

Verificarea și Validarea Sistemelor Soft

Curs 8. Corectitudine (Floyd. Hoare. Dijkstra)

Partea I

Lector dr. Camelia Chisăliță-Crețu

Universitatea Babeș-Bolyai
Cluj-Napoca

25 Aprilie 2023

- 1 Evaluarea calității softului
 - Evaluarea calității softului
 - Verificarea programelor
- 2 Metoda lui Floyd
 - Metoda aserțiunilor inductive
 - Metoda lui Floyd. Parțial corectitudine
 - Metoda lui Floyd. Terminare
- 3 Axiomatizarea lui Hoare
 - Triplete Hoare. Semantică
 - Parțial corectitudine. Reguli deductive
 - Total corectitudine. Reguli deductive
- 4 Bibliografie

Evaluarea calității softului

- **calitatea softului** –
 - conformitatea cu **cerințele funcționale** și de **performanță** precizate, **documentate explicit** în **standardele de dezvoltare** și **caracteristici implicite** ale unui produs soft dezvoltat. [Scott Pressman, 2005]
- **corectitudine** – proprietate a unui program de a respecta specificațiile și a oferi rezultate corecte [Fre10].

Verificarea programelor

- **metode formale** pentru verificarea programelor:
 - bazate pe demonstrarea corectitudinii:
 - asistate de calculator, presupune *verificarea corectitudinii codului sursă asociat programului*;
 - aplicate programelor care trebuie să se termine și să obțină un rezultat (**curs 08**);
 - bazate pe modele:
 - automate, presupune *verificarea proprietăților programului*;
 - aplicate sistemelor concurente; se aplică în etapele post-dezvoltare, e.g., verificarea modelelor.

Metode pentru demonstrarea corectitudinii programelor

- Metoda lui Floyd –
 - metoda aserțiunilor inductive;
- Axiomatizarea lui Hoare –
 - axiome și reguli deducțive;
 - dezvoltarea algoritmilor din specificații;
- Limbajul lui Dijkstra –
 - instrucțiuni cu santinelă;
 - non-determinism;
 - derivarea formală a programelor.

- Robert W Floyd
(8 Iunie 1936 – 25 Septembrie 2001)



- Sir Charles Antony Richard Hoare
(11 January 1934)



- Edsger Wybe Dijkstra
(11 Mai 1930 – 6 August 2002)



Metoda lui Floyd. Metoda aserțiunilor inductive

Aplicabilitate:

- pentru a demonstra:
 - 1 parțial corectitudinea programului;
 - 2 terminarea programului;
 - 3 **total corectitudinea = parțial corectitudinea programului + terminarea programului.**

Folosește:

- **precondiție** – condiția satisfăcută de datele de intrare ale programului;
- **postcondiție** – condiția care trebuie satisfăcută de rezultatele programului;
- **algoritmul** – descrierea programului (codul sursă);

Etape de aplicare:

- 1 identificarea unui punct de tăietură în fiecare buclă;
- 2 identificarea unei mulțimi de aserțiuni inductive;
- 3 construirea și demonstrarea condițiilor de verificare/terminare.

Parțial corectitudine. Etape de realizare

- ❶ se aleg puncte de tăietură în cadrul algoritmului:
 - două puncte de tăietură particulare: un punct de tăietură la începutul algoritmului, un punct de tăietură la sfârșitul algoritmului;
 - cel puțin un punct de tăietură în fiecare instrucțiune repetitivă;
- ❷ pentru fiecare punct de tăietură se alege câte un predicat invariant (aserțiune):
 - punctul de intrare - $\varphi(X)$;
 - punctul de ieșire - $\psi(X, Z)$;
- ❸ se construiesc și se demonstrează condițiile de verificare:
 - ❶ $\forall X \forall Y (P_i(X, Y) \wedge R_{\alpha_{i,j}}(X, Y) \rightarrow P_j(X, r_{\alpha_{i,j}}(X, Y)))$;
 - ❷ Y – vector de variabile cu rezultate intermediare;
 - ❸ $\alpha_{i,j}$ – drumul de la punctul de tăietură i la punctul de tăietură j ;
 - ❹ P_i și P_j – predicate invariante în punctele de tăietură i și j asociate;
 - ❺ $R_{\alpha_{i,j}}(X, Y)$ – predicat care dă condiția de parcurgere a drumului α ;
 - ❻ $r_{\alpha_{i,j}}(X, Y)$ – funcție care indică transformările variabilelor Y de pe drumul α ;

Theorem

1. Dacă toate condițiile de verificare sunt adevărate atunci programul P este parțial corect în raport cu specificația $(\varphi(X), \psi(X, Z))$. [Fre10]

Parțial corectitudine. Exemplu.

