Title: On the Computation of Local Interchangeability in Finite CSPs

Authors: B. Y. Choueiry & G. Noubir

Proc.: AAAI'98 (TR KSL-98-24)

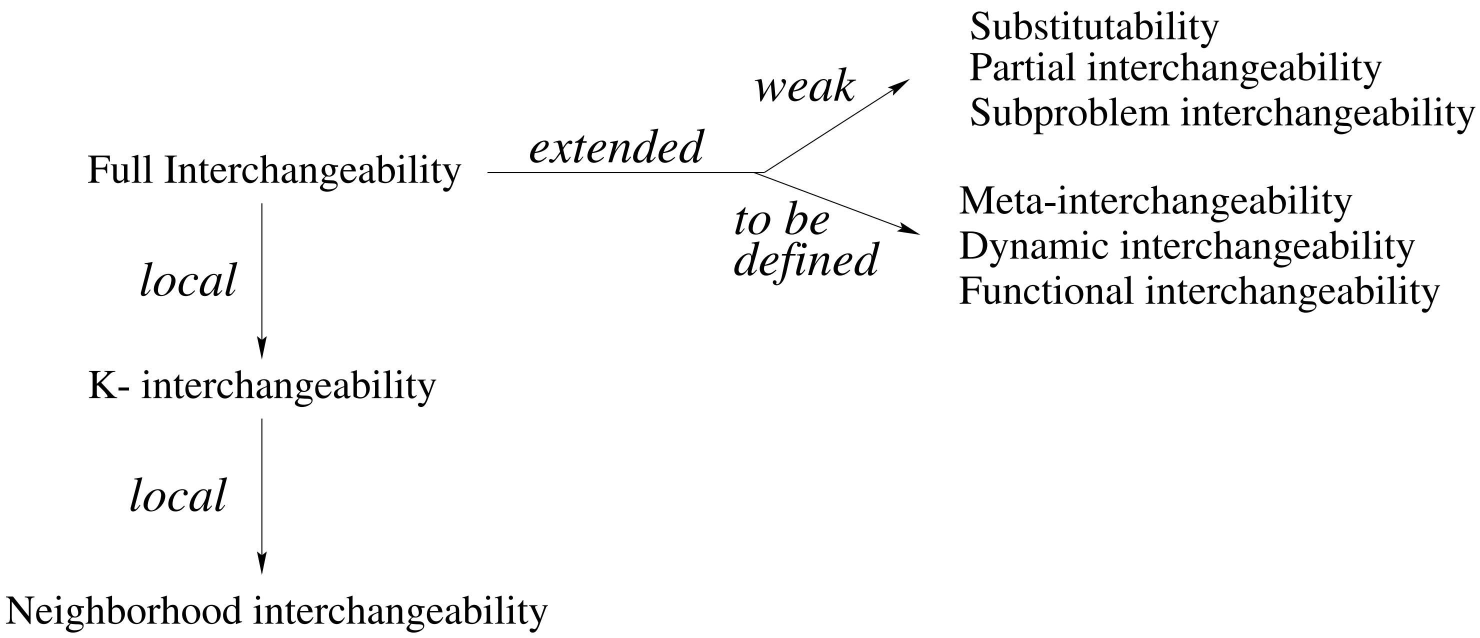
Year: 1998

Theory and Practice of Constraint Satisfaction

CSCE 990-05, Fall 1999

www.cse.unl.edu/~choueiry/CSE990-05/

Types of interchangeability: reminder [Freuder'91]



- Complexity?
- Relationship: semantics, implication?
- Advantages: cost, size of sets

Topic

On the Computation of Local Interchangeability
in Finite Binary CSPs

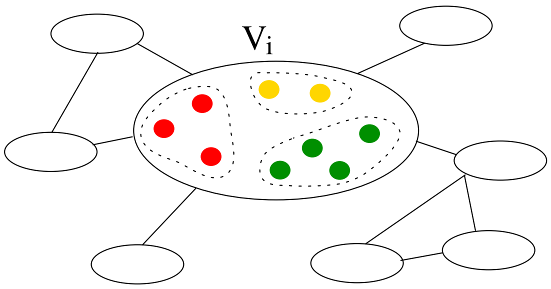
Justification

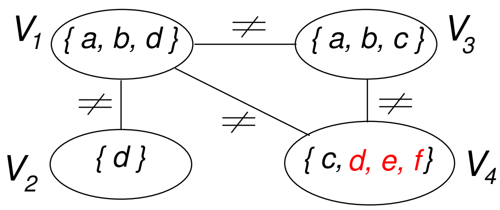
Locality ensures tractability
Weak symmetries create more opportunities

