고급 소프트웨어 공학
Coffee Vending Machine

200973359 최 정 한
jhchoi@db.konkuk.ac.kr
200973348 양 형 식
hsyang@db.konkuk.ac.kr
Database Lab. In Konkuk Univ.

차례

- Coffee Vending Machine 설계
- 정의된 Automata
- 각 Automata 에 대한 설명
- 검증 사례
- 검증 확인

Coffee Vending Machine 설계(1/2)

❖ Coffee Vending Machine

❖ 현금(지폐 & 동전)
- ◆ 현금으로만 이용이 가능
- ◆ 지폐는 1000원, 동전은 10,50,100,500원 사용 가능
- ◆ 현금 입력이 없을 시 버튼을 선택하여도 음료가 나오지 않음
- ◆ 버튼 입력이 10초 동안 없을 시 반환되고 초기상태로 됨
- ◆ 현금 잔액은 5000원을 넘어갈 수 없음

❖ 반환 버튼
- ◆ 반환버튼을 누를 경우 현재 잔액이 반환
- ◆ 반환된 후에는 초기 상태로 되돌아 감

❖ 선택 버튼
- ◆ 음료는 커피, 우유, 핫쵸코 3종류
- ◆ 음료 선택 버튼은 동시에 누를 수 없음
- ◆ 현재 금액에서 차액을 제외한 현 금액이 표시
- ◆ 현재 선택한 음료가 5초 이내에 나옴 (5초간 음료 배출구의 램프가 점등)

Coffee Vending Machine 설계(2/2)

❖ Coffee Vending Machine

❖ 음료 배출구
- ◆ 음료를 만드는 중에는 다른 버튼을 누를 수 없음
- ◆ 음료를 만드는 중에는 현금을 입력할 수 없음
- ◆ 음료가 배출 완료 시에는 현금 입력, 반환이 활성화

❖ 동전 배출구
- ◆ 동전 반환 버튼 누를 시 현재 금액이 반환

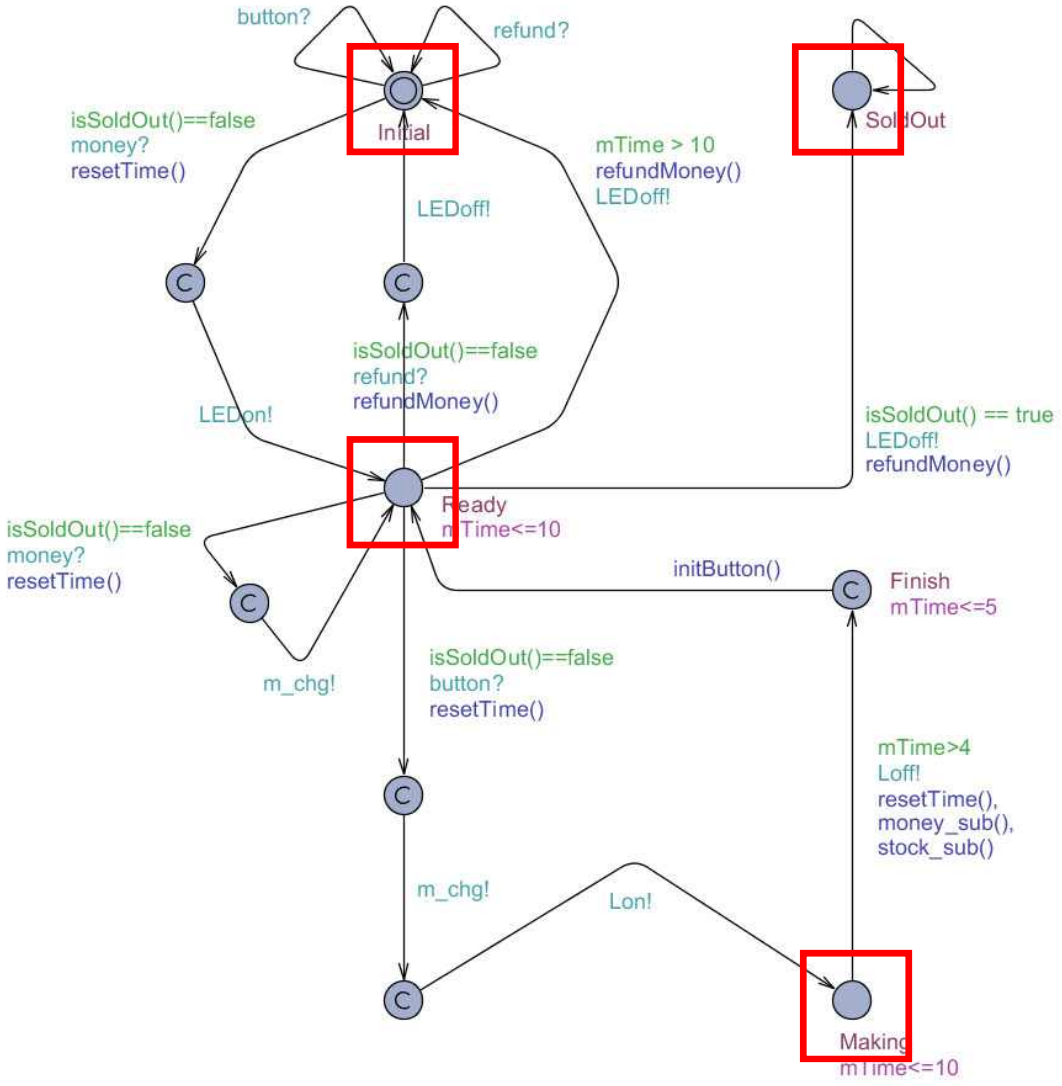
❖ 추가사항
- ◆ 음료제작에 필요한 재료가 떨어지면 떨어진 음료는 선택할 수 없음
- ◆ 음료제작에 필요한 재료가 모두 떨어지면 판매가 중지됨
- ◆ 10원짜리 잔돈이 떨어지면 판매가 중지됨

