

동적 심볼릭 매퍼 기반 심볼릭 오토마타 능동학습

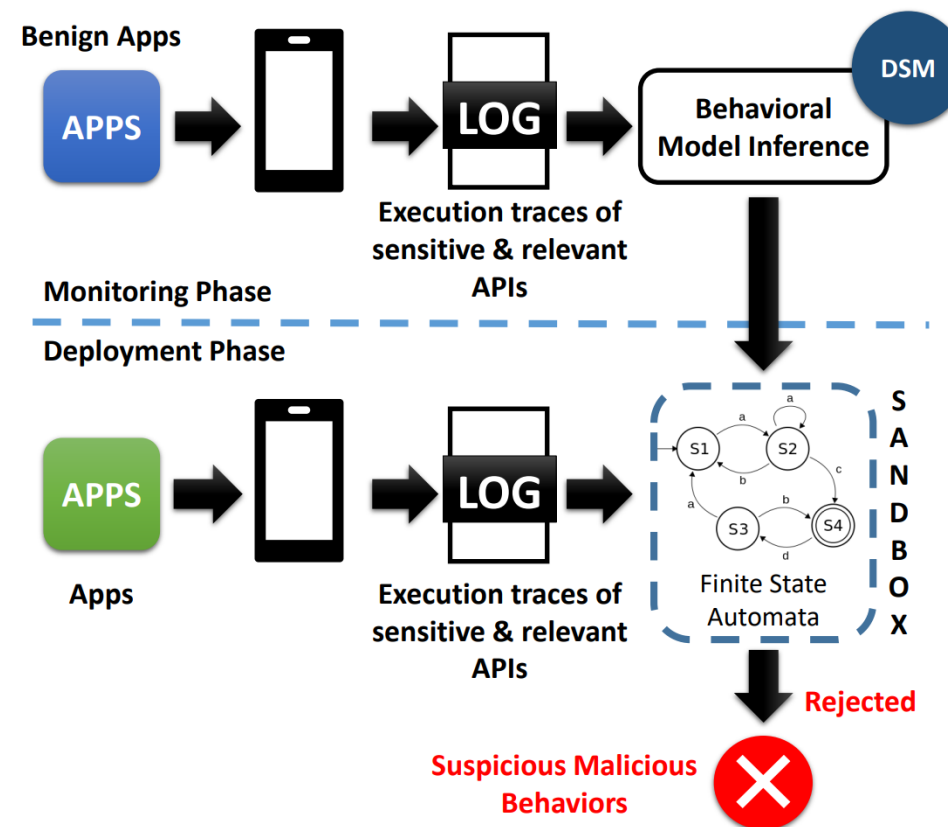
김요엘

소프트웨어안전공학연구실

2026.01.15 BK 학술교류회

동기: 오토마타는 왜 필요한가

- 오토마타: 시스템의 복잡한 행동을 상태와 전이로 요약한 행동 모델
- 오토마타의 활용:
 - API 오남용 탐지 [FSE'18]
 - 네트워크 모델 검증 [PLDI'25]
 - 모델 기반 테스트 생성 [ICSE'22]
- 목표: 오토마타를 자동으로, 효율적으로 학습



