Title: CSP 101 (cont.)

Material: Constraint Satisfaction, Dechter MIT Encyclopedia
CSP, An Overview, Pedro Meseguer
Constraint Processing, Chapter 1 (Dechter)

Foundations of Constraint Satisfaction
CSCE 990-005, Fall 2000

Berthe Y. Choueiry (Shu-we-ri)
Ferguson Hall, Room 104
`choueiry@cse.unl.edu`, Tel: (402)472-5444

Domain types

Given $\mathcal{P} = (\mathcal{V}, \mathcal{D}, \mathcal{C}) \begin{cases} \mathcal{V} = \{V_1, V_2, \ldots, V_n\} \\ \mathcal{D} = \{D_{V_1}, D_{V_2}, \ldots, D_{V_n}\} \\ \mathcal{C} \text{ set of constraints} \end{cases}$

$$C_{V_a, V_b, \ldots, V_i} = \{\, (x, y, \ldots, z)\} \subseteq D_{V_a} \times D_{V_b} \times \ldots \times D_{V_i}$$

Domains:

$\longrightarrow$ Finite (discrete), enumeration works

$\longrightarrow$ Continuous, sophisticated algebraic techniques are needed

Terminology

Given $\mathcal{P} = (\mathcal{V}, \mathcal{D}, \mathcal{C}) \ldots$

Query:

- Find a solution
- Find all solutions
- Find number of solutions
- Can you extend a partial solution, *etc.*

Constraints:

- *arity*: universal, unary, binary, ternary, ... global
- *scope*: set of variables to which the constraint applies
- *definition*: implicit, explicit (set of tuples, binary matrix, relation)

Representation: (but not necessarily)

- contraint graph for binary constraints
- contraint network for non-binary constraints

Complexity of CSP

Characterization:

Decision problem. In general, NP-complete by reduction from 3SAT

Steps for proving NP-completeness:

check handout from Chapter 3 of Garey & Johnson

What is SAT?

Assignement of truth values to Boolean variables such that a (conjunction of a) set of logical clauses is satisfied.

Boolean variable → term

Term or its negation → literal

Clause → disjunction of literals

SAT → conjunction of clauses

Problem reduction:

Example: CSP into SAT

Notation: variable-value pair = VVP

- VVP $\rightarrow$ term
- VVPs of a variable $\rightarrow$ disjunction of terms
- Constraint:
  tuple $\rightarrow$ conjunction of terms
  constraint $\rightarrow$ disjuction of these conjunctions
  $\longrightarrow$ transform into conjunctive nomal form (CNF)

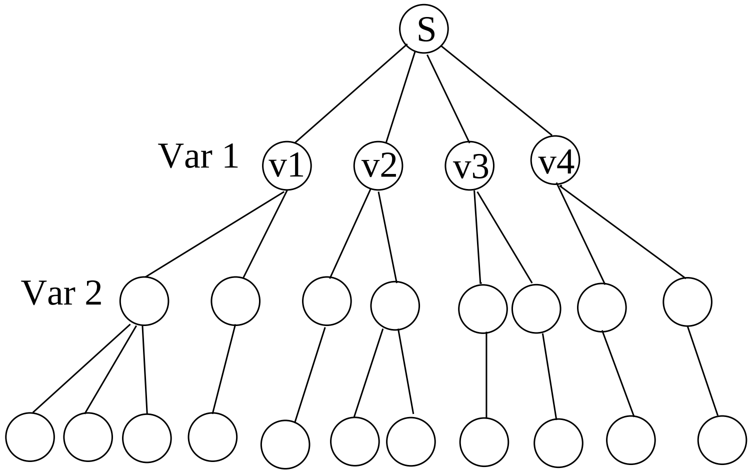
How to solve a CSP? search $\begin{cases} \text{Constructive, systematic} \\ \text{Iterative repair} \end{cases}$

Systematic search: $\rightarrow$ Starting from a root node

$\rightarrow$ Consider all values for a variable V_1

$\rightarrow$ For every value for V_1, consider all values for V_2

$\rightarrow$ etc..



