Title: Constraint Satisfaction

Required reading:

Chapter 3 Section 3.7 and Chapter 4, page 114 (AIMA)
Introduction to CSPs (Bartak's on-line guide)

Recommended reading:

"Algorithms for Constraints Satisfaction problems: A Survey" by Vipin Kumar. AI Magazine, Vol 13, No 1, 32-44, 1992.
Constraint Programming: In Pursuit of the Holy Grail. Bartak

Introduction to Artificial Intelligence
CSCE 476-876, Spring 2002
URL: `www.cse.unl.edu/~choueiry/S02-476-876`

Berthe Y. Choueiry (Shu-we-ri)
`choueiry@cse.unl.edu, (402)472-5444`

Glossary 5:

Assigned: Monday February 25, 2002.
Due: Monday, March 4, 2002

- Anytime algorithm
- Arc consistency
- Binary constraint
- Chronological backtracking
- Constraint (in a CSP)
- Constraint arity
- Constraint Satisfaction Problem (not AIMA!)
- Constructive search
- Forward checking
- Global constraint

Glossary 5: (cont')

Instantiated variable
Intelligent backtracking
Iterative repair, local search
Lookahead (partial, directional, full)
Maintaining Arc Consistency (MAC)
Size of a CSP
Ternary constraint
Unary constraint
Universal constraint
Variable domain
Variable/value ordering heuristic

Constraint Processing

- Constraint Satisfaction: problem(s) and algorithms
- Constraint Languages: $\begin{cases} \text{functional} \\ \text{logic} \\ \text{object-oriented} \end{cases}$

Constraint Satisfaction Problems (CSP)

- Course: Foundations of Constraint Satisfaction
- CSCE421/821, Fall 2002
- Fall'01 URL: `cse.unl.edu/~choueiry/F01-421-821/`
- Fall'00 URL: `cse.unl.edu/~choueiry/F00-CSCE990/`
- Fall'99 URL: `cse.unl.edu/~choueiry/CSE990-05/`

Outline

- Definition
- Practical usefulness
- Underlying 'science'
- Commercial successes & job market

What is this about?

Context: You are a senior in college

Problem: You need to register in 4 courses for Spring'2000

Possibilities: Many courses offered in Math, CSE, EE, *etc.*

Constraints: restrict the choices you can make

- *Unary:* Courses have prerequisites you have/don't have
  Courses/instructors you like/dislike
- *Binary:* Courses are scheduled at the same time
- *n-ary:* In CompEng, 4 courses from 5 tracks such as at least 3 tracks are covered

You have choices, but are restricted by constraints

$\longrightarrow$ Make the right decisions

Practical Applications

Adapted from E.C. Freuder

- Radio resource management (RRM)
- Databases (computing joins, view updates)
- Temporal and spatial reasoning
- Planning, scheduling, resource allocation
- Design and configuration
- Graphics, visualization, interfaces
- Hardware verification and software eng.
- HCInteraction and decision support
- Molecular biology
- Robotics, machine vision and computational linguistics
- Transportation
- Qualitative and diagnostic reasoning

Constraint Satisfaction

Given

- A set of variables *4 courses at UNL*
- For each variable, a set of choices (values)
- A set of constraints that restrict the combinations of values the variables can take at the same time

Questions

- Does a solution exist? *classical decision problem*
- How two or more solutions differ? How to change specific choices without perturbing the solution?
- If there is no solution, what are the sources of conflicts? Which constraints should be retracted?
- *etc.*

Constraint Processing is about

- solving a decision problem
- while allowing the user to state arbitrary constraints in an expressive way and
- providing concise and high-level feedback about alternatives and conflicts

Power of Constraints Processing

- flexibility & expressiveness of representations
- interactivity, users can $\left\{ \begin{array}{c} \text{relax} \\ \text{reinforce} \end{array} \right\}$ constraints

Related areas: AI, OR, Algorithmic, DB, Prog. Languages, *etc.*

CSP: Definition

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

$$C_{V_k,V_l,V_m} = \{\,(x, y, z)\} \subseteq D_{V_k} \times D_{V_l} \times D_{V_m}$$

Query: can we find one value for each variable
such that all constraints are satisfied?