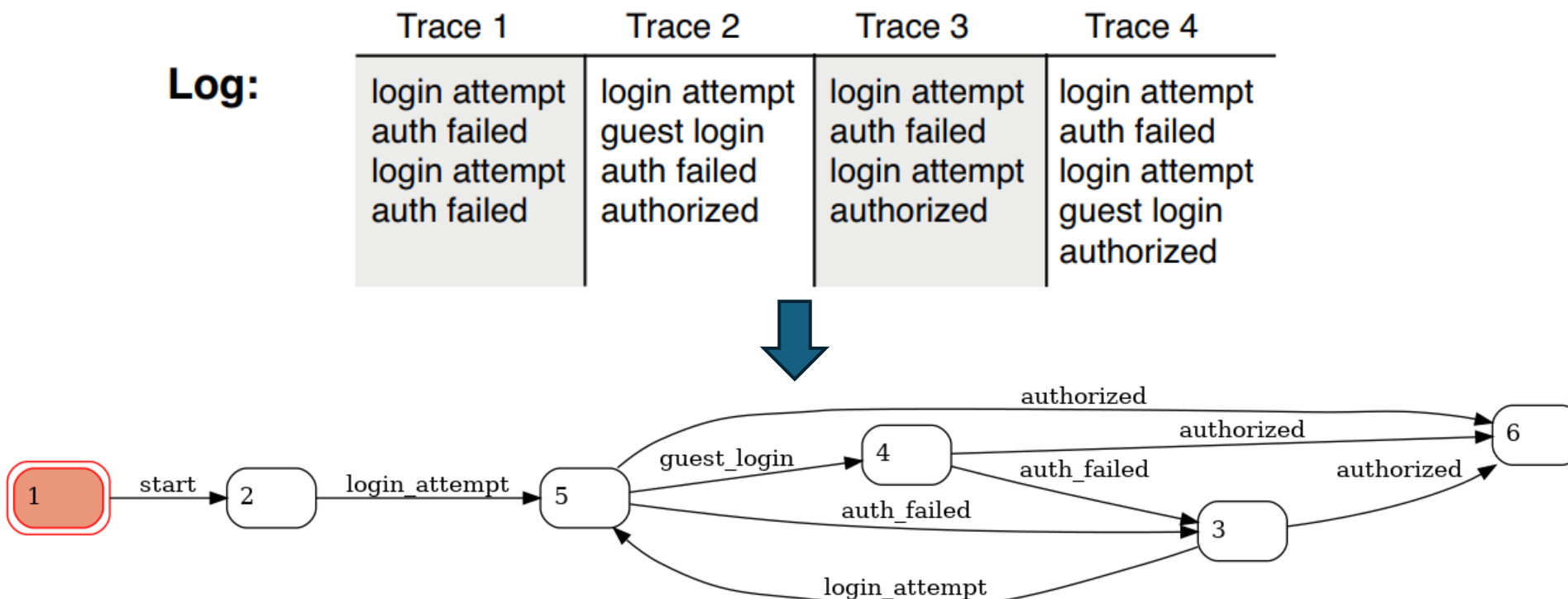
[1] Deep Specification Mining, FSE 2018

[2] Active Learning of Symbolic NetKAT Automata, PLDI 2025

[3] MOREST: Model-based RESTful API Testing with Execution Feedback, ICSE 2022

배경: 기존의 오토마타 학습

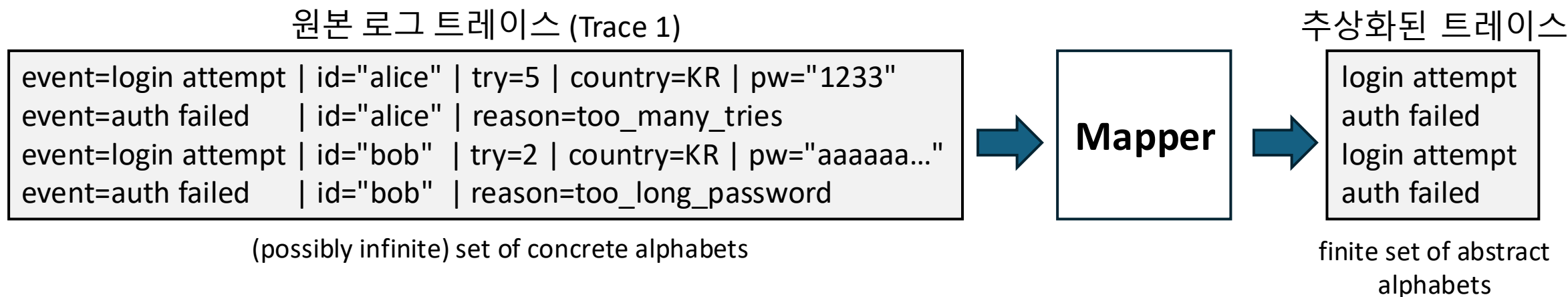
- 시스템의 행동 = 시스템을 실행해서 얻은 로그 트레이스



- 의문: 과연 실제 시스템의 로그 메시지가 이렇게 단순할까?

