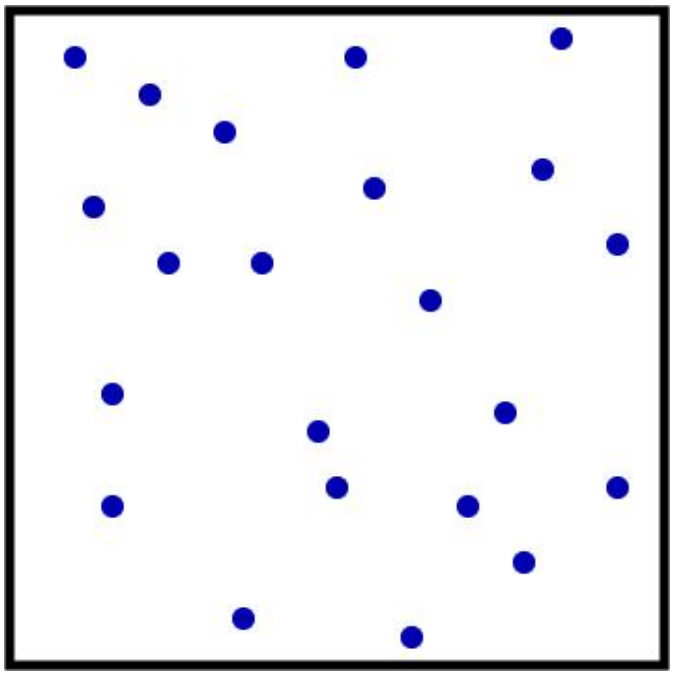
Monte Carlo for particle systems

Interacting molecules in a volume

$$\langle A \rangle = \frac{1}{Z} \int \prod_{i=1}^{N} dx_i^d A(\{x_i\}) \mathrm{e}^{-E(\{x_i\})/k_B T}$$

$$Z = \int \prod_{i=1}^{N} dx_i^d \mathrm{e}^{-E(\{x_i\})/k_B T}$$



We will consider simple spherical particles (atoms)

$$E(\{x_i\}) = \sum_{i=1}^{N} U(\vec{x}_i) + \sum_{i \neq j} V(\vec{x}_i, \vec{x}_j)$$

V=V(r); spherically symmetric. More complicated molecules involve more difficult energy calculations

Long-range interactions (e.g. Coulomb) lead to N^2 scaling of the computational effort (CPU time).

Lennard-Jones (e.g., for noble gases):

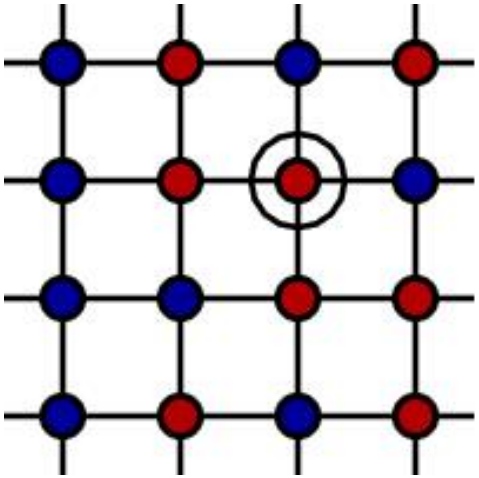
$$V(r) = \frac{a}{r^{12}} - \frac{b}{r^6}$$

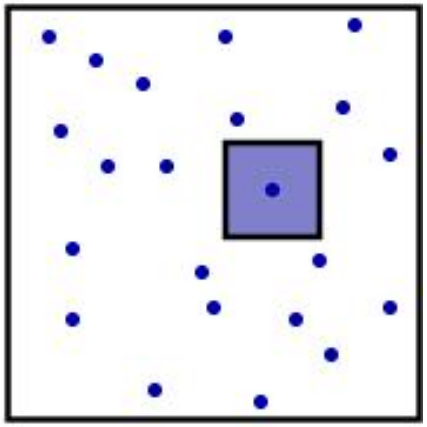
- can be truncated

Metropolis Monte Carlo scheme

Principle same as in Ising simulation; update involves

- selecting a particle at random
- attempt to move it within a box
- calculate energy change (more complicated than Ising)





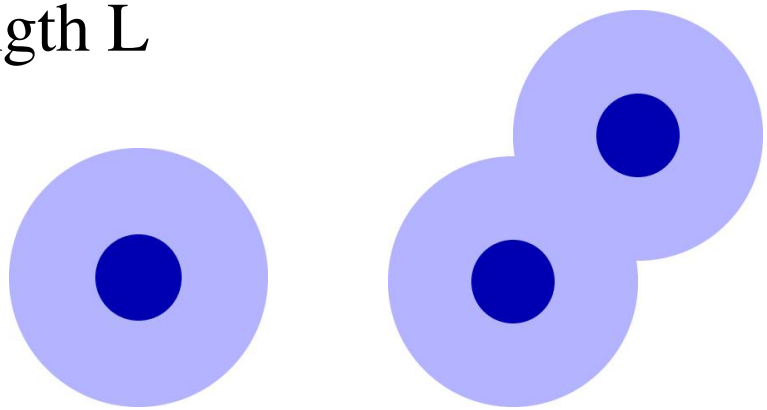
Adjust size of the box so that the acceptance rate is close to 50%

Typically periodic boundary conditions are used

Program for molecules with a simplified potential

N particles in periodic 3D box of length L

$$V(r) = \begin{cases} \infty, & r <\leq r_1 \\ -V, & r_1 < r \leq r_2 \\ 0, r > r_2 \end{cases}$$



