Seminar: Search and Optimization

Directional Consistency

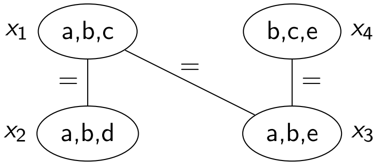
Gabi Röger

Universität Basel

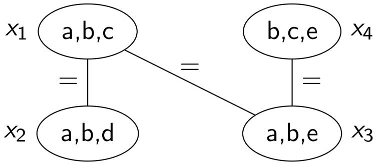
November 6, 2014

Directional Arc-consistency

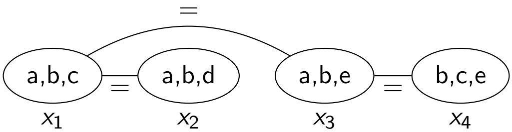
Example



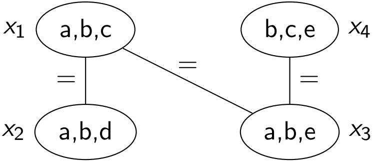
Example



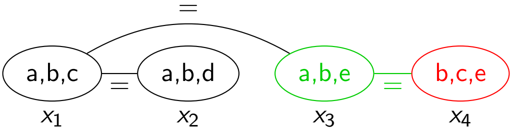
Assume we search with variable order x_1, x_2, x_3, x_4



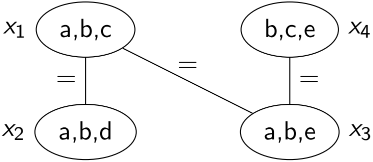
Example



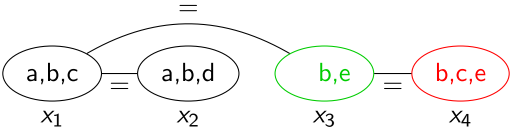
Assume we search with variable order x_1, x_2, x_3, x_4



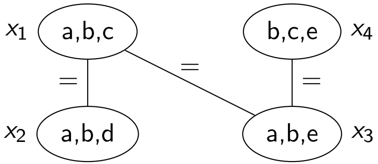
Example



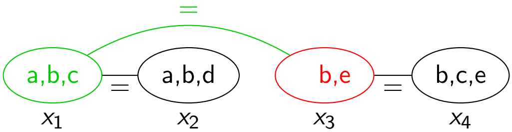
Assume we search with variable order x_1, x_2, x_3, x_4



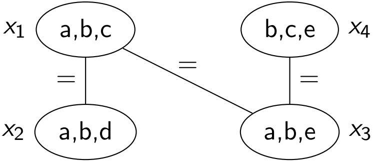
Example



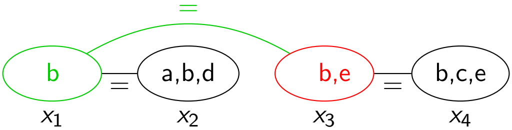
Assume we search with variable order x_1, x_2, x_3, x_4



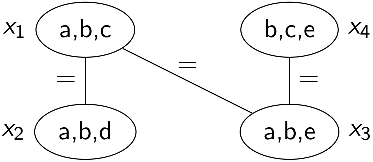
Example



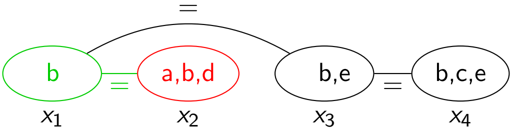
Assume we search with variable order x_1, x_2, x_3, x_4



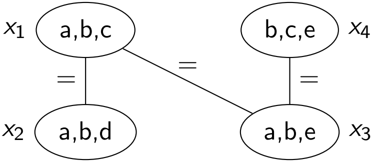
Example



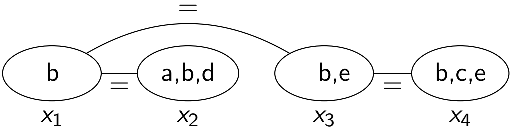
Assume we search with variable order x_1, x_2, x_3, x_4



