FILE: FRAPEXP.F90

```
1  program rw
2
3     use global_env
4     use calendar
5     use walker
6     use cell
7     use frap
8
9     implicit none
10
11    integer :: max_events
12    integer :: action, opt1, opt2
13
14    real(DP) :: time, time_limit, time_stat
15    real(DP) :: sigma, rho, xx
16
17    integer, dimension(:,:), allocatable :: p0
18
19    real(DP), dimension(:), allocatable :: rn
20
21    integer :: i, j, k, l, m, n
22    integer :: n_errors
23
24    character(len=10) :: arg
25
26 !!!!!! My Variables
27
28    logical :: bleached = .false.
29    real(DP) :: ratio
30
```

```
!!!!!!!!!!!!!!!!!!!!!

    ! Extract the needed data from the configuration file

    open(10,file='config.dat')
    read(10,*) n_walkers
    read(10,*) box_size
    read(10,*) cell_size
    read(10,*) k_3d, k_2d
    read(10,*) k_on, k_off
    read(10,*) time_limit
    read(10,*) time_stat
    close(10)

    ! open log file for output

    open(fmain,file='rw.log')

    write(fmain,*) '*** KMC FRAP *** '
    write(fmain,*) 'Started at ',time_stamp()
    write(fmain,'(/a,i5)')  'Number of Random Walkers=',n_walkers
    write(fmain,'(3(a,i4))') 'Box size=',box_size(X),' x',
       box_size(Y),' x',box_size(Y)
    write(fmain,'(3(a,i4))') 'Cell size=',cell_size(X),' x',
       cell_size(Y),' x',cell_size(Y)
    write(fmain,'(2(a,f8.4),a)') 'Transition rate=',k_3d,'(3d)
       ',k_2d,'(2d)'
    write(fmain,'(2(a,f8.4),a)') 'Transition rate=',k_on,'(on)
       ',k_off,'(off)'
    write(fmain,'(a,e15.3)') 'Total simulation time=',time_limit
    write(fmain,*)
```

```
58
59      ! open a data file #1
60      write(fmain,*)  'outout: onoff.dat'
61      open(fdat1,file='onoff.dat')
62
63      ! open a data file #2
64      write(fmain,*) 'output: diffusion.dat'
65      open(fdat2,file='diffusion.dat')
66
67      open(72, file='frap.dat')
68
69      ! set the maximum number of walkers.
70      !(this determines the size of array)
71      max_walkers = 2*n_walkers
72
73      ! set the maximum number of events recorded in the calendar
           at a time.
74      max_events = 2*max_walkers
75
76      ! initialize the event calendar
77      call calendar_init(n_walkers,max_events)
78
79      ! initialize the random walkers
80      call walker_init(n_walkers)
81
82      ! initial position of walkers
83      !(for example uniformally random distribution)
84      allocate(rn(n_walkers)) ! Create a vector for the random
           numbers
85      call random_number(rn)
86      walker_pos(1:n_walkers,X) = ceiling(rn*box_size(X))
```

```
87      call random_number(rn)
88      walker_pos(1:n_walkers,Y) = ceiling(rn*box_size(Y))
89      call random_number(rn)
90      walker_pos(1:n_walkers,Z) = ceiling(rn*box_size(Z))
91      deallocate(rn) ! Not needed anymore, so there isn't any sense
            in
92      ! letting rn take up any memory
93
94      ! initialize the cell configuration
95      call cell_init(n_walkers,max_walkers,walker_pos)
96      ! This is used for determining particle collisions
97
98      ! predict the jump time for all walkers
99      do i=1,n_walkers
100          call walker_predict_event(i)
101     end do
102
103     ! save the initial position
104     allocate(p0(max_walkers,3))
105     p0 = walker_pos
106
107     ! schedule the first data output
108     call calendar_schedule_event(11,0,0,time_stat)
109     call calendar_schedule_event(12,0,0,time_stat)
110
111     n_collisions = 0
112     time_current = 0.0_DP
113
114     do while (time_current<=time_limit)
115
116        ! find the next event to happen
```

```
117       call calendar_find_event(action,opt1,opt2)
118
119       ! execute the event
120       if (action<=10)  then   ! simple jump event
121
122          ! walker 'opt1' jumps
123          call walker_action(opt1,action)
124          ! predict next jump of walker 'opt1'
125          call walker_predict_event(opt1)
126
127       else if(action<=20) then   ! non-physical events
128
129          select case(action)
130             case(11)
131                !  evaluate surface density
132                sigma = count(walker_pos(:,Z)>box_size(Z))
133                rho = count(walker_pos(:,Z)==box_size(Z))
134                write(fdat1,'(f10.2,2e13.5)') time_current, rho,
                      sigma
135                !  set the next evaluation time
136                call calendar_schedule_event(11,0,0,time_current+
                      time_stat)
137             case(12)
138                !  evaluate mean square displacement
139                xx = real(sum((walker_pos-p0)**2),kind=DP)/real(
                      n_walkers,kind=DP)
140                write(fdat2,'(f10.2,e13.5)') time_current, xx
141                !  set the next evaluation time
142                call calendar_schedule_event(12,0,0,time_current+
                      time_stat)
143             case default
```

```
144                 ! unknown event
145                 write(fmain,'(f10.2,a)') time_current, 'unknown
                       event'
146                 ! emergency stop
147                 stop
148            end select
149
150         else if(action>20) then
151            ! binary event (\such as reaction
152            write(fmain,'(f10.2,a)') time_current, 'binary event'
153            ! binary events have not implimented yet.
154         end if
155
156         !!!!!!!!!!! My FRAP Code
157
158         if (time_current .gt. time_limit / 2) then
159
160           if (bleached .eqv. .false.) then
161               call bleach()
162               bleached = .true.
163           end if
164
165           call measure(ratio)
166           write(72, *) time_current, ratio
167         end if
168
169         !!!!!!!!!!!!!!!!!!!!!!!!!!!1
170
171      end do
172
173      close(72)
```

```
174
175     write(fmain,'(a,i10)') 'Number of collisions =',n_collisions
176
177     if(debug) call cell_test(n_walkers)
178
179     ! close output files
180
181     close(fdat1)
182     close(fdat2)
183
184     !
185     open(fdat1,file='walkers.dat')
186     ! creation of particles has not been considered yet.
187     write(fdat1,'(3i5)') (walker_pos(i,:), i=1,n_walkers)
188     close(fdat1)
189
190     if(debug) close(fdbg)
191
192     write(fmain,*) '*** Done at ',time_stamp(), '***'
193 end program rw
```