Some Strategies for the Semi-Automatic Use of Hoare Logic

Juliana Carpes Im

Edward Hermann Hae

XIV EBL

OUTLINE

- Introduction;
- Hoare Logic rules used;
- The strategies;
- Correctness and Completeness;
- Correctness of a loop without invariant;
- Problem with arrays; and
- Conclusions.

INTRODUCTION

- Proving the correctness of a program: propositional and first-order problems (size, time and undecidability);
- Finding non-trivial properties is undecidable (Rice's Theorem);
- Hoare Logic: the task of finding the stronger invariant of a loop is undecidable;
- Using heuristics;
- The aim is to find a set of strategies for a semi-automatic Hoare Logic prover to minimize time, human interference and calls to a first-order prover;

INTRODUCTION

- Use strategies to reduce the search space so that the proof is constructed in a more efficient way;
- Some rules have first-order sentences that must be proved;
- Some proofs can be syntactically co with non-demonstrable sentences;
- So, it is possible to have many proofs t satisfy the rules but with non-demonstrable sentences;
- It is worth trying to find valid sentences without using a theorem prover; and

INTRODUCTION

$$\frac{\{true\}\ a := 5\ \{true\} \quad true \Rightarrow a = 5}{\{true\}\ a := 5\ \{a = 5\}}$$

$$\frac{true \Rightarrow 5 = 5 \quad \{5 = 5\}\ a := 5\ \{a = 5\}}{\{true\}\ a := 5\ \{a = 5\}}$$

- Demonstrating sentences is often in[illegible]nt and can lead the prover to an infinite l[illegible]

HOARE LOGIC RULES USED

$$\frac{\{P \wedge B\}\ C_1\ \{Q\} \qquad \{P \wedge \neg B\}\ C_2\ \{Q\}}{\{P\}\ \mathit{if}\ B\ \mathit{then}\ C_1\ \mathit{else}\ C_2\ \mathit{fi}\ \{Q\}}$$

$$\frac{\{P \wedge B\}\ C\ \{Q\} \qquad P \wedge \neg B \rightarrow Q}{\{P\}\ \mathit{if}\ B\ \mathit{then}\ C\ \mathit{fi}\ \{Q\}}$$

$$\frac{}{\{P(V/E)\}\ V := E\ \{P\}}$$

$$\frac{\{P\}\ C_1\ \{R\} \qquad \{R\}\ C_2\ \{Q\}}{\{P\}\ C_1; C_2\ \{Q\}}$$

$$\frac{P \rightarrow R \qquad \{R\}\ C\ \{Q\}}{\{P\}\ C\ \{Q\}}$$

$$\frac{}{\{P\}\ \mathit{skip}\ \{P\}}$$

$$\frac{\{P \wedge B\}\ C\ \{P\}}{\{P\}\ \mathit{while}\ B\ \mathit{do}\ C\ \mathit{od}\ \{P \wedge \neg B\}}$$

$$\frac{\{P\}\ C\ \{R\} \qquad R \rightarrow Q}{\{P\}\ C\ \{Q\}}$$

THE STRATEGIES

- **Assumption: the program does not have arrays;**
- **The loop's invariants are given by the user;**
- **From the right to the left:**
  - **The post-condition is known;**
  - **If a pre-condition is found, it is weakened:**
  - $$\frac{P \Rightarrow Q \quad \{ Q \}\ C\ \{ R \}}{\{ P \}\ C\ \{ R \}}$$ **P is known**
  - $$\frac{P \wedge B \Rightarrow R \quad \{ R \}\ C\ \{ P \}}{\{ P \wedge B \}\ C\ \{ P \}} \qquad \frac{}{\{ P \}\ \textbf{while } B \textbf{ do } C \textbf{ od } \{ P \wedge \neg B \}}$$ **loop**

