5
Parallel Logic Regression on High Dimensional Boolean Space

410510026 張皓儒
309510165 陳臻和
310510154 蕭婷云

Outline

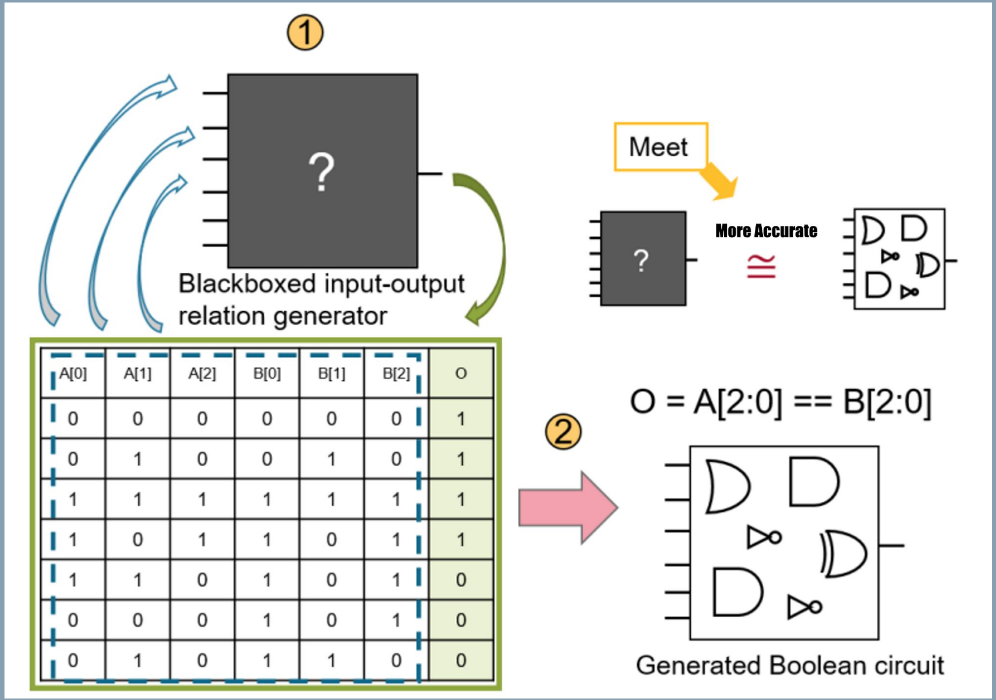
- **Introduction/motivation**
- **Problem statement**
- Proposed solution
- Evaluation
- Contributions of each member
- Conclusion

Introduction/motivation

- Logic regression could approximately reconstruct the unknown system
- Analyze a black-box model
  - Testing
  - Verification
  - Data analysis

- 2019 ICCAD CAD Contest
  - Given a block box circuit
  - Find a minimal Boolean logic circuit
  - Number of input variables ranges from 25 to hundreds
    - Even the smallest size of the truth table has 2^25 implications
    - Need a more effective method to solve