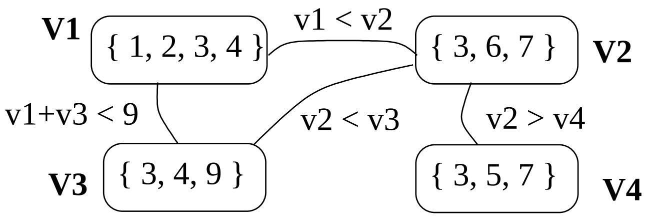
In general, **NP-complete**

Representation

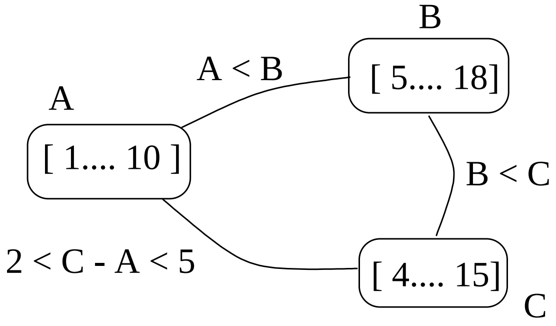
Given $\mathcal{P} = (\mathcal{V}, \mathcal{D}, \mathcal{C}) \begin{cases} \mathcal{V} = \{V_1, V_2, \ldots, V_n\} \\ \mathcal{D} = \{D_{V_1}, D_{V_2}, \ldots, D_{V_n}\} \\ \mathcal{C} \text{ set of constraints} \end{cases}$

$$C_{V_k,V_l,V_m} = \{\, (x, y, z)\} \subseteq D_{V_k} \times D_{V_l} \times D_{V_m}$$

Constraint **graph**

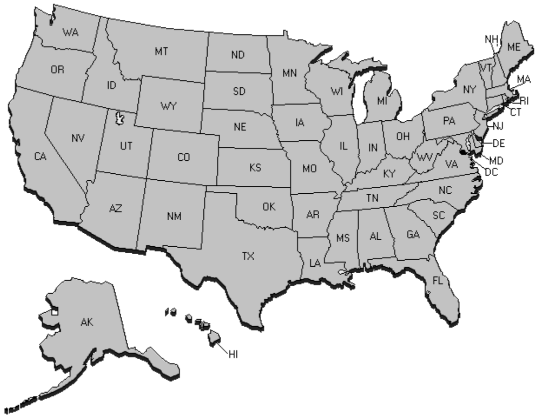


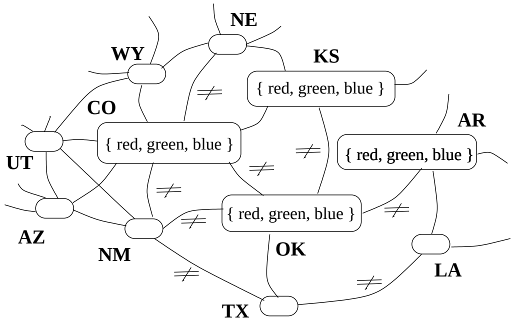
Example I: Temporal reasoning



Example II: Map coloring

Using 3 colors (R, G, & B), color the US map such that no two adjacent states do have the same color





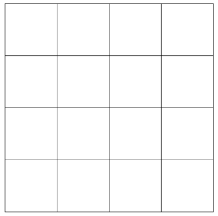
Variables? Domains? Constraints?

Example III: n-queens problem

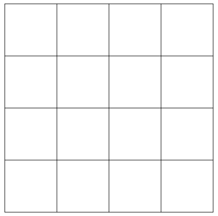
Variables?

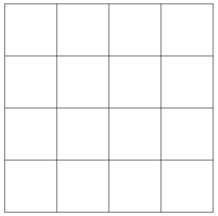
Domains?

Constraints?



Incremental formulation $\left\{\begin{array}{l}\text{State?}\\ \\ \text{Action?}\end{array}\right.$





In general

What is a state in a CSP?

What is an action in a CSP?

What is the goal state in a CSP?