THE STRATEGIES

$$\frac{\{ P \} \text{ while } \ldots \{ P \wedge \neg B \} \quad P \wedge \neg B \Rightarrow R \qquad \{ P \} \text{ while } \ldots \{ R \} \quad \{ R \} C \{ Q \}}{\{ P \} \text{ while } \ldots ; C \{ Q \}}$$

sequence of commands after a loop

- To the while, skip and assignment commands it is only necessary to apply the right rule;
- To the sequence of commands too, unless the first one is a while command; and
- if-then-else command:

$$\frac{P \wedge B \rightarrow P_1 \quad \{P_1\}\ C_1\ \{Q\} \qquad P \wedge \neg B \rightarrow P_2 \quad \{P_2\}\ C_2\ \{Q\}}{\{P \wedge B\}\ C_1\ \{Q\} \qquad \{P \wedge \neg B\}\ C_2\ \{Q\}}$$

$$\{P\}\ \textit{if } B \textit{ then } C_1 \textit{ else } C_2 \textit{ fi}\ \{Q\}$$

$P = (B \rightarrow P_1) \wedge (\neg B \rightarrow P_2)$ or $P = (B \wedge P_1) \vee (\neg B \wedge P_2)$

THE STRATEGIES FROM THE LEFT TO THE RIGHT

- Only used if the post-condition is not known and only after a loop;
- The strategies find the strongest po condition for each command;
- The skip command is proved correct using its rule and the while command is proved correct using the other approach;
- The post-condition of a command is the pre-condition of the next one;

THE STRATEGIES FROM THE LEFT TO THE RIGHT

- **Assignments:**
  - **`{ P } x := E { Q };`**
  - **`{ P } x := E( x ) { Q };`**
  - **`{ P( x ) } x := E { Q };` and**
  - **`{ P( x ) } x := E( x ) { Q };`**

$$\frac{P \rightarrow (P \wedge E = E) \qquad \{P \wedge E = E\}\ x := E\ \{P \wedge x = E\}}{\{P\}\ x := E\ \{P \wedge x = E\}}$$

$$\frac{P \rightarrow (P \wedge \exists y(E(x) = E(y))) \quad \{P \wedge \exists y(E(x) = E(y))\}\ x := E(x)\ \{P \wedge \exists y(x = E(y))\}}{\{P\}\ x := E(x)\ \{P \wedge \exists y(x = E(y))\}}$$

THE STRATEGIES FROM THE LEFT TO THE RIGHT

$$P(x) \to (P(a) \wedge E = E) \qquad \{P(a) \wedge E = E\}\ x := E\ \{P(a) \wedge x = E$$

$$\{P(x)\}\ x := E\ \{P(a) \wedge x = E\}$$

$$P(x) \to (\exists a P(a) \wedge E = E) \qquad \{\exists a P(a) \wedge E = E\}\ x := E\ \{\exists a P(a) \wedge x = E$$

$$\{P(x)\}\ x := E\ \{\exists a P(a) \wedge x = E\}$$

$$P(x) \to (P(a) \wedge E(x) = E(a)) \qquad \{P(a) \wedge E(x) = E(a)\}\ x := E(x)\ \{P(a) \wedge x = E(a)$$

$$\{P(x)\}\ x := E(x)\ \{P(a) \wedge x = E(a)\}$$

$$(x) \to \exists a(P(a) \wedge E(x) = E(a)) \qquad \{\exists a(P(a) \wedge E(x) = E(a))\}\ x := E(x)\ \{\exists a(P(a) \wedge x = E(a)$$

$$\{P(x)\}\ x := E(x)\ \{\exists a(P(a) \wedge x = E(a))\}$$

THE STRATEGIES FROM THE LEFT TO THE RIGHT

- **If-then-else:**
  - **If P has information to evaluate B, th[illegible]one between P Ù B and P Ù +B is a contrad[illegible]; and**
  - **A Ú ^º A .**

$$\frac{\dfrac{\{P \wedge B\}\ C_1\ \{Q\} \qquad Q \rightarrow Q \vee R}{\{P \wedge B\}\ C_1\ \{Q \vee R\}} \qquad \dfrac{\{P \wedge \neg B\}\ C_2\ \{R\} \qquad R \rightarrow Q \vee R}{\{P \wedge \neg B\}\ C_2\ \{Q \vee R\}}}{\{P\}\ \mathit{if}\ B\ \mathit{then}\ C_1\ \mathit{else}\ C_2\ \mathit{fi}\ \{Q \vee R\}}$$

