UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA

*Trabajo de Investigación para el Programa de Doctorado en Tecnologías Industriales*

# MCNP5 improvements in memory requirements for geometry storage and lost particles during transport simulations

Autor: Javier Alguacil Orejudo
Tutor: Patrick Sauvan
Departamento de ingeniería energética
E.T.S. Ingenieros Industriales
2017

**MCNP5 improvements in memory requirements for geometry storage and lost particles during transport simulations**

Trabajo de investigación presentado por Javier Alguacil para el Programa de Doctorado en Tecnologías Industriales, siendo tutor del mismo Patrick Sauvan.

VºBº del Tutor: | Alumno:

Dr. D. Patrick Sauvan | D. Javier Alguacil

**MCNP5 improvements in memory requirements for geometry storage and lost particles during transport simulations**

(a rellenar por el tribunal calificador)

**TRIBUNAL CALIFICADOR**

PRESIDENTE:

VOCAL:

SECRETARIO:

FECHA DEFENSA:

CALIFICACIÓN:

| Vocal | Presidente | Secretario |
| --- | --- | --- |
| Fdo: | Fdo: | Fdo: |

PALABRAS CLAVE:

CÓDIGOS UNESCO:

PROGRAMA DE DOCTORADO EN TECNOLOGÍAS INDUSTRIALES E.T.S. INGENIEROS INDUSTRIALES

**MCNP5 improvements in memory requirements for geometry storage and lost particles during transport simulations**

## AUTOR:

Javier Alguacil

## TUTOR ACADÉMICO:

Patrick Sauvan

## RESUMEN:

En este trabajo se ha realizado un estudio sobre el método de almacenamiento en memoria de la geometría por el código MCNP. En este estudio se ve que el factor dominante en términos de memoria reservada es el almacenamiento de las definiciones de las celdas. Un análisis más detallado revela que la memoria reservada para este fin está sobreestimada. Para su reducción se propone, por un lado una guía a seguir durante la elaboración de la geometría, y también una modificación que recorta la parte de memoria desaprovechada.

Por otra parte se estudia el transporte de partículas por MCNP, particularmente la parte correspondiente al cambio de celda, donde se han observado la pérdida partículas, una al cruzar a una celda contenedor llena de un universo transportado, y la otra al cruzar de una celda contenedor a una celda contenido definida con planos coincidentes nombrados en la dirección contraria al movimiento de la particula. En ambos casos se proponen modificaciones del código para solucionar el problema.

## PALABRAS CLAVE:

MCNP, Geometria, Particulas Perdidas, Transporte

PROGRAMA DE DOCTORADO EN TECNOLOGÍAS INDUSTRIALES E.T.S. INGENIEROS INDUSTRIALES


## ABSTRACT:

In this work a study about geometry the storage of the geometry in RAM by MCNP is done. The most relevant factor about the reserved memory is identified as the geometric definition of the cells. Furthermore, a more deeply study revels that this size is overestimated. In order to generate a more efficient geometry in terms of memory requirements, a guideline is presented. A code modification that cut down excess of memory is also presented.

In the other hand, a description of the transported, realized by MCNP, is done with special attention when the particle enter in a new cell, because some types of lost particles events have been observed when the particle crosses a surface. A type of event appears when the particle moves from a container cell filled with a translated universe. The other event happens when the particle moves from container cell to content cell which has a coincident surface defined opposite to particle direction. For both cases, a code modification is proposed in order to perform a correct transport



## KEYWORDS:

MCNP, Geometry, Lost Particles, Transport

# Index

# Figure List

# Table List

# 1. Introduction and Objectives

This work has been realized in the group TEC3FIR, whose members work in the nuclear fusion investigation fild. In order to realized their analysis, activation and transport codes are highly needed. Some of these analysis, where these codes are employed, have to answer questions with very specific boundary conditions (these include a lot of specific geometric details). Some of these code simulate transport of neutrons and photons in order to give responses.

Many codes are able to simulate the transport of these particles, particularly, one of the codes used in the group is MCNP (see section 1.1). One of the advantages of MCNP is that they are other tools, that can handled the build of these complex geometries (that contains all specific geometric details). These tools have been grown while the last years, together with the grown of these tools, and the bigger necessity of more exact answers, the computational resources that are needed to achieve the answers are also growing, and MCNP capacities are being tested to the limit.

The results of this work try to contribute to the development of better computational tool based on MCNP code (see section 1.1), in terms of efficiency and precision.

For this, we take notice of two problems that have been observed in current ITER calculations (see section1.2). One of them refers to the reserved memory in RAM, and the other is about the particle transport.

MCNP is often run in mpi mode. In these conditions, the RAM memory needs to run the case should be multiplied by the number of processes in a computational node.

Currently commercial nodes have about 120GB per node with 32 processors per node. In mpi mode, this means that, if all processors are used, each process has available 3.75GB of RAM memory. The description of the current geometry models from some fusion facilities like ITER already takes up this amount of memory. Beside, it is necessary to reserve more memory for other thing, as tallies (a tally is a calculated result by MCNP). This way, if all processors are used, any of them has not enough memory to run the case. This situation is solved using less processes per node, in order to each process has more memory. This means that a great amount of computational resources are wasted. This has a direct impact on the economics and calendars of nuclear analysis.

The section 2 presents a study of MCNP geometry definition in terms of memory consumption. A guideline for geometry elaboration is proposed in order to do inputs wasting less computational resources. A more optimized solution that has required a modification of the MCNP source is also presented.

In order to achieve complex geometric description in a reasonable time, some programs have been developed. These programs help to build MCNP geometric files from CAD models, making use of algorithms. The geometries produced by these programs may give rise to new special situations during transport that the code has not taken into account and, for this reason, some particles are lost. This is the second type of problem that is treated in section 3.

In that section, MCNP particle transport is described, together with the two types of lost particles events. When MCNP loses a particle, it reports a message where details about the trajectory of the particle are written. In special, this data includes the last surface that the particle crosses and the point and the cell where the particle is lost. The particularity of these lost particles events is that the point where the particle is lost, belongs to the surface, but the surface is not used to define the cell where the point is. This means that the particle is lost in middle of a cell, after to cross a surface that is not actually defined there. A code modifications of MCNP, that have taken into account these situations, are proposed.

The section 4 shows how lost particles affect to the considering results, for this, volume calculation are done with MCNP in regions where the particles are lost. Finally, conclusions of this work are shown in section 5.

## 1.1. MCNP

MCNP is a general purpose code for solving the radiation transport equation (neutrons, photons and electrons), in a general geometry and employs Monte Carlo method to achieve this.

This code has been development by Los Álamos National Laboratory (LANL) in USA, and currently is considered the reference code for most of transport analysis in ITER. Besides, many codes takes MCNP as code base for transport of particle and include new features, for example the capability of the calculating residual dose (D1S or R2S codes) [3].

A code based on Monte Carlo is a code that use pseudo-aleatory numbers for solving this problem. MCNP uses these numbers to sample interaction probabilities and estimate the point where the particle collisions, and simulate possible trajectories.

Monte Carlo codes calculate the different magnitudes as the average of all samples realized (equation [1]). A good feature of this method is that it is possible to estimate the statistical uncertainty of the average(equation [2]).

$$\bar{x} = \int_R x \cdot p(x) dx \approx \frac{1}{N} \sum_i^N x_i(p(x)) \tag{1}$$

where $\bar{a}$ is the average of $a$ and $p(x)$ is the probability density function.

$$\sigma_{\bar{x}} \approx \frac{1}{\sqrt{N}}\sqrt{\bar{x^2} - \bar{x}^2} \quad (2)$$

The desired magnitude in problems of transport is usually the flux or any magnitude derived from the flux.

In this work, the manuals of MCNP have been employed as guide while the code was studied and modified [4][5][6].

## 1.2. ITER

ITER (International Thermonuclear Experimental Reactor) will be the largest tokamak in the world and is currently under construction in Cadarache (France) as a project of Europe, Japan, Russia, USA, China, India and South Korea. Some of ITER principal objectives are the production of 500MW of power during more than 6 minutes with only 50MW input power with a deuterium-tritium plasma in which the reaction is sustained through internal heating, and guarantee that in future reactors, tritium can be obtained in the reactor by interaction of neutrons with special ceramics. In this experimental reactor, studies about correct work of different components under high levels of neutron flux will be realized [1] [7].

The principal elements of ITER are showed in figure 1. Some of these are: 10.000 tonnes of superconducting magnet at 4 Kelvin will initiate, confine, shape and control the ITER plasma from 51 Gigajoules of magnetic energy. The cryostat which ensure an ultra-cool vacuum environment will be the greatest in the word with 16000m$^3$ of which 860 will be occupied by the plasma.

To ensure safety of the components before of the construction, many studies have to be realized. Some facilities will be built to experimental studies in order to the behavior of the components can be researched in ITER physical conditions. The other type of studies to guarantee the correct work of these components is the simulation [1].

Studies about dose of people would receive are necessary in order to plan maintenance of ITER components. In the absence of experimental data, simulations are the principal form to face the problem. MCNP can calculate dose maps from the installation.

Other types of studies where computational tools have proved to be useful is in the study of future treatment of ITER components when its useful life ends. Or the possible people damage that a fail in ITER would cause [8].

To be able to solve these issues accurately, new computational tools are developed and tested [3]. This is also the final aim of this work.

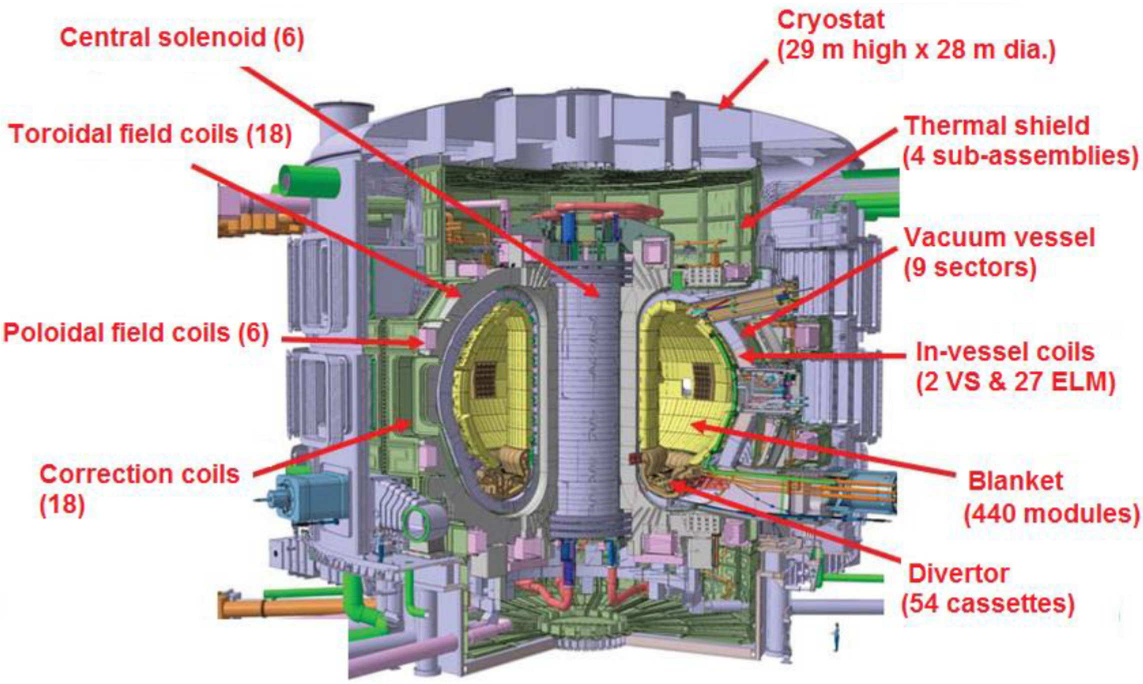


Figure 1: ITER tokamak [1]

# 2. Geometry storage in memory

The first part of this work is about the memory that MCNP reserves to the geometry description. The necessity of this study arose with ITER geometries. Specially in simulations with C-model.

C-model is the latest MCNP representation of ITER inside the bio-shield. It models a region of 16 m×9 m ×30 m with details in the range of 1 mm for thousands of components. In the figure 2 a image of the model is shown.

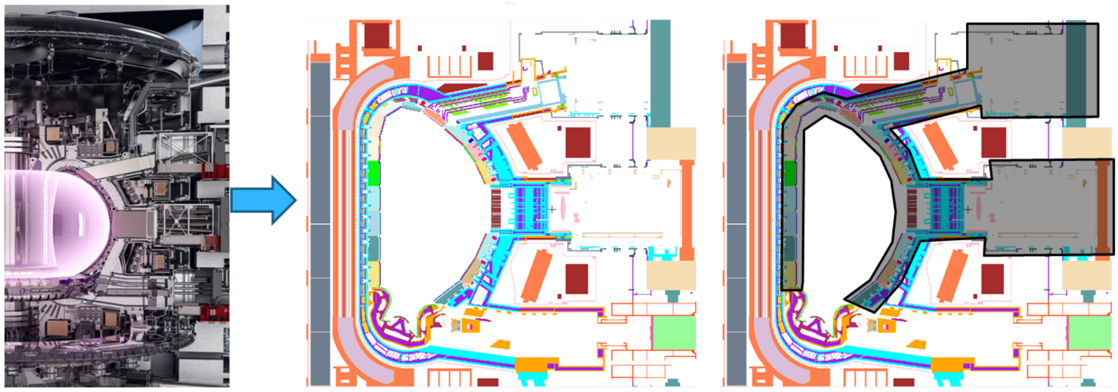

Figure 2: ITER bio-shield and C-model plane [2].