What is a state in a CSP?
→ A set of instantiated variables
and a set of uninstantiated variables

What is an action in a CSP?
→ The instantiation of one more variables
(instantiation = assignment of a value to a variable)

What is the goal state in a CSP?
→ All variables are instantiated such that
all constraints are satisfied

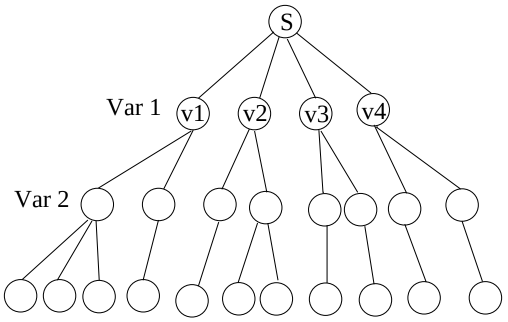
How to solve a CSP?

Search

1. Constructive, systematic
2. Iterative repair

Systematic search

→ Starting from a root node
→ Consider all values for a variable V_1
→ For every value for V_1, consider all values for V_2
→ etc..



For n variables, each of domain size d:
- Maximum depth? *fixed!*
- Maximum number of paths? *size of search space, size of CSP*

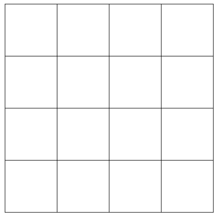
Before looking at search..

Consider

1. Importance of modeling/formulating
   to control the size of the search space
2. Preprocessing: consistency filtering
   to reduce size of search space

Importance of modeling

N-**queens**: formulation 1

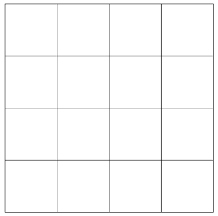


Variables?

Domains?

Size of CSP?

N-**queens**: formulation 2



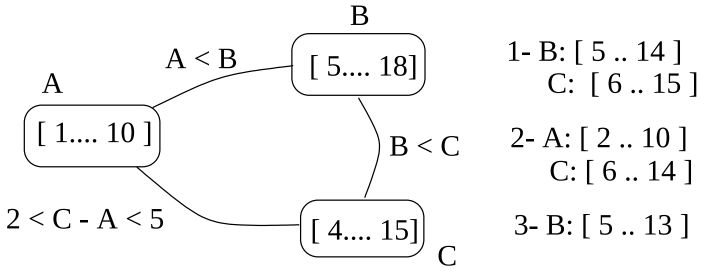
variables?

domains?

size of csp?

Constraint checking

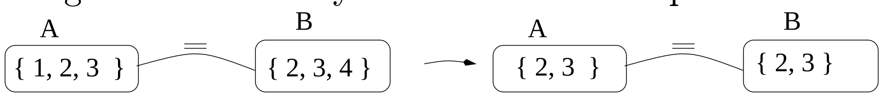
→ arc-consistency



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

⟶ Constraint filtering, constraint checking, etc..
eliminate non-acceptable tuples <u>prior</u> to search

→ Warning: arc-consistency does not solve the problem

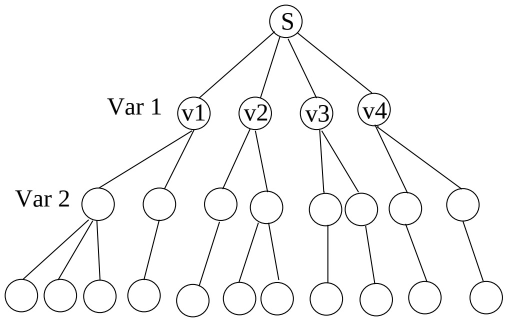


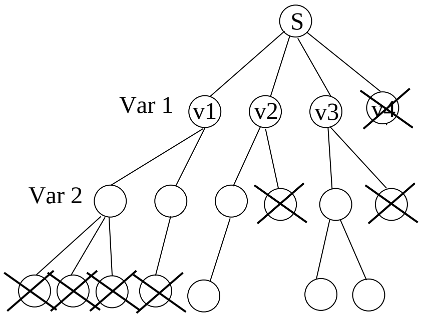
(A=2) ∧ (B=3) still isn't a solution!