Problem statement

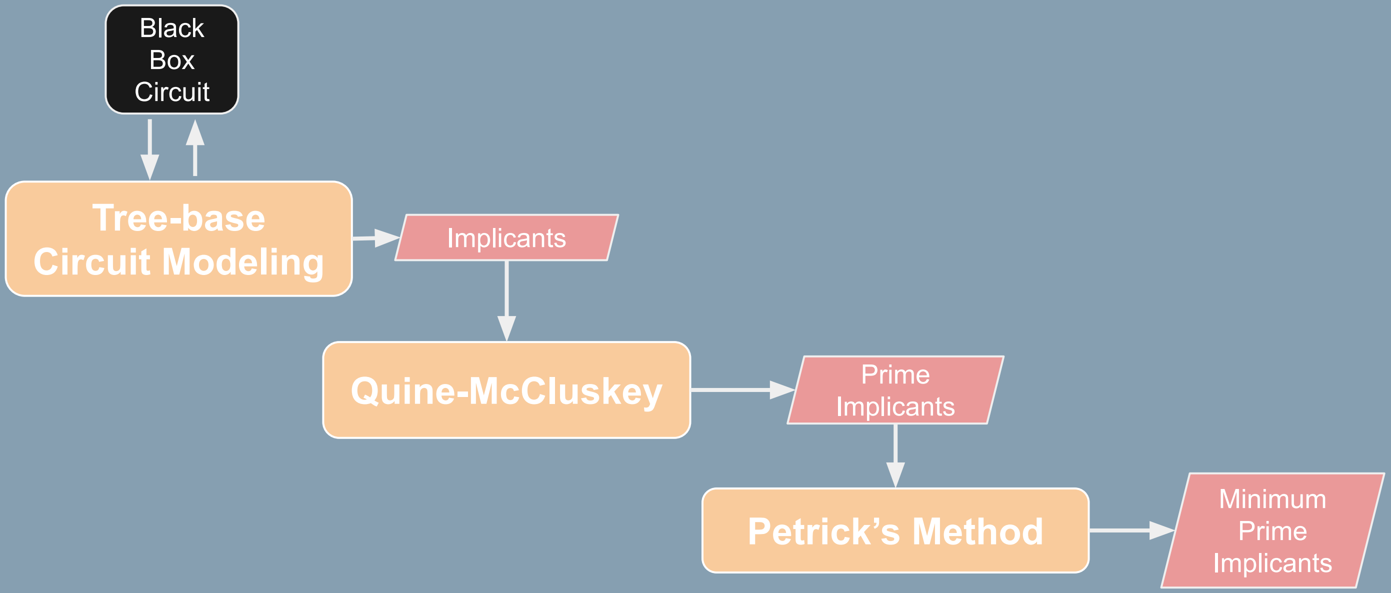


A[0]	A[1]	A[2]	B[0]	B[1]	B[2]	O
0	0	0	0	0	0	1
0	1	0	0	1	0	1
1	1	1	1	1	1	1
1	0	1	1	0	1	1
1	1	0	1	0	1	0
0	0	0	1	0	1	0
0	1	0	1	1	0	0

Outline

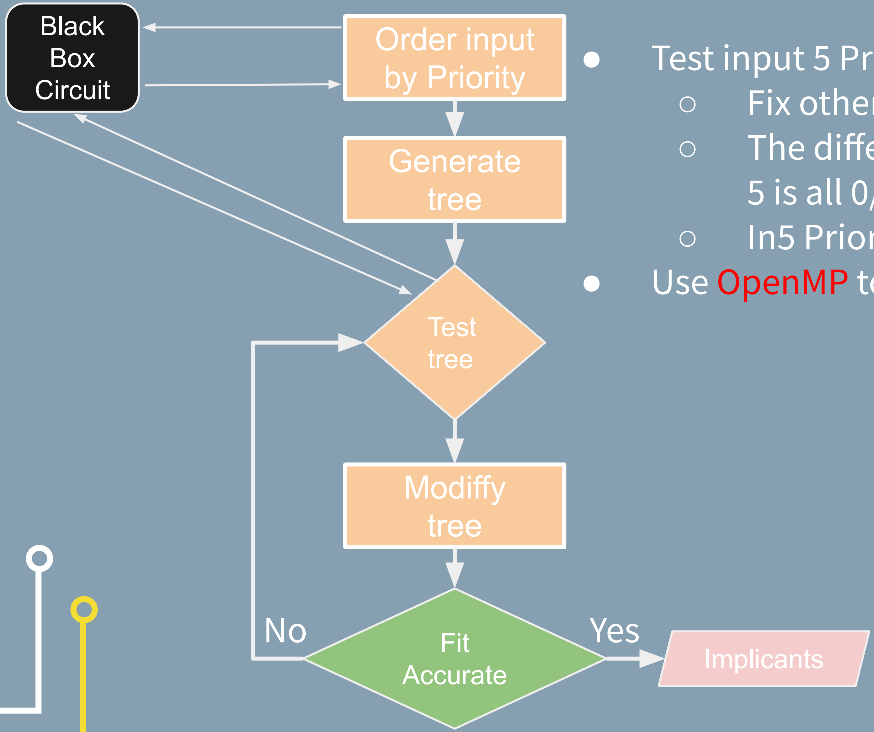
- Introduction/motivation
- Problem statement
- **Proposed solution**
  - Flow
  - Tree-base circuit modeling
  - Quine-McCluskey
  - Petrick’s Method
- Evaluation
- Contributions of each member
- Conclusion