In the next section, necessary geometric concepts to build models to MCNP will be introduced. After, a description of the most important parameters which defines the memory consumed by geometry storage. The last part of this section presents results about a code modification that reduces the currently requirements of memory.

## 2.1. MCNP geometry description

A brief description of the general component used to build the geometry is realized. Its code interpretation is also presented (this means, as definition is stored in RAM memory). Finally, the results of which of these components is the most important, in terms of the memory reserved for its storage, are presented.

There are three important elements which are required for define the geometry, these are surfaces, cells and universes. MCNP also allows to use another elements as repeated structures (lattices), however this kind of description is not usual in ITER geometric descriptions, hence, this type of definitions will not be described in this work.

A surface is defined by a equation. MCNP only allows quadratic surfaces (equation [3]), and torus (the equation [4] is the equation of a torus orientated on X direction).

$$GQ = A \cdot X^2 + B \cdot Y^2 + C \cdot Z^2 + D \cdot XY + E \cdot YZ + F \cdot ZX + H \cdot X + I \cdot Y + J \cdot Z + K \quad (3)$$

$$TX = \frac{(x-\bar{x})^2}{B^2} + \frac{\left(\sqrt{(y-\bar{y})^2 + (z-\bar{z})^2} + A\right)^2}{C^2} - 1 \quad (4)$$

Each equation, [3] and [4], divides the space in two region, depending on whether the equation is positive or negative in this region.

A cell is defined as union or intersection of the regions defined by the signed surfaces. The exclusion operator is also allowed in a cell definition. There, this operator is applied to a cell or a region, and its effect is to remove, from the defined volume, this cell or region. A cell can contain a material (called content cells) or another universe (container cells).

The last concept is the universe, each cell belongs to only one universe. In order to have a good definition of the geometry, there should not be non defined regions, or regions defined by more than one cell into the same universe. The different universes are connected through containers cells.

If a particle is being transported in a universe, and enter in a container cell, it is moved to the another universe (indicated in cell definition), and continue moving in this new universe while the particle is into this container cell. The transport of particles will be treated more in depth in section 3.

Below, the way how MCNP inputs are built and interpreted, and how MCNP saves this information, is described.

MCNP inputs are divided in three blocks, definition of cells, definition of surfaces, and the rest of specifications about the simulation, as source, physical considerations, number of histories or tallies, etc.

In surface definition, non zero parameters of equations [3] and [4] are given. MCNP saves the type of surface and its parameters.

If there are two equal surface type whose differences between all its parameters satisfy

$$|P_i - P_j| \leq |P_i + P_j| \cdot 10^{-12} \quad (5)$$

where $P_i$ is the parameter of surface $i^{th}$, then, one of them is eliminated from cell definitions (not from the RAM memory).

Planes have special considerations which are relevant for MCNP transport of particles (explained in section 3.1). These considerations are related to the concept of coincident planes. In this section, this concept of two planes could be understood as two planes which are considered identical because they are very close to each other. This concept is well defined in the section 3.1 by means of the explication of its features during the transport.

The first common feature of the two coincident planes is its orientation. In this sense, MCNP classifies the planes in groups of parallels planes according to their original definition in the second block of the input. In order to realize this classification, a number is associated to each surface. This way, for example all planes px ($x = const$) have a same associated number, which is different than the number associated to planes py or any other orientation.

Besides the orientation, MCNP also takes into account the distance between the two planes in order to consider these planes coincident planes. In fact, MCNP considers that two planes are coincident if these planes are parallel (in its definition) and the distance that particle has to walk between them is less than a constant parameter *coincd* (by default $10^{-4}$, notice that this distance is much larger than the distance to two equal surfaces expressed in equation [5]). MCNP takes into account this concept only if more than one universe exists. These planes are not replaced or eliminated, and its special considerations will be developed in section 3.

In cell definition, the following information can be given:

- Name of the cell
- Material and density
- Description of the region of Cell
- Universe Card
- Fill Card
- Lattices Card
- Like n But Card
- Importance Cards

A cell with a *fill* card in its definition is a container cell. This card contains the name of the universe with which this cell is filled, and the possible transformation of this universe in order to the universe fits in this cell in the desired way.

If a translation is realized with a *fill* card, a plane of the content cell may get closer to another parallel plane, making them coincident although they are not in theirs original definition. However, if a rotation is realized and two planes are closer than *coincd* distance after of the rotation, these two planes are not coincident because these are not parallels in the MCNP classification, this means, they are not parallels in its original definition.

For example, considering that we have a plane $py = 0$ and $px = 0$, one of them is a surface of a container cell, and the other is a surface of content cell. If a rotation of $\pi/2$ is applied to the universe, the two planes are physically identical, but since a $px$ is not parallel to $py$, MCNP does not consider them coincident.

Transformations can also be done to surfaces, but in this case theirs parameters are modified before to be saved in RAM. This means, it is like if the final definition of the surface would be written in the input file before starting run MCNP.

## 2.2. Size of cell definitions.Guideline

In this section, we are focusing in the most important parameter related to the reserve of memory for the geometry storage. First, a brief description of the determination of this parameter is presented. Before, the estimation of this parameter is studied in terms of the input description of the geometry. According to this study, a guideline for reducing the reserved memory for the geometry storage is done.

Thus, the first presented description is relating to search of the most important parameter in terms of reserved memory for the geometry storage.

Most of array's dimension in MCNP only are defined one time. In order to estimate these dimensions, MCNP reads the input before loading the input into the RAM. We compare these estimations of some inputs with the purpose of determinate the most important factor in waste of consumed memory for geometric definition. The results point to the description of the cells as the most important factor in terms of memory reserved to storage of the geometry. The difference with the next most import factor is several orders of magnitude.

The geometries used in this work are:

- FNG [9]
- JET [10]
- Four different versions of Clite [11] [12]
- Cmodel [13]

This result may be conditioned for the type of inputs studied, because there is almost not lattices.

The study of this parameter is presented right before.

This parameter, that define the reserved memory for the cell definitions storage, is the size of the array where all cells descriptions are stored. The value of this parameter is stated in equation [6].

$$Long = \sum_{i=1}^{N_{cel}} \alpha_i \cdot Nws_i + 2 \cdot 17 \cdot Nw_M \cdot Nop + \sum_{i=1}^{N_{FS}} Nfs_i \qquad (6)$$

where *long* measures the number of integers that are needed to store cell definitions, this parameter is calculated by a iterative method, and its value is redefined each time that a cell is read from the input file (this is that a new term of the sum in the equation 6 is added), $\alpha_i$ can be 7 or 17 depending of the *long* value when its estimator is incremented while the geometry is been reading. Usually $\alpha_i = 7$ for most of this estimator increments in ITER inputs. $N_{cel}$ is the number of cells defined in the geometry, $Nw_i$ is the number of words used in the definition of cell $i^{th}$, and $Nw_M$ is the greater value of $Nw_i$. $Nop$ is the number of exclusion operators in all input file.

In this context, a word is understood as a surface, parenthesis, exclusion operator or union operator. Intersection operator is implicit between two regions.

The last term refers to segment options tally $FS$, $N_{FS}$ is the number of tallies with segment option $FS$ that are defined, and $Nfs_i$ the number of surfaces used in the segment option tally $i^{th}$ (a tally is a result ordered by the user). This tally calculate flux (or any magnitude related) in regions of a cell delimited by these surfaces.

The array used to stored the cell definitions should reserve enough space in the memory to be able to store all word used in the input for definitions of the cells (see equation [6]). Besides, a extra space is reserved for exclusion operators (the second term in the equation [6]). Both terms are overestimated because MCNP needs this space in the memory to realize operations during the processing of the geometry. At the end of this process, the array for cell definitions storage is ready for its use during the transport. This means that all exclusion operators have been expanded, and all extra parenthesis have been removed. At this moment, when the geometry has just been processed, the array can be divided in two part. The first part is the useful data that contains the definition of the cell, and the second part is the extra space reserved in memory for the operation during the process of the geometry. This part of the vector will not be used anymore. This second space contains only zero values.

In order to quantify the size of each term of the equation [6], the values of the terms were obtained from some MCNP inputs, locating in the code the position in which each one of this terms is evaluated.

In table 1, the different terms of equation [6] are shown for different geometries presented in section 2.1. These data were obtained of inputs where any tally was defined so, the term of segment option tallies is always zero and is not presented. Data contained in the second and third column refers to the useful and the total number of integer reserved for cell definition storage respectively. The data of the next two columns are the values of the two terms of the equation [6] relatives to the parameter *long* (third column data). The last two columns of data refer to the largest number of word used to defined a cell, and to the number of exclusion operators in the input file. The second term of the equation 6

| **Input** | **Used Long** | ***Long*** | $\sum_i \alpha_i Nw_i(\%)$ | $34 \cdot Nop \cdot Nw_M(\%)$ | $Nw_M$ | $Nop_e$ |
|---|---|---|---|---|---|---|
| fng | 1723 | 15712 | 100 | 0 | 49 | 0 |
| JET | 20092 | 1309762 | 4.49 | 95.51 | 84 | 438 |
| Clite 1 | 153254 | 2873961 | 47.13 | 52.87 | 1090 | 41 |
| Clite 2 | 938741 | 25636536 | 30.32 | 69.68 | 1090 | 482 |
| Clite 3 | 1179745 | 108356482 | 8.72 | 91.28 | 2108 | 1380 |
| Clite 4 | 1242727 | 58255565 | 17.39 | 82.61 | 1313 | 1078 |
| Cmodel | 8143423 | 241033506 | 27.18 | 72.82 | 3261 | 1583 |

Table 1: Relevant parameters, which define the reserved memory for the geometry storage, described in equation [6].

can be obtained with these last two parameters.

The overestimation of the memory for cell definition storage can be seen if the data of the second and third column is compared. If the second term of the equation [6] is not zero (this means that there are exclusion operators), most of the memory reserved for the cell definition storage is due to this second term, at least in the studied cases. Below, the guideline about the optimization of the memory reserved for the cell definition storage is presented.

- Expanding exclusion operators before run MCNP. This way, the second term is null.

  The total number of words would be greater, however, as the data of the table 1 shows, the most relevant term that defines the total number of words for the cell definition storage is set to zero.

  To carry out this expansion, we have to copy excluded cell definition in the place where operator is, and to include next changes:

  - Change surface signed.
  - Change union operator to intersection operator and the contrary.

- If we could not eliminate the second term, we can try make it smaller.

  The other factor is the greatest number of words that define a cell, despite the rest of cells can be much smaller than this one (in number of word used to the definition), the second term of the equation [6] takes the same value, so the memory reserved for the cell definition storage is the same. It would better that all cells have, more or less, the same number of words in its definition. In order of achieve this, the bigger cells can be divided in smaller cell.

  In this sense, ITER suggests limiting the number of surfaces per cells in building MCNP geometry. This limits, somehow, a maximum number of words of the greatest cell.

- Last tip is about the first term of the equation [6], the total number of words. There is not only one way to define a same region, and different ways require different size of reserved memory.

  For example, when voids regions are built with old versions of MCAM [14], the void cells have a structure of intersections of big regions defined as union of a lot of regions. However newer version of MCAM builds the void cells with a structure of unions of big regions defined as intersections of a lot of regions. This last version reduces the number of union operators and increases number of intersection operators but, as intersection operator is not a word, the total number of words that defined a void cell is reduced with the newer version of MCAM.

  We do not know an efficient method of building cells, however we think important to keep this fact in mind for building geometry.

## 2.3. Code Modification

Although guideline would be applied, a large amount of reserved memory is not necessary. Equation [6] shows that even the best definition of the input will overestimate the necessary memory for the definition cell storage, this is due to $\alpha_i \neq 1$. For example, the data about the FNG model in the table 1 can be seen. The input file of FNG model has not any exclusion operator, so when MCNP processes the definition of the cells, it has not to expand any term. So the cell definitions are almost copied word by word from the input to the array where the definitions are stored in the RAM. Maybe a few less words are copied because in the definition of some cells can be unnecessary parenthesis, however the relevant term responsible of the overestimation of the memory reserved is the existence of the term $\alpha_i$ in the equation [6] (over 9 times more memory is reserved). For this reason, a code modification that allows to release the useless part of the vector that stores the cell definitions was implemented.

The implemented option for the optimization of the reserved memory is to resize the array containing the cell definitions to the real size used during the transport, this means that the part of the array containing only values of zero is cut down. Furthermore, as MCNP realizes some checks over all array, if this cut is realized before of these checks, the time spent to perform these checking will be reduced. However, the cut region of the array is only given up using after of the expansion of the exclusion operators. For these reasons, the localization of the code, where the cut is realized, is just after of the expansion of the exclusion operators. This place is before the checks over the array containing the cell definition, hence a time solving in the checks is achieved.

To guarantee that this modification only cuts down memory containing useless information, a check that certify all values eliminated are zero is included.

This section tries to quantify the gain in time and memory. For that, the same geometries named in section 2.1 have been employed.

For the time measurement, the input loading and processing was evaluated, cross sections loading was not taken into account. In MCNP code, this correspond to the spent time in the subroutine *imcn*. These values were obtained with fortran subroutine *cpu_time* which has a precision of 1ms. This time is much smaller than measured times. Other possible source of uncertainty is the cpu efficiency, this quantity has not been taken into account, however the time of the measure was long enough to have a good average of the efficiency. For Clite 2, 3, 4 and Cmodel toke more than an hour.

Two magnitudes was used to measure the memory used for the geometry storage.