복잡한 로그의 추상화: 매퍼

- 같은 이벤트도 서로 다른 파라미터(id/try/country/pw ...) 때문에 종류가 매우 다양함
- 이 때문에, 오토마타 학습을 위해선 **매퍼**가 사실상 필수



- 그러나, 매퍼는 도메인마다 사용자가 수작업으로 구현해야 함

대안: 심볼릭 오토마타 학습

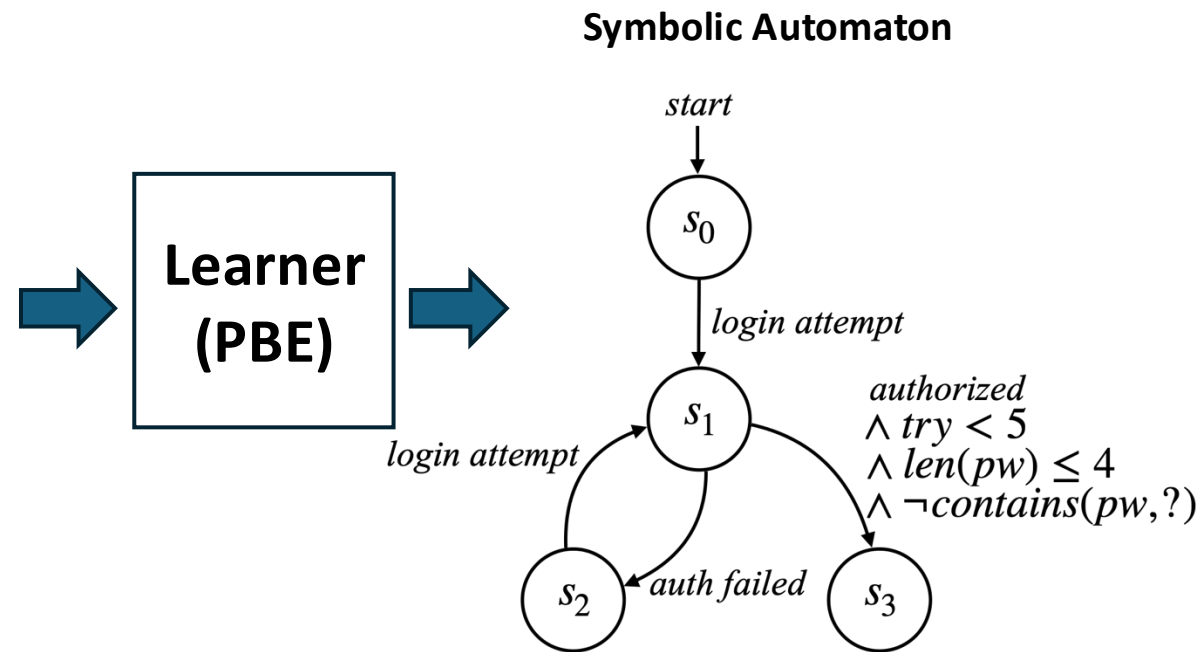
- 심볼릭 오토마타: 로그의 파라미터 값을 관계식(predicate)으로 일반화 및 추상화
 - 예제 기반 프로그램 합성 기법(PBE): $(try < 5)$, $(len(pw) \leq 4)$ 와 같은 관계식을 자동 합성

