СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ

Дубна

Р2-99-125

В.С.Барашенков[1], А.Г.Соловьев[2], А.Н.Соснин[3]

МОДЕЛИРОВАНИЕ ТОЧКИ ВЗАИМОДЕЙСТВИЯ И ПОТЕРЬ ЭНЕРГИИ КАСКАДНОЙ ЧАСТИЦЫ В ВЕЩЕСТВЕ МИШЕНИ

[1]E-mail: barash@cv.jinr.ru
[2]E-mail: solovjev@decimal.jinr.dubna.su
[3]E-mail: sosnin@decimal.jinr.dubna.su

1999

1. Введение

В настоящей работе[4] дано описание новой версии подпрограммы **ENLOSS**, являющейся составной частью программы моделирования межъядерных каскадов **CASCAD**. Подпрограмма **ENLOSS** моделирует пробег частицы (протона, нейтрона, $\pi^{\pm}$- и π^0-мезонов, тяжелого иона). При этом определяется точка взаимодействия налетающей частицы с ядрами мишени, точка ее остановки вследствие потерь энергии на ионизацию, точка распада (для $\pi^{\pm}$- и π^0-мезонов), а также координаты точки вылета частицы из мишени. В подпрограмме учитывается гетерогенная структура мишени (если таковая имеется).

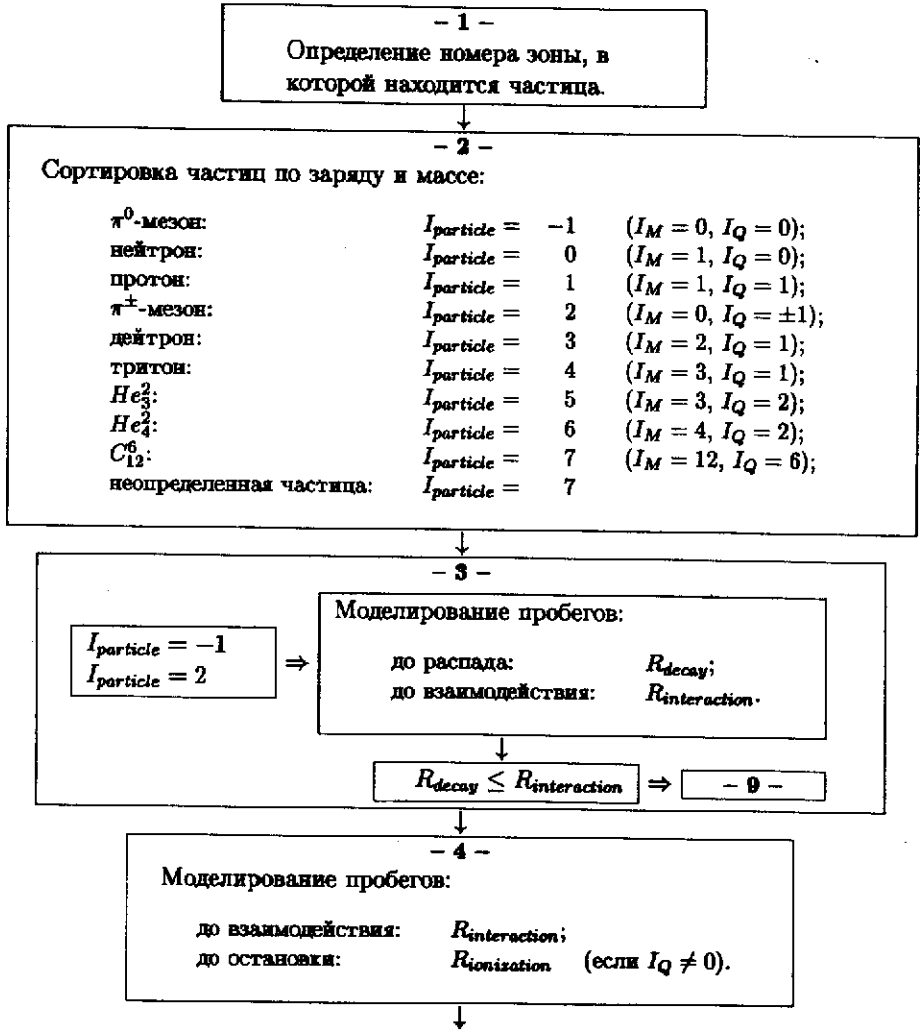
Отличия новой версии подпрограммы от существующих состоят в том, что указанные выше задачи решаются в одной подпрограмме (раньше этой цели служили несколько подпрограмм — отдельно для каждого типа частиц). Подпрограмма определяет тип частицы, а затем реализуется соответствующий алгоритм, учитывающий этот тип. Создание нового геометрического модуля (см. [1]) позволяет упростить подпрограмму **ENLOSS** в том смысле, что теперь весь анализ гетерогенной структуры мишени вынесен в этот модуль. Благодаря этим особенностям подпрограмма **ENLOSS** упростилась, стала легко читаемой и (в случае необходимости) легко модернизируемой.

