```
!=====================================================================!
!                                                                     !
!   Software Name : HEC-MW Library for PC-cluster                     !
!         Version : 2.5                                               !
!                                                                     !
!     Last Update : 2006/06/01                                        !
!        Category : Linear Solver                                     !
!                                                                     !
!            Written by Kengo Nakajima (Univ. of Tokyo)               !
!                                                                     !
!     Contact address :  IIS,The University of Tokyo RSS21 project   !
!                                                                     !
!     "Structural Analysis System for General-purpose Coupling       !
!      Simulations Using High End Computing Middleware (HEC-MW)"      !
!                                                                     !
!=====================================================================!

!C
!C***
!C*** module hecmw_solver_CG_33
!C***
!C
      module hecmw_solver_CG_33
      contains
!C
!C*** CG_33
!C
      subroutine hecmw_solve_CG_33( hecMESH,  hecMAT, ITER, RESID, ERROR, &
     &                              Tset, Tsol, Tcomm )

      use hecmw_util
      use m_hecmw_solve_error
      use m_hecmw_comm_f
      use hecmw_matrix_misc
      use hecmw_solver_misc
      use hecmw_solver_misc_33
```

```
37
38          implicit none
39
40          type(hecmwST_local_mesh) :: hecMESH
41          type(hecmwST_matrix) :: hecMAT
42          integer(kind=kint ), intent(inout):: ITER, ERROR
43          real   (kind=kreal), intent(inout):: RESID, Tset, Tsol, Tcomm
44
45          integer(kind=kint ) :: my_rank
46          integer(kind=kint ) :: ITERlog, TIMElog
47          real(kind=kreal), pointer :: B(:), X(:)
48
49          real(kind=kreal), dimension(:,:), allocatable :: WW
50
51          integer(kind=kint), parameter ::  R= 1
52          integer(kind=kint), parameter ::  Z= 2
53          integer(kind=kint), parameter ::  Q= 2
54          integer(kind=kint), parameter ::  P= 3
55          integer(kind=kint), parameter :: BT= 1
56          integer(kind=kint), parameter ::TATX=2
57          integer(kind=kint), parameter :: WK= 4
58
59          integer(kind=kint ) :: MAXIT
60          integer(kind=kint ) :: totalmpc
61
62    ! local variables
63          real   (kind=kreal) :: TOL
64          integer(kind=kint )::i
65          real   (kind=kreal)::S_TIME,S1_TIME,E_TIME,E1_TIME, START_TIME, END_TIME
66          real   (kind=kreal)::BNRM2
67          real   (kind=kreal)::RHO,RHO1,BETA,C1,ALPHA,DNRM2
68
69          S_TIME= HECMW_WTIME()
70
71    !C===
72    !C +-------+
```

```
!C | INIT. |
!C +-------+
!C===
      my_rank = hecMESH%my_rank
      X => hecMAT%X
      B => hecMAT%B

      ITERlog = hecmw_mat_get_iterlog( hecMAT )
      TIMElog = hecmw_mat_get_timelog( hecMAT )
      MAXIT  = hecmw_mat_get_iter( hecMAT )
      TOL   = hecmw_mat_get_resid( hecMAT )

      totalmpc = hecMESH%mpc%n_mpc
      call hecmw_allreduce_I1 (hecMESH, totalmpc, hecmw_sum)

      ERROR = 0

      allocate (WW(3 * hecMAT%NP, 4))
      WW = 0.d0

      call hecmw_mpc_scale(hecMESH)

!C===
!C +-----------------------------------------+
!C | {r0}= [T'](\{b} - [A]{xg}) - [T'][A][T]{xini} |
!C +-----------------------------------------+
!C===

!C-- {bt}= [T'](\{b} - [A]{xg})
      if (totalmpc.eq.0) then
        do i = 1, hecMAT%N * 3
          WW(i,BT) = B(i)
        enddo
       else
        if (TIMElog.eq.1) then
        call hecmw_trans_b_33(hecMESH, hecMAT, B, WW(:,BT), WW(:,WK), Tcomm)
```

```
109           else
110           call hecmw_trans_b_33(hecMESH, hecMAT, B, WW(:,BT), WW(:,WK))
111           endif
112         endif
113
114   !C-- compute ||{bt}||
115         if (TIMElog.eq.1) then
116         call hecmw_InnerProduct_R(hecMESH, 3, WW(:,BT), WW(:,BT), BNRM2, Tcomm)
117         else
118         call hecmw_InnerProduct_R(hecMESH, 3, WW(:,BT), WW(:,BT), BNRM2)
119         endif
120         if (BNRM2.eq.0.d0) then
121           iter = 0
122           MAXIT = 0
123           RESID = 0.d0
124           X = 0.d0
125         endif
126
127   !C-- {tatx} = [T'] [A] [T]{x}
128         if (totalmpc.eq.0) then
129           if (TIMElog.eq.1) then
130           call hecmw_matvec_33(hecMESH, hecMAT, X, WW(:,TATX), Tcomm)