Energy = -V times number of "overlapping" particle pairs

Module with system parameters and variables

```
module systemvariables
real, parameter :: r2mol1=1.0        ! square of r1
real, parameter :: r2mol2=9.0        ! square of r2
real, parameter :: potential=-1.0
integer :: n                 ! number of particles
real    :: l                 ! system length
real    :: temp              ! temperature
real    :: delta             ! max x,y,z change = +/- delta/2
real, allocatable :: xyz(:,:)    ! particle coordinates
```

Main program

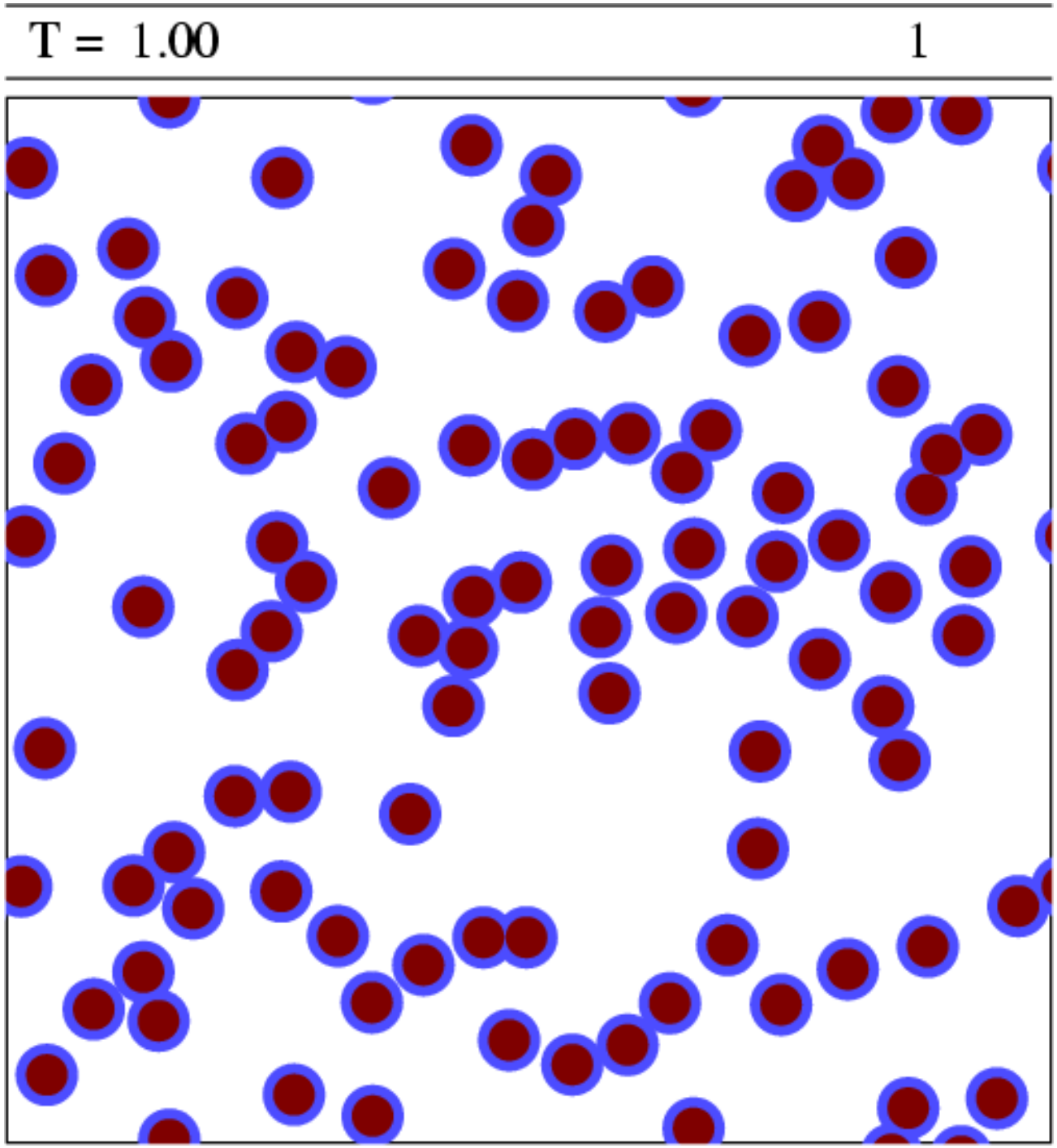
Highest T = `tmax`, # of T = `nt`, delta-T = `dt`