Outline

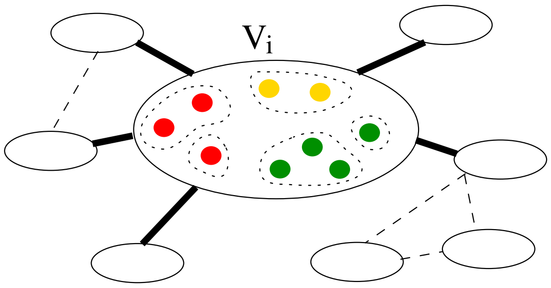
- Computing local interchangeability
  - $\rightarrow$ basis for 2 reformulation procedures
  - $\rightarrow$ use in practice
- Mutual exclusion
- Related work & future research

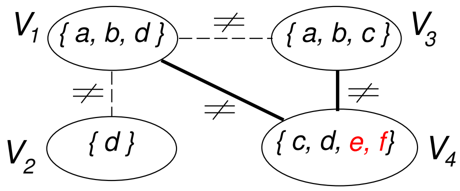
Full (FI)





Neighborhood (NI), restricted to $\texttt{Neigh}(\{V_i\})$





FI/NI values $\rightarrow$ 1 abstract value

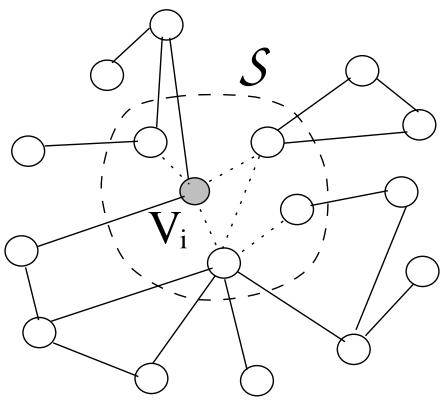
Substitutability (SUB)

d, e, f SUB c for $V_4 \Rightarrow$

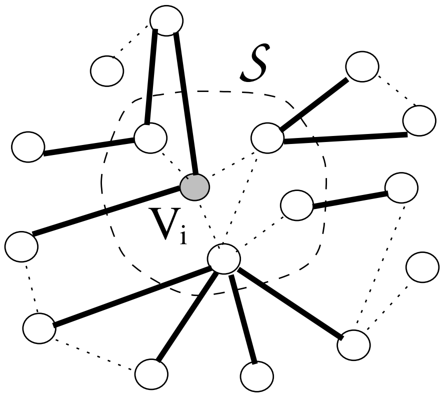
d, e, f can replace c in all solutions with $V_4 = c$

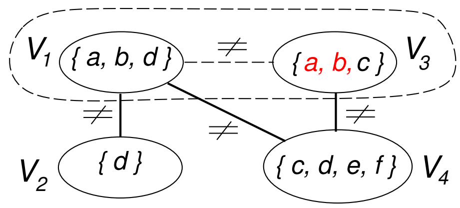
$\rightarrow$ convert solutions to reinforce preferences

Partial (PI) [Freuder'91]



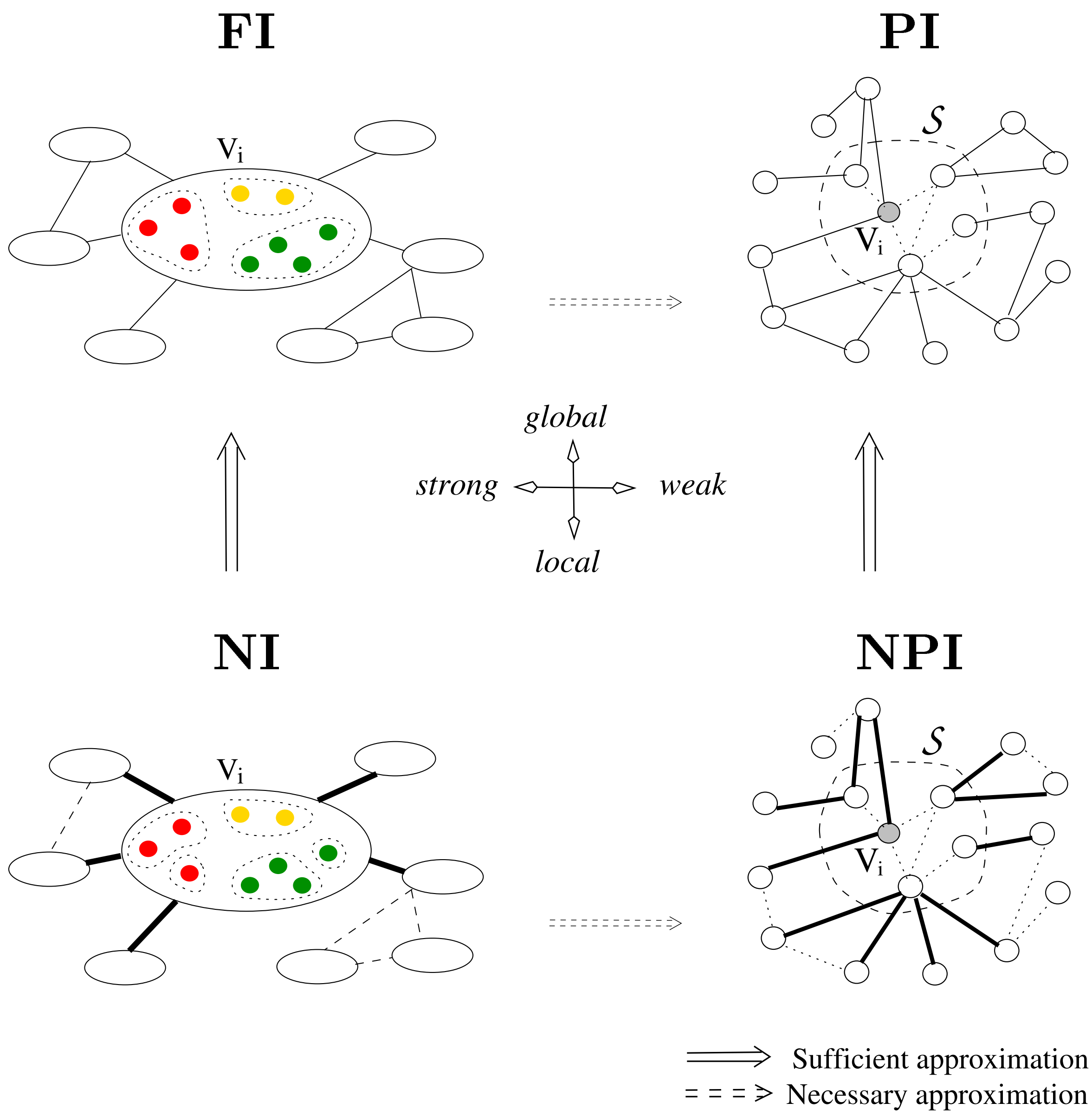
Neighborhood partial(NPI)