The first of them was the value of *long* and the size at which the array containing the cell definitions was cut down (this data is contained in the second and third columns in table 1 for all used models), this magnitudes are used to see how much memory is gained respect to the original consumed memory for cell definition storage.

The second of these magnitudes was the *runtpe* size. The *runtpe* is a file where all necessary data to continue the simulation is recorded, it includes geometry data, cross sections data, tallies data, etc. The difference between the size of this file, and the total RAM memory used for run the case is small. For this reason, this magnitude is used as estimator of the memory gain with respect to the necessary memory to carry out the real case with the unmodified MCNP version. When these cases were run, no tallies were requested.

The results are presented in table 2. This data shows that a reduction of the memory used to store the geometry, is always obtained. However, the gain of the total memory reserved to carry out the case, is case depending, because it depends on the relative reserved memory for the geometry storage and the total reserved memory.

For example, in the case of FNG, the memory containing the geometry definition is less than a 1 %. The rest of the data contains cross section, or source definition, ect. This model has the simplest geometry studied. The most complex is Cmodel, in this case, more than 70 % of the memory was reserved to contains geometry definition (despite as this section shows, the most part of this memory is useless).

| **Input** | **Time Gain (%)** | **Geometry Size Gain (%)** | **Runtpe Size Gain (%)** |
|---|---|---|---|
| FNG | 3.22 | 89.03 | 0.3 |
| JET | 55.54 | 98.47 | 5.1 |
| Clite 1 | 30.75 | 94.67 | 11.3 |
| Clite 2 | 27.31 | 96.34 | 37.0 |
| Clite 3 | 61.14 | 98.91 | 69.4 |
| Clite 4 | 53.33 | 97.87 | 59.4 |
| Cmodel | 17.12 | 96.62 | 71.9 |

Table 2: Results about gain in time and memory.

A last verification of this code modification was done. In order to do this, both versions of MCNP(modified and unmodified) was used to run cases with tallies. The results obtained by both version are exactly the same in all run cases. The geometries used for this verification are all geometries used for obtain the data of the table 2.

Coming back to the problem for which this study has been realized. This problem was the memory reserved for CModel geometry description storage was too big. The memory available in current supercomputers like Marconi is about 120Gb per node. The CModel runtpe has about 3.7 GB. As Marconi has 32 processors per node, this is also the memory that each node would have available if all processor are used. As about of 70 % of the memory that is needed for run the case is released, with this modification, about 2GB are available to tallies, or other requiriments.

# 3. Lost particles

A lost particle event is a MCNP error. This error happens when the inputs parameters are not consistent with the logic of the MCNP method for the particle transport, and MCNP cannot continue the simulation of this particle. For example, the transport logic needs that, if a particle crosses a surface, a cell should always be defined behind this surface. This cell has to belong to a specific universe, and, beside this cell has to have this surface in its definition. This means that there is not undefined region (see section 2.1).

Some of these cases happen most often because the the geometry is not correctly defined. Few others happen because, the logic of the transport does not interpret correctly the cell definition.

The first case can be solved fixing the errors in the geometry definition, whereas in the second case the error is due to the code itself. This section describes two kinds of lost particle events where the logic of the transport does not interpret well the defined geometry despite the geometry is defined according to the rules presented in the manual [5].

These two types of particle events described in this section show coincident features, observed in the messege that MCNP reports when these events happen. This message contains data about the these lost particles events, including the last position of the particle, the cell to which this last position belongs, the last surface that this particle crossed, etc.

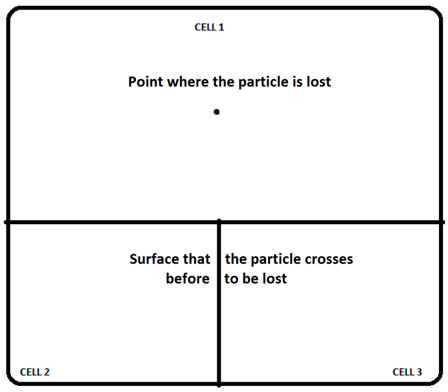


Figure 3: Simple representation of this type of lost particle

The figure 3 shows the coincident features of these lost particles events. All observed lost particles are lost in a point in middle of a cell. The point belongs to the cell that the message of MCNP indicates, and the point also belongs to the surface that the particle has crossed (according to the message of MCNP). However, this surface is not actually defined in this point (this means that this point does not belong to a cell boundary). In fact, this surface may not be used in the definition of the cell where the particle is lost.

In the example of the figure 3, the message reports that the particle is lost in a point of the cell 1, after the surface that divides the cell 2 and 3 was crossed. The point, where the particle is lost, satisfies the equation of the surface that divides the cell 2 and 3. However, this surface is not defined in this region. In order to this surface would be really defined in this region, this surface should to be used in the definition of the cell 1. However the cell 1 does not need to use this surface in its definition.

The other important common feature, that was observed while these lost particles events were studied, is that the wrong logic interpretation of the input data during the transport may not lose the particle, although, the transport of this particle is wrong. Despite all particles are not lost, in this work, we call 'lost' particle event to all events in which this erroneous logic interpretation can happen.

The two types of lost particles events treated in this section were observed in two different inputs of Clite and Cmodel. One of these types is seen when a container cell is filled with a universe which is translated to fit in the cell. The other type happens when a particle crosses from a container cell to a content cell, where this cell is defined with coincident planes. Furthermore, the scalar product between the direction of the particle motion and the normal of the plane that defines the content cell is negative (in this work, the normal of a surface, that define a cell, is defined in the direction of the region that this surface defines in the cell).

The geometric concepts about cells and planes are defined in section 2.1. In this section, there is only a description of the transport method, and the lost particles events conditions. The possible consequences of the wrong transport are exposed in the section 4.

## 3.1. Particle transport in MCNP

In this section, a brief description of how MCNP transports a particle is done. Special attention is paid to how change between cells, when a particle crosses through a surface, because it is in this point of the code, where the particle is lost. A change between cells is to change the particle parameter, that indicates the cell to which the particle localization belongs. The parameter changes from the cell where the particle was before to cross a surface, to the cell where the particle is after to cross this surface.

When a particle history starts, the value of the particle variables as type, weight, etc, is assigned according to input indications. Specially, the position and the direction of motion of the particle are assigned.

This assigned position, where the simulated particle starts, belongs to one cell. If this cell is a container cell filled with a universe, another cell in this new universe is looked for. This is done until the highest cell level where the particle is transported. This cell is filled with a material (vacuum is also understood as material).

In this context, the level is defined as the number of container cells containing the cell where the particle is actually moving.

When the all particle parameters are defined, the transport starts. The distance, that the particle will walk, is the smallest distance of all possible kinds of distances. These types of distance are associated to several process, for example, the distance to a possible collision, the distance to nearest surface, the distance to the DXTRAN detector, or some distances related with variance reduction. The argument of this section only considers that the particle can walk two types of distances (the possible collision, and the nearest surface), because the rest of possible distances are not relevant for these kinds of lost particles events that are studied in this work.

The first type of distance considered, is the distance to a possible collision [4].

The probability of a next interaction of the particle between $x$ and $x + dx$ in the direction of its motion is

$$p(x)dx = const \cdot \exp\left(-\sigma_t \rho x\right) dx \tag{7}$$

where $\sigma_t$ is the total cross section of the material, and $\rho$ is the density, in unit of number of targets in a defined volume. This density of probability $p(x)$ can be sampled as

$$x = -\frac{1}{const \cdot \sigma_t \rho} ln\left(\xi\right) \tag{8}$$

where $\xi$ is a random number between $[0, 1)$.

The other distance is the distance to the nearest surface. The process for obtaining the nearest surface is explained in section 3.1.3.

The distance, that the particle walks, is the smallest of all them.

When a distance is selected, the particle is moved to its next location and after, the rest of its parameters (cell, cross surface, etc) are modified depending the type of distance (to a surface, or a collision,etc). If after these modifications, the parameters allows to continue the simulation of this particle (it is means, the energy or weight is inside the range of the simulation), the distances are sampled again, else a new particle history is simulated.

If the nearest surface is selected, this means that the particle exits of the cell. Hence, the code has to look for the new cell where the particle goes in its transport. The next sections describe this process. A particular case of particle transport in the vacuum is detailed in Annex A.

### 3.1.1. New cell

The search of the most highest level cell, where the particle goes when it crosses a surface, is explained in this section. In the same part of the code where this search is realized, some important changes are done in the particle parameters. These tasks are carried out in a subroutine called *newcel*.

This search of the most highest level cell can be divided in two steps. The first step of the search is to found the cell behind the crossed surface. The second, that is only carried out if the found cell is a container cell, is the search of the cell in a higher level. This search is reapeated until the cell found is not a container cell (this means, the cell is a content cell, where the transport is carried out). Finally, this subroutine may also change the name of the crossed plane, if the cell definition contains a coincident plane.

For this subroutine, the parameters, obtained in the search of the nearest surface are important, just like the subroutine (*chkcel*), that allows to know if a point is inside a cell. The following description is about this subroutine. After a description of the known parameters for this search is presented. These parameters are the crossed surface, the crossed point, and the level of the cell.

*Chkcel* is a subroutine that works like a function of a point and a cell, and this subroutine returns a logical value refers to this points is or not inside the cell. This subroutine also returns an associated vector used to respond the question about the point is inside or outside of the cell. More details about this subroutine are described in section 3.1.2.

Important parameters for this subroutine are obtained in other subroutines, for this reason, the location in the code where this subroutine *newcel* is called, is important. This location is described right before.

If the code searches a new cell, it is because the particle has just crossed a surface. This means that the particle has just to moved the distance to the nearest surface. When this distance was calculated (see section 3.1.3), the name of the nearest surface, and the level of the surface used to define the cell, are also obtained. The other data, that code has in this moment, is the position of the particle, located on the crossed surface. This point is the same as the current particle position. The first two data (the universe, and the crossed surface) are needed for the search, the last one (the particle position), it is necessary for define a coincident plane when the algorithm changes the name of the crossed surface.

The following description is the search of the cell between of the crossed surface. For this search, a good defined geometry and the previous data are needed.

A good definition of the geometry implies that there is not undefined region. For this requirement can be achieved, a cell in the same universe, with the crossed surface in its definition (in order to any gap would be defined), should exist. Thus, a cell in the same universe as the cell the particle comes from, and with the crossed surface in its definition, is looked for.

The crossed surface is known, as it is showed before. The universe is not known, however, it is the same universe as the cell the particle comes from, and this cell is identified by the level where the surface is defined (because it is defined in this cell). Thus, the code determines the two parameters for the search of the cell behind the crossed surface. These are the crossed surface, and the universe where the cell that is searched, is defined.

When these parameters are known, the search, of the cell behind the crossed plane, is realized cell by cell, if the cell satisfies the previously conditions explained for these parameters, the code uses the subroutine *chkcel* (see section 3.1.2) to know if the localization of the particle is inside this new cell. If the geometry is correctly defined, only one of these cells can contains the localization of the particle.

When MCNP found a cell behind the crossed surface, the code starts the search of the cell in the highest level. If the found cell is a content cell, the search is not realized, because this is the cell in the highest level. Otherwise, a search in order to find the cell containing the localization of the particle in the next level is carry out. If the found cell is also a container cell, the search is realized once again until a content cell will be found. The condition of the cell that is being looked for each time, is that the cell belongs to the universe that fills the cell previously found. As in the previous case (when MCNP looks for the cell behind the crossed plane), when the cell satisfies the search criterion, the code uses *chkcel* to known whether the particle point is inside the cell.

Finally, a last process, in which the parameter that points to the crossed surface maybe modified, so as to this parameters points to a coincident plane, it is also realized in this part of the code. If the crossed surface is a plane, another coincident plane can be used to define the cell of the highest level containing the particle. If these two coincident planes exist, the parameter that points to the current crossed surface changes, and points to the coincident plane used to define the cell in the highest level. As they are coincident, they are actually the same plane although their name are different in their definition in the input. This change is important in other point of the transport, when the nearest surface is calculated (see section 3.1.3).

### 3.1.2. Inside or outside the cell. *Chkcel*

In this section, the process to answer the question about if a particular point, where the particle is located, is inside a specific cell, is described. This process is realized in the subroutine called *chkcel*. To achieve this aim, the algorithm builds a logic array (which is named logical function here) whose evaluation is the results of the question about if the point is inside the cell.

Before to explain how this function is built and how it works, some expressions often used in this work, whose use is not exactly, are explained. However, the use of these sentences make to the reading be easier.

In this work, a point 'inside a surface' is understood as the point that is inside the region defined by the surface, according to the sign of the surface in the definition of the region. By the same token, when a operator performs over a surface, it performs over the region defined by the surface.

How the logical function is calculated, is explained below.

To calculate the logical vector, the description of the cell is used, because it is a logical function of this question, where the signed surfaces are true (1) if the point is inside the surface. And the union and intersection are the *or* and *and* logical operators respectively.

For example, a point is inside a region defined as the intersection of two surfaces, if this point is inside both surfaces. This shows that the intersection operator performs just like the *and* logical operator. For the case of the union operator, a point is inside a region defined as the union of two surfaces if the point is inside, at least one of these surfaces. This shows that the union operator performs just like the *or* logical operator.

With this rules about the building of the logical function in mind, this function is written coping each word of the cell definition to a new array (logical vector). The surfaces can be used more than once in the cell definition, and with different signs. Each time a surface appears in a cell definition, its associated logical value (1 if the point is inside or 0 if it is outside of the signed surface) is written in the logical function. Each time this works will refer to a logical value of a surface, it refers to the value of a specific element of the logical vector.