- algoritmul pentru ridicarea la putere prin înmulțiri repetate: $z = x^y$;
- Algoritmul $\text{putere}(x, y, z)$ este:

$$\mathbf{A: } \varphi(X) ::= (x > 0 \wedge y \geq 0)$$

$$z := 1; u := x; v := y;$$

$$\text{cattimp } (v > 0) \text{ execută}$$

$$\mathbf{B: } \eta(X, Y) ::= z * u^v = x^y$$

$$\text{dacă } (v \% 2 == 0)$$

$$\text{atunci } u := u * u; v := v/2;$$

$$\text{altfel } v := v - 1; z := z * u;$$

$$\text{sfdacă}$$

$$\text{sfcattimp}$$

$$\mathbf{C: } \psi(X, Z) ::= z = x^y$$

$$\text{sfAlg};$$
- se aleg punctele de tăietură: A, B și C;
- se stabilesc predicatele invariante pentru punctele de tăietură alese: $\varphi(X)$, $\psi(X, Z)$ și $\eta(X, Y)$;
- drumurile α între punctele de tăietură: $\{\alpha_{AB}, \alpha_{BB}, \alpha_{BC}, \alpha_{AC}\}$
 $\Rightarrow \{\alpha_{AB}, \alpha_{BB_{atunci}}, \alpha_{BB_{altfel}}, \alpha_{BC}, \alpha_{AC}\}$;
- $R_{\alpha_{i,j}}(X, Y)$ – predicate pentru parcurgerea drumurilor $\alpha_{i,j}$;
- $r_{\alpha_{i,j}}(X, Y)$ – funcții care indică transformările variabilelor Y de pe drumurile $\alpha_{i,j}$;
- pentru fiecare drum α se construiește și se demonstrează condiția de verificare de forma
 $\forall X \forall Y (P_i(X, Y) \wedge R_{\alpha_{i,j}}(X, Y) \rightarrow P_j(X, r_{\alpha_{i,j}}(X, Y)))$;

Terminarea algoritmului. Etape de realizare.

- 1 se aleg punctele de tăietură în cadrul algoritmului;
- 2 pentru fiecare punct de tăietură se alege câte un predicat invariant;
- 3 se alege o **mulțime convenabilă** M (i.e., o mulțime parțial ordonată, care nu conține nici un șir descrescător infinit) și o funcție descrescătoare u_i ;
 - în punctul de tăietură i funcția aleasă este $u_i : D_X \times D_Y \rightarrow M$;
- 4 se scriu condițiile de terminare:
 - condiția de terminare pe drumul $\alpha_{i,j}$ este:
$$\forall X \forall Y (\varphi(X) \wedge R_{\alpha_{i,j}}(X, Y) \rightarrow (u_i(X, Y) > u_j(X, r_{\alpha_{i,j}}(X, Y))));$$
 - dacă s-a demonstrat parțial corectitudinea, atunci condiția de terminare poate fi:
$$\forall X \forall Y (P_i(X) \wedge R_{\alpha_{i,j}}(X, Y) \rightarrow (u_i(X, Y) > u_j(X, r_{\alpha_{i,j}}(X, Y))));$$
- 5 se demonstrează condițiile de terminare:
 - la trecerea de la un punctul de tăietură i la j valorile funcției u descresc, i.e., $u_i > u_j$.

Theorem

2. Dacă toate condițiile de terminare sunt adevărate atunci programul P se termină în raport cu predicatul $\varphi(X)$. [Fre10]

Sistemul axiomatic al lui Hoare

- Relații și notații:
 - **deductibilitate:** \models ;
 $g_1, g_2, \dots, g_m \models h$ are semnificația: *“formula predicativă h (concluzia) este deductivă din formulele predicative g_1, g_2, \dots, g_m (premisele)”*;
 - **implicația:** \Rightarrow ;
 $P \Rightarrow P'$ are semnificația: *“dacă P este satisfăcut atunci are loc și P' ”*;
 - **negația:** \neg ;
 $\neg b$ are semnificația: *“negația expresiei logice b ”*;
- contribuțiile lui Hoare:
 - **triplet** – precondiție, bloc de instrucțiuni, postcondiție;
 - **axioma atribuirii** pentru: instrucțiunea de atribuire;
 - **reguli deductive** pentru: structura secvențială, structura alternativă și structura repetitivă;
 - **demonstrarea parțial și total corectitudinii, dezvoltarea corectă a algoritmilor folosind triplete.**