LOOP'S CORRECTNESS WITHOUT INVARIANT

$$\dfrac{\dfrac{\dfrac{\dfrac{I \wedge B \rightarrow R \qquad \{R\}\ C\ \{P\}}{\{I \wedge B\}\ C\ \{P\}} \qquad P \rightarrow I}{\{I \wedge B\}\ C\ \{I\}}}{\{I\}\ \mathit{while}\ B\ \mathit{do}\ C\ \mathit{od}\ \{I \wedge \neg B\}}\ I \wedge \neg B \rightarrow Q}{P \rightarrow I \qquad \{I\}\ \mathit{while}\ B\ \mathit{do}\ C\ \mathit{od}\ \{Q\}}$$

$$\{P\}\ \mathit{while}\ B\ \mathit{do}\ C\ \mathit{od}\ \{Q\}$$

LOOP'S CORRECTNESS WITHOUT INVARIANT

- **P must be known;**
- **I Ù B ® R and P ® I Þ P Ù B ®**
- **I Ù ØB ® R and P ® I Þ P Ù ØB**
- **I may be the interpolant;**
- **In some cases, the invariant is not necessary; and**
- **The need of user help is reduced, but the size of the proof increases.**

CORRECTNESS AND COMPLETENESS

- The sentences shown above can be proved and the proofs above obey the Hoare Logic;
- Proving the correctness from the right to the left can be considered complete because it is based in the known process Dijkstra's Weakest Pre-Condition;

CORRECTNESS AND COMPLETENESS

- So, if { P } C { Q } is proved from the right to the left and P is known, P can be weakened to the pre-condition found when proving the program this way;
- No relevant information is discarded when proving the program from the left to the right (the strongest post-condition result)
- So, if { P } C { Q } is proved from the left to the right and Q is known, the post-condition found when proving the program this way can be weakened to Q;

CORRECTNESS AND COMPLETENESS

- **Lemma 1: The strategies for the assignments produce the strongest post-condition;**
- **Lemma 2: The strategies for the if commands produce the strongest post-condition;**
- **Theorem: Using the strategies for proving the correctness from the left to the right always produce the strongest post-condition for the program;**
- **The concept of information is used;**

CORRECTNESS AND COMPLETENESS

- When using the strategies it is only possible to have one proof of correctness;
- When proving the correctness of a command manually, one can weaken pre-condition, strengthen the post-condition (if they are known) or apply a rule to the command being verified;
- The weakenings can be put upwards or downwards the proof if they are not done to a while command:

CORRECTNESS AND COMPLETENESS

$$\frac{\{ R \}\ C_1\ \{ S \} \qquad \{ S \}\ C_2\ \{ Q \}}{\dfrac{P \rightarrow R \qquad \{ R \}\ C_1 ; C_2\ \{ Q \}}{\{ P \}\ C_1 ; C_2\ \{ Q \}}}$$

$$\frac{P \rightarrow R \qquad \{ R \}\ C_1\ \{ S \}}{\dfrac{\{ P \}\ C_1\ \{ S \} \qquad \{ S \}\ C_2\ \{ Q \}}{\{ P \}\ C_1 ; C_2\ \{ Q \}}}$$

$$\frac{\dfrac{\{ P \}\ C_1\ \{ S \} \qquad \{ S \}\ C_2\ \{ R \}}{\{ P \}\ C_1 ; C_2\ \{ R \}} \qquad R \rightarrow Q}{\{ P \}\ C_1 ; C_2\ \{ Q \}}$$

$$\frac{\{ P \}\ C_1\ \{ S \} \qquad \dfrac{\{ S \}\ C_2\ \{ R \} \qquad R \rightarrow Q}{\{ S \}\ C_2\ \{ Q \}}}{\{ P \}\ C_1 ; C_2\ \{ Q \}}$$

CORRECTNESS AND COMPLETENESS

$$\frac{\{ R \wedge B \}\ C_1\ \{ Q \} \qquad \{ R \wedge \neg B \}\ C_2\ \{ Q \}}{P \rightarrow R \qquad \{ R \}\ \text{if } B \text{ then } C_1 \text{ else } C_2 \text{ fi}\ \{ Q \}}$$
$$\{ P \}\ \text{if } B \text{ then } C_1 \text{ else } C_2 \text{ fi}\ \{ Q \}$$