The rules to know if a point is inside or outside of a surface are presented below, and these rules are based on the consistence with the rules used to describe the cell (see section 2.1). However, there are two exceptions, when the evaluated surface is the crossed surface, and when the evaluated plane is a coincident plane.

In order to know if a point is inside or outside the surface, the surface equation is evaluated in this particular point ($x, y, z$ in equations 3 or 4). The sign of this evaluation is compared with the sign of the surface in the cell definition. If these signs are the same, the point is inside.

The logical values of the two exception are calculated with the following rules:

- The evaluated surface is the same as the crossed surface.

  In this case, the evaluation of the equation should be zero. The particle enters by the side that its direction of motion points. For this reason, the point is considered inside if this direction points to the same side as the normal of the surface. This means that the scalar product between the direction of the particle motion and the normal of the surface is positive.

- Coincident planes (between the evaluated plane and crossed plane). This definition is only applied when a cell of a higher level is being looked for (see section 3.1.1).

  In this case, the coincident plane is the same as the crossed plane (in the MCNP logical). For this reason, the used rule is the same as the one applied to the crossed plane.

The definition of coincident planes is included, in the MCNP logic, to consider cases like the figure 4. In the example, a particle enters in a container cell crossing surface 1, but as the plane 1 and 2 are coincident (this means that the volume between the two surfaces does not exist actually) the particle goes to cell 2 directly. Even if this region is not defined in the input, this will not lead to geometric error. The introduction of coincident planes in MCNP logic allows to fit easier the universes to the cells containing this universes.

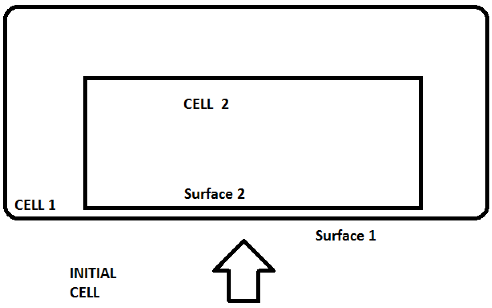


Figure 4: Particle crosses a coincident plane. The particle must go to the cell 2.

To prove that the definition of coincident planes allows the transport of the particle as previously described (the particle transport of the figure 4), the logical vector of the cell 2 is calculated right after.

An this point in the code, where this logical value is calculated (*newcel*), the particle has just crossed the surface 1, (its position is over this surface), the cell behind the surface 1 has been found, and the code is looking for in the universe content in that cell.

The cell 2 is a content cell that is described as a box. Thus, it can be defined as the intersection of all its surfaces (inwards). With this definition, the point is obviously inside all the surfaces except the surface 2. The evaluation of the logical vector in this case would be outside, because the point is really outside. However, if the surface 2 is coincident to the surface 1, the direction of the particle motion is the same as the normal of the surface used to define the cell 2, for this reason, the logic of this surface is inside. Thus, the point is inside the cell 2.

If this is done for the excluded region of this cell, the result is that the point is outside, thus the geometry is well defined (because the point is defined only in one cell). The excluded region is defined as the union of all surfaces (outward) of the box. The point is outside of all them except the surface 2, however, as this surface is coincident to 1, and the scalar product between the normal of surface 2 (outward of box), and the direction of the particle motion is negative, the logical value is 0 (outside the cell).

### 3.1.3. The nearest surface

This section explains how the nearest surface is searched. This surface depends on the particle trajectory. MCNP calculates the distance, between the point and the surface, along a line defined by the particle trajectory. In order to know if the particle leave the cell, MCNP ask if the point just behind the surface encountered in the particle trajectory, is outside of the cell. The nearest surface whose point just behind the surface is outside of the cell, is the nearest surface that the particle should crosses to leave of the cell. This search is realized in the cells at all levels. This means, in the content cell where the particle is being transported, and in all container cells that contents this content cell, because the nearest surface can be defined in any level. This is done in the subroutine *track*

When the algorithm starts the works, its first task is to calculate the distances, between the point and all the surfaces used to define the cell, along a line defined by the particle trajectory. However, if the particle has just crossed a surface, the algorithm rejects the smallest distance to this surface, because this should be zero. If the surface is plane, this plane is not considered because a line and a plane have only a intersection point. The rest of surfaces may have another intersection points.

The next task is to ask, in a orderly manner, point by point, if the point behind the surface is still inside the cell. In order to responds to this question, MCNP uses the logical vector, calculated in *chkcel* (see section 3.1.2). The evaluation of this vector is the answer whether the point is inside the cell.

When the subroutine starts, this vector is built at the point where the particle enters into the content cell. Instead of calculating all this function for each point, MCNP changes the logical value associated to the surface (of which the evaluated point is above). This logical value refers to the points is inside or outside of this surface (see section 3.1.2). If only one surface is crossed, only the logical values refer to this surface, and this way is faster to calculated the whole function to each point.

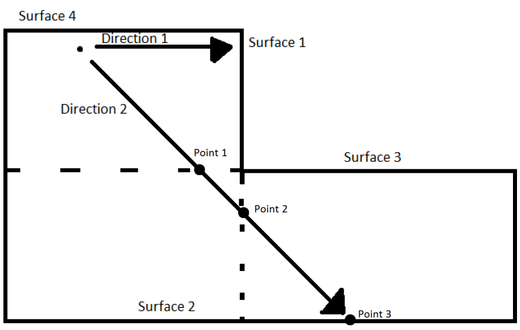


Figure 5: Example where different trajectories has different surfaces like the most nearest

For example, in the figure 5, two possible trajectories are printed. Each trajectories crosses a different nearest boundary surface. If we pay attention to the direction 2 in this figure, the ordered surfaces are 3, 1, and 2. The algorithm asks, in this order, if the point behind these surface are inside the cell, until one of them is outside. In the example case, points 1 and 2 (behind the surface 3 and 1 respectively) are inside the cell. The response for the point 3 (behind surface 2) is outside, this means that the nearest boundary surface, in this example, is the surface 2.

This same process is realized to all higher lever cells. For each cell, the logical function of the point, where the particle enters into the cell, is obtained using the subroutine *chkcel* (see section 3.1.2). As explained in that section, the logical values of coincident planes are calculated with the general rules (this means that the special rules of coincident planes are not used). This is consistent with the search method, as it is shown below.

In order to show this consistency, we can think in a case like the figure 4, where the cell 2 is a container cell. The particle crosses trough the surface 1. As this cell is not the highest cell, the crossed surface is not changed to surface 2 (see section 3.1.1).

If we think in the cell 2 as a box, it can be defined as the intersection of all its surfaces (inward). As the particle is over the surface 1, the logical value of the surface 2 (for cell 2) is outside, the rest of logical value, associated to the rest of surfaces, are inside. When the nearest boundary surface is searched, the surfaces are ordered according to their distances to the point (and a determinate trajectory). These ordered surfaces are the surface 2, and the surface to the end of the box. Thus, the first surface where the algorithms tries if the points behind is outside, is the surface 2. As the logical value associated to this surface changes (to inside), the logic response is that this point is inside. The next surface is the surface of the end of the box, when its associated logical value changes (to outside), and the logic response is outside. This is the correct surface found in this example.

## 3.2. Problem type 1

The first type of problem happens when the particle crosses a surface, and the next surface that the particle must cross is the same surface (this implies that both surfaces are called with the same name). For that can happen, a container cell must be filled with a translated universe. Thus, a cell inside this universe can be defined with the same surface, and as this universe has been moved, this means, this cell (with all its surfaces) has been translated, the particle can find this surface again.

MCNP only allows to define a container cell and content cell with the same surface if this surface is a plane, and this type of geometric definitions is not recommended. However, little translations are common, in ITER geometries, to fit well the geometry and solving lost particle events.

The following description presents a geometry example where this type of lost particles was found.

The figure 6 shows a simple scheme of geometry where this kind of lost particle events can

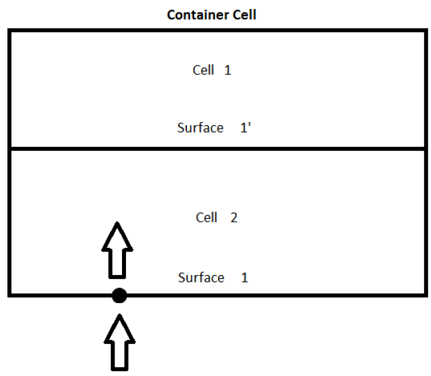


Figure 6: Scheme of geometry where this particles of type 1 can be lost.

be detected. As section 3.1 describes, when the particle enter in cell 2, the nearest distances are calculated. As the particle has just crossed the surface 1, MCNP does not calculate the distance to this surface, despite the translation is long enough for this distance has to be considered.

The simulation of this simple geometry, in vacuum, shows that the particle crosses the surface 1, and enters into the cell 2, and after, its next position is out of the box. The particle never enters in cell 1.

To solve this type of situations, a code modification has been realized. When the nearest surface is calculated, the surface, where the particle is located, is always calculated, however, this distance is only kept into account if the distance to this surface is long enough for the evaluated plane is not coincident with the crossed plane.

Thus, version of MCNP with this modifications also simulates situations like the figure 4, with the same results as unmodified version. Because, as the plane 2 is close enough to be coincident, this is the crossed plane (see section 3.1.1), and is not considered. Furthermore, in these new situations (like the figure 6), as the distance is bigger enough for the plane is not coincident with itself (because this plane has been translated), the surface is considered like a possible the nearest surface, and will be chosen if it is actually the closest surface (like the case of the figure 6).

Some inputs were run with this modification without found new lost particles and consistent results was obtained. These results were also compared with unmodified versions, and these results were consistent. This means, if there is not any lost particle of this type the result was exactly the same.

## 3.3. Problem type 2

The second type of lost particles events happens in geometries with universes, where coincident planes are defined in content cells. The problem happens because the rules, for calculation of the logical function, are not consistent with the rules for the search of the most nearest surface.

A simple case, where this error during the transport appears, is presented in this section. Another equivalent case, in terms of the transport, (it means, the particle trajectory should be the same in both cases) it is also presented. A comparison, between logical values of a specific surface,it points to the reason by the particle is lost. Finally, two possible modifications of the code are proposed.

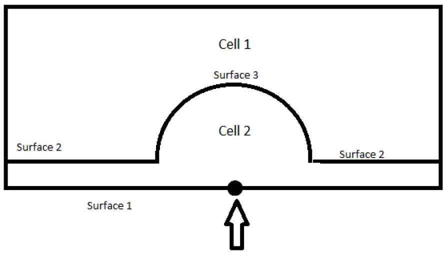


Figure 7: Scheme of geometry where this particles of type 2 can be lost.

The simple case is shown in the figure 7. The particle, in its trajectory, would enter into the cell 2 (by the surface 1), after, the particle would cross to the cell 1 (by the surface 3), and finally the particle would leave of the box.

The simulation of this case revels that the particle never enters into the cell 1. The whole transport of the particle, step by step, is detailed in the Annex A. The logical value (3.1.2) of the surface 2 is calculated respect of the cell 2 (described as the Annex A, in the input of the annex, this is the surface number 6).

All transport processes, that are used in the following description, are detailed in the section 3.1. The mistake is that MCNP does not find the correct nearest surface in a content cell. In order to find this surface, MCNP uses a logical vector, in special, the detailed study of the case points to the logical value of the surface 2 ( Annex A). In this case, as the surface is in the definition of a content cell, and the particle has just crossed a surface, this logical vector was obtained in the subroutine *newcel* (see section 3.1.1). It is due to the code point where this funciton logic was calculated that the logical value of coincident planes is calculated as the scalar product between the normal of the surface, and the direction of the particle motion. Thus, when the code is looking for the nearest surface to the particle, the logical value associated to the surface 2 (respect to the cell 1) is outside of this cell.

A continuation the analogue case, in transport, is described. The same logical value is calculated in the same code point.

The same geometry can be simulated, but, in this case, the particle has a collision inside the cell 2 (for that, this cell has to be filled with a material). The distance that the particle walks to the new position, and the deviation of its trajectory, are negligible. In this case, the logical fuction is calculated out of the subroutine *newcel*. In this case, logical value is calculated as the comparison between the signs of the surface equation in the particle localization and the sign of the surface in the cell description (see section 3.1.2). The logical value of the same surface 2, is inside in this case, while this same value was outside in the simulated case.

The consequence of this difference is that, another nearest surface is found, because these values are used in its search. In order to solve this kind of situations, two modifications are proposed.

The first of them is based on the example case presented. Just before to start the search of the nearest surface, the logical function is calculated again, without the special rules for coincident planes.

The other modification is based on the differences between the cases where the definition of coincident planes works and this particular case. In the other cases where the logic works, the name of the crossed surface is change to the coincident plane, that is used to define that content cell. The logic in the search understands that this crossed plane may not be the actually crossed plane where the particle is, despite this is a coincident plane (whose logical value is calculated with the special rules of the coincident planes), this plane is different to the crossed plane in its definition. The second modification also allows to change the name of the crossed plane for this type of coincident planes. Only this change is not realized if the coincident plane is behind the particle (according to the direction of the particle motion), and the scalar product between the normal of the surface and the direction of the particle motion is negative, because this plane can never be the crossed plane.

The both two modified versions were tested in several inputs. The results are almost the same for the two modified versions. Only one lost particle differs the results of the two version.

# 4. Calculation of volumes. Effect of lost particles in the results

The aim of this section is to check the modifications presented in the section 3. Furthermore, this section also shows that results of modified MCNP version are more accurate. Some results, that confirm this improvement, are presented with simple examples, in which the results are known, and the effect of the lost particles over the result is also known. In this section, the modified version contains both modifications presented in section 3.