Flow

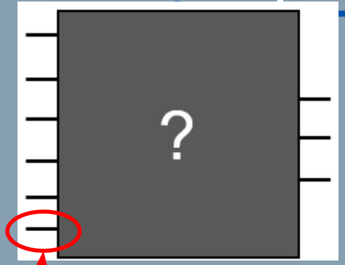


Black Box Circuit
Tree-base Circuit Modeling
Implicants
Quine-McCluskey
Prime Implicants
Petrick’s Method
Minimum Prime Implicants

Tree-base Circuit Modeling
Order input by Priority



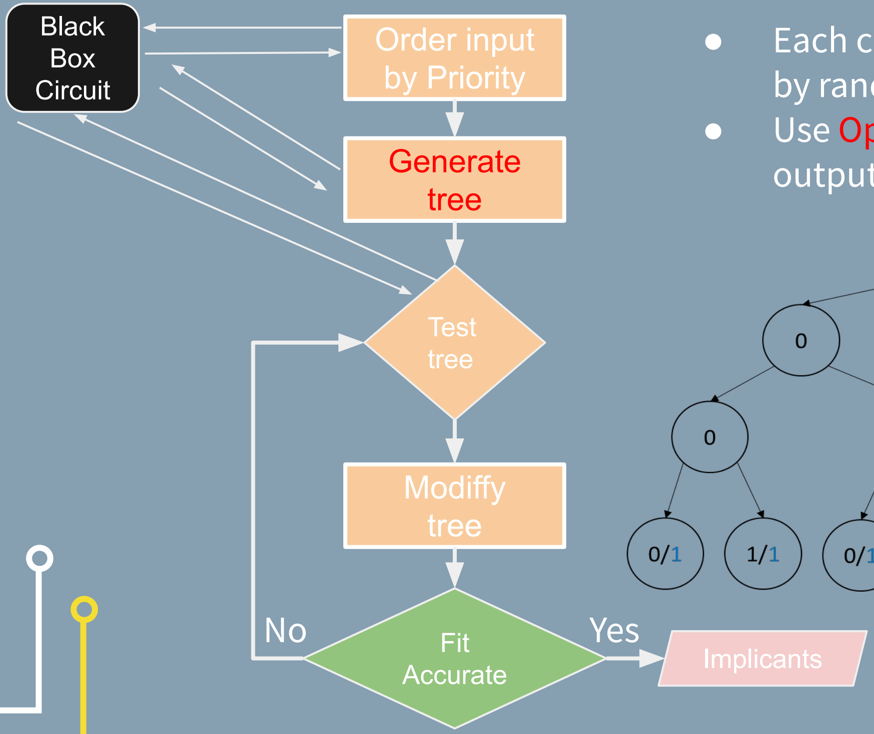
- Test input 5 Priority
  - Fix other input random pattern
  - The differ of output when input 5 is all 0/1
  - In5 Priority = ¾ = 0.75
- Use OpenMP to parallel test all inputs