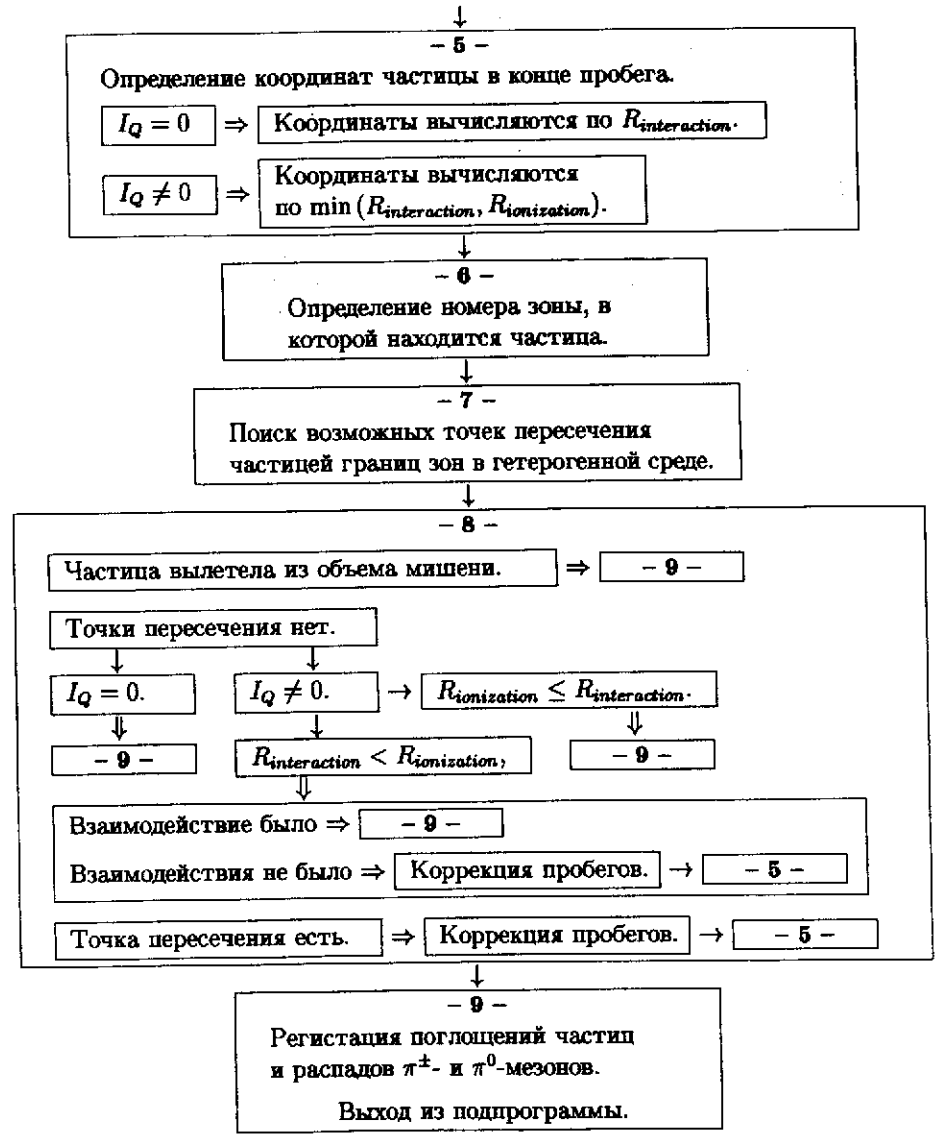
Ниже приводится блок-схема алгоритма подпрограммы **ENLOSS**, в Приложении I — ее полный текст (на языке FORTRAN'77). Кроме того, в Приложении II приводятся некоторые подпрограммы геометрического модуля, которые были усовершенствованы.

[4]Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (коды проектов 97 – 01 – 00746, 98 – 01 – 00190).

2. Алгоритм

Алгоритм подпрограммы ENLOSS организован следующим образом.





Следует отметить, что при моделировании и коррекции пробегов частицы подпрограмма ENLOSS учитывает наличие или отсутствие вещества в той зоне мишени, где в данный момент находится частица. Учитываются также свойства

этого вещества, тип частицы и проверяется, превосходит ли энергия частицы заданную для каждого типа частиц пороговую величину, которая определяет возможность взаимодействия частицы с ядрами мишени (на приведенной схеме эта проверка не обозначена, подробнее см. Приложение I).

Заключение

Используемые версии подпрограммы ENLOSS (а также подпрограмм геометрического модуля) могут отличаться от данного здесь их описания, что может быть связано с постоянным развитием алгоритмов, расширением класса решаемых задач и т. п. Со всеми вопросами и замечаниями, касающимися этих модулей, следует обращаться непосредственно к авторам.

