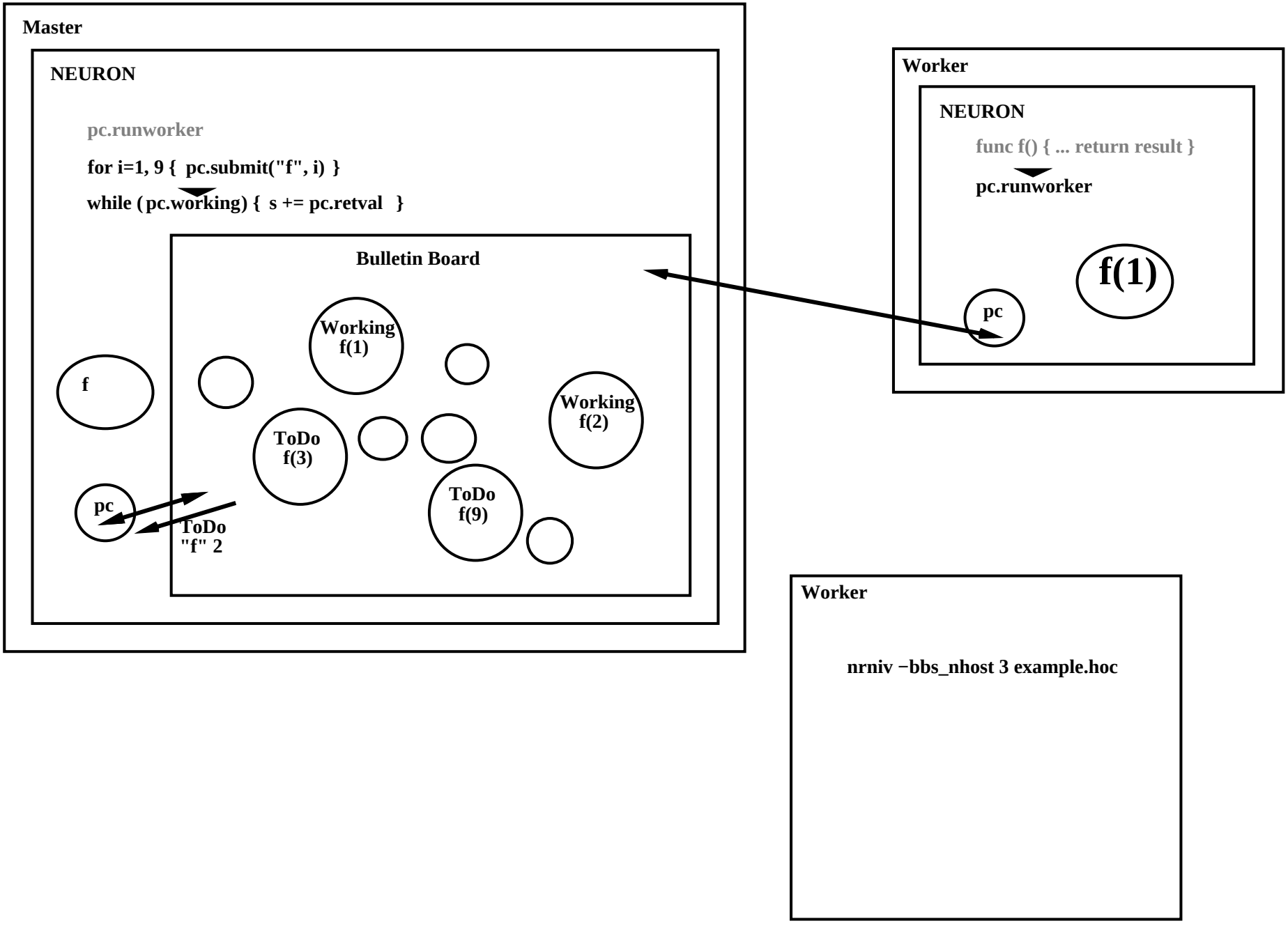

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
ToDo
f(1)
ToDo
f(2)
ToDo
f(3)
f
pc
Anything
to do?
Worker
NEURON
func f() { ... return result }
pc.runworker
f
pc
Worker
nrniv −bbs_nhost 3 example.hoc

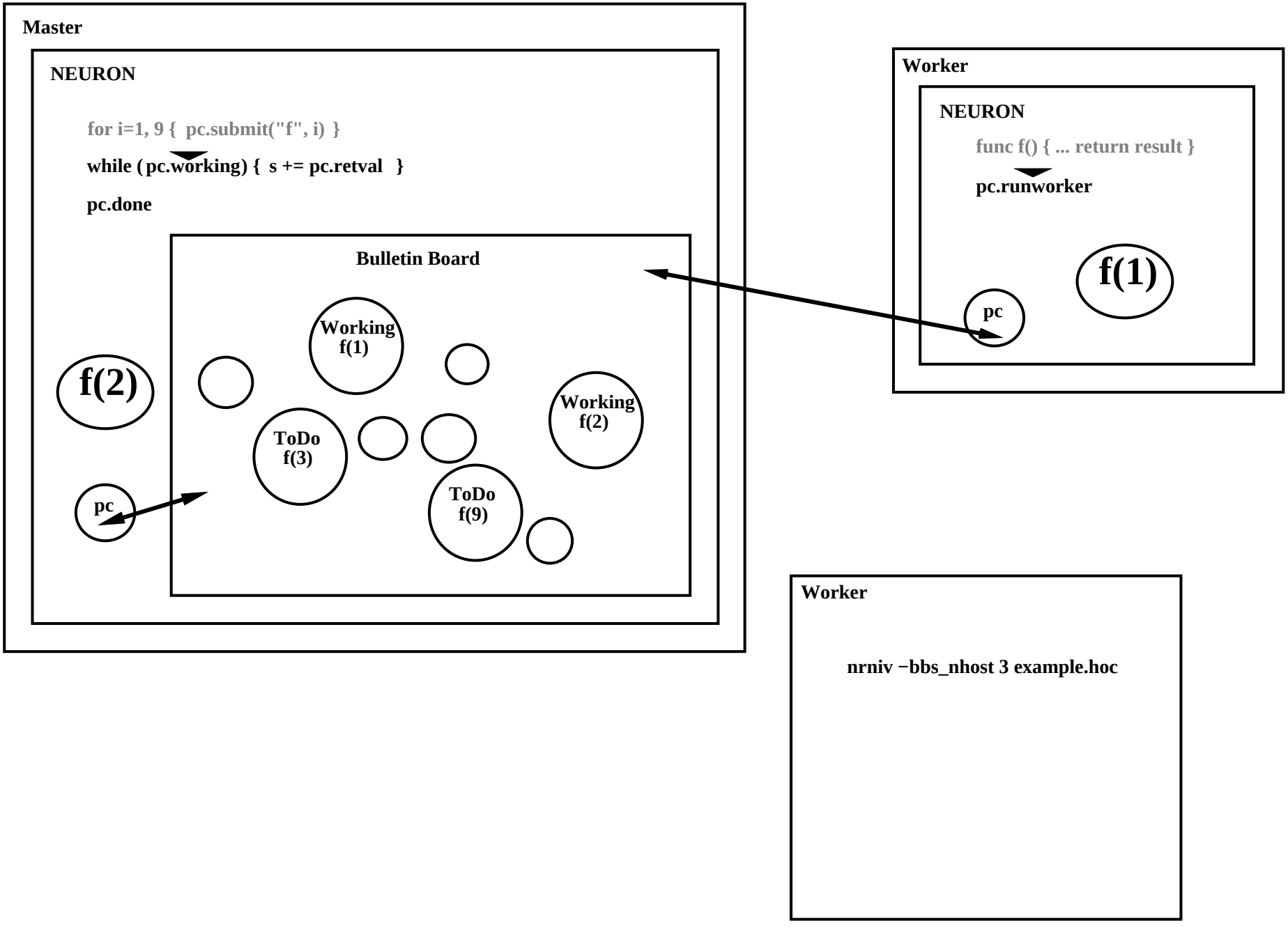

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
Working
f(1)
f
ToDo
f(3)
ToDo
f(2)
pc
ToDo
"f" 1
Worker
NEURON
func f() { ... return result }
pc.runworker
f
pc
Worker
nrniv −bbs_nhost 3 example.hoc

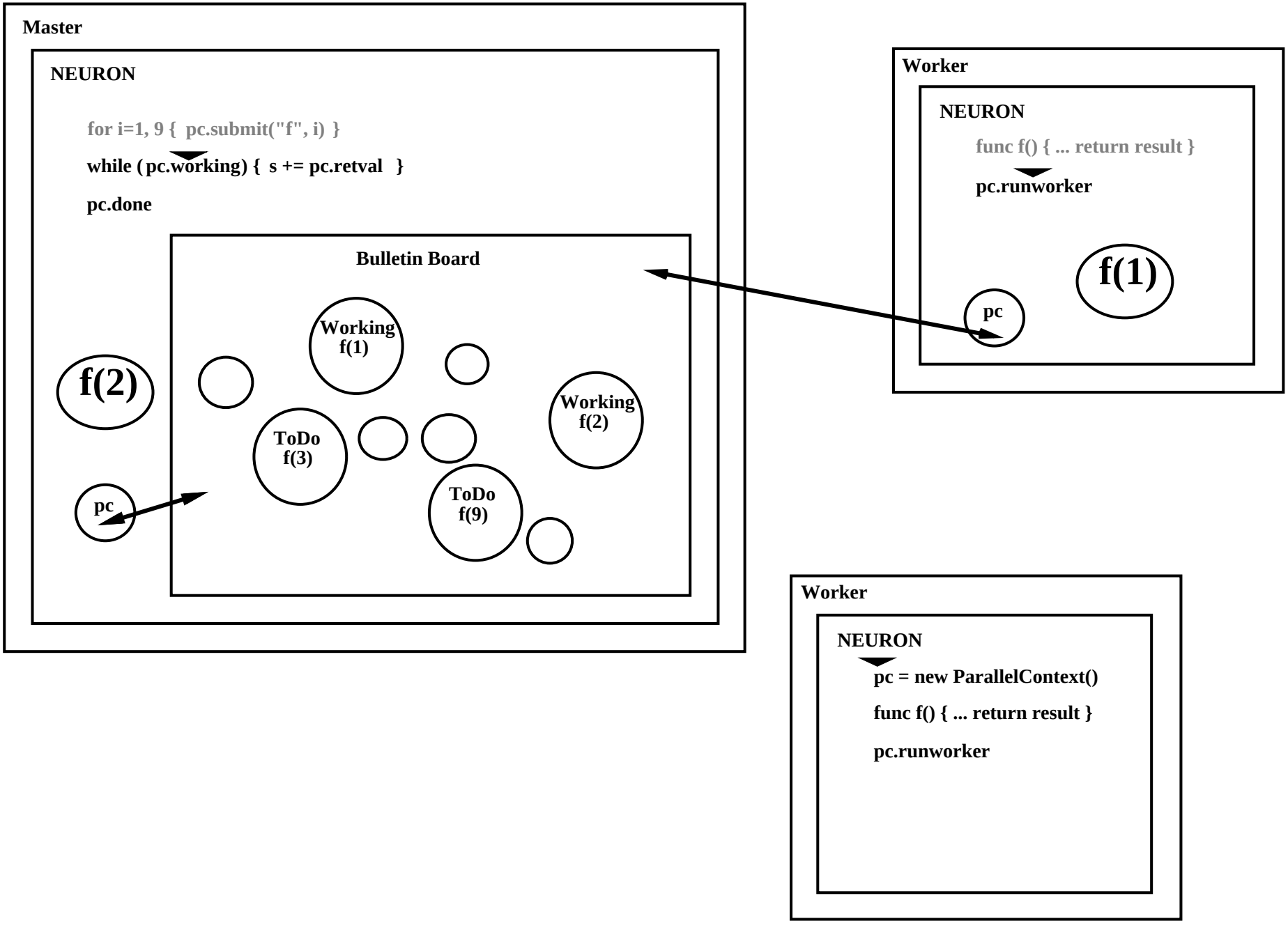

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
Working
f(1)
f
ToDo
f(2)
ToDo
f(3)
pc
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
nrniv −bbs_nhost 3 example.hoc

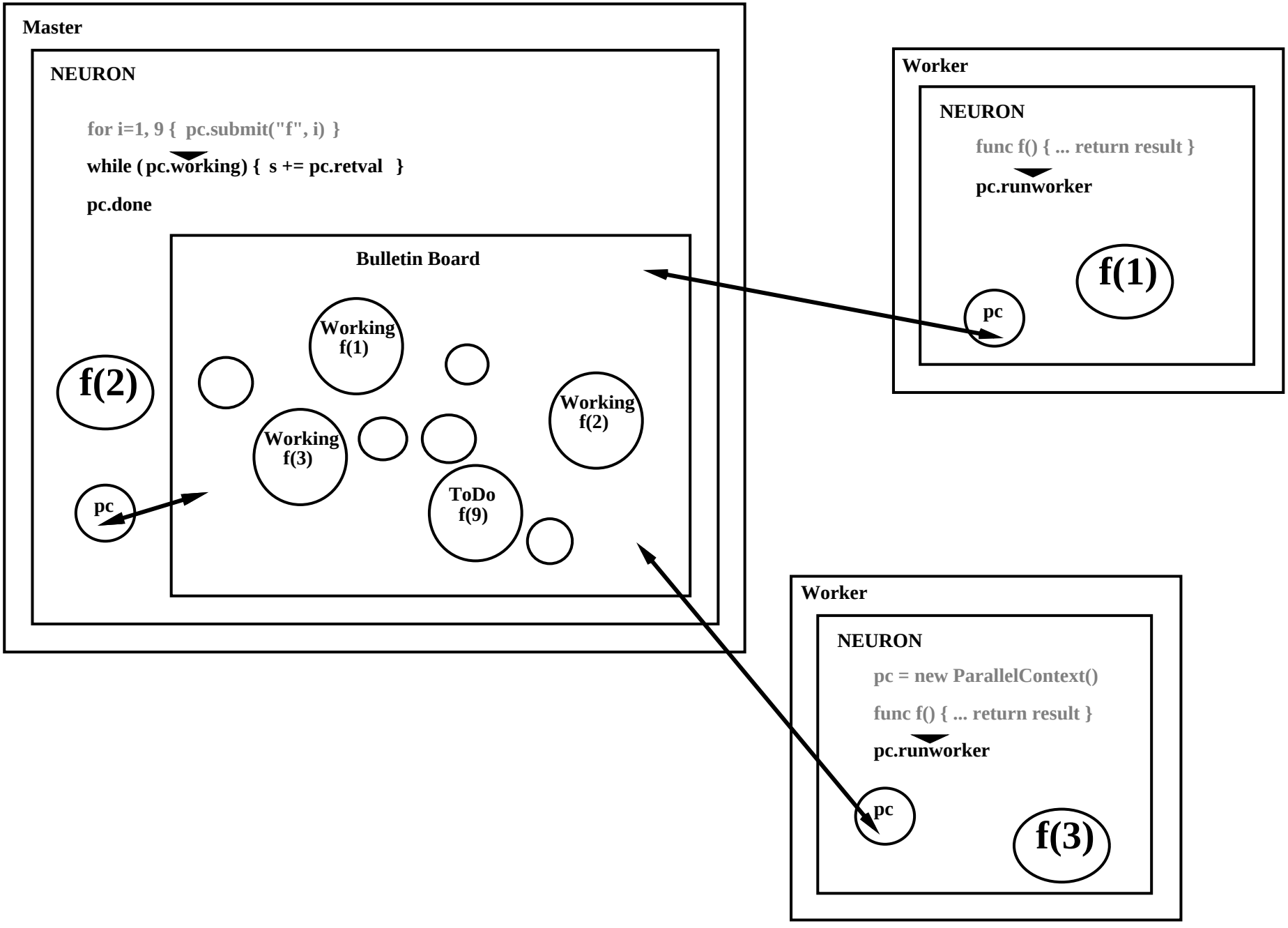

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
Working
f(1)
f
ToDo
f(2)
ToDo
f(3)
ToDo
f(9)
pc
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
nrniv −bbs_nhost 3 example.hoc