정의된 Automata

- Machine
  - Coffee Vending machine
- Money
  - Test 를 위한 Input Data 를 생성하는 오토마타
- Buttons
  - 음료를 선택하거나 반환을 위한 버튼
- Lamp
  - 음료가 만들어지는 동안 on
- LED
  - 금액 표시용 LED

Automata – Machine

- Machine
  - 강아지!

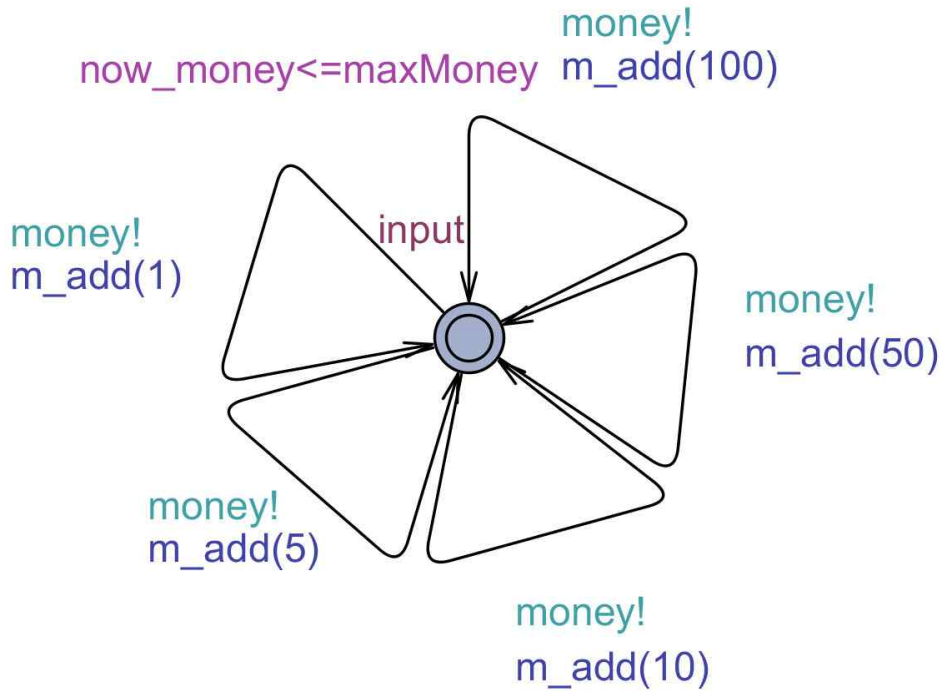


State 설명 - machine

- Initial
  - 초기 상태
- Ready
  - 재고와 잔돈이 충분한 상황에 현금이 들어온 상태
- Making
  - 음료를 만들고 있는 상태
- SoldOut
  - 모든 음료의 재고가 0 이거나 잔돈이 부족한 경우 진입하는 상태

