INL/MIS-24-76612-Revision-0

Going under the hood of MontePy: A python API for MCNP input files

March 2024

Micah D Gale, Travis J Labossiere-Hickman

Changing the World's Energy Future

Idaho National Laboratory

INL is a U.S. Department of Energy National Laboratory operated by Battelle Energy Alliance, LLC

INL/MIS-24-76612-Revision-0

Going under the hood of MontePy: A python API for MCNP input files

Micah D Gale, Travis J Labossiere-Hickman

March 2024

Idaho National Laboratory
Idaho Falls, Idaho 83415

http://www.inl.gov

Prepared for the
U.S. Department of Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

March 7, 2024

Micah Gale
Travis Labossiere-Hickman
MontePy developers

Going under the hood of MontePy

A python API for MCNP input files

INL/MIS-24-76612

Idaho National Laboratory

Review of MCNP inputs

- Fortran & Punchcard
- Predates: XML, JSON, SGML
- Very specialized, and pointer based
- Cell example

```
123456      654321      -10.02      -1  2  3  4  imp:n=1.0 imp:p=0.5  fill = 10 ( 0 10.0 20.0)
```

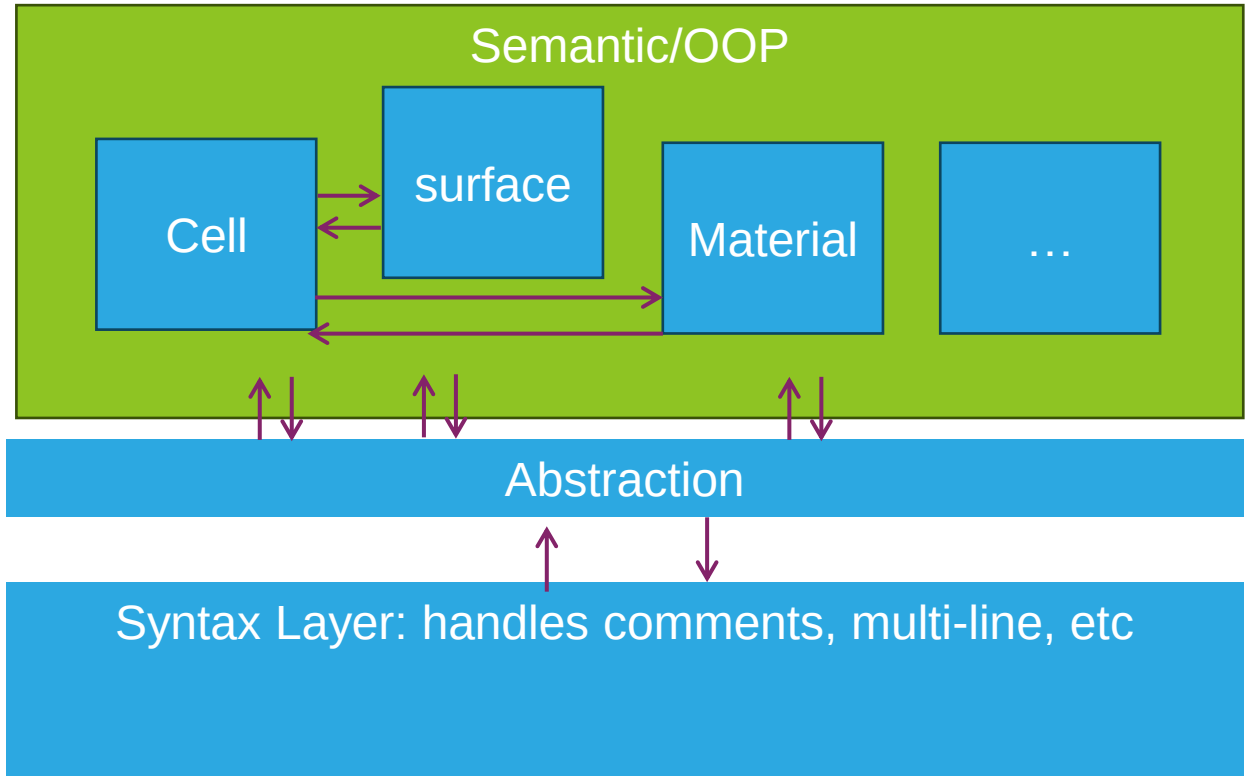
Cell #	mat num	density	geometry	parameters

