Fundamentele limbajelor de programare

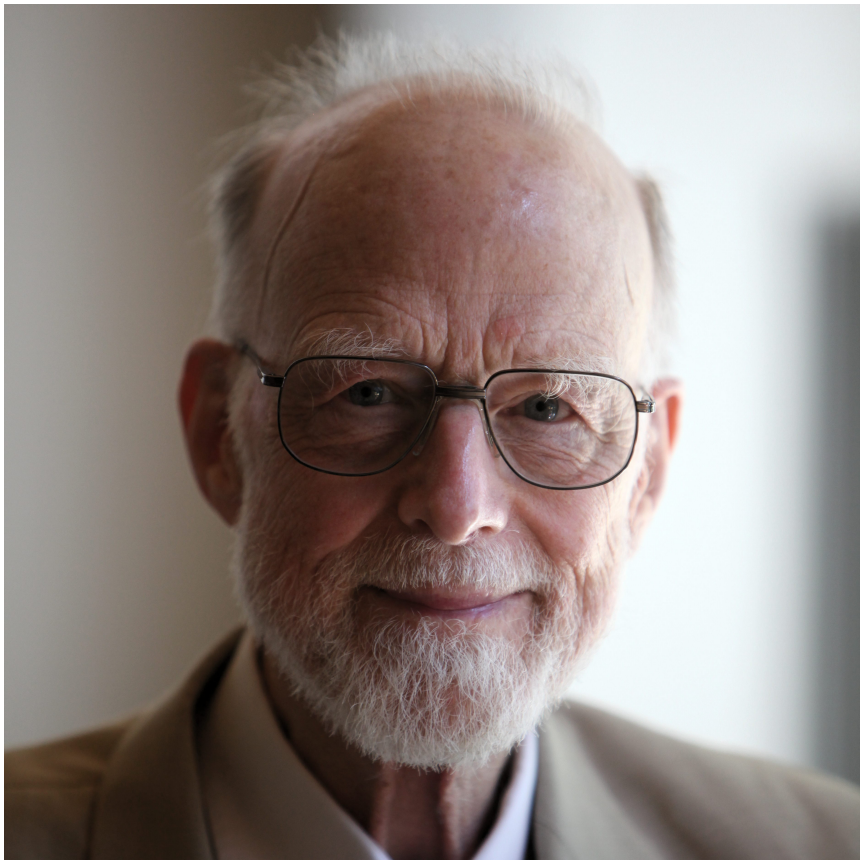
CURS 4

Andrei Sipoș[1]

Facultatea de Matematică și Informatică, DL Info, Anul II

Semestrul II, 2025/2026

[1] Curs realizat împreună cu Traian Florin Șerbănuță.



Sir Charles Antony Richard "Tony" Hoare (1934–2026)

"Although one of his most well-known discoveries is the quicksort algorithm, taught in every introductory algorithms course, I've had the unique pleasure of struggling with applying Hoare logic (a method of embedding and verifying the specification of an algorithm within the code itself) to a program I wrote in a Software Development exam during a rainy day of June almost three years ago – I think that this is an experience to which every wannabe computer scientist should submit at least once in the process of learning. Also, this Hoare logic is tightly linked to algebraic and logical verification of computer programs, a topic I found dear for a non-trivial period of my life."

A. S., 2014

Logica Hoare

Reamintim din Introducerea istorică faptul că Hoare a introdus reguli de a raționa cu enunțuri de forma

$$\{A\}c\{B\},$$

însemnând faptul că un program c, care satisface la începutul rulării sale proprietatea A, va satisface la sfârșit proprietatea B. Aceste reguli oferă o **semantică axiomatică** pentru un limbaj imperativ, care este numită **logică Hoare**, și care are rolul de a formaliza adecvat procesul **verificării programelor**.

Pentru a o defini, va trebui să extindem conceptele introduse anterior de expresie aritmetică și expresie booleană.

Expresii aritmetice extinse

Fixăm o mulțime V, ale cărei elemente vor fi numite **variabile logice**, care vor avea oarecum același rol ca variabilele din logica de ordinul I (altul decât cel al variabilelor din IMP, numite și **variabile de program**, care reprezentau locuri în memorie, ca în limbajele imperative uzuale).