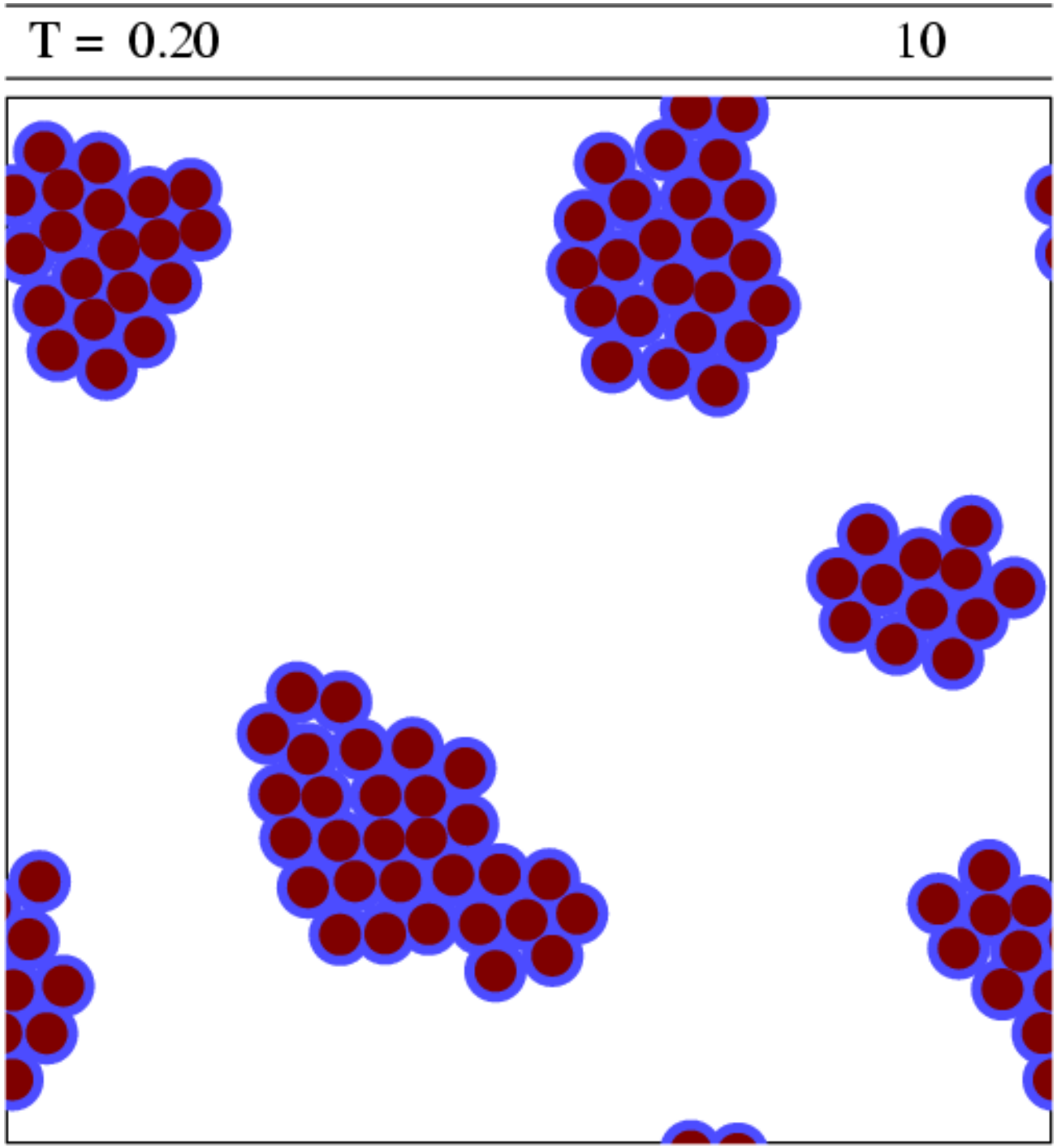
```
do it=0,nt-1
   temp=tmax-dt*it

   arate=0.
   do i=1,steps
      call mcstep(accepted)
      arate=arate+accepted
      if (mod(i,steps/20)==0) call adjustedelta(steps/20,arate,delta,l)
   end do

   do j=1,bins
      call resetbindata
      do i=1,steps
         call mcstep(accepted)
         call measure(accepted)
      end do
      call writebindata(it,steps)
   end do

end do
```