Triplete Hoare

- $\{\varphi\} P \{\psi\}$ – triplet Hoare, unde:
 - φ este precondiția;
 - ψ este postcondiția;
- notația are semnificația:
“dacă execuția programului P începe dintr-o stare care satisface φ , atunci starea în care se ajunge după execuția lui P va satisface ψ ”;

Triplete Hoare. Exemple (1)

- Care dintre următoarele triplete sunt valide?
 - 1 $\{x = 5\} \ x := x * 2 \ \{true\};$
 - 2 $\{x = 5\} \ x := x * 2 \ \{x > 0\};$
 - 3 $\{x = 5\} \ x := x * 2 \ \{x = 10 \ || \ x = 5\};$
 - 4 $\{x = 5\} \ x := x * 2 \ \{x = 10\};$
- toate tripletele sunt valide;
- $\{x = 5\} \ x := x * 2 \ \{x = 10\}$ – cel mai util triplet;
- $\{x = 10\}$ – *cea mai puternică postcondiție*.

Triplete Hoare. Exemple (2)

- Care dintre următoarele triplete sunt valide?
 - 1 $\{x = 5 \ \&\& \ y = 10\} \ z := x/y \ \{z < 1\};$
 - 2 $\{x < y \ \&\& \ y > 0\} \ z := x/y \ \{z < 1\};$
 - 3 $\{y \neq 0 \ \&\& \ x/y < 1\} \ z := x/y \ \{z < 1\};$
- toate tripletele sunt valide;
- $\{y \neq 0 \ \&\& \ x/y < 1\} \ z := x/y \ \{z < 1\}$ – cel mai util triplet;
- $\{y \neq 0 \ \&\& \ x/y < 1\}$ – cea mai slabă condiție.

Semantica tripletelor Hoare

● corectitudine parțială

- notăție: $\models_{par} \{\varphi\}P\{\psi\}$
- **tripletul $\{\varphi\}P\{\psi\}$ este satisfăcut relativ la corectitudinea parțială**, dacă pentru orice stare care satisface φ , starea rezultată după execuția programului P satisface postcondiția ψ , *având condiția că programul se termină*;
- nu garantează că P se termină;

● corectitudine totală

- notăție: $\models_{tot} \{\varphi\}P\{\psi\}$
- **tripletul $\{\varphi\}P\{\psi\}$ este satisfăcut relativ la corectitudinea totală**, dacă pentru orice stare care satisface φ , programul P se termină, iar starea rezultată după execuția programului P satisface postcondiția ψ ;
- garantează că P se termină.

Parțial corectitudine. Reguli deductive

- axioma atribuirii;
- regula compunerii secvențiale;
- regula consecinței;
- regula alternanței;
- regula iterației.

Axioma atribuirii

- $\models_{par} \{\varphi(x|e)\} x := e \{\psi(x)\}$
are semnificația *“dacă prin înlocuirea lui x în $\varphi(x)$ cu e obținem o afirmație adevărată, atunci după atribuirea $x := e$ afirmația $\psi(x)$ va fi adevărată.”*
- Fie tripletul $\{P\} X := Y + 2 \{Q\}$
 - fiind dat Q , care este predicatul pentru care P are loc?
 - pentru orice P astfel încât $[P \Rightarrow (X \leftarrow Y + 2) (Q)]$

Regula compunerii secvențiale

- dacă $\models_{par} \{\varphi\} S \{w\}$ și $\models_{par} \{w\} T \{\psi\}$
atunci $\models_{par} \{\varphi\} S; T \{\psi\}$;

Regula consecinței

- dacă

$\varphi_1 \Rightarrow \varphi_2, \models_{par} \{\varphi_2\} P\{\psi_2\}$ și $\psi_2 \Rightarrow \psi_1$

atunci

$\models_{par} \{\varphi_1\} P\{\psi_1\}$

Regula alternanței

- dacă

$\models_{par} \{\varphi \wedge \text{cond}\} S\{\psi\}$ și $\models_{par} \{\varphi \wedge \neg \text{cond}\} T\{\psi\}$

atunci propoziția

$\{\varphi\} \text{ IF } (\text{cond}) \text{ THEN } S \text{ ELSE } T \text{ END}\{\psi\}$ este parțial corectă în raport cu specificația (φ, ψ) .