Neighborhood interchangeability	$\longrightarrow$	NI
Neighborhood partial interchang.	$\longrightarrow$	NPI
Substitutability	$\longrightarrow$	SUB
Neighborhood of $\mathcal{S}$	$\longrightarrow$	$\texttt{Neigh}(\mathcal{S})$

Control parameters $\left\{ \begin{array}{l} \text{choice of } \mathcal{S} \\ \text{neighborhood of } \mathcal{S} \end{array} \right.$



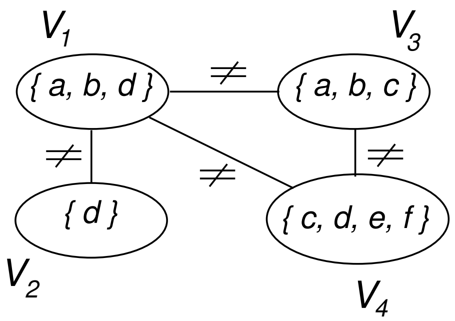
FI, PI (likely) intractable & PI is cheaper

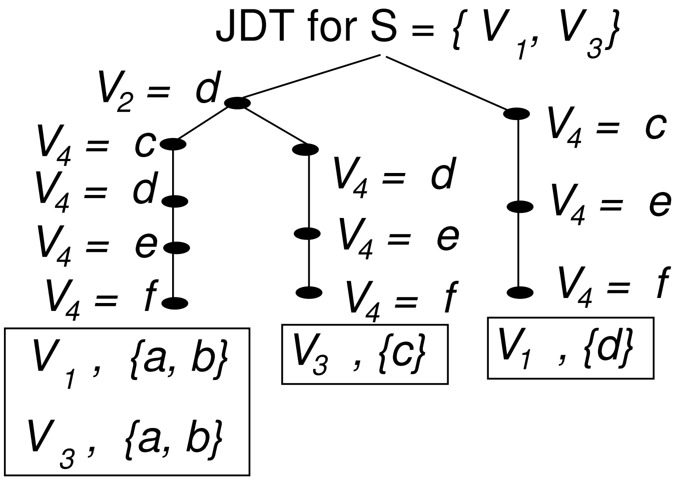
NI, NPI polynomial & NPI is cheaper

Joint discrimination tree $\begin{cases} \mathcal{S} \\ \texttt{Neigh}(\mathcal{S}) \\ \rightarrow \text{tree} \end{cases}$

$\forall\ V_i \in \mathcal{S}$,

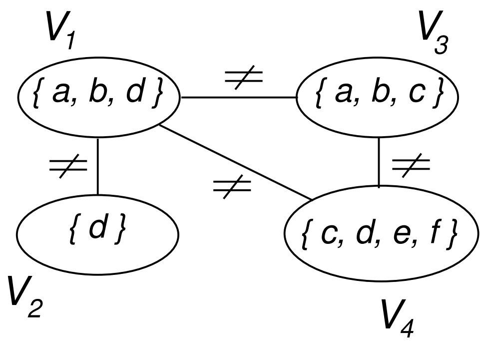
- $\forall\ v_i \in D_{V_i}$
  - $\forall\ V_j \in \texttt{N}eigh(\mathcal{S})$
    - $\forall\ v_j \in D_{V_j}$
    - v_i, v_j consistent? ...
  - Annotate node with(V_i, v_i)
- Return to root

