18. Formální specifikace programu. Verifikace pomocí metod automatického dokazování a metody model-checking.(A4M33TV

1. Formální verifikace (resp. specifikace) programu

- *Formální verifikace* = technika založená na formálních metodách, stavějící na matematicky založených jazycích, která umožňujé specifikaci a verifikaci systémů, dále jen verifikace
  - *Specifikace* = zapsání požadavků na systém v matematickém jazyce
  - *Verifikace* = formální důkaz toho, že systém splňuje požadavky
- **Princip verifikace**:
  - VSTUPY:
    - *model systému* (matematický); formální model M
    - specifikace požadavků kladených na systém; formule φ temporální logiky
  - PROCES VERIFIKACE:
    - ověření, že systém splňuje specifikaci; rozhodnutí, zda M je modelem formule φ, tj M |= φ
- **Techniky verifikace**:
  - *Statická analýza* = ověření chování programu, aniž by se musel spustit
    - *Abstraktní statická analýza* (např. analýza ukazatelů v modern. kompilátorech)
    - *Verifikace modelů* = úplné procházení dosažitel. stavů programu
    - *Omezená verifikace modelů* = viz. předchozí, ale jen do určité hloubky
  - *Dokazování vět* = nalezení důkazu vlastnosti, kdy systém i jeho vlastnosti jsou vyjádřeny jako formule v nějáké matematické logice

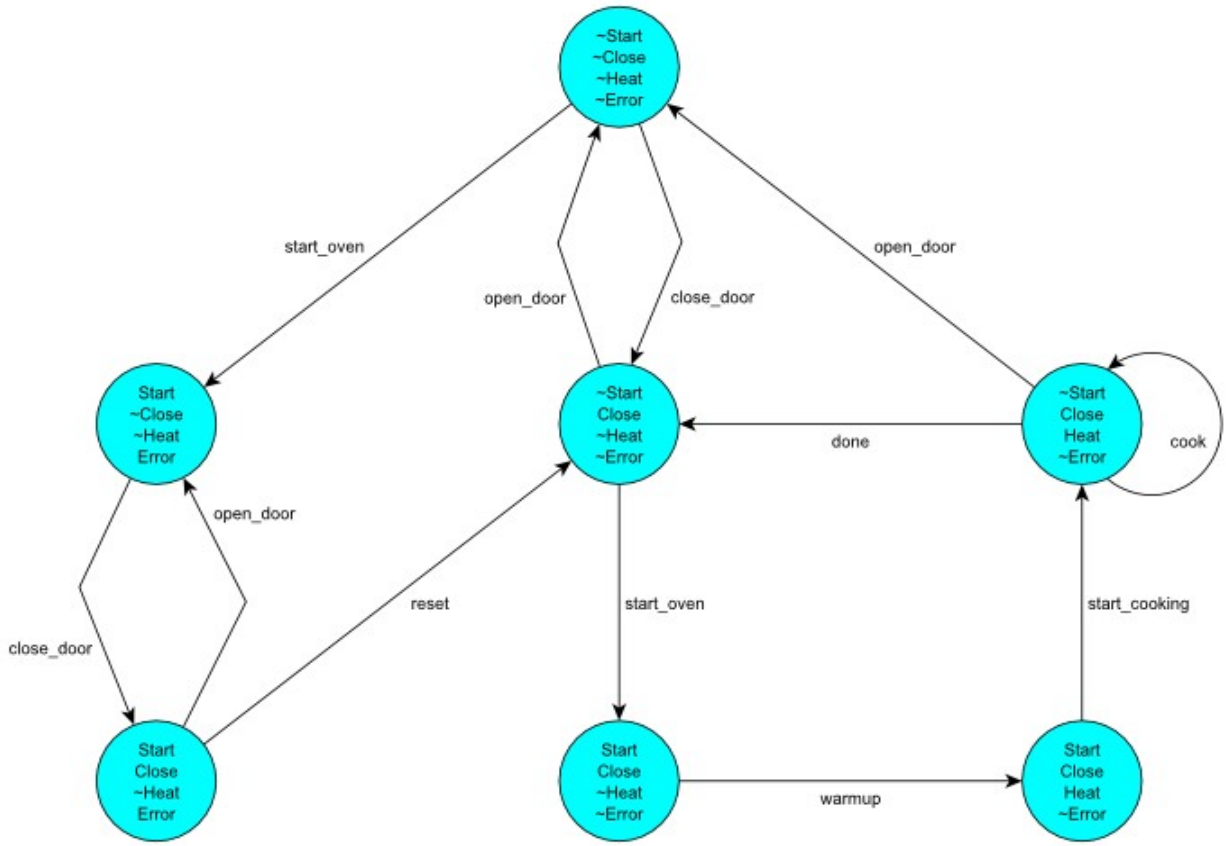
2. Verifikace modelů (model checking)