O **expresie aritmetică extinsă** va avea exact una dintre următoarele forme:

- un număr întreg n;
- o variabilă de program X (element al lui L);
- o variabilă logică x (element al lui V);
- $a_0 + a_1$, $a_0 - a_1$, $a_0 * a_1$, unde a_0 și a_1 sunt expresii aritmetice extinse.

Expresii booleene extinse (aserțiuni)

O **expresie booleană extinsă**, numită și **aserțiune**, va avea exact una dintre următoarele forme:

- o valoare booleană (**true** sau **false**);
- $a_0 = a_1$, $a_0 \leq a_1$, unde a_0 și a_1 sunt expresii aritmetice extinse;
- $\neg A_0$, $A_0 \wedge A_1$, $A_0 \vee A_1$, unde A_0 și A_1 sunt aserțiuni;
- $\forall x A$, unde $x \in V$ și A este o aserțiune.

Considerăm cunoscută prescurtarea $A \rightarrow B$ pentru $(\neg A) \vee B$.

Substituția

Putem defini în modul absolut natural substituțiile de forma $A[x := a]$ sau $A[X := a]$, unde A este o aserțiune, $X \in L$, $x \in V$, iar a este o expresie aritmetică – nu extinsă: de aceea, nu trebuie să ne preocupăm de redenumiri de variabile logice (cum o facem la logica de ordinul I), deoarece acestea nu pot fi „capturate accidental". Enunțăm doar clauza:

$$(\forall z A)[x := a] := \begin{cases} \forall z A, & \text{dacă } z = x, \\ \forall z (A[x := a]), & \text{altfel.} \end{cases}$$

Evaluarea expresiilor aritmetice extinse

Vom numi **interpretare** o funcție de la V la $\mathbb{Z}$. Pentru orice $\sigma \in \Sigma$ și orice interpretare $I : V \to \mathbb{Z}$, definim o funcție $|\cdot|_\sigma^I$ care evaluează expresii aritmetice extinse în numere întregi, în mod recursiv, în felul următor:

- pentru orice $N \in \mathbb{Z}$, $|N|_\sigma^I := N$;
- pentru orice $X \in L$, $|X|_\sigma^I := \sigma(X)$;
- pentru orice $x \in V$, $|x|_\sigma^I := I(x)$;
- pentru orice expresii aritmetice extinse a_0, a_1, avem
  $|a_0 + a_1|_\sigma^I := |a_0|_\sigma^I + |a_1|_\sigma^I$,
  $|a_0 - a_1|_\sigma^I := |a_0|_\sigma^I - |a_1|_\sigma^I$,
  $|a_0 * a_1|_\sigma^I := |a_0|_\sigma^I * |a_1|_\sigma^I$.

Avem că, pentru orice expresie aritmetică a, $|a|_\sigma^I = e_\sigma(a) = [\![a]\!](\sigma)$.
Analog cu cele anterioare, pentru orice $I : V \to \mathbb{Z}$, $x \in V$ și $N \in \mathbb{Z}$, definim interpretarea $I_{x \mapsto N}$, pentru orice $y \in V$, prin:

$$I_{x \mapsto N}(y) := \begin{cases} N, & \text{dacă } y = x, \\ I(y), & \text{altfel.} \end{cases}$$

Evaluarea aserțiunilor

Definim acum, pentru $\sigma \in \Sigma$, $I : V \to \mathbb{Z}$ și A aserțiune, relația $\sigma \models^I A$, în mod recursiv, în felul următor:

- $\sigma \models^I \textbf{true}$, $\sigma \not\models^I \textbf{false}$;
- pentru orice expresii aritmetice extinse a_0, a_1, avem
  $\sigma \models^I a_0 = a_1$ dacă și numai dacă $|a_0|^I_\sigma = |a_1|^I_\sigma$,
  $\sigma \models^I a_0 \leq a_1$ dacă și numai dacă $|a_0|^I_\sigma \leq |a_1|^I_\sigma$;
- pentru orice aserțiuni A_0, A_1, avem
  $\sigma \models^I \neg A_0$ dacă și numai dacă $\sigma \not\models^I A_0$,
  $\sigma \models^I A_0 \wedge A_1$ dacă și numai dacă $\sigma \models^I A_0$ și $\sigma \models^I A_1$,
  $\sigma \models^I A_0 \vee A_1$ dacă și numai dacă $\sigma \models^I A_0$ sau $\sigma \models^I A_1$;
- pentru orice $x \in V$ și orice aserțiune A, avem
  $\sigma \models^I \forall x A$ dacă și numai dacă, pentru orice $N \in \mathbb{Z}$,
  $\sigma \models^{I_{x \mapsto N}} A$.