Приложение I.
Подпрограмма ENLOSS

```
      SUBROUTINE ENLOSS
C-----------------------------------------------------------------------
C     ENLOSS IS THE MONTE CARLO SUBROUTINE FOR THE
C     CALCULATION OF THE PARTICLE ENERGY LOSSES (DE/DX).
C-----------------------------------------------------------------------
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /BLOK7/TR(85) /BLOK8/RN(1700) /BLOK9/RP(1700)
     *       /BLOK10/PI2G /BLOK11/POGL/PMABSR/PIMR /BLGR8/PRLEN
     *       /BLOK13/TOBRN,TOBRP /BLOK23/TOBR(5)
     *       /BLOK14/XG,YG,ZG /ITT/IT /TKCH/TKCH
     *       /WRPR/WP(9) /WRPRIN/WPIN(9)
     *       /BLOK1/GH(9,30) /GHAUX/GHAUX(30)
     *       /GOOUT/GOT /GOTO1/GOT1,GOTO /PRBEG/RO
     *       /DEMAX/BX1(3),EDX1(3),DEDX1(3) /IEMAX/IEMAX
     *       /BLOK7I/TRI(90) /BLOK8I/RNI(90)
C-----------------------------------------------------------------------
C     Paremeters used to determine the average nuclear range.
      DIMENSION WAYM(7)
      DATA WAYM/2.284,3.269,2.194,1.914,1.914,1.914,1.914/
C-----------------------------------------------------------------------
C     Function PASSNC defines the range of a particle according to its
C     total macroscopic cross section.
C     Function EINT makes either exponential or logarithmic
C     interpolation.
C-----------------------------------------------------------------------
C     Auxiliary parameter used in function EINT:
      IT=0
C     Parameter indicating that the particle is absorbed (POGL > 0):
      POGL=-1.0D0
C     Parameter indicating that Pi-meson (0) is decayed into two
C     gammas (PI2G > 0):
      PI2G=-1.0D0
C     Parameter indicating that Pi-meson (-) is stopped due to its
C     energy losses (PIMR > 0):
      PIMR=-1.0D0
C     Parameter indicating that ENLOSS operates with neutrons at
C     energies exceeding 10.5 MeV (PRLEN > 0):
      PRLEN=-1.0D0
C     Starting values for nuclear and ionization ranges:
      PROBEG=0.0D0
      RO=0.0D0
C-----------------------------------------------------------------------
```

```
C-----------------------------------------------------------------------
C     Determination of the particle place inside the target volume:
      CALL OUT(WP(1),WP(2),WP(3))
      NZN=IDINT(-GOT+0.3D0)
C-----------------------------------------------------------------------
C     Some new variables:
      IM=IDINT(WP(9)/1000.0D0+0.3D0)
      IF(WP(8).GE.0.0D0)IQ=IDINT(WP(8)+0.3D0)
      IF(WP(8).LT.0.0D0)IQ=IDINT(WP(8)-0.3D0)
C     Sorting of the particle type.
C     Particle is not identified:
      IPART=7
C     Pi-meson (0):
      IF(IM.EQ.0.AND.IQ.EQ.0)IPART=-1
C     Neutron:
      IF(IM.EQ.1.AND.IQ.EQ.0)IPART=0
C     Proton:
      IF(IM.EQ.1.AND.IQ.EQ.1)IPART=1
C     Pi-meson (-,+):
      IF(IM.EQ.0.AND.IQ.NE.0)IPART=2
C     Deutron:
      IF(IM.EQ.2.AND.IQ.EQ.1)IPART=3
C     Triton:
      IF(IM.EQ.3.AND.IQ.EQ.1)IPART=4
C     He-3:
      IF(IM.EQ.3.AND.IQ.EQ.2)IPART=5
C     He-4:
      IF(IM.EQ.4.AND.IQ.EQ.2)IPART=6
C     Carbon-6/12:
      IF(IM.EQ.12.AND.IQ.EQ.6)IPART=7
C     Unidentified particle:  IPART=7.
C-----------------------------------------------------------------------
      IF(IPART.NE.2.AND.IPART.NE.-1)GO TO 1000
C-----------------------------------------------------------------------
C     The range up to the decay point:
      B=RNDM(-1)
      BL=-DLOG(B)
      IF(IQ.EQ.0)TPIZ=0.0000054D0
      IF(IQ.NE.0)TPIZ=760.45D0
      PRRASP=BL*TPIZ*(SQRT(WP(7)*(WP(7)+280.0D0))/WP(9))
C-----------------------------------------------------------------------
      IF(GHAUX(NZN).EQ.0.0D0)PROBEG=1000.0D0
      IF(GHAUX(NZN).NE.0.0D0)PROBEG=-DLOG(RNDM(-1))*PASSNC(WP(9),WP(7))
      IF(PRRASP.LE.PROBEG)GO TO 1015
C-----------------------------------------------------------------------
```



```