Regula iterației

- Care sunt condițiile de realizare ale structurii repetitive `while`, astfel încât:
 $\{\varphi\} \text{ WHILE } (cond) \text{ DO } S \text{ END } \{\psi\}$
 - presupunem că instrucțiunea `while` se termină , i.e., $\neg cond$;
 - în general, nu se cunoaște de câte ori se va executa `S`;
- considerăm un predicat η care rămâne satisfăcut după execuția `S`:
 - $\{\eta\} S \{\eta\}$ η este un predicat invariant;
 - la ieșirea din buclă avem $\eta \wedge \neg cond$;
 - pentru stabilirea post-condiției, $\{\eta\}$ trebuie ales astfel încât $[\eta \wedge \neg cond \Rightarrow \psi]$.

Regula iterației (cont.)

- dacă $\models_{par} \{\varphi \wedge cond\} S \{\psi\}$
atunci $\{\varphi\} \text{ WHILE } (cond) \text{ DO } S \text{ END } \{\psi\}$,
cu condiția că există un predicat invariant η asociat buclei, astfel încât:
 - $[\varphi \Rightarrow \eta]$ η este satisfăcut la intrare în buclă;
 - $[\eta \wedge \neg cond \Rightarrow \psi]$ η obține pe ψ la ieșirea din buclă;
 - $\{cond \wedge \eta\} S \{\eta\}$ η este satisfăcut la fiecare iterație.

Regula iterației. Exemple

Demonstrarea parțial corectitudinii folosind regula iterației:

- **Exemplu 1.** $z = 2^N$;

Dezvoltarea algoritmilor (parțial corecți), folosind regula iterației:

- **Exemplu 2.** $R = A * B$;
- **Exemplu 3.** $R = A^B$.

Regula iterației. Exemplu 1.

- efectuarea calculului: $z = 2^N$:
 - $\varphi : \{N \geq 0\}$
 $m := 0; y := 1;$
 $\eta : \{y = 2^m\}$
 $WHILE (m \neq N) DO \eta : \{y = 2^m\}$
 $y := 2 * y;$
 $m := m + 1$
 END
 $\psi : \{y = 2^N\}$
- trebuie demonstrat că invariantul η
 - este satisfăcut la intrare în buclă;
 - rămâne satisfăcut în buclă $\{\eta\} y := 2 * y; m := m + 1; \{\eta\}$
 - obține post-condiția $[\eta \wedge (m = N) \Rightarrow (y = 2^N)]$.

Regula iterației. Exemplu 2.

- înmulțire prin adunări repetate – “ R este A adunat de B ori”: $R = A * B$:

- $\varphi : \{B \geq 0\}$
- $\psi : \{R = A * B\} \Rightarrow \{B \geq 0\}$ $S\{R = A * B\}$
- rezolvare (dezvoltarea tripletului):

$\varphi : \{B \geq 0\}$
“init R ”
WHILE (cond) DO
“update R ”
END
 $\psi : \{R = A * B\}$

- **regulă:** se înlocuiește în postcondiția ψ unul din termeni cu o variabilă pentru a obține predicatul invariant η asociat buclei, astfel încât $[(\eta \wedge \neg \text{cond}) \Rightarrow \psi]$;
 - se introduce variabila b în ψ și se determină invariantul η asociat buclei, descris prin: $R = A * b$;
 - pentru a obține postcondiția, se alege cond să fie $(b \neq B)$, unde $[(R = A * b) \wedge \neg(b \neq B) \Rightarrow (R = A * B)]$.

Regula iterației. Exemplu 2 (cont.)

- înmulțire prin adunări repetate:
 - invariantul – $\eta: (R = A * b)$;
 - condiția de execuție a buclei (santinela) – $\text{cond}: (b \neq B)$;
 - pentru a asigura că invariantul este satisfăcut inițial, se efectuează inițializarea: $R := 0; b := 0$;
 - în fiecare iterație: (1) b este incrementat cu 1; (2) R este actualizat, obținând:

$\varphi: \{B \geq 0\}$

$R := 0; b := 0;$

WHILE $(b \neq B)$ DO $\eta: \{R = A * b\}$

$R := ? \Rightarrow R := R + A$

$b := b + 1$

END

$\psi: \{R = A * B\}$

Regula iterativă. Exemplu 3.