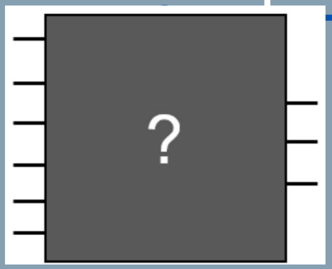
same

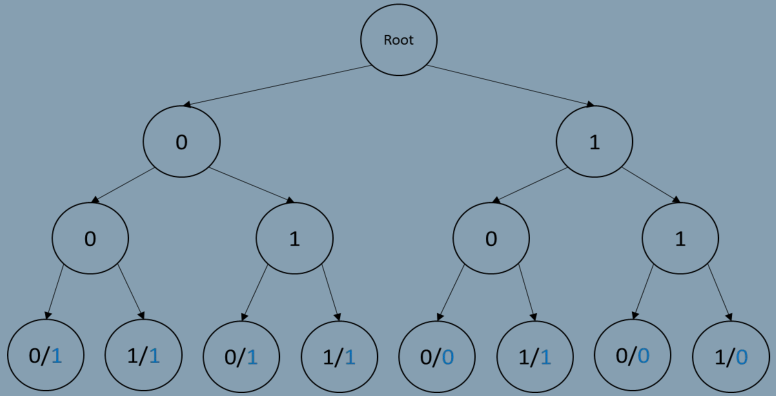
In0	In1	In2	In3	In4	In5	O1
0	0	0	0	1	0	1
1	0	1	0	0	0	1
0	1	0	0	1	0	1
1	1	1	1	0	0	1
0	0	0	0	1	1	0
1	0	1	0	0	1	1
0	1	0	0	1	1	1
1	1	1	1	0	1	1

Tree-base Circuit Modeling
Generate Tree



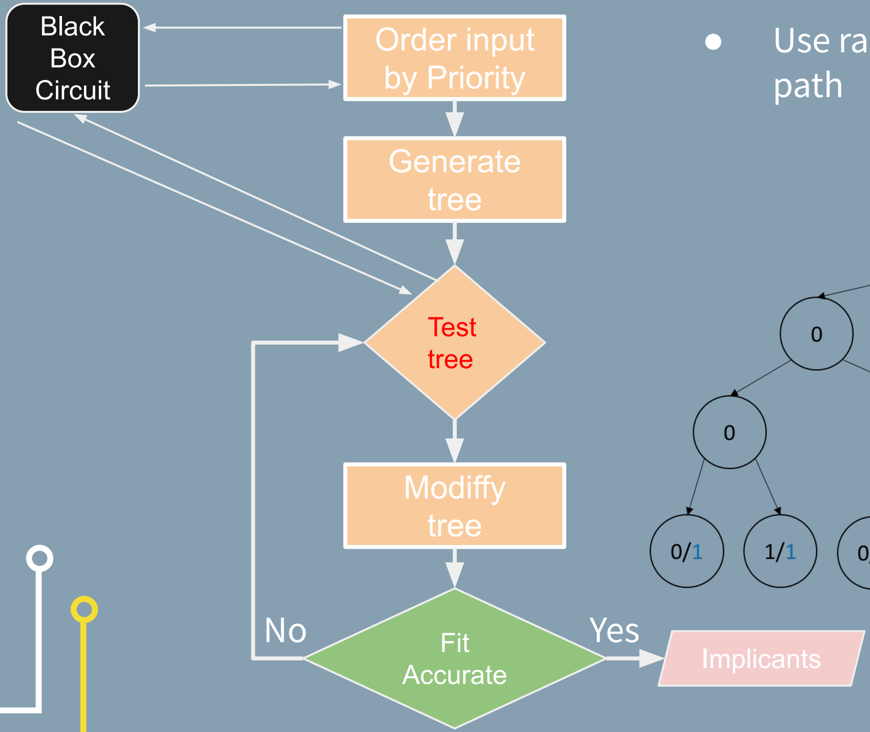
- Each circuit output has a tree construct by random pattern
- Use OpenMP to parallel construct all output tree



