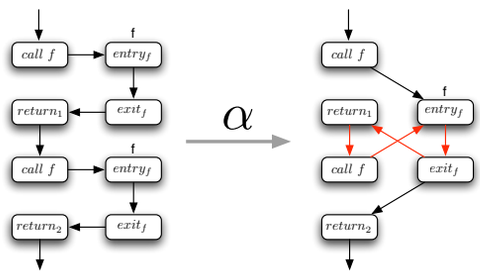
Hakjoo Oh, Seoul National University

Avoiding Spurious Interprocedural Cycles Matters For Scalable Analyses

1. Motivation and Goal

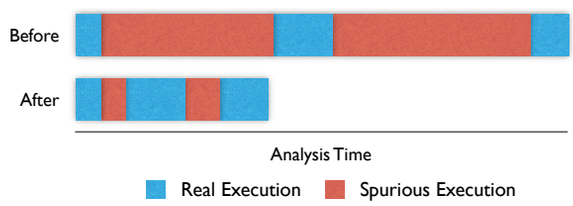
- Motivation: A bottleneck of scalability

• Abstraction of procedure call induces spurious cycles.



• Analysis can spend most of its time analyzing them.
• This is undesirable since they are spurious executions.

- Goal: Reducing spurious executions for efficiency



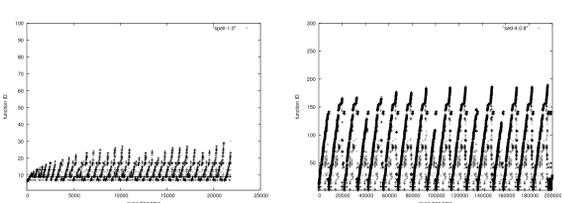
2. Why Are Spurious Cycles Expensive?

• Real C programs often contain large spurious cycles.

Program	Procedures Ratio	Nodes Ratio
spell-1.0	24/31(77%)	751/782(95%)
gzip-1.2.4a	100/135(74%)	5,988/6,271(95%)
sed-4.0.8	230/294(78%)	14,559/14,976(97%)
tar-1.13	205/222(92%)	10,194/10,800(94%)
wget-1.9	346/434 (80%)	15,249/16,544 (92%)
bison-1.875	410/832(49%)	12,558/18,110(69%)
proftpd-1.3.1	940/1,096(85%)	35,386/41,062(86%)
apache-2.2.2	1,364/2,075(66%)	71,719/95,179(75%)

-Nearly entire program should be re-analyzed due to small changes inside the cycle.

• Larger cycles require more repetitions during analysis.



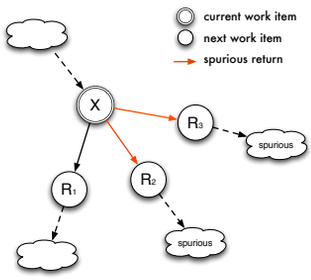
spell-1.0
(2,213 LOC, > 30 repetitions)

sed-4.0.8
(26,807 LOC, > 150 repetitions)

-As programs are getting larger, the inefficiency due to such cycles is getting more significant.

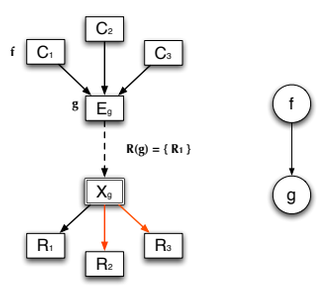
3. Solution

- Idea



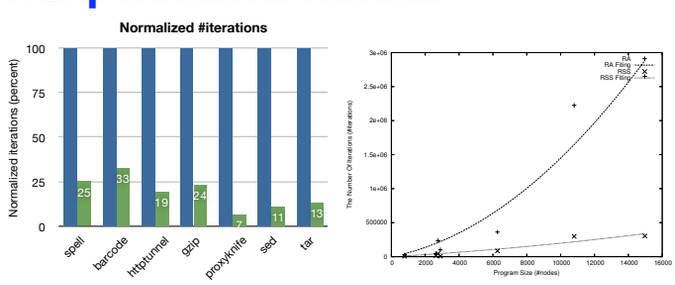
During worklist algorithm, spurious executions are identified and pruned. Two issues:
• How to detect spurious executions?
• Is it sound? (Some executions are pruned)

- How to detect spurious executions?



• For each call, its single return site is remembered, and the callee returns to only that remembered site.
• Procedure calls are handled in a mutual exclusive manner: each procedure calls lock the callee and the others wait until the lock is released.
• Recursion handling

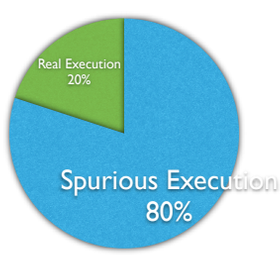
4. Experimental Results



• The amount of computation has been decreased by, on average, 81.15%.
• Analysis complexity becomes nearly linear.

Conclusion

• Abstract interpretation (or data flow) analysis performance problems due to spurious executions are critical.
• Spurious interprocedural cycles mainly contribute to spurious executions.
• Static analysis can be much more efficient by avoiding them.



Programming Research Laboratory http://ropas.snu.ac.kr