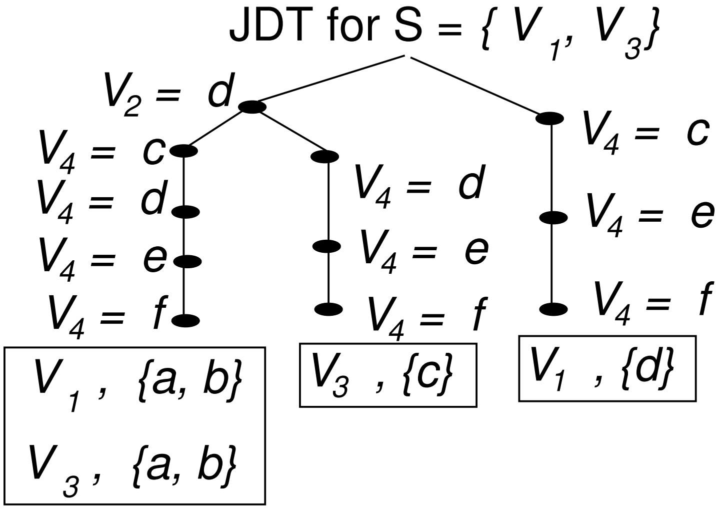




Complexity $O(s(n-s)d^2)$ $\begin{cases} s = |\mathcal{S}| \\ n = |\mathcal{V}| \\ d = \text{max domain size} \end{cases}$

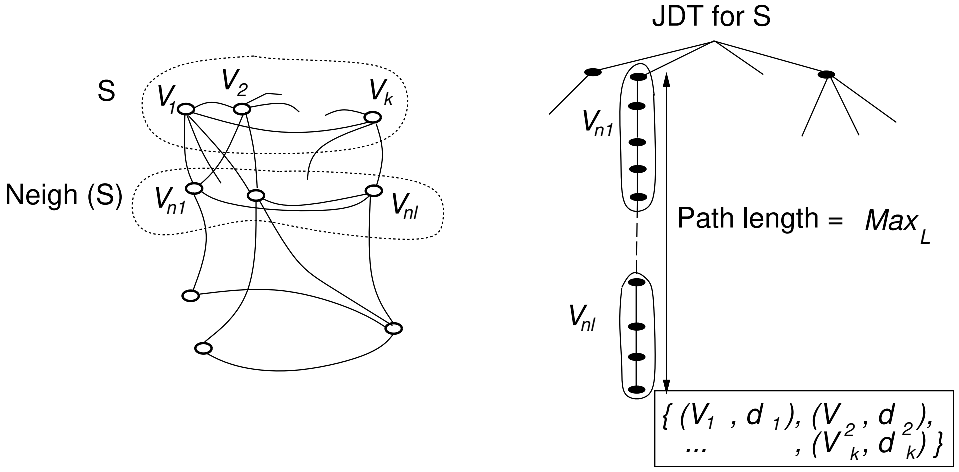
Joint discrimination tree (cont')





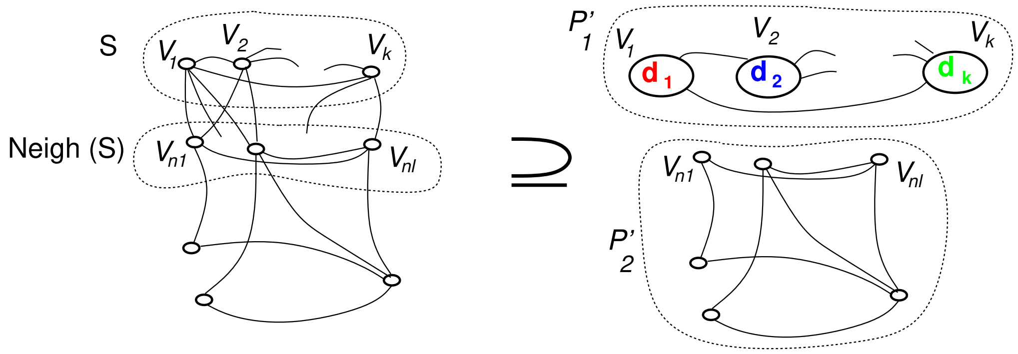
- Annotations partition domains of variables in $\mathcal{S}$
- Non-singleton sets in partition $\rightarrow$ NPI
- NPI sets for all variables in $\mathcal{S}$
- For one variable: $O(s(n-s)d^2) \rightarrow O((n-s)d^2)$