The better accuracy of the modified version is due to those particles that are not lost with this modified version. When MCNP loses a particle, the contribution of this history is not summed to the tallies (a history is the simulation of this source particle, and these particles whose production have been caused, directly or indirectly, by this source particle). For the case of the lost particles studied in the section 3, this fact makes that the contributions of a specific type of particles are null (those that crossed a specific coincident plane in a determinate direction). Or, as it is presented in the section 3, the 'lost' particle maybe not lost, nevertheless, its simulated trajectory is incorrect. In this case, there are more trajectories than there would be, and these trajectories may contribute to the tallies.

The phase space is a space where are defined all possible trajectories with all possible energy at each point of the trajectory. With this definition, the previous paragraph can be summarized as: the lost particle effect is a bad sampling of the phase space. As the MCNP results are based on a good sampling of the phase space, the result of the modified version should be more accurate.

In order to achieve the aim of this section, the volumes of the two types of geometries are calculated with the modified and unmodified versions of MCNP. The results of both type of inputs help to check these modifications. However each type of geometry has a second aim because each type has a different complexity.

The first type of inputs are the simplest geometries that are shown as examples in the sections 3.2 and 3.3. As the phenomenology of the particle has been studied, the results helps to see the possible problematic associated to the lost particle events. These results are presented in the section 4.2.

The second type, is a typical example used in ITER works. Many results for ITER are been calculated with unmodified MCNP5. The volumes of this geometry are calculated in order to quantify the error introduced by the lost particles. Previous dose maps in ITER were obtained with inputs in which there are these types of lost particles, and the map results do not show any atypical value (out of the expected range), because of this, the expected differences between the results of the modified and unmodified versions should be small. These results are presented in the section 4.3.

The followings section (section 4.1) explains how MCNP calculates volumes based on track length tallies. The track length is the distance that is walked by the particle. For the track length contributes to the tally, this distance must be walked inside the tally region.

### 4.1. Calculation of volumes with MCNP

The flux in the cell is calculated in MCNP evaluating the track length. The definition of average flux, and the value calculated by MCNP is shown in the equation [9]

$$\bar{\phi} = \frac{1}{V}\int dE \int dV \int ds N\left(\vec{r}, E, s\right) \approx \frac{\sum W \cdot T}{V} = \bar{\phi}_{MCNP}(V) \tag{9}$$

where $N$ is the density of particles, and $ds$ is the differential of the track length, $W$ is the weight of the particle, we use $W = 1$ for the explanation of this work (as an analogue simulation), $T$ is the track length, and $V$ the volume [4], and $\bar{\phi}$ is the average flux in the cell.

As the equation 9 shows, MCNP need to know the volume of the region in order to be able to evaluate the flux. MCNP can calculate the analytic volume of some simple cells, but it cannot do this calculation for all cells. For that reason, MCNP allows to introduce the value of $V$ for the estimation of the flux given by the equation 9. This value is introduced by the user.

From the equation 9, we can write

$$\bar{\phi}_{MCNP}(V') = \frac{\sum W \cdot T}{V'} \qquad \rightarrow \qquad \bar{\phi}_{MCNP}(V' = 1) = \sum W \cdot T. \tag{10}$$

As MCNP allows to introduce the volume by the user, the flux in the cell with $V' = 1$ ($\bar{\phi}_{MCNP}(V' = 1)$) can be calculated with MCNP.

From the same equation 9 the volume can be expressed in terms of this flux

$$\bar{\phi} \approx \frac{\sum W \cdot T}{V} = \frac{bar\phi_{MCNP}(V' = 1)}{V} \qquad \rightarrow \qquad V \approx \frac{\bar{\phi}_{MCNP}(V' = 1)}{\bar{\phi}} \tag{11}$$

As the term $\bar{\phi}_{MCNP}(V' = 1)$ can be obtained by the simulation, in order to can evaluate the equation 11, the real average flux $\bar{\phi}$ must be known. This average flux is easy to evaluate for a region in which the flux is constant for every point of the region, because, as the equation [12] shows

$$p\bar{h}i = \frac{\int_V \phi\left(x, y, z\right) dV}{V} = \frac{\int_V dV}{V}\phi = \phi = const \tag{12}$$

The equation [12] is correct for the average of the whole volume where the flux is constant, as a smaller region inside of this region. Thus, if we know a constant flux over a extended region of the geometry, produce by a determinate source, we will be able to evaluate volumes of the cells inside of this region.

In this works, two source geometry, that produce constant known flux, are used to calculate the volume. One of them is a plane from where the particles are thrown in the direction of the normal of the plane (a square bigger than the dimension that we wish to measure), the second is a spherical source whose emission is isotropic inward-directed. Both sources, in void geometries, produce constant flux in a determined region. For the spherical source this region is a sphere of the same size than the source, and for the plane source, this region is all region in front of the plane, in the direction to the particles are thrown.

The reasoning for obtaining the analytic flux produced by these sources, in any point of the region where the flux is constant, it is explained below.

For the plane source. If these thrown particles are sampled with a uniform probability in the source plane, the number of particles that crosses a surface by specific point is the same for any irradiated point, hence, the flux is constant in all irradiated region. The number of particles that crosses a surface by a specific point per unit of surface and unit of time is, (this means, the flux is) $\phi(x,y,z) = n \cdot L^{-2}\ particles \cdot cm^{-2} \cdot s^{-1}$, where n is the number of thrown particles, and $L^2$ is the surface of emitting plane. Applying this result to the equation [11].

$$V_{plane} \approx \frac{\bar{\phi}_{MCNP}(V' = 1)}{\bar{\phi}} = \frac{\bar{\phi}_{MCNP}(V' = 1) \cdot L^2}{n} \tag{13}$$

The spherical source, whose emission is isotropic inward-directed, is the source suggested by MCNP [4]. This source also achieves a uniform flux in all points inside the sphere. This flux is proportional to the number of particles thrown by the source per unit of surface. However, each one of the $n$ thrown particles carry on a different trajectory, because of this, the distance that each particle walks between the source and a surface, (in this case, the sphere that is just in front of the source), is different.

The time interval that a particle, thrown over the sphere, takes in order to arrive to the spherical surface is

$$\bar{v} = \frac{\bar{r}}{\Delta T} \qquad \rightarrow \Delta T = \frac{\Delta x}{v_x \cdot \cos\theta} \tag{14}$$

where $\theta$ is the angle between the direction of the particle and the shortest distance between the source and the sphere. In the equation 14 the distance $\Delta x$ is constant (is the shorted distance between the point and the surface), regardless of the particle direction, because the sphere is close enough to the source.

Hence, as the flux over the sphere is the number of particles thrown over the surface, per unit of surface and unit of time, this flux is proportional to the angle between the direction of the particle and the shortest distance between the source and the sphere, $\theta$. If all possible direction are averaged, the average of this angle in a emission is isotropic inward-directed is

$$\int_{\Omega} \cos\theta d\Omega = \int_{0}^{2\pi} d\phi \int_{0}^{\pi/2} sin\theta cos\theta d\theta = \frac{1}{4} \tag{15}$$

Therefore, the flux over a sphere produced by a spherical source with the same radius $r$, whose emission is isotropic inward-directed, as the suggested source by MCNP [4], it is, the number of particles $n$ that crosses the surface per unit of surface and unit of time $\bar{\phi}_{sphere} = \left(\pi \cdot r^2\right)\ particles \cdot cm^{-2} \cdot s^{-1}$. According with this result and the equation 11, the volume of a region inside the sphere is

$$V = \frac{\pi r^2 \cdot \bar{\phi}_{MCNP}(V' = 1)}{n}. \tag{16}$$

These results of the volume, for both source, are presented dependent on the source intensity (number of particles thrown by the source in the equations [13] and [16]). In order to obtain the same flux as MCNP, this parameter must be 1 particle, because MCNP returns its results normalized to 1 particle thrown by source. Thus, the calculated volumes are

$$V_{plane} = \bar{\phi}_{MCNP}(V' = 1) \cdot L^2 \tag{17}$$

$$V_{sphere} = \pi r^2 \cdot \bar{\phi}_{MCNP}(V' = 1) \tag{18}$$

for the plane and spherical source respectively.

### 4.2. Simple Case

The volume of the geometries proposed as examples of the secion 3.2 and 3.3, were calculated with both sources described in the section 4.1, and with modified and unmodified MCNP5 versions (see section 3). The results of both geometries are presented, and analyzed in this section. This analysis allows showing the common features of the particles for this error happens, just like the associated possible error for the tally (if the particle is not lost). Finally, common conclusions and problems are presented.

The first case presented is the geometry example in lost particle events of type 1. The geometry is a cube that is cut in two volumes by the same plane used in the definition of the cube (because it is used in a universe cell that is translated, see section 3.2). The figure 6 shows the scheme of this geometry. The dimension of the box is $20cm \times 20cm \times 20cm$. Two versions of this geometry is simulated, dependent on the position of the plane that cuts the box. In one of them, the plane cuts the box by the middle, in the other case, the box is cut in a quarter and three quarters. The whole box is called the *entire* cell. The box, that has a quarter of the whole volume, is called the small cell, and the other, the big cell. The small cell is defined between the two planes with the same name. In the figure 6, this cell is the lower cell.

As it was previously named, two types of sources were used for the measurements of the volumes. First of them is a spherical source, bigger than the box. The second is a plane parallel to the surface that cutting the box. This source emits forward the moved plane of the box (in the figure 6, the source plane is below of the box, and it emits upwards). In order to achieve a better statistic result, the surface of this source plane is the same than the surface of the box face in front of the source.

The data is presented in table 3. It is remarkable, in these data, the differences between the results of the different MCNP versions. These differences are commented below.

The data obtained with modified and unmodified version is presented in the table 3. All result obtained with the modified version is in agreement with the real volume (within the uncertainty range). The unmodified version shows different results.

Particles of unmodified code do not see the plane that cut the box. When a particle crosses the surface (the logical fail of the lost particle type 1 happens, see section 3.2), the particle enters to the small cell, and its next displacement is to exit of the *entire* cell. This is like whether there would not be a surface cutting the box. It is mean that, the track length associate to the flux of the small box is full box length, and as the surface that the particle crosses is the final box (instead of the plane that divide the box), this particle never crosses through of the bigger box (although the position xyz of the particle was inside this box).

For this reason, the measured volume for the small cell is the whole volume of the box, and it is exactly the same in the two types of this geometry (the box cutting by the middle, and the cutting by a quarter), because all trajectories are exactly the same. In relation with the volume of the big cell, any track length is associated to this cell, because any particle never enters here, thus, the result of the big cell volume is zero.

In the spherical case, when the particle enters by a face that is not the moved plane, the transport is well done, in particular, the transport of all particle that enters in the cell by the bigger box is well done, despite these particles cross the moved plane after, because, as it is said in the section 3.1.3, if the plane is in the highest level, this plane is considered as one of the possibles nearest surfaces, and in this case, this plane is chosen as the nearest plane if it is actually. The contribution of this particles is well calculated.

Only when the particle enter in the box across of the moved surface, and has to cross it again, the error is produced, because MCNP logic does not allow that the particle crosses again the plane that it has just crossed. This error in the transport causes a overestimation of the flux in the smaller cell (because the track length is bigger), and an underestimation in the bigger (because these 'lost' particles track lengths do not contribute to the bigger cell). This was verified using a plane source from the opposite side of the cube. The obtained volumes was correct, because all trajectories are the same as the trajectories described in the previously paragraph.

| **Case** | ***Entire* Cell** | | **Small cell** | | **Big Cell** | |
|---|---|---|---|---|---|---|
| Analytic $50-50$ | 8000 | | 4000 | | | 4000 |
| Analytic $25-75$ | 8000 | | 2000 | | | 6000 |
| Plane $50-50$ | 8005.74 | 0.0007 | 8005.74 | 0.0007 | 0.0 | 0.0 |
| Plane $25-75$ | 8005.74 | 0.0007 | 8005.74 | 0.0007 | 0.0 | 0.0 |
| Modified Plane $50-50$ | 8005.74 | 0.0007 | 4002.87 | 0.0007 | 4002.87 | 0.0007 |
| Modified Plane $25-75$ | 8005.74 | 0.0007 | 2001.42 | 0.007 | 6004.3 | 0.0007 |
| Sphere $50-50$ | 7979.10 | 0.0068 | 4344.71 | 0.0085 | 3634.39 | 0.0089 |
| Sphere $25-75$ | 7979.10 | 0.0068 | 2702.1 | 0.0104 | 5277.0 | 0.0080 |
| Modified Sphere $50-50$ | 7979.10 | 0.0068 | 3997.75 | 0.0084 | 3981.35 | 0.0084 |
| Modified Sphere $25-75$ | 7979.10 | 0.0068 | 2015.61 | 0.0100 | 5963.49 | 0.0075 |

Table 3: Volume of the geometry of the figure 6, calculated with a plane or spherical source, with the modified and nor modified version of MCNP. The uncertainty of the results are relatives. The units of the measure are the MCNP units ($cm^3$).

The next case presents the geometry example of the lost particle events of type 2. The geometry of the input is a box cut by a cylinder, and a coincident plane. Two different types of this geometry are simulated. The dimensions of the whole box are $20cm \times 20cm \times 20cm$ or $20cm \times 20cm \times 10cm$ (the changed dimension is in the same direction as the particle is moving in the figure 7). The radius of the cylinder, that cuts the cell, is 5 $cm$. In this case, there are three defined cells, the *entire* cell (the whole box), the cut cell, and the cylinder cell.