C-----------------------------------------------------------------------
C     Calculation of the particle ranges.
 1000 CONTINUE
      IF(GHAUX(NZN).NE.0.0D0)GO TO 1002
C     The particle is in empty space:
 1001 PROBEG=1000.0D0
      IF(IQ.NE.0)RO=1000.0D0
      GO TO 1004
C     The particle is inside the matter:
 1002 B=RNDM(-1)
      BL=-DLOG(B)
      IF(IPART.LE.0)FL=PASSNC(WP(9),WP(7))
      IF(IPART.GT.0)FL=WAYM(IPART)
      PROBEG=FL*BL
 1003 CONTINUE
      IF(IQ.EQ.0)GO TO 1004
      NK=85*(NZN-1)+1
      IF(IPART.EQ.1)RO=EINT(WP(7),TR,RN(NK),85,0)
      IF(IPART.EQ.2)RO=EINT(WP(7),TR,RP(NK),85,0)
      IF(IPART.LT.3)GO TO 1004
      TE=940.0D0*WP(7)/WP(9)
      AM=WP(9)/(940.0D0*WP(8)*WP(8))
      IF(WP(7).LE.EDX1(3))RO=EINT(WP(7),TRI,RNI,IEMAX,0)
      IF(WP(7).GT.EDX1(3))RO=AM*EINT(TE,TR,RN(NK),85,0)
C-----------------------------------------------------------------------
 1004 DO 1005 I=1,9,1
 1005 WPIN(I)=WP(I)
C     Determination of the new particle co-ordinates:
      IF(IQ.EQ.0)CALL NEWCOR(PROBEG,+1)
      IF(IQ.NE.0)CALL NEWCOR(DMIN1(PROBEG,RO),+1)
C-----------------------------------------------------------------------
      IF(IQ.EQ.0.OR.PROBEG.GE.RO)GO TO 1007
C     Calculation of  FL when PROBEG < RO:
      ROL=RO-PROBEG
      IF(IPART.EQ.1)WP(7)=EINT(ROL,RN(NK),TR,85,4)
      IF(IPART.EQ.2)WP(7)=EINT(ROL,RP(NK),TR,85,4)
      IF(IPART.LT.3)GO TO 1006
      IF(WP(7).LE.EDX1(3))WP(7)=EINT(ROL,RNI,TRI,IEMAX,4)
      IF(WP(7).LE.EDX1(3))GO TO 1006
      AMR=ROL/AM
      WP(7)=(WP(9)/940.0D0)*EINT(AMR,RN(NK),TR,85,4)
 1006 FL=PASSNC(WP(9),WP(7))
C-----------------------------------------------------------------------
```

```
C-----------------------------------------------------------------------
 1007 GOT1=GOT
      CALL OUT(WP(1),WP(2),WP(3))
C     Looking for a cross-point:
      CALL COROUT(DEL,GOT1,WP(7))
C-----------------------------------------------------------------------
      IF(GOT.GT.0.0D0)GO TO 1014
      IF(IQ.EQ.0.OR.PROBEG.LT.RO)GO TO 1008
C-----------------------------------------------------------------------
C     Ionization range < or = interaction range:
      IF(GOT.EQ.GOT1)GO TO 1012
      GO TO 1010
C-----------------------------------------------------------------------
C     Interaction range < ionization range:
 1008 CONTINUE
      IF(IPART.EQ.1.AND.WP(7).LE.TOBRN)GO TO 1013
      IF(IPART.EQ.2.AND.WP(7).LE.TOBRP)GO TO 1013
      IF(IPART.LT.3)GO TO 1009
      IF(WP(7).LE.TOBR(IPART-2))GO TO 1013
 1009 CONTINUE
      IF(GOT.EQ.GOT1)GO TO 1011
C-----------------------------------------------------------------------
C     The cross-point is found:
 1010 NZ=IDINT(-GOT1+0.3)
      NZN=IDINT(-GOT+0.3)
      IF(GHAUX(NZN).EQ.0.0D0)GO TO 1001
      IF(GHAUX(NZ).EQ.0.0D0)GO TO 1002
      IF(IPART.NE.0)PROBEG=DEL
      IF(IPART.NE.0)GO TO 1003
      BL=BL-(PROBEG-DEL)/FL
      FL=PASSMC(WP(9),WP(7))
      PROBEG=FL*BL
      GO TO 1004
C-----------------------------------------------------------------------
C     The cross-point is not found:
 1011 CONTINUE
      IF(IPART.LE.0)RETURN
      IF(FL*RNDM(-1).LT.WAYM(IPART))RETURN
      B=RNDM(-1)
      BL=-DLOG(B)
      FL=WAYM(IPART)
      PROBEG=FL*BL
      RO=ROL
      GO TO 1004
C-----------------------------------------------------------------------
```



```
C-----------------------------------------------------------------------
C     Determination of some parameters:
 1012 NZ=IDINT(-GOT1+0.3)
      IF(IPART.GT.0.AND.GHAUX(NZ).NE.0.0D0)WP(7)=0.0D0
 1013 POGL=1.0D0
      IF(IPART.EQ.2.AND.IQ.LT.0)PIMR=1.0D0
      RETURN
 1014 CONTINUE
      IF(IPART.NE.-1)RETURN
 1015 PI2G=1.0D0
      CALL NEWCOR(PRRASP,+1)
      CALL OUT(WP(1),WP(2),WP(3))
      RETURN
C-----------------------------------------------------------------------
      END
```