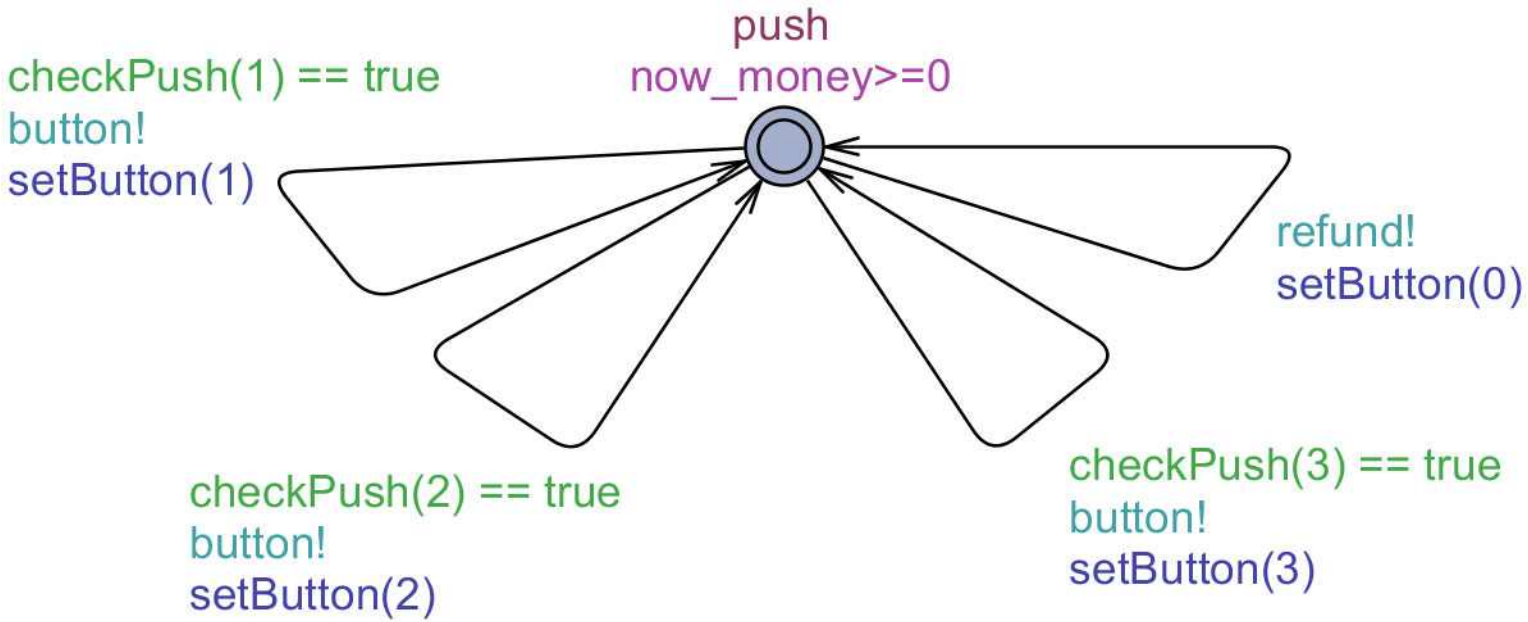
Automata – Money

- Money
  - input
    - 금액을 입력



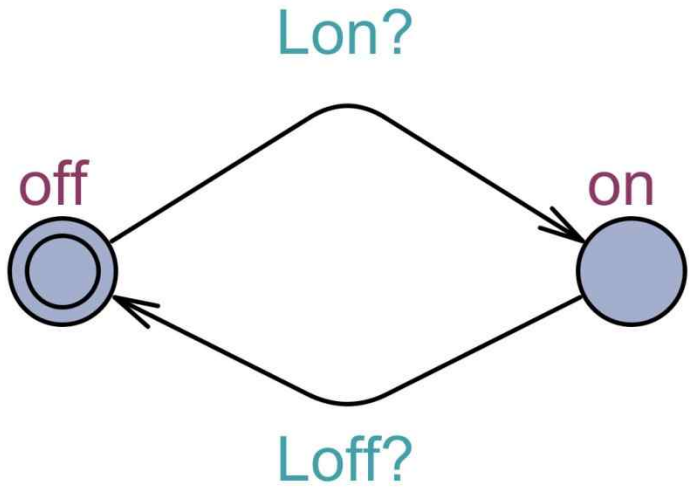
Automata – Buttons

- Buttons
  - push
    - 버튼 입력 (커피, 우유, 핫초코, 반환)