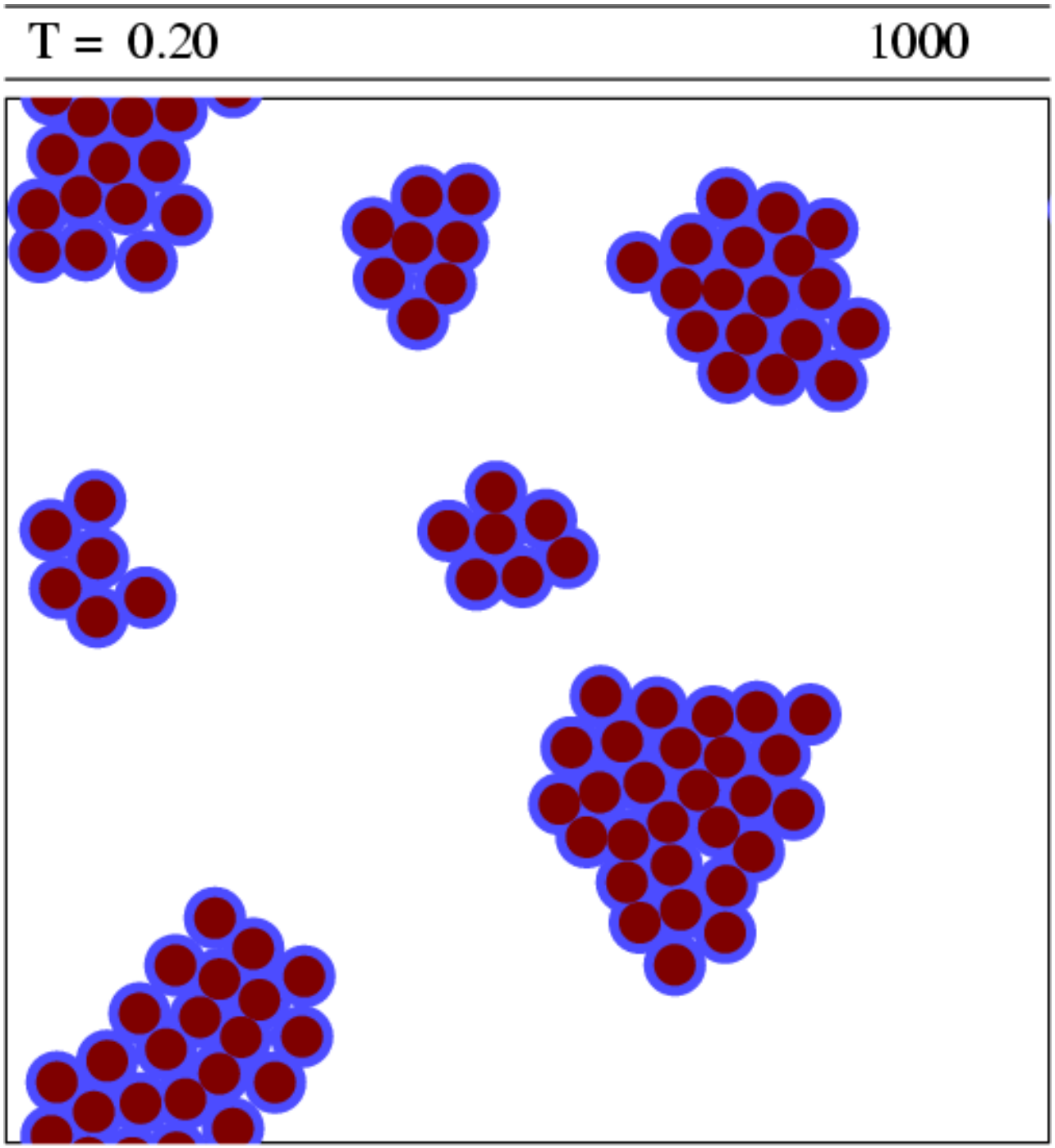
```
subroutine mcstep(accepted)

accepted=0.
do j=1,n
   x0=xyz(1,j); y0=xyz(2,j); z0=xyz(3,j)
   call newcoordinates(l,delta,x0,y0,z0,x1,y1,z1)

   n0=0; n1=0   ! For computing acceptance probability
   do i=1,n
      if (i/=j) then
         dx=abs(x1-xyz(1,i)); dx=min(dx,l-dx)
         dy=abs(y1-xyz(2,i)); dy=min(dy,l-dy)
         dz=abs(z1-xyz(3,i)); dz=min(dz,l-dz)
         r2=dx**2+dy**2+dz**2
         if (r2 < r2mol1) goto 1  ! Not allowed, cancel
         if (r2 < r2mol2) n1=n1+1
         dx=abs(x0-xyz(1,i)); dx=min(dx,l-dx)
         dy=abs(y0-xyz(2,i)); dy=min(dy,l-dy)
         dz=abs(z0-xyz(3,i)); dz=min(dz,l-dz)
         r2=dx**2+dy**2+dz**2
         if (r2 < r2mol2) n0=n0+1
      endif
   enddo
```

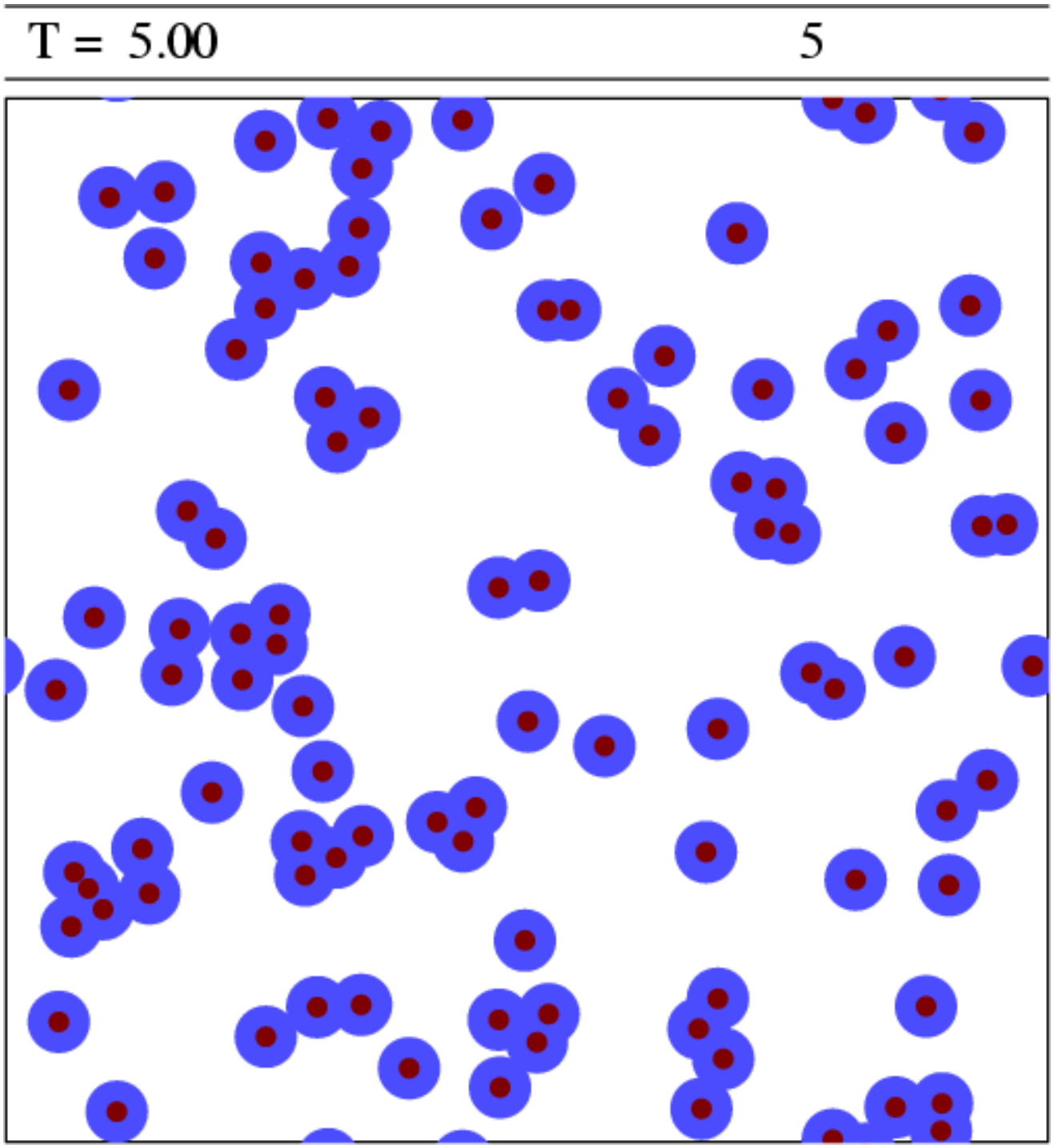
```
   accept=.true.
   if (n1 < n0) then
      call random_number(r)
      if (r > exp(-potential*real(n1-n0)/temp)) accept=.false.
   endf
   if (accept) then
      xyz(1,j)=x1; xyz(2,j)=y1; xyz(3,j)=z1
      accepted=accepted+1.
   endif
   1 continue
enddo
accepted=accepted/real(n)


subroutine newcoordinates(l,delta,x0,y0,z0,x1,y1,z1)
call random_number(r); x1=x0+delta*(r-0.5)
if (x1 < 0) x1=x1+l; if (x1 > l) x1=x1-l
call random_number(r); y1=y0+delta*(r-0.5)
if (y1 < 0) y1=y1+l; if (y1 > l) y1=y1-l
call random_number(r); z1=z0+delta*(r-0.5)
if (z1 < 0) z1=z1+l; if (z1 > l) z1=z1-l
```