As it was previously named, two types of sources were used for the measurements of the volumes. These are a spherical source, bigger than the box, and a plane parallel, with the same size that, the the surface cutting the box, and the source emits forward the coincident planes, (in the figure 7, the source plane is below of the box, and it emits upwards). The data is presented in table 4. It is remarkable, of this data, the differences between the results uncertainty of both type of source, and the differences between the results of the different MCNP versions. These differences are commented below.

The data obtained with modified and unmodified version is presented in the table 3. All result obtained with the modified version is in agreement with the real volume (within the uncertainty range). The unmodified version shows different results.

For the particles of the unmodified version, the transport is well done when the particles do not enter to the *entire* box though the cylinder cell (this is, crossing the coincident plane in the region of the plane inside the cylinder). The 'lost' particle event only happens when the particle enters by this way. For a plane source in this case, as it is detailed in the Annex A, the particles do not see the cylinder. The particles enters into the cylinder cell, and its next position is out of the box, thus, the associated track length for these particles is the length of the box. For that reason, the estimation of each volume is the same (within the statistical uncertainty), because the area of the plane that is cut inside the cylinder, it is the same as the rest of the plane area, and the particles are sampled uniformly in this area, the same number of particles enter by each surface. AS the same number of particles contribute to the flux of each cell with the same track length, the calculated volume is the same.

In the case of the spherical source, the track length for the particles that enters to the box across the cylinder cell, depends of the angle and the position where the particle crosses the coincident plane. The overestimation of these track lengths is also rested of the cut box volume. The rest of trajectories are well simulated.

| **Case** | ***entire* Cell** | | **Cut Box** | | **Cylinder** | |
|---|---|---|---|---|---|---|
| Analytic short | 4000 | | 3214.6018 | | 785.3982 | |
| Analytic long | 8000 | | 7214.6018 | | 785.3982 | |
| Plane short | 3999.36 | 0.0002 | 1999.78 | 0.0005 | 1999.58 | 0.0005 |
| Plane long | 7988.72 | 0.0002 | 3999.56 | 0.0005 | 3999.16 | 0.0005 |
| Modified plane short | 3999.36 | 0.0002 | 3214.08 | 0.0005 | 785.282 | 0.0005 |
| Modified plane long | 7998.72 | 0.0002 | 7213.43 | 0.0005 | 785.282 | 0.0005 |
| Sphere short | 3994.03 | 0.0027 | 2912.46 | 0.0029 | 1081.56 | 0.0044 |
| Sphere long | 7966.35 | 0.0022 | 6704.86 | 0.0023 | 126.15 | 0.0047 |
| Modified sphere short | 3994.03 | 0.0027 | 3209.48 | 0.0027 | 784.55 | 0.0044 |
| Modified sphere long | 7966.35 | 0.0022 | 7181.8 | 0.0022 | 784.55 | 0.0044 |

Table 4: Volumes of the geometry of the figure 7, calculated with a plane or spherical source, with the modified and unmodified version of MCNP. The uncertainty of the results are relatives. The units of the measure are the MCNP units ($cm^3$).

The final part of this section is a comment about general features observed in these lost particle events.

The first, these lost particles events are dependent on the case and the source. This means, that the quantitative uncertainty introduced by these particles depends on the geometry (for example in the first case, depend on the position of the plan that cuts the box, and in the second example, depends on the full length of the box). It also depends on the source, because it depends on the angle and point with which the particle enters.

The second common feature of both type of lost particle events, is that the particle may not be lost, just like in the cases of this section. When MCNP loses a particle, it reports a message where the information about this lost particle is presented. In the cases run for this section, any message about lost particles was reported. However, as the data shows, the transport of these 'lost' particles is incorrect. In other geometries, the code may not find a nearest surface, or a cell behind this surface may not exist, they are these cases where the particles maybe actually lost.

The last common feature is that the particle can be transport a while outside of the cell, to which the code has assigned the particle for this position in its transport, because the code may overestimate the distance to the nearest surface. For this feature happens, the cell has to be filled with a material.

For example, in the case of the lost particle type 1, in the figure 6, with a plane source. When the particle enters to the box, the nearest surface is a distance of the whole length of the box ($20cm$). If the cell has material, and the distance to the next collision is less than this $20cm$, the particle does not exit the small box, but, the particle has a collision, even if this collision is further than the surface that cuts the box. As the particle does not cross a surface, the code does not search the cell at this new point. The next track lengths of the particle are associated to the small box, until the particle crosses a surface (that defines the small box or a container cell of this box).

## 4.3. Volumes in Clite

In this section, the volumes of a typical geometry in ITER (Clite), were calculated in a determined region, where these types of lost particles, described in the section 3, were observed. These volumes were calculated, with modified and unmodified version of MCNP5 presented in the sections 3.2 and 3.3.

This section is divided in several parts. The first part of this section presents the different types of measurements for calculating volumes in this region. The advantages and issues of these methods are also described. All these are presented in the section 4.3.1.

After, the reasons for the choice of the source, and the region where the volumes were measured, it is argued in the section 4.3.2. The last part, is the presentation and the discussion of the volume results of the Clite region. This is presented in the section 4.3.3.

### 4.3.1. The measurement method. Advantage and issues.

Relating to the measurement of the volumes, two different measurements are realized with MCNP tallies based on track length. Another measurement of the volume is realized with MCNP, using a statistic method, and the last measure is realized with *Space Claim* [15].

*Space Claim* is a useful computational tool for building complex geometries that, among its uses, it is able to calculate the volume of the cells. The volume calculated by *Space Claim* is the volume of the whole cell. In this work, we assumed that this calculation has not associated uncertainty.

The MCNP tallies based on track length are, cell tally (as the tallies used in the previously sections), and *Fmesh* tally with *multiflux* option, this *Fmesh* tally is explained below.

These tallies with this specific option of *multiflux* is a special feature of meshtally implemented of R2SUNED[16]. This tool can calculate only the volume of the cell region where the particle is transported, if a part of a cell is cut by a container cell and the particle never enters in this part, the volume of this region is not calculated.

*Fmesh* tally is a tally based on track length, where a mesh is superimposed on a region of the geometry (we chose a mesh whose mesh elements were box). The average flux over each mesh element is calculated as if the mesh element was a cell (with the estimation of the equation [11]). Furthermore, if the *multiflux* option is chosen, the flux is calculated for the region of the cell inside the mesh element. The equation used to estimate this flux is the equation [11], this equation needs the volume of the region of the cell inside the mesh element. A statistical method is used in order to calculate this volume. This statistical method is the last method used for the volume calculation in this section, and it is described right after.

As summarize, the *Fmesh* results, with the *multiflux* option activated, are the flux in each cell inside each mesh element (calculated with the track length equation), and the statistic volume of these cells (used for the calculation of these fluxes).

This statistical method consists of a uniform sampling of points into the mesh element (although the methods based on track length are also statistical methods, we refers this one as the statistical method). Each point belongs only to one cell, the relative volume, of the region of the cell inside the mesh element, it is the ratio of the number of point that belongs to this cell, over the total number of point sampling in this mesh element. The statistical uncertainty is calculated as

$$\sigma_s^2 = \frac{\bar{x^2} - \bar{x}^2}{N} = \frac{\bar{x} - \bar{x}^2}{N} \tag{19}$$

where $\bar{x}$ is the fraction of volume of this cell with respect to the mesh element volume, $N$ is the number of sampling points, and $\sigma_s$ is the statistical uncertainty. This measure has a resolution uncertainty, because, if the points are uniformly distributed, one point measure a volume of $V_{me}/N$ in MCNP units $cm^3$, where $V_{me}$ is the volume of the mesh element. The equation 20 expresses the average volume and the statistical uncertainty associated to this volume, of this statistic method.

$$\bar{V} = \frac{N_{in}}{N} \cdot V_{me} = \bar{x} \cdot V_{me} \qquad\qquad \sigma_V^2 = \left(\frac{1}{N^2} + \frac{\bar{x} - \bar{x}^2}{N}\right) \cdot V_{me}^2. \tag{20}$$

The total volume of the cell inside the mesh of the *Fmesh* tally is obtained summing, the volume of cell inside the each mesh element. The total volume of the $i^{th}$ cell is

$$V_i = \sum_{j}^{Nme} V_{ij} \tag{21}$$

where $Nme$ is the number of mesh elements, and $V_{ij}$ is the estimation of the volume of the $i^{th}$ cell inside the $j^{th}$ mesh element.

It is important to recall that the use of MCNP flux and volume in this work is equivalent due to the equation 11, for this reason the sum of flux (this means the sum of the volume) in different points (regions) makes sense.

The previously description shows the different types of volume measurement, the following argument explains that measurements are compared, and the advantages and issues of these measurements.

- **Cell Flux and Space Claim volumes.**

These both methods measures the whole volume of the cell, hence, the results of these methods can be comparable between them. The advantages of these method are:

- The volume is estimated in a bigger region than the measurement based on *Fmesh* tally (see issues of the *Fmesh* methods). It is more probable that the 'lost' particles events of other regions, can also contribute for this tally.
- The statistical uncertainty is the correct one associated to this measure (see issues of the *Fmesh* methods).

The issues of these method are:

- The volume calculated by both method may be different, because the cell tally does not calculate the volume of the cell region outside of the container cell, while the *space claim* method calculates the whole cell volume (even this cell is cut by a container cell). Although geometries, where most of its cell are cut, are unusual, at least in ITER works.
- A previous calculation is needed in order to know the cells that are defined in the interest region (where the lost particle events happen). For obtaining this data, we used the results of *Fmesh* tally. The reasons, for the chose the interest region, are presented in the section 4.3.2.

The other two comparable methods are:

- ***Fmesh* Flux and Statistic volumes. *Fmesh* methods.**

Both methods measures the volume of the cell inside the mesh of the *Fmesh* tally, as the sum of the volume of the cell in each mesh element. The advantages of these method are:

- It is sure that both methods measure the same volume. The volume of region of cell that is cut by a container cell, it is not measured for any of this methods (the region where the particle is never transported).
- To know the cells inside the region of interest before of the simulation, it is not necessary, because MCNP finds them while it is calculating the statistical volume.

The issues of these method are:

- The statistic error of *Fmesh* flux is underestimated, because the correlation, that the track length causes in the measurement of the cell flux inside each mesh element, cannot be taken into account in the sum of the volumes of each mesh element (the correlation between the measurement of the volume of the same cell inside two different mesh element).
- The region where the volumes are calculated, is smaller than the whole cell. This size is chosen as small because this way the heterogeneity of the finite sampled is reduced (because the exactly number of points is sampled in each mesh element).

### 4.3.2. The region where the volumes are calculated

A specific region where the volumes were calculated, was chosen. The reason and process, why this region was chosen, are explained in this section.

First step, we used *MCAM*,[14] in order to print the lost particles events detailed in the MCNP output.

*MCAM* is another useful tool for building MCNP geometries, that among its uses, it is able to read a MCNP output and prints the lost particles positions and directions, when the lost particle events happened (this information is contained in the output of MCNP).

This image, obtained from *MCAM*, allowed us determining a small region where several particles events happened. A lot of these lost particles events were of the types described in the section 3. Then, a mesh, located in the region where these lost particles events happened, was defined with a size of 160 $dm^3$, and $16 \cdot 10^4$ mesh elements. The mesh element size is 1 $cm^3$.

The plane, where the fail of these lost particles events happened, is near of this region. The spherical source, centered in this region, was used for the calculation of the volumes.

This sphere was big enough that a great part of this sphere emits particles that cross the surface that produces this logical fail of the lost particle events. Furthermore, its size also allows that other lost particles that crossed a different plane (that also produces logical fail of these lost particles, but in other point) may also contribute to the tally results.

After the cells of this region were obtained, as result of the *Fmesh* tally, a figure of the geometry was printed with *space claim*. Only one cell is inside the region defined by the mesh of the *Fmesh* tally. This region is painted in green in the figure 8, where the geometry of the studied region is shown. The mesh of the mesh tally is over 340 times bigger than this cell. This size is still smaller than the volume of the whole geometry shown in the figure 8. The volume of all cells is 1637 $dm^3$, this is more than 10 times bigger than the mesh of *Fmesh*.

As the cell is whole inside the mesh of the *Fmesh*, all track length that contributes to the *Fmesh* tally, also contributes to the cell tally. Hence, this magnitude is comparable.

The difference between both modified and unmodified version are also comparable between the results of the cell and *Fmesh* tally. If all lost particles only go across this geometry though the mesh, the differences would be the same.

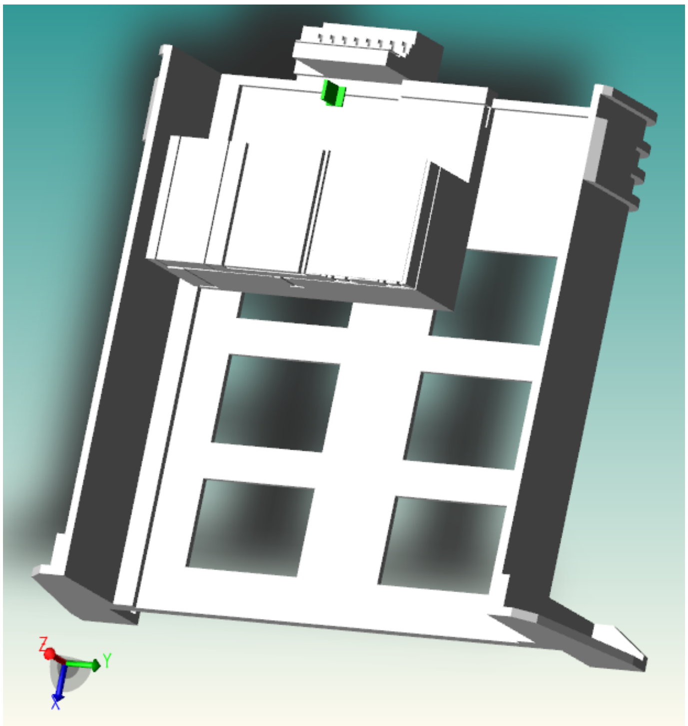


Figure 8: Geometry of all cells inside the interest region. The green cell is the only cell fully contained in the mesh of *Fmesh* tally. Image obtain with *space claim*. Some cells are not printed in order to show the green cell.

