Constraint Propagation Algorithms

- Introduction
- General propagation algorithms
- Algorithms for partial orderings
- Algorithms for Cartesian products of partial orderings
- Partial orders $\rightarrow$ CSPs
- Node consistency algorithm
- Arc consistency algorithm

Generic procedure Solve

```
Var continue: BOOLEAN;
continue := TRUE;
While continue And NOT Happy Do
  Preprocess;
  Constraint Propagation;
  If NOT Happy Then
    If Atomic Then
      continue:=FALSE
    Else
      Split;
      Proceed By Cases
    End
  End
End
```

Preliminaries - Set Theory

- Partial order is a pair $(D, \sqsubseteq)$, where D is a set and $\sqsubseteq$ is a reflexive, antisymmetric and transitive relation on D.
- In strict partial order $\sqsubseteq$ is antireflexive.
- Partial order is **well-founded** if no infinite sequence of elements $d_0, d_1, \ldots$ of D exists such that $d_{i+1} \sqsubset d_i$.
- Example: $(\mathcal{P}(D), \supseteq)$ is a well-founded partial order. $\mathcal{P}(D)$ is the powerset of set D and $\supseteq$ the reversed.

Partial Order - Definitions

- Sequence $d_0, d_1, \cdots \in D$ **eventually stabilises at d** if for some $j \geq 0$, $d_i = d \forall i \geq j$.
- Iteration of $\mathcal{F}$ is a sequence $d_0, d_1, \ldots$ from D defined inductively
  $d_0 := \bot$
  $d_j := F_{n_j}(d_{j-1})$
  where $j > 0$ and n_j is an element of $[1 \ldots k]$.
- Lemma on stabilisation
  - Consider a partial ordering $(D, \sqsubseteq)$ with the least element $\bot$ and a finite set F of monotonic functions on D.
  - Suppose than an iteration of F eventually stabilises at a common fixpoint d of the functions from F. Then d is the least common fixed point of the functions from F.

Commutativity

- Lemma on Commutativity
  - Partial ordering $(D, \sqsubseteq)$ with the least element $\perp$. Let $F := \{f_1, \ldots, f_k\}$ be a finite set of functions on D such that
    - each $f \in F$ is monotonic and idempotent
    - all f and $g \in F$ commute
  - then for each permutation $\pi[1 \ldots k] \rightarrow [1 \ldots k] f_{\pi(i)} f_{\pi(2)} \cdots f_{\pi(k)}(\perp)$ is the least common fixpoint of the functions from F.
- Proof: by commutativity $f_{\pi(i)} f_{\pi(2)} \cdots f_{\pi(k)} = f_1 f_2 \cdots f_k(\perp)$
- $f_i f_1 f_2 \cdots f_k(\perp) = f_1 f_i f_2 \cdots f_k(\perp) = \cdots = f_1 f_2 \cdots f_i f_i \cdots f_k(\perp) = f_1 f_2 \cdots f_k(\perp)$.

Semi-Commutativity

- Lemma: Consider a partial ordering $(D, \sqsubseteq)$ with the least element $\bot$. Let $F := f_1, \ldots f_k$ be a finite sequence of functions on D such that
  - each f_i is monotonic, inflationary and idempotent
  - each f_i semi-commutes with f_j for $i > j$ that is $f_i f_j(x) \sqsubseteq f_j f_i(x)$ for all x.
- Then $f_1 f_2 \cdots f_k(\bot)$ is the least common fixpoint of the functions from F.

Least Fixed Point

- Lemma: Any iteration F on a finite partial ordering that is regular eventually stabilises at the least common fixpoint of the functions from F.
- F is regular if for all $f \in F$ and $m \geq 0$ if $f(d_m) \neq d_m$, then f is activated at some step $> m$.

Direct Iteration Algorithm

$d := \perp$;
$G := F$;
While $G \neq \emptyset$ Do
choose $g \in G$;
$d := g(d)$;
$G := G - \{g\}$
End

- Direct Iteration algorithm terminates and computes in d the least common fixpoint of the functions from F.
- F is a finite set of monotonic and idempotent functions on D that commute with each other.
- This is direct consequence of Commutativity Lemma.

Generic Iteration Algorithm

$d := \bot;$

$G := F;$

While $G \neq \emptyset$ Do

choose $g \in G$;

If $d \neq g(d)$ Then

$G := G \bigcup update(G, g, d);$

$d := g(d)$

Else

$G := G - \{g\}$

End

End

- where for all G, g, d: A
  $\{f \in F - G | f(d) = d \bigwedge fg(d) \neq g(d)\} \subseteq update(G, g, d).$

Generic Iteration Algorithm Continued

- Theorem: Every execution of the Generic Iteration Algorithm terminates and computes in d the least common fixpoint of the functions from F. Here F is a finite set of monotonic and inflationary functions on set D with partial ordering $\sqsubseteq$.
- Proof is based on lexicographical ordering of the strict partial orderings $(D, \sqsubseteq)$ and $(\mathcal{N}, <)$, defined on the elements of $D \times \mathcal{N}$ by $(d_1, n_1) <_{lex} (d_2, n_2)$ iff $d_1 \sqsupset d_2$ or ($d_1 = d_2$ and $n_1 < n_2$).

