CS 292C Computer-Aided Reasoning for Software

Lecture 11: Reasoning about Programs using Hoare logic I

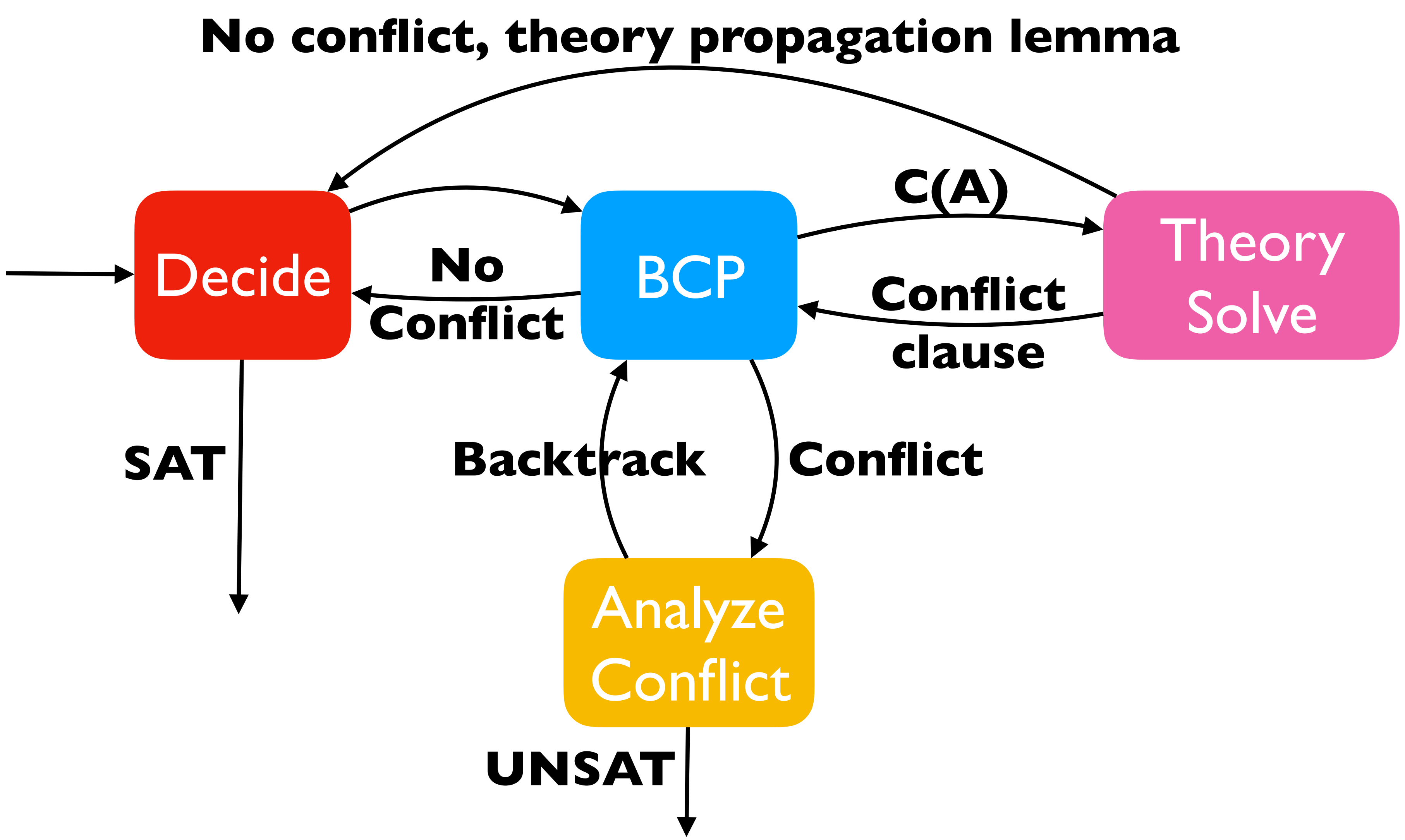
Yu Feng

Fall 2019

Summary of previous lecture

- The 4th reading assignment is due now
- First half of the class: foundation of SAT/SMT solvers
- DPLL(T) algorithm