Search (I) ***Back-checking***

Systematic search generates d^n possibilities

Are all possibilities acceptable?



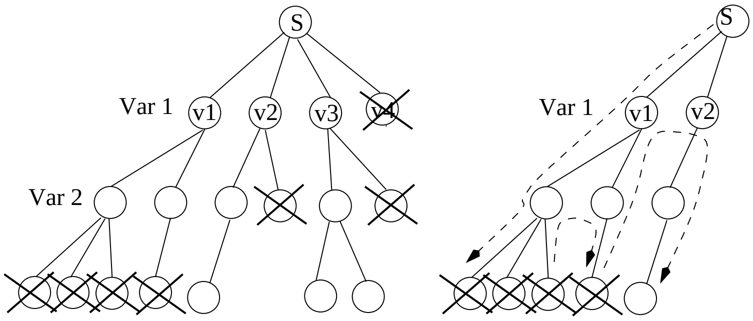


→ Expand a partial solution only when consistent

⟶ **early pruning**

Search (II) *Chronological backtracking*

What if only <u>one</u> solution is needed?



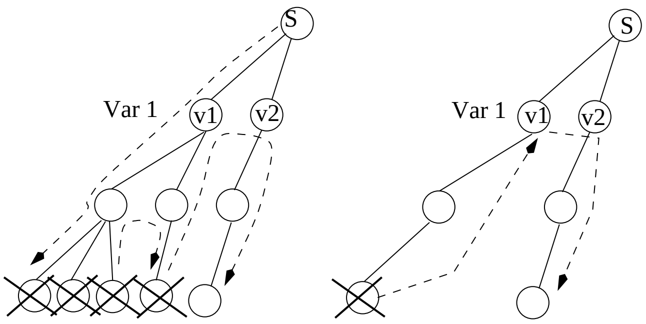
⟶ **Depth-first search & chronological backtracking**

→ DFS: soundness? completeness?

Search (III) *Intelligent backtracking*

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

Backtrack to source of conflict!!

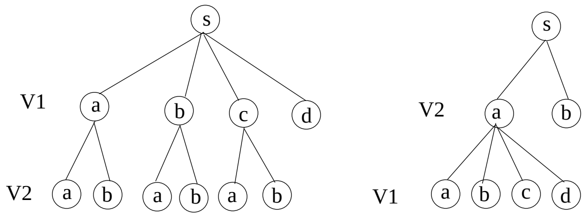


→ Backjumping, conflict-directed backjumping, etc.