- ridicare la putere prin înmulțiri repetate – “ R este A înmulțit de B ori”:

$R = A^B$;

- $\{\varphi : (A > 0) \wedge (B \geq 0)\} \text{ S } \{\psi : R = A^B\}$
- rezolvare (dezvoltarea tripletului):
 - pentru obținerea invariantului se înlocuiește în ψ o constantă cu o variabilă, obținându-se: $\eta : R = A^b$;

$\varphi : \{(A > 0) \wedge (B \geq 0)\}$

$R := ?; b := 0; \Rightarrow R := 1;$

WHILE $(b \neq B)$ DO $\eta : \{R = A^b\}$

$R := ?; \Rightarrow R := R * A;$

$b := b + 1$

END

$\psi : \{R = A^B\}$

Total corectitudine. Reguli deductive

- **atribuire**

$\{\varphi\} X := E \{\psi\}$ cu condiția că $[\varphi \Rightarrow (X \leftarrow E)(\psi)]$;

- **compunere**

$\{\varphi\} S; T \{\psi\}$ cu condiția că
există R astfel încât $\{\varphi\} S \{R\}$ și $\{R\} T \{\psi\}$;

- **alternanță**

$\{\varphi\} IF (cond) THEN S ELSE T END \{\psi\}$ cu condiția că
 $\{\varphi \wedge cond\} S \{\psi\}$ și $\{\varphi \wedge \neg cond\} T \{\psi\}$

- **Observație:** similar cu regulile corectitudinii parțiale!

Total corectitudine. Iterația.

- fie tripletul $\{\varphi\} \text{ WHILE } (cond) \text{ DO } S \text{ END } \{\psi\}$
- cum demonstrăm că execuția buclei se termină?
- soluție:
 - se identifică o expresie întreagă V astfel încât:
 - valoarea V este non-negativă (i.e., $V \geq 0$) și
 - valoarea V este strict descrescătoare la fiecare iterație, $\{V = K\} S \{V < K\}$
- V – “invariant al buclei”, expresia își păstrează caracteristicile de la o iterație la alta.

Total corectitudine. Exemplu

- ridicare la putere prin înmulțiri repetate – “ R este A înmulțit de B ori”:

$$R = A^B;$$

- $\{(A > 0) \wedge (B \geq 0)\} \text{ S } \{R = A^B\}$
- invariantul buclei este: $\eta : R = A^b \wedge (B \geq b)$;
 $\varphi : \{(A > 0) \wedge (B \geq 0)\}$
 $R := 1; b := 0;$
WHILE $(b \neq B)$ **DO** $\eta : R = A^b \wedge (B \geq b);$
 $R := R * A;$
 $b := b + 1$
END
 $\psi : \{R = A^B\}$

- se definește V – o construcție care variază la nivelul buclei – descris prin expresia $(B - b)$;
- V este strict descrescătoare la fiecare iterație a buclei, deoarece $[(B - (b + 1)) < (B - b)]$
- Cum demonstrăm că V este o expresie non-negativă?
 - demonstrând că $(B \geq b)$ este un invariant al buclei.

Total corectitudine. Regula iterației (rezumat)

- pentru a demonstra

$\models_{tot} \{\varphi\} \text{ WHILE } (cond) \text{ DO } S \text{ END } \{\psi\}$

se identifică un predicat invariant η al buclei și o expresie V , invariantă la nivelul buclei, astfel încât:

- η este satisfăcut inițial $[\varphi \Rightarrow \eta]$;
- η determină obținerea post-condiției prin condiția de ieșire din buclă $[(\eta \wedge \neg cond) \Rightarrow \psi]$;
- η se menține satisfăcut după execuția blocului S , i.e., $\{\eta\} S \{\eta\}$;
- expresia V este strict descrescătoare la fiecare iterație $\{V = K\} S \{V < K\}$;
- expresia V este întotdeauna non-negativă; $[\eta \Rightarrow (V \geq 0)]$.

Pentru examen...

- metoda lui Floyd:
 - demonstrarea parțial corectitudinii, terminării și total corectitudinii ([Fre10], Cap.1) – probleme:
 - căutarea unei valori într-un șir ordonat (**Seminar 5**);
 - determinarea celui mai mare divizor comun a două numere naturale (**Seminar 5**);

Urmează...

- Limbajul Dijkstra

Bibliografie I

- [Fre10] M. Frentiu.
Verificarea și validarea sistemelor soft.
Presa Universitară Clujeană, 2010.