Lecture 3

Complete Constraint Solvers
Outline

- Introduce a simple proof theoretic framework
- Use it to define complete solvers
- Show how the standard unification problem can be interpreted as CSP
- Discuss Gauss-Jordan Elimination and Gaussian Elimination algorithms for solving linear equations over reals
Proof Theoretic Framework

- Rules that transform CSP's
  \[
  \frac{C ; \mathcal{D} \mathcal{E}}{\langle C' ; \mathcal{D} \mathcal{E}' \rangle}
  \]

- A rule
  \[
  \frac{\phi}{\psi}
  \]

  is equivalence preserving if $\phi$ and $\psi$ are equivalent

- All considered rules will be equivalence preserving
Types of Rules

Domain reduction rules
- $DE := x_1 \in D_1, \ldots, x_n \in D_n$
- $DE' := x_1 \in D'_1, \ldots, x_n \in D'_n$
- for $i \in [1..n]$
  - $D'_i \subseteq D_i$
- $C'$: restriction of all constraints in $C$ to the domains $D'_1, \ldots, D'_n$

Transformation rules
- Not domain reduction rules
- $C' \neq \emptyset$
- $DE'$ extends $DE$
Examples: Domain Reduction Rules

- **Linear Disequality**
  \[
  \langle x < y ; x \in [l_x..h_x], y \in [l_y..h_y] \rangle \\
  \langle x < y ; x \in [l_x..h'_x], y \in [l'_y..h_y] \rangle
  \]
  where \( h'_x = \min(h_x, h_y - 1), l'_y = \max(l_y, l_x + 1) \)

- **Equality**
  \[
  \langle x = y ; x \in D_x, y \in D_y \rangle \\
  \langle x = y ; x \in D_x \cap D_y, y \in D_x \cap D_y \rangle
  \]

- **Disequality**
  \[
  \langle x \neq y ; x \in D, y = a \rangle \\
  \langle ; x \in D - \{a\}, y = a \rangle
  \]
  (domain expression \( y = a \) stands for \( y \in \{a\} \))
Examples: Transformation Rules

- **Disequality Transformation**
  \[
  \begin{align*}
  &\langle s \neq t ; \mathcal{DE} \rangle \\
  \implies &\langle x \neq t , x = s ; \mathcal{DE} , x \in \mathbb{Z} \rangle
  \end{align*}
  \]
  where
  - \( s \) is not a variable
  - \( \mathcal{DE} \) includes all variables present in \( s \) and \( t \)
  - \( x \) does not appear in \( \mathcal{DE} \)

- **Variable Elimination**
  \[
  \begin{align*}
  &\langle C ; \mathcal{DE} , x = a \rangle \\
  \implies &\langle C \{x/a\} ; \mathcal{DE} , x = a \rangle
  \end{align*}
  \]
  where \( x \) occurs in \( C \)
Rule Applications

- Application of a rule (informally): replace in a CSP the part that matches the premise by the conclusion.

- Relevant application of a rule (informally): the result differs from the initial CSP.

- A CSP $\mathcal{P}$ is closed under the applications of $R$ if
  - $R$ cannot be applied to $\mathcal{P}$, or
  - no application of it to $\mathcal{P}$ is relevant.
Recap: Solved and Failed CSP's

- A constraint is **solved** if it equals the Cartesian product of the domains of its variables.
- CSP is **solved** if all its constraints are solved.
- CSP is **failed** if
  - it contains the false constraint \( \bot \), or
  - some of its domains or constraints is empty.
Derivations

Given: a finite set of proof rules

- **Derivation**: a sequence of CSP's s.t. each is obtained from the previous one by an application of a proof rule
- A finite derivation is called
  - **successful**: last element is first solved CSP in this derivation
  - **failed**: last element is first failed CSP in this derivation
  - **stabilising**: last element is first CSP closed under the applications of the proof rules
Derivation: Example

Take

- Equality

\[
\begin{align*}
\langle x = y ; x \in D_x, y \in D_y \rangle & \\
\Rightarrow & \\
\langle x = y ; x \in D_x \cap D_y, y \in D_x \cap D_y \rangle
\end{align*}
\]