For n variables, each of domain size d:

- Maximum depth? *fixed!*

- Maximum number of paths? *size of search space, size of CSP*

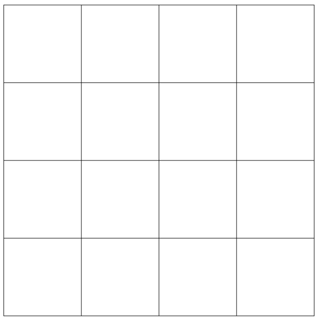
Before looking at search, consider

1. Importance of modeling/formulating
   to control the size of the search space

2. Preprocessing: consistency checking/ filtering, constraint propagation
   to reduce size of search space

Importance of modeling

N-**queens**: formulation 1

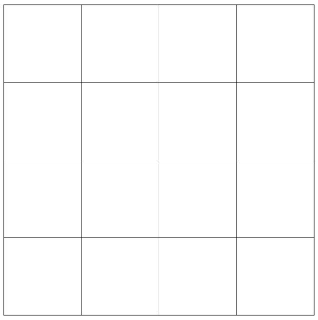


Variables?

Domains?

Size of CSP?

N-**queens**: formulation 2



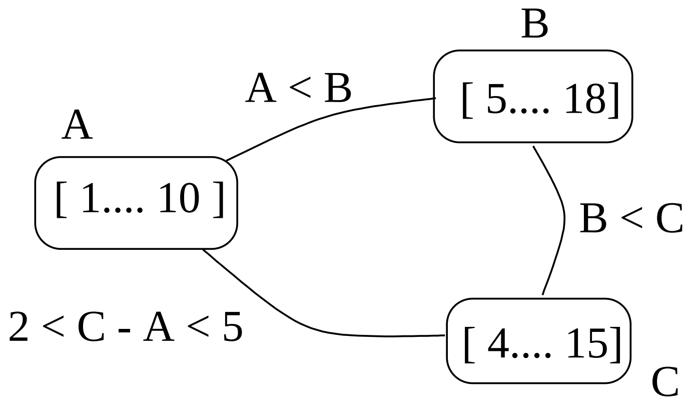
variables?

domains?

size of csp?

Constraint checking

→ arc-consistency



1- B: [5 .. 14]
C: [6 .. 15]

2- A: [2 .. 10]
C: [6 .. 14]

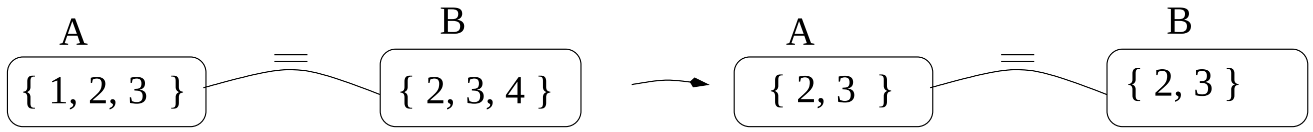
3- B: [5 .. 13]

⟶ 3-consistency, k-consistency ($k \leq n$)

⟶ Constraint filtering, constraint checking, etc..