Problem

- Can’t use standard library (e.g., `import json`)
- Updating pointers can be very tedious
- Finding objects by number is difficult

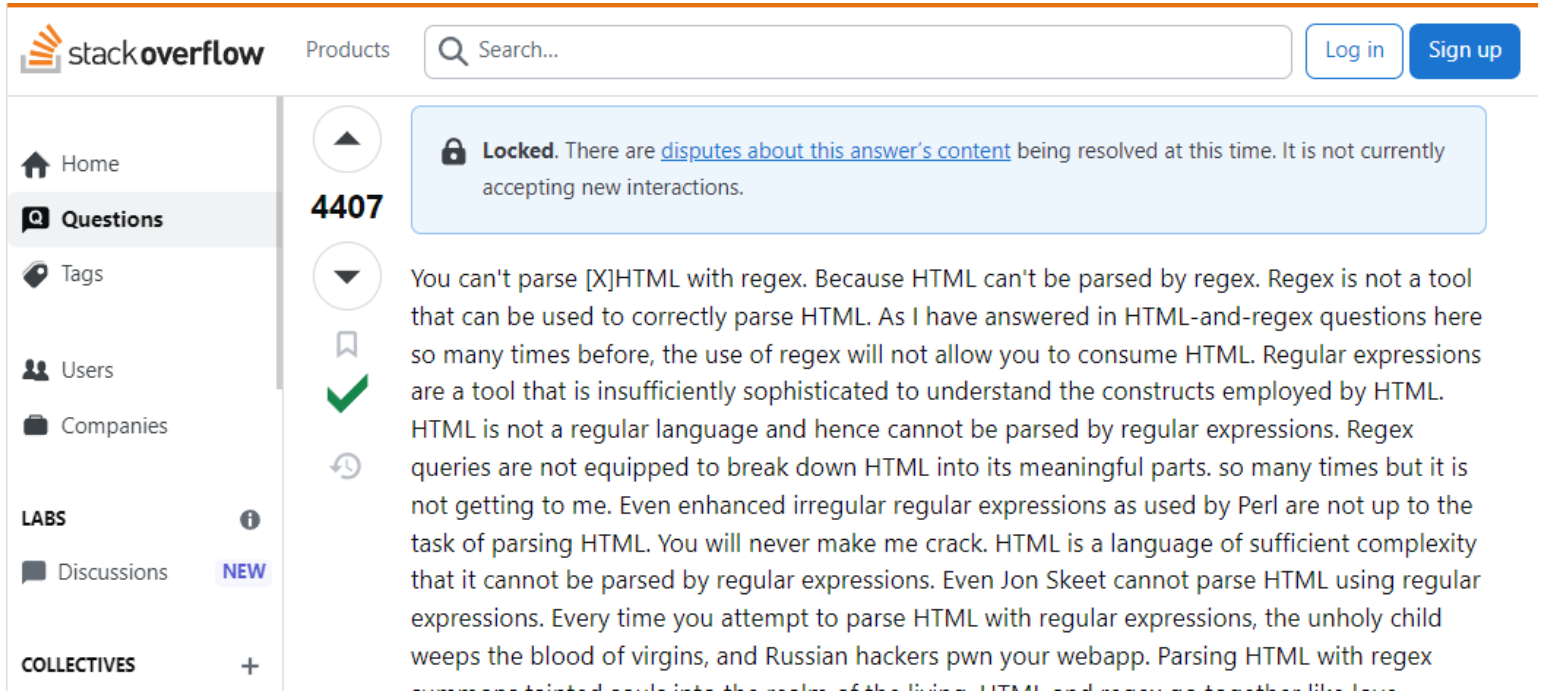
Solution: MontePy

- Object oriented interface



Very Naïve: Regular expressions

- MCNP is not a Regular language!