Avem că, pentru orice expresie booleană b, $\sigma \models^I b$ dacă și numai dacă $e_\sigma(b) = [\![b]\!](\sigma) = 1$. Notăm cu $\models A$ faptul că, pentru orice σ și I, $\sigma \models^I A$. Notăm și $A^I := \{\sigma \in \Sigma \mid \sigma \models^I A\}$.

Enunțuri Hoare și semantica lor

Vom numi **enunț Hoare** o expresie de forma $\{A\}c\{B\}$, unde A și B sunt aserțiuni, iar c este o instrucțiune. Vom defini semantica acestora în felul următor:

- pentru orice σ și I, $\sigma \models^I \{A\}c\{B\}$ dacă $\sigma \models^I A$ implică faptul că, pentru orice σ' cu $(\sigma, \sigma') \in [\![c]\!]$, $\sigma' \models^I B$;
- pentru orice I, $\models^I \{A\}c\{B\}$ dacă, pentru orice σ, $\sigma \models^I \{A\}c\{B\}$, sau, echivalent, pentru orice $(\sigma, \sigma') \in [\![c]\!]$, $\sigma \models^I A$ implică $\sigma' \models^I B$;
- $\models \{A\}c\{B\}$ dacă, pentru orice I, $\models^I \{A\}c\{B\}$.

Prezentăm în continuare regulile de deducție pentru aceste enunțuri Hoare.

Regulile de deducție Hoare

$$\frac{}{\{A\}\textbf{skip}\{A\}} \qquad \frac{}{\{B[X := a]\}X := a\{B\}}$$

$$\frac{\{A\}c_0\{C\} \quad \{C\}c_1\{B\}}{\{A\}c_0; c_1\{B\}} \qquad \frac{\{A \wedge b\}c_0\{B\} \quad \{A \wedge \neg b\}c_1\{B\}}{\{A\}\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1\{B\}}$$

$$\frac{\{A \wedge b\}c\{A\}}{\{A\}\textbf{while } b \textbf{ do } c\{A \wedge \neg b\}} \qquad \frac{\models A \to A' \quad \{A'\}c\{B'\} \quad \models B' \to B}{\{A\}c\{B\}}$$

Notăm faptul că un enunț Hoare $\{A\}c\{B\}$ poate fi dedus prin aceste reguli cu $\vdash \{A\}c\{B\}$. Observăm și că ultima regulă, numită **regula consecinței**, este de fapt o **pseudo-regulă**, având în vedere că nu este decidabil dacă o aplicare a ei este corectă (de ce?).

Un exemplu concret

Lemă

Fie $X \in L$, a o expresie aritmetică și A o aserțiune cu proprietatea că X nu apare nici în a, nici în A. Atunci

$$\vdash \{A\}X := a\{A \wedge X = a\}.$$

Demonstrație

Din regula pentru atribuire, $\vdash \{A \wedge a = a\}X := a\{A \wedge X = a\}$.
Cum $\models A \rightarrow A \wedge a = a$, din regula consecinței obținem concluzia.

Fixăm N, S, $I \in L$ (distincte două câte două) și $n \in \mathbb{N}$ și considerăm programul p, definit ca:

$$((N := n; S := 0); I := 0); (\textbf{while } I \leq N \textbf{ do } (S := S + I; I := I + 1)).$$

Vom arăta că $\vdash \{\textbf{true}\}p\{2 * S = n * (n + 1)\}$. Vom nota blocul **while** din p cu w.

Un exemplu concret

Din lemă, avem $\vdash \{\mathbf{true}\} N := n \{\mathbf{true} \wedge N = n\}$, deci, din regula consecinței, $\vdash \{\mathbf{true}\} N := n \{N = n\}$. Tot din lemă, avem $\vdash \{N = n\} S := 0 \{N = n \wedge S = 0\}$. Din regula pentru secvențiere, avem $\vdash \{\mathbf{true}\} N := n; S := 0 \{N = n \wedge S = 0\}$. Similar, obținem apoi $\vdash \{\mathbf{true}\} (N := n; S := 0); I := 0 \{N = n \wedge S = 0 \wedge I = 0\}$.