Trace 1

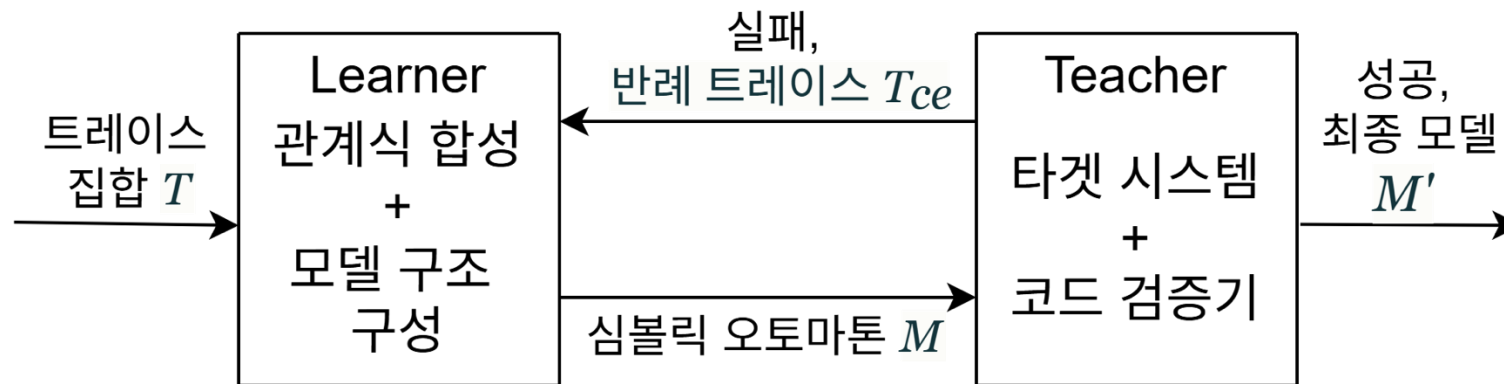
event=login attempt	id="alice"	try=5	country=KR	pw="1233"
event=auth failed	id="alice"	reason=too_many_tries		
event=login attempt	id="bob"	try=2	country=KR	pw="aaaaaa..."
event=auth failed	id="bob"	reason=too_long_password		

Trace 2

event=login attempt	id="alice"	try=1	country=KR	pw="12?4"
event=auth failed	id="alice"	reason=invalid_character		
event=login attempt	id="alice"	try=2	country=KR	pw="1234"
event=authorized	id="alice"			



SOTA: 심볼릭 오토마타 능동학습



• Teaching 알고리즘:

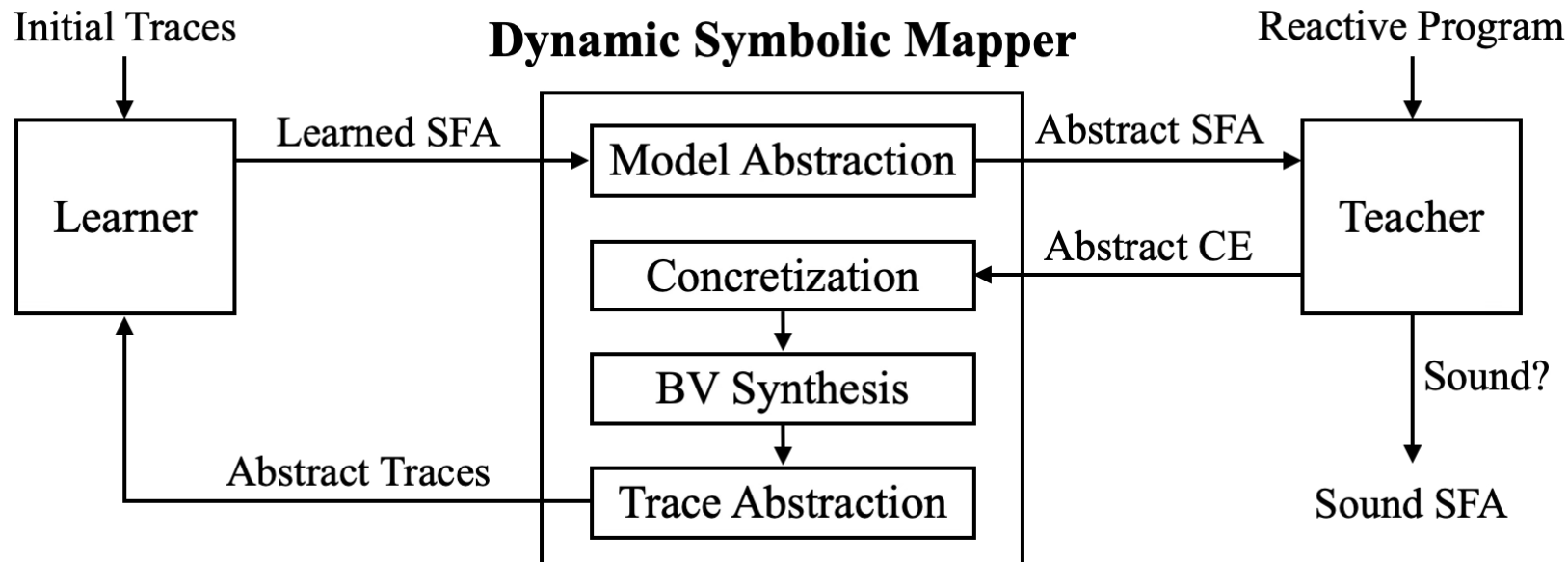
1. 타겟 시스템에는 존재하나, 현재 모델 M 에서는 누락된 행동을 **반례 후보**로 선택
2. **Feasibility checking**: 코드 검증기로 그 후보가 시스템에서 실제로 가능한 실행인지 확인
 - 가짜 반례의 경우: 해당 후보를 배제하고 다른 후보를 탐색
 - 진짜 반례의 경우: Learner에게 전달
3. 더 이상 진짜 반례가 없을 때까지 반복