eliminate non-acceptable tuples prior to search

→ Warning: arc-consistency does not solve the problem

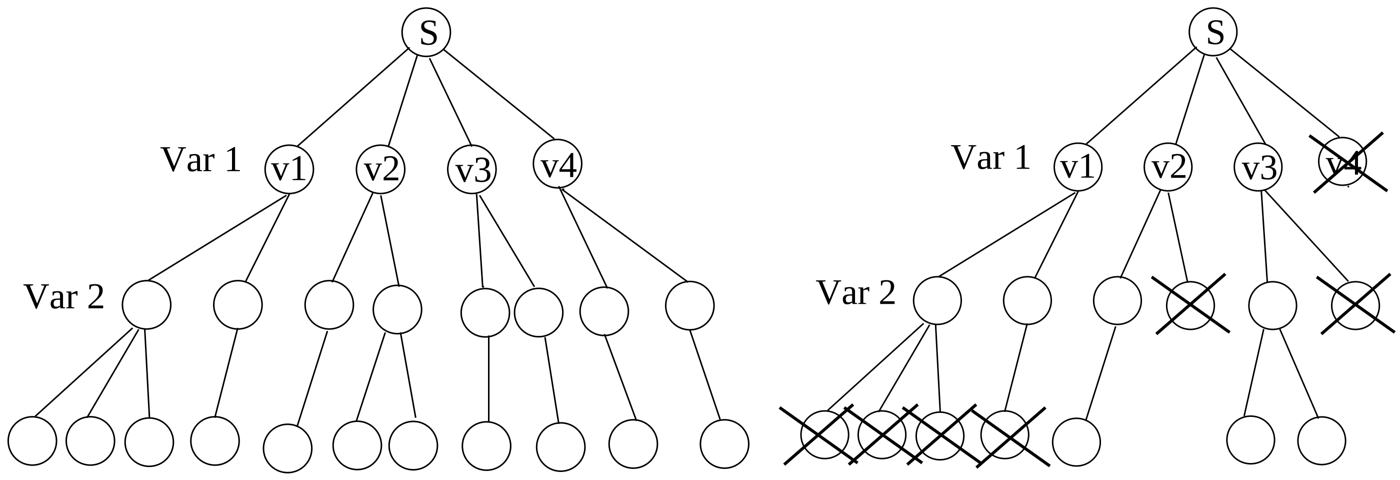


(A=2) ∧ (B=3) still isn't a solution!

Systematic search (I) ***Back-checking***

Systematic search generates d^n possibilities

Are all possibilities acceptable?

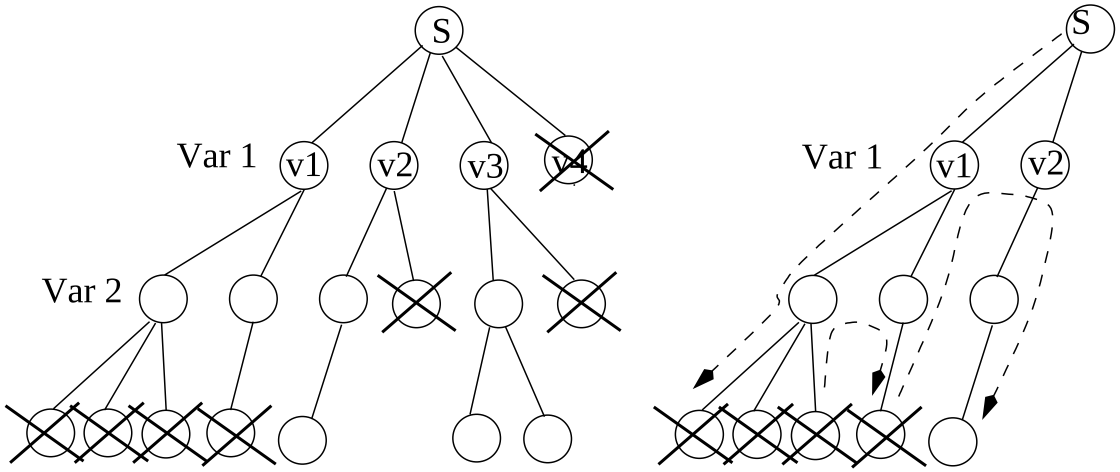


$\rightarrow$ Expand a partial solution only when consistent

$\longrightarrow$ **early pruning**

Systematic search (II) ***Chronological backtracking***

What if only one solution is needed?



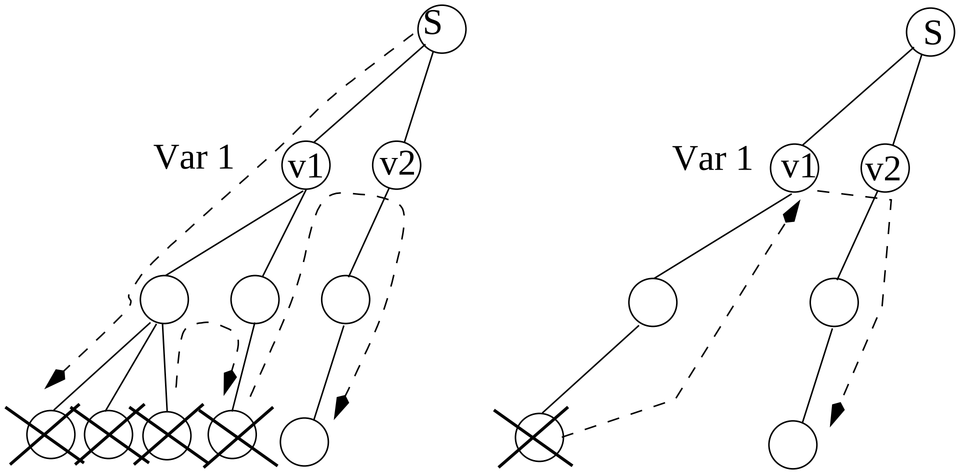
$\longrightarrow$ **Depth-first search & chronological backtracking**

$\rightarrow$ DFS: soundness? completeness?

Systematic search (III) ***Intelligent backtracking***

What if the reason for failure was higher up in the tree?