Maximal-length path $Max_L = \sum_{i=n_1}^{i=n_l} |D_{V_i}|$



$\rightarrow$ Path is unique

Decomposition



Problem size $\searrow$, decomposition $\Rightarrow$ Cost $\searrow\searrow$

#Solutions $\searrow$ $\Rightarrow$ sufficient approximation

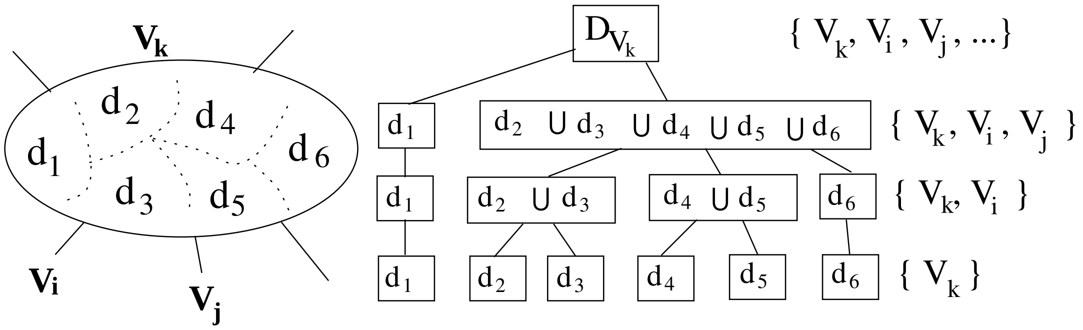
Isolated subproblem

$\surd$ IS $\rightarrow$ necessary and sufficient approximation

$\times$ IS $\rightarrow$ no-good set (FOF)

Hierarchy of domain partitions

JDT partitions the domains of variables in $\mathcal{S}$

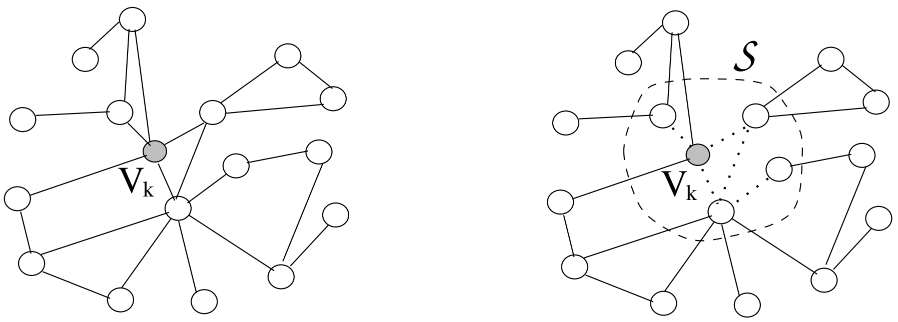


$\mathcal{S}$ controls coarseness of a partition:

$$\text{Refining } \mathcal{S} \begin{cases} \#\text{ sets in a partition} & \nearrow \\ \text{size of a set in a partition} & \searrow \end{cases}$$

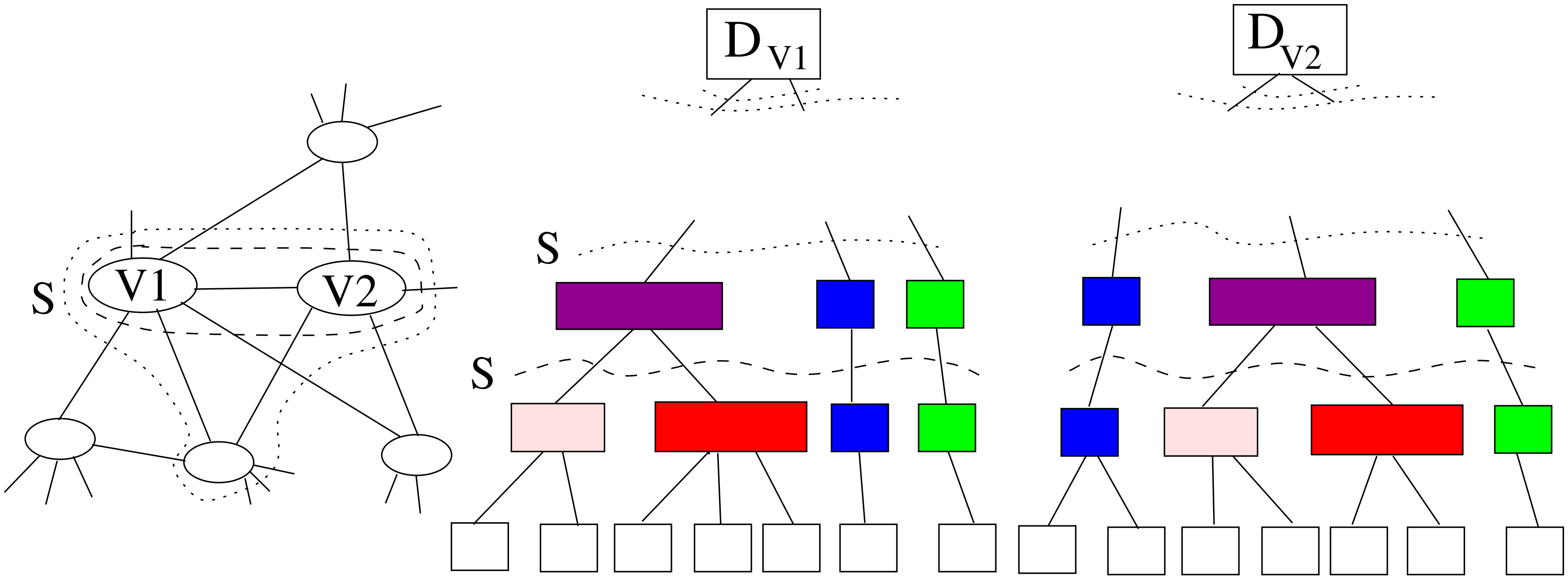
Enlarging $\mathcal{S}$ ...

