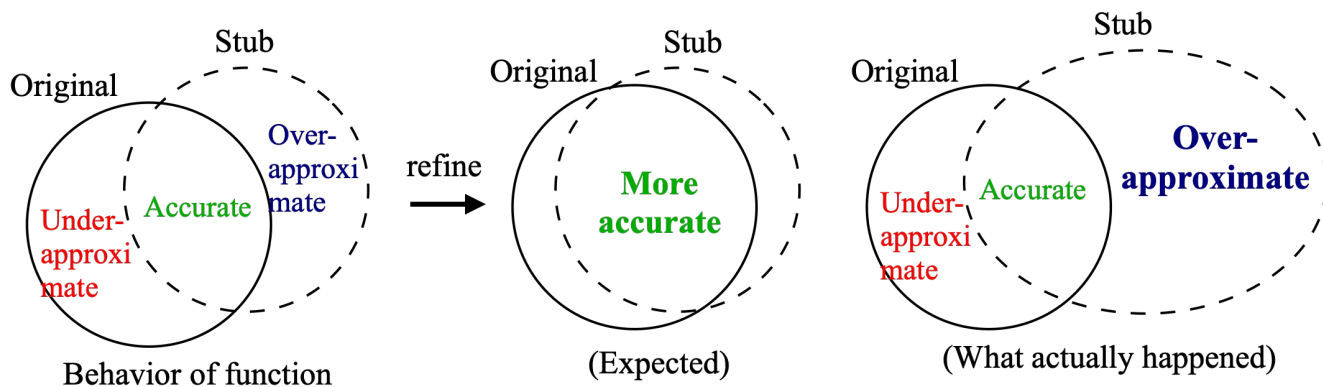


PBE 기반 스텝 자동 생성 및 스텝 코드 분할을 통한 모델 검증 효율 향상

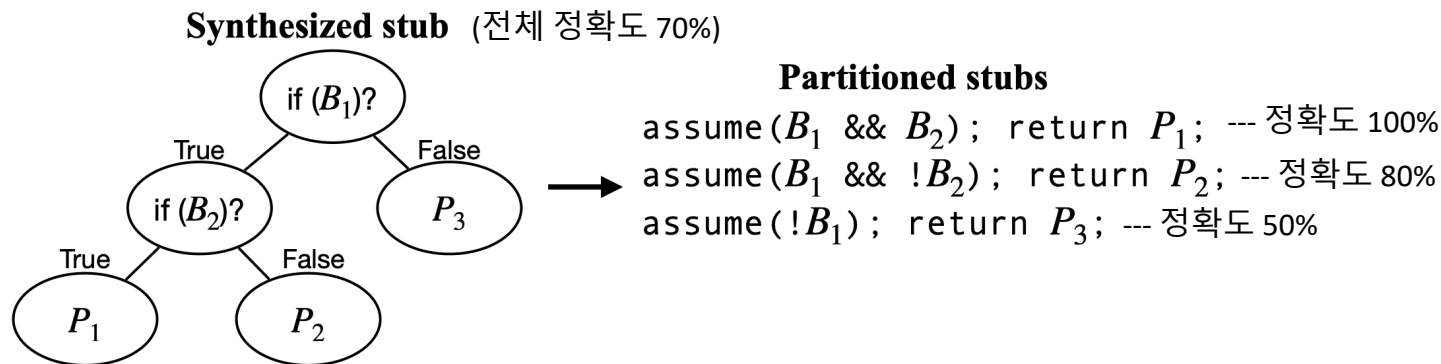
김요엘, 최윤자

연구 개요

- **Goal:** 원본 함수보다 비교적 정확하면서 더 간단한 스텝 함수를 PBE를 통해 자동 생성하고, 이를 원본 함수 대신 사용하여 검증 효율성 높이기
- **Problem:** 기존 정제 방식은 새로운 반례가 발생할 때마다 스텝 함수를 재합성하여 진행. 그러나, 반례 수가 부족하여 합성을 통한 일반화 (generalization)가 되지 않고 정제될 때마다 새로운 오탐이 계속 발생
- **Approach:** 스텝을 path에 따라 서브 코드들로 분할하고, 이를 각각 검증하여 반례를 수집한 뒤, 수집된 여러 반례로 한 번에 합성하는 정제 방식 제안