Example

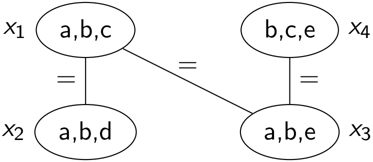


Assume we search with variable order x_1, x_2, x_3, x_4

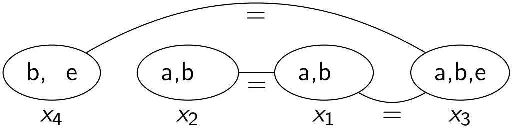


Backtrack-free search

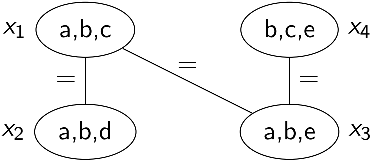
Example



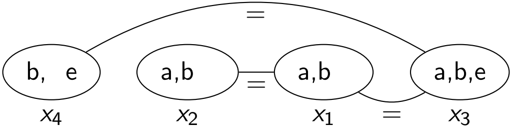
Assume we search with variable order x_4, x_2, x_1, x_3



Example



Assume we search with variable order x_4, x_2, x_1, x_3



Not necessarily backtrack-free search

Directional arc-consistency: Definition

Definition (Directional arc-consistency)

A network is directional arc-consistent relative to variable order $d = (x_1, \ldots, x_n)$ iff every variable x_i is arc-consistent relative to every variable x_j such that $i \leq j$.

Directional Arc-consistency: Function DAC

function DAC:

for $i = n, \ldots, 1$:
  for each $j < i$ such that $R_{ji} \in \mathcal{R}$:
    $D_j \leftarrow D_j \cap \pi_j(R_{ji} \bowtie D_i)$
    (remove values from D_j that don't have a partner in D_i)

Input: Constraint network $\mathcal{R} = (X, D, C)$
with variable ordering $d = (x_1, \ldots, x_n)$

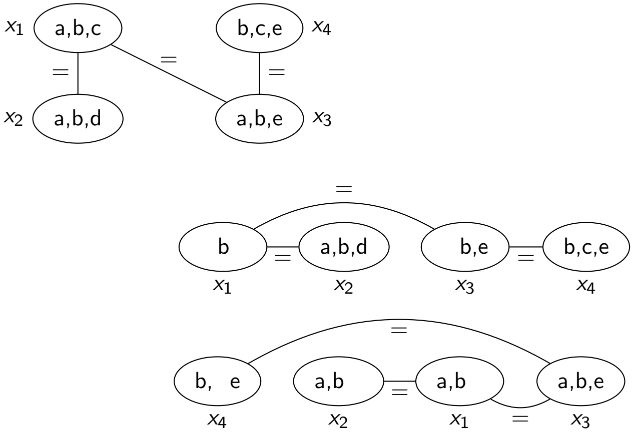
Effect: Enforces directional arc-consistency along d.

Time complexity: $O(ek^2)$ with e binary constraints and maximal domain size k.

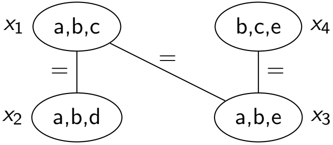
Directional Arc-consistency: Questions

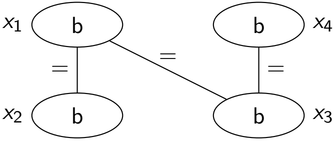
- How does directional arc-consistency relate to full arc-consistency?
- Is there a criterion when directional arc-consistency leads to backtrack-free search?
- Can we find a suitable variable ordering?