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 $\{(V_1 = c), (V_2 = b)\}$

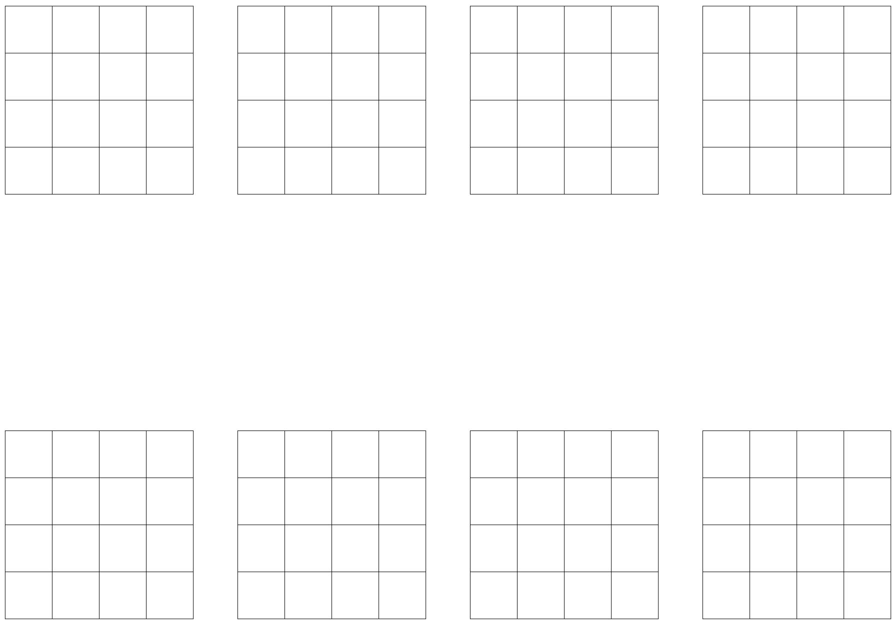


Heuristics: $\left\{ \begin{array}{l} \text{most } \underline{\text{constrained}} \text{ variable first (reduce branching factor)} \\ \text{most } \underline{\text{promising}} \text{ value first (find quickly first solution)} \end{array} \right.$

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

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- 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: Hill climbing, taboo search, simulated annealing, GSAT, WalkSAT, Genetic Algorithms, etc.

$\longrightarrow$ Incomplete & not sound

Advantage: anytime algorithm
Shortcoming: Cannot tell that no solution exist

Domain types

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

$$C_{V_k, V_l, V_m} = \{ (x, y, z)\} \subseteq D_{V_k} \times D_{V_l} \times D_{V_m}$$

Domains:

$\longrightarrow$ Finite (discrete), enumeration works

$\longrightarrow$ Continuous, sophisticated algebraic techniques are needed

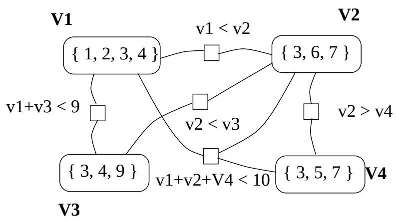
Constraint arity

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

$$C_{V_k, V_l, V_m} = \{ (x, y, z)\} \subseteq D_{V_k} \times D_{V_l} \times D_{V_m}$$

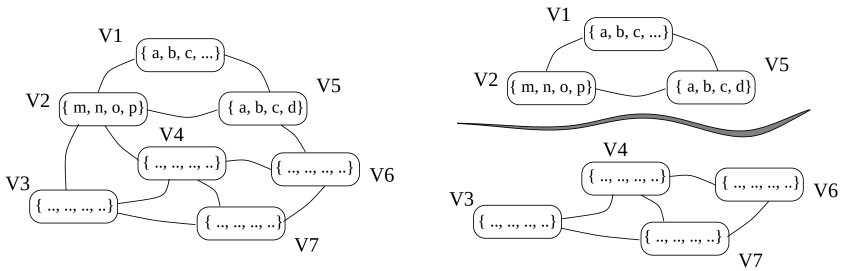
Constraints: universal, unary, binary, ternary, ... global

Constraint network

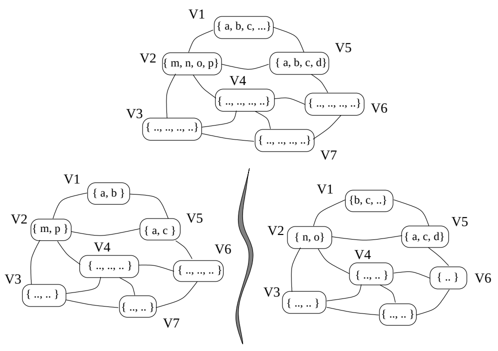


Decomposition { conjunctive / disjunctive

Conjunctive



Disjunctive



Deep analysis

Uncover particular properties, *e.g.*

- bounds the required level of consistency
- predicts ease/difficulty of solving a given instance

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

Multi agents

1. **Computational tasks:**
   problem decomposed, replicated
2. **Types of Agent:**
   broker, solver, problem, solver+problem
3. **Architecture and authority:**
   hierarchical, egalitarian, priority-based (e.g., vote)
4. **Nature of Communication:**
   agent-to-agent, group, broadcast
5. **Interaction strategies:**
   cooperating vs. competing
   transparent vs. secretive
   negotiation + alliance formation

Computational tasks

At one end of the spectrum, agents may be involved in solving heterogeneous, distinct and completely independent problems and request other agents to supply specific functionalities for the completion of the individual tasks.