SOTA의 한계: 두 가지 병목

- Learner의 병목:
 - 매 teaching 이후, 관계식 합성을 반복 (기존 관계식을 버리고 완전히 처음부터 재합성)
 - 파라미터 수/ 값 도메인 및 트레이스 수가 커질 수록 **합성 탐색공간이 급증**
- Teacher의 병목:
 - 반례 후보들에 대해 **feasibility checking**을 반복 수행해야 함
 - 파라미터 수 / 값 도메인이 커질 수록 **가능한 반례 후보 조합이 폭발**
- EX) 엘리베이터 컨트롤러 프로그램:
 - 이벤트: INIT, IDLE, RUN, STOP, EMERGENCY
 - 파라미터: floor (1~4층), timer (0~9), direction (up, down, stay), button[12]
 - 모든 가능한 후보 반례 조합: $5 \times 4 \times 10 \times 3 \times 2^{12} = 2,457,600$ 개

핵심 아이디어 1: 동적 심볼릭 매퍼

- 핵심: 로그 파라미터를 **Boolean Variable (BV)** 벡터로 맵핑
 - BV 예: $b_1 \Leftrightarrow (try < 5)$, $b_2 \Leftrightarrow (len(pw) \leq 4)$
- 왜 Symbolic? 심볼릭 오토마타 학습을 위한 최초의 매퍼
- 왜 Dynamic? 학습 도중 필요에 따라 BV 벡터를 자동 합성/갱신



트레이스 추상화

- 심볼릭 매퍼가 다음과 같은 3개의 BV들을 가지고 있다고 가정

$b_1: (try < 5), b_2: (len(pw) \leq 4), b_3: (contains(pw, ?))$

Trace 1

login attempt	id="alice"	try=5	country=KR	pw="1233"
auth failed	id="alice"	reason=too_many_tries		
login attempt	id="bob"	try=2	country=KR	pw="aaaaaa..."
auth failed	id="bob"	reason=too_long_password		

Trace 2

login attempt	id="alice"	try=1	country=KR	pw="12?4"
auth failed	id="alice"	reason=invalid_character		
login attempt	id="alice"	try=2	country=KR	pw="1234"
authorized	id="alice"			

**Symbolic
Mapper**

Abstract Trace 1

login attempt	b1=false	b2=true	b3=false
auth failed			
login attempt	b1=true	b2=false	b3=false
auth failed			