Backtrack to source of conflict!!



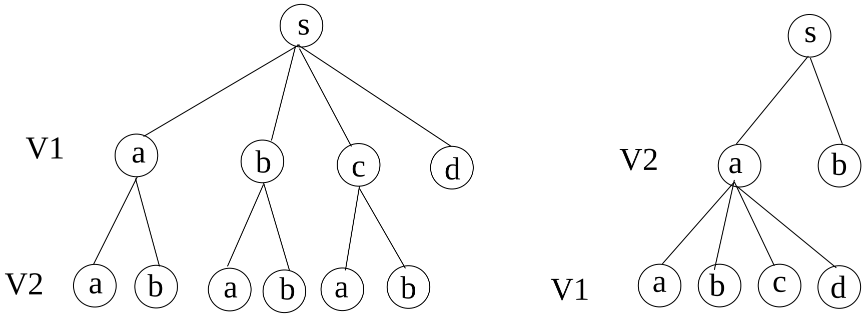
$\rightarrow$ Backjumping, conflict-directed backjumping, etc.

Systematic search (IV) — ***Ordering heuristics***

Which variable to expand first?

Exp: $V_1, V_2,\ D_{V_1} = \{a, b, c, d\}, D_{V_2} = \{a, b\}$

Sol: $\{(V_1 = c), (V_2 = a)\}$ and $\{(V_1 = c), (V_2 = b)\}$

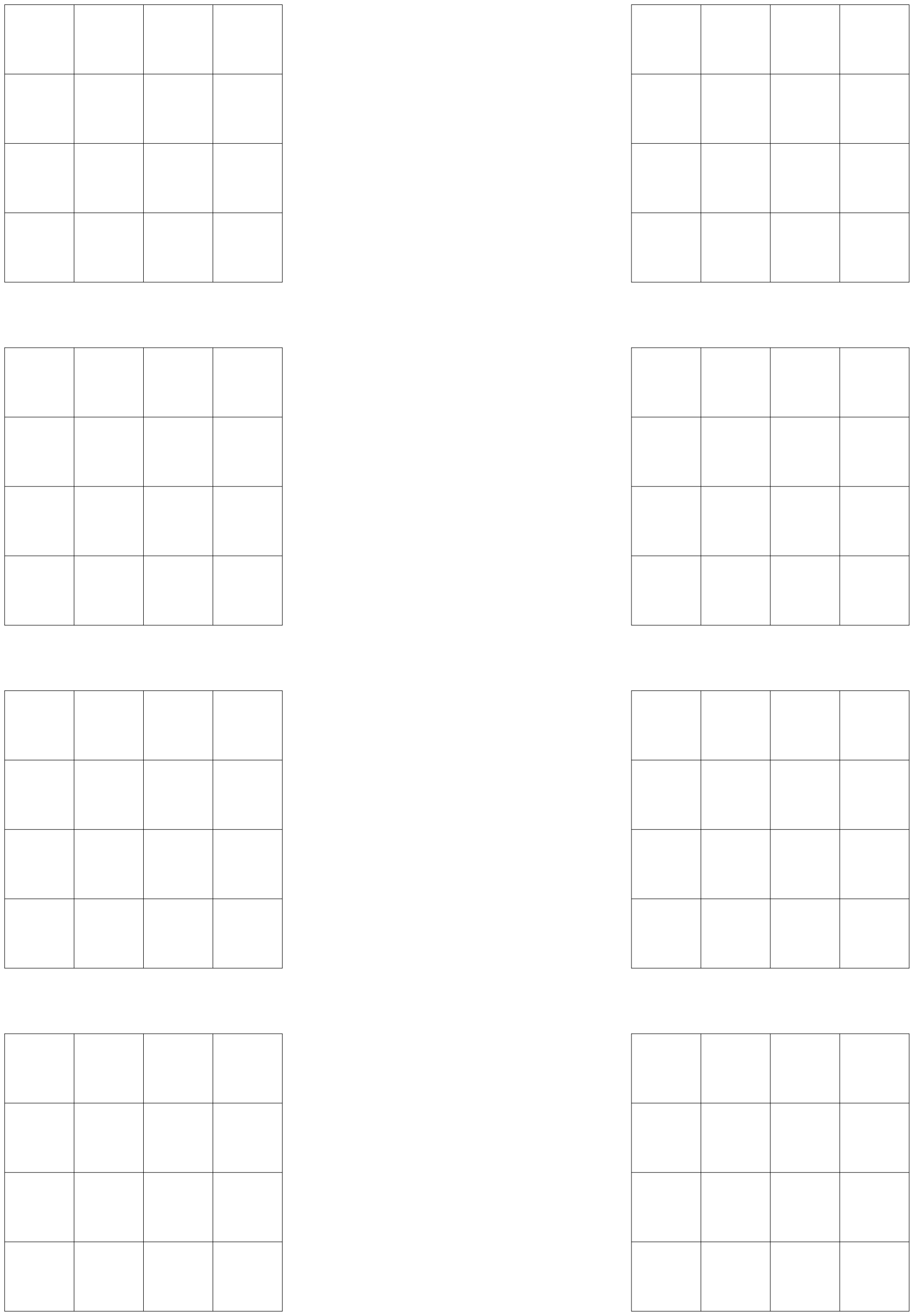


$$\text{Heuristics: } \begin{cases} \text{most } \underline{\text{constrained}} \text{ variable first (reduce branching factor)} \\ \text{most } \underline{\text{promising}} \text{ value first (find quickly first solution)} \end{cases}$$

$$\text{Strategies for } \left\{ \begin{array}{l} \text{variable ordering} \\ \text{value ordering} \end{array} \right\} \text{ could be } \begin{cases} \text{static} \\ \text{dynamic} \end{cases}$$