Приложение II.
Новая версия геометрического модуля

После публикации [1] геометрические модули программы моделирования межъядерных каскадов подверглись некоторому усовершенствованию. В связи с этим приводим тексты подпрограмм, которых коснулись изменения. [5]

Отличия от предыдущей версии состоят в следующем:

1) для каждой зоны задаются номера зон, помещающихся внутри нее;
2) для каждой поверхности автоматически определяется номер зоны, границей которой она является.

Это позволяет сразу определить, будет ли иметь место пересечение с данной поверхностью, или нет. Поэтому те поверхности, с которыми нет пересечений, заранее исключаются из рассмотрения. Это существенно сокращает время расчетов.

Основные изменения коснулись подпрограммы COROUT и информации о зонах мишени, задаваемой в файле INPUT.DAT. Соответствующий фрагмент INPUT.DAT имеет вид:

```
.........................................
C                        TARGET PARAMETERS
C                 Number of zones in the target ( < or = 30 )
                                  4
C                   Geometrical description of the zones.
C  Two strings are used for each zone:
C  1) geometrical parameters for this zone
C     (see communication of JINR, R2-98-221 for detales):
C   X1      X2      Y1      Y2      Z1      Z2      R1      R2      Ntype
    0.      0.      0.      0.      0.0     23.8    15.7    0.      2.3
2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
    0.      0.      0.      0.      2.4     21.4    13.3    0.      2.3
3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
    0.      0.      0.      0.      4.9     18.9    10.8    0.      2.3
4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
    0.      0.      0.      0.      4.9     18.9    1.3     0.      2.3
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
.........................................
```

(В качестве примера рассмотрен тот же случай, что и в [1] — четыре вложенных последовательно один в другой цилиндра).

[5] В подпрограммах OUT и CHECK были обнаружены и устранены ошибки, касающиеся конусов. Здесь приводятся исправленные версии этих подпрограмм.

Соответствующий фрагмент BEMPAK, в котором считываются эти данные, имеет теперь вид:

```
.........................................
      READ (5,*)
      READ (5,*)
C     Number of zones in the target ( < or = 30 )
      READ (5,*)NZONE
      READ (5,*)
      READ (5,*)
      READ (5,*)
      READ (5,*)
      READ (5,*)
      READ (5,*)
C     Geometrical description of the zones.
      DO 3 I=1,NZONE,1
      READ (5,*)(GH(J,I),J=1,9,1)
      READ (5,*)(INGH(I,INZ),INZ=1,29,1)
    3 CONTINUE
      CALL GEOMETRY
.........................................
```

В целочисленном массиве COMMON /INSIDE/INGH(30,29)) заданы, таким образом, для каждой зоны номера всех находящихся внутри нее зон.

Напомним, что подпрограмма GEOMETRY (она не изменилась) для каждой зоны в зависимости от типа ее геометрии вызывает соответствующую подпрограмму TYPE..., которая вычисляет коэффициенты поверхностей, ограничивающих эту зону, параметры, определяющие условия ограниченности этих поверхностей, и число поверхностей. Теперь, кроме того, в каждой подпрограмме TYPE... определяется, к какой зоне относится данная поверхность. Соответствующий фрагмент одной из подпрограмм TYPE... имеет вид:

```
.........................................
C     Zone:
      INSF(K)=I
.........................................
```

Здесь K — номер поверхности, а I — номер зоны. Таким образом, в целочисленном массиве COMMON /INSIDES/INSF(60) задается соответствие между зонами и поверхностями. В остальном подпрограммы TYPE... не изменились.

