About consistencies

Sami Nurminen

Seminar on Theoretical Computer Science, 2009

Introduction

- I will present different local consistencies.
- Local consistencies define properties which the problem should satisfy after applying the constraint propagation.
- Motivation: Enforcing local consistency helps avoiding state explosion problem in the search part.
- How the this is achieved is left open and is up to algorithms(not covered here).

Introduction to Montanari's properties

- Properties of networks that can be made globally consistent in polynomial space.

Definition of decomposability

Definition

- A relation ρ(scheme X) is binary-representable iff there exists a network with scheme X and $sol(N) = \rho$.
- A relation ρ(scheme X) is decomposable in the sense of Montanari iff for all $Y \subseteq X \pi_Y(\rho)$ is binary-representable.
- A network is decomposable in the sense of Montanari iff $sol(N)$ is a decomposable relation.

Example about decomposable property

- Variables are x_1, x_2, x_3, x_4 and domain is $\{1, 2\}$ for x_2, x_3, x_4. Domain is $\{1, 2, 3, 4\}$ for x_1.
- Constraints are $c_{12} = (10, 21, 31, 41)$, $c_{13} = (10, 20, 31, 41)$ and $c_{14} = (10, 21, 30, 41)$.
- Solution $sol(N) = \{(1000), (2101), (3110), (4111)\}$. Tuples are in order x_1, x_2, x_3, x_4.
- Now $\pi_{\{x_2,x_3,x_4\}}(R) = \{(000), (101), (110), (111)\} = \{x_2 = x_3 \vee x_4\}$ cannot be presented by a binary network. So the network is not decomposable.

Introduction to path consistency

- Path consistency remove pair of values $(v_i, v_j) \in D(x_i)xD(x_j)$ from domain.
- They work in binary normalized networks.
- Non-binary constraints will be discarded.

Definition of path consistency

Definition

Given two variables x_i and x_j in X, the pair of values $(v_i, v_j) \in D(x_i)xD(x_j)$ is path consistent iff for any sequence of variables $Y = (x_i = x_{k_1}, x_{k_2}, ..., x_{k_p} = x_j)$ such that for all $q \in [1..p-1], c_{k_q,k_{q+1}} \in C$, there exists a tuple of values $(v_i = v_{k_1}, v_{k_2}, ..., v_{k_p} = v_j) \in \pi_Y(D)$ such that for all $q \in [1..p-1], (v_{k_q}, v_{k_{q+1}}) \in c_{k_q,k_{q+1}}$

Example about path consistency

- Variables are x_1, x_2, x_3 and each has a domain $\{1, 2\}$.
- Constraints are $c_{12} = x_1 \neq x_2$ and $c_{23} = x_2 \neq x_3$
- Pairs $((x_1, 1), (x_3, 2))$ and $((x_1, 2), (x_3, 1))$ cause that network is not path consistent.
- If constraint $x_1 = x_3$ is added to the network, then it is path consistent.

Definition of 2-path consistency

Definition

Given two variables x_i and x_j in X, the pair of values $(v_i, v_j) \in D(x_i)xD(x_j)$ is 2-path consistent iff for any third variable $x_k \in X$ with $c_{ik} \in C$ and $c_{kj} \in C$, there exist a value $v_k \in D(x_k)$ such that $(v_i, v_k) \in c_{ik}$ and $(v_j, v_k) \in c_{kj}$.

Problems with path consistency

- It is possible, that after enforcing path consistency, some constraints cannot be presented as a function anymore.

Example about problem with path consistency

- Variables are x_1, x_2, x_3, x_4 and domain is $\{1, 2, 3, 4\}$ for each variable.
- $c_{12} = |x_1 - x_2| \geq 2$, $c_{23} = x_2 \neq x_3$, $c_{13} = x_1 \neq x_3$
- Path consistency removes $(2, 4)$ and $(3, 1)$ from c_{13}.
- Specific propagation algorithm no longer works.

Introduction to k-Consistencies

- Now we consider instantiations on set of variables.
- Question is: Can we add any new variable to instantiation and it is still locally consistent?

Definition of k-consistency

Definition

Given a set of variables $Y \subseteq X$ with $|Y| = k - 1$ a locally consistent instantiation I on Y is k-consistent iff for any kth variable $x_{i_k} \in X \setminus Y$ there exists a value $v_{i_k} \in D(x_{i_k})$ such that $I \cup \{(x_{i_k}, v_{i_k})\}$ is locally consistent.

Definition of k-consistency(cont.)

Definition

The network N is k-consistent iff for any set Y of $k-1$, any locally consistent instantiation on Y is$k$-consistent.

