Esterel의 선언형 확장

황준형 윤정한 한태숙
한국과학기술원 프로그래밍언어연구실
2010년 10월 21일

차례

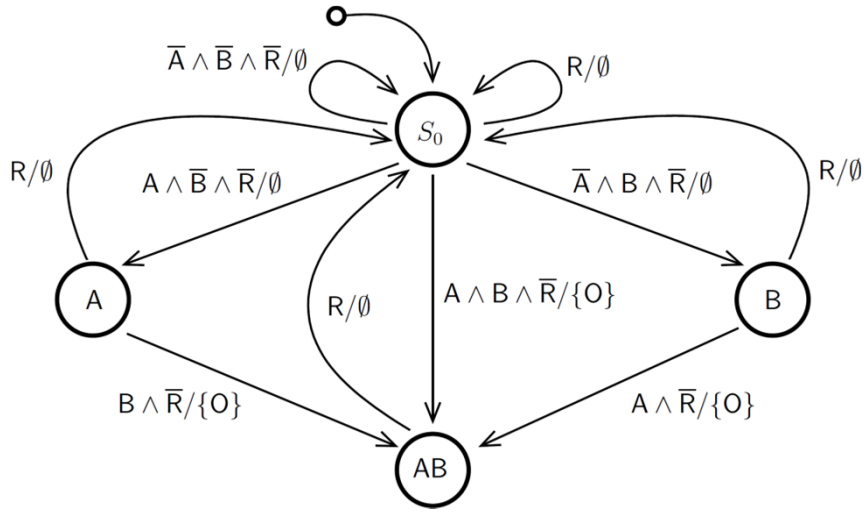
- Esterel 언어 소개
- Lustre 언어 소개
- Esterel 언어의 선언형 확장 제안
- 확장한 문법을 기존 개발 환경에 적용하는 방법

Esterel 언어

- 동기식 synchronous
  - 실행을 단위 시간으로 끊어서 생각
- 반응형 reactive
  - 외부에서 주어진 자극에 대해 정해진 시간 안에 반응
  - 대부분의 반응형 시스템은 결정적 deterministic
- 명령형 imperative
  - 시스템의 실행은 상태 변화를 통해 이루어짐

Esterel 언어로 작성한 프로그램의 예

```
module ABRO
input A, B, R ;
output O ;
loop
 [ await A || await B ];
 emit O
each R
end module
```



Esterel 언어의 장점

- 병렬 수행이나 선점 구조를 편리하게 나타냄
- 하드웨어와 소프트웨어를 모두 지원
- 엄밀하게 정의된 언어

Esterel 언어의 약점

- 신호들 사이의 관계를 나타내기 불편함
  - `relation` 구문은 실행에 적용할 수 없음
  - 단위 시간마다 반복해서 신호를 바꾸어야 함
- 신호들 사이의 관계는 자주 쓰임
  - 내장형 시스템은 상태를 바꾸면서 동작함
  - 한 상태에서는 일정한 방식으로 동작할 때가 많음

선언형 언어 Lustre

- 신호의 값을 등식equation으로 표현
- 등식은 `let...tel` 안에서 사용
- 치환 원리 substitution principle
- 정의 원리 definition principle

Lustre 언어로 작성한 프로그램의 예

```
node COUNT (init, incr: int; reset: bool)
 returns (n: int);
let
 n = init -> if reset then init
                      else pre(n) + incr;
tel
```

Lustre의 연산자

- pre(S)
- R -> S
- R when S
- current(S)

Esterel의 선언형 확장

- Esterel의 문법에 `let` 구문을 추가
  - Esterel의 `signal` 구문
  - Lustre의 `let` 구문

확장한 문법을 기존 개발 환경에 적용하기

- 확장한 문법으로 나타낸 부분을 기존 문법으로 표현
  - 인과 관계 확인 (순환 논리 제거)
  - 각 신호의 시계가 올바른지 확인
- 각 식에 해당하는 코드를 병렬 수행
- 복잡한 식을 변환할 때에는 일부에 해당하는 신호들을 만들고 이들을 이용해서 전체 식에 해당하는 신호를 생성

변환 사례

- *A* = pre(pre(*S*))

```
loop
 present S then pause; emit preS else pause end
end
||
loop
 present preS then pause; emit A else pause end
end
```

향후 과제

- 실제로 구현하고 예상한 결과를 얻는지 확인
- 프로그램 분석을 이용한 코드 생성 최적화

고맙습니다

- 질문 부탁드립니다