Далее приводятся тексты подпрограмм OUT, COROUT, CROSS и CHECK.

```
      SUBROUTINE GOT(X,Y,Z)
C-----------------------------------------------------------------------
C     DETERMINATION OF A ZONE NUMBER.
C     IF GOT > 0, THEN THE POINT IS OUTSIDE THE TARGET.
C     IF GOT < 0, THEN GOT = - ZONE NUMBER.
C-----------------------------------------------------------------------
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /BLOK1/GH(9,30)
     *       /NZONE/NZONE
     *       /GOOUT/GOT
C-----------------------------------------------------------------------
      DO 100 I=NZONE,1,-1
C     Selection of a geometry type :
      IF(GH(9,I).EQ.1.0D0)GO TO 1
      IF(GH(9,I).EQ.2.1D0)GO TO 21
      IF(GH(9,I).EQ.2.2D0)GO TO 22
      IF(GH(9,I).EQ.2.3D0)GO TO 23
      IF(GH(9,I).EQ.3.1D0)GO TO 31
      IF(GH(9,I).EQ.3.2D0)GO TO 32
      IF(GH(9,I).EQ.3.3D0)GO TO 33
      IF(GH(9,I).EQ.4.0D0)GO TO 4
C-----------------------------------------------------------------------
 1    CONTINUE
C     Parallelepiped :
      IF(X.GE.GH(1,I).AND.X.LE.GH(2,I).AND.
     +   Y.GE.GH(3,I).AND.Y.LE.GH(4,I).AND.
     +   Z.GE.GH(5,I).AND.Z.LE.GH(6,I))GO TO 200
      GO TO 100
C-----------------------------------------------------------------------
 21   CONTINUE
C     Cylinder along x-axes :
      IF(X.GE.GH(1,I).AND.X.LE.GH(2,I).AND.
     +   DSQRT((Y-GH(3,I))**2+(Z-GH(5,I))**2).LE.GH(7,I))GO TO 200
      GO TO 100
C-----------------------------------------------------------------------
 22   CONTINUE
C     Cylinder along y-axes :
      IF(Y.GE.GH(3,I).AND.Y.LE.GH(4,I).AND.
     +   DSQRT((Z-GH(5,I))**2+(X-GH(1,I))**2).LE.GH(7,I))GO TO 200
      GO TO 100
C-----------------------------------------------------------------------
```



```
C-----------------------------------------------------------------------
 23   CONTINUE
C     Cylinder along z-axes :
      IF(Z.GE.GH(5,I).AND.Z.LE.GH(6,I).AND.
     +   DSQRT((X-GH(1,I))**2+(Y-GH(3,I))**2).LE.GH(7,I))GO TO 200
      GO TO 100
C-----------------------------------------------------------------------
 31   CONTINUE
C     Cone along x-axes :
      IF(X.GE.GH(1,I).AND.X.LE.GH(2,I).AND.
     +   DSQRT((Y-GH(3,I))**2+(Z-GH(5,I))**2).LE.
     +   GH(7,I)+(X-GH(1,I))*(GH(8,I)-GH(7,I))/(GH(2,I)-GH(1,I)))
     +GO TO 200
      GO TO 100
C-----------------------------------------------------------------------
 32   CONTINUE
C     Cone along y-axes :
      IF(Y.GE.GH(3,I).AND.Y.LE.GH(4,I).AND.
     +   DSQRT((Z-GH(5,I))**2+(X-GH(1,I))**2).LE.
     +   GH(7,I)+(Y-GH(3,I))*(GH(8,I)-GH(7,I))/(GH(4,I)-GH(3,I)))
     +GO TO 200
      GO TO 100
C-----------------------------------------------------------------------
 33   CONTINUE
C     Cone along z-axes :
      IF(Z.GE.GH(5,I).AND.Z.LE.GH(6,I).AND.
     +   DSQRT((X-GH(1,I))**2+(Y-GH(3,I))**2).LE.
     +   GH(7,I)+(Z-GH(5,I))*(GH(8,I)-GH(7,I))/(GH(6,I)-GH(5,I)))
     +GO TO 200
      GO TO 100
C-----------------------------------------------------------------------
 4    CONTINUE
C     Sphere :
      IF(DSQRT((X-GH(1,I))**2+(Y-GH(3,I))**2+(Z-GH(5,I))**2).LE.
     +   GH(7,I))GO TO 200
      GO TO 100
C-----------------------------------------------------------------------
 100  CONTINUE
      GOT=1.0D0
      RETURN
 200  GOT=-FLOAT(I)
      RETURN
      END
```

```
      SUBROUTINE COROUT(BACK,SIDE,FORV)
C-----------------------------------------------------------------------
C     DETERMINATION OF THE CLOSEST INTERSECTION POINT.
C-----------------------------------------------------------------------
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /WRPRIN/WPIN(9)
     *       /WRPR/WP(9)
     *       /DXYZ/DX,DY,DZ
     *       /GOOUT/GOT
     *       /BLOK1/GH(9,30)
     *       /INSIDE/INGH(30,29)
     *       /INSIDES/INSF(60)
     *       /GHAUX/GHAUX(30)
     *       /NSURFACES/NSF
      DIMENSION RC1(3),RC2(3)
C-----------------------------------------------------------------------
      BACK=0.0D0
      IF(SIDE.GT.0.0D0)RETURN
      NZ=IDINT(-SIDE+0.3)
      NZ1=IDINT(-GOT+0.3)
      IF(NZ.EQ.NZ1.AND.INGH(NZ,1).EQ.0)RETURN
      DEL1=0.0D0
      DEL2=0.0D0
C     Let's determin the following values to use them in CROSS:
      DX=WP(1)-WPIN(1)
      DY=WP(2)-WPIN(2)
