고려대학교 프로그래밍 연구실
오학주 교수님

고려대학교 전민석

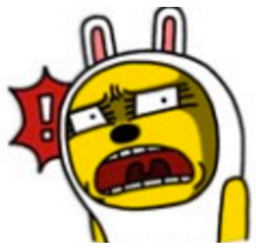
불완전한 프로그램을 완전한 프로그램으로 합성하기

1. 연구 동기

- 기존 합성기들의 문제점
  - 합성가능한 프로그램의 범위가 적다.
    (예: 단 하나의 루프로 변환가능한 재귀함수)

  - 명세 입력이 어렵다. (프로그램인지 명세인지?)

```
def merge(in1: List, in2: List) = {
  require(isSorted(in1) && isSorted(in2))
  choose { (out : List) ⇒ isSorted(out) &&
      (content(out) == content(in1) ++ content(in2)) } }
```



2. 목표

- 사용자가 명세를 **쉽게** 표현하고
- **충분하게** 표현하면서
- **더 광범위한** 프로그램을 합성할 수 있는
  (중첩 반복문 이상으로 복잡한)

명령형 프로그램 합성기를 만들자!

3. 문제 정의

불완전한 프로그램을 주어진 예제를 만족하는 완전한 프로그램으로 합성하기

Q. 입력된 정수를 뒤집어 주는 프로그램을 만드시오.

입출력 예제: 1->1, 12->21, 123->321

사용될 숫자와 변수: int: 1,10 / var: n,r,x (실제보다 과추정 가능)

불완전한 프로그램:

```
fun n->
r = 0;
while (n>0){
      ?
}
return r;
```

1.9 sec

완전한 프로그램:

```
fun n->
r = 0;
while (n>0){
  r = r * 10;
  x = n % 10;
  r = r + x;
  n = n/10;
}
return r;
```

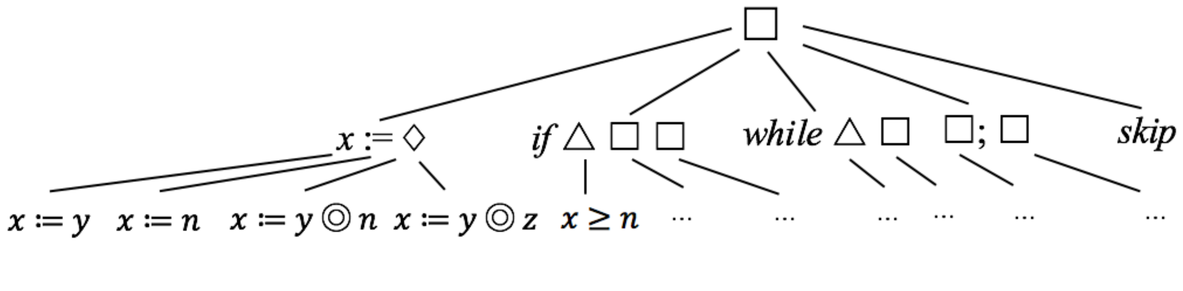
4. 합성 알고리즘

1) 대상 언어

$$
\begin{array}{rcl}
lv & \rightarrow & x \mid x[a] \\
a & \rightarrow & n \mid lv \mid a_1 \oplus a_2 \mid x.size \mid \ ? \\
b & \rightarrow & a_1 \geq a_2 \mid a_1 = a_2 \mid \neg b \mid \ ? \\
c & \rightarrow & skip \mid lv := a \mid c_1; c_2 \mid if\ b\ c_1\ c_2 \mid while\ b\ c \mid \ ? \\
p & \rightarrow & fun\ x\ \{c; return\ a\}
\end{array}
$$

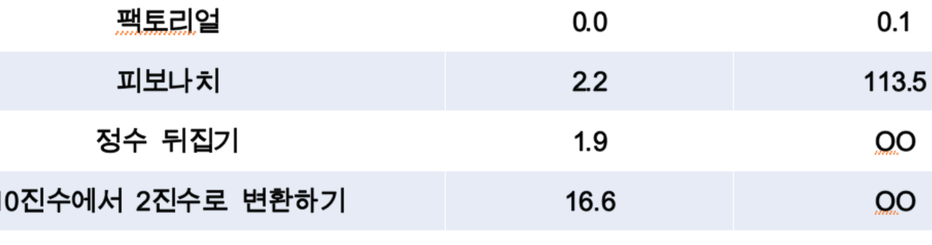
2) 기본 알고리즘

- 언어 문법에 따라 모든 상태공간 탐색
- 간단한 프로그램 우선 탐색(Best-first Enumerative search)



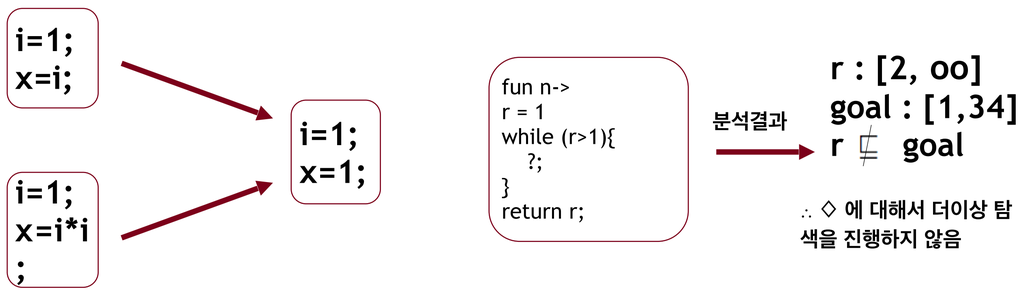
3) 해결방법 **요약해석에 기반**한 효과적인 탐색 전략

A. 도달 불가능한 명령문 (Unreachable command) 정규화



B. 변수의 상수화 (Constant folding)

C. 해를 가질수 없는 상태 가지치기



5. 실험

- **정수 영역**을 다루는 문제들
- **대학교 1학년 프로그래밍 연습문제**를 대상으로
- 기본 알고리즘과 **해결 기법**을 적용하여 비교
- 각 벤치마크별로 학생들이 **실수할 수 있는 부분**을 채워넣는 시나리오를 구성.

문제	가지치기 적용(초)	기본 알고리즘(초)
팩토리얼	0.0	0.1
피보나치	2.2	113.5
정수 뒤집기	1.9	OO
10진수에서 2진수로 변환하기	16.6	OO
배열에서 각 원소를 제곱한 것들의 합	0.5	1.8

Time out > 5 min

실험 환경 MacBook Pro / OS X El Capitan 10.11.1 / 2.2 GHz Intel Core i7 / 16GB 1600 MHz DDR3