

# Zastosowanie metod formalnych do weryfikacji struktur wskaźnikowych w systemie PVS

Karol Kozłowski,

Wydział elektryczny, Politechnika Warszawska

5 kwietnia 2025

- 1 Wprowadzenie do metod formalnych
- 2 Wprowadzenie do PVS
- 3 Wyzwania dla struktur wskaźnikowych
- 4 Metodologia pracy
- 5 Studium przypadku: lista
- 6 Wyzwania i ograniczenia
- 7 Zastosowanie metod formalnych: TLA+

# Metody formalne w inżynierii oprogramowania

## Dlaczego metody formalne?

- Krytyczne systemy: medyczne, kosmiczne, transportowe
- Koszt błędów: katastrofy vs koszt wdrożenia
- Przykład: NASA i system PVS

## Kluczowe zalety

- Pełna weryfikacja własności
- Wykrywanie wycieków pamięci
- Gwarancja niezmienników strukturalnych

## System PVS w pigułce

- Logika wyższego rzędu
- Mechanizm dowodzenia twierdzeń
- Automatyczne generowanie TCC (Type Correctness Conditions)
- Parametryzacja teorii

```
pointer_env [P: TYPE, T: TYPE]: THEORY
BEGIN
  pointer: TYPE = P + {nil}
  env: TYPE = [pointer -> (T + {undefined})]
END pointer_env
```

## Problem: Weryfikacja struktur dynamicznych

### Główne wyzwania

- Dynamiczna alokacja pamięci
- Aliasing wskaźników
- Zachowanie niezmienników po operacjach
- Cykliczne struktury danych

### Przykładowa specyfikacja listy

- $\forall l_1 \neq l_2 \Rightarrow \neg \exists n \in (l_1 \cap l_2)$
- $\forall p \in \text{pointer} \Rightarrow \exists ! l : p \in l$

## Proces weryfikacji w PVS

- 1 Modelowanie środowiska wskaźnikowego
- 2 Definicja niezmienników strukturalnych
- 3 Generowanie i dowodzenie TCC
- 4 Specyfikacja operacji (predykaty)
- 5 Dowód zachowania niezmienników

```
list_member?(l: pointer, vl: list): bool =  
  IF nil?(vl) THEN false  
  ELSE l = vl OR list_member?(l, next(vl))  
  ENDIF
```

## Studium przypadku: Lista jednokierunkowa

### Kluczowe niezmienniki

- Spójność typów (TCC)
- Rozłączność list
- Pełna pokrycie pamięci
- Brak cykli

### Przykładowe twierdzenie

```
member_last: LEMMA
FORALL (v1: list):
  NOT nil?(v1) => list_member?(last(v1), v1)
```

## Wyzwania i wnioski

### Główne trudności

- Czasochłonność dowodów (do 1 tygodnia na predykat)
- Ograniczenia PVS w pracy z wieloma teoriami
- Trudności w automatyzacji dla grafów

### Podsumowanie

- Metoda skuteczna dla list i drzew
- Wymaga dużego nakładu pracy
- Obiecujące wyniki dla systemów krytycznych



## Czym jest TLA+?

- TLA+ (Temporal Logic of Actions) - język specyfikacji formalnej opracowany przez Lesliego Lamportą.
- Służy do opisu systemów współbieżnych i rozproszonych.
- System definiowany jako zbiór zmiennych i akcji - przejść między stanami.
- Deklaratywny - skupia się na zachowaniu, nie na implementacji.
- Bazuje na logice temporalnej i matematyce zbiorów.

## Narzędzie TLC i model checking

- Weryfikacja poprawności specyfikacji odbywa się z użyciem narzędzia TLC.
- TLC używa techniki model checking:
  - automatyczne przeszukiwanie przestrzeni stanów,
  - sprawdzanie właściwości bezpieczeństwa i ciągłości.

## Metoda 1: Modelowanie matematyczne

- Określenie przestrzeni stanów i przejść między nimi (akcji).
- Przykład: bufor FIFO - kolejka, enqueue, dequeue.
- Modelowanie oparte na funkcjach, zbiorach, wektorach.
- Oddzielenie specyfikacji od implementacji.

## Metoda 2: Inwarianty bezpieczeństwa

- Inwarianty - warunki logiczne obowiązujące w każdym stanie.
- Przykład:  $\text{Len}(\text{queue}) \geq 0$ .
- Weryfikacja wszystkich możliwych trajektorii przez TLC.
- Zapobieganie błędom - np. usunięcie z pustego bufora.







## Metoda 3: Własności temporalne

- Logika temporalna LTL - analiza zachowań w czasie.
- Przykłady:
  - „zawsze po A następuje B”,
  - „stan X zostanie osiągnięty w przyszłości”.
- Formuły: bezpieczeństwo ( $[]$  (LockCount  $\leq 1$ )), ciągłość ( $\langle \rangle$  (queue = « »)).

## Zastosowania TLA+

- Firmy: Amazon, Microsoft, Google.
- Projektowanie i weryfikacja systemów:
  - rozproszonych,
  - bazodanowych,
  - komunikacyjnych,
  - chmurowych.
- Identyfikacja błędów przed implementacją - kluczowa dla systemów krytycznych.

## Bibliografia

-  Leslie Lamport. *Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers*.
-  Chris Newcombe et al. *How Amazon Web Services Uses Formal Methods*.
-  Igor Konnov, Jure Kukovec, Thanh-Hai Tran. *TLA+ Model Checking Made Symbolic*.
-  Hillel Wayne. *Practical TLA+: Planning Driven Development*. Lospinato Books, 2018.
-  S. Owre et al. *PVS System Guide*. SRI International, 1999.
-  S. Poreda. *Wykorzystanie metod formalnych do specyfikacji struktur wskaźnikowych*. Uniwersytet Warszawski, 2023.