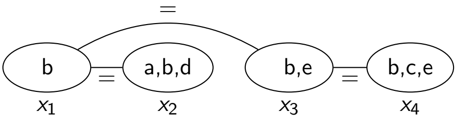
Directional AC vs. Full AC

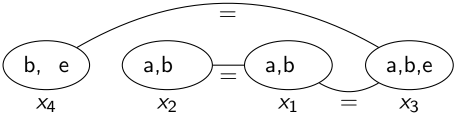


Directional AC vs. Full AC

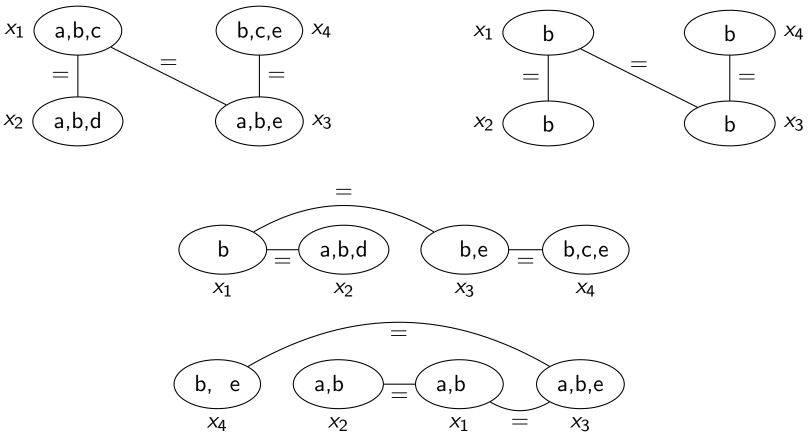








Directional AC vs. Full AC



Enforcing full AC eliminates everything directional AC does

. . . and more

Width of a Graph

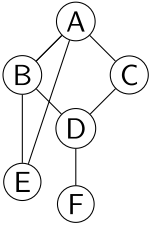
Definition (Width of a graph)

Let $G = (V, E)$ be an undirected graph and $d = (v_1, \ldots, v_n)$ be an ordering of its the nodes.

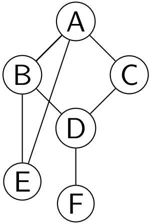
- The parents of a node v are its neighbours that precede it in the ordering.
- The width of a node is the number of its parents.
- The width of the ordering is the maximum width over all nodes.

The width of graph G is the minimum width over all orderings.

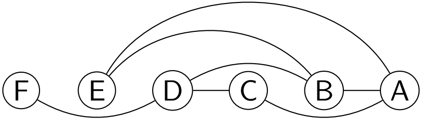
Width of a graph: Example



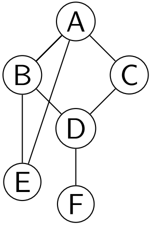
Width of a graph: Example



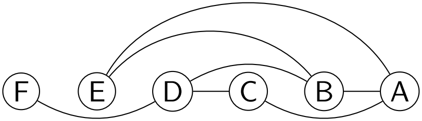
F,E,D,C,B,A:



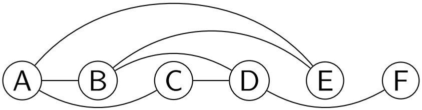
Width of a graph: Example



F,E,D,C,B,A:



A,B,C,D,E,F:



Width of Graph: Algorithm

function MIN-WIDTH:

$d \leftarrow$ array of size $|V|$
for $i = n, \ldots, 1$:
    $r \leftarrow$ a node in G with smallest degree
    $d[i] \leftarrow r$
    Remove all adjacent edges of r from E
    Remove r from V

Input: Graph $G = (V, E)$

Effect: d contains minimum width ordering of nodes.

Width of Graph and Directional Arc-Consistency

Theorem

A graph is a tree iff it has width 1.

Definition

A constraint network is backtrack-free relative to a given ordering $(x_1, \ldots, x_n)$ if for every $i < n$, every partial solutions of $(x_1, \ldots, x_i)$ can be consistently extended to include x_{i+1}

Theorem

Let d be a width-1 ordering of a constraint tree T. If T is directional arc-consistent relative to d then the network is backtrack-free along d.

Application: Algorithm for Trees

function TREE-SOLVING:

Generate width-1 ordering $(x_1, \dots, x_n)$ for $\mathcal{R}$ along a rooted tree.
Let $x_{p(i)}$ denote the parent of x_i in the rooted tree.
for $i = n, \dots, 1$:
  $D_{p(i)} \leftarrow D_{p(i)} \cap \pi_{p(i)}(R_{p(i)i} \bowtie D_i)$
  if $D_{p(i)} = \emptyset$:
    exit (inconsistent network)
Extract solution with (backtrack-free) search.

Input: Constraint network $\mathcal{R} = (X, D, C)$

Output: Solution (or inconsistent network).

Time complexity: $O(nk^2)$ with n variables and maximal domain size k.