      DZ=WP(3)-WPIN(3)
C     Looking for the closest intersection point amongst all surfaces:
C-----------------------------------------------------------------------
      DO 4 I=1,NSF,1
C     Some checkings for the surface of number I:
      IF(NZ.NE.NZ1.AND.INSF(I).EQ.NZ)GOTO 2
      DO 1 INZ=1,29,1
      IF(INGH(NZ,INZ).NE.0.AND.INSF(I).EQ.INGH(NZ,INZ))GOTO 2
 1    CONTINUE
      GOTO 4
```



```
C     Determination of intersection points for the surfase of number I:
 2    CALL CROSS(I,RC1,RC2,IND1,IND2)
C     Checking for these intersection points:
      IF(IND1.LE.0)GOTO 3
      CALL CHECK(I,RC1(1),RC1(2),RC1(3),IND)
      IF(IND.GT.0)DEL1=DMAX1(DEL1,
     +DSQRT((WP(1)-RC1(1))**2+(WP(2)-RC1(2))**2+(WP(3)-RC1(3))**2))
 3    IF(IND2.LE.0)GOTO 4
      CALL CHECK(I,RC2(1),RC2(2),RC2(3),IND)
      IF(IND.GT.0)DEL2=DMAX1(DEL2,
     +DSQRT((WP(1)-RC2(1))**2+(WP(2)-RC2(2))**2+(WP(3)-RC2(3))**2))
 4    CONTINUE
C-----------------------------------------------------------------------
C     Choose ot the closest intersection point:
      BACK=DMAX1(DEL1,DEL2)
      IF(BACK.LE.0.0D0)RETURN
C     To avoid stoping:
      BACK=BACK-1.0D-7
      PATH=DSQRT(DX**2+DY**2+DZ**2)
      WP(1)=WP(1)-DX*BACK/PATH
      WP(2)=WP(2)-DY*BACK/PATH
      WP(3)=WP(3)-DZ*BACK/PATH
C     Determination of a zone number:
      CALL OUT(WP(1),WP(2),WP(3))
C     Correction of the energy for the charged particles:
      IF(WP(8).EQ.0.0D0)RETURN
      PATH=
     +SQRT((WP(1)-WPIN(1))**2+(WP(2)-WPIN(2))**2+(WP(3)-WPIN(3))**2)
      IF(GHAUX(NZ).GT.0.0D0)WP(7)=TNEW(PATH,SIDE)
      FORV=WP(7)
C-----------------------------------------------------------------------
      RETURN
      END
```

```
      SUBROUTINE CROSS(I,RC1,RC2,IND1,IND2)
C-----------------------------------------------------------------------
C     CALCULATION OF INTERSECTION POINTS :
C  Surface:        SF(1,I) * X^2 +   SF(2,I) * Y^2 +   SF(3,I) * Y^2 +
C             + 2 SF(4,I) * XY  + 2 SF(5,I) * YZ  + 2 SF(6,I) * ZX  +
C             + 2 SF(7,I) * X   + 2 SF(8,I) * Y   + 2 SF(9,I) * Z   +
C             +   SF(10,I) = 0.
C
C  Interval:       X = (1-T) * WPIN(1) + T * WP(1),
C                  Y = (1-T) * WPIN(2) + T * WP(2),
C                  Z = (1-T) * WPIN(3) + T * WP(3),          0 < T < 1.
C
C  To find intersection points, we should solve the
C  equation:       A * T^2 + 2 B * T + C = 0,                0 < T < 1.
C-----------------------------------------------------------------------
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /SURFACE/SF(10,60)
     *       /WRPRIN/WPIN(9)
     *       /WRPR/WP(9)
     *       /DXYZ/DX,DY,DZ
      DIMENSION RC1(3),RC2(3)
C-----------------------------------------------------------------------
      IND1=-1
      IND2=-1
      T1=0.0D0
      T2=0.0D0
      DO 1 J=1,3,1
      RC1(J)=0.0D0
 1    RC2(J)=0.0D0
C-----------------------------------------------------------------------
C     Coeficients for the equation:
      A=SF(1,I)*DX*DX+SF(2,I)*DY*DY+SF(3,I)*DZ*DZ+
     +2.0D0*(SF(4,I)*DX*DY+SF(5,I)*DY*DZ+SF(6,I)*DZ*DX)
      B=SF(1,I)*DX*WPIN(1)+SF(2,I)*DY*WPIN(2)+SF(3,I)*DZ*WPIN(3)+
     +SF(4,I)*(DX*WPIN(2)+DY*WPIN(1))+
     +SF(5,I)*(DY*WPIN(3)+DZ*WPIN(2))+
     +SF(6,I)*(DZ*WPIN(1)+DX*WPIN(3))+
     +SF(7,I)*DX+SF(8,I)*DY+SF(9,I)*DZ
      IF(DABS(A).LE.1.0D-10.AND.DABS(B).LE.1.0D-10)RETURN
      C=SF(1,I)*WPIN(1)**2+SF(2,I)*WPIN(2)**2+SF(3,I)*WPIN(3)**2+
     +2.0D0*((SF(4,I)*WPIN(2)+SF(7,I))*WPIN(1)+
     +(SF(5,I)*WPIN(3)+SF(8,I))*WPIN(2)+
     +(SF(6,I)*WPIN(1)+SF(9,I))*WPIN(3))+
     +SF(10,I)
C-----------------------------------------------------------------------
```



```
C-----------------------------------------------------------------------
      IF(DABS(A).GT.1.0D-10)GO TO 2
C     If A=0, then the equation is the one of the first order.
C     Root of the equation:
      T1=-0.5D0*C/B
      IF(T1.LE.0.0D0.OR.T1.GE.1.0D0)RETURN
C     If T1 is in (0,1), then the intersection point:
      IND1=+1
      RC1(1)=WPIN(1)+T1*DX
      RC1(2)=WPIN(2)+T1*DY
      RC1(3)=WPIN(3)+T1*DZ
      RETURN
C-----------------------------------------------------------------------
C     Determinant of the equation:
 2    D=B*B-A*C
      IF(D.LE.0.0D0)RETURN
      DET=DSQRT(D)