Abstract Trace 2

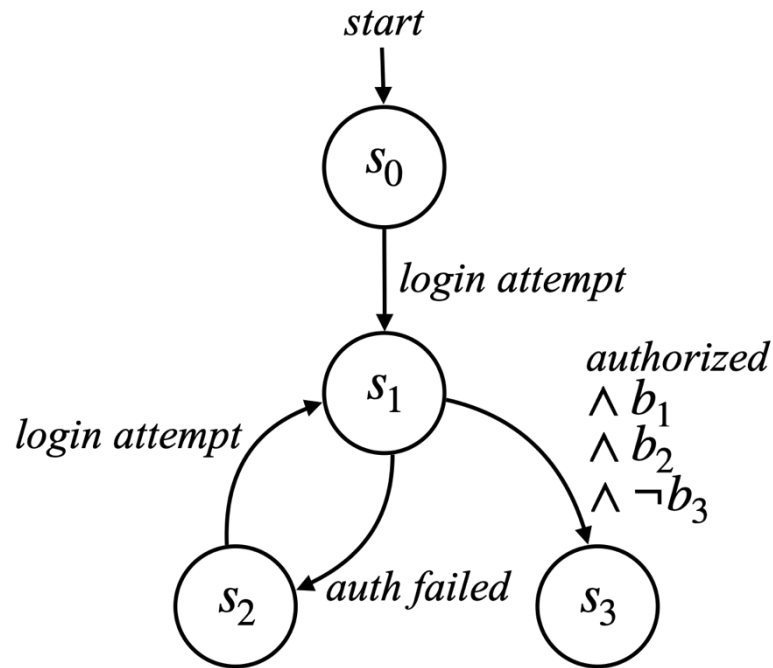
login attempt	b1=true	b2=true	b3=true
auth failed			
login attempt	b1=true	b2=true	b3=false
authorized			

Learner 입장: BV들 간의 논리 연산 (예: and, or, not)만으로도 관계식 합성 가능

모델 추상화 및 추상 반례 구체화

- 모델 추상화: 심볼릭 오토마타 내의 관계식들을 BV로 치환
- 반례 후보 조합 검사 비용 > 추상 반례 후보 조합 검사 + 추상 반례 구체화 비용

Abstract Symbolic Automaton



login attempt에서 authorized로
전이하는 추상 반례 후보 조합

b1 = false, b2 = true, b3 = false
b1 = true, b2 = false, b3 = false
b1 = true, b2 = true, b3 = true
b1 = false, b2 = false, b3 = false
b1 = false, b2 = true, b3 = true
b1 = true, b2 = false, b3 = true
b1 = false, b2 = false, b3 = true

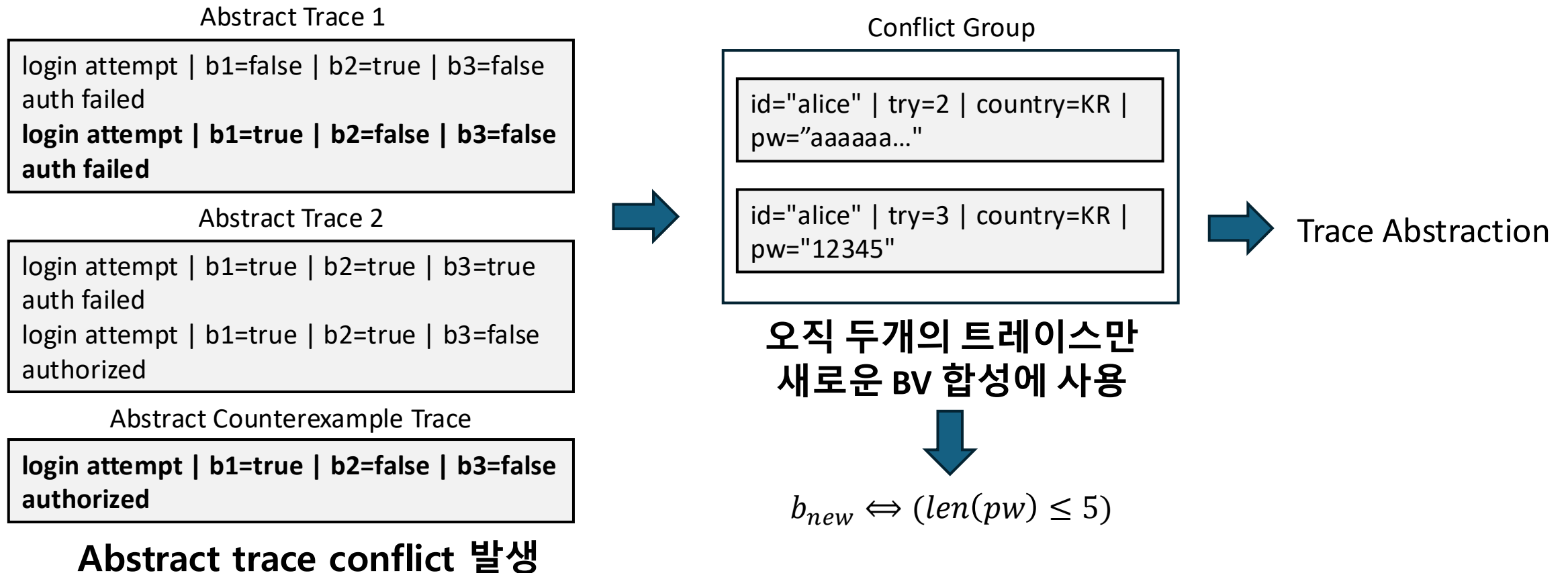
추상 반례 ($b_1 \wedge \neg b_2 \wedge \neg b_3$)
구체화(Concretization)