Directional Path-consistency

(Strong) directional path-consistency: Function DPC

function DPC:

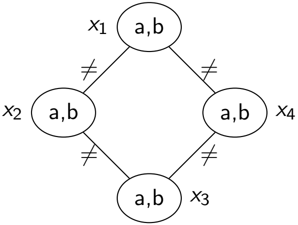
$E' \leftarrow E$
for $k = n, \ldots, 1$:
  for each $i < k$ such that $(x_i, x_k) \in E'$:
    $D_i \leftarrow D_i \cap \pi_i(R_{ik} \bowtie D_k)$
  for each $i, j < k$ such that $(x_i, x_k), (x_j, x_k) \in E'$:
    $R_{ij} \leftarrow R_{ij} \cap \pi_{ij}(R_{ik} \bowtie D_k \bowtie R_{kj})$
    $E' \leftarrow E' \cup (x_i, x_j)$

Input: Constraint network $\mathcal{R} = (X, D, C)$ with constraint graph $G = (V, E)$ and variable ordering $d = (x_1, \ldots, x_n)$

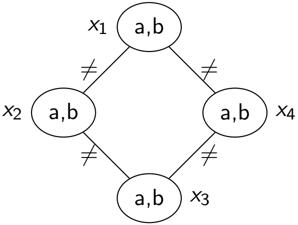
Effect: Enforces directional arc- and path-consistency along d.

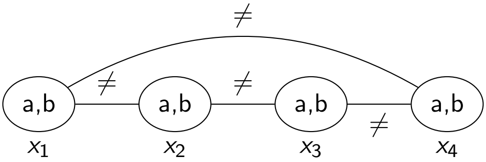
Time complexity: $O(n^3 k^3)$ with n variables and maximal domain size k.