- **Princip**:
  1. Budování konečného modelu systému
  2. Kontrola, zda požadovaná vlastnost je modelem dodržena
  3. Založeno na úplném prohledávání stavového prostoru
- **Vlastnosti**:
  1. Manipulace s obrovskými prohledávacími prostory
  2. Odpověď je ANO či NE, v záporném případě systém poskytuje příklad, kdy běh systému neodpovídá vlastnosti
- V **praxi** se používá pro ověření HW (obvody), protokolů, analýza specifikace sw systémů
- **Přístupy**:
  - *Temporální verifikace modelů*:
    - Použití *temporální logiky* (vyjádření času)
    - Systémy modelovány jako přechodové systémy s konečným počtem stavů
  - *Automatový přístup*:
    - Specifikace i model vyjádřen jako automaty
    - Oba automaty se porovnávají
- **Výhody model checkingu**:
  - Úplná automatizace

- Vysoká rychlost
- Možnost verifikace i částečných specifikací
- Produkuje protipříklady (při nesplnění)

◆ **Nevýhody**:
- Exploze stavů (je možné zvládnout systémy s 10^{120} stavy)

◆ **Stavový prostor**:
- Formulován pomocí atomických výroků a Kripkeho struktury
- *Atomický výrok* = základní tvrzení popisující daný systém (výrazy, konstanty, predikátové symboly)
  - je algoritmicky rozhodnutelný na základě daného stavu (ohodnocení všech proměnných)
- *Kripkeho strukrura* = typ nedeterministického konečného automatu
  - Mějme množinu atomických pozic AP
  - **Kripkeho struktura** je trojice (S, T, I), kde
    - S = konečná množ. stavů
    - $T \subseteq SxS$ je přechodová relace
    - $I: S \rightarrow 2^{AP}$ je interpretace AP
  - **Rozšířená Kripkeho struktura** je čtveřice (S, T, I, s_0), kde
    - (S, T, I) je Krip. Struktura
    - s_0 je počáteční stav
  - **Kripkeho přechodový systém** je pětice (S, T, I, s_0, L), kde
    - (S, T, I, s_0) je Rozšiř. Krip. Struktura
    - $L: T \rightarrow Act$ je značkovací funkce
    - Mikrovlnka ze slidů přednášky 7:



- **UPPAAL** = nástroj integrující prostředí pro modelování, simulaci a verifikaci realných systémů
  - **Model** = sada nedeterm. procesů s konečnou řídící strukturou a realnými hodinami, komunikace pomocí kanálů nebo sdílených proměnných
  - **Komponenty** systému UPPAAL:
    - Jazyk popisu: jazyk nedeterm. podmíněných příkazů, jednoduché datové typy, sítě automatů s hodinami a datovými proměnnými
    - Simulátor: vyšetřování možných dynamických běhů systému, detekce vad modelu
    - Verifikátor: prověření všech možností dynamického chování modelu, kontrola invariant, dosažitelnost stavů
  - **Dotazovací jazyk**:
    - *Stavové formule* popisující individuální stavy
      - výraz, který lze vyhodnotit pro daný stav
      - nadmnožinou guardů (stráží), povoluje disjunkce
      - **deadlock** = speciální stavová formule, která je splněna pro všechny zablokované stavy (nemající žádný akční přechod či zpožděného následníka)
    - *Běhové formule* vyhodnocující se podél cest a stop modelu
      - **Dosažitelnost**
        - požaduje, zda-li existuje možnost, že daná stavová formule φ je splněna v kadém dosažitelném stavu, tj. existuje cesta z počátečního stavu s_0 taková, že φ bude splněna podél této cesty (má příjemce šanci, že dostane zprávu od vysílače)
        - v UPPAAL E[]φ
      - **Bezpečnost**
        - něco špatného nikdy nenastane (teplata reaktoru ve Fukušimě bude stále pod určitým prahem)
        - v UPPAAL se formuluje pozitivně A[]φ
        - φ by měla být pravdivá ve všech stavech
      - **Živost**
        - něco jednoho dne určitě nastane (stisknu ON a PC se někdy zapne)
        - v UPPAAL A<>φ
        - φ bude vždy jednou splněna
- **Vynecháno**: CTL a LTL logika a detaily jazyka UPPAAL