### 4.3.3. Results and conclusions

The measured volumes with the method presented in the section 4.3.1, and both modified and unmodified MCNP5 versions are calculated in this section. The comparisons are also described in the sections 4.3.1 and 4.3.2.

Before presenting the data and the comparisons, a brief description about the number of lost particles of each version, and the consequences of these lost particles in the data is done below. In the final part of this section, a brief conclusion of the results of this section is commented.

When these results were obtained, the unmodified MCNP lost 200 particles, while the modified version only lost 16 particles. In both cases, $10^8$ histories were simulated. The differences between the results of modified and unmodified version of MCNP, in both *Fmesh* and cell tally, are due to the particles that in one version are lost, and in the other are well simulated. This is because, the sequence of random number for particles that are not lost is the same, despite that when the lost particle is completely simulated, MCNP uses more random number. This is because MCNP reserves a range of random number for each history, independently of the number of random number used in other histories.

The first data presented of this section, is the volumes of the whole cell. The table 5 shows this data obtained with cell tallies and *space claim*. The results obtained with cell tallies are consistent with the results of *space claim*, this marks that any regions of the calculated cell is cut by a container cell, or these regions are really small (within the uncertainty range).

The differences in the data of the table 5, between modified and unmodified versions, are only observed in 5 cells, in all of them, the value is more accurate in the new version. However these differences are inside the statistic uncertainty range. As the ratio between the lost particles and the total number of simulated ones is constant, these differences would be kept of the same order when more particles are simulated (this means, the uncertainty introduced by the lost particles is systematic uncertainty), while, the statistic uncertainty is reduced as $nps^{-0.5}$, where $nps$ is the number of simulated histories. In order to see clearly if the value of this simulation is more exactly, a continue run will must be done until $10^{11}$ histories.

The last data presented of this work, is the volumes of the cell region inside the mesh of the *Fmesh* tally. The table 6 shows this data, obtained with the *Fmesh* tally, with the flux and the statistical method. In this case, the differences between both data are bigger than the uncertainty range, because the uncertainty of the flux data is underestimated (see section 4.3.1).

As the measurement methods are verified (as proof we can compare the value of the cell 14 with the data of the table 5 of the same cell, that is the cell that is whole inside the mesh), the real statistical uncertainty of the flux data of the table 6, is of the order of the differences between both methods. These differences are bigger than the differences

| Cells | Space Claim | Flux Original Version | | Flux Modified Version | |
|---|---|---|---|---|---|
| **X** 1 | 8708.539 | 8681.91 | 0.0066 | 8682.31 | 0.0066 |
| 2 | 3377.816 | 3410.47 | 0.0088 | 3410.47 | 0.0088 |
| 3 | 22970.261 | 23148.5 | 0.0047 | 23148.5 | 0.0047 |
| **X** 4 | 8694.9433 | 8677.92 | 0.0066 | 8678.32 | 0.0066 |
| 5 | 3377.816 | 3344.46 | 0.0089 | 3344.46 | 0.0089 |
| **X** 6 | 8675.616 | 8633.36 | 0.0066 | 8633.77 | 0.0066 |
| 7 | 3377.816 | 3384.89 | 0.0089 | 3384.89 | 0.0089 |
| 8 | 160333.799 | 160340 | 0.0025 | 160340 | 0.0025 |
| **X** 9 | 3264.88 | 3256.15 | 0.0039 | 3256.26 | 0.0039 |
| 10 | 115954.848 | 115856 | 0.0027 | 115856 | 0.0027 |
| **X** 11 | 643531.392 | 642576 | 0.0009 | 642577 | 0.0009 |
| 12 | 8086.4 | 8066.84 | 0.0034 | 8066.84 | 0.0034 |
| 13 | 3367.5599 | 3369.64 | 0.0054 | 3369.64 | 0.0054 |
| 14 | 478.3865 | 471.882 | 0.0150 | 471.882 | 0.0150 |
| 15 | 612.450099 | 625.584 | 0.0106 | 625.584 | 0.0106 |
| 16 | 4367.9997 | 4381.84 | 0.0057 | 4381.84 | 0.0057 |
| 17 | 612.4001 | 620.415 | 0.0107 | 620.415 | 0.0107 |
| 18 | 418189.875 | 419079 | 0.0018 | 419079 | 0.0018 |
| 19 | 195984.873 | 195698 | 0.0023 | 195698 | 0.0023 |

Table 5: Estimation of the volumes with cell flux data. The error are relative. X mark points to differences between modified and no modified versions. To obtain flux data $10^8$ histories was simulated. The units of the measure are the MCNP units ($cm^3$).

between both version of MCNP, this means that the uncertainty of the lost particles is also lower than the statistical uncertainly, in the same order as the previous data presented. For the same reason that in the previously argument, in order to see clearly if the value of this simulation is more exactly, a continue run should be done until $10^{11}$ histories.

The last comparison is between the data obtained from MCNP flux (with tallies based on track length), of the tables 5 and 6.

The first comparable value is the cell 14, as this cell is whole inside the mesh of the *Fmesh*, exactly the same particles trajectories contribute to both tallies, for this reason, the measured flux in both cases is exactly the same, however the uncertainties of these measurements are different. This comparison evidences one of the issues of the *Fmesh* measured flux, pointed in the section 4.3.1. As the volume of this cell is calculated as the sum over all mesh elements, of the volume of this cell contained inside this mesh element, for the uncertainty calculation is needed to know the correlations between the volume of the cell inside the each two different mesh elements. The measurements are related through of the track length of the each particular history.

The last comparisons are about the differences between the both tables 5 and 6. The differences between the modified and unmodified versions in the table 6 should also be

| **Cells** | **Stadistic vol** | | **Flux Original Version** | | **Flux Modified Version** | |
|---|---|---|---|---|---|---|
| 1 | 726.507 | 0.00010 | 703.45372 | 0.00511 | 703.45372 | 0.00511 |
| 2 | 609.384 | 0.00011 | 611.43882 | 0.00549 | 611.43882 | 0.00549 |
| 3 | 886.0005 | 0.00009 | 877.96804 | 0.00456 | 877.96804 | 0.00456 |
| **X** 4 | 2252.2240 | 0.00005 | 2255.17009 | 0.00286 | 2255.37228 | 0.00286 |
| 5 | 1889.0466 | 0.00005 | 1864.89186 | 0.00315 | 1864.89186 | 0.00315 |
| **X** 6 | 1671.1107 | 0.00005 | 1685.99103 | 0.00331 | 1686.29434 | 0.00331 |
| 7 | 1401.6940 | 0.00005 | 1422.2588 | 0.00361 | 1422.25288 | 0.00361 |
| **X** 8 | 80069.8522 | 0.00001 | 80327.77248 | 0.00048 | 80328.08427 | 0.00048 |
| **X** 9 | 306.7336 | 0.00043 | 300.86269 | 0.00686 | 300.87323 | 0.00686 |
| 10 | 731.1396 | 0.00027 | 730.12631 | 0.00443 | 730.12631 | 0.00443 |
| 11 | 831.9959 | 0.00008 | 844.23803 | 0.00467 | 844.23803 | 0.00467 |
| 12 | 1330.0815 | 0.00007 | 1319.32226 | 0.00375 | 1319.32226 | 0.00375 |
| 13 | 633.0918 | 0.00018 | 624.74539 | 0.00523 | 624.74539 | 0.00523 |
| 14 | 478.2209 | 0.00022 | 471.88238 | 0.00601 | 471.88238 | 0.00601 |
| 15 | 432.0252 | 0.00026 | 439.23001 | 0.00610 | 439.23001 | 0.00610 |
| 16 | 2310.2762 | 0.00010 | 2321.74554 | 0.00270 | 2321.74554 | 0.00270 |
| 17 | 432.09560 | 0.00027 | 443.27048 | 0.00607 | 443.27048 | 0.00607 |
| 18 | 62036.6298 | 0.00001 | 62300.43585 | 0.00055 | 62300.43585 | 0.00055 |
| 19 | 807.8577 | 0.00025 | 801.07026 | 0.00423 | 801.07026 | 0.00423 |

Table 6: Estimation of the volumes with Fmesh data.The error are relative. X mark points to differences between modified and no modified versions. To sample the point in the box, $10^5$ point was used by box, and to obtain flux data $10^8$ histories was simulated. The units of the measure are the MCNP units ($cm^3$).

presented in the table 5, because if the particle go across a region of the cell inside of the mesh, this particle also contributes to the whole cell tally.

On one hand, these lost particles contribute in the differences of the cells 1, 4, 6, 9 and 11 for the measurements of the whole volume of the cell (data in the table 5). On the other hand, the lost particles contribute for the measured differences of the volumes of the cells 4, 6, 8 and 9 (data in the table 6).

The differences between both versions, in all coincident cell (4,6 and 9) are bigger in the whole volume, this suggests that the track length of the trajectories of these 'lost' particles is bigger than the dimension of the mesh, hence its contributions are bigger for the whole volume.

The cells 1 and 11, only show differences in the case of the whole volume of the cell, it implies that the trajectories of these lost particles never enter in the region defined by the mesh. As the mesh was fitted for the region where the lost particles events were observed, this result suggests that 'lost' particles of other region also contributes to these tallies. This phenomenology may also cause the differences in the cells 4, 6 and 9.

The last case is the cell 8. In this case, the differences are only observed in the data of the *Fmesh* tally. This is due to this difference is lower than the last representative digit that MCNP returns as result when the cell flux was measured.

Before presenting a conclusion about this work, we recall that the use of MCNP flux and volume in this work is equivalent due to the equation 11, for this reason the sum of flux (this means the sum of the volume) in different points (regions) makes sense. As final conclusion of this section, the error introduced by the lost particles trajectories, this is, the difference between the data of the modified and unmodified versions, lower than the statistical uncertainly. The order of the statistic uncertainty is in the range of the statistic uncertainty of data usually presented in ITER works. Besides, the lost particles ratio, found in the unmodified version, is relatively high. For these reasons, it is highly probable that the uncertainty of the data calculated with original version of MCNP in other works of ITER, is also bigger than the uncertainty of these lost particles events.

# 5. Conclusion and future work

In this work a study on the storage of the geometry in the RAM memory, and the particle transport in MCNP5, have been realized.

It was found that the memory reserved for the geometry storage was overestimated, resulting in a waste of a large amount of memory in RAM.

The problem of the total memory reserved for the geometry storage during the simulation had been solved. The storage of the cell definition is pointed as the cause of the problem.

A modification for solving the problem is proposed and implemented in MCNP. This modification cuts down the memory reserved for the cell definition storage. The results of this modified version are good enough to allow a better exploitation of computational resources.

The second part of the work was dedicated to the transport simulation, and particularly to the study of lost particles events. It was observed that there were lost particles during the simulation of the transport, even if the geometry was correctly defined. Two kinds of events producing lost particles were identified and solved.

Some calculations have been realized in order to quantity the effect of these lost particles, the results points that it is probable that previews calculations in ITER inputs with these lost particles events are correct, because the statistic uncertainty is bigger than the error introduced by these lost particles events. However, as this problem is case dependent, this result does not assure that all data obtained previously are correct. A more exhaustive evaluation of the obtained data will be done in order to check the assumptions realized in function of this data.

Despite the solution presented about lost particles, there are other situations that MCNP does not consider well. This situations will be located and solved in order to obtain a more reliable tool.

# 6. Acknowledgment

This work has carried out using an adaptation of C-lite MCNP model which was developed as a collaborative effort between the FDS team of ASIPP china, university of Wisisconsin-Madison, ENEA Frascati, CCFE UK, JAEA NAKA and ITER organitaion. It has been also supported by the Spanish MINECO (Ministerio de Economía y Competitividad) and by FSE (Fondo Social Europeo) under Programa Estatal I+D+I-Retos, Proyecto BES-2016-078586.

# Annex A. Transport details

In this annex, an example of transport of an individual particle is described step by step. For this reason, the section 3.1 should be read before, because this section explains the particle transport. This example also shows a lost particle event of type 2, described in the section 3.3.

As there are more than a way to describe a same geometry, first the particular input file is written, after a brief description of the geometry and the concrete aim of this annex is presented.

C INPUT FILE:

C Cells
1 0 10 6 u=1 imp:n 1 imp:p 1
2 0 -10:-6 u=1 imp:n 1 imp:p 1
3 0 1 -2 3 -4 5 -7 fill 1 imp:n 1 imp:p 1
4 0 -100 #3 imp:n 1 imp:p 1
99 0 100 imp:n 0 imp:p 0

C Surfaces
1 px -10
2 px 10
3 py -10
4 py 10
5 pz 0
6 pz 0.00001
7 pz 10
10 cx 5
100 rpp -100 100 -100 100 -100 100

C Source
nps 1
sdef pos 0 0 -10 vec 0 0 1 dir 1 par 1 erg 14

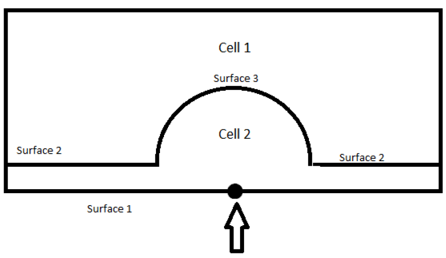


Figure 9: Scheme of geometry where this particles of type 2 can be lost.