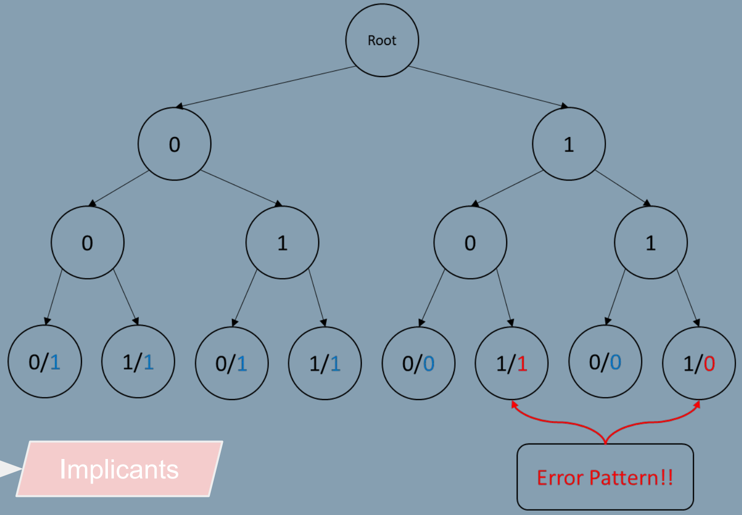


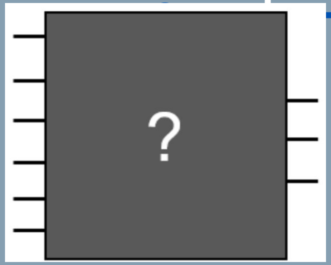
In0	In1	In2	In3	In4	O1
0	0	0	X	X	1
0	0	1	X	X	1
0	1	0	X	X	1
0	1	1	X	X	1
1	0	0	X	X	0
1	0	1	X	X	1
1	1	0	X	X	0
1	1	1	X	X	0

Tree-base Circuit Modeling
Test Tree



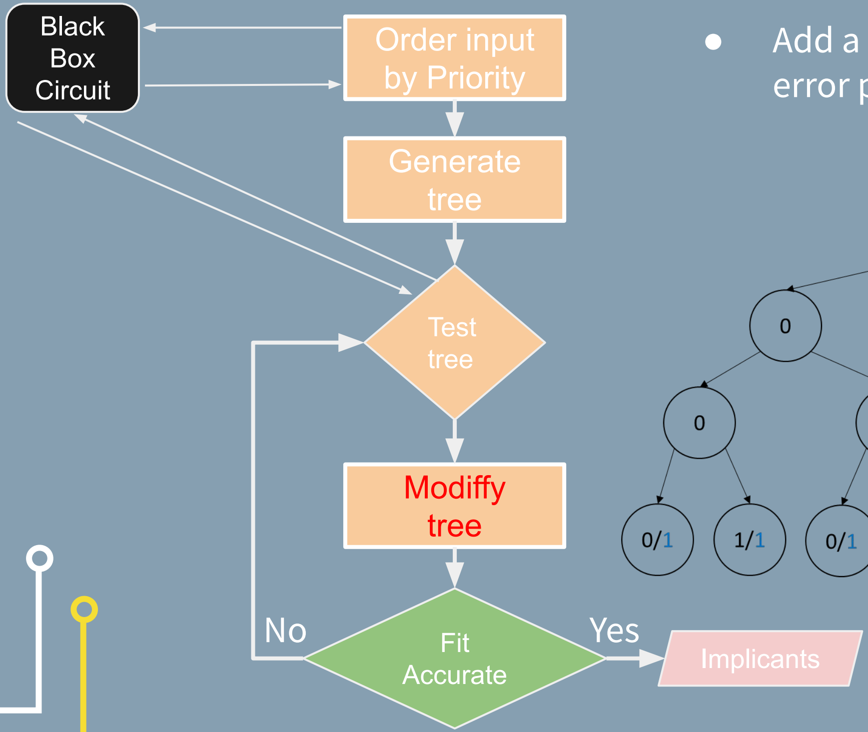
- Use random pattern to test the error path