Directional Path-consistency: Example

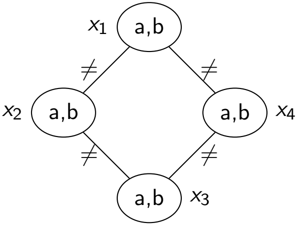


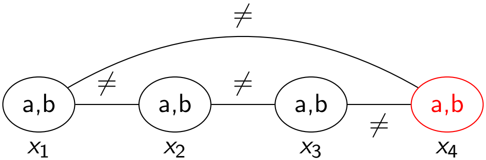
Directional Path-consistency: Example



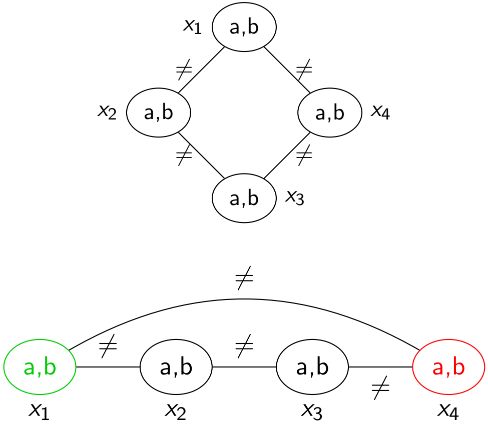


Directional Path-consistency: Example

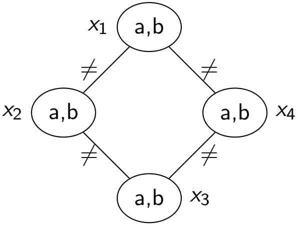


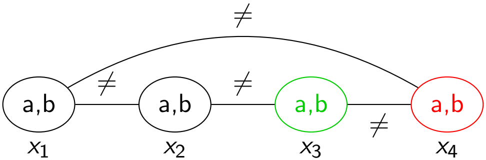


Directional Path-consistency: Example

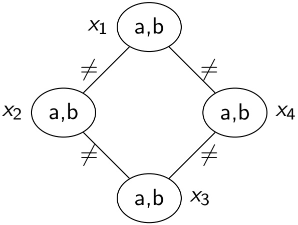


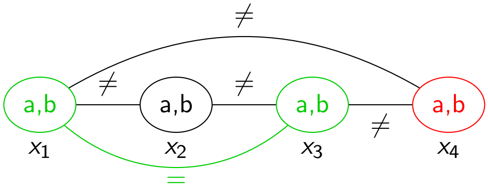
Directional Path-consistency: Example



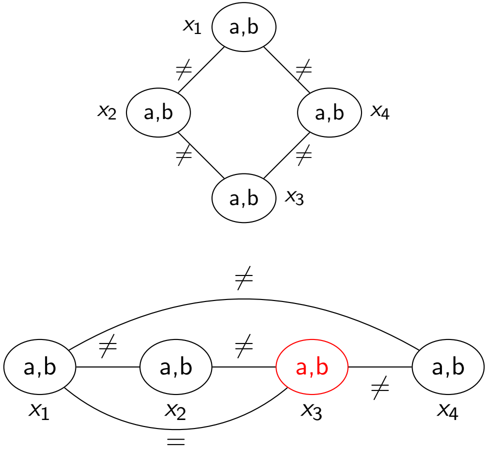


Directional Path-consistency: Example

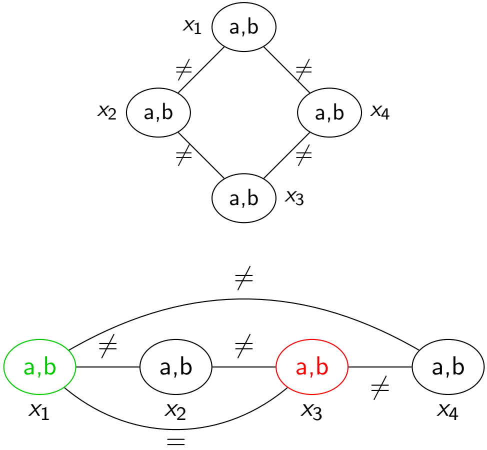




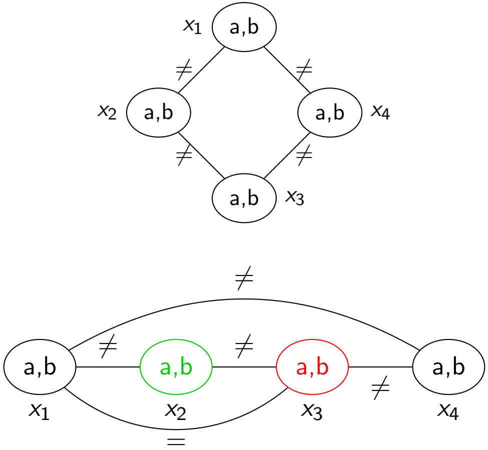
Directional Path-consistency: Example



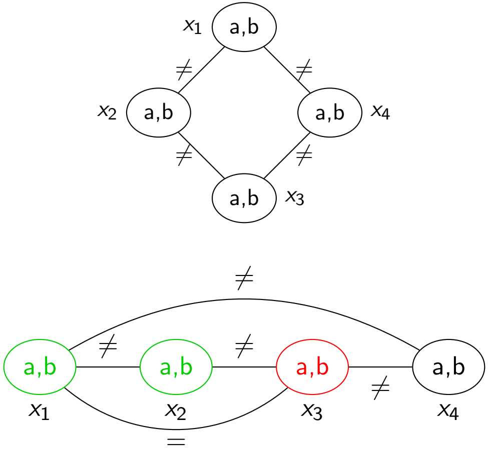
Directional Path-consistency: Example



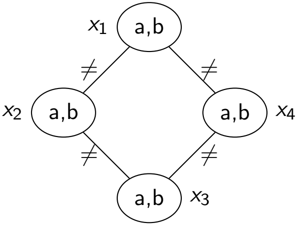
Directional Path-consistency: Example

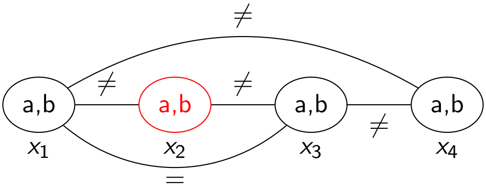


Directional Path-consistency: Example

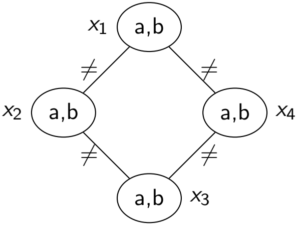


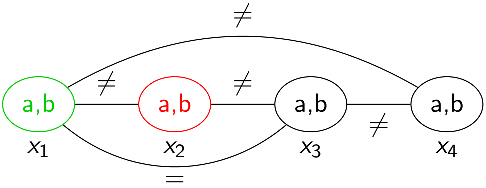
Directional Path-consistency: Example



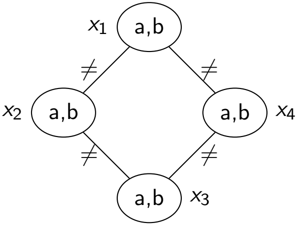


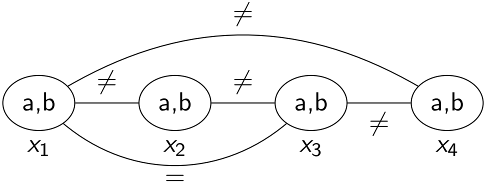
Directional Path-consistency: Example





Directional Path-consistency: Example





Directional Path-consistency: Questions

- Is there a criterion when (strong) directional path-consistency leads to backtrack-free search?
- Can we find a suitable variable ordering?