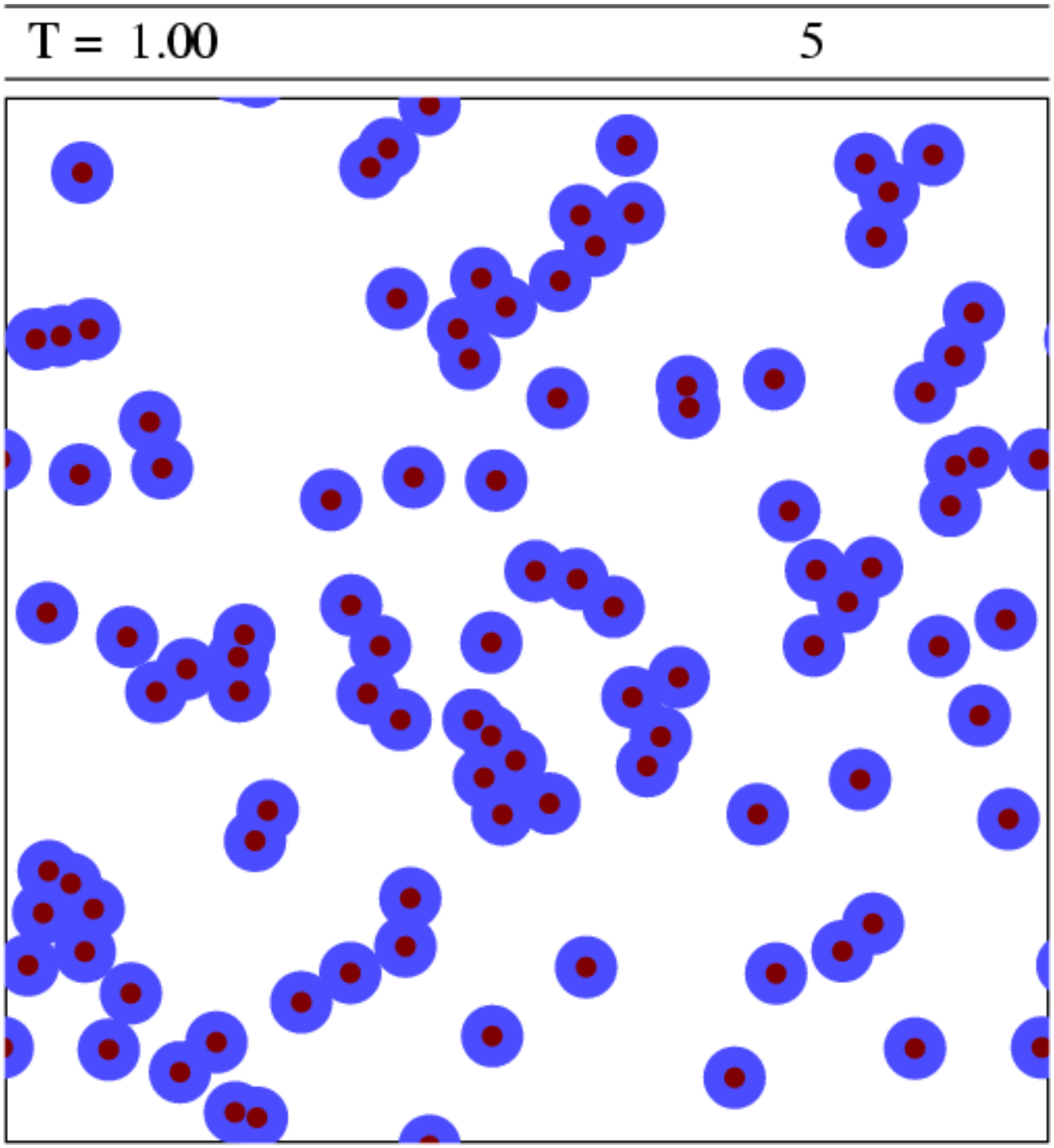
```
subroutine adjustedelta(steps,arate,delta,l)
arate=arate/real(steps)
if (arate < 0.4) delta=delta/1.5
if (arate > 0.6 .and. delta < l/4.) delta=delta*1.5
call openlog
write(10,*)'Acceptance rate, delta : ',arate,delta
call closelog
arate=0.
```

