Operační, denotační, a axiomatická sémantika programovacích jazyků. CPO, věta o pevném bodě a její použití. Hoareova logika, její korektnost a úplnost. Temporální logiky lineárního a větvícího se času a jejich fragmenty, sémantika neukončených a paralelních programů.

1 Operačná, denotačná a axiomatická sémantika

- Program: Syntaktický strom — atomy (sú z nejakej domény, potenciálne s vlastným syntaktickým stromom), operácie (nulárne operácie = pomenované konštanty)
- Sémantika: Pre každý program definuje prechodový systém zodpovedajúci výpočtu programu.
- Uvažujeme tento jazyk:
  - Základné domény: $Var = \{A, B, ...\}$, $Bool = \{tt, ff\}$, $Num = \{0, 1, -1, 2, -2, ...\}$.
  - Aritmetické výrazy: $a ::= n \in Num \mid X \in Var \mid a_0 + a_1 \mid a_0 \cdot a_1 \mid a_0 - a_1$
  - Pravdivostné výrazy: $b ::= t \in Bool \mid a_0 = a_1 \mid a_0 \leq a_1 \mid \text{not } b_0 \mid b_0 \text{ and } b_1 \mid b_0 \text{ or } b_1$
  - Príklazy: $c ::= \text{skip} \mid c_0; c_1 \mid X \in Var := a \mid \text{if } b \text{ then } c_0 \text{ else } c_1 \mid \text{while } b \text{ do } c$
- "Big-step"štrukturálna operačná sémantika:
  - Stav/konfigurácia $\sigma : Var \to \mathbb{Z}$, množina všetkých konfigurácií je Σ.
  - Množina akcií sú príkazy, prechod zodpovedá vykonaniu príkazu.
  - Tri prechodové relácie: $\to_A, \to_B, \to_C$ — C sa definuje pomocou A a B.
  - Idea: Aritmetické výrazy sa vyhodnocujú na čísla, pravdivostné výrazy na tt/ff, príkazy sa vyhodnocujú na nový stav. Relácie sú definované ako odvodzovacie systémy.
  - Príklad: $a + b$ v stave σ sa vyhodnotí na n ak a sa vyhodnotí na n_1, b sa vyhodnotí na n_2 a $n_1 + n_2 = n$; while b do c v stave σ sa vyhodnotí na σ ak b sa vyhodnotí na ff a na σ' ak b sa vyhodnotí na tt, c sa vyhodnotí na σ'' a while b do c sa v σ'' vyhodnotí na σ'.
  - Pozor: Dôkazový strom môže byť nekonečný - vtedy prechod neexistuje.
- "Small-step"štrukturálna operačná sémantika:
  - Stav/konfigurácia je $\sigma : Comm \times \Sigma$ (program a valuácia).
  - Množina akcií je prázdna (res. mám len jedno akciu τ), prechod zodpovedá vykonaniu jednej inštrukcie v príkaze.
  - Idea: Výrazy sa vyhodnocujú postupne "z ľava do prava"— ak mám výraz plne došpecifikovaný, môžem ho vykonať, inak vykonávam vnorené výrazy. While sa rozbalí na if.

- Príklad: Stav $(a + b, \sigma)$ sa vyhodnotí buď na číslo, ak a a b sú čísla, alebo na $(a' + b, \sigma)$ ak (a, σ) sa vyhodnotí na (a', σ) alebo na $(a + b', \sigma)$ (obdobne, len v prípade že a už nejde viac vyhodnotiť). (while b do c, σ) sa vyhodnotí tak, že sa rozbalí na jeden if a tento if už sa potom rieši štandardne (b sa dovyhodnotí a podľa hodnoty sa nahradí za jednu z vetiev).
- Pozn.: skip je konečná (deadlock) konfigurácia. Vykonaním inštrukcie sa inštrukcia v programe "zamení"za skip. Zreťazenie skip-c viem zameniť za c a tak pokračovať vo výpočte.