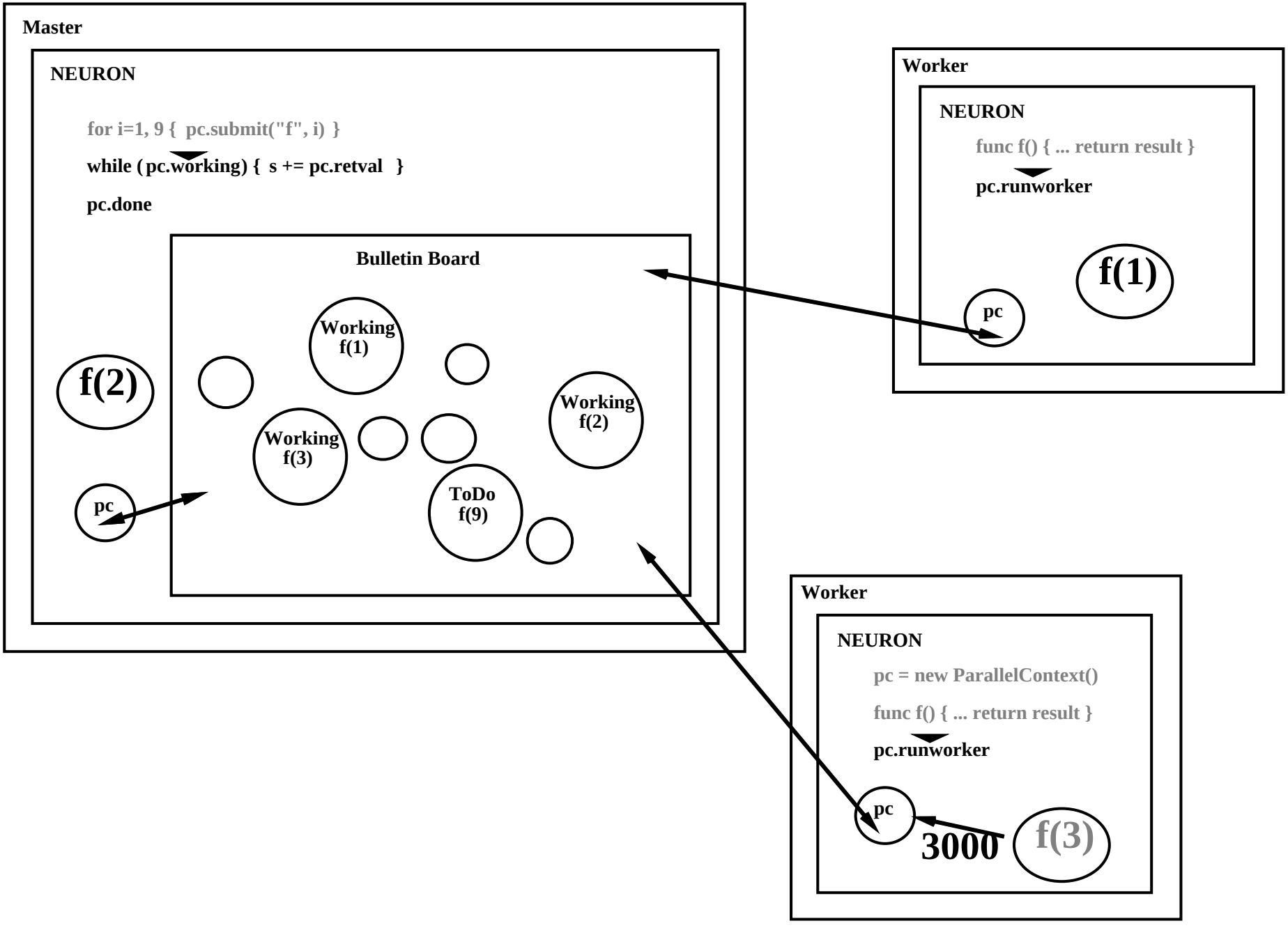

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
Working
f(1)
f
ToDo
f(2)
ToDo
f(3)
ToDo
f(9)
pc
Anything
to do?
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
nrniv −bbs_nhost 3 example.hoc

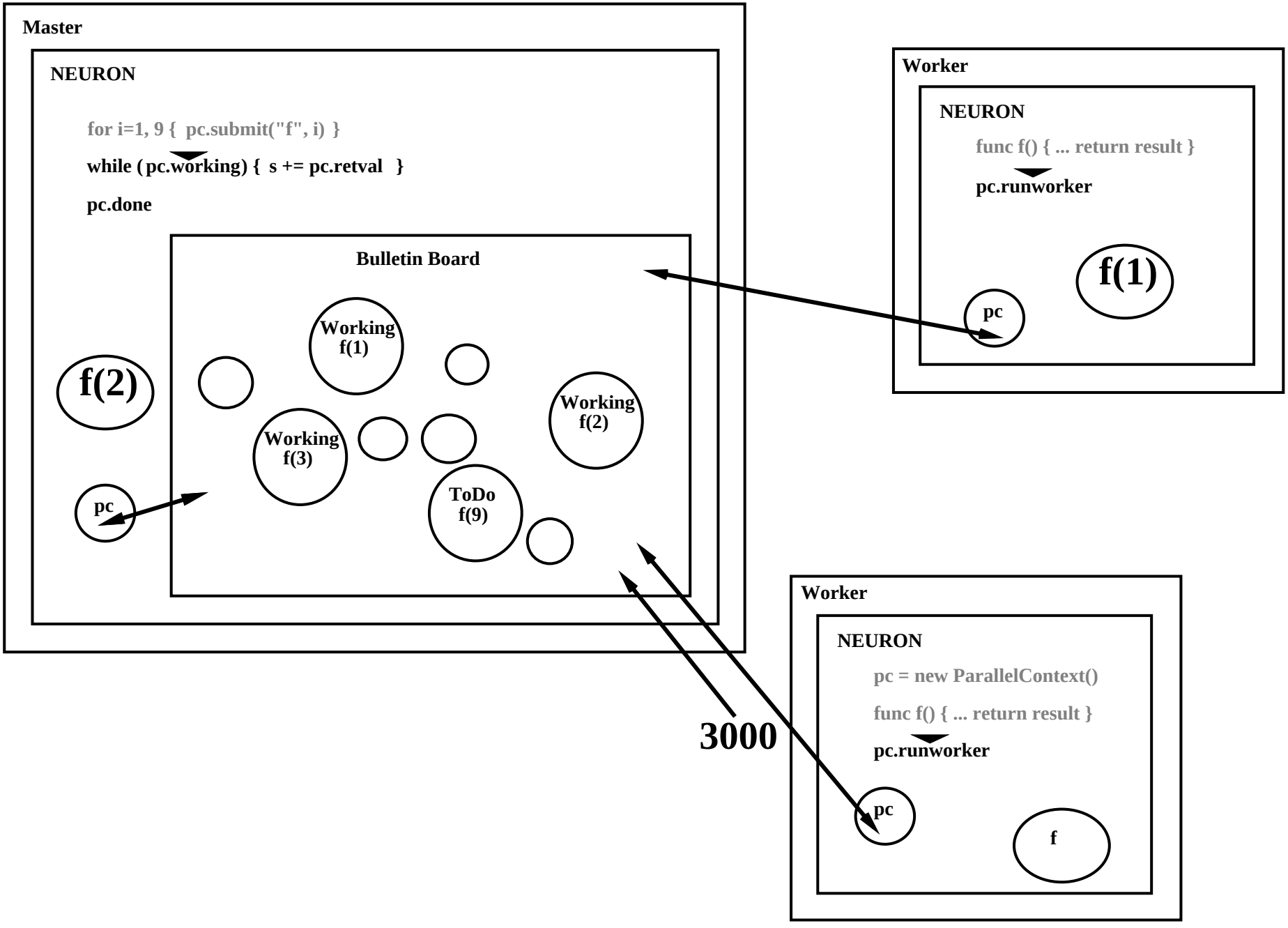

Master
NEURON
pc.runworker
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
Bulletin Board
Working
f(1)
f
ToDo
f(3)
Working
f(2)
pc
ToDo
"f" 2
ToDo
f(9)
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
nrniv −bbs_nhost 3 example.hoc

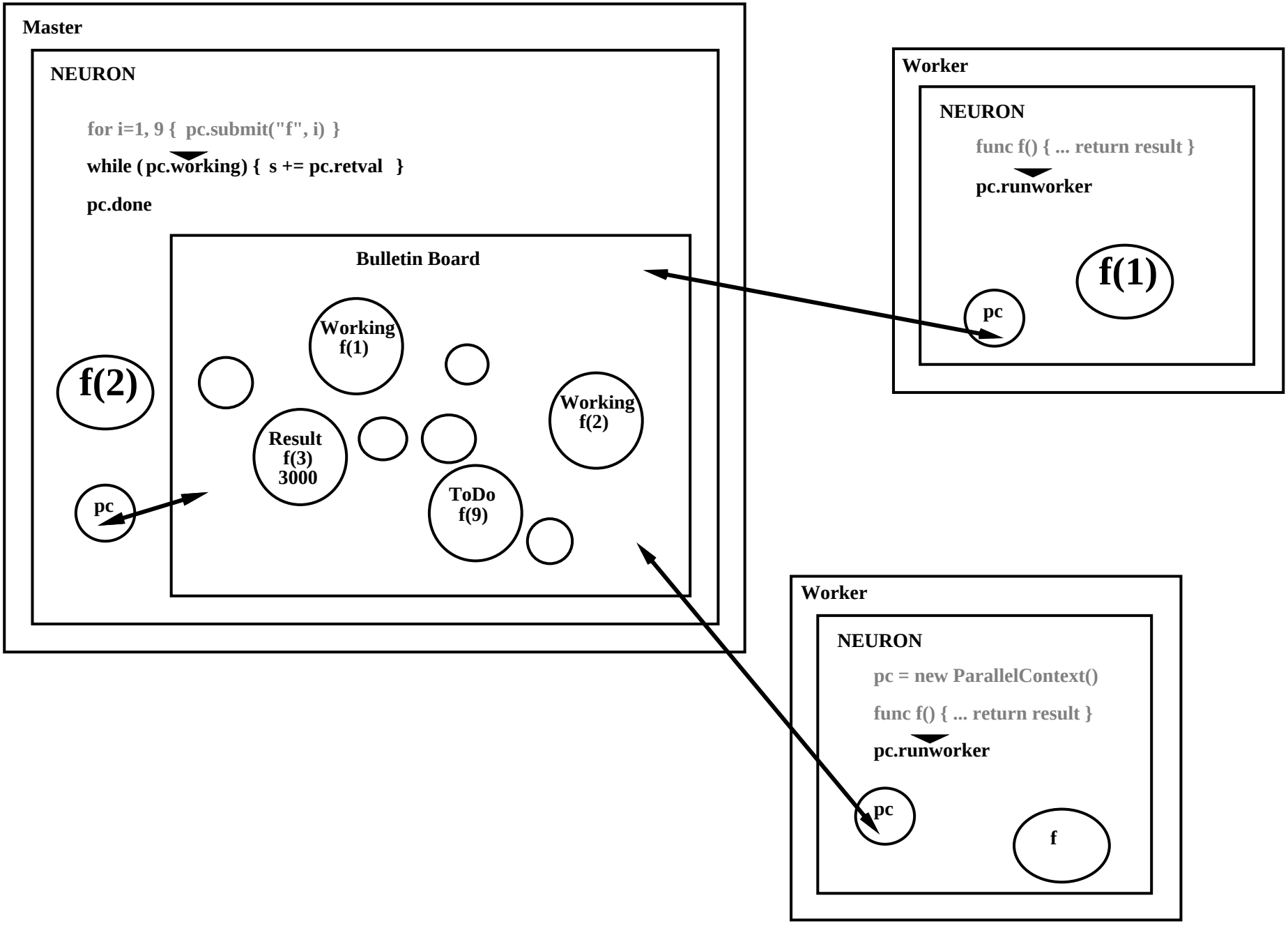

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
Working
f(1)
f(2)
ToDo
f(3)
Working
f(2)
pc
ToDo
f(9)
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
nrniv −bbs_nhost 3 example.hoc

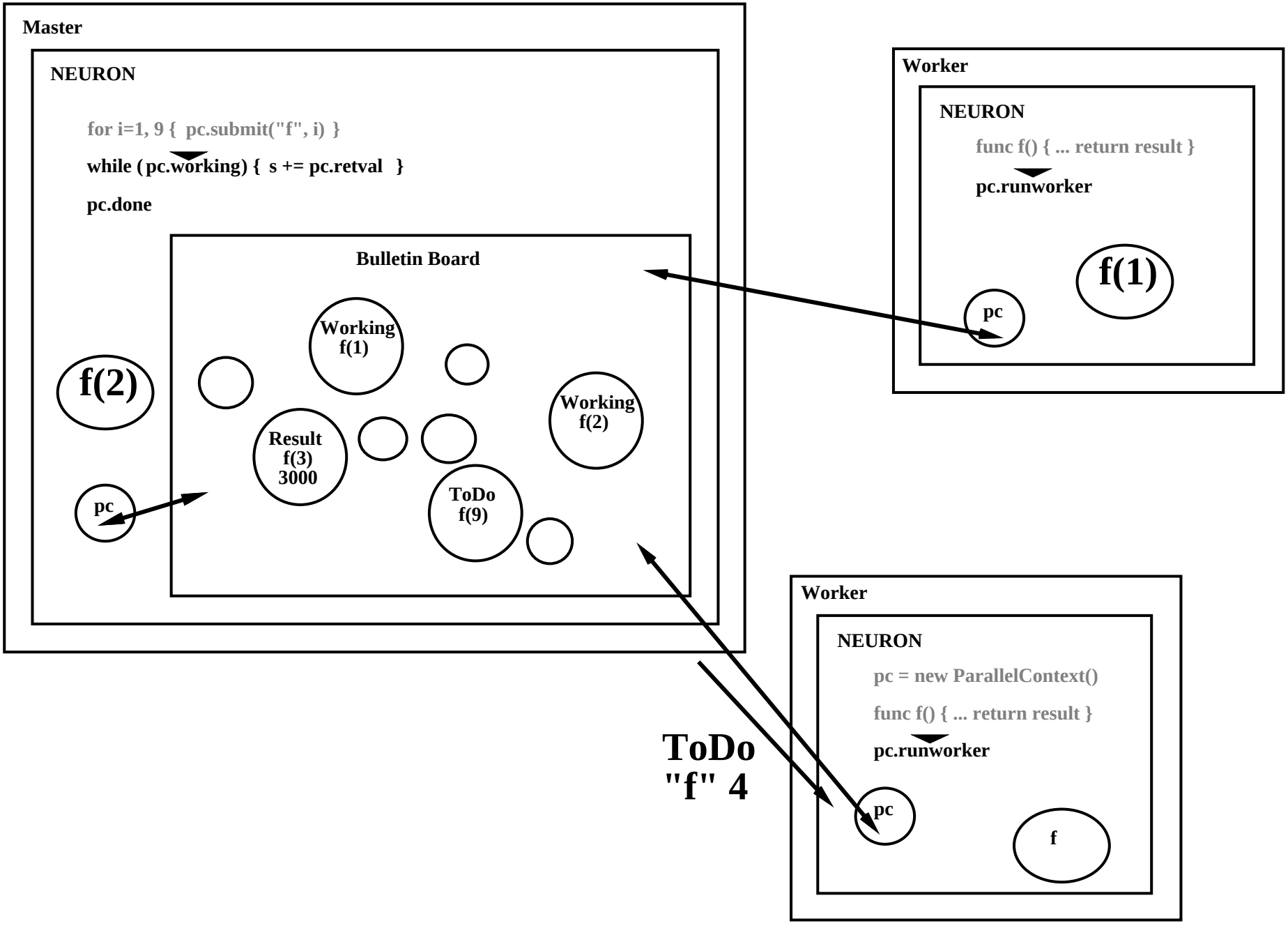

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
Working
f(1)
f(2)
ToDo
f(3)
Working
f(2)
pc
ToDo
f(9)
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker

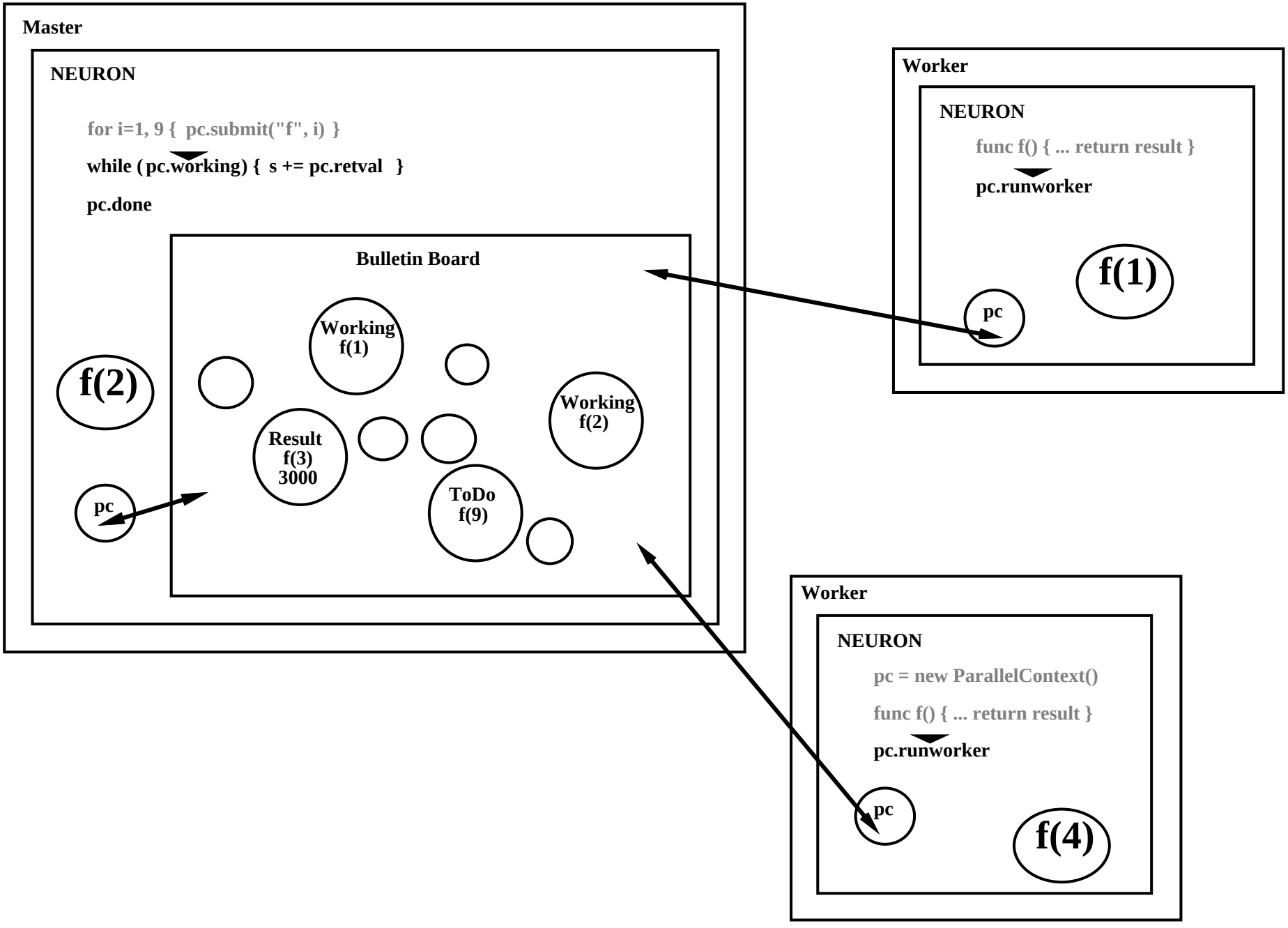

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
Working
f(1)
f(2)
Working
f(2)
Working
f(3)
ToDo
f(9)
pc
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(3)

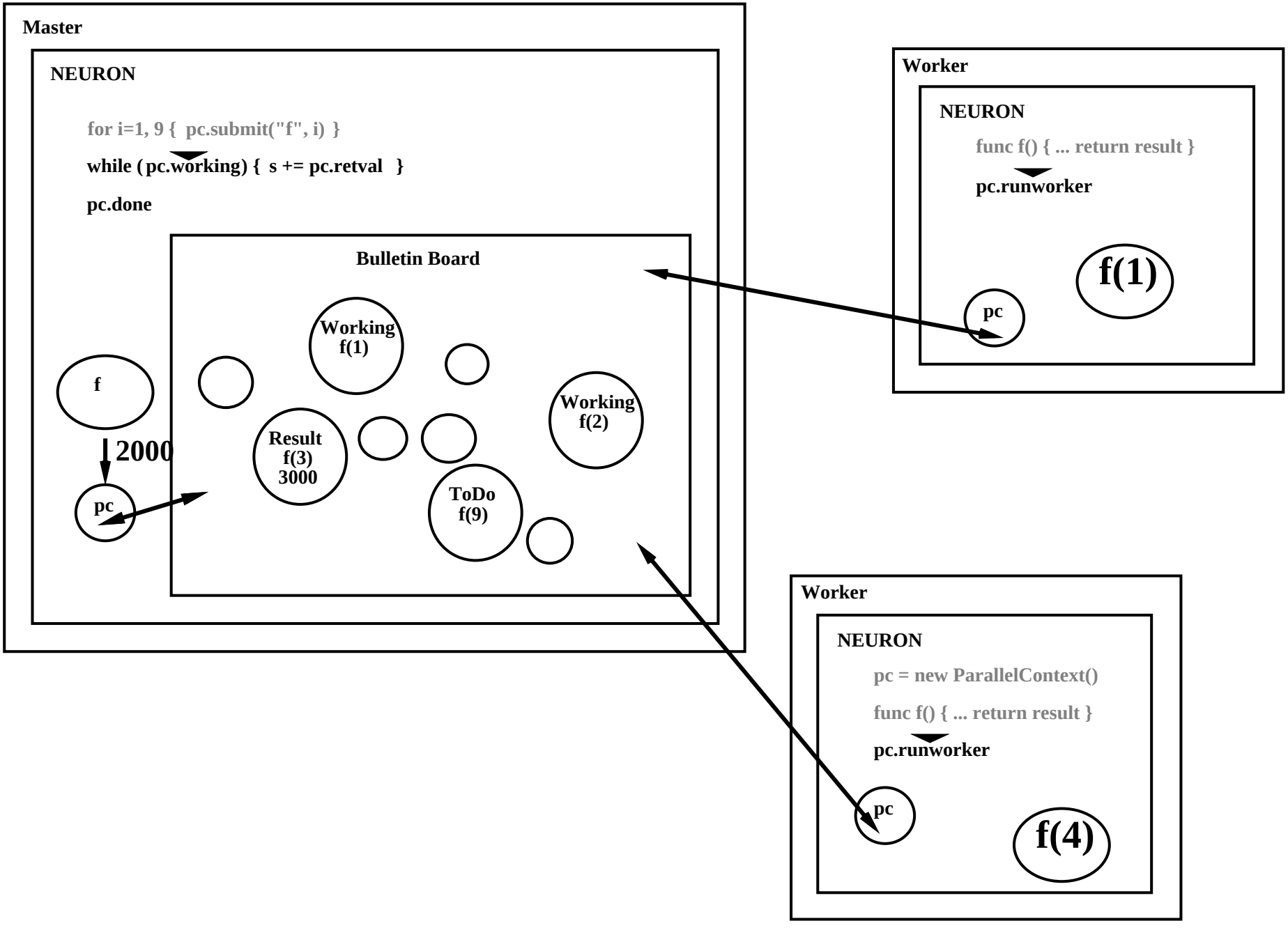

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
Working
f(1)
f(2)
Working
f(2)
Working
f(3)
ToDo
f(9)
pc
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
3000
f(3)

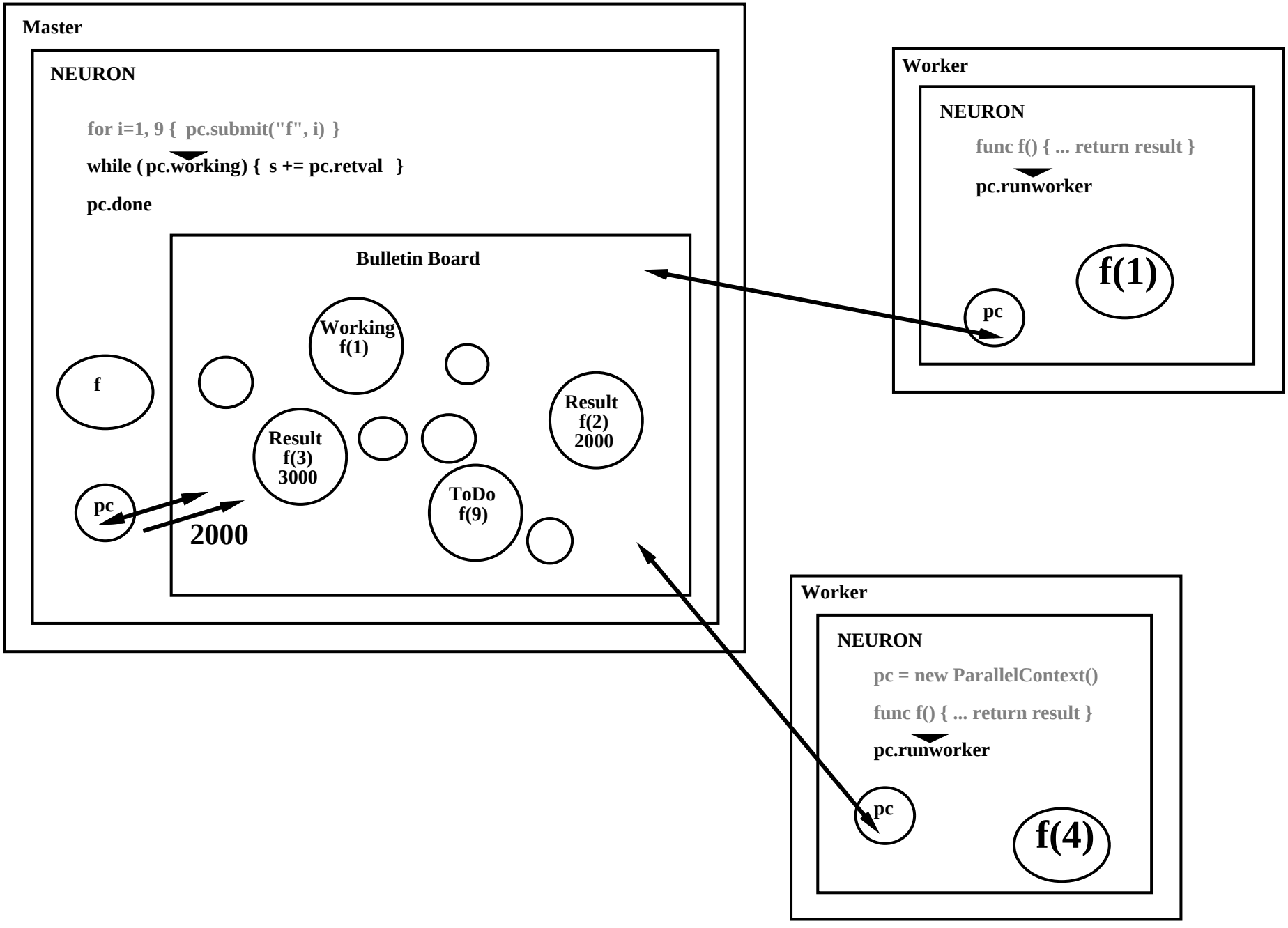

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
Working
f(1)
f(2)
Working
f(3)
Working
f(2)
ToDo
f(9)
pc
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
3000
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f