Directional Path-consistency: Questions

- Is there a criterion when (strong) directional path-consistency leads to backtrack-free search?
- Can we find a suitable variable ordering?

- Directional path-consistency can change the constraint graph.
- Width of contraint graph no longer sufficient.

Directional Path-consistency: Questions

- Is there a criterion when (strong) directional path-consistency leads to backtrack-free search?
- Can we find a suitable variable ordering?

- Directional path-consistency can change the constraint graph.
- Width of contraint graph no longer sufficient.
- Use induced width instead.

Induced Width of a Graph

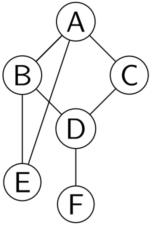
Definition (Induced width of a graph)

Let $G = (V, E)$ be an undirected graph and $d = (v_1, \ldots, v_n)$ be an ordering of its the nodes.

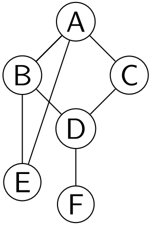
- Obtain graph G_d^* by processing the node ordering backwards and adding edges for each to parents of the processed node.
- The induced width w_d^* of the ordering is the width of G_d^*.

The induced width w^* of graph G is the minimal induced width over all orderings.

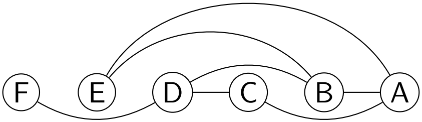
Induced Width of a Graph: Example



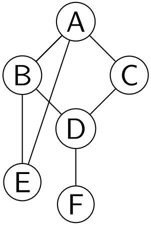
Induced Width of a Graph: Example



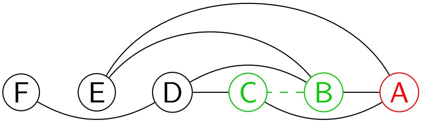
F,E,D,C,B,A:



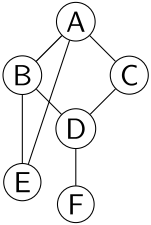
Induced Width of a Graph: Example



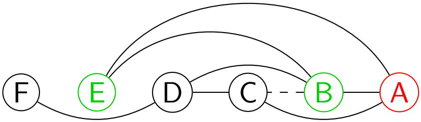
F,E,D,C,B,A:



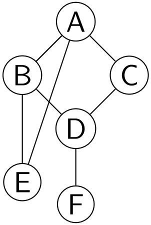
Induced Width of a Graph: Example



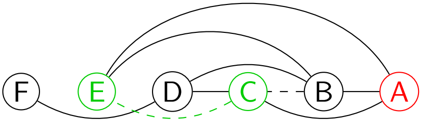
F,E,D,C,B,A:



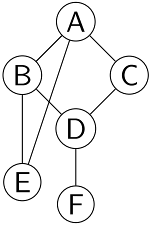
Induced Width of a Graph: Example



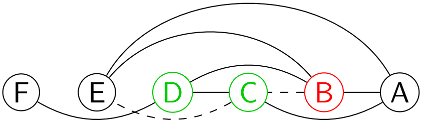
F,E,D,C,B,A:



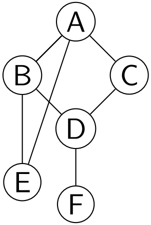
Induced Width of a Graph: Example



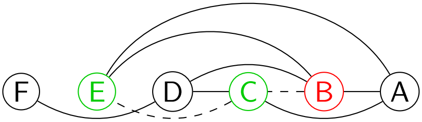
F,E,D,C,B,A:



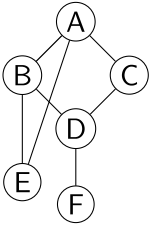
Induced Width of a Graph: Example



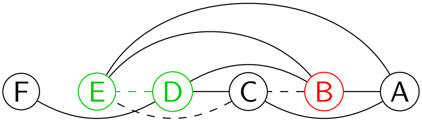
F,E,D,C,B,A:



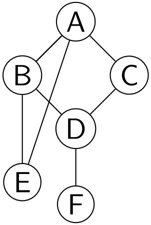
Induced Width of a Graph: Example



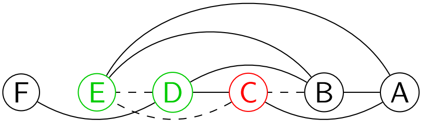
F,E,D,C,B,A:



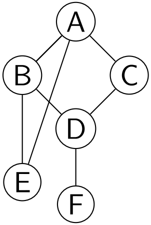
Induced Width of a Graph: Example



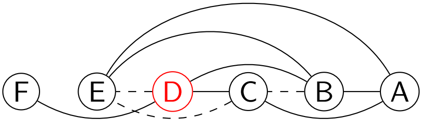
F,E,D,C,B,A:



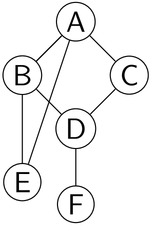
Induced Width of a Graph: Example



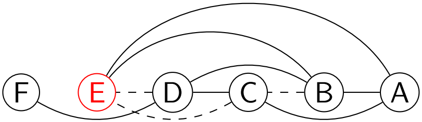
F,E,D,C,B,A:



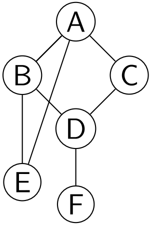
Induced Width of a Graph: Example



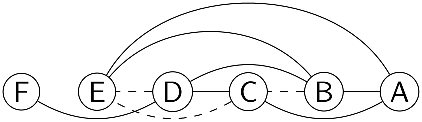
F,E,D,C,B,A:



Induced Width of a Graph: Example



F,E,D,C,B,A:



Induced width $w^*_{(F,E,D,C,B,A)}$: 3

Induced Width of Graph: Algorithm 1

- Determining the induced width of a graph is NP-hard
- Find good ordering in polynomial time

Induced Width of Graph: Algorithm 1

- Determining the induced width of a graph is NP-hard
- Find good ordering in polynomial time

function MIN-DEGREE:

$d \leftarrow$ array of size $|V|$
for $i = n, \ldots, 1$:
    $r \leftarrow$ a node in G with smallest degree
    $d[i] \leftarrow r$
    Connect r's parents: $E \leftarrow E \cup \{(v, v') \mid (v, r), (v', r) \in E\}$
    Remove all adjacent edges of r from E
    Remove r from V

Input: Graph $G = (V, E)$

Effect: d contains ordering with small induced width.

Induced Width of Graph: Algorithm 2

function MIN-FILL:

$d \leftarrow$ array of size $|V|$
for $i = n, \ldots, 1$:
  $r \leftarrow$ a node in G with fewest missing edges between parents
  $d[i] \leftarrow r$
  Connect r's parents: $E \leftarrow E \cup \{(v, v') \mid (v, r), (v', r) \in E\}$
  Remove all adjacent edges of r from E
  Remove r from V

Input: Graph $G = (V, E)$

Effect: d contains ordering with small induced width.

Width of Graph and Directional Arc-Consistency

Theorem

Let G be the constraint graph of a binary network $\mathcal{R}$ and let d be a variable ordering. If DPC is applied to $\mathcal{R}$ with ordering d then the resulting constraint graph is subsumed by the Graph G_d^*.

Width of Graph and Directional Arc-Consistency

Theorem

Let G be the constraint graph of a binary network $\mathcal{R}$ and let d be a variable ordering. If DPC is applied to $\mathcal{R}$ with ordering d then the resulting constraint graph is subsumed by the Graph G_d^*.

Theorem

Given a binary network $\mathcal{R}$ and an ordering d, the time complexity of DPC along d is $O((w_d^*)^2 \cdot n \cdot k^3)$.

Width of Graph and Directional Arc-Consistency

Theorem

Let G be the constraint graph of a binary network $\mathcal{R}$ and let d be a variable ordering. If DPC is applied to $\mathcal{R}$ with ordering d then the resulting constraint graph is subsumed by the Graph G^*_d.

Theorem

Given a binary network $\mathcal{R}$ and an ordering d, the time complexity of DPC along d is $O((w^*_d)^2 \cdot n \cdot k^3)$.

Previously: $O(n^3 k^3)$
Lesson learned: Prefer orderings with small induced width

Adaptive Consistency

Motivation

- Concept of directional arc- and path-consistency can be generalized to directional i-consistency.