id="alice" | try=3 | country=KR | pw="12345"

Teacher 입장: Boolean 벡터 조합만 탐색하면 됨
(예: BV가 n개면 최대 2^n 개의 추상 반례 후보)

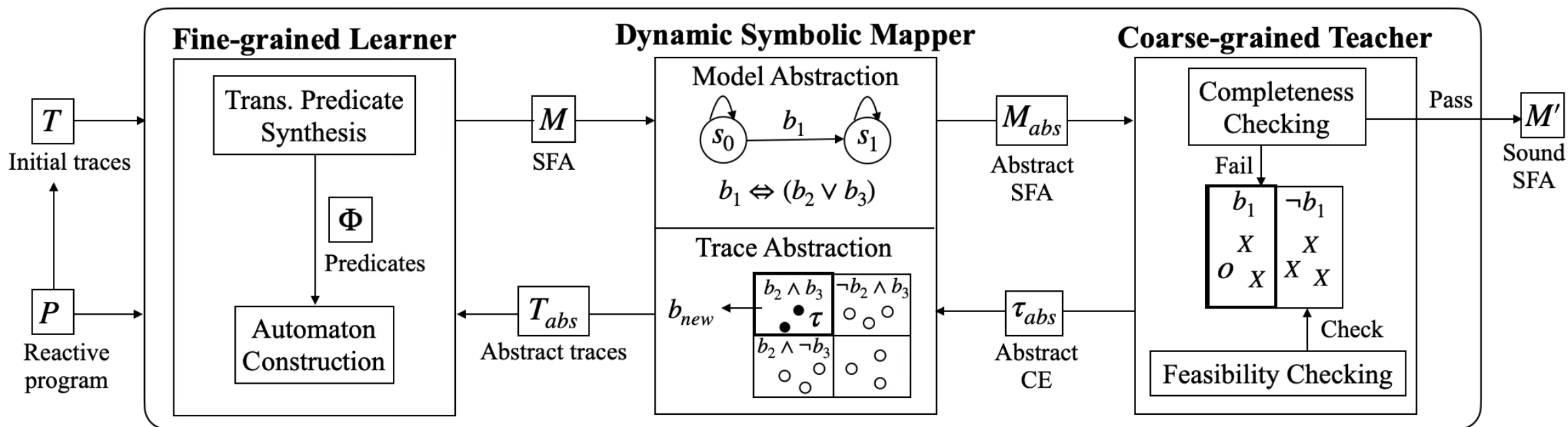
BV 벡터 정제

- Abstract trace conflict: 같은 Boolean 벡터 값을 가지나, 서로 다른 다음 이벤트를 가짐
- 원본 트레이스를 통한 학습 비용 > 새로운 BV 합성 비용 + 추상 트레이스를 통한 학습 비용



핵심 아이디어 2: 세분도 교대 전략

- 세분도(granularity): 관계식이 쪼개진 정도
 - Teacher 입장: BV 수가 적으면 → 고려해야 하는 추상 반례 후보 조합도 적어짐
 - Learner 입장: BV 수가 많으면 → 새로운 BV 합성에 필요한 트레이스 수가 적어짐