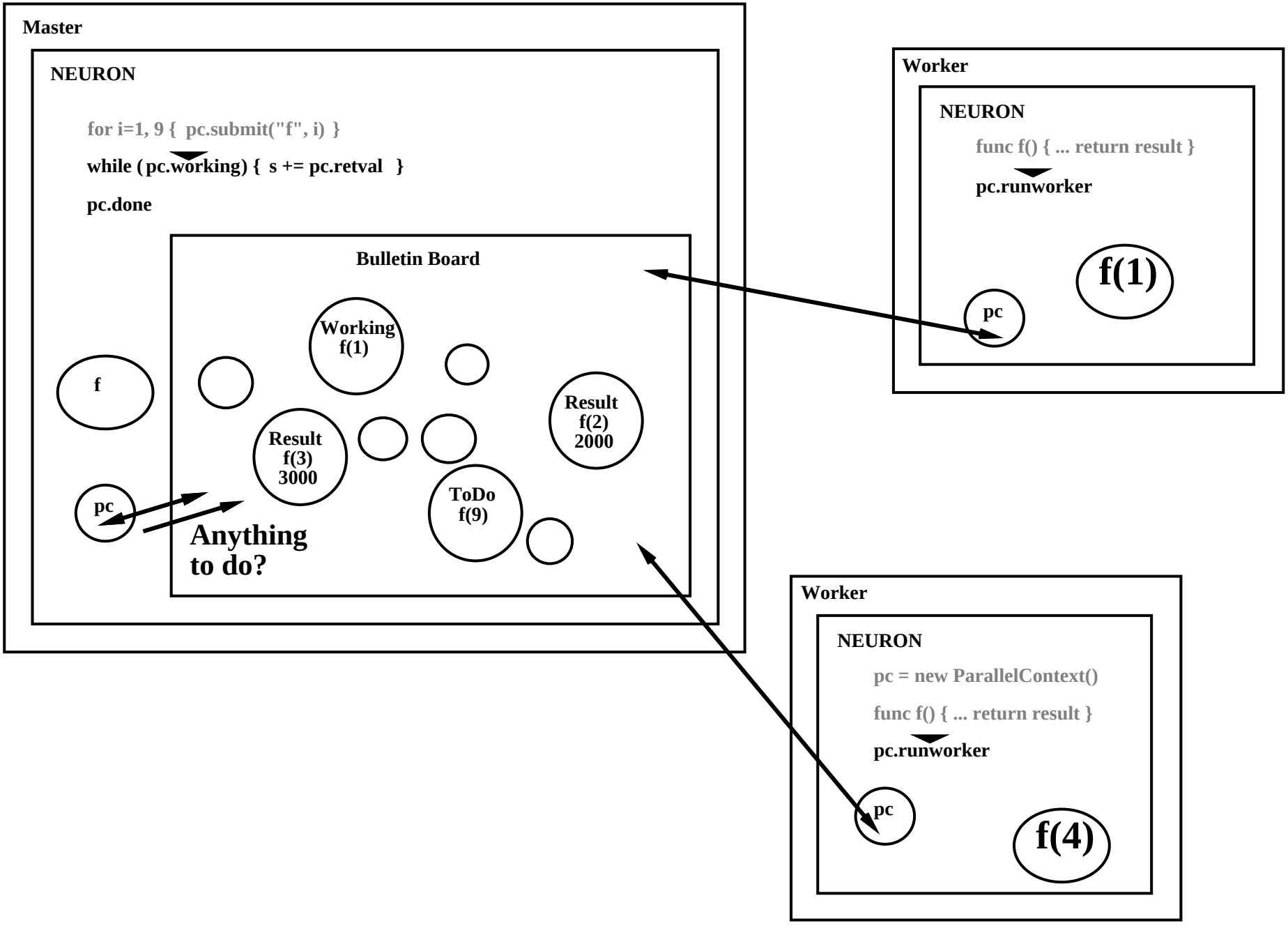

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
Working
f(1)
f(2)
Result
f(3)
3000
Working
f(2)
pc
ToDo
f(9)
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f

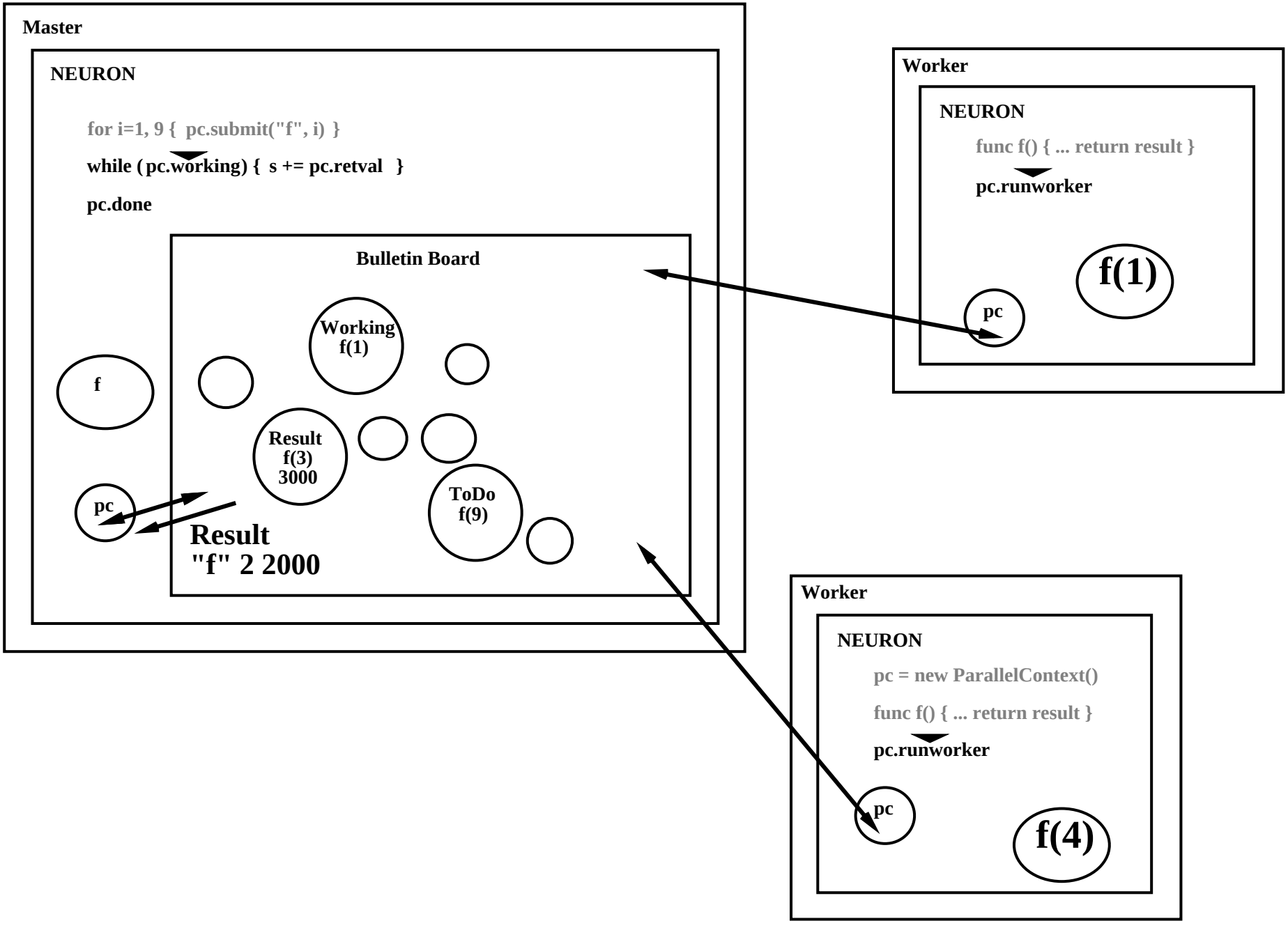

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
Working
f(1)
f(2)
Result
f(3)
3000
Working
f(2)
pc
ToDo
f(9)
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f
ToDo
"f" 4

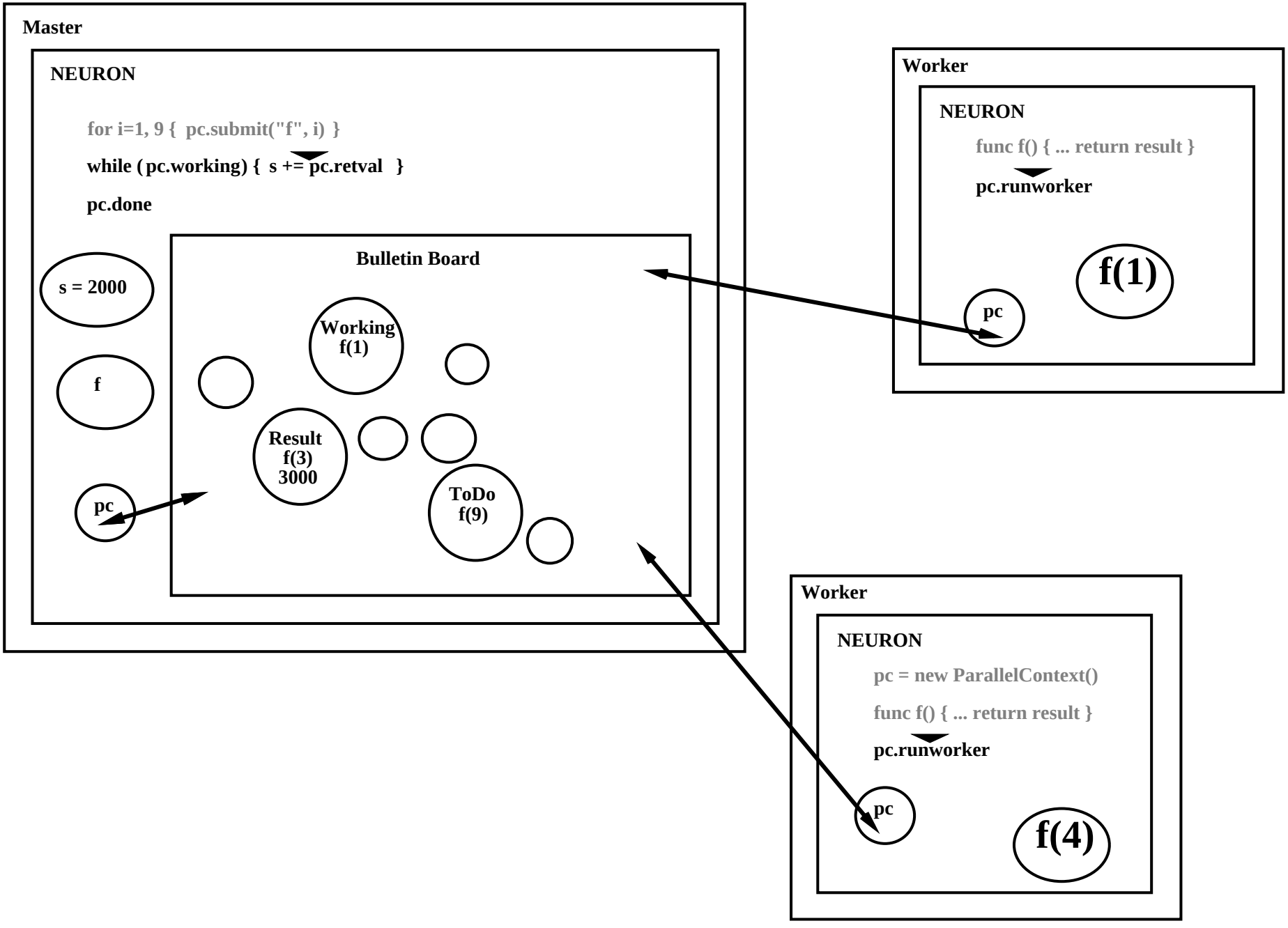

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
Working
f(1)
f(2)
Working
f(2)
Result
f(3)
3000
ToDo
f(9)
pc
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(4)

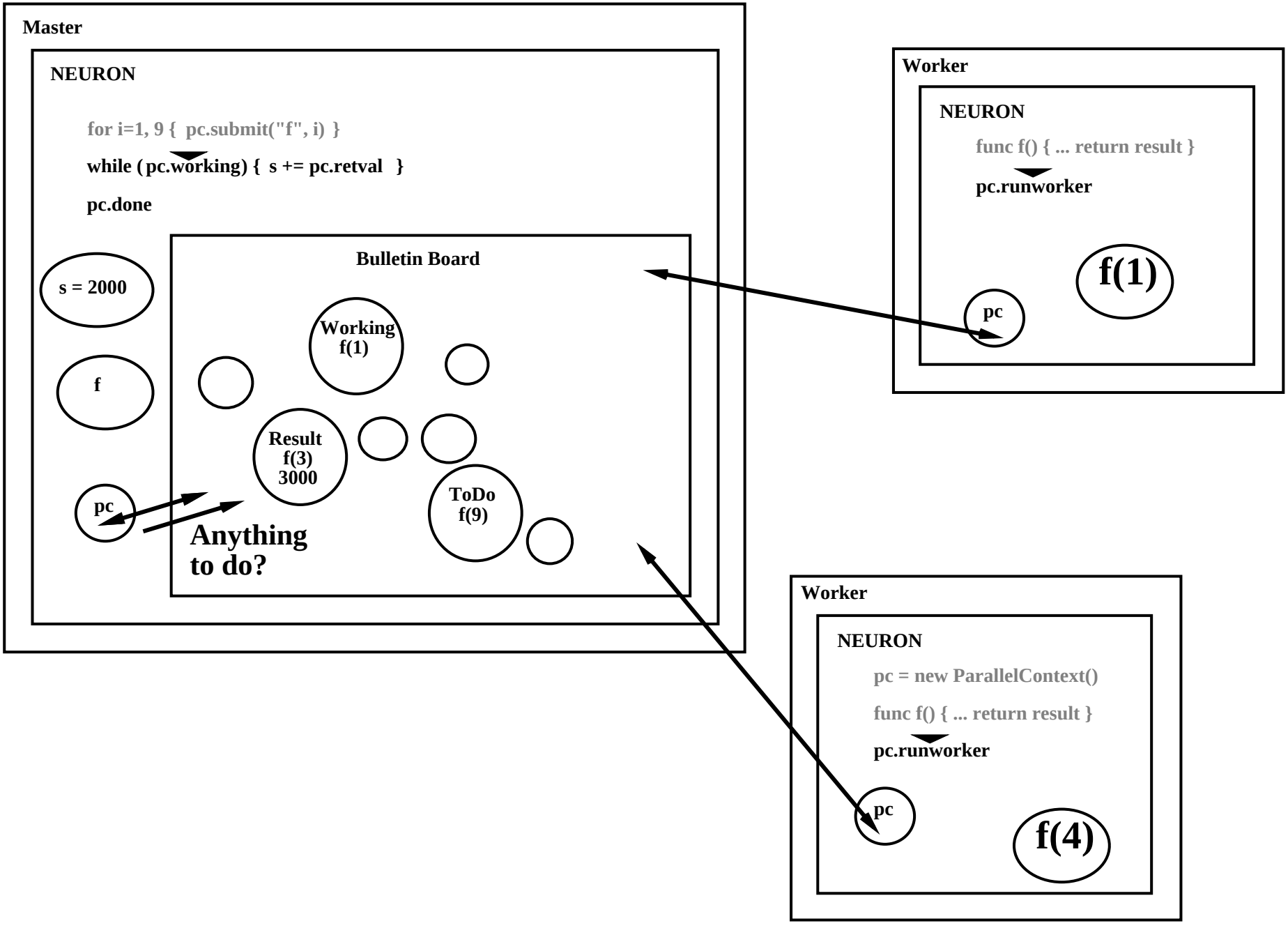

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
Working
f(1)
f
2000
pc
Result
f(3)
3000
Working
f(2)
ToDo
f(9)
Worker
NEURON
func f() { ... return result }
pc.runworker
pc
f(1)
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(4)