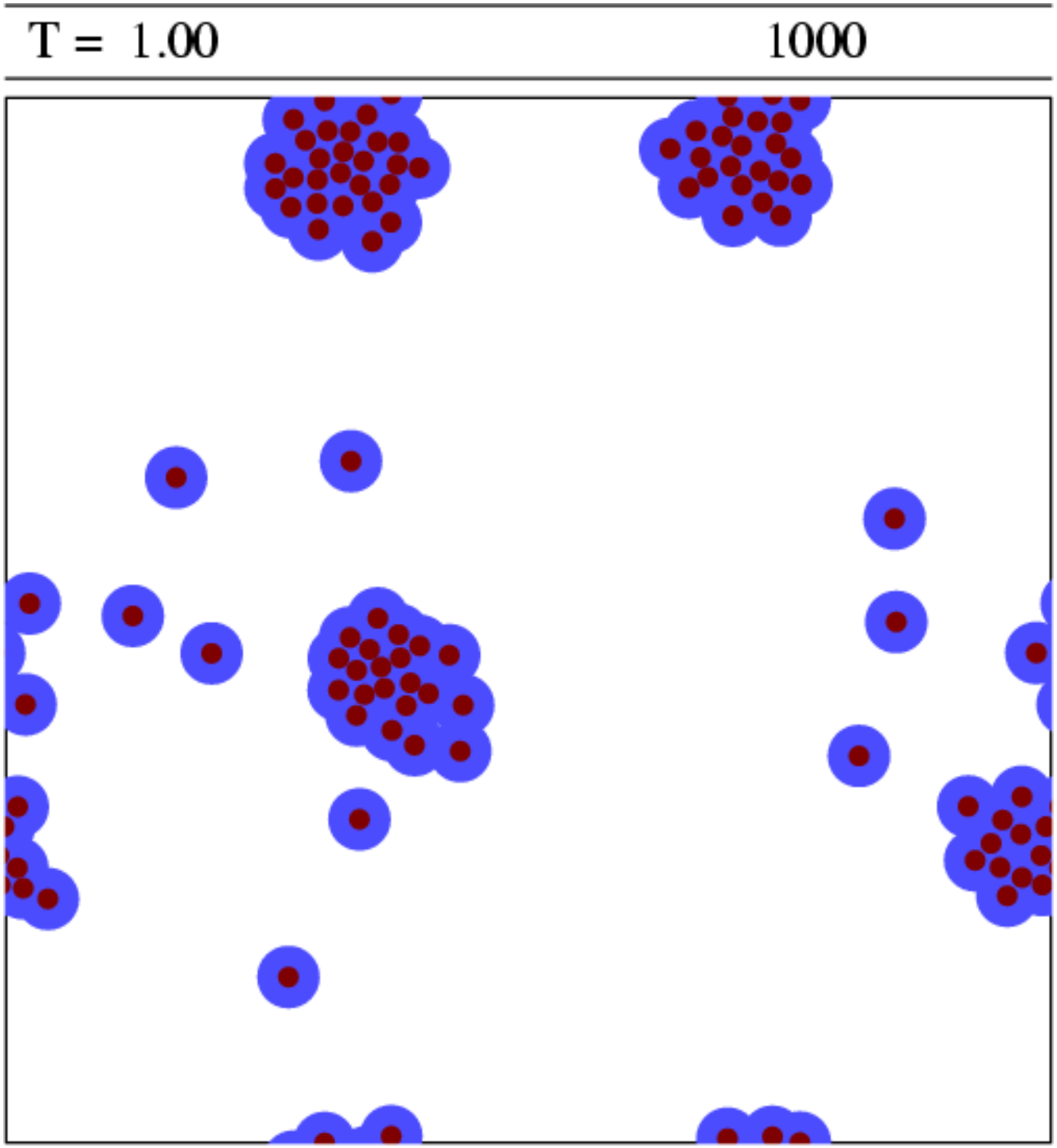

T = 1.00
1


T = 0.20
10

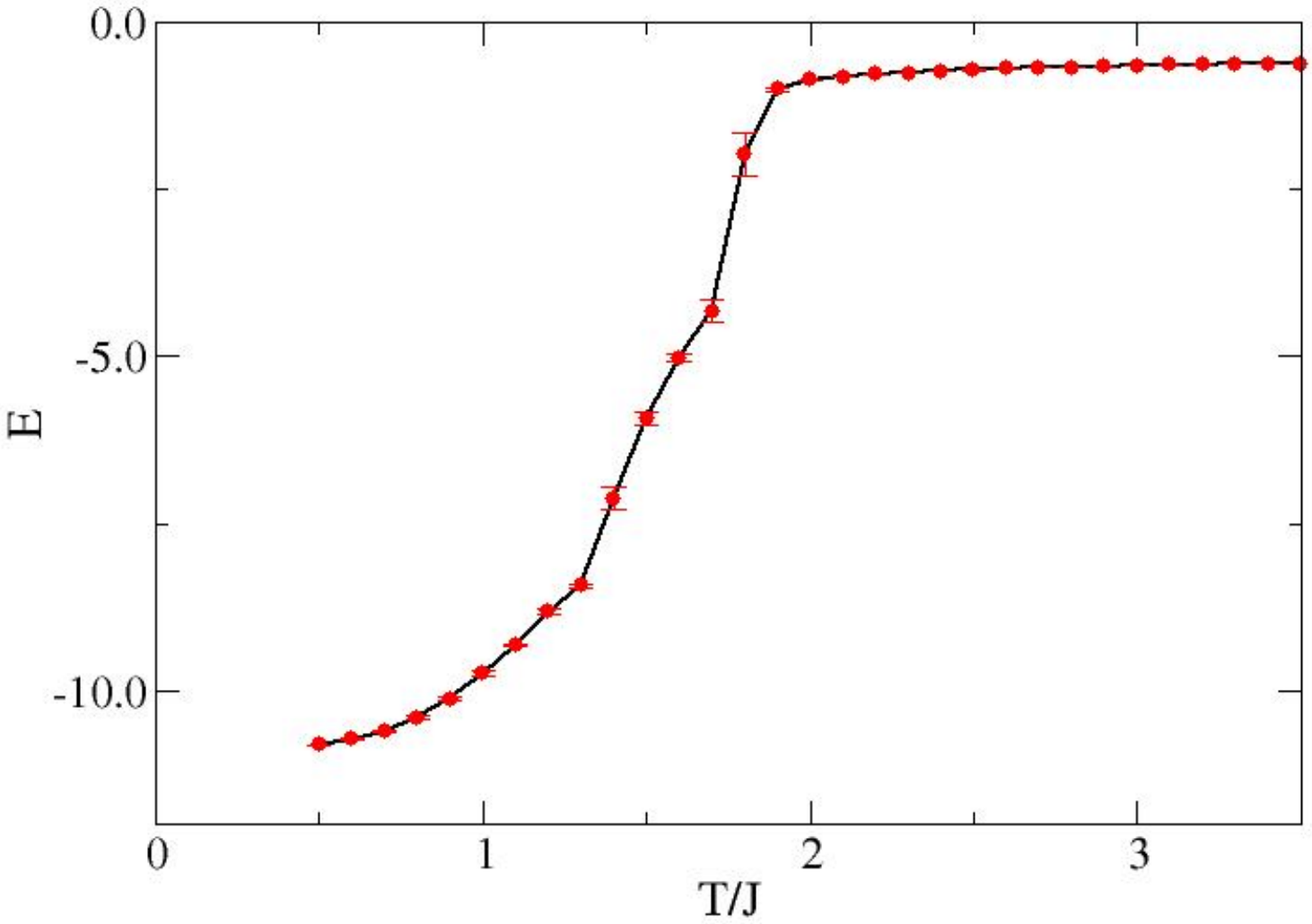

T = 0.20
1000


T = 5.00
5

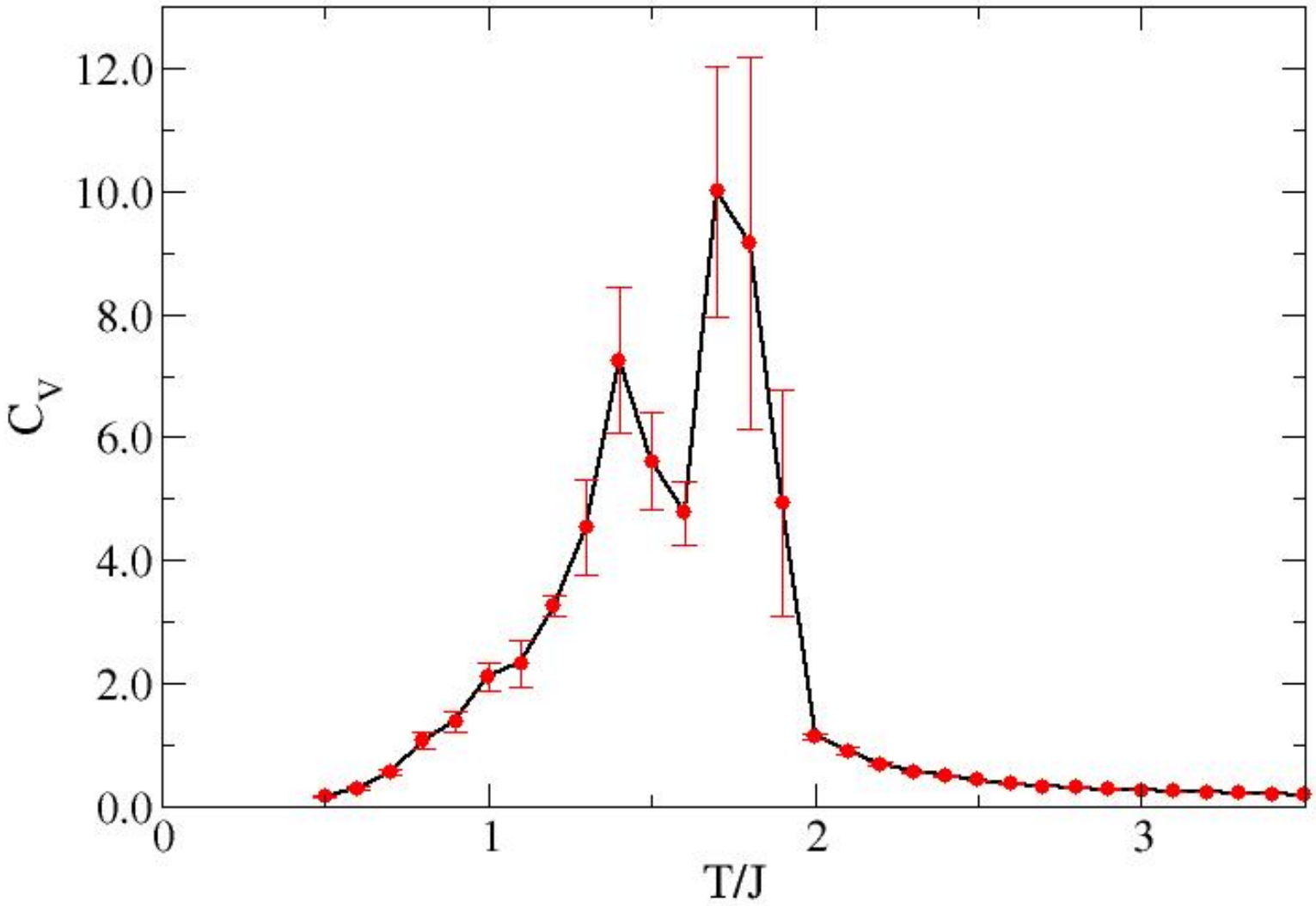

T = 1.00
5


T = 1.00
1000

Internal energy



Specific heat



Simulated annealing

