Nested Try Anti Pattern Detection

I. Introduction

The anti-pattern is a commonly-used process, structure or pattern of action that, despite initially appearing to be an appropriate and effective response to a problem, has more bad consequences than good ones. I have tried implementing an anti-pattern called ”Nested Try”.

II. Nested Try

Every statement that we enter a statement in try block, context of that exception is pushed onto the stack. For example, the inner try block can be used to handle ArrayIndexOutOfBoundsException while the outer try block can handle the ArithemeticException (division by zero).

The ”nested try” antipattern refers to a situation in programming paradigm where one or more try blocks are nested within another try block. This can lead to complex and hard-to-understand code, as well as potential performance issues. The primary issue with nested try blocks is that they can make it difficult to understand the control flow of the code, as well as the specific exceptions that may be thrown. In addition, nesting try blocks can make the code more difficult to maintain and modify, as changes to one try block may require changes to other nested try blocks. We have to write a program to detect the nested try which is an antipattern.

```
python_files = find_python_files(input_directory)

results = {}
for file in python_files:
    with open(file, 'r') as f:
        try:
            tree = ast.parse(f.read())
            nested_tries = set()
            find_nested_tries(tree, file, nested_tries)
            if nested_tries:
                results[file] = nested_tries
        except SyntaxError:
            pass
```

Fig. 1. Usage of Abstract Syntax Tree

III. Execution

An ”AST” (Abstract Syntax Tree) refers to a data structure that represents the abstract syntactic structure of source code. It is often used in compilers, interpreters, and other language tools to analyze, transform, or generate code.

The above function named find nested tries takes three arguments: an AST node representing the root of the AST, a filename (as a string), and a set of line numbers.The function recursively walks through the AST, looking for nested try statements within the try blocks and except blocks.The

```
def find_nested_tries(node, filename, line_numbers):
    if is_try(node):

        for nested_node in node.body:
            if is_try(nested_node):
                line_numbers.add(nested_node.lineno)
        for handler in node.handlers:
            for nested_node in handler.body:
                if is_try(nested_node):
                    line_numbers.add(nested_node.lineno)
    for child_node in ast.iter_child_nodes(node):
        find_nested_tries(child_node, filename, line_numbers)
```

Fig. 2. find nested try function

implementation of the function first checks if the input node represents a try statement, by calling the is try function. If the node is a try statement, the function iterates through its body and handlers, looking for nested try statements. If a nested try statement is found, its line number is added to the set of line numbers.

IV. Results

I have executed the python code on large python projects to detect antipattern, and was able to obtain the results.

case 1:

python file single nested try

```
1  try:
2      # This is the outer try block
3      print('Outer try block')
4      try:
5          # This is the inner try block
6          print('Inner try block')
7          raise Exception('Inner exception')
8      except:
9          # This is the inner catch block
10         print('Inner catch block')
11         raise
12 except:
13     # This is the outer catch block
14     print('Outer catch block')
15
```

Fig. 3. single nested try function

Case 2 :

Multiple nested try functions. Refer figure 4

Results for above execution: Refer figure 5

case 3 :

Executing and finding nested try anti patterns on a large python project. Refer figure 6 and figure 7

Here is the snapshot of the project files, written piece of code iteratively detects each python file in the project and detects the anti-pattern

```python
try:
    # This is the outer try block
    print('Outer try block')
    try:
        # This is the inner try block
        print('Inner try block')
        raise Exception('Inner exception')
    except:
        # This is the inner catch block
        print('Inner catch block')
        try:
            # This is a nested try block inside the inner catch block
            print('Nested try block')
            raise Exception('Nested exception')
        except:
            # This is the nested catch block
            print('Nested catch block')
            raise
except:
    # This is the outer catch block
    print('Outer catch block')

```

Fig. 4. multiple nested try function

```
C:\Users\kosur\OneDrive\Desktop\assignment presentation\assignment\python files\nested-try-block-02.py:
        Line 4
        Total nested try blocks: 1
C:\Users\kosur\OneDrive\Desktop\assignment presentation\assignment\python files\nested-try-block-03.py:
        Line 11
        Line 4
        Total nested try blocks: 2
```

Fig. 5. results for case 1 and case 2

```
C:\Users\kosur\OneDrive\Desktop\assignment presentation\assignment>python detect_nested_tries.py
nested-try-block-02.py
nested-try-block-03.py
call_center.py
__init__.py
deck_of_cards.py
__init__.py
hash_map.py
__init__.py
lru_cache.py
__init__.py
online_chat.py
__init__.py
parking_lot.py
__init__.py
mint_mapreduce.py
mint_snippets.py
__init__.py
pastebin.py
__init__.py
query_cache_snippets.py
__init__.py
sales_rank_mapreduce.py
__init__.py
social_graph_snippets.py
__init__.py
web_crawler_mapreduce.py
web_crawler_snippets.py
__init__.py
```

Fig. 6. list of python files in the project

```
C:\Users\kosur\OneDrive\Desktop\assignment presentation\Python-calculus\calculus\solutions\object_oriented_design\call_center\call_center.py:
        Line 74
        Total nested try blocks: 1
C:\Users\kosur\OneDrive\Desktop\assignment presentation\Python-calculus\calculus\solutions\object_oriented_design\lru_cache\lru_cache.py:
        Line 84
        Line 45
        Total nested try blocks: 2
C:\Users\kosur\OneDrive\Desktop\assignment presentation\Python-calculus\calculus\solutions\object_oriented_design\online_chat\online_chat.py:
        Line 108
        Total nested try blocks: 1
C:\Users\kosur\OneDrive\Desktop\assignment presentation\Python-calculus\calculus\solutions\object_oriented_design\parking_lot\parking_lot.py:
        Line 104
        Line 142
        Total nested try blocks: 2
```

Fig. 7. results

V. Project accessibility

Github link for the project : https://github.com/indraneel-git/SOEN—Anti-pattern-Detection

References

[1] https://en.wikipedia.org/wiki/Anti-pattern
[2] https://www.javatpoint.com/nested-try-block: :text=It
[3] https://methodpoet.com/nested-try-catch/ Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.