Algorithms for Cartesian Products of Partial Orderings

- Definition: Cartesian product $(D, \sqsubseteq)$ of partial orderings $(D_i, \sqsubseteq_i)$ is a partial order:
  - $D = D_1 \times \cdots \times D_n$
  - $(d_1, \ldots, d_n) \sqsubseteq (e_1, \ldots, e_n)$ iff $d_i \sqsubseteq_i e_i$ for all $i \in [1 \ldots n]$ and $(d_1, \ldots d_n)$ and $(e_1, \ldots, e_n) \in D$
- A scheme s is a sequence of elements from $[1 \ldots n]$.
- $(D_s, \sqsubseteq_s)$ is the projection of D to the elements of the scheme.
- A function f is with scheme s if it depends only on elements that are in s.
- f^+ is extension of f to all elements of D.

Direct Iteration for Compound Domains Algorithm

$d := (\bot_1 \ldots \bot_n);$

$G := F_0;$

While $G \neq \emptyset$ Do

choose $g \in G$

$d[s] := g(d[s])$, where s is the scheme of g

End

Direct Iteration for Compound Domains

- Suppose that $(D, \sqsubseteq)$ is a partial ordering that is a Cartesian product of n partial orderings, each with the least element $\bot_i$ with $i \in [1 \ldots n]$. Let F_0 be a finite set of functions with schemes.
- Suppose that all functions in F_0 are monotonic, idempotent and commute with each other. Then the DICD algorithm terminates and computes in d the least common fixpoint of the functions from $F := \{f^+ | f \in F_0\}$.

From partial orderings to CSPs

- Partial orderings with least elements
  - Cartesian product of the partial orderings $(\mathcal{P}(D_i), \supseteq)$, and $(\mathcal{P}(C_i), \subseteq)$.
  - The domain ordering is used for node, arc, hyper-arc and directional arc consistency. The constraint ordering is used for path, directional path, k-, and relational consistency notions.
- Monotonic and inflationary functions correspond to the domain reduction rules and specific transformation rules used in Chapter 5 to characterise the local consistency notions.
- Common fixpoints correspond to the CSPs that satisfy the considered notion of local consistency.

Node Consistency

- The rule: $\frac{\langle C; x \in D \rangle}{\langle C; x \in C \bigcap D \rangle}$.
- Same rule: $\pi_0(X) := X \bigcap C$. π_0^+ is a function on $\mathcal{P}(D_1) \times \cdots \mathcal{P}(D_n)$.
- Lemma: A CSP $\langle C; x_1 \in D_1, \ldots, x_n \in D_{\rangle} >$ is node consistent iff $(D_1, \ldots, D_n)$ is a common fixpoint of all functions π_0^+ associated with the unary constraints from C.

  All functions π_0 associated with a unary constraint C are

  - monotonic w.r.t. the componentwise ordering $\supseteq$
  - idempotent
  - commute with each other

Node Consistency Algorithm

$S_0 := \{C|C$ is a unary constraint from $C\}$;

$S := S_0$;

While $S \neq \emptyset$ Do

choose $C \in S$; suppose C is on x_i;

$D_i := C \bigcap D_i$; %apply the function π_0 associated with C

$S := S - \{C\}$

End

Arc Consistency

Arc Consistency 1

$$\frac{\langle C; x \in D_x, y \in D_y \rangle}{\langle C; x \in D'_x, y \in D_y \rangle} \tag{1}$$

- This rule can be viewed as a function π_1 on $\mathcal{P}(D_x) \times \mathcal{P}(D_y)$:

$$\pi_1(X, Y) := (X', Y) \tag{2}$$

  where $X' := \{a \in X | \exists b \in Y (a, b) \in C\}$.

- The same applies to rule 2 in which Y is reduced instead of X.

Arc Consistency

- A CSP $\langle C; x_1 \in D_1, \ldots, x_n \in D_n \rangle$ is arc consistent iff $(D_1, \ldots, D_n)$ is a common fixpointt of all functions π_1^+ and π_2^+ associated with the binary constraints from C.
- Each projection function π_i associated with binary constraint C is
  - inflationary w.r.t. the componentwise ordering $\supseteq$.
  - monotonic w.r.t. the componentwise ordering $\supseteq$.

Arc Consistency Algorithm

$S_0 :=$

$\{C|C$ is a binary constraint from $C\} \bigcup \{C^T|C$ is a binary constraint from $\}$;

$S := S_0$;

While $S \neq \emptyset$ Do choose $C \in S$; suppose C is on x_i, x_j;

$D_i := \{a \in D_i\}|\exists b \in D_j(a, b) \in C\}$; % apply π_1 associated with C

If D_i changed Then

$S := S \bigcup \{C' \in S_0|C'$ is on y, z where y is x_i or z is $x_i\}$

Else

$S := S - \{C\}$

End

End

Conclusions

- CSPs can be studied as partial orders and thus generic algorithms that produce least fixed points for partial orders are useful for CSPs
- The least common fixed point of the partial orders corresponds to the maximal domains that satisfy the studied local consistency notion.

Harjoitustehtävät

1. Tarkastellaan rajoiteongelmaa
   $\langle x + y \leq 9, x \cdot y > 5, x < 10, y > 8; x \in [0 \ldots 20], y \in [0 \ldots 20] \rangle$
   (a) Käytä kirjan solmukonsistenssialgoritmia (7.3) ja tee annetusta rajoiteongelmasta solmukonsistentti.
   (b) Käytä kirjan kaarikonsistenssialgoritmia (7.4) ja tee annetusta rajoiteongelmasta kaarikonsistentti.
   Huom. esitä ratkaisussasi algoritmin toiminta vaihe vaiheelta.
2. Kirjan tehtävä 7.4
   Todista solmukonsistenssilemma