131           else
132           call hecmw_matvec_33(hecMESH, hecMAT, X, WW(:,TATX))
133           endif
134          else
135           if (TIMElog.eq.1) then
136           call hecmw_TtmatTvec_33(hecMESH, hecMAT, X, WW(:,TATX), WW(:,WK), Tcomm)
137           else
138           call hecmw_TtmatTvec_33(hecMESH, hecMAT, X, WW(:,TATX), WW(:,WK))
139           endif
140         endif
141
142   !C-- {r} = {bt} - {tatx}
143         do i = 1, hecMAT%N * 3
144           WW(i,R) = WW(i,BT) - WW(i,TATX)
```

```
145         enddo
146
147         E_TIME = HECMW_WTIME()
148         Tset = Tset + E_TIME - S_TIME
149
150         Tcomm = 0.d0
151         S1_TIME = HECMW_WTIME()
152   !C
153   !C*************************************************** Conjugate Gradient Iteration start
154   !C
155         do iter = 1, MAXIT
156
157   !C===
158   !C +----------------+
159   !C | {z}= [Minv]{r} |
160   !C +----------------+
161   !C===
162         if (TIMElog.eq.1) then
163         call hecmw_precond_33(hecMESH, hecMAT, WW(:,R), WW(:,Z), WW(:,WK), Tcomm)
164         else
165         call hecmw_precond_33(hecMESH, hecMAT, WW(:,R), WW(:,Z), WW(:,WK))
166         endif
167
168   !C===
169   !C +---------------+
170   !C | {RHO}= {r}{z} |
171   !C +---------------+
172   !C===
173         if (TIMElog.eq.1) then
174         call hecmw_InnerProduct_R(hecMESH, 3, WW(:,R), WW(:,Z), RHO, Tcomm)
175         else
176         call hecmw_InnerProduct_R(hecMESH, 3, WW(:,R), WW(:,Z), RHO)
177         endif
178
179   !C===
180   !C +----------------------------+
```

```
!C | {p} = {z} if      ITER=1    |
!C | BETA= RHO / RHO1  otherwise |
!C +----------------------------+
!C===
      if ( ITER.eq.1 ) then
        do i = 1, hecMAT%N * 3
          WW(i,P) = WW(i,Z)
        enddo
       else
         BETA = RHO / RHO1
         do i = 1, hecMAT%N * 3
           WW(i,P) = WW(i,Z) + BETA*WW(i,P)
         enddo
      endif

!C===
!C +---------------------+
!C | {q}= [T'][A][T] {p} |
!C +---------------------+
!C===
      if (totalmpc.eq.0) then
        if (TIMElog.eq.1) then
        call hecmw_matvec_33(hecMESH, hecMAT, WW(:,P), WW(:,Q), Tcomm)
        else
        call hecmw_matvec_33(hecMESH, hecMAT, WW(:,P), WW(:,Q))
        endif
       else
        if (TIMElog.eq.1) then
        call hecmw_TtmatTvec_33(hecMESH, hecMAT, WW(:,P), WW(:,Q), WW(:,WK), Tcomm)
        else
        call hecmw_TtmatTvec_33(hecMESH, hecMAT, WW(:,P), WW(:,Q), WW(:,WK))
        endif
      endif

!C===
!C +---------------------+
```

```
217   !C | ALPHA= RHO / {p}{q} |
218   !C +---------------------+
219   !C===
220         if (TIMElog.eq.1) then
221         call hecmw_InnerProduct_R(hecMESH, 3, WW(:,P), WW(:,Q), C1, Tcomm)
222         else
223         call hecmw_InnerProduct_R(hecMESH, 3, WW(:,P), WW(:,Q), C1)
224         endif
225
226         ALPHA= RHO / C1
227
228   !C===
229   !C +----------------------+
230   !C | {x}= {x} + ALPHA*{p} |
231   !C | {r}= {r} - ALPHA*{q} |
232   !C +----------------------+
233   !C===
234         do i = 1, hecMAT%N * 3
235            X(i)  = X(i)    + ALPHA * WW(i,P)
236           WW(i,R)= WW(i,R) - ALPHA * WW(i,Q)
237         enddo
238
239         if (TIMElog.eq.1) then
240         call hecmw_InnerProduct_R(hecMESH, 3, WW(:,R), WW(:,R), DNRM2, Tcomm)
241         else
242         call hecmw_InnerProduct_R(hecMESH, 3, WW(:,R), WW(:,R), DNRM2)
243         endif
244
245         RESID= dsqrt(DNRM2/BNRM2)
246
247   !C##### ITERATION HISTORY
248         if (my_rank.eq.0.and.ITERLog.eq.1) write (*,'(i7, 1pe16.6)') ITER, RESID
249   !C#####
250
251         if ( RESID.le.TOL   ) exit
252         if ( ITER .eq.MAXIT ) ERROR = -300
```

```
253
254         RHO1 = RHO
255
256         enddo
257   !C
258   !C*************************************************** Conjugate Gradient Iteration end
259   !C
260         if (totalmpc.ne.0) then
261           if (TIMElog.eq.1) then
262           call hecmw_tback_x_33(hecMESH, X, WW(:,WK), Tcomm)
263           else
264           call hecmw_tback_x_33(hecMESH, X, WW(:,WK))
265           endif
266         endif
267   !C
268   !C-- INTERFACE data EXCHANGE
269   !C
270         START_TIME= HECMW_WTIME()
271         call hecmw_update_3_R (hecMESH, X, hecMAT%NP)
272         END_TIME = HECMW_WTIME()
273         Tcomm = Tcomm + END_TIME - START_TIME
274
275         E1_TIME = HECMW_WTIME()
276         Tsol = E1_TIME - S1_TIME
277
278         deallocate (WW)
279
280         end subroutine hecmw_solve_CG_33
281         end module     hecmw_solver_CG_33
282
```