C     Roots of the equation:
      T1=(-B-DET)/A
      T2=(-B+DET)/A
C-----------------------------------------------------------------------
      IF(T1.LE.0.0D0.OR.T1.GE.1.0D0)GOTO 3
C     It T1 is in (0,1), then the first intersection point:
      IND1=+1
      RC1(1)=WPIN(1)+T1*DX
      RC1(2)=WPIN(2)+T1*DY
      RC1(3)=WPIN(3)+T1*DZ
 3    IF(T2.LE.0.0D0.OR.T2.GE.1.0D0)RETURN
C     It T2 is in (0,1), then the second intersection point:
      IND2=+1
      RC2(1)=WPIN(1)+T2*DX
      RC2(2)=WPIN(2)+T2*DY
      RC2(3)=WPIN(3)+T2*DZ
C-----------------------------------------------------------------------
      RETURN
      END
```

```
      SUBROUTINE CHECK(I,X,Y,Z,IND)
C-----------------------------------------------------------------------
C     CHECK: DOES THE INTERSECTION POINT BELONG TO A ZONE OF NUMBER I ?
C     IF IND > 0, THEN YES.
C     IF IND < 0, THEN NO.
C-----------------------------------------------------------------------
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /CONDITION/CN(9,60)
C-----------------------------------------------------------------------
      IND=+1
C     Selection of a condition type :
      IF(CN(9,I).EQ.1.0D0)GO TO 1
      IF(CN(9,I).EQ.2.1D0)GO TO 21
      IF(CN(9,I).EQ.2.2D0)GO TO 22
      IF(CN(9,I).EQ.2.3D0)GO TO 23
      IF(CN(9,I).EQ.3.1D0)GO TO 31
      IF(CN(9,I).EQ.3.2D0)GO TO 32
      IF(CN(9,I).EQ.3.3D0)GO TO 33
      IF(CN(9,I).EQ.4.0D0)GO TO 4
C-----------------------------------------------------------------------
 1    CONTINUE
C     Parallelepiped :
      IF(X.GT.CN(1,I)-1.0D-10.AND.X.LT.CN(2,I)+1.0D-10.AND.
     +   Y.GT.CN(3,I)-1.0D-10.AND.Y.LT.CN(4,I)+1.0D-10.AND.
     +   Z.GT.CN(5,I)-1.0D-10.AND.Z.LT.CN(6,I)+1.0D-10)RETURN
      GO TO 100
C-----------------------------------------------------------------------
 21   CONTINUE
C     Cylinder along x-axes :
      IF(X.GT.CN(1,I)-1.0D-10.AND.X.LT.CN(2,I)+1.0D-10.AND.
     +   DSQRT((Y-CN(3,I))**2+(Z-CN(5,I))**2).LT.CN(7,I)+1.0D-10)RETURN
      GO TO 100
C-----------------------------------------------------------------------
 22   CONTINUE
C     Cylinder along y-axes :
      IF(Y.GT.CN(3,I)-1.0D-10.AND.Y.LT.CN(4,I)+1.0D-10.AND.
     +   DSQRT((Z-CN(5,I))**2+(X-CN(1,I))**2).LT.CN(7,I)+1.0D-10)RETURN
      GO TO 100
C-----------------------------------------------------------------------
 23   CONTINUE
C     Cylinder along z-axes :
      IF(Z.GT.CN(5,I)-1.0D-10.AND.Z.LT.CN(6,I)+1.0D-10.AND.
     +   DSQRT((X-CN(1,I))**2+(Y-CN(3,I))**2).LT.CN(7,I)+1.0D-10)RETURN
      GO TO 100
C-----------------------------------------------------------------------
```



```
C-----------------------------------------------------------------------
 31   CONTINUE
C     Cone along x-axes :
      IF(X.GT.CN(1,I)-1.0D-10.AND.X.LT.CN(2,I)+1.0D-10.AND.
     +   DSQRT((Y-CN(3,I))**2+(Z-CN(5,I))**2).LT.
     +   CN(7,I)+(X-CN(1,I))*(CN(8,I)-CN(7,I))/(CN(2,I)-CN(1,I))+
     +   1.0D-10)RETURN
      GO TO 100
C-----------------------------------------------------------------------
 32   CONTINUE
C     Cone along y-axes :
      IF(Y.GT.CN(3,I)-1.0D-10.AND.Y.LT.CN(4,I)+1.0D-10.AND.
     +   DSQRT((Z-CN(5,I))**2+(X-CN(1,I))**2).LT.
     +   CN(7,I)+(Y-CN(3,I))*(CN(8,I)-CN(7,I))/(CN(4,I)-CN(3,I))+
     +   1.0D-10)RETURN
      GO TO 100
C-----------------------------------------------------------------------
 33   CONTINUE
C     Cone along z-axes :
      IF(Z.GT.CN(5,I)-1.0D-10.AND.Z.LT.CN(6,I)+1.0D-10.AND.
     +   DSQRT((X-CN(1,I))**2+(Y-CN(3,I))**2).LT.
     +   CN(7,I)+(Z-CN(5,I))*(CN(8,I)-CN(7,I))/(CN(6,I)-CN(5,I))+
     +   1.0D-10)RETURN
      GO TO 100
C-----------------------------------------------------------------------
 4    CONTINUE
C     Sphere :
      IF(DSQRT((X-CN(1,I))**2+(Y-CN(3,I))**2+(Z-CN(5,I))**2).LT.
     +CN(7,I)+1.0D-10)RETURN
      GO TO 100
C-----------------------------------------------------------------------
 100  IND=-1
      RETURN
      END
```

Литература

1. *Барашенков В.С., Соловьев А.Г. Сосник А.Н.* Алгоритм расчета пересечений частицей зон в гетерогенной среде (Геометрические модули программы моделирования межъядерных каскадов). Сообщение ОИЯИ **Р2-98-221**, Дубна, 1998.

Рукопись поступила в издательский отдел
23 апреля 1999 года.