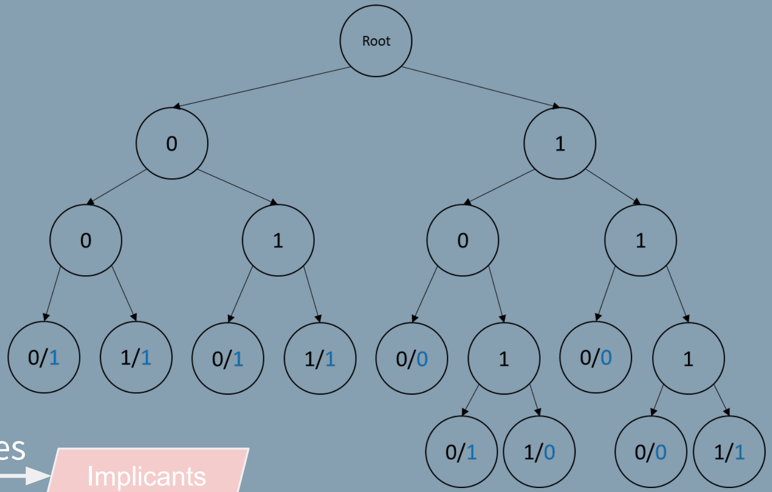


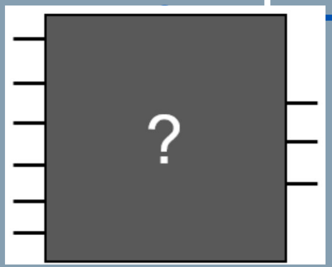
In0	In1	In2	In3	In4	O1
0	0	0	X	X	1
0	0	1	X	X	1
0	1	0	X	X	1
0	1	1	X	X	1
1	0	0	X	X	0
1	0	1	0	X	1
1	0	1	1	X	0
1	1	0	X	X	0
1	1	1	0	X	0
1	1	1	1	X	1