Din regula pentru atribuire, avem

$$\begin{array}{l} \vdash \{2 * (S + I) + 2 * (I + 1) = (I + 1) * (I + 2) \wedge I + 1 \leq N + 1 \wedge N = n\} \\ \quad S := S + I \\ \quad \{2 * S + 2 * (I + 1) = (I + 1) * (I + 2) \wedge I + 1 \leq N + 1 \wedge N = n\}, \end{array}$$

iar din regula consecinței, scoatem

$$\begin{array}{l} \vdash \{2 * S + 2 * I = I * (I + 1) \wedge I \leq N \wedge N = n\} \\ \quad S := S + I \\ \quad \{2 * S + 2 * (I + 1) = (I + 1) * (I + 2) \wedge I + 1 \leq N + 1 \wedge N = n\}. \end{array}$$

Un exemplu concret

Tot din regula pentru atribuire, obținem

$$\vdash \{2 * S + 2 * (I + 1) = (I + 1) * (I + 2) \wedge I + 1 \leq N + 1 \wedge N = n\}$$
$$I := I + 1$$
$$\{2 * S + 2 * I = I * (I + 1) \wedge I \leq N + 1 \wedge N = n\}.$$

Din regula pentru secvențiere, avem

$$\vdash \{2 * S + 2 * I = I * (I + 1) \wedge I \leq N \wedge N = n\}$$
$$S := S + I; I := I + 1$$
$$\{2 * S + 2 * I = I * (I + 1) \wedge I \leq N + 1 \wedge N = n\}.$$

Din regula consecinței, obținem

$$\vdash \{2 * S + 2 * I = I * (I + 1) \wedge I \leq N + 1 \wedge N = n \wedge I \leq N\}$$
$$S := S + I; I := I + 1$$
$$\{2 * S + 2 * I = I * (I + 1) \wedge I \leq N + 1 \wedge N = n\}.$$

Un exemplu concret

Aplicând regula pentru **while**, avem:

$$\vdash \{2 * S + 2 * I = I * (I + 1) \wedge I \leq N + 1 \wedge N = n\}$$
$$\textbf{while } I \leq N \textbf{ do } (S := S + I; I := I + 1)$$
$$\{2 * S + 2 * I = I * (I + 1) \wedge I \leq N + 1 \wedge N = n \wedge \neg(I \leq N)\}.$$

Din regula consecinței, scoatem

$$\vdash \{N = n \wedge S = 0 \wedge I = 0\} w \{2 * S = n * (n + 1)\}.$$

Aplicând apoi regula pentru secvențiere pentru acest enunț și pentru cel obținut mai devreme, anume

$$\vdash \{\textbf{true}\}(N := n; S := 0); I := 0\{N = n \wedge S = 0 \wedge I = 0\},$$

obținem $\vdash \{\textbf{true}\} p \{2 * S = n * (n + 1)\}$, ceea ce trebuia demonstrat.

Teorema de corectitudine

Teorema de corectitudine

Pentru orice A, B, c cu $\vdash \{A\}c\{B\}$, avem $\models \{A\}c\{B\}$.

Demonstrație

Demonstrăm prin inducție după regulile Hoare. Tratăm cazul regulii pentru **while**. Notăm $w := \textbf{while}\ b\ \textbf{do}\ c$. Presupunem $\models \{A \wedge b\}c\{A\}$. Vrem $\models \{A\}w\{A \wedge \neg b\}$. Fie $I : V \to \mathbb{Z}$. Notând $W := \{(\sigma, \sigma') \in \Sigma^2 \mid \sigma \models^I A \text{ implică } \sigma' \models^I A \wedge \neg b\}$, vrem să arătăm $[\![w]\!] \subseteq W$. Este suficient să arătăm $\Gamma^{[\![b]\!]}_{[\![c]\!]}(W) \subseteq W$. Fie $(\sigma, \sigma') \in \Gamma^{[\![b]\!]}_{[\![c]\!]}(W)$ și tratăm cazul $e_\sigma(b) = 1$. Știm $\sigma \models^I A$ și vrem $\sigma' \models^I A \wedge \neg b$. Din definiția lui $\Gamma^{[\![b]\!]}_{[\![c]\!]}$, avem că există σ'' cu $(\sigma, \sigma'') \in [\![c]\!]$ și $(\sigma'', \sigma') \in W$. Cum $e_\sigma(b) = 1$, $\sigma \models^I A \wedge b$. Cum $\models \{A \wedge b\}c\{A\}$, avem $\sigma'' \models^I A$, iar, cum $(\sigma'', \sigma') \in W$, avem $\sigma' \models^I A \wedge \neg b$, ceea ce trebuia demonstrat.