General, very useful optimization method

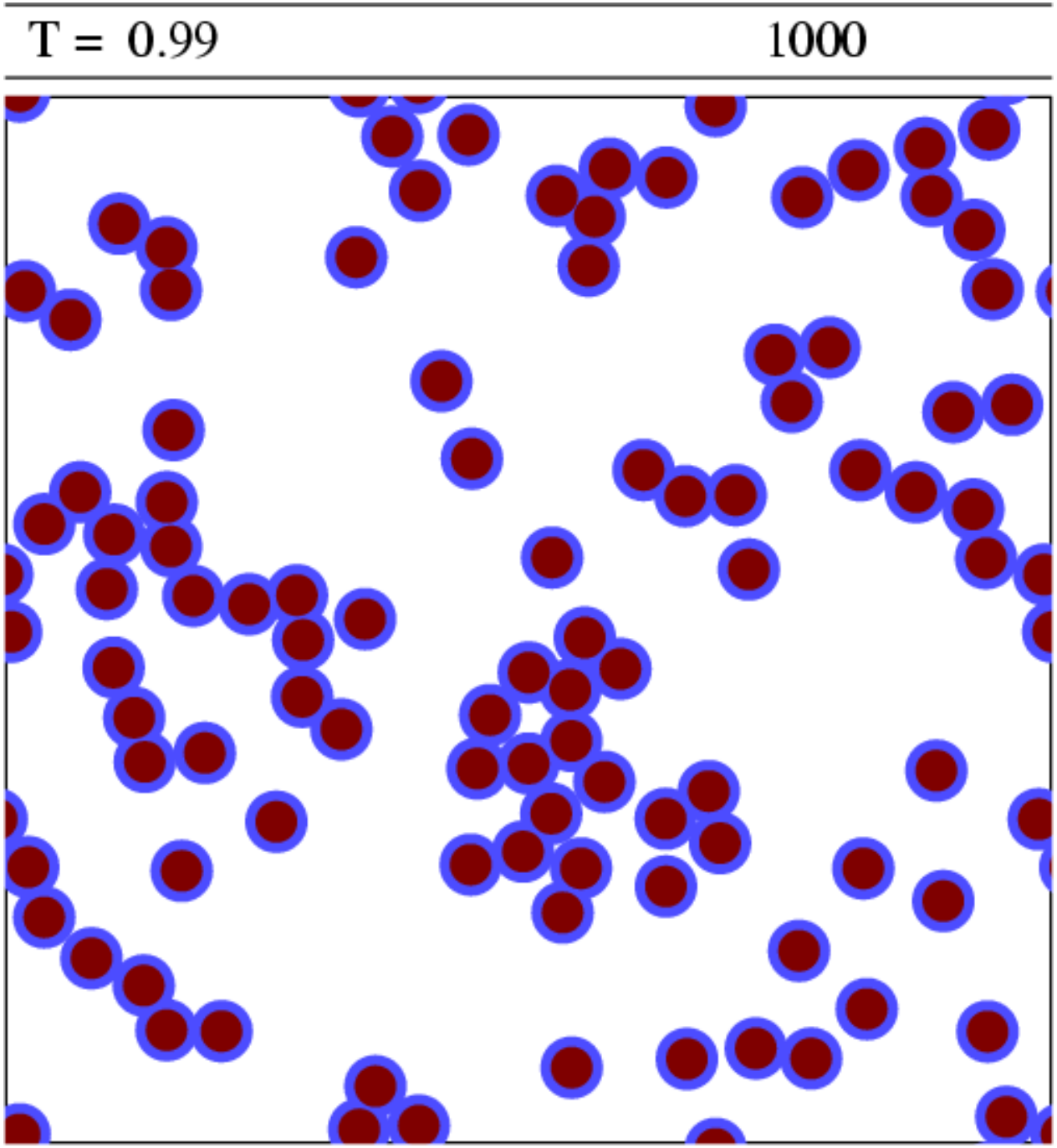
In optimization, some function of a number of variables is to be minimized or maximized.

In many systems one can think of this function (or its negative value) as an energy of a many-body system.

Introduce a fictitious temperature

Minimize E in a Monte Carlo simulation with slowly decreasing temperature.

May not give the absolutely best solution within finite time, but typically a very good silution is obtained.


T = 0.99
1000