- Disequality

\[
\begin{align*}
\langle x \neq y ; x \in D, y = a \rangle & \\
\Rightarrow & \\
\langle ; x \in D - \{a\}, y = a \rangle
\end{align*}
\]

and consider CSP
\[
\langle x = y, y \neq z, z \neq u; x \in \{a,b,c\}, y \in \{a,b,d\}, z \in \{a,b\}, u = b \rangle
\]
Derivation: Example, ctd

\[ \langle x = y, y \neq z, z \neq u; x \in \{a,b,c\}, y \in \{a,b,d\}, z \in \{a,b\}, u = b \rangle \]
Apply Equality rule
\[ \langle x = y, y \neq z, z \neq u; x \in \{a,b\}, y \in \{a,b\}, z \in \{a,b\}, u = b \rangle \]
Apply Disequality rule to \( z \neq u \)
\[ \langle x = y, y \neq z; x \in \{a,b\}, y \in \{a,b\}, z = a, u = b \rangle \]
Apply Disequality rule to \( y \neq z \)
\[ \langle x = y; x \in \{a,b\}, y = b, z = a, u = b \rangle \]
Apply Equality rule
\[ \langle x = y; x = b, y = b, z = a, u = b \rangle \]

Last CSP is solved: the derivation is successful
Term Equations

Alphabet
- variables
- function symbols, each with a fixed arity
- parentheses: “(” and “)"
- comma: “,”

Terms
- a variable is a term
- if \( f \) is an \( n \)-ary function symbol and \( t_1, \ldots, t \) are terms, then \( f(t_1, \ldots, t_n) \) is a term
Substitutions

- Finite mappings from variables to terms:
  \[ \{x_1/t_1, \ldots, x_n/t_n\} \]

  where
  - \(x_1, \ldots, x_n\) are different variables
  - \(t_1, \ldots, t_n\) are terms
  - for \(i \in [1..n]\), \(x_i \neq t_i\)

- \(\theta\) is more general than \(\tau\) if for some substitution \(\eta\)
  \[ \tau = \theta \eta \]
Standard Unification

- $\theta$ is a unifier of a set of term equations \{\(s_1 = t_1, \ldots, s_n = t_n\)\} if \(s_i \theta \equiv t_i \theta\) for \(i \in [1..n]\)
- $\theta$ is an mgu (most general unifier) of \(E\) if
  - $\theta$ is a unifier of \(E\)
  - $\theta$ is more general than all unifiers of \(E\)
- Two sets of equations are equivalent if they have the same set of unifiers
Connection with CSP's

- **Domains:** $\mathcal{T}$, the set of all terms in the considered alphabet

- $s = t$ with variables $x_1, ..., x_n$ represents the constraint
  \[
  \{(x_1\eta, ..., x_n\eta) \mid \eta \text{ unifier of } s \text{ and } t\}
  \]

- $\{s_1 = t_1, ..., s_k = t_k\}$ with variables $x_1, ..., x_n$ represents
  \[
  \langle s_1 = t_1, ..., s_k = t_k ; x_1 \in \mathcal{T}, ..., x_n \in \mathcal{T} \rangle
  \]

**Note:**

\[
\text{Sol}(\langle E ; x_1 \in \mathcal{T}, ..., x_n \in \mathcal{T} \rangle) = \{(x_1\eta, ..., x_n\eta) \mid \eta \text{ unifier of } E\}
\]
Unif Proof System

Decomposition

\[
\frac{f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)}{s_1 = t_1, \ldots, s_n = t_n}
\]

Failure 1

\[
\frac{f(s_1, \ldots, s_n) = g(t_1, \ldots, t_m)}{\bot}
\text{ (where } f \neq g)\]

Deletion

\[
x = x
\]
Unif Proof System, ctd

Transposition

\[
\frac{t = x}{x = t} \quad \text{(where } t \text{ is not a variable)}
\]

Substitution

\[
\frac{x = t, E}{x = t, E \{ x / t \}} \quad \text{(where } x \notin \text{Var}(t) \text{ and } x \in \text{Var}(E))
\]

Failure 2

\[
\frac{x = t}{\bot} \quad \text{(where } x \in \text{Var}(t) \text{ and } x \neq t)\]
Martelli-Montanari Algorithm

Given:
- CSP $\mathcal{P} := \langle C; D\mathcal{E} \rangle$
- Rule
  $\mathcal{R} := \frac{\langle C; D\mathcal{E} \rangle}{\langle C'; D\mathcal{E}' \rangle}$
- $\langle C'; D\mathcal{E}' \rangle$ is the result of applying $\mathcal{R}$ to $\mathcal{P}$
- This rule application of $\mathcal{R}$ is called global