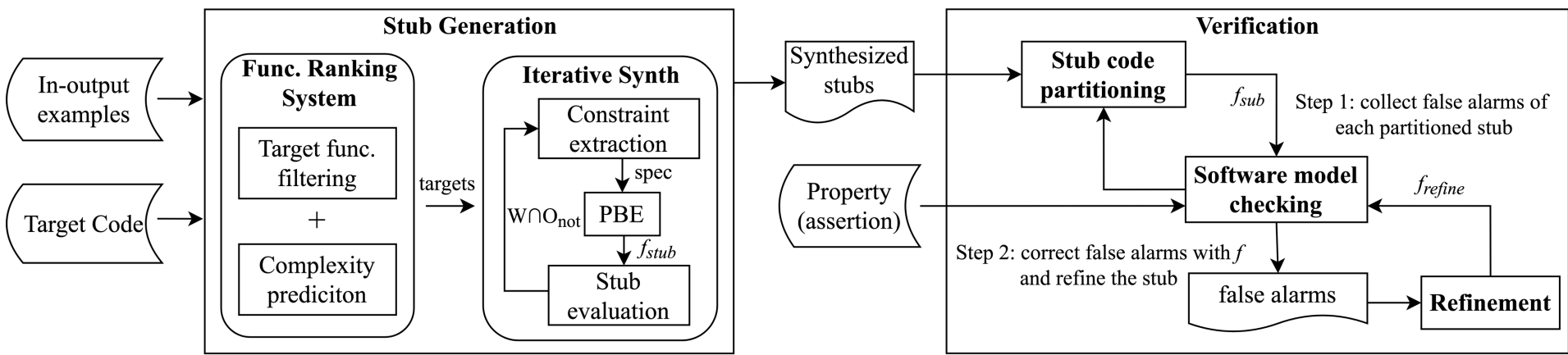


스텝 코드 분할



- 스텝 코드를 분할한 후, **정확도가 높은 순으로** 먼저 검증
- 정확도가 높은 부분을 먼저 탐색함으로써, 진탐 (true alarm)을 빠르게 찾아낼 수 있음
- 정확도가 낮고 복잡한 전체 스텝 코드를 사용하는 것 보다, 정확도가 더 높고 분할된 스텝 코드를 사용하는 것이 검증 효율 면에서 더 좋음

전체 프로세스



실험

Program	Property	Bound	Using stub partitioning					Without partitioning					Only CBMC	
			# Iter.	Verif. Time(s)	Refine Time(s)	Total(s)	Result	# Iter.	Verif. Time(s)	Refine Time(s)	Total(s)	Result	Verif. Time(s)	Result
object follower	p1	50	42	23722.1	0.0	23722.1	satisfied	5	2480.5	0.0	2480.5	satisfied	575.6	satisfied
	p2	105	1	45960.9	0.0	45965.6	violated	1	>24h	0.0	>24h	timeout	>24h	timeout
	p3	55	1	1011.3	0.0	1014.1	violated	1	1716.2	0.0	1718.5	violated	7519.1	violated
	p4	65	2	4152.7	0.0	4158.4	violated	1	6533.9	0.0	6536.8	violated	16124.3	violated
	p5	55	1	920.2	0.0	922.9	violated	2	4014.9	53.8	4072.7	violated	6283.0	violated
elevator controller	p1	105	1	13092.3	0.0	13092.3	violated	2	25518.8	17.1	25573.9	violated	14907.6	violated
	p2	105	1	7749.4	0.0	7749.4	violated	2	16441.2	16.6	16490.3	violated	11435.5	violated
gcd 1+newton 1 6		2	1	12.9	0.0	12.9	violated	25	17.8	22.6	55.4	violated	3.2	violated
gcd 3+newton 1 4		2	1	6.6	0.0	6.6	violated	36	17.2	34.4	73.1	violated	3.5	violated
pals_lcr.5.ufo.BOUNDED-10.pals+ Problem12_label03		6	1	533.8	0.0	533.8	violated	1	540.3	0.0	540.3	violated	1201.4	violated
pals_lcr.7.1.ufo.BOUNDED-14.pals		17	5	1.0	44.2	47.4	violated	3	1.0	48.4	50.9	violated	0.3	violated
pals_lcr.8.1.ufo.BOUNDED-16.pals		17	6	1.7	46.9	50.5	violated	3	0.9	24.2	26.0	violated	0.4	violated
token_ring.05.cil-2		6	10	8.1	0.0	8.1	satisfied	6	3.8	300.0	304.7	violated	1.4	violated

refinement을 하지 않고 더 빠르게 진탐을 찾을 수 있음

진행 중인 연구

- 스텝 코드를 직접 refinement하는 대신, 스텝 코드의 over-approximate 부분을 block하는 guarding 함수를 따로 합성하여 적용하는 방안 모색
- 스텝 함수의 잘못된 입출력 예제는 false로, 원본 함수의 교정된 (corrected) 입출력 예제는 true로 설정한 제약 조건을 통해 합성

$$\begin{array}{lcl} f_{stub}(1) = 3 & \rightarrow & (\text{constraint } (= (\text{guard } 1 \ 3) \ \text{false})) \\ f(1) = 1 & \rightarrow & (\text{constraint } (= (\text{guard } 1 \ 1) \ \text{true})) \\ & \rightarrow & y = f_{stub}(x); \\ & & \text{assume}(\text{guard}(x, y)); \end{array}$$