https://stackoverflow.com/a/1732454

Naïve first attempt: fake parsing (<=0.1.7)

- Handle syntax
- Split on spaces

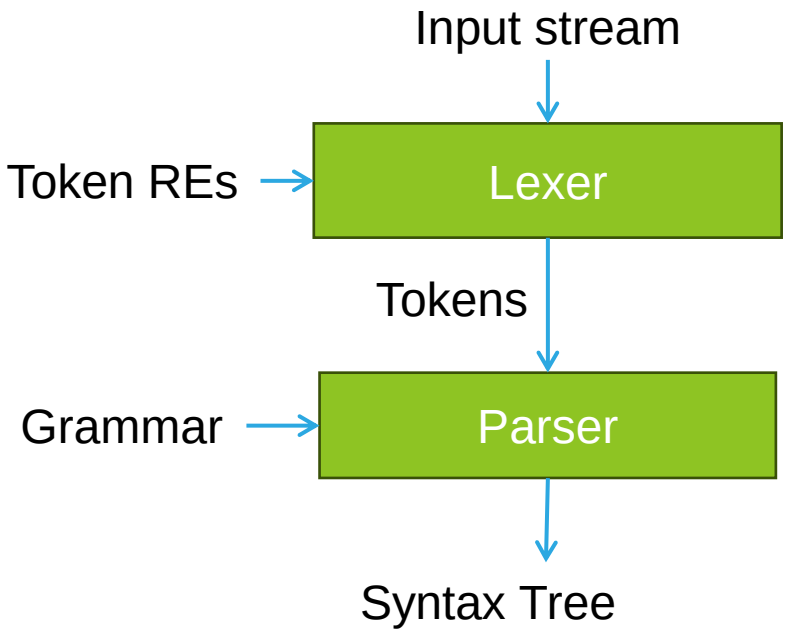
```
123456      654321  -10.02      -1  2  3  4
          imp:n = 1.0 imp:p 0.5
          fill = 10 ( 0 10.0 20.0)
```

```
["123456","654321","-10.02","-1", "2","3","4",
"imp:n","=","1.0","imp:p", "0.5", "fill"…]
```

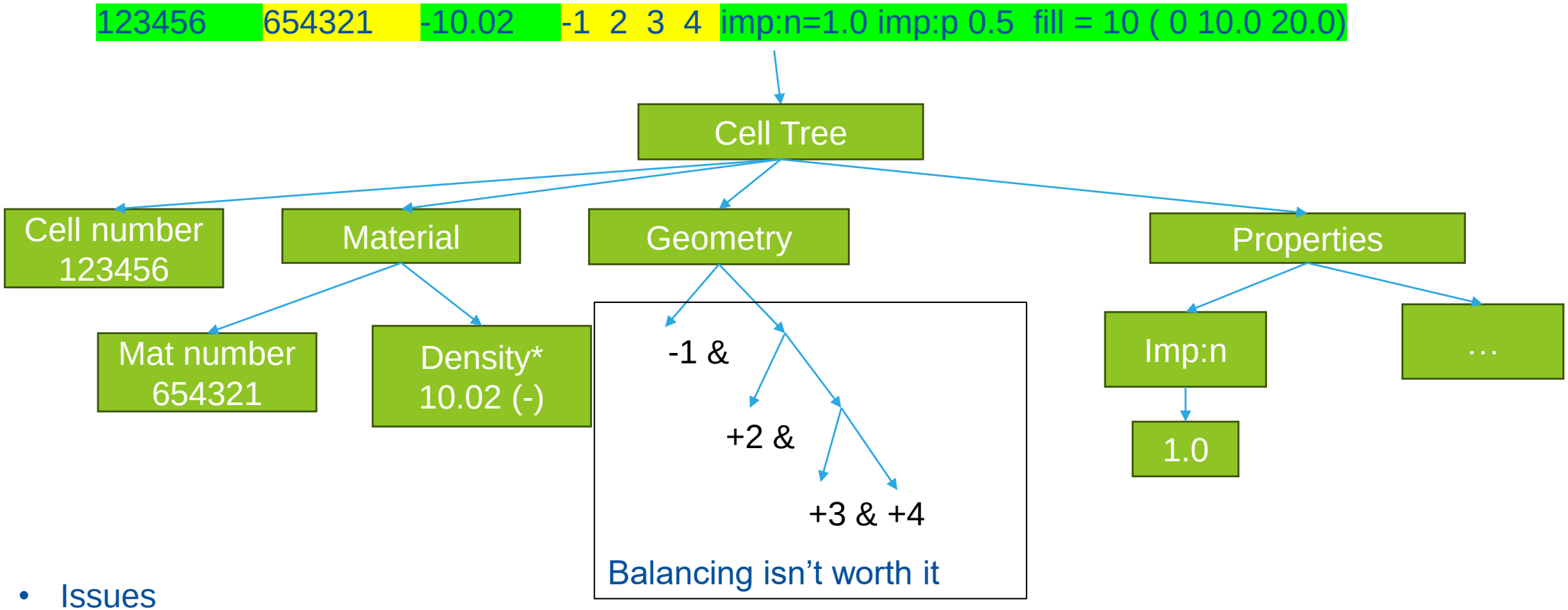
- Lots of formatting reconstruction headaches
- Users actually like their formatting?