Motivation

- Concept of directional arc- and path-consistency can be generalized to directional i-consistency.
- If a network $\mathcal{R}$ has induced width $i - 1$ for ordering d and it is strong directional i-consistent for d then $\mathcal{R}$ is backtrack-free along d.

Motivation

- Concept of directional arc- and path-consistency can be generalized to directional i-consistency.
- If a network $\mathcal{R}$ has induced width $i-1$ for ordering d and it is strong directional i-consistent for d then $\mathcal{R}$ is backtrack-free along d.
- Algorithm idea for CSP solving:

Motivation

- Concept of directional arc- and path-consistency can be generalized to directional i-consistency.
- If a network $\mathcal{R}$ has induced width $i - 1$ for ordering d and it is strong directional i-consistent for d then $\mathcal{R}$ is backtrack-free along d.
- Algorithm idea for CSP solving:
    1. Select ordering d with small width.

Motivation

- Concept of directional arc- and path-consistency can be generalized to directional i-consistency.
- If a network $\mathcal{R}$ has induced width $i-1$ for ordering d and it is strong directional i-consistent for d then $\mathcal{R}$ is backtrack-free along d.
- Algorithm idea for CSP solving:
  1. Select ordering d with small width.
  2. Compute its induced width w_d^*.

Motivation

- Concept of directional arc- and path-consistency can be generalized to directional i-consistency.
- If a network $\mathcal{R}$ has induced width $i-1$ for ordering d and it is strong directional i-consistent for d then $\mathcal{R}$ is backtrack-free along d.
- Algorithm idea for CSP solving:
  1. Select ordering d with small width.
  2. Compute its induced width w_d^*.
  3. Apply strong directional $w_d^* + 1$-consistency.

Motivation

- Concept of directional arc- and path-consistency can be generalized to directional i-consistency.
- If a network $\mathcal{R}$ has induced width $i - 1$ for ordering d and it is strong directional i-consistent for d then $\mathcal{R}$ is backtrack-free along d.
- Algorithm idea for CSP solving:
  1. Select ordering d with small width.
  2. Compute its induced width w_d^*.
  3. Apply strong directional $w_d^* + 1$-consistency.
  4. Determine solution with backtrack-free search.

Motivation

- Concept of directional arc- and path-consistency can be generalized to directional i-consistency.
- If a network $\mathcal{R}$ has induced width $i-1$ for ordering d and it is strong directional i-consistent for d then $\mathcal{R}$ is backtrack-free along d.
- Algorithm idea for CSP solving:
  1. Select ordering d with small width.
  2. Compute its induced width w_d^*.
  3. Apply strong directional $w_d^* + 1$-consistency.
  4. Determine solution with backtrack-free search.
- Idea: Combine steps 2 and 3

Adaptive Consistency: Function ADC

function ADC:

$E' \leftarrow E, C' \leftarrow C$

for $k = n, \dots, 1$:

$S \leftarrow$ parents of x_k w.r.t. E' and d

$R_S \leftarrow \textsc{revise}(S, x_k)$

$C' \leftarrow C' \cup R_S$

$E' \leftarrow E' \cup \{(x_i, x_j) \mid x_i, x_j \in S, x_i \neq x_j\}$

Input: Constraint network $\mathcal{R} = (X, D, C)$ with constraint graph $G = (V, E)$ and variable ordering $d = (x_1, \dots, x_n)$

Effect: Enforces strong directional $w_d^* + 1$-consistency and the resulting network has width bounded by w_d^*.
$\mathcal{R}$ consistent $\Rightarrow$ resulting network backtrack-free along d.

Time complexity: $O(n \cdot (2k)^{w_d^*+1})$

Tractable Class of Constraint Satisfaction Problems

If the induced width for a problem is bounded by a constant b, we can efficiently find an ordering d with $w_d^* \leq b$.

Tractable Class of Constraint Satisfaction Problems

If the induced width for a problem is bounded by a constant b, we can efficiently find an ordering d with $w_d^* \leq b$.

Theorem

The class of constraint problems whose induced width is bounded by a constant b is solvable in polynomial time and space.

Summary

Summary

- Directional arc- and path-consistency can be used as preprocessing algorithm or for interleaved reasoning during search.
- Guarantee backtrack-free search for problems with induced width 1 (for directional arc-consistency) and 2 (for strong directional path-consistency), respectively.
- Identified a tractable class of constraint satisfaction problems
- Purely structural criterion: induced width of constraint graph