At the other end of the spectrum, the same problem could be replicated and assigned to all agents, which can then share their individual results with other agents incrementally in order to speed up the execution of the global computational task.

Types of agent

An agent in the system can be any of the following:

- an agent that collects data from the environment and formulate the CSP
- a reasoning module with specific computational characteristics (e.g., various search algorithms)
- brokers that facilitate matching between a service seeker and a number of service providers (e.g., CORBA brokers).

Agent architecture and how is authority granted

Agents could be organized in a strict hierarchy in which a given agent has full control over the activities of the agents that lie underneath it in the hierarchy. It decides how the lower level agents may cooperate while ensuring coordination with the higher-level agent.

Agents could be in a strictly flat structure competing for services and rewards, either chaotically or according to some strict priority policy, based on voting or time-responsiveness.

Nature of the communication environment

Communications among agents may be conducted according to:

- a one-to-one schema
- multi-cast (i.e., one-to-group), or
- broadcast, where all agents in the environment have access to the content of the communicated information.

Type of supported interactions

Agents may be cooperative, pooling their resources and capabilities to achieve a common, global objective, or they could be competitive trying to win rewards and optimize their individual gain.

They could also adopt a midway strategy, dynamically forming coalitions and gathering support to acquire more resources and realize greater gains.

Also, agents may be transparent about their intentions, resources, needs, and constraints or may be secretive, hiding one or the other of their strengths or weaknesses.

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 vs. empirical studies
random problems?

Cross-fertilization:

SAT, DB, mathematical programming,
interval mathematics, planning, etc.

Modeling & Reformulation

Multi agents:

Distribution and negotiation
→ decomposition & alliance formation

CSP in a nutshell (I)

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

Enhancing search: $\begin{cases} \text{intelligent backtrack} \\ \text{variable/value ordering} \\ \text{consistency checking} \\ \text{hybrid search} \\ \heartsuit\ \text{symmetries} \\ \heartsuit\ \text{decomposition} \end{cases}$

CSP in a nutshell (II)

Deep analysis: exploit problem structure
- ♡ graph topology
- ♡ constraint semantics
- phase transition

Research:
- k-ary constraints, soft constraints
- continuous vs. finite domains
- evaluation of algorithms (empirical)
- cross-fertilization (mathematical program.)
- ♡ reformulation and approximation
- ♡ architectures (multi-agent, negotiation)

Constraint Logic Programming (CLP)

A merger of

$\surd$ Constraint solving

$\longrightarrow$ Logic Programming, mostly Horn clauses (*e.g.,* Prolog)

Building blocks

- Constraint: primitives *but also user-defined*
  - cumulative/capacity (linear ineq), MUTEX, cycle, *etc.*
  - domain: Booleans, natural/rational/real numbers, finite
- Rules (declarative): a statement is a conjunction of constraints and is tested for satisfiability before execution proceeds further
- Mechanisms: satisfiability, entailment, delaying constraints

Your future: jobs

Commercial companies: I2 Technologies, Trilogy, Ilog, PeopleSoft/Red Pepper, Carmen Systems (Sweden), J.D. Edwards/Numetrix, COSYTEC, *etc.*

Prestigious research centers: NASA Ames, JPL, BT Labs (UK), IBM Watson+Alamaden, *etc.*

Start your own: Blue Pumpkin, Parc Technologies Ltd, Calico, Firepond, *etc.*

Academic: constraint languages, modeling, representation, automated reasoning, *etc.*

Constraint Processing Techniques are the basis of new languages:

> *Were you to ask me which programming paradigm is likely to gain most in commercial significance over the next 5 years I'd have to pick Constraint Logic Programming (CLP), even though it's perhaps currently one of the least known and understood. That's because CLP has the power to tackle those difficult combinatorial problems encountered for instance in job scheduling, timetabling, and routing which stretch conventional programming techniques beyond their breaking point.*
> *Though CLP is still the subject of intensive research, it's already being used by large corporations such as manufacturers Michelin and Dassault, the French railway authority SNCF, airlines Swissair, SAS and Cathay Pacific, and Hong Kong International Terminals, the world's largest privately-owned container terminal.*

Byte, Dick Pountain