A cut of a plane YZ of the region of this input geometry, is shown in the figure 9. This region is formed by the cells 1, 2 an 3 of the input, and it is in this region where the particle transport is detailed.

The cell 3 is a box of 20cm ×20cm ×10cm. This box is a container cell that contains the universe 1 (this means that, this cell has a *fill* 1 card). The cells belonging the universe 1, are the cells 1 and 2. The cell 1 is all region over the plane $pz = 0.00001$ *and*, besides out of the cylinder with radius 5cm. The cell 2 is the rest of the universe, this is, all regions below the plane $pz = 0.00001$ *or*, the region inside the cylinder. As this universe is cut by its container cell (cell 3), the cell 3 is the whole box of the figure 9, the cell 1 is the superior cell inside the box, and the cell 2 is the inferior cell inside the box.

As the planes 1 and 2 are close enough to be coincident for most of the particle trajectories (the distance between two planes, in transport, is measured as the distance that the particle must walk between them, and this distance depends on the trajectory, see section 2.1), the geometry that particle would see, is a cell 2 as the region inside the cylinder into the cell 3, and the cell 1 as the rest of the cell 3 outside of the cylinder.

Before the detailed description of the transport, the general steps, that the particle realizes, are briefly described.

As the material of all content cell is the vacuum, the particle never collides. For this input the only finite distance is the nearest surface, therefore, the particle goes from a surface to other surface, crossing from one cell to other. The most important steps realized by the code, for the particle transport, after the particle was defined, are a repetition of the next scheme: the code searches the nearest surface (*track* subroutine, see section3.1.3), the particle is moved, and a new cell behind this surface is searched (*newcel* subroutine, see section3.1.1). These steps are realized again and again, until the parameters of the particle do not allow continuing the particle simulation, in this input, it happens when the particle enters in a cell with neutron importance zero (cell 99).

In this annex only one repetition of these steps is described (since the particle found the cell 1). Below, this description is presented for unmodified MCNP5 versions, after, the modifications presented in the section 3.3 are also commented.

The source definition (sdef) indicates that the neutron particle is defined in the point (0,0,-10) (cell 4), and its direction of motion is the direction (0,0,1). In the figure 9, the particle is defined under the box, and its motion is upward. The nearest surface found is the plane 5 (surface 1 in the figure 9). The particle is moved to the position (0,0,0), and the parameter that marks the crossed surface, is updated. This parameter is $jsu = 5$

From here, the next steps are: found the new cell where the particle goes, and found the nearest surface where the particle should be moved. These steps are detailed right after.

The first step is the subroutine *newcel* (this subroutine is described in section 3.1.3). The first task in this subroutine, of the three tasks defined in that section, is to find the cell behind this surface. For this task, all cells in the same universe, that the cell from where the particle comes, and that in its definition appears the crossed plane, these cells are checks as possible cells behind the surface.

The cell where the particle comes from, is the cell 4. Its universe is the universe 0 (because there is not a $u$ card in its description). The crossed plane is pointed by the parameter $jsu$, this is the plane 5. The only cell that satisfies these conditions is the cell 3. In order to verify if this is the cell behind the surface 5, the point where the particle is (over the plane) must belong to this cell. For verifying this, the code uses the subroutine *chkcel* (see section 3.1.2).

This subroutine builds the logical function (see section 3.1.2), and returns the evaluation of this function, that is the response of whether the point where the particle is localized, is inside the cell.

In order to build this function, the cell description is copied, changing the surfaces by its logical value (true if the point is inside the surface). This logical value is obtained, comparing the equation, that defined the surface, evaluated at the particle localization, with the sign of this element of the cell description.

In two types of situations, this logical value is obtained by a different way. For the crossed plane, and for coincident planes in *newcel*, but only when the code is looking for the cell of the highest level inside a container cell.

In these situations, the logical value is calculated as the sign of scalar product between the normal of the surface (in the direction of the defined region) and the direction of the particle motion.

The cell 3 is defined as the intersection of 6 planes, the definition is shown in the input file (this input is presented in the beginning of the Annex A), one of these planes is the crossed plane. For the rest of them, the plane equation is evaluated, and the result is compared with the sign in the cell definition. The equation of the planes defined in the position $-10$,(of each coordinate), evaluated in the point (0,0,0) is positive (equation 22), as the sign of the definition is also negative, its logical values are true (inside).

$$r - Pr = 0 - (-10) = 10 > 0 \tag{22}$$

The equation of the planes in the position 10 (of each coordinate), evaluated in the point (0,0,0) is negative (equation 23), as the sign of the definition is also negative, its logical values are true (inside).

$$r - Pr = 0 - 10 = -10 < 0 \tag{23}$$

The last one is the logical value of the crossed plane (surface 5). The scalar product between the normal of the surface (upward) and the direction of the particle motion (upward) is positive, the logical value is also true (the point is inside the region defined by this plane). The logical function and its evaluation is written in the equation [24]. In the logic equations of this section, the sign . represents the *and* logic operator (this is the intersection operator), and the sign : represented the *or* logic operator, (this is the union operator).

$$1.1.1.1.1.1 = 1 \tag{24}$$

As the results is true, the point, where the particle is located, is inside this cell. This means that the cell behind the surface 5, is the cell 3. Now, as the cell 3 is a container cell (there is a *fill* card in its definition), the code searches the highest level cell inside this cell.

The cells, that are checked as possible cells inside cell 3, must belong to the universe that fills the cell 3, this is the universe 1. The possible cells, that satisfy this condition, are cells 1 and 2. The definition of these cells are shown in the input file (this input is presented in the beginning of the Annex A). The check is realized through the evaluation of the logical function built in *chkcel*.

Both cells are only defined with the surfaces 10 (cylinder) and 6 (plane). The evaluation of the point (0,0,0), where the particle is located, in the cylinder equation is shown in the equation [25]. For the plane, the calculation of its logical value does not need the evaluation of the particle localization in the plane equation, because it is a coincident plane (because the distance is closer than *coincd* parameter, this is closer than $10^{-4}$, and the plane is parallel to the crossed plane 5, this is, both are defined as pz).

$$y^2 + z^2 - R^2 = 0^2 + 0^2 - 5^2 = -25 < 0 \tag{25}$$

In first place, the code tries with the cell 1. The logical value, for the cylinder, is false (the evaluated equation has different sign as the cell definition, the point is outside of the region defined by the cylinder for this cell). For the plane, the scalar product between the normal of the surface (upward) and the direction of the particle motion (upward) is positive, the logical value is true. The region is defined as the intersection of both surfaces. The logical function for the point (0,0,0) and the cell 1, and its evaluation are shown in the equation [26].

$$0.1 = 0 \tag{26}$$

The result of the logical function evaluation marks that the point is outside of the cell 1. As this is not the searched cell, the code tries the next cell that satisfies the search conditions. In this case, the cell that the code tries, is the cell 2.

The logical function is built for this cell. The logical value, for the cylinder, is true (the evaluated equation has the same sign as the cell definition, the point is inside the region defined by the cylinder for this cell). For the plane, the scalar product between the normal of the surface (upward) and the direction of the particle motion (upward) is positive, the logical value is true. The region is defined as the union of both surfaces. The logical function for the point (0,0,0) and the cell 2, and its evaluation are shown in the equation [27].

$$1 : 0 = 1 \tag{27}$$

The result of the logical function evaluation marks that the point is inside the cell 2, and as this cell is a content cell (there is not a *fill* card in its definition), the search end. The cell where the particle is transported, is the cell 2, which is inside the cell 3.

The last task, in the *newcel* subroutine, is a possible change in the parameter $jsu$ (the crossed surface). If the cell 2 is defined by a coincident plane, that is defined to a positive distance, and whose scalar product between its normal definition and the direction of the particle motion is positive, then the crossed surface ($jsu$) is changed for this coincident plane. In this case, the scalar product with the only plane in the definition is negative, hence, any plane of this cell definition satisfy the conditions, thus, this parameter is not changed.

As this has been named in this section, when the subroutine *newcel* ends, the next step, in the simulation, is the subroutine *track*, where the nearest surface is searched (see section 3.1.3).

This subroutine calculates the distance, level by level, to the nearest surface in each level (according to the particle trajectory), and chooses the nearest surface between these distances. The first level, where the subroutine calculates, is in the transport level (cell 2). As the code has just exited of the *newcel* subroutine, the code knows the logical function for this cell 2 and the point where the particle is (this function is shown in the equation [27]).

First, the distances to the surfaces are calculated, except the distance to the crossed point, in this case,except to the distance to the crossed plane (this plane that is pointed by the parameter $jsu$, is the plane 5). Then, the surfaces are ordered (the nearest surface before), this order is the plane 6 and the cylinder 10. In order to know which surface is crossed, the values of the logical function are changed value by value associated to the surface according with the order.

When the logical value associated to the plane 6 is changed, the logical function is like the equation 28

$$1 : 1 = 1 \tag{28}$$

The evaluation of the logical function is true, therefore, the plane 6 is not the nearest surface in the particle trajectory. The next surface is the cylinder. The logical function, after of the change of the logical value associated for the cylinder, is shown in the equation[29]

$$0 : 1 = 1 \tag{29}$$

The evaluation of the logical function is true, therefore, the cylinder 10 is not the nearest surface in the particle trajectory. There is not anymore surfaces in this cell, hence, according to the results that the code calculates, there is not anymore surface in the cell 2 in the trajectory. This result would not be obtained, it is the logical fail that causes the lost particles type 1.

The next level is also calculated, the cell in this level is the cell 1, the logical function is calculated again, and the obtained result is shown in the equation [24]. The only surface, in the particle trajectory, is the surface 7. When its logical value is changed in the logical function of the equation [24] leads to equation [30].

$$1.1.1.1.1.0 = 0 \tag{30}$$

This is a possible nearest surface in the cell 1, and as the other level has not another possible distance smaller, this is the nearest surface that the particle crosses along its trajectory.

When the code exits of the *track* subroutine, the particle is put on the surface 7. As it was said, this is a logic error. In the section 3.3 two possible solutions are proposed. Below, both solution are shown.

If the first proposed solution in section 3.3 is applied, the logical function of cell 2 is recalculated before the search the nearest surface. In this occasion, the equation of the plane is used for calculating the logical value of the plane 6. The equation evaluation is negative, and the cell definition associated to this cell is negative, then the logic equation is

$$1 : 1 = 1 \tag{31}$$

when the logical value associated to this plane is change, the particle is still into the region, but when the cylinder is also change the logical function is

$$0 : 0 = 0 \tag{32}$$

In this case, the transport level has also a nearest surface, and as this surface is nearer than the plane 7, hence, the nearest surface is the cylinder 10.

If the second proposed solution in section 3.3 is applied, the parameter $jsu$ is changed to the surface 6 in the *newcel* subroutine, when the nearest surface is searched, as the surface 6 is the crossed plane, it is not taken into account. For this reason, the logical value changed from the logical function of the equation [27] is the logical value associated to the cylinder, the result is

$$0 : 0 = 0 \tag{33}$$

In this case, the cylinder is also found.

# References


[1] Laura Estévez Núñez. Evaluation of the effects of ionizing radiation in iter's plasma position reflectomery system. Master's thesis, Técnico Lisboa, 2016.

[2] M.Garcia F.Ogando J.Sanz, R.Juarez. Desarrollo técnico del área de simulación computacional. Technofusion(II), 2017.

[3] Gabriel Pedroche Sánchez. Ejercicios de verificación de la herramienta computacional d1suned y análisis de la propagación de incertidumbres en la metodología r2s para el cálculo de dosis residual. Master's thesis, UNED, 2016.

[4] X-5 Monte Carlo Team. *MCNP – A General Monte Carlo N-Particle Transport Code Volume I: Overview and Theory.* Los Alamos, version 5 edition, Aoruk 2003.

[5] X-5 Monte Carlo Team. *MCNP – A General Monte Carlo N-Particle Transport Code Volume II: User's Guide.* Los Alamos, version 5 edition, Aoruk 2003.

[6] X-5 Monte Carlo Team. *MCNP – A General Monte Carlo N-Particle Transport Code Volume III: Developer's Guide.* Los Alamos, version 5 edition, Aoruk 2003.

[7] www.iter.org.

[8] J.Sanz. *Seguridad e impacto medioabliental de instalaciones de fusión nuclear. Metodologá de análisislisis y aplicaciones.* Universidad de Educación a Distancia. Departamento de Ingeniería Energética. Ingeniería Nuclear, 2008.

[9] Experimental validation of shutdown dose rates, Final Report, ITER Task T-426, June 2001; available reports and data: NEA-1553/55, OECD-NEA.

[10] Shutdown dose rate at JET with the new ILW and prediction of the expected dose level after future tritium experiment, Final Report, JET Task JW12-FT-5.43, Nov. 2014; available reports and data: on request, June 2001.

[11] R.Juarez. New ep#12 interspace modelling (local contribution)(privated).

[12] DO05. model of ep#11 port plug interspace based on modular concept (privated).

[13] OMF. 331 lot 1 to6 - deliverable d1 - technical report: Local mcnp model of epp #16 (26ur5t v1.2).

[14] http://www.fds.org.cn/en/index.asp.

[15] http://www.spaceclaim.com.

[16] F.Ogando R.Juárez P.Sauvan, J.P. Catalán and J.Sanz. Development of the r2suned code system for shutdown dose rate calculations. *IEEE Nuclear and Plasma Sciences Society*, February 2016.