Cea mai slabă precondiție

Pentru orice instrucțiune c, orice aserțiune B și orice interpretare I, definim **cea mai slabă precondiție** (**semantică**) a lor prin

$$ws^I(c, B) := \{\sigma \in \Sigma \mid \text{pentru orice } \sigma' \in \Sigma \text{ cu } (\sigma, \sigma') \in [\![c]\!],\ \sigma' \models^I B\}.$$

Avem că, pentru orice A, B, c, I, $\models^I \{A\}c\{B\}$ dacă și numai dacă $A^I \subseteq ws^I(c, B)$ (exercițiu!).

În continuare, vom demonstra că această construcție poate fi capturată de o aserțiune, în sensul că, pentru orice instrucțiune c și orice aserțiune B, există o aserțiune A, numită și ea **cea mai slabă precondiție** (**sintactică**, de data aceasta), astfel încât, pentru orice I, $A^I = ws^I(c, B)$. În particular, A este unică până la o echivalență semantică. Vom construi o asemenea aserțiune prin inducție structurală și o vom nota cu $wp(c, B)$. Așadar, pentru orice c și B, $\models \{wp(c, B)\}c\{B\}$, iar, pentru orice I, $wp(c, B)^I$ va fi egal cu $ws^I(c, B)$.

Definirea celei mai slabe precondiții

Vom avea:

$$
\begin{aligned}
wp(\textbf{skip}, B) &:= B \\
wp(X := a, B) &:= B[X := a] \\
wp(c_0; c_1, B) &:= wp(c_0, wp(c_1, B)) \\
wp(\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1, B) &:= (b \wedge wp(c_0, B)) \vee (\neg b \wedge wp(c_1, B)).
\end{aligned}
$$

Definiția pentru instrucțiunea **while** va fi destul de migăloasă și, de aceea, doar vom schița ideile care stau la baza ei. Vom nota în continuare, ca de obicei, $w := \textbf{while } b \textbf{ do } c$. Scopul este de a defini $wp(w, B)$.

Despre **while**

Claim 1: Avem că, pentru orice $(\sigma, \sigma') \in \Sigma^2$, $(\sigma, \sigma') \in [\![w]\!]$ dacă și numai dacă există $n \geq 0$ și un șir finit de stări $(\sigma_i)_{i \leq n}$ cu $\sigma_0 = \sigma$, $\sigma_n = \sigma'$, $[\![b]\!](\sigma_n) = 0$ și, pentru orice i cu $0 \leq i < n$, $[\![b]\!](\sigma_i) = 1$ și $(\sigma_i, \sigma_{i+1}) \in [\![c]\!]$.

Demonstrația acestui fapt se regăsește în cursul trecut.

Claim 2: Avem că, pentru orice σ și I, $\sigma \in ws^I(w, B)$ dacă și numai dacă pentru orice $n \geq 0$ și orice șir finit de stări $(\sigma_i)_{i \leq n}$ cu $\sigma_0 = \sigma$ și, pentru orice i cu $0 \leq i < n$, $[\![b]\!](\sigma_i) = 1$ și $(\sigma_i, \sigma_{i+1}) \in [\![c]\!]$, avem $\sigma_n \models^I b \vee B$.

Vom dori în continuare să reducem șirurile de stări la șiruri de numere. Pentru aceasta, ne folosim de faptul că locurile de memorie menționate în aserțiuni și instrucțiuni sunt în număr finit, și numai valorile acelora sunt cele care contează.

