LLVM 컴파일러의 최적화 단계에 대한 검증된 검산기

최준원, 조성근, 강지훈, 허충길 교수님, 이광근 교수님

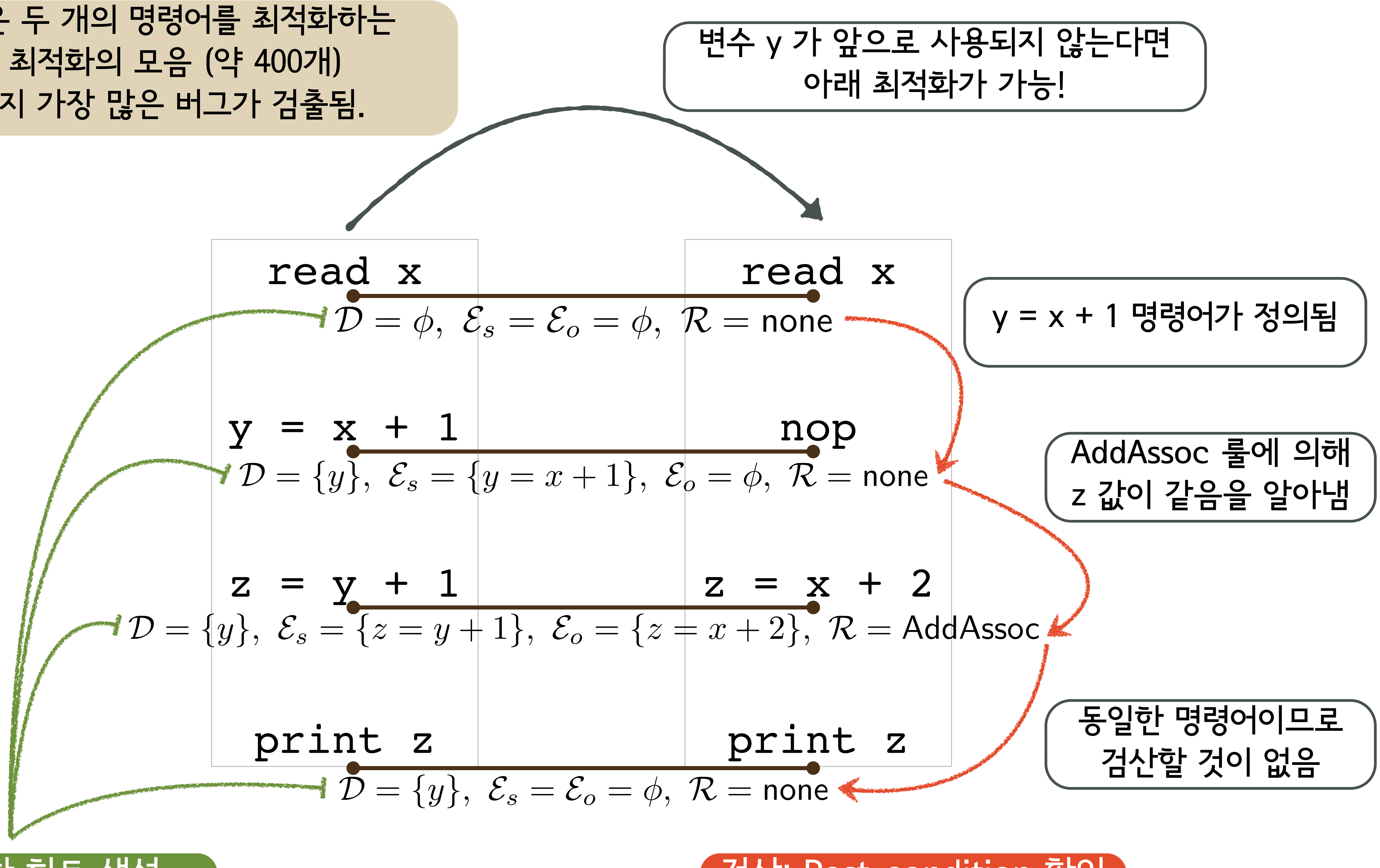
효과적인 컴파일러 힌트로 빠르게 컴파일러 검산기 만들기

1) 컴파일러 결과 검산 연구는 오랫동안 진행되어 왔다. (GCC, CompCert 등)
2) 하지만 검산기 구현에 걸리는 시간 + 검산기 동작 시간 모두 너무 오래 걸린다.
3) 효과적인 컴파일러 힌트로 빠르게 검산기를 만들고 동작 시간도 빠르게 하자!

InstCombine 검산하기 : 힌트 생성 및 검산

InstCombine 최적화 단계?

- 하나 혹은 두 개의 명령어를 최적화하는 간단한 최적화의 모음 (약 400개)
- 지금까지 가장 많은 버그가 검출됨.



검산을 위한 힌트 생성

- 힌트는 각 프로그램 라인마다 생성된다.
- 힌트는 세 가지로 구성
  * MayDiff(D) : 값이 다를 수 있는 변수
  * Equations(E) : 변수 사이에 성립하는 등식
  * InfRule(R) : 어떤 최적화를 사용했는가

검산: Post-condition 확인

- 현재 지점 직전까지의 조건(pre-condition)과 현재 명령어를 조합하여 힌트로 제시된 post-condition을 만족할 수 있는지 확인.

최종 목표: Simulation 증명

$$\forall s_1.\forall s_2.\ (s_1, s_2) \in R \Leftrightarrow \begin{cases} \forall s_2'.\forall e \in E.s_2 \xrightarrow{e} s_2' \Rightarrow \\ s_1 \rightarrow \texttt{error} \vee \\ \exists s_1'.s_1 \xrightarrow{e} s_1' \wedge (s_1', s_2') \in R \end{cases}$$

예제에서는 “print z” 의 행동이 같은지 증명하면 됨. MayDiff 에 z가 없으므로 z값이 같음. 따라서 증명 가능!

- 최적화된 프로그램이 원래 프로그램의 의미를 따른다는 것을 증명하고 싶다.
- 의미 : observational behaviors 가 같다.
(read/write 와 같은 system call 에서 같은 행동을 한다)

ROSAEC center
Research On Software Analysis for Error-free Computing
소프트웨어 무결점 연구센터 NRF ERC