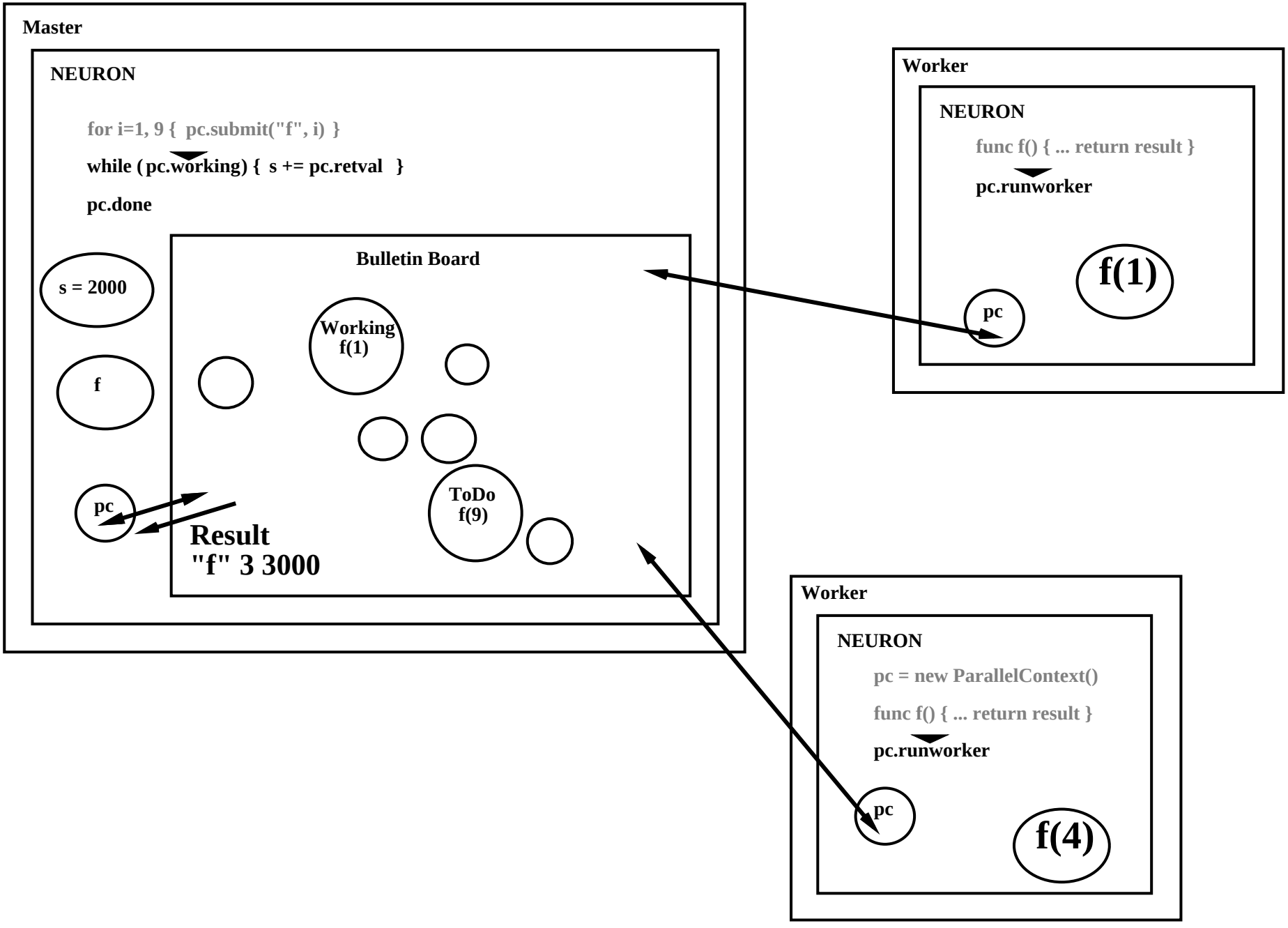

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
Working
f(1)
f
Result
f(3)
3000
Result
f(2)
2000
ToDo
f(9)
pc
2000
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(4)

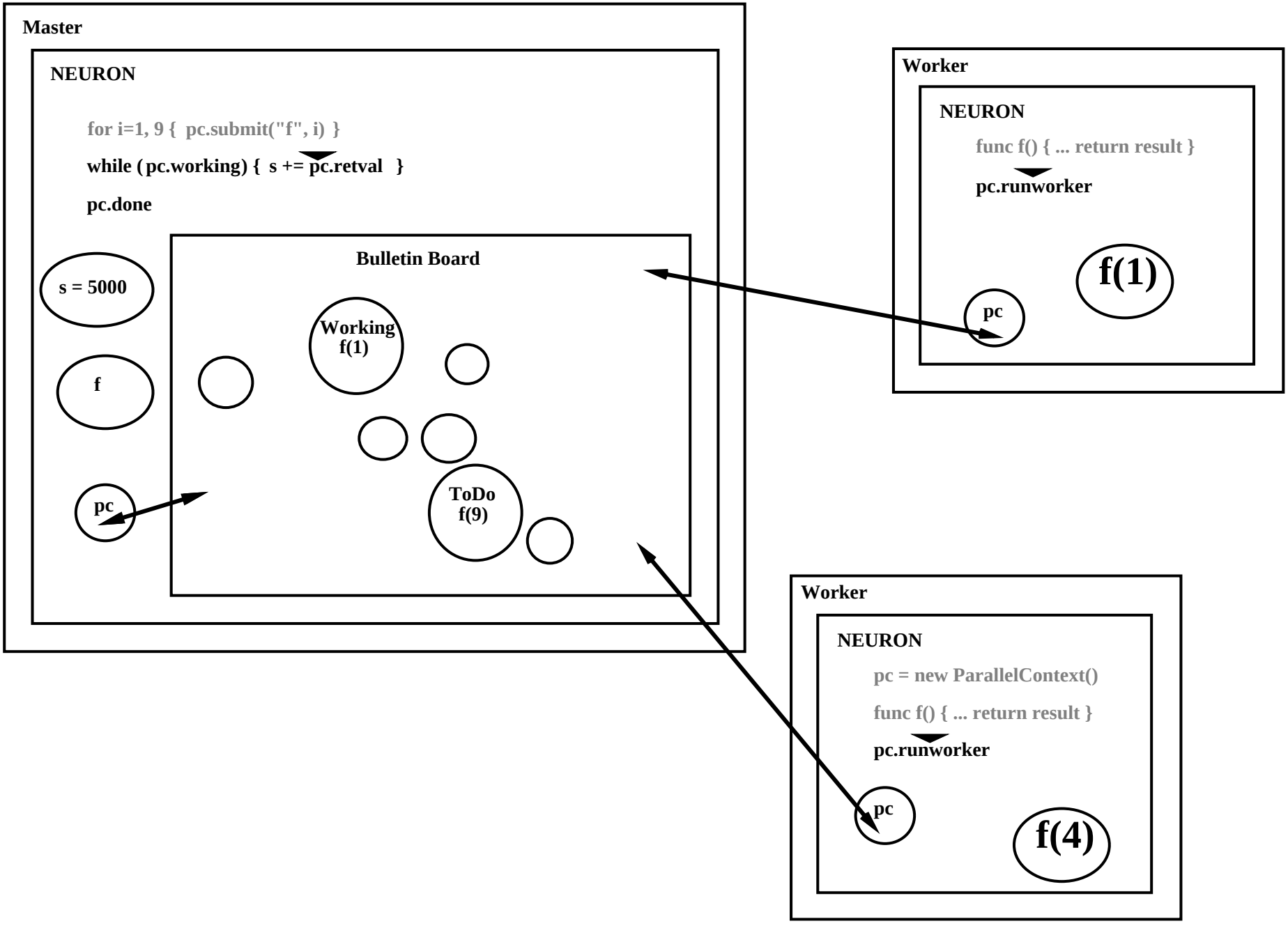

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
Working
f(1)
f
Result
f(3)
3000
Result
f(2)
2000
ToDo
f(9)
pc
Anything
to do?
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(4)

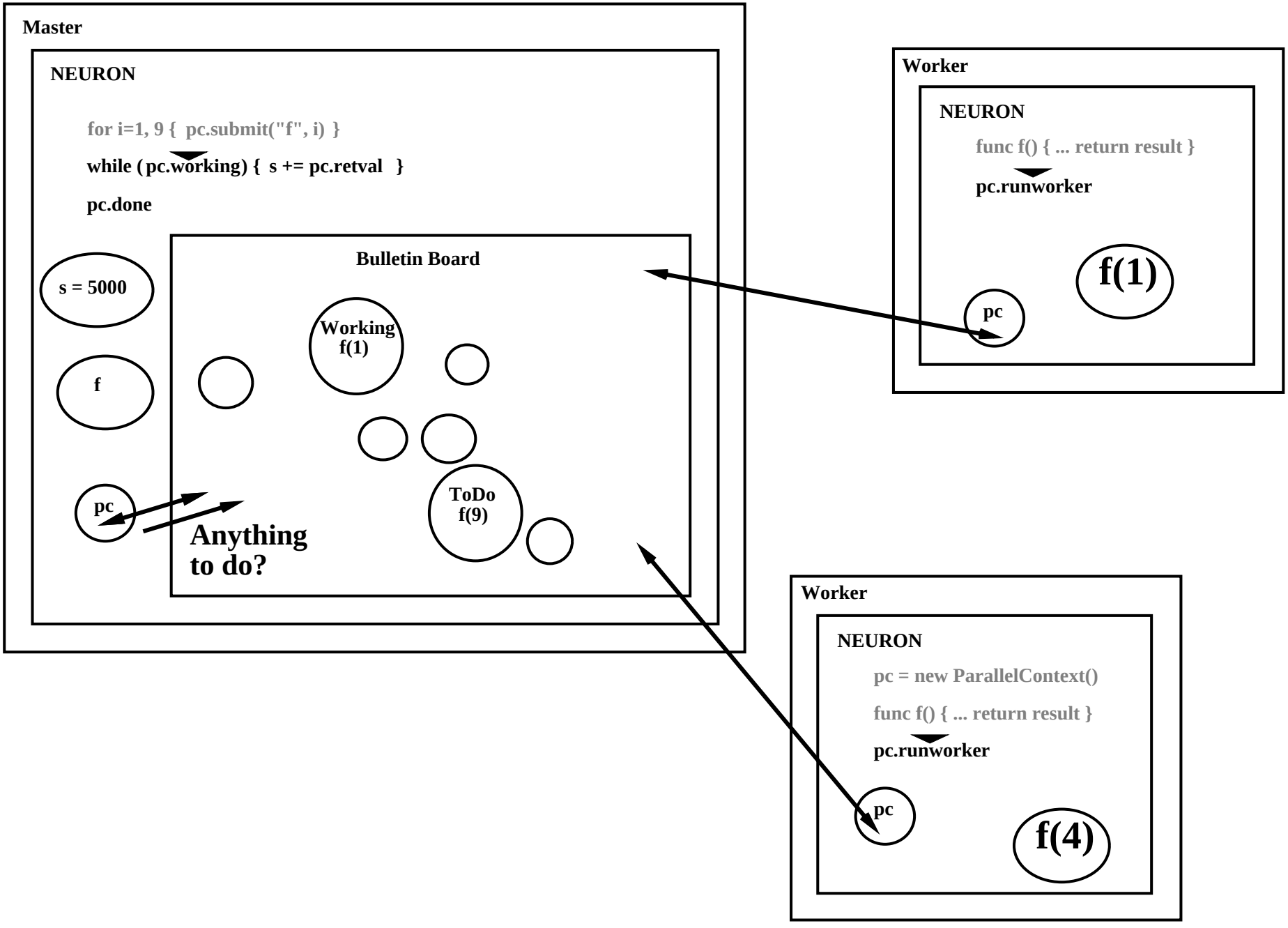

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
Working
f(1)
f
Result
f(3)
3000
pc
ToDo
f(9)
Result
"f" 2 2000
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(4)

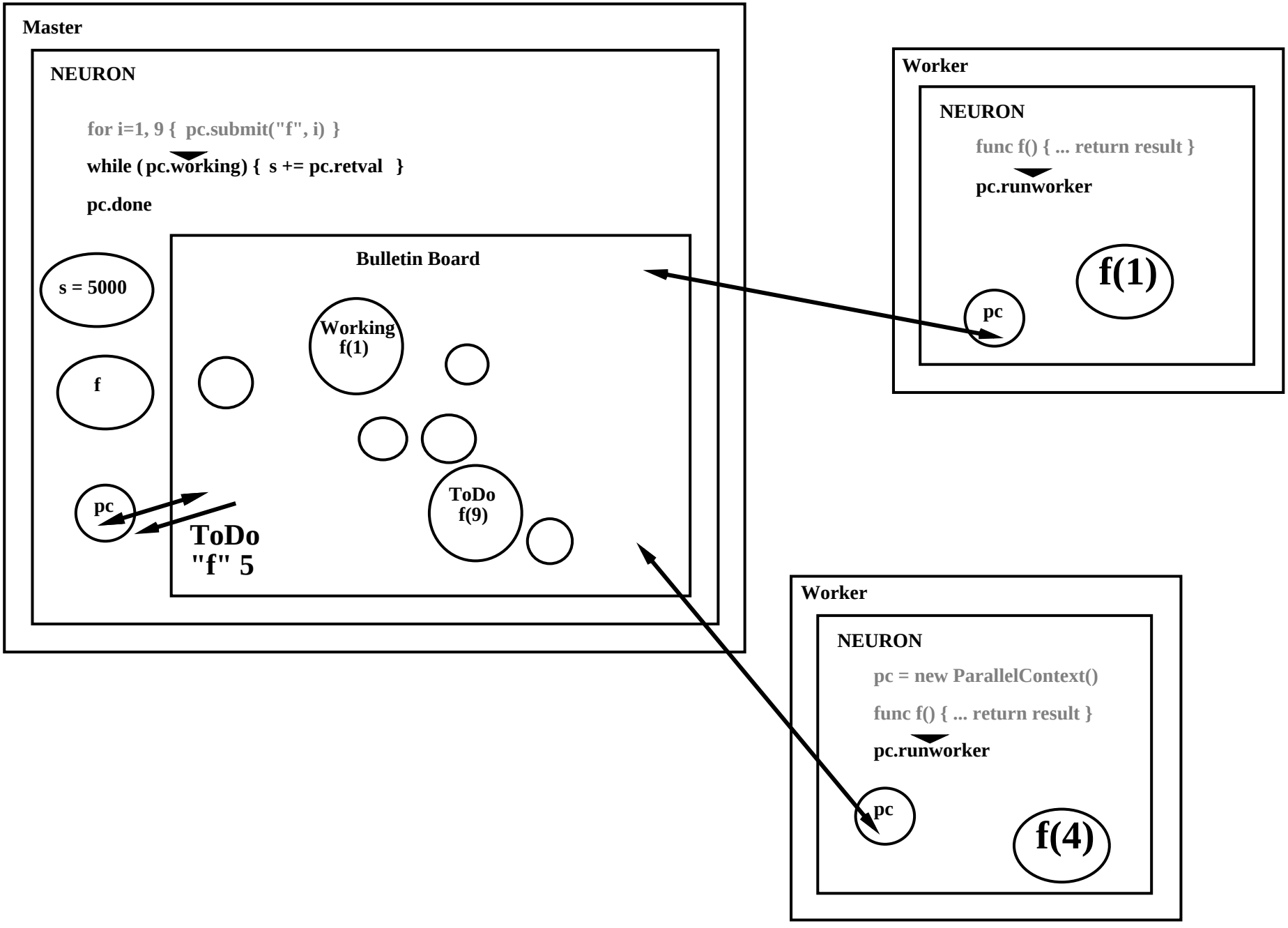

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 2000
f
pc
Working
f(1)
Result
f(3)
3000
ToDo
f(9)
Worker
NEURON
func f() { ... return result }
pc.runworker
pc
f(1)
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(4)