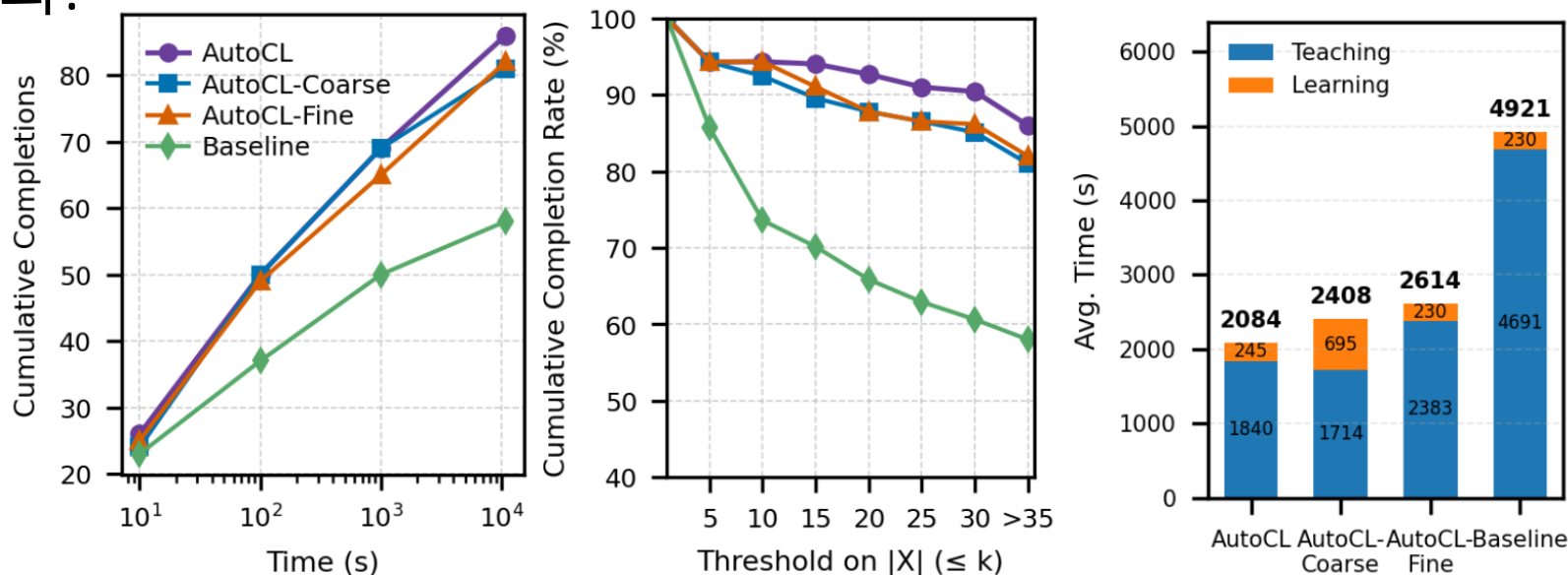


실험 및 평가

- 벤치마크: 총 100개 예제

Benchmark	$ P $	# Nest	LoC	$ Dom(t) $	$ X $	# Bool	# Enum	# Num
Simulink	45	1	250	3.7	6.5	1.2	3.9	1.4
SV-COMP	25	3	1,521	4.8	34.8	1.6	10.8	22.4
LeetCode	25	18	186	5.2	25.9	4.3	1.5	20.1
Embedded	5	4	828	7.4	11.2	0.6	5.0	5.6
Avg.			581	4.5	18.7	2.0	5.1	11.6

- 실험 결과:



진행 중인 작업 (FSE'26 Revision)

1. Embedded 벤치마크 추가 실험
 - 현재 인공위성 탑재 프로그램, 드론 제어 프로그램 등으로 추가 실험 중
2. 정적 심볼릭 매퍼 vs. 동적 심볼릭 매퍼
 - 현재 정적 코드 분석을 통해 BV들을 학습 초기에 제공하는 기능 구현 완료
3. 처음부터 끝까지 동작하는 running / motivating example 그림 그리기
4. 알고리즘 soundness 증명
5. Related work 보강 등...

질문: 방금 발표에서 가장 이해하기 어려웠던 부분은 무엇인가요?

발표/슬라이드에 주고 싶은 피드백이 있다면?