Martelli-Montanari Algorithm
- Unif proof rules
- All applications of the Substitution rule are global
Linear Equations over Reals

Alphabet

- each real number is a constant
- for each real number $r$ unary function symbol ‘$r \cdot$’
- binary function symbol ‘$+$’ (written in infix notion)

Linear expressions and equations

- Linear expression over reals: a term in this alphabet
- Linear equation over reals:
  \[ s = t \]
  where $s$, $t$ linear expressions
Normal Forms

Assume ordering $<$ on the variables

- Linear expression in normal form:
  \[ \sum_{i=1}^{n} a_i x_i + r \]
  where $n \geq 0$ and $x_1, ..., x_n$ are ordered w.r.t. $<$

- Linear equation in normal form:
  \[ \sum_{i=1}^{n} a_i x_i = r \]
  where $n \geq 0$ and $x_1, ..., x_n$ are ordered w.r.t. $<$

- Linear equation in pivot form:
  \[ x = t \]
  if $x \notin \text{Var}(t)$ and $t$ is in normal form

- Each linear equation can be rewritten (normalises) to a unique linear equation in normal form.
Substitutions

- **Substitution**: finite mapping from variables to linear expressions in normal form
  To each variable $x$ in its domain a linear expression different from $x$ is assigned.

- Given: substitutions $\theta$ and $\gamma$
  **Composition** $\theta \gamma$ of $\theta$ and $\gamma$ uniquely determined by
  
  $\eta(x) := \text{norm}((x\theta)\gamma)$

- $\theta$ is a unifier of $s = t$ if $s\theta = t\theta$ normalises to $0 = 0$
Pivot Forms

Three types of normal forms:

- $0 = 0$
- $0 = r$ where $r$ is a non-zero real
- $\sum_{i=1}^{n} a_i x_i = r$, where $n > 0$

Pivot forms of linear equations

- Each linear equation $e$ normalises to a normal form
- Linear equations with normal form $0 = 0$ or $0 = r$ have no pivot form
- Otherwise each equation

$$x_j = \sum_{i \in \{1..j-1|\cup|j+1..n\}} -\frac{a_i}{a_j} x_i + \frac{r}{a_j}$$

is a pivot form of $e$
Lin Proof System

Deletion

\[ s = v \]

if \( s = v \) normalises to \( 0 = 0 \)

Failure

\[ s = v \]
\[ \perp \]

if \( s = v \) normalises to \( 0 = r \) and \( r \) non-zero real
Lin Proof System, ctd

- $\text{norm}(s)$: normal form of $s$
- $\text{stand}(s = t) := \text{norm}(s) = \text{norm}(t)$

Substitution

\[
\frac{s = v, E}{x = t, \text{stand}(E \{ x / t \})}
\]

where $x = t$ is a pivot form of $s = v$
Gauss-Jordan Elimination

- Lin proof rules
- All applications of the Substitution rule are global and condition $x \in \text{Var}(E)$ holds

**Theorem**

Given: finite set of linear equations $E$
- Gauss-Jordan Elimination always terminates
- If $E$ has a solution, then each execution of the algorithm terminates with a set of linear equations that determines an mgu of $E$
  Otherwise each execution terminates with a set containing $\bot$. 
Gaussian Elimination

Forward substitution phase:
Repeatedly take the leftmost equation that has not yet been considered
- Deletion applicable: delete the equation and consider the next equation
- Failure applicable: terminate with failure
- Substitution applicable: apply it taking as $E$ the set of equations lying to the right of the current equation

Backward substitution phase:
Repeatedly take the rightmost equation that has not yet been considered
Apply Substitution taking as $E$ the set of equations to the left of the current equation.
Gaussian Elimination: Correctness

**Theorem**
Given: finite set of linear equations $E$
- Gaussian Elimination always terminates
- If $E$ has a solution, then each execution of the algorithm terminates with a set of linear equations that determines an mgu of $E$.
  Otherwise each execution terminates with a set containing $\bot$. 
Objectives

- Introduce a simple proof theoretic framework
- Use it to define complete solvers
- Show how the standard unification problem can be interpreted as CSP
- Discuss Gauss-Jordan Elimination and Gaussian Elimination algorithms for solving linear equations over reals