Adaptive hierarchical consistency (AHC)



$\rightarrow$ Run k-consistency checking algorithm

Combination of hierarchies of partitions



‘Synchronized’ combinations/refinements

Properties of NPI

Local & weak

→ ensures tractability

→ creates more opportunities

$$\left.\begin{array}{l}\text{Selecting}\\ \text{Enlarging}\\ \text{Restricting}\end{array}\right\} \mathcal{S}\ \textbf{to control} \left\{\begin{array}{l}\text{decomposition}\\ \text{coarseness of NPI-sets}\end{array}\right.$$

Hierarchical organization

→ rules out explosion of combinations

→ simplifies the (cheap) computation of JDT

Practical use of NPI

Interactive problem solving

→ Show alternatives in a concise manner

→ Allow users to reinforce preferences

→ Local modifications in a dynamic environment

Search

→ informed backtracking

→ adaptive hierarchical consistency

→ Monitor search to explore qualitatively different solutions, useful in local search

Explanation

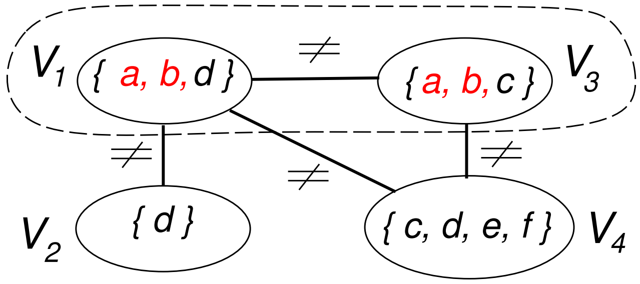
→ Interchangeable objects share characteristics, Use background knowledge to generate concise descriptions **dynamically**

Constraints of mutual exclusion

Highly disjunctive: $\begin{cases} \text{big branching-factor} \\ \text{poor propagation} \end{cases}$

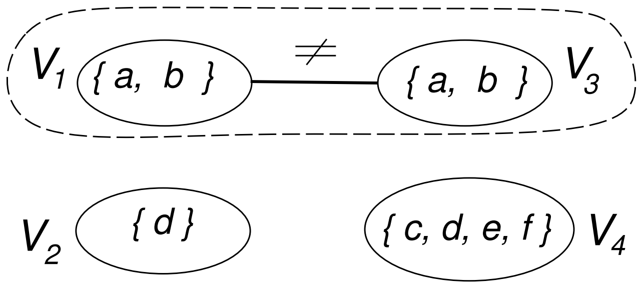
→ values for V_i are NPI iff $\notin$ domains of $\texttt{Neigh}(V_i)$

$$D_{V_i} - \bigcup_{V_j \in \texttt{Neigh}(\mathcal{S}) \cap \texttt{Neigh}(\{V_i\})} D_{V_j}$$



→ NPI-set is unique

→ NPI sets ⇒ independent subproblem



→ NPI values are NSUB for all other values in D_{V_i}

Empirical tests

- **Real-world applications**

  $$\rightarrow \text{Resource allocation, VAD [IJCAI'95]} \begin{cases} \text{NI} \quad \surd \\ \text{NPI not yet} \\ \text{IS not yet} \end{cases}$$

  → Nurse scheduling [Weil & Heus'95]

- **Random problems**

  → NI in [Benson & Freuder'92]

