OECD/NEA WPEC SG45

Policy driven input file generation applied to MCNP

W. Haeck, J. Alwin, K. Spencer

May 11, 2020

NNSA
National Nuclear Security Administration

Managed by Triad National Security, LLC for the U.S. Department of Energy's NNSA

Introduction

Most organisations make choices to handle specific things when preparing input files for validation purposes

- Which elemental abundance data to use
- How to decompose specific elements
- Setting calculation precision

Consequences

- Differing calculation results between different organisations for the same case
  - Even when using the same code!
  - Illustrated by the intercomparison excercise
- Input files prepared by one organization are not necessarily correct for another

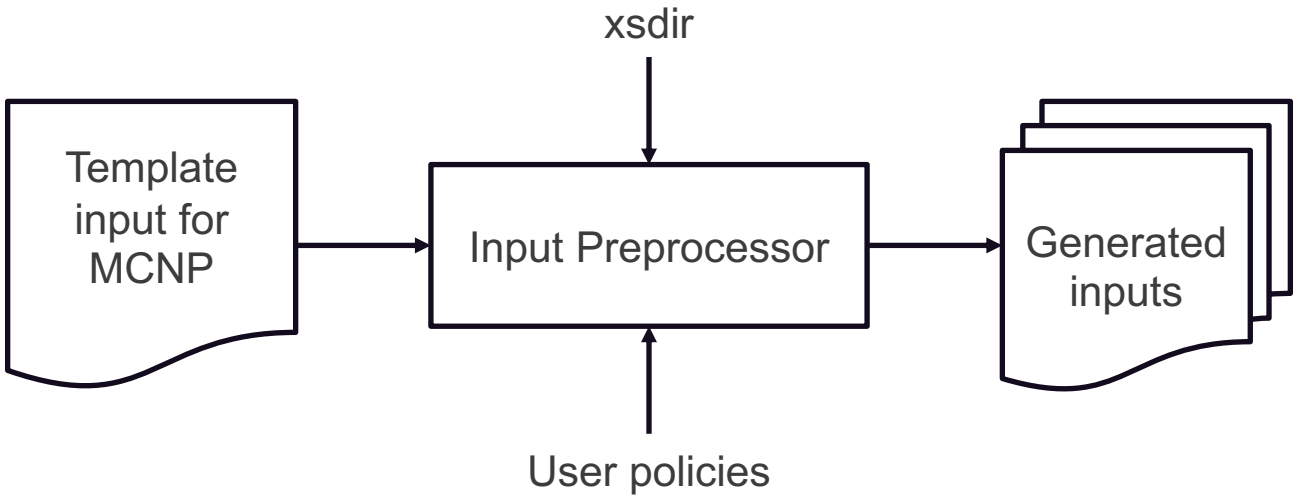
A solution : policy driven input generation

Maintain a single set of “template” input files for each code

- Follow the benchmark description as close as possible
- Devoid of specific library information or other organisational “choices”

Generate the input files at the request of the user

- Decompose elements based on nuclear data availability and user choices, etc.



Policy overview – elemental decompostion

An overview of identified policies available in the QA document

- See https://git.oecd-nea.org/science/wpec/sg45/documents

Abundance data policies

- There will be a default choice for the abundance data
- User defined abundance data can be provided as json files

Default treatment of isotopes with missing nuclear data files

- The missing isotopes in a natural element are corrected by renormalizing the abundances (i.e. the abundance of the missing isotope is distributed over all other isotopes)

```
{
  "abundances": {
    "H" : { "H1" : 1.0 },
    "He" : {"He3" : 1.34e-06, "He4": 0.99999866 },
    "Li" : { "Li6" : 0.0759, "Li7": 0.9241 },
    "C" : { "C12" : 0.9893, "C13": 0.0107 },
    "N": { "N14" : 0.99636, "N15": 0.00364 } }
}
```

Policy overview – elemental decompostion

Default decomposition can be overriden

- Through a user provided abundance file
- Applying special user policies

Decomposition policies will cover user overrides for select elements

- Decompose a user defined element as a single or its most abundant isotope
  - For example: H as H1, C as C12, N as N14, O as O16
- If a user defined isotope is missing, add its contribution to another user defined isotope (of the same element)
  - For example: add O18 to O17, add W180 to W182

A very special case (which should only happen for old libraries):

- Both the element and every isotope of the element have no nuclear data
- The missing element is removed and the atom density of the material is adjusted

Header information

Header information to trace these choices

- Keeping track of these choices is important
- Including them in the generated input files is the best way

```
c policy : userAbundanceFile
c abundances : /home/janedoe/abundances/abundances.lanl.ncs.json
c
c policy : userDecomposeSpecificElement
c element : O
c isotopes : O16
c
c policy : userDecomposeSpecificElement
c element : N
c isotopes : N14
c
c policy : userCalculationPrecision
c particles : 10000
c inactive cycles : 100
c active cycles : 500
```

Python coding

We've been working on an advanced MCNP input parser

```
# parse an existing input file
mcnp = McnpInput( 'pu-met-fast-001-000-rev4.mcnp' )

# retrieve and modify gallium in the materials
pu = mcnp.materials()[0]
ga = pu.composition[ '31000' ]
del pu.composition[ '31000' ]
pu.composition.update( { '31069' : 0.000826605216, '31071' : 0.000548594784 } )

# set kcode values
kcode = mcnp.criticalityCalculation()
kcode.nsrck = 10000
kcode.rkk = 1.
kcode.ikz = 100
kcode.kct = 600

# add kopts card
mcnp.data.append( McnpCriticalityCalculationOptions( kinetics = True ) )

# write the new file
newfile = mcnp.write()
```

Python coding

We have serialisation, deserialisation of json abundance files and decomposition of compositions using the underlying abundances

```
# create an abundance table
table = AbundanceTable( '/home/janedoe/abundances/abundances.lanl.ncs.json' )

# decompose gallium
composition = table.decompose( Element( 'Ga' ), 1.3752e-3 )

# transform to MCNP composition
mcnp_composition = { str( nuclide.z * 1000 + nuclide.a ) : comp
                     for nuclide, comp in composition.items() }
```

Example

Template MCNP file

```
pu-met-fast-001-000-rev4
1   1 4.029014e-2  -1   imp:n=1
2   0               1   imp:n=0

1   so  6.39157

mode n
totnu
kcode
ksrc 0. 0. 0.
c material 1: Plutonium
m1   94239 3.7047e-2
     94240 1.7512e-3
     94241 1.1674e-4
     31000 1.3752e-3
```

Generated MCNP file

```
pu-met-fast-001-000-rev4
c policy : defaultAbundanceFile
c policy : defaultCalculationPrecision
c policy : userCalculateEffectiveDelayedNeutronFraction
1   1 4.029014e-2  -1   imp:n=1
2   0               1   imp:n=0

1   so  6.39157

mode n
totnu
kcode 10000 1. 100 600
kopts kinetics=yes
ksrc 0. 0. 0.
c material 1: Plutonium
m1
     94239 0.037047
     94240 0.0017512
     31069 0.000826605216
     31071 0.000548594784
```