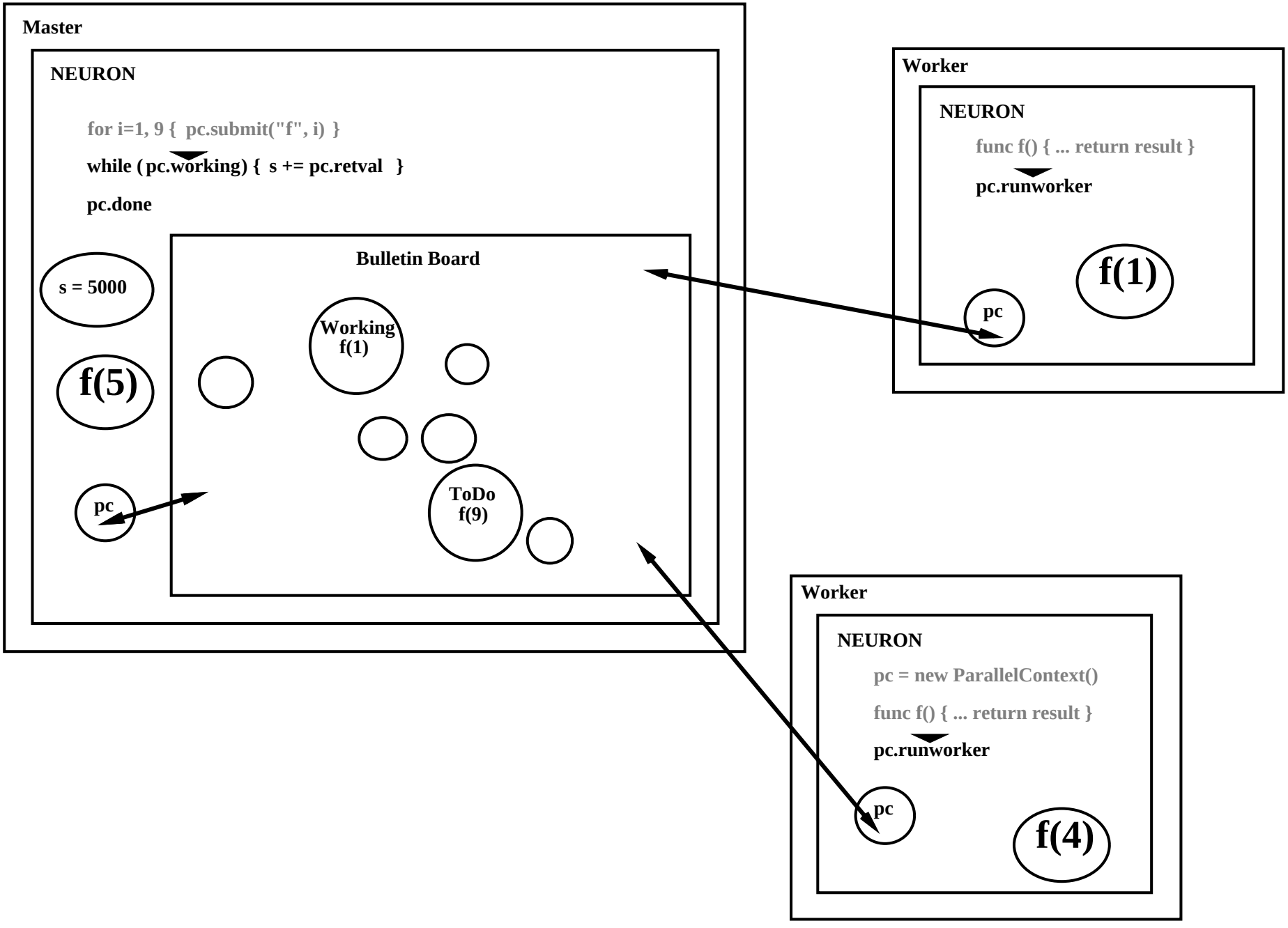

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 2000
f
Working
f(1)
Result
f(3)
3000
ToDo
f(9)
pc
Anything
to do?
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(4)

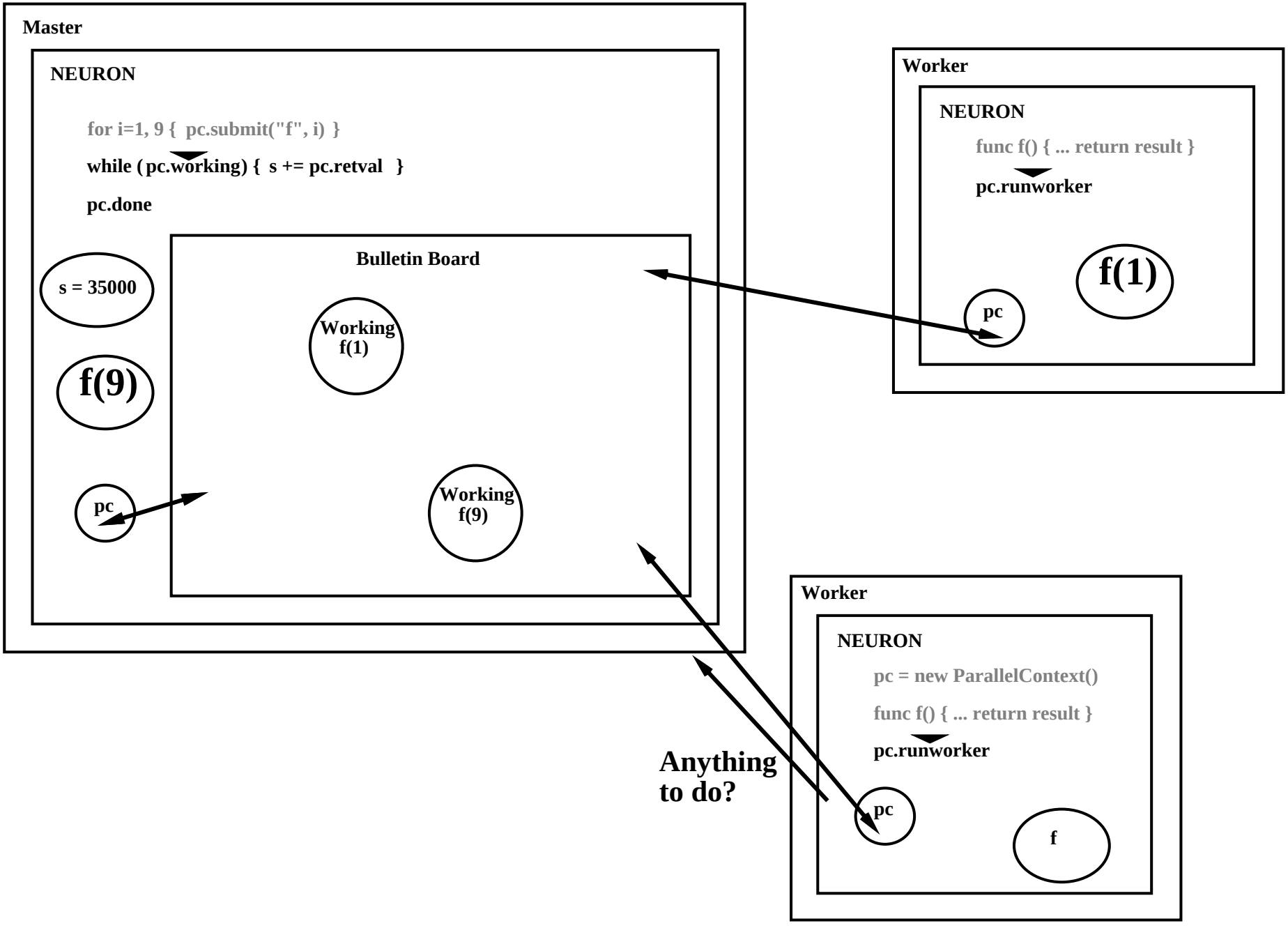

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 2000
f
pc
Working
f(1)
ToDo
f(9)
Result
"f" 3 3000
Worker
NEURON
func f() { ... return result }
pc.runworker
pc
f(1)
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(4)

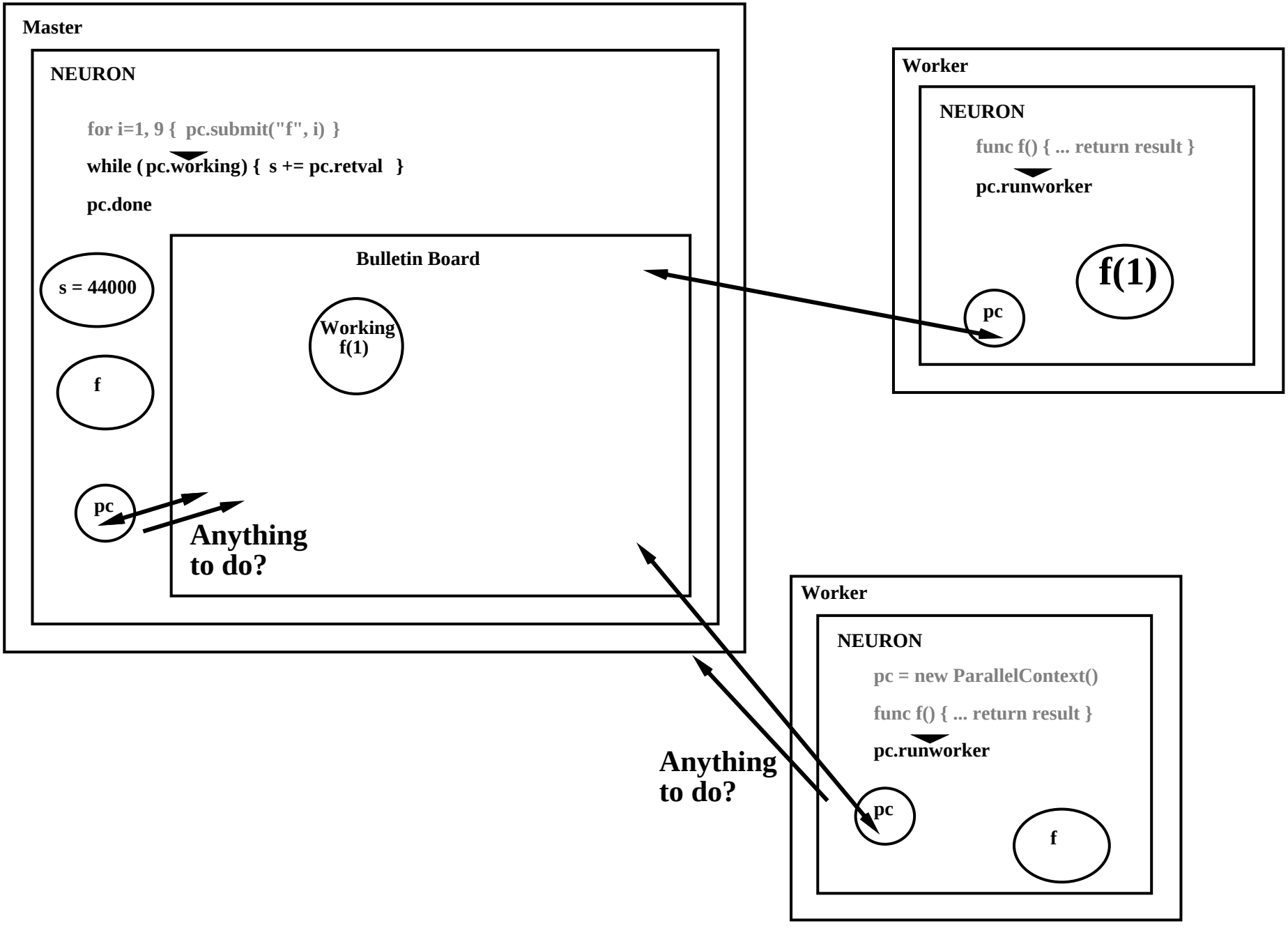

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 5000
f
pc
Working
f(1)
ToDo
f(9)
Worker
NEURON
func f() { ... return result }
pc.runworker
pc
f(1)
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(4)

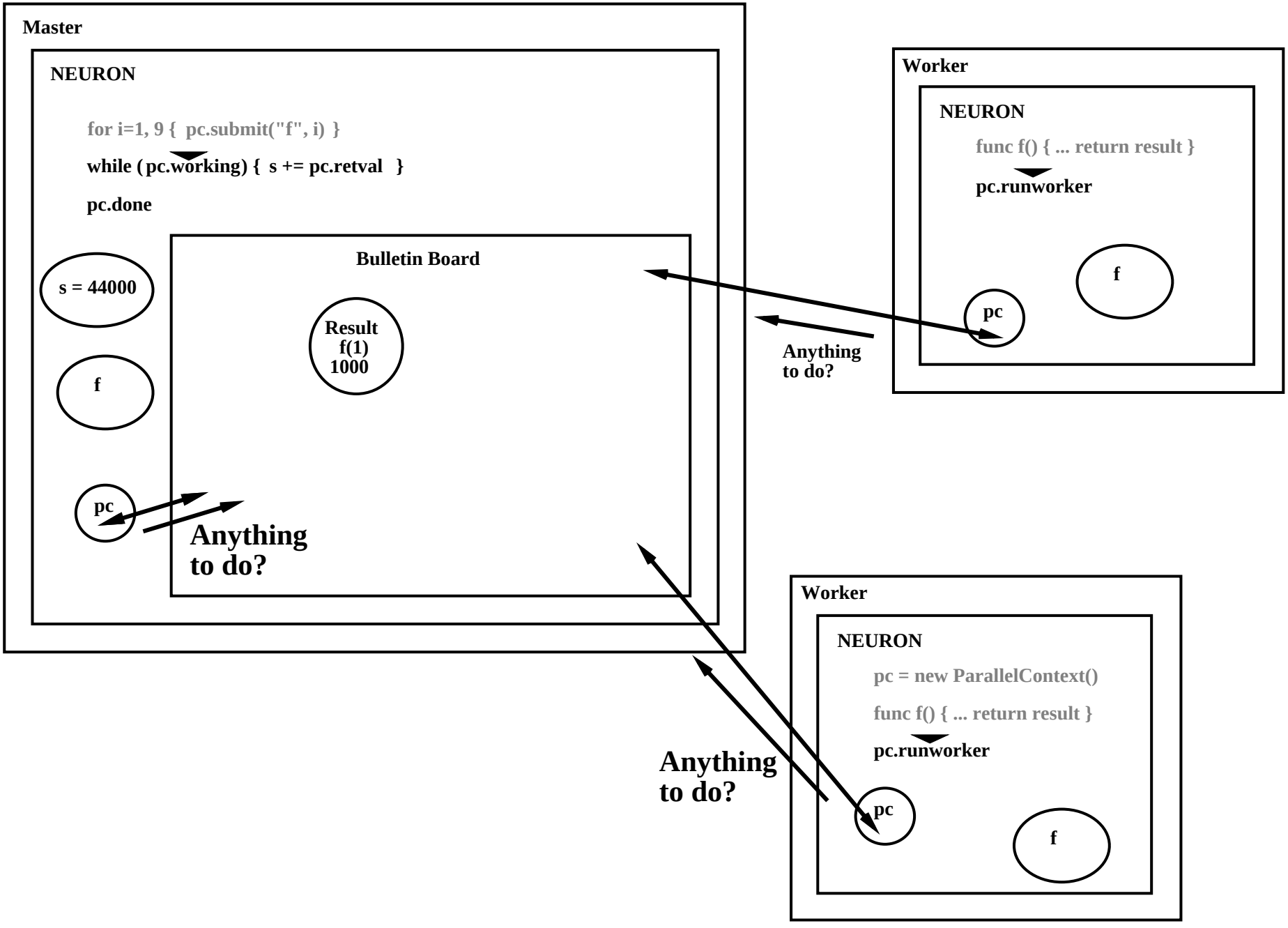

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 5000
f
Working
f(1)
ToDo
f(9)
pc
Anything
to do?
Worker
NEURON
func f() { ... return result }
pc.runworker
pc
f(1)
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(4)