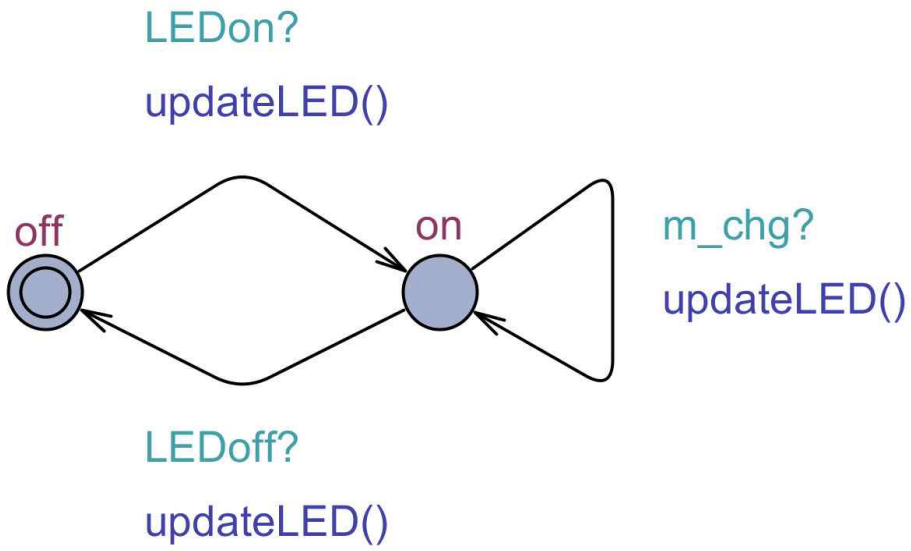
Automata –Lamp

- Lamp
  - On
    - 음료가 만들어지고 있는 상태
  - Off
    - 음료 제작이 끝난 상태



Automata – LED

- LED
  - on
    - Machine 에 돈이 들어온 경우
  - off
    - Machine 에 돈이 없는 경우



검증 사례(1/3)

- Machine.Making --> Machine.Finish
  - 음료를 만들기 시작하면 언제가는 음료가 나온다.
- Machine.Making --> Machine.Finish imply (Machine.mTime > 4 and Machine.mTime <= 5)
  - 음료를 만들기 시작하면 5초 이내에 완성되어야 한다
- A[] Machine.Making imply (now_money >= 350 and Buttons.push)
  - 모든 path 에서 항상 머신상태가 Making 이면 현재 금액은 350원 이상이다.
- A[] not now_money > 5000
  - 모든 path 에서 항상 입력된 돈은 5000원을 넘지 않는다
- A[] Machine.Making imply (Machine.isSoldOut() == false)
  - 모든 path 에서 항상 머신 상태가 Making 이면 머신은 품절 상태가 아님
- A[] Machine.Ready imply (bt == 0)
  - 레디상태이면 그 이전에 버튼상태는 초기화된다

검증 사례(2/3)

- A[] Machine.Ready imply (now_money >= 0 && Machine.mTime <= 10)
  - 머신이 레디상태가 되면 그 이전에 잔액이 0보다 크거나 같고 머신의 시간은 10 이하이다
- Machine.Initial and now_money > 0 --> Machine.Ready
  - 돈이 들어오면 Ready 상태로 간다
- A[] (Machine.Ready or Machine.Initial) imply (Money.input or Buttons.push)
  - Initial 상태와 Ready 상태에서 돈이 들어올 수 있고 버튼이 눌릴 수 있다.
- A[] Machine.Initial imply (now_money == 0)
  - 모든 path 에서 항상 머신이 Initial 상태면 잔액은 0 이다
- A[] Machine.SoldOut imply ((Machine.Ready or Machine.SoldOut) and Machine.isSoldOut() == true)
  - 머신 상태가 품절이면 계속 품절 상태이거나 ready 상태에서 모든 음료 품목의 재고가 0 이다

검증 사례(3/3)

- A[] Machine.Making imply Lamp.on
  - 음료를 제작중이면 음료 배출구 램프가 on 이다
- A[] Machine.Finish imply Lamp.off
  - 음료를 제작이 완료되면 음료 배출구 램프가 off 이다
- A[] (Machine.Initial or Machine.SoldOut) imply not LED.on
  - Initial 상태와 SoldOut 상태에서는 LED 가 on 일 수 없다
- Machine.Ready --> Machine.Initial imply LED.off
  - Machine.Ready 상태에서 Machine.Initial 으로 갈때 LED 는 off 상태이다
- Machine.Ready and Machine.mTime>10 --> Machine.Initial
  - 버튼 입력이 10초 동안 없을 시 반환되고 초기상태로 간다
- A[] not deadlock
  - 데드락 없음

검증 확인

❖ 검증 완료

Overview

Machine.Making --> Machine.Finish
Machine.Making --> Machine.Finish imply (Machine.mTime > 4 and Machine.mTime <= 5)
A[] Machine.Making imply (now_money >= 350 and Buttons.push)
A[] not now_money > 5000
A[] Machine.Making imply (Machine.isSoldOut() == false)
A[] Machine.Ready imply (bt == 0)
A[] Machine.Ready imply (now_money >= 0 && Machine.mTime <= 10)
Machine.Initial and now_money > 0 --> Machine.Ready
A[] (Machine.Ready and Machine.Initial) imply (Money.input and Buttons.push)
A[] Machine.Initial imply (now_money == 0)
A[] Machine.SoldOut imply ((Machine.Ready or Machine.SoldOut) and Machine.isSoldOut() == true)
A[] Machine.Making imply Lamp.on
A[] Machine.Finish imply Lamp.off
A[] (Machine.Initial or Machine.SoldOut) imply not LED.on
Machine.Ready --> Machine.Initial imply LED.off
Machine.Ready and Machine.mTime>10 --> Machine.Initial
A[] not deadlock