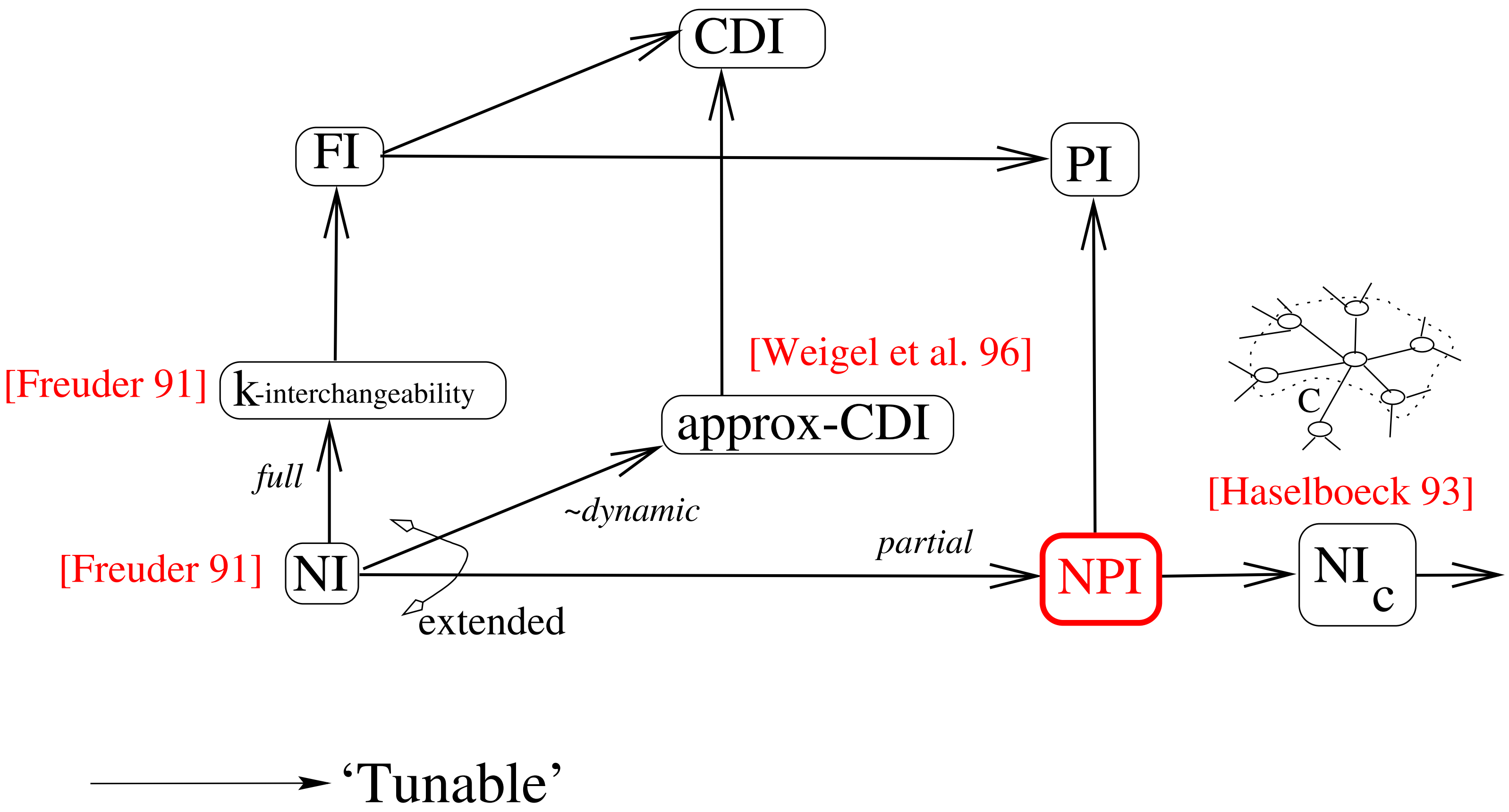
  → NI_C [Haselböck'93]

  → NI in subproblems [Weigel & Faltings'97]

  .. **and since**, NPI is weaker than NI, then..

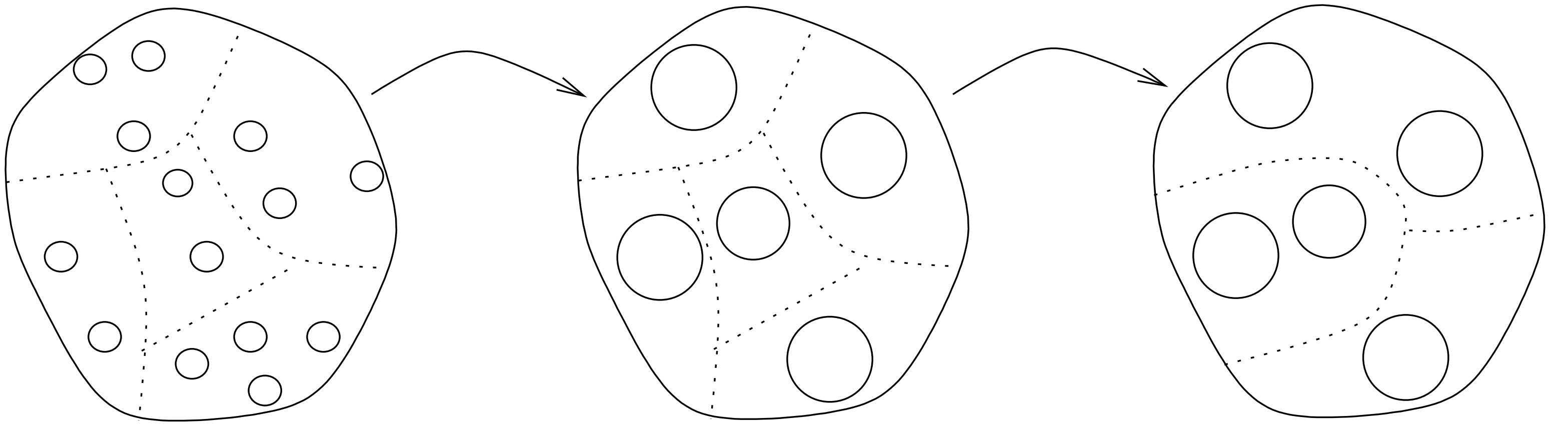
  but, testing on random problems is debatable