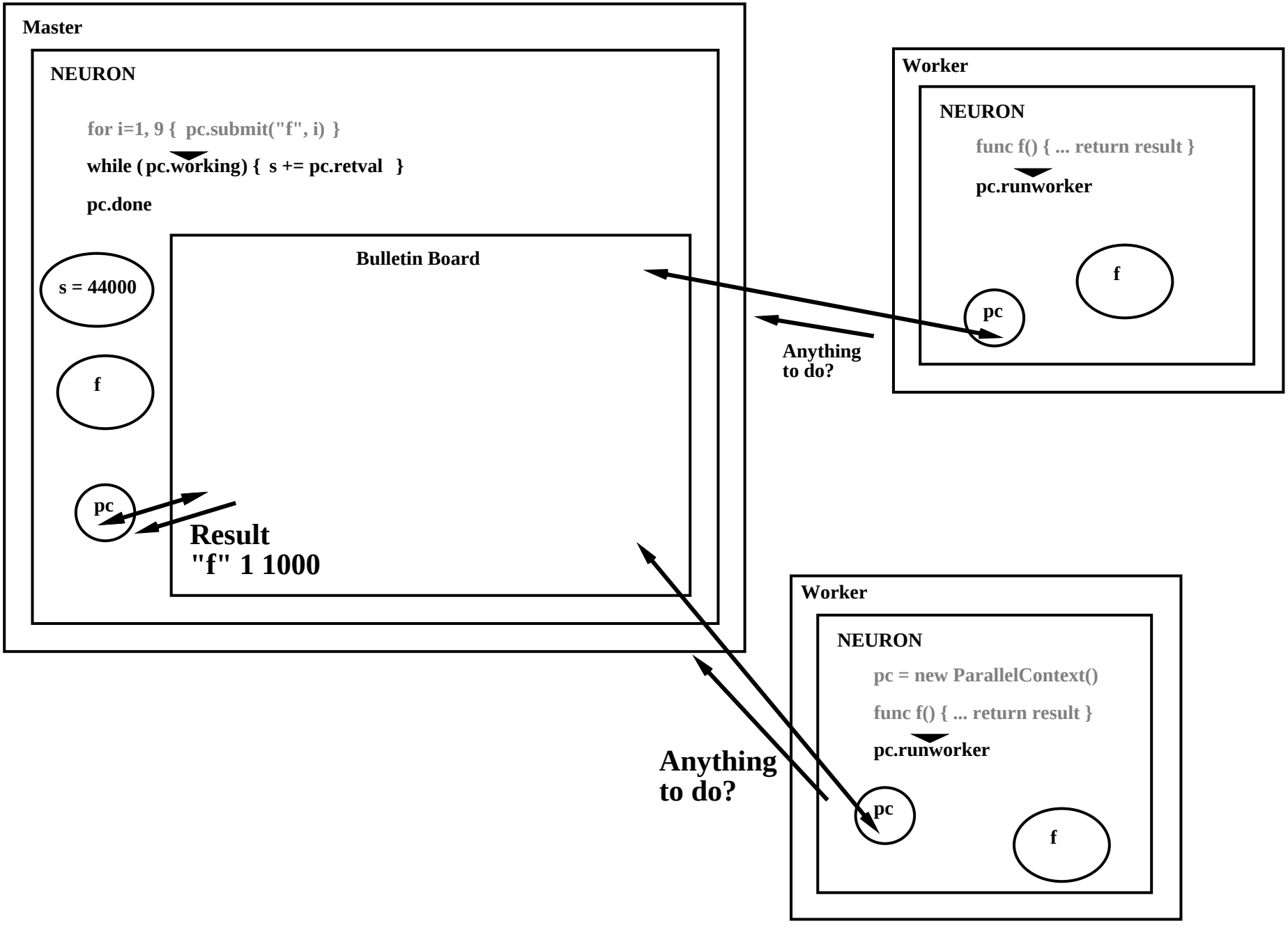

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 5000
f
pc
Working
f(1)
ToDo
f(9)
ToDo
"f" 5
Worker
NEURON
func f() { ... return result }
pc.runworker
pc
f(1)
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(4)

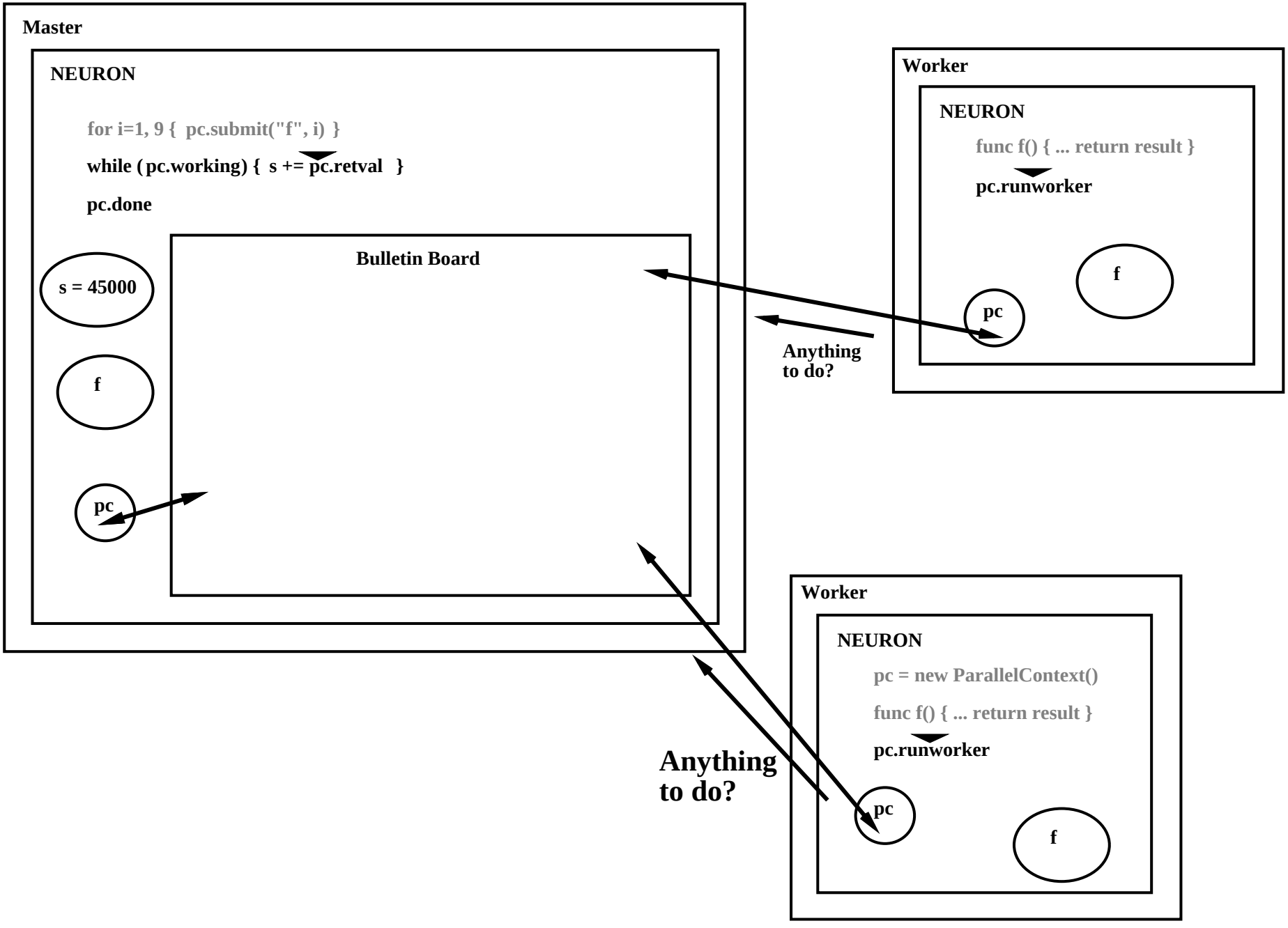

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 5000
f(5)
Working
f(1)
ToDo
f(9)
pc
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f(4)

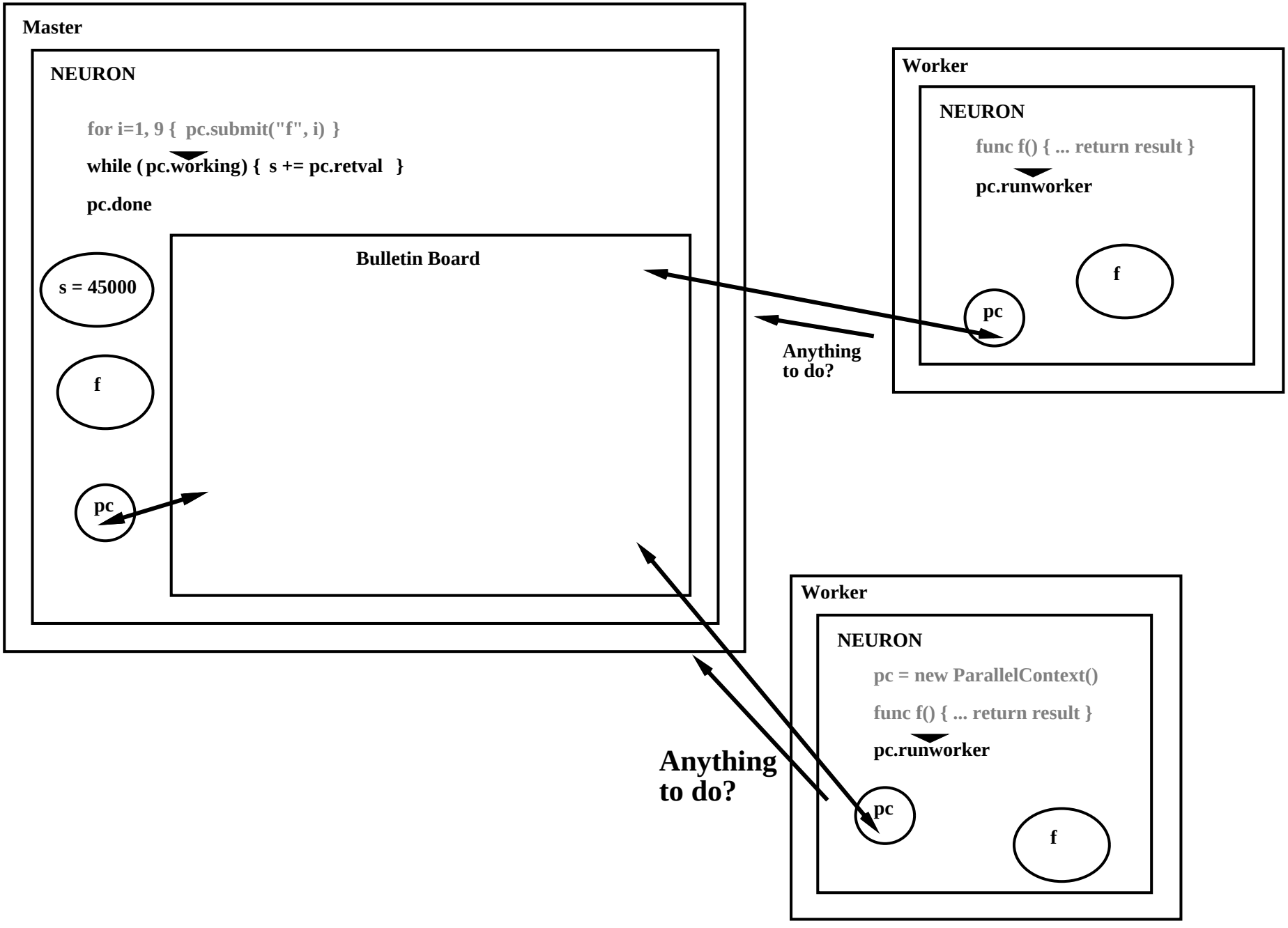

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
s = 35000
f(9)
pc
Bulletin Board
Working
f(1)
Working
f(9)
Worker
NEURON
func f() { ... return result }
pc.runworker
pc
f(1)
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f
Anything
to do?

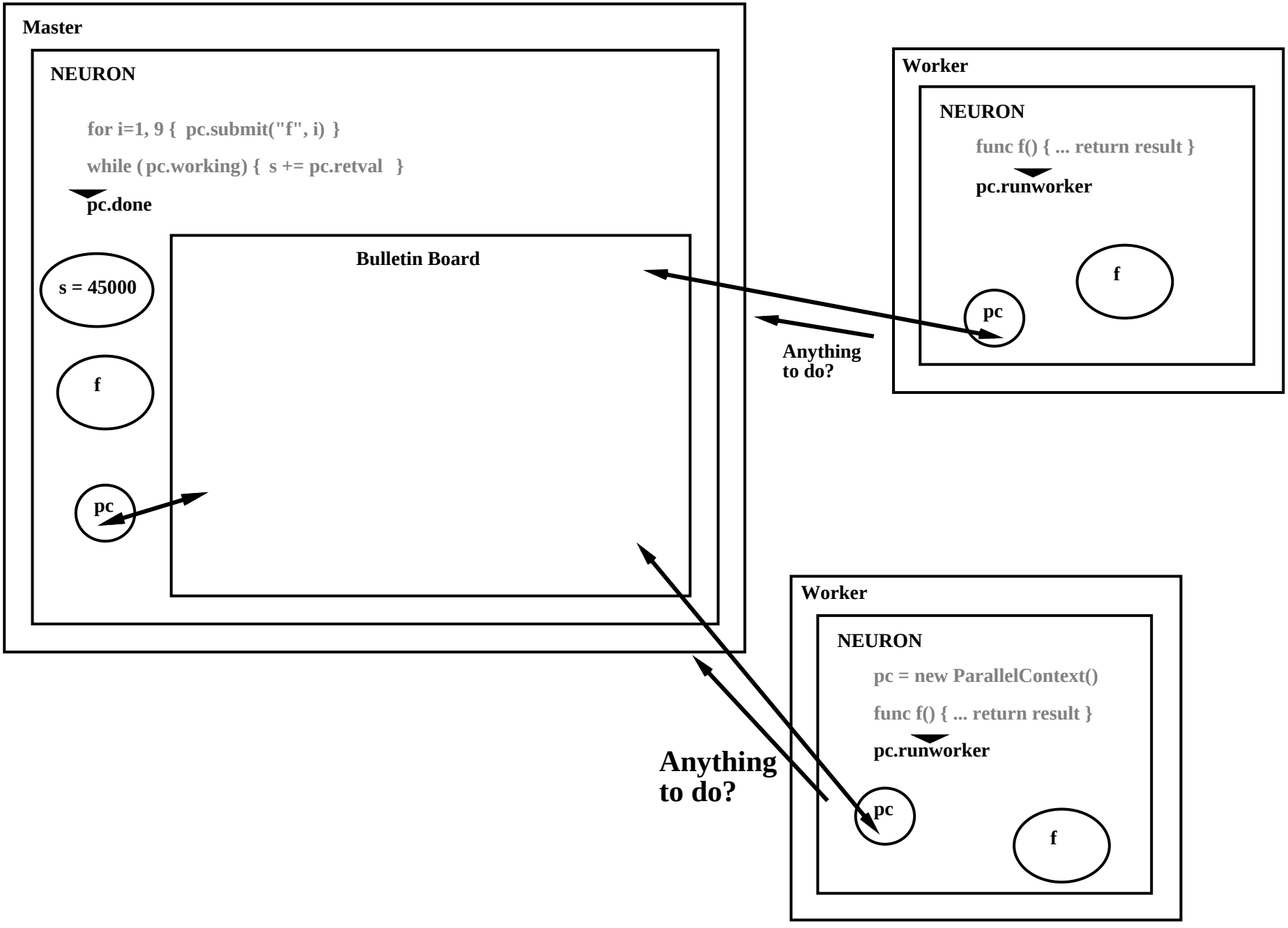

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
s = 44000
f
pc
Bulletin Board
Working
f(1)
Anything
to do?
Worker
NEURON
func f() { ... return result }
pc.runworker
f(1)
pc
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f
Anything
to do?