Tree-base Circuit Modeling
Modify Tree



- Add a new input from priority at the error path

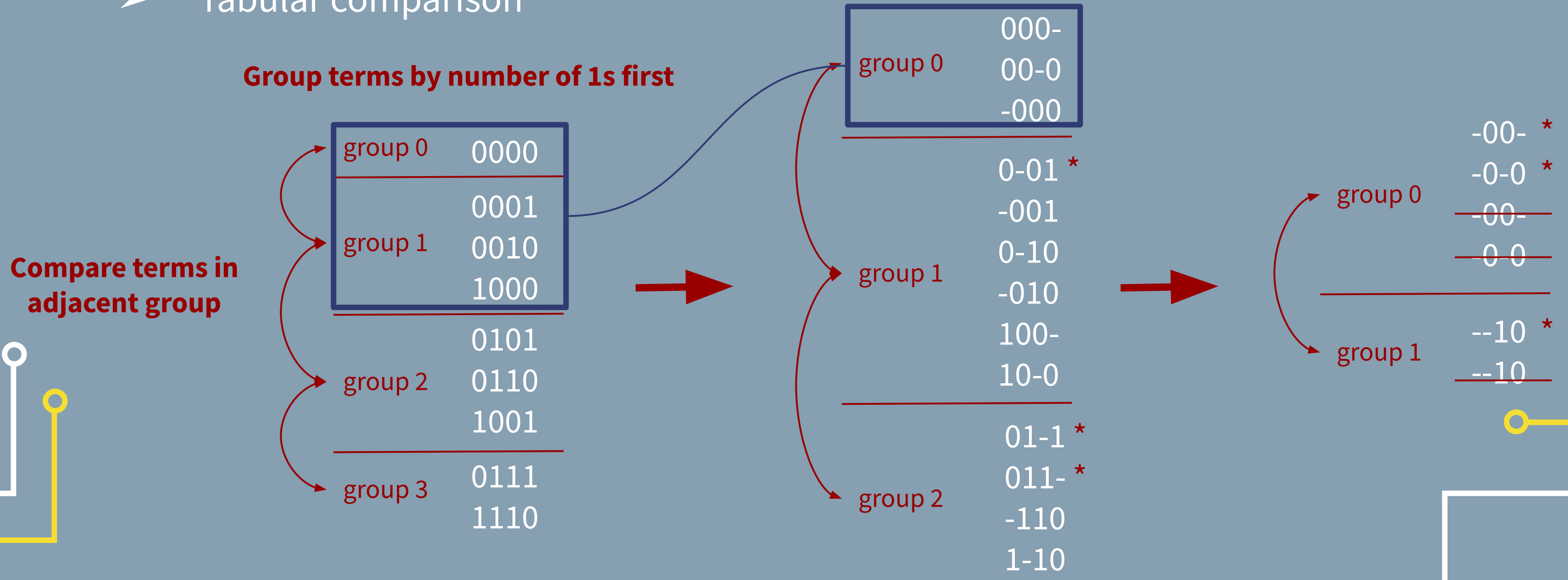




In0	In1	In2	In3	In4	O1
0	0	0	X	X	1
0	0	1	X	X	1
0	1	0	X	X	1
0	1	1	X	X	1
1	0	0	X	X	0
1	0	1	0	X	1
1	0	1	1	X	0
1	1	0	X	X	0
1	1	1	0	X	0
1	1	1	1	X	1

Quine McCluskey

- ➢ Combining two terms
  - ○ ABC + ABC’ = AB
  - ○ 111 110 11-
- ➢ Tabular comparison



Quine McCluskey - Pseudo Code

➢ Serial version

group 0	0000
group 1	0001 0010 1000
group 2	0101 0110 1001
group 3	0111 1110

```
for each adjacent groups g, h
{
    for i = 1 to g's size
    {
        for j = 1 to h's size
        {
            compare_and_merge( g[i], h[j] )
        }
    }
}
```

Quine McCluskey - Parallelization (task)

```
for each adjacent groups g, h
{
    #pragma omp parallel
    {
        #pragma omp single
        {
            for i = 1 to g's size
            {
                for j = 1 to h's size
                {
                    #pragma omp task
                    compare_and_merge( g[i], h[j] )
                }
            }
        }
    }
}
```

Low computation overhead

Quine McCluskey - Parallelization (for)

```
for each adjacent groups g, h
{
    #pragma omp parallel for collapse ( 2 )
    for i = 1 to g's size
    {
        for j = 1 to h's size
        {
            compare_and_merge( g[i], h[j] )
        }
    }
}
```

Petrick's Method

$P = (P_1 + P_2)(P_1 + P_3)(P_2 + P_4)(P_3 + P_5)(P_4 + P_6)(P_5 + P_6)$

$= P_1P_1P_2P_3P_4P_5 + P_2P_1P_2P_3P_4P_5 + \ldots\ldots + P_2P_3P_4P_5P_6P_6$ → 2*2*2*2*2*2 = 64 terms

$= P_1P_4P_5 + P_1P_2P_5P_6 + P_2P_3P_4P_5 + P_1P_3P_4P_6 + P_2P_3P_6$ → find minimum cost

			0	1	2	5	6	7
P_1	(0, 1)	$a'b'$	×	×				
P_2	(0, 2)	$a'c'$	×		×			
P_3	(1, 5)	$b'c$		×		×		
P_4	(2 6)	bc'			×		×	
P_5	(5, 7)	ac				×		×
P_6	(6, 7)	ab					×	×

Method 1 - task

- A single thread allocate memory & doing expansion
- Other threads calculating cost of each term & delete memory after used

