A Generalized Framework for Constraint Planning

Roman Barták

Department of Theoretical Computer Science
Charles University in Prague

e-mail: bartak@kti.mff.cuni.cz
URL: http://kti.ms.mff.cuni.cz/~bartak/

Constraints

constraint = a relation over some domain D

Example: X=5, length([a,b,c,],N)…

constraint system = a finite (multi)set of constraints

solution = a valuation of variables in constraints such that all constraints are satisfied

Example:{X/5,N/3}

under-constrained / over-constrained systems

Why constraints?

- efficiency issues
- declarative character
- various domains

areas of usage:

scheduling, planning, computer graphic, CLP

© Roman Barták 1997

Constraint Hierarchies

approach to solving over-constrained systems

hard (required) and soft (preferential) constraints => hierarchy

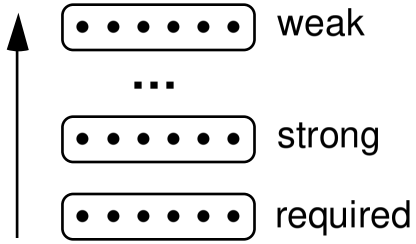
hierarchy = a finite (multi)set of labeled constraints

solution = a valuation of variables in constraints such that all required constraints are satisfied and "as many as possible" preferential constraints are satisfied

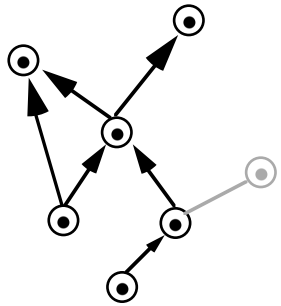
rule: satisfaction of a stronger constraint is preferred to satisfaction of arbitrary number of weaker constraints

algorithms for solving constraint hierarchies:

refining method



local propagation



© Roman Barták 1997

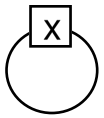
Constraint Cells

constraint cell = a finite (multi)set of equally preferred constraints with given sets of input and output variables

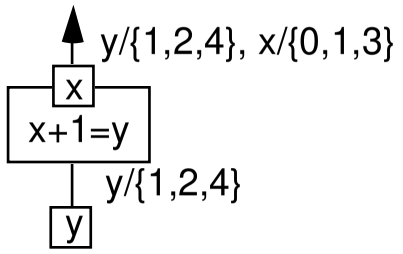
classification:

current approach (local propagation)

free variable

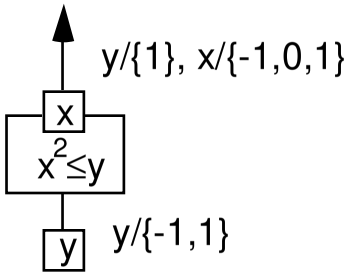


functional cell

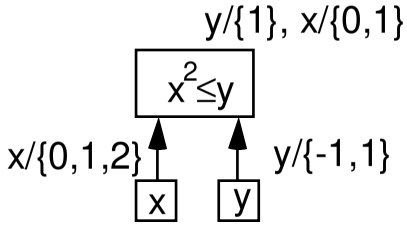


new types (potentially unsatisfied cells)

generative cell



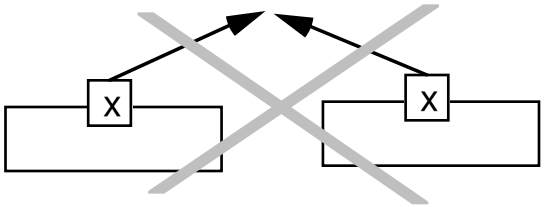
test cell



© Roman Barták 1997

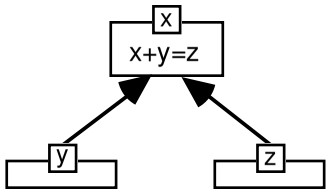
Constraint Networks