Systematic search (V) ***Forward checking***

Why not filter right away effects of an action?



CSP: a decision problem (NP-complete)

1- Modeling:

— abstraction and reformulation

2- Preprocessing techniques:

— eliminate non-acceptable tuples prior to search

3- Systematic search:

— potentially d^n paths of fixed lengths

— chronological backtracking

— intelligent backtracking

— variable/value ordering heuristics

— forward checking

4- Search 'hybrids':

— Mixing preprocessing with forward checking (MAC)

Non-systematic search

Iterative repair, local search

modifies a global but **inconsistent** solution

to decrease the number of violated constraints

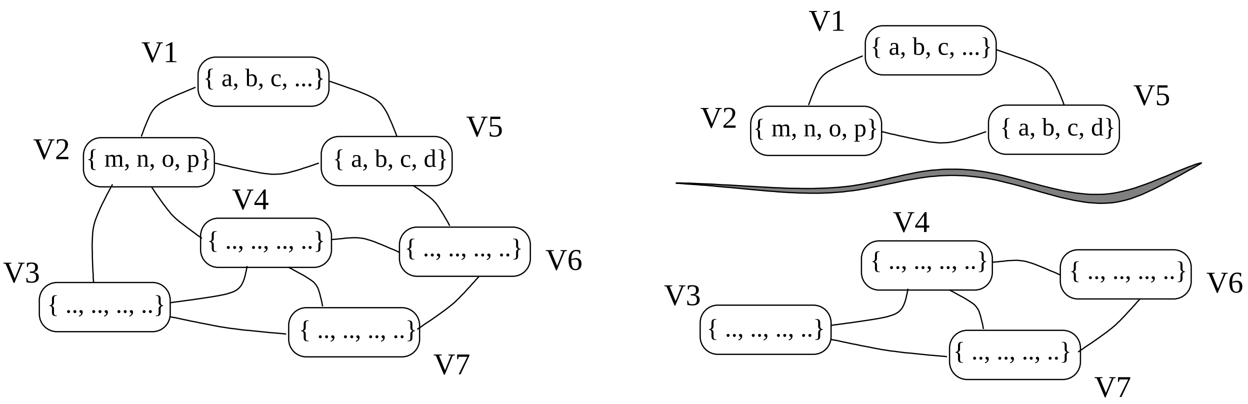
Examples: Simulated annealing, hill climbing, taboo,

GSAT, Genetic Algorithms, etc.