Finally doing it right: Lexers Parsers and syntax trees, oh my (>= 0.2.0)

- Requires Context-free grammar
- Uses Lookahead left-to-right (LR) Parser (LALR(1))
- MCNP grammar is contextual
  - Break into context-free chunks
  - Build multiple lexers and parsers



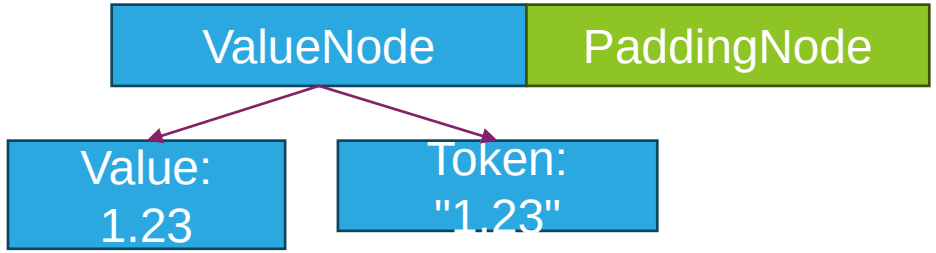
Example Abstract Syntax Tree



- Issues
  - Whitespace is an operator/ implicit operators
  - Users like formatting
    - Need "black", not "python"

Concrete Syntax Trees

- Introduce padding
  - Whitespace + comments
- Glue to leaves



- Reverse Engineering formatting
  - `"10.02 " -> "10.03 "`, *not* `"10.021115 "`
  - `"6.1+23 $ hi" -> "1.0+01 $ hi"`

Everything is an Object, and every variable a pointer

- Objects point to each other
- When possible, properties of objects point to ValueNode
  - When reexported the ValueNode doesn't need to be updated in the syntax tree
  - Everything handled by a decorator to keep it DRY.

```
@make_prop_val_node("_number", int, validator=_number_validator)
def number(self):
    """
    The current cell number that will be written out to a new input.

    :rtype: int
    """
    pass

@make_prop_pointer("_material", (Material, type(None)), deletable=True)
def material(self):
    """
    The Material object for the cell.

    If the material is None this is considered to be voided.

    :rtype: Material
    """
    pass
```

Acknowledgements

- This research made use of Idaho National Laboratory's High Performance Computing systems located at the Collaborative Computing Center and supported by the Office of Nuclear Energy of the U.S. Department of Energy and the Nuclear Science User Facilities under Contract No. DE-AC07-05ID14517.
- Prepared for the U.S. Department of Energy Through the Advanced Fuels Campaign Program and the Isotopes Production Program Under DOE Idaho Operations Office Contract DE-AC07-05ID14517