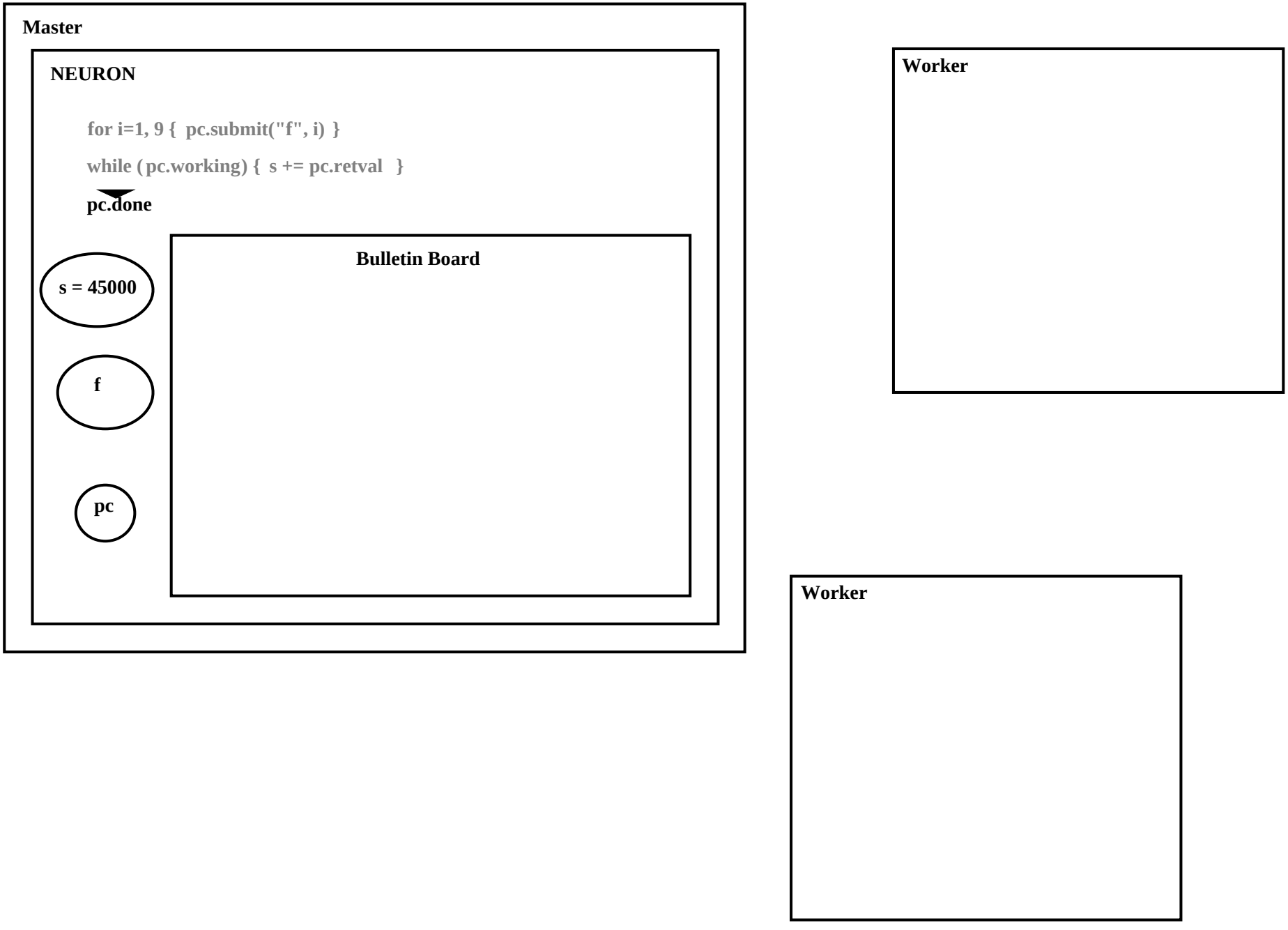

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 44000
Result
f(1)
1000
f
pc
Anything
to do?
Worker
NEURON
func f() { ... return result }
pc.runworker
f
pc
Anything
to do?
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
Anything
to do?
pc
f

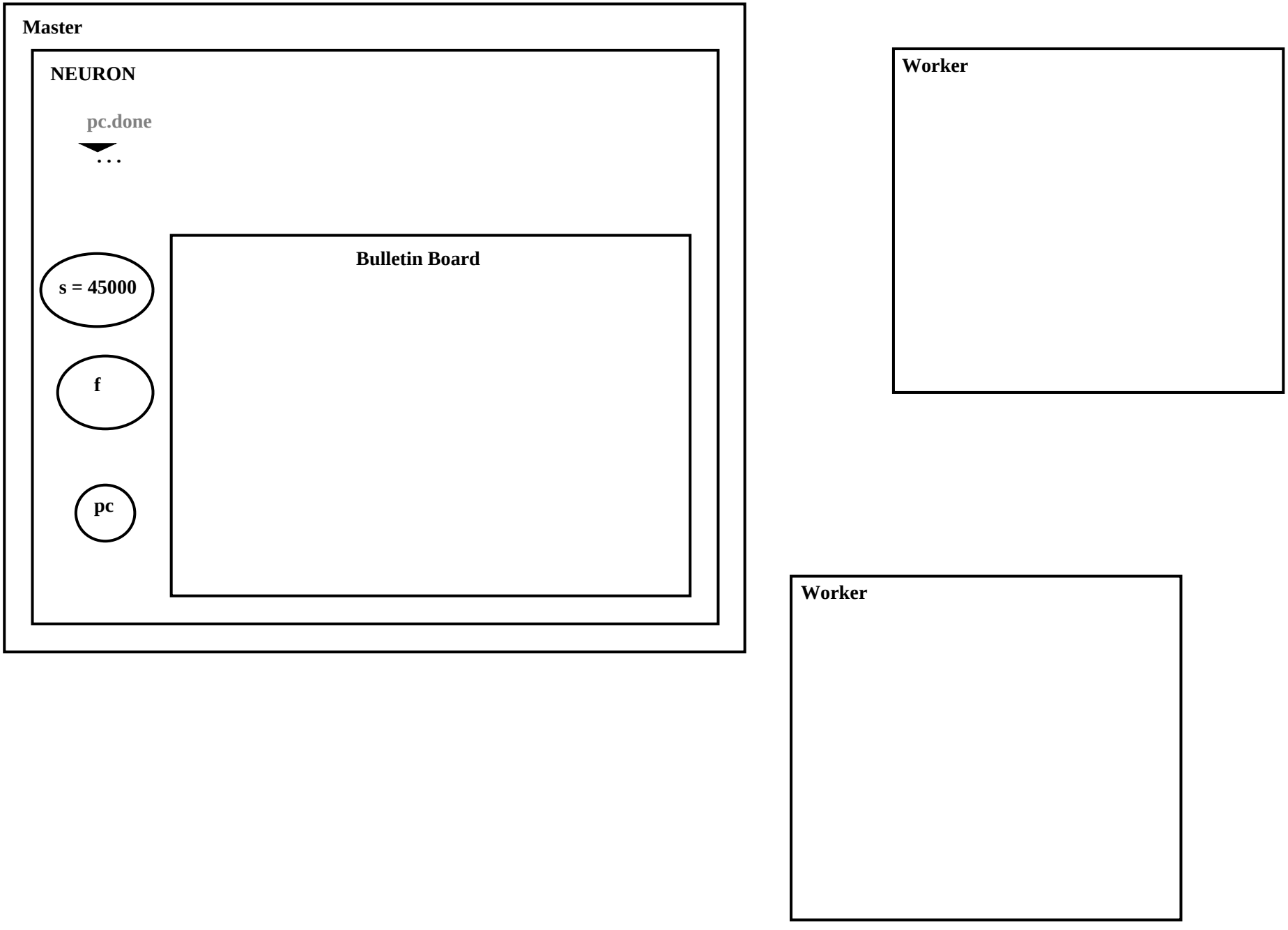

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 44000
f
pc
Result
"f" 1 1000
Worker
NEURON
func f() { ... return result }
pc.runworker
f
pc
Anything
to do?
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f
Anything
to do?

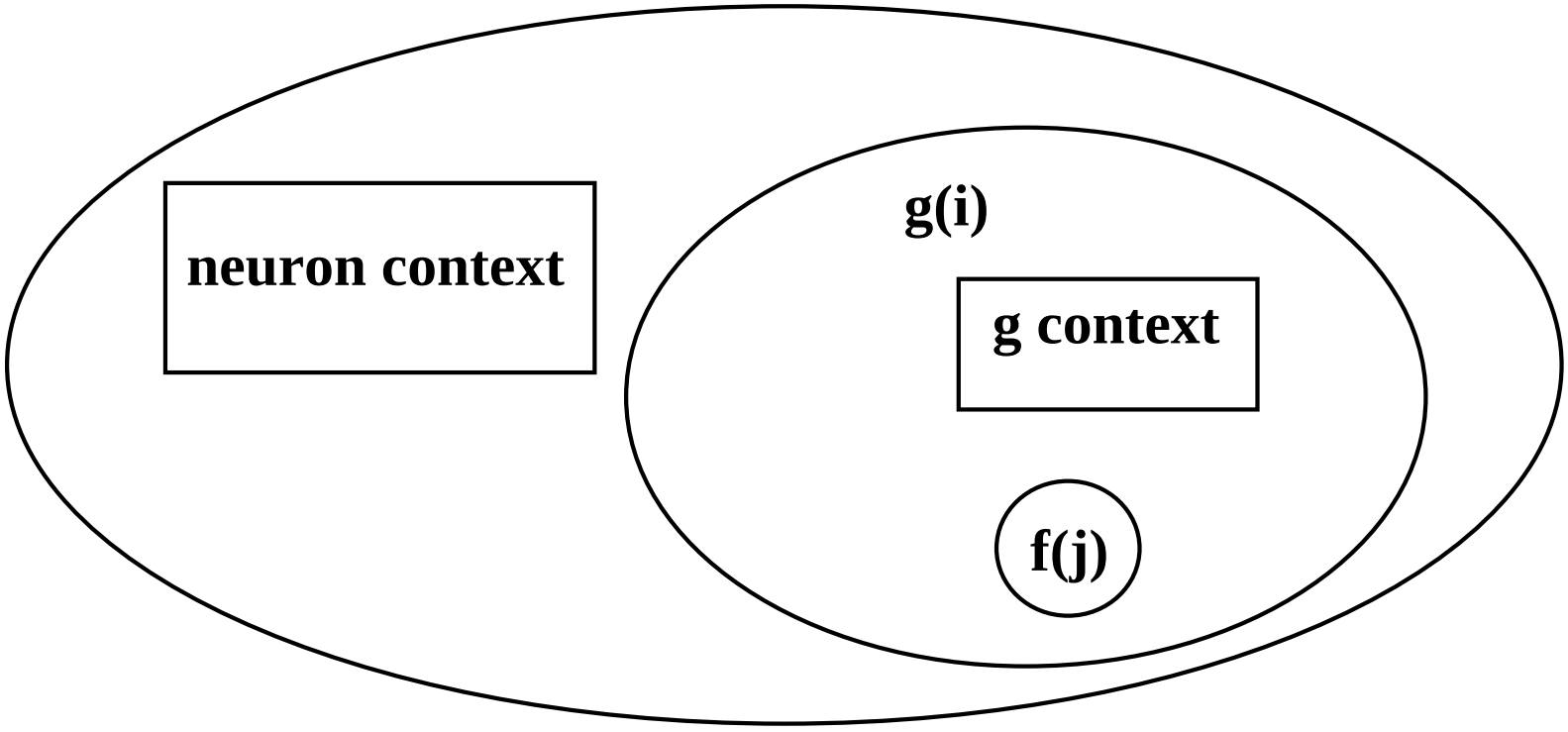

Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 45000
f
pc
Anything
to do?
Worker
NEURON
func f() { ... return result }
pc.runworker
f
pc
Anything
to do?
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f


Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 45000
f
pc
Worker
NEURON
func f() { ... return result }
pc.runworker
f
pc
Anything
to do?
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f
Anything
to do?


Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 45000
f
pc
Worker
NEURON
func f() { ... return result }
pc.runworker
f
pc
Anything
to do?
Worker
NEURON
pc = new ParallelContext()
func f() { ... return result }
pc.runworker
pc
f
Anything
to do?


Master
NEURON
for i=1, 9 { pc.submit("f", i) }
while (pc.working) { s += pc.retval }
pc.done
Bulletin Board
s = 45000
f
pc
Worker
Worker


Master
NEURON
pc.done
. . .
s = 45000
f
pc
Bulletin Board
Worker
Worker



post → **Bulletin board**

take
look ← **Bulletin board**
look_take

context("stmt") : stmt executed on every worker

Some Performance Results

150 runs

one machine

150 tasks user=40.68 sys=0.04 elapsed=44.27 92%

15 machines

master: 10 tasks user=3.07 sys=0.35 elapsed=6.58 47%
14 machines like
[tc0003] 10 tasks user=2.91 sys=0.03 elapsed=6.16 47%

user time ratio: 13
elapsed time ratio: 7

1500 runs

15 machines

master: 88 tasks user=25.35 sys=2.77 elapsed=32.05 79%
12 machines like
[t100002] 101 tasks user=27.69 sys=0.09 elapsed=31.6 88%
2 machines like
[t3c0002] 100 tasks user=27.78 sys=0.07 elapsed=31.54 88%

elapsed time ratio
(40.68*10 + 4)/32.05 = 13