Example about k-consistency

- Variables x_1, x_2, x_3, x_4 and domain for each is $\{1, 2\}$.
- Contraint is $c(x_1, x_2, x_3) = \{(1, 1, 1), (2, 2, 2)\}$
- Instantiation $(x_1 = 1, x_2 = 2)$ is locally consistent, but it cannot be extended consistently to x_3.
- Network is not 3-consistent.

Strong k-consistency

Definition

A network is strongly k-consistent iff it is j-consistent for all $j \leq k$.

Introduction to pairwise consistency

- This consistency is based on constraints.
- Take two constraints and check if both can be extended to an instantiation on $X(c_1) \cup X(c_2)$ and both are satisfied.

Definition of pairwise consistency

Definition

- Consider an embedded network.
- Two constraints c_1 and c_2 are pairwise consistent iff any tuple on $X(c_1)$ satisfying c_1 can be extended to an instantiation on $X(c_1) \cup X(c_2)$ satisfying c_2 and same kind of extension can be done to c_2, so, iff $\pi_{X(c_1) \cap X(x_2)}(c_1) = \pi_{X(c_1) \cap X(x_2)}(c_2)$

Definition of pairwise consistency (cont.)

- Network is pairwise consistent iff all pairs of its constraints are pairwise consistent.

Example about pairwise consistency

- Variables are x_1, x_2, x_3, x_4.
- Domain for each variable is $\{1, 2\}$.
- Constraints are $c_1(x_1, x_2, x_3) = \{(121), (211), (222)\}$ and $c_2(x_2, x_3, x_4) = \{(111), (222)\}$
- Tuple $(121) \in c_1$ cannot be extend to any tuple in c_2 so network is not pairwise consistent.

Introduction to k-wise consistency

- Kind of extension compared to pairwise consistency.
- Proposed in database contex.

Definition of k-wise consistency

Definition

- Consider an embedded network
- A set of contraints $\{c_1, ..., c_k\}$ is k-wise consistent if any tuple on $X(c_i)$, $i \in [1..k]$, satisfying c_i can be extended to an instantiation on $\bigcup_{j=1}^{k} X(c_j)$ and instantiation satisfies c_j, $j \in [1..k]$

Introduction to restricted path consistency

- This consistency doesn't change constraints.
- Remove more inconsistent values with less cost that path consistency.
- Idea is that we extend to only those pairs of values that if removed, then value would become inconsistent. This is "optimization" compared to path consistency.

Definition of restricted path consistency

Definition

- Network in definition is binary and normalized.
- Network is RPC iff it is arc consistent and for all $x_i \in X$, for all $v_i \in D(x_i)$ for all $c_{ij} \in C$ such that (x_i, v_i) has a unique support $((x_i, v_i), (x_j, v_j))$ on c_{ij}, for all $x_k \in X$ linked to both x_i and x_j by a constraint, there exists $v_k \in D(x_k)$ such that $(v_i, v_k) \in c_{ik}$ and $(v_j, v_k) \in c_{jk}$.

Introduction to path inverse consistency

- Leaves set of constraints untouched. Removes only values.

Definition of path inverse consistency

Definition

- Network is binary and normalized.
- Network is PIC iff for all $x_i \in X$, for all $v_i \in D(x_i)$, for all $x_j, x_k \in X$ there exists $v_j \in D(x_j)$ and $v_k \in D(x_k)$ such that $((x_i, v_i), (x_j, v_j), (x_k, v_k))$ is locally consistent.

RPC is weaker that PIC.

Introduction to neighborhood inverse consistency

- Ensures that value $v_i \in x_i$ can be extended to all its neighbors.
- Neighborhood defined as variables involved in constraint with x_i.

Definition of neighborhood inverse consistency

Definition

A network is NIC iff for all $x_i \in X$, for all $v_i \in D(x_i)$ the instantiation (x_i, v_i) can be extended to a locally consistent instantiation on the set of all variables involved in a constraint with x_i.

Introduction

- It is a general technique.
- Try different assignments to a variable and then perform constraint propagation.

Definition

Definition

A network $N = (X, D, C)$ is singleton arc consistent iff for all $x_i \in X$, for all $v_i \in D(x_i)$, subproblem $N|_{x_i=v_i}$ is arc consistent.

- Arc consistency is the local consistency used in the definition.
- You can translate definition to any other local consistency.

Time Complexity

If it takes polynomial time to make network locally consistent using local consistency Φ, then enforcing singleton consistency based on Φ, has a polynomial worst-case time complexity.

Comparison

From strongest to weakest:

- SAC
- PIC
- RPC
- AC

Pairwise and k-wise consistency not included in comparison because they try to make constraints consistent.