Overview of DPLL(T)



Outline of this week

- Reasoning about (partial) correctness of programs
  - Hoare Logic (today)
  - Verification with Dafny (next lecture)

History of Hoare logic

- **1967**: Assigning Meaning to Programs (Floyd)
  - 1978 Turing Award

- **1969**: An Axiomatic Basis for Computer Programming (Hoare)
  - 1980 Turing Award

- **1975**: Guarded Commands, Nondeterminacy and Formal Derivation of Programs (Dijkstra)
  - 1972 Turing Award

Simple Imperative Programming Language

Expression E

- $Z \mid V \mid E_1 + E_2 \mid E_1 * E_2$

Conditional C

True | False | $E_1 = E_2 \mid E_1 \leq E_2$

Statement S

- skip (Skip)
- abort. (Abort)
- V := E (Assignment)
- $S_1; S_2$. (Composition)
- **if** C **then** S_1 **else** S_2 (If)
- **while** C **do** S (While)

Simple Imperative Programming Language

Expression E

- $Z \mid V \mid E_1 + E_2 \mid E_1 * E_2$

Conditional C

True | False | $E_1 = E_2 \mid E_1 \leq E_2$

A minimalist programming language for demonstrating key features of Hoare logic.

Statement S

- skip (Skip)
- abort. (Abort)
- V := E (Assignment)
- $S_1; S_2$. (Composition)
- **if** C **then** S_1 **else** S_2 (If)
- **while** C **do** S (While)

Specifying correctness in Hoare logic

- Hoare triple
  - S is a program statement (in IMP).
  - P and Q are FOL formulas over program variables.
  - P is called a ***precondition*** and Q is a ***postcondition***.

{P} S {Q}

- Partial correctness
  - If S executes from a state satisfying P, and if its execution terminates, then the resulting state satisfies Q.
- Total correctness
  - If S executes from a state satisfying P, then its execution terminates and the resulting state satisfies Q.

[P] S [Q]

Specifying correctness in Hoare logic

- Hoare triple
  - S is a program statement (in IMP).
  - P and Q are FOL formulas over program variables.
  - P is called a ***precondition*** and Q is a ***postcondition***.
- Partial correctness
  - If S executes from a state satisfying P, and if its execution terminates, then the resulting state satisfies Q.
- Total correctness
  - If S executes from a state satisfying P, then its execution terminates and the resulting state satisfies Q.

{P} S {Q}

Safety

[P] S [Q]

Specifying correctness in Hoare logic

- Hoare triple
  - S is a program statement (in IMP).
  - P and Q are FOL formulas over program variables.
  - P is called a ***precondition*** and Q is a ***postcondition***.
- Partial correctness
  - If S executes from a state satisfying P, and if its execution terminates, then the resulting state satisfies Q.
- Total correctness
  - If S executes from a state satisfying P, then its execution terminates and the resulting state satisfies Q.

{P} S {Q}

Safety

[P] S [Q]

Specifying correctness in Hoare logic

- Hoare triple
  - S is a program statement (in IMP).
  - P and Q are FOL formulas over program variables.
  - P is called a ***precondition*** and Q is a ***postcondition***.
- Partial correctness
  - If S executes from a state satisfying P, and if its execution terminates, then the resulting state satisfies Q.
- Total correctness
  - If S executes from a state satisfying P, then its execution terminates and the resulting state satisfies Q.

{P} S {Q}

Safety

[P] S [Q]

Liveness

Specifying correctness in Hoare logic

- Hoare triple
  - S is a program statement (in IMP).
  - P and Q are FOL formulas over program variables.
  - P is called a ***precondition*** and Q is a ***postcondition***.
- Partial correctness
  - If S executes from a state satisfying P, and if its execution terminates, then the resulting state satisfies Q.
- Total correctness
  - If S executes from a state satisfying P, then its execution terminates and the resulting state satisfies Q.