Despre **while**

Fie $X_1, \ldots, X_l$ acele locuri de memorie care apar în w și B.
Claim 3: Avem că, pentru orice σ și I, $\sigma \in ws^I(w, B)$ dacă și numai dacă pentru orice $n \geq 0$ și orice $(s_{ij})_{i \leq n,\ j \leq l} \subseteq \mathbb{N}$, dacă avem $\sigma \models^I \bigwedge_{j=1}^{l} X_j = s_{0j}$ și, pentru orice i cu $0 \leq i < n$,

$$\sigma \models^I b[X_1 := s_{i1}] \ldots [X_l := s_{il}]$$

și

$$\sigma \models^I \left(wp\left(c, \bigwedge_{j=1}^{l} X_j = s_{(i+1)j}\right) \wedge \neg wp(c, \textbf{false}) \right) [X_1 := s_{i1}] \ldots [X_l := s_{il}],$$

avem

$$\sigma \models^I (b \vee B)[X_1 := s_{n1}] \ldots [X_l := s_{nl}].$$

Practic, acum, dacă scriem ca aserțiune această proprietate a lui ws, îl vom obține pe wp – singurul obstacol este faptul că avem cuantificare după șiruri finite de numere, care nu există în limbajul nostru.

Lema beta a lui Gödel

Pentru a exprima idei care se referă la șiruri de numere, se folosește un rezultat tehnic, numit lema beta a lui Gödel, folosit de acesta cu un scop similar în demonstrarea teoremei sale de incompletitudine (un mod mai cunoscut de a codifica șiruri de numere ca numere simple este cel care folosește descompunerea în factori primi).

Lema beta a lui Gödel

Există o aserțiune β astfel încât, pentru orice $k \in \mathbb{N}$ și orice șir de numere întregi $(a_i)_{i \leq k}$ există n, $m \in \mathbb{N}$ astfel încât, pentru orice $j \in \{0, \ldots, k\}$ și orice $x \in \mathbb{Z}$, avem

$$\beta(n, m, j, x) \Leftrightarrow x = a_j.$$

În acest moment, considerăm că am terminat definirea lui *wp*.

O variantă

Putem exprima mai ușor $wp(w, B)$, și îl putem și implementa fezabil la laborator, dacă facem următoarele presupuneri:

- avem disjuncții infinitare în limbaj;
- w se termină, adică pentru orice σ există σ' cu $(\sigma, \sigma') \in [\![w]\!]$.

Definim acum $P_0 := \neg b \wedge B$ și, pentru orice k,

$$P_{k+1} := b \wedge wp(c, P_k).$$

Atunci vom pune

$$wp(w, B) := \bigvee_{k \in \mathbb{N}} P_k.$$

A se observa că, în această ipoteză, nu mai avem nevoie de cuantificatori, deci nici de variabile logice, de expresii aritmetice extinse sau de interpretări, fapt care se va reflecta în suportul de laborator.

O variantă

Vom demonstra acum că definiția este adecvată.

Fie I o interpretare. Vom arăta că $wp(w, B)^I = ws^I(w, B)$.

Demonstrație

Pentru „$\subseteq$”, fie σ cu $\sigma \models^I \bigvee_{k\in\mathbb{N}} P_k$, deci există $k \in \mathbb{N}$ cu $\sigma \models^I P_k$. Facem inducție după k.

Pentru $k = 0$, avem $\sigma \models^I \neg b \wedge B$, deci $e_\sigma(b) = 0$. Fie σ' cu $(\sigma, \sigma') \in [\![w]\!]$, deci $\sigma' = \sigma$ și $\sigma' \models^I B$.

Pentru pasul de inducție, presupunem $\sigma \models^I b \wedge wp(c, P_k)$, deci $e_\sigma(b) = 1$. Fie σ' cu $(\sigma, \sigma') \in [\![w]\!]$, deci există σ'' cu $(\sigma, \sigma'') \in [\![c]\!]$ și $(\sigma'', \sigma') \in [\![w]\!]$. Cum $\sigma \models^I wp(c, P_k)$, avem $\sigma'' \models^I P_k$. Din ipoteza de inducție, avem $\sigma'' \in ws^I(w, B)$, deci $\sigma' \models^I B$.

O variantă

Demonstrație (cont.)

Pentru „$\supseteq$”, fie $\sigma \in ws^I(w, B)$. Vrem $k \in \mathbb{N}$ cu $\sigma \models^I P_k$. Aici aplicăm presupunerea noastră și luăm σ' cu $(\sigma, \sigma') \in [\![w]\!]$, deci $\sigma' \models^I B$.