Lattice of some interchangeability relations



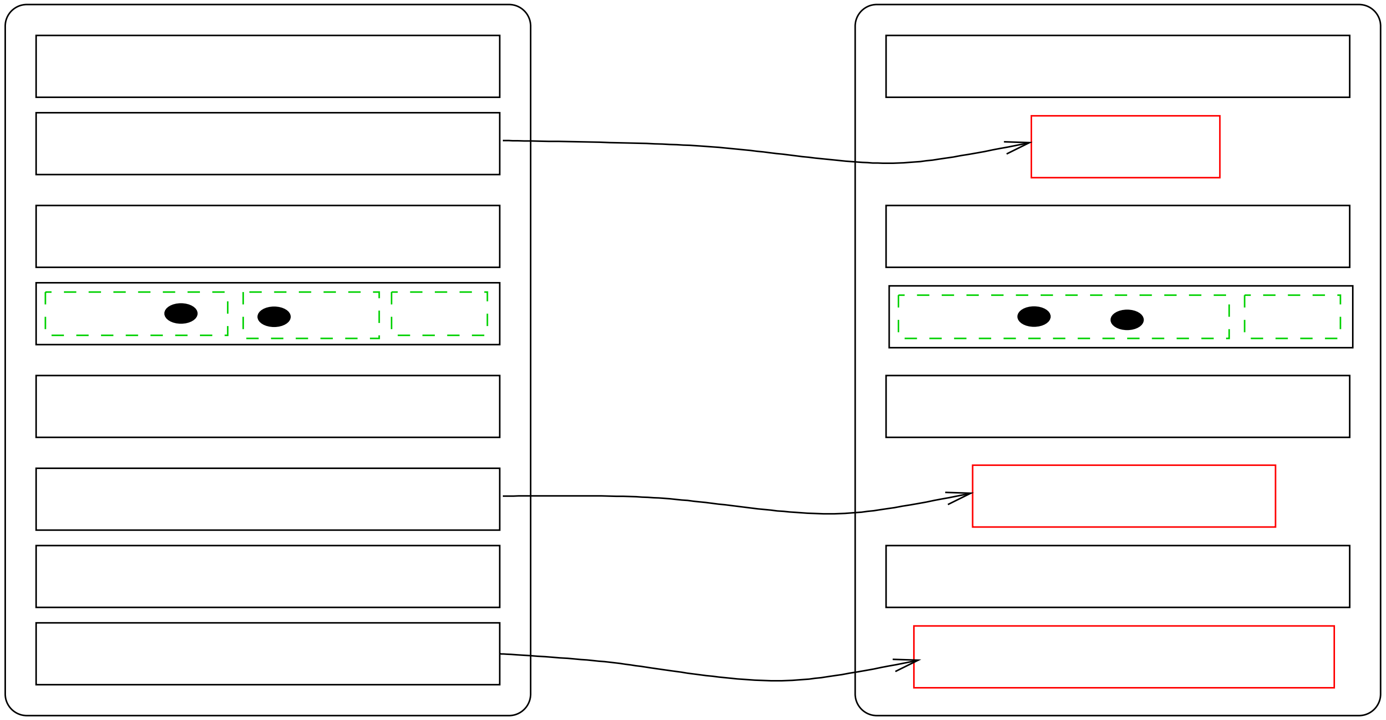
Meta-interchangeability for variables

Weigel & Faltings'97



Context-Dependent Interchangeability

Weigel, Faltings & Choueiry'96



Contributions

- **Partial interchangeability**

  $\rightarrow$ NPI: definition & computation

  $\rightarrow$ Identification of independent subproblems

  $\rightarrow$ Hierarchical structure of domain partitions

- **Special case: mutual exclusion**

- **Related work**

  $\rightarrow$ Empirical results $\begin{cases} \text{Random problems: questionable} \\ \text{Real problems: too few} \end{cases}$

  $\rightarrow$ Conceptual side: Lattice

Further research

- Evaluation on **real-world** problems
  random problems are not relevant

- Strategies for selecting, enlarging & refining $\mathcal{S}$ to
  $\rightarrow$ generate 'good' decompositions
  $\rightarrow$ determine 'appropriate' coarseness of NPI-sets

- As for `Mutex`, types of constraints for which
  local interchangeability is easily computed

- Extension: n-ary constraints, continuous domains