{P} S {Q}

Safety

[P] S [Q]

Liveness

Examples of Hoare triples

Examples of Hoare triples

{false} S {Q}

Examples of Hoare triples

{false} S {Q}

- Valid for all S and Q.

Examples of Hoare triples

{false} S {Q}

- Valid for all S and Q.

{P} while (true) do skip {Q}

Examples of Hoare triples

{false} S {Q}

- Valid for all S and Q.

{P} while (true) do skip {Q}

- Valid for all P and Q.

Examples of Hoare triples

{false} S {Q}

- Valid for all S and Q.

{P} while (true) do skip {Q}

- Valid for all P and Q.

{true} S {Q}

Examples of Hoare triples

{false} S {Q}

- Valid for all S and Q.

{P} while (true) do skip {Q}

- Valid for all P and Q.

{true} S {Q}

- If S terminates, the resulting state satisfies Q.

Examples of Hoare triples

{false} S {Q}

- Valid for all S and Q.

{P} while (true) do skip {Q}

- Valid for all P and Q.

{true} S {Q}

- If S terminates, the resulting state satisfies Q.

{P} S {true}

Examples of Hoare triples

{false} S {Q}

- Valid for all S and Q.

{P} while (true) do skip {Q}

- Valid for all P and Q.

{true} S {Q}

- If S terminates, the resulting state satisfies Q.

{P} S {true}

- Valid for all P and S.

Simple Imperative Programming Language

Expression E

- $Z \mid V \mid E_1 + E_2 \mid E_1 * E_2$

Conditional C

True | False | $E_1 = E_2 \mid E_1 \leq E_2$

Statement S

- skip (Skip)
- abort. (Abort)
- V := E (Assignment)
- $S_1; S_2$. (Composition)
- **if** C **then** S_1 **else** S_2 (If)
- **while** C **do** S (While)

Simple Imperative Programming Language

Expression E

- Z |V | $E_1 + E_2$ | $E_1 * E_2$

Conditional C

True | False | $E_1 = E_2$ | $E_1 \leq E_2$

Statement S

- skip (Skip)
- abort. (Abort)
- V := E (Assignment)
- $S_1; S_2$. (Composition)
- **if** C **then** S_1 **else** S_2 (If)
- **while** C **do** S (While)

One inference rule for every statement in the language:

Simple Imperative Programming Language

Expression E

- $Z \mid V \mid E_1 + E_2 \mid E_1 * E_2$

Conditional C

True | False | $E_1 = E_2 \mid E_1 \leq E_2$

Statement S

- skip (Skip)
- abort. (Abort)
- V := E (Assignment)
- $S_1; S_2$. (Composition)
- **if** C **then** S_1 **else** S_2 (If)
- **while** C **do** S (While)

One inference rule for every statement in the language:

$$\frac{\vdash\{P_1\}S_1\{Q_1\} \;\ldots\; \vdash\{P_n\}S_n\{Q_n\}}{\vdash\{P\}S\{Q\}}$$

Simple Imperative Programming Language

Expression E

- $Z \mid V \mid E_1 + E_2 \mid E_1 * E_2$

Conditional C

True | False | $E_1 = E_2 \mid E_1 \leq E_2$

Statement S

- skip (Skip)
- abort. (Abort)
- V := E (Assignment)
- $S_1; S_2$. (Composition)
- **if** C **then** S_1 **else** S_2 (If)
- **while** C **do** S (While)

One inference rule for every statement in the language:

$$\frac{\vdash\{P_1\}S_1\{Q_1\} \ \ldots \ \vdash\{P_n\}S_n\{Q_n\}}{\vdash\{P\}S\{Q\}}$$

If Hoare triples $\{P_1\}S_1\{Q_1\},\ldots,\{P_n\}S_n\{Q_n\}$ are provable in our proof system, then $\{P\}S\{Q\}$ is also provable.