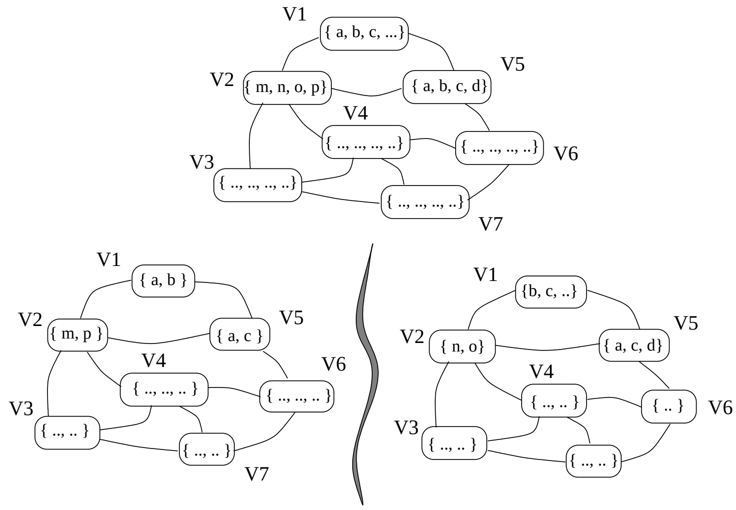
$\longrightarrow$ Incomplete & not sound

Decomposition $\begin{cases} \text{conjunctive} \\ \text{disjunctive} \end{cases}$

Conjunctive



Disjunctive



Properties

$$\mathcal{P} \longrightarrow \{\ \mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_i\ \}$$

Conjunctive vs. disjunctive

Consistent	No constraint is removed
Simplifying	Size($\mathcal{P}_i$) $<$ Size($\mathcal{P}$)
Semi-complete	At least one solution is kept
Complete	No solution is lost
Redundant	Solutions replicated in $\{\mathcal{P}_i\}$
Reducible	$\sum_i$ Size($\mathcal{P}_i$) may be $<$ Size($\mathcal{P}$)

Deep analysis

Uncover particular properties, *e.g.*
- bounds the required level of consistency
- predicts ease/difficulty of solving a given instance

- **C-graph structure, topology**
  tree, DAGs, chordal, etc.
- **Constraints type**
  subsets of Allen's relations,
  all-diffs, functional,
  monotonic, row-convex, etc.
- **Order parameter** (phase transition)

Research directions

Preceding (i.e., search, backtrack, iterative repair, V/V/ordering, consistency checking, decomposition, symmetries & interchangeability, deep analysis) + ...

- **Evaluation of algorithms**:
  worst-case analysis, empirical studies
  random problems?
- **Cross-fertilization**:
  SAT, DB, math. programming,
  interval math, planning, etc.
- **Modeling & Reformulation**
- **Multi-agents**:
  Distribution and negotiation
  $\rightarrow$ decomposition & alliance formation

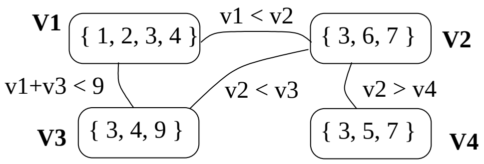
Multi-agent

- Problem:
  decomposed, replicated
- Agent:
  broker, solver, problem, solver+problem
- Architecture and authority:
  hierarchical, egalitarian
  priority-based (e.g., vote)
- Communication:
  agent-to-agent, group, broadcast
- Interaction strategies:
  cooperative, competitive
  transparent, secretive

CSP in a nutshell (I)

Definition: $\mathcal{P} = (\mathcal{V}, \mathcal{D}, \mathcal{C})$

$$\begin{cases} \mathcal{V} = \{V_1, V_2, \ldots, V_n\} \\ \\ \mathcal{D} = \{D_{V_1}, D_{V_2}, \ldots, D_{V_n}\} \\ \\ \mathcal{C} \text{ set of constraints} \end{cases}$$



Constraint graph.
Finite domains.
Binary constraints, universal constraints.

Examples: map coloring, puzzles, resource allocation, temporal reasoning, product configuration, databases, spreadsheets, graphical layouts, graphical user-interfaces, bioinformatics, etc.

CSP in a nutshell (II)

Solution technique: Search $\begin{cases} \text{constructive} \\ \text{iterative repair} \end{cases}$

Enhancing search:

$$\begin{cases} \text{intelligent backtrack} \\ \text{variable/value ordering} \\ \text{consistency checking} \\ \text{forward-checking} \\ \text{symmetries (interchangeability)} \\ \text{decomposition} \end{cases}$$

Deep analysis: $\begin{cases} \text{graph structure} \\ \text{constraints semantics} \\ \text{phase transition} \end{cases}$

Open:

$$\begin{cases} k\text{-ary constraints} \\ \text{evaluation of algorithms} \\ \text{cross-fertilization (mathematical program.)} \\ \text{reformulation and approximation} \\ \text{architectures (multi-agent, negotiation)} \end{cases}$$