$$\frac{P \wedge B \rightarrow R \wedge B \quad \{ R \wedge B \}\ C_1\ \{ Q \} \qquad P \wedge \neg B \rightarrow R \wedge \neg B \quad \{ R \wedge \neg B \}\ C_2\ \{ Q \}}{\{ P \wedge B \}\ C_1\ \{ Q \} \qquad \{ P \wedge \neg B \}\ C_2\ \{ Q \}}$$
$$\{ P \}\ \text{if } B \text{ then } C_1 \text{ else } C_2 \text{ fi}\ \{ Q \}$$

$$\frac{\{ P \wedge B \}\ C_1\ \{ R \} \qquad \{ P \wedge \neg B \}\ C_2\ \{ R \}}{\{ P \}\ \text{if } B \text{ then } C_1 \text{ else } C_2 \text{ fi}\ \{ R \} \qquad R \rightarrow Q}$$
$$\{ P \}\ \text{if } B \text{ then } C_1 \text{ else } C_2 \text{ fi}\ \{ Q \}$$

$$\frac{\{ P \wedge B \}\ C_1\ \{ R \} \quad R \rightarrow Q \qquad \{ P \wedge \neg B \}\ C_2\ \{ R \} \quad R \rightarrow Q}{\{ P \wedge B \}\ C_1\ \{ Q \} \qquad \{ P \wedge \neg B \}\ C_2\ \{ Q \}}$$
$$\{ P \}\ \text{if } B \text{ then } C_1 \text{ else } C_2 \text{ fi}\ \{ Q \}$$

PROBLEM WITH ARRAYS

- When arrays are included, the construction of proofs is not automatically feasible anymore;
- The strategies shown above might not behave properly with assignments a[i] := E(a, i);
- The properties describing arrays are usually related with all its elements and not with a specific one;
- User interaction is used in the implementation;

PROBLEM WITH ARRAYS

- **Example:**

$$\frac{P \rightarrow Q \quad \{ Q \}\ a[\,i\,] := i\ \{ Q \}}{\{ P \}\ a[\,i\,] := i\ \{ Q \}}$$

$$\frac{\{P \wedge i = i\}\ a[i] := i\ \{P \wedge a[i] = i\} \qquad P \wedge a[i] = i \rightarrow Q}{\dfrac{P \rightarrow P \wedge i = i \qquad \{P \wedge i = i\}\ a[i] := i\ \{Q\}}{\{P\}\ a[i] := i\ \{Q\}}}$$

where $P = \forall j(j < i \rightarrow a[j] = j)$ and $Q = \forall j(j < i + 1 \rightarrow a[j] = j)$.

PROBLEM WITH ARRAYS

- **Reducing the array problem to the invariant problem;**
- **The other positions are not changed;**
- **The assignment to an array position also has an invariant;**

$$\dfrac{\dfrac{}{\{\, I \wedge E(a) = E(a)\,\}\ a[i] := E(a)\ \{\, I \wedge a[i] = E(a)\,\}} \qquad I \wedge a[i] = E(a) \rightarrow I \wedge E(a) = E(a)}{\{\, I \wedge E(a) = E(a)\,\}\ a[i] := E(a)\ \{\, I\,\}}$$

$$\dfrac{\{\, I \wedge B\,\}\ a[i] := E(a)\ \{\, I\,\}}{\{\, I\,\}\ \text{while } B \text{ do } a[i] := E(a) \text{ od } \{\, I \wedge \neg B\,\}}$$

PROBLEM WITH ARRAYS

- **Remove ®, « , $ and non-atomic Ø (whenever possible);**
- **Only " , Ù, Ú, and atomic Ø;**
- **Put inside Ø expressions;**
- **From the right to the left: only use the array is not mentioned in the post-condition;**

PROBLEM WITH ARRAYS

- From the left to the right: weaken the pre-condition;
- " when it is alone or inside a Ù,

$$\forall j \; P(i, j, a) \rightarrow \forall j \; (i \neq j \rightarrow P(i, j, a))$$

- Now the proof can be done using one o the strategies for the assignment; and
- In the other situations, I may need the hel from the user.

CONCLUSIONS

- Because of the array problem and loops' invariants, user interaction is needed;
- Implementation in Prolog validated the strategies;
- If there are no arrays, when a proof of { P } C { Q } exists, the prototype shows one, and every sentence is provable;
- Future steps:
  - Provide a reasonable graphic output; and
  - Find more strategies for arrays.