Aplicăm primul claim de mai devreme pentru a obține că, pentru orice $(\sigma, \sigma') \in \Sigma^2$, $(\sigma, \sigma') \in [\![w]\!]$ dacă și numai dacă există $n \geq 0$ și un șir finit de stări $(\sigma_i)_{i \leq n}$ cu $\sigma_0 = \sigma$, $\sigma_n = \sigma'$, $[\![b]\!](\sigma_n) = 0$ și, pentru orice i cu $0 \leq i < n$, $[\![b]\!](\sigma_i) = 1$ și $(\sigma_i, \sigma_{i+1}) \in [\![c]\!]$.

Se arată, apoi, că, pentru orice j cu $0 \leq j \leq n$, avem $\sigma_{n-j} \models^I P_j$, prin inducție după j (exercițiu!). Avem, deci, că $\sigma = \sigma_0 \models^I P_n$.

A se vedea și cursul de master “Program Verification”:

https://cs.unibuc.ro/~ddiaconescu/2019/pv/

Teorema de completitudine

Lemă

Fie c, B cu $\vdash \{wp(c, B)\}c\{B\}$. Fie A cu $\models \{A\}c\{B\}$. Atunci $\vdash \{A\}c\{B\}$.

Demonstrație

Din regula consecinței, e suficient să arătăm $\models A \to wp(c, B)$. Fie I. Cum $\models^I \{A\}c\{B\}$, avem $A^I \subseteq ws^I(c, B) = wp(c, B)^I$, deci $\models^I A \to wp(c, B)$.

Teorema de completitudine

Pentru orice A, B, c cu $\models \{A\}c\{B\}$, avem $\vdash \{A\}c\{B\}$.

Demonstrație

Demonstrăm prin inducție structurală după c. Din lema anterioară, rezultă că este suficient, la fiecare pas, să arătăm că, pentru orice B, $\vdash \{wp(c, B)\}c\{B\}$.

Demonstrația teoremei de completitudine

Demonstrație (cont.)

Vom trata cazurile instrucțiunilor **if** și **while**.

Pentru **if**, notăm $i := \textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1$. Știm:

$$wp(i, B) = (b \wedge wp(c_0, B)) \vee (\neg b \wedge wp(c_1, B)).$$

Este imediat că $\models wp(i, B) \wedge b \rightarrow wp(c_0, B)$. Din ipoteza de inducție, știm $\vdash \{wp(c_0, B)\}c_0\{B\}$, așadar, din regula consecinței, scoatem $\vdash \{wp(i, B) \wedge b\}c_0\{B\}$. Analog, $\vdash \{wp(i, B) \wedge \neg b\}c_1\{B\}$. Concluzia rezultă aplicând regula pentru **if**.

Pentru **while**, notăm $w := \textbf{while } b \textbf{ do } c$ și $A := wp(w, B)$.

Demonstrația teoremei de completitudine

Demonstrație (cont.)

Claim 1: $\models \{A \wedge b\}c\{A\}$.
Dem. claim: Fie I și $(\sigma, \sigma'') \in [\![c]\!]$. Presupunem $\sigma \models^I A \wedge b$, deci $[\![b]\!](\sigma) = 1$. Vrem $\sigma'' \in A^I = wp(w, B)^I = ws^I(w, B)$. Fie σ' cu $(\sigma'', \sigma') \in [\![w]\!]$. Vrem $\sigma' \models^I B$. Cum $(\sigma, \sigma'') \in [\![c]\!]$ și $[\![b]\!](\sigma) = 1$, avem $(\sigma, \sigma') \in [\![w]\!]$. Cum $\sigma \models^I A$, adică $\sigma \in ws^I(w, B)$, rezultă $\sigma' \models^I B$.

Claim 2: $\models (A \wedge \neg b) \rightarrow B$.
Dem. claim: Fie σ, I cu $\sigma \models^I A \wedge \neg b$. Vrem $\sigma \models^I B$. Cum $[\![b]\!](\sigma) = 0$, $(\sigma, \sigma) \in [\![w]\!]$, iar cum $\sigma \in A^I = ws^I(w, B)$, avem $\sigma \models^I B$.

Demonstrăm acum că $\vdash \{A\}w\{B\}$. Aplicând ipoteza de inducție pe primul claim, avem $\vdash \{A \wedge b\}c\{A\}$, deci, din regula pentru **while**, avem $\vdash \{A\}w\{A \wedge \neg b\}$. Aplicând regula consecinței și al doilea claim, avem $\vdash \{A\}w\{B\}$.