Hoare logic rules

$$\frac{}{\vdash \{P\}\ \mathrm{Skip}\ \{P\}}$$

$$\frac{\vdash \{P\}\ S_1\ \{R\} \qquad \vdash \{R\}\ S_2\ \{Q\}}{\vdash \{P\}\ S_1; S_2\ \{Q\}}$$

$$\frac{}{\vdash \{\mathrm{true}\}\ \mathrm{abort}\ \{\mathrm{false}\}}$$

$$\frac{\vdash \{P \wedge C\}\ S_1\ \{Q\} \qquad \vdash \{P \wedge \neg C\}\ S_2\ \{Q\}}{\vdash \{P\}\ \mathbf{if}\ C\ \mathbf{then}\ \mathbf{S_1}\ \mathbf{else}\ \mathbf{S_2}\ \{Q\}}$$

$$\frac{}{\vdash \{Q[E/x]\}\ x := E\ \{Q\}}$$

$$\frac{\vdash \{P \wedge C\}\ S\ \{P\}}{\vdash \{P\}\ \mathbf{while}\ C\ \mathbf{do}\ S\ \{P \wedge \neg C\}}$$

$$\frac{\vdash \{P_1\}\ S\ \{Q_1\} \quad P \Rightarrow P_1 \quad Q_1 \Rightarrow Q}{\vdash \{P\}\ S\ \{Q\}}$$

Hoare logic rules

$$\frac{}{\vdash \{P\}\ \text{Skip}\ \{P\}}$$

$$\frac{\vdash \{P\}\ S_1\ \{R\} \qquad \vdash \{R\}\ S_2\ \{Q\}}{\vdash \{P\}\ S_1; S_2\ \{Q\}}$$

$$\frac{}{\vdash \{\text{true}\}\ \text{abort}\ \{\text{false}\}}$$

$$\frac{\vdash \{P \wedge C\}\ S_1\ \{Q\} \qquad \vdash \{P \wedge \neg C\}\ S_2\ \{Q\}}{\vdash \{P\}\ \textbf{if}\ C\ \textbf{then}\ \mathbf{S_1}\ \textbf{else}\ \mathbf{S_2}\ \{Q\}}$$

$$\frac{}{\vdash \{Q[E/x]\}\ x := E\ \{Q\}}$$

$$\frac{\vdash \{P \wedge C\}\ S\ \{P\}}{\vdash \{P\}\ \textbf{while}\ C\ \textbf{do}\ S\ \{P \wedge \neg C\}}$$

$$\frac{\vdash \{P_1\}\ S\ \{Q_1\} \quad P \Rightarrow P_1 \quad Q_1 \Rightarrow Q}{\vdash \{P\}\ S\ \{Q\}}$$

独孤九剑

Soundness and completeness

If a Hoare triple is valid, written $\models$ {P} S {Q}, we want a proof system to prove its validity

Use notation $\vdash$ {P} S {Q} to indicate that we can prove validity of Hoare triple

Soundness:

If $\vdash$ {P} S {Q} then $\models$ {P} S {Q}

Completeness (relative)

If $\models$ {P} S {Q} then $\vdash$ {P} S {Q}

Proof rule for assignment

$$\frac{}{\vdash \{Q[E/x]\}\ x := E\ \{Q\}}$$

- To prove Q holds after assignment x := E , sufficient to show that Q with E substituted for x holds before the assignment.
- Using this rule, which of these are provable?
    - {y=4} x:=4 {y=x}
    - {x+1=n} x:=x+1 {x=n}
    - {y=x} y:=2 {y=x}
    - {z=3} y:=x {z=3}

Proof rule for assignment

$$\frac{}{\vdash \{Q[E/x]\}\ x := E\ \{Q\}}$$

- To prove Q holds after assignment x := E , sufficient to show that Q with E substituted for x holds before the assignment.
- Using this rule, which of these are provable?
  - {y=4} x:=4 {y=x} 😀
  - {x+1=n} x:=x+1 {x=n}
  - {y=x} y:=2 {y=x}
  - {z=3} y:=x {z=3}