- Denotačná sémantika:
  - Nieč "ako"ale "čo"program počíta.
  - Pre jednotlivé typy výrazov definujem funkciu ktorá vracia funkciu ktorá počíta daný výraz: $\mathcal{C} : Com \to (\Sigma \to \Sigma)$ (obdobne pre $\mathcal{A}$ a $\mathcal{B}$) - zapisujem $\mathcal{C}[c](\sigma)$.
  - Všetko je triviálne (proste počítam to čo počíta inštrukcia) až na while. $\mathcal{C}[while]$ je least fixed point od funkcie $\Gamma(\varphi) = \{(\sigma, \sigma) \mid \mathcal{B}[b](\sigma) = ff\} \cup \{(\sigma, \sigma') \mid \mathcal{B}[b](\sigma) = tt \wedge \sigma' = (\varphi \circ \mathcal{C}[c])(\sigma)\}$.
  - Complete Partial Order (CPO) - partial order v ktorom ľubovoľná dvojica prvkov má supremum. (každá nekonečná neklesajúca postupnosť má supremum).
  - f je spojitá, ak je monotónna a zachováva supréma.
  - Pre monotónnu f platí že množina jej fixed pointov je tiež CPO a že fixpoint $f(\emptyset)$ je least fixed point.

- Axiomatická sémantika:
  - Odvodzovací systém v ktorom $\{A\}c\{B\}$ práve vtedy keď pre všetky $\sigma \models A$ po vykonaní c platí $\sigma' \models B$. (čiastočná korektnosť - pokiaľ program neskončí, B môže byť ľubovoľné)
  - Na výpočet "efektu"c použijeme funkciu $\mathcal{C}$ z denotačnej sémantiky, ale dodefinujeme ju pre nekončiace programy na $\perp$.
  - Tvrdenia o programoch (assertions) — artimetické výrazy rozšírené o špeciálne premenné $IntVar = \{i, j, ...\}$, logika prvého rádu s kvantifikáciou cez $IntVar$: $T, F, =, \leq, \wedge, \vee, \neg, \exists i :, \forall i :$.
  - Interpretácia $I : IntVar \to \mathbb{Z}$. Všetky interpretácie $\mathcal{I}$. Sémantická funkcia rozšírených aritmetických výrazov $E : Aexp+ \to (\mathcal{I} \to (\Sigma \to \mathbb{Z}))$, s ňou už potom definujem sémantiku assertions prirodzene (v závislosti na I).
  - $\sigma \models^I AcB \iff (\sigma \models^I A \implies \mathcal{C}[c](\sigma) \models B)$, $\models^I AcB \iff \forall \sigma : \sigma \models AcB$, $\models AcB \iff \forall I : \models^I AcB$ (AcB je platné tvrdenie).
  - Pozn.: Z praktického hľadiska sú Σ a $\Sigma_\perp$ skoro ekvivaletné, keďže $\perp$ spĺňa všetko.

- Hoareho odvodzovací systém:
  - Axiom: $A|skip|A$
  - Axiom: $A[X/a]|X := a|A$
  - Rule: Ak $A|c_0|C$ a $C|c_1|B$, tak $A|c_0; c_1|B$.
  - Rule: Ak $A \wedge b|c_0|B$ a $A \wedge \neg b|c_1|B$, tak $A|$if b then c_0 else $c_1|B$
  - Rule: Ak $A \wedge b|c|A$, tak $A|$while b do $c|A \wedge \neg b$
  - Rule (dôsledok): Ak $\models (A \implies A')$, $\models (B \implies B')$ a $A'|c|B'$, tak aj $A|c|B$.
- Korektnosť: Ak $\vdash A|c|B$, tak aj $\models A|c|B$
- Úplnosť: Ak $\models A|c|B$, tak $\vdash A|c|B$ (vyplýva z existencie weakest precondition)
- Weakest precondition $wp^I[c, B]$: Množina stavov z ktorých keď vykonám c, bude platiť B. Existuje assertion výraz taký že $A[c, B] = wp^I[c, B]$ pre všetky I.

- Paralelné programy: Pridám operátor || a rozšírim sémantiku — vzniká nedeterminizmus. No biggie.
- Neukončené programy: Nekonečná postupnosť konfigurácií, nekonečný odvodzovací strom, etc.

2 Temporálne logiky

- CTL, LTL