P0 with *Exception Handling*

Slides and Demo Prepared by
Fanping Jiang, Meijing Li, Kevin Zhou

Instructor: Dr. Emil Sekerinski
Course: Computer Science 4TB3
Dept. of Computing and Software, McMaster University
April, 2022

Background

- **Exceptions**
  - Handle errors at runtime
  - A fundamental element of coding
  - The latest version of WABT recently adds exception
- **P0**
  - Python based Compiler
  - Generates WebAssembly code
  - ***To be extended to generate WebAssembly exceptions***

Exception Structures Handled in Java

- ***try-catch*** statement

```
try {
      statements
} catch (Exception e1) {
      statements
} catch (Exception e2) {
      statements
}
...
```

- ***throw*** statement

```
throw Exception(e)
```

Note:
e is a variable, storing the exception-type object

Exception Structures Handled in P0

- ***try-catch*** statement

```
try
    statements
catch i1
    statements
catch i2
    statements
...
```

- ***throw*** statement

```
throw i
```

Note:
i is a non-negative integer, as the index of **exception**

Implicit Exception

- **Why we need Implicit Exception Handling?**
  - Most errors can be detected by old P0 and corresponding exceptions will be raised by function *mark()* - no wat file can be generated
  - Some run-time errors can still exist in WebAssembly

***Index Out of Bound*:**

```
var a: [1..2] → integer
i := 3
b := a[i]
write(b)
```

Result:

- Valid wat and wasm files will be generated;
- runwasm function will output the value stored in memory that is outside the array **(dangerous!)**

Implicit Exception

- **Why we need Implicit Exception Handling?**
  - Most errors can be detected by old P0 and corresponding exceptions will be raised by function *mark()* - no wat file can be generated
  - Some run-time errors can still exist in WebAssembly

Div/Mod by 0:

```
y := 0
b := x div y
write(b)
```

Result:

- Valid wat and wasm files will be generated;
- runwasm function will raise the exception **(too late to detect!)**

Implicit Exception

We need to add implicit exception handling in P0 to detect the errors and generate WebAssembly with exceptions.

- **Why we need Implicit Exception Handling?**
  - Most errors can be detected by old P0 and corresponding exceptions will be raised by function *mark()* - no wat file can be generated
  - Some run-time errors can still exist in WebAssembly

Div/Mod by 0:

```
y := 0
b := x div y
write(b)
```

Result:

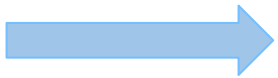
- Valid wat and wasm files will be generated;
- runwasm function will raise the exception **(too late to detect!)**

Implicit Exceptions Handling

Integer Exception Tag *(arbitrary numbers)*	Exception	Pre-defined Exception keyword
110	Index Out of Bound	**'indexoutofbound'**
111	Div by 0	**'zerodiv'**
112	Mod by 0	**'zeromod'**

- ***Index Out of Bound*:**

```
var a: [1..2] → integer
try
   i ← read()
   b := a[i]
catch indexoutofbound
   …
```



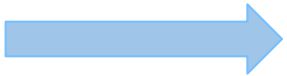
```
var a: [1..2] → integer
i ← read()
try
   if i < 1 or i ≥ 3
   then throw 110
   else b := a[3]
catch 110
   …
```

Underlying logic

Implicit Exceptions

- *Div/Mod by 0*:

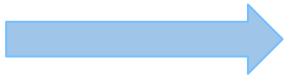
```
try
    y ← read()
    b := x div y
catch zerodiv
    …
```



```
try
    y ← read()
    if y = 0
    then throw 111
    else b = x div y
catch 111
    …
```

Underlying logic

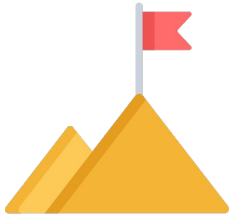
```
try
    y ← read()
    b := x mod y
catch zeromod
    …
```



```
try
    y ← read()
    if y = 0
    then throw 112
    else b = x mod y
catch 112
    …
```

Underlying logic

Implementation



- New Const added to ***SC***
  - THROW, TRY, CATCH
  - INDEXOUTOFBOUND, ZERODIV, ZEROMOD
- New statements added to ***P0***
  - Throw statement
  - Try-catch statement

- New functions added to ***CGwat***
  - genThrow(n)
  - genTry()
  - genCatch(n)
  - …

statement ::= ... | "throw" integer

| "try" statementSuite {"catch" (integer| implicitExcp) statementSuite}

implicitExcp ::= “indexoutofbound” | “zerodiv” | “zeromod”’

Translate Scheme

S	code(S)	
throw i	throw $ei	$ei is an exception tag added by the compiler automaticly
try S; catch i_1; S_1; ...; catch i_n; S_n	try code(S) catch ei_1; drop; code(S_1) ... catch ei_n; drop; code(S_n) catch_all end	note that drop is to pop out the exception tag 1

Translate Example

Input String

```
procedure sqrt(x: integer) → (r:
integer)
    if x < 0  then throw 39
    else
        r := 1
        while (r × r) ≤ x do r := r + 1
        r := r - 1
program test
    var x, a: integer
    try
        x ← read(); a ← sqrt(x); write(a)
    catch 39
        write(-1)
```

Compile as

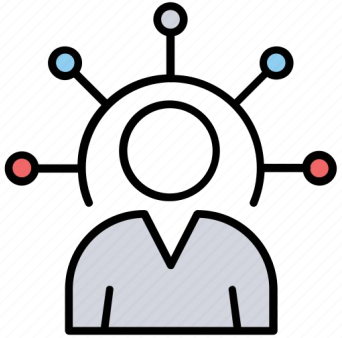
WebAssembly

```
(module
  ...
  (tag $e39 (param i32))
  (func $sqrt (param $x i32)
        (result i32)
  local.get $x
  i32.const 0
  i32.lt_s
  if
    i32.const 39
    throw $e39
  else
  ...
  )
```

```
(func $program
try
...
catch $e39
  drop
  i32.const -1
  call $write
catch_all
end
)
(memory 1)
(start $program)
)
```

Demo

Challenges

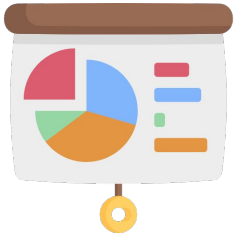


- Completely new topic
- WABT is recently updated with exceptions - lack of resources online
- How exception tag being used in *throw* and *try-catch* statements
- Tricky to handle implicit exceptions

Statistics, Documentation, Testing

- Documentation available in *readme* and *Wiki*
- Code change:
  - *SC*: 8 lines of codes added (Mainly Keywords)
  - *CGwat*: 63 lines of codes added, 9 added or modified functions
  - *P0*: 48 lines of codes added, 6 added or modified functions
- Test Cases:
  - A total of 15 test cases with descriptions and expected outputs to test all newly added components & functions

Resources



- WebAssembly/exception-handling documentation: https://github.com/WebAssembly/exception-handling
- WebAssembly exception code examples: https://github.com/WebAssembly/wabt/blob/main/test/parse/expr/try.txt
- P0 Source Code: based on the Ch 5

Visit Us

https://gitlab.cas.mcmaster.ca/cs4tb3-winter22/group-12

Fanping Jiang, Meijing Li, Kevin Zhou