Proof rule for assignment

$$\frac{}{\vdash \{Q[E/x]\}\ x := E\ \{Q\}}$$

- To prove Q holds after assignment x := E , sufficient to show that Q with E substituted for x holds before the assignment.
- Using this rule, which of these are provable?
  - {y=4} x:=4 {y=x} 😀
  - {x+1=n} x:=x+1 {x=n} 😀
  - {y=x} y:=2 {y=x}
  - {z=3} y:=x {z=3}

Proof rule for assignment

$$\frac{}{\vdash \{Q[E/x]\}\ x := E\ \{Q\}}$$

- To prove Q holds after assignment x := E , sufficient to show that Q with E substituted for x holds before the assignment.
- Using this rule, which of these are provable?
  - {y=4} x:=4 {y=x} 
  - {x+1=n} x:=x+1 {x=n} 
  - {y=x} y:=2 {y=x} 
  - {z=3} y:=x {z=3}

Proof rule for assignment

$$\frac{}{\vdash \{Q[E/x]\}\ x := E\ \{Q\}}$$

- To prove Q holds after assignment x := E , sufficient to show that Q with E substituted for x holds before the assignment.
- Using this rule, which of these are provable?
  - {y=4} x:=4 {y=x} 
  - {x+1=n} x:=x+1 {x=n} 
  - {y=x} y:=2 {y=x} 
  - {z=3} y:=x {z=3} 

Precondition strengthening

- Is the Hoare triple {z = 2} y := x {y = x} valid?
- Is it provable using our assignment rule?

$$\frac{\vdash \{P_1\}\ S\ \{Q\} \quad P \Rightarrow P_1}{\vdash \{P\}\ S\ \{Q\}}$$

$$\frac{\dfrac{\vdash \{y = x[x/y]\} y = x \{y = x\}}{\vdash \{true\} y := x \{y = x\}} \qquad z = 2 \Rightarrow true}{\vdash \{z = 2\} y := x \{y = x\}}$$

Precondition strengthening

- Is the Hoare triple {z = 2} y := x {y = x} valid?
- Is it provable using our assignment rule?

$$\frac{\vdash \{P_1\}\ S\ \{Q\} \quad P \Rightarrow P_1}{\vdash \{P\}\ S\ \{Q\}}$$

Precondition strengthening

$$\frac{\dfrac{\vdash \{y = x[x/y]\} y = x \{y = x\}}{\vdash \{true\} y := x \{y = x\}} \qquad z = 2 \Rightarrow true}{\vdash \{z = 2\} y := x \{y = x\}}$$

Postcondition weakening

$$\frac{\vdash \{P\}\ S\ \{Q_1\} \quad Q_1 \Rightarrow Q}{\vdash \{P\}\ S\ \{Q\}}$$

- Suppose we can prove {true} S $\{x = y \wedge z = 2\}$.
- Which of these can be proved?
  - {true} S $\{x=y\}$
  - {true} S $\{z = 2\}$
  - {true} S $\{z > 0\}$
  - {true} S $\{y > 2\}$

Postcondition weakening

$$\frac{\vdash \{P\}\ S\ \{Q_1\} \quad Q_1 \Rightarrow Q}{\vdash \{P\}\ S\ \{Q\}}$$

Postcondition weakening

- Suppose we can prove {true} S $\{x = y \wedge z = 2\}$.
- Which of these can be proved?
  - {true} S {x=y}
  - {true} S {z = 2}
  - {true} S {z > 0}
  - {true} S {y > 2}

Proof rule for If statement

$$\frac{\vdash \{P \wedge C\}\ S_1\ \{Q\}}{\vdash \{P \wedge \neg C\}\ S_2\ \{Q\}}$$

$$\vdash \{P\}\ \textbf{if}\ C\ \textbf{then}\ \mathbf{S_1}\ \textbf{else}\ \mathbf{S_2}\ \{Q\}$$

- Prove the correctness of this Hoare triple
  - {true} if x > 0 then y := x else y := −x {y ≥ 0}

TODOs by next lecture

- The 2nd homework assignment will be due
- The 5th reading assignment will be out
- Start to work on your final report and project!