➢ Very slow
➢ Run time of the single thread > Run time of other threads

Method 2 - parallel for loop

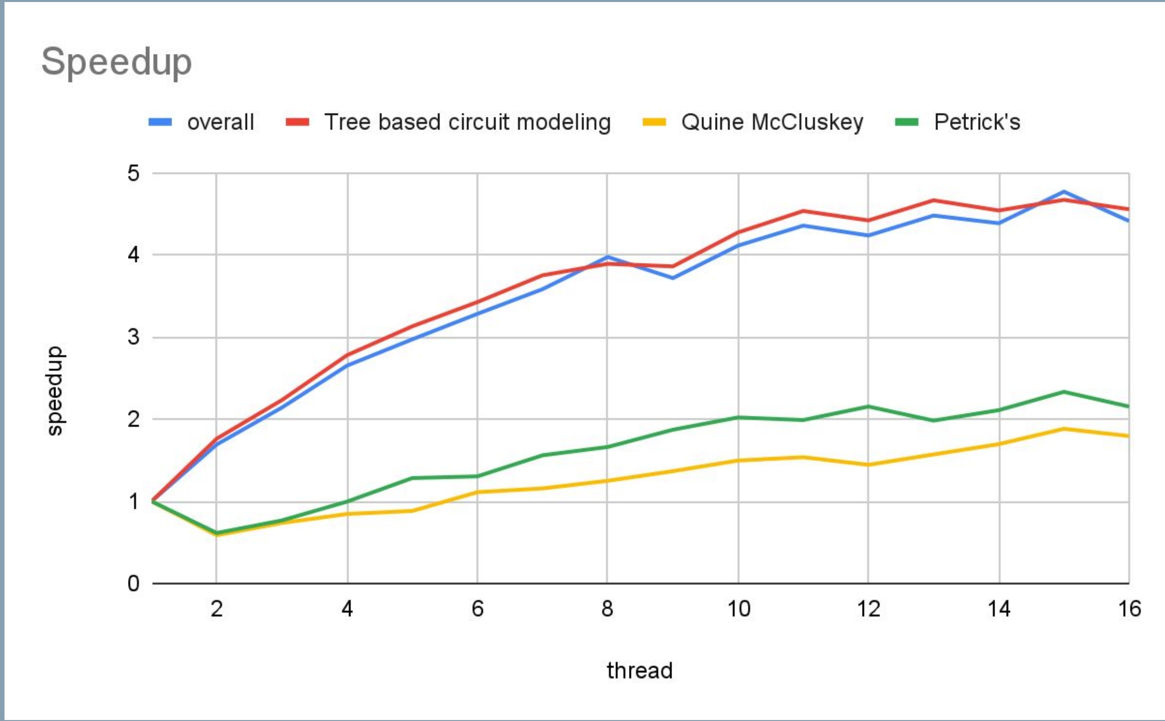
- Allocate total memory needed & expansion
- Calculate cost of terms

➢ Better parallelization. But need large memory space

Outline

- Introduction/motivation
- Problem statement
- Proposed solution
- **Evaluation**
  - **Result**
  - **Platform**
- Contributions of each member
- Conclusion

Result


Speedup
overall
Tree based circuit modeling
Quine McCluskey
Petrick's
speedup
0
1
2
3
4
5
2
4
6
8
10
12
14
16
thread

Platform

- Intel(R) Xeon(R) Silver 4208 CPU @ 2.10GHz * 2
  - 16 cores, 32 threads
- 64GB RAM

Outline

- Introduction/motivation
- Problem statement
- Proposed solution
- Evaluation
  - Platform
  - Evaluations on different parallelization metrics
- **Contributions of each member**
- **Conclusion**

Contributions of each member

- Tree-base circuit modeling： 410510026 張皓儒
- Quine-McCluskey: 309510165 陳臻和
- Petrick’s Method: 310510154 蕭婷云

Conclusion

- Limitation of concurrent read/write file — shared resources
- Random processes will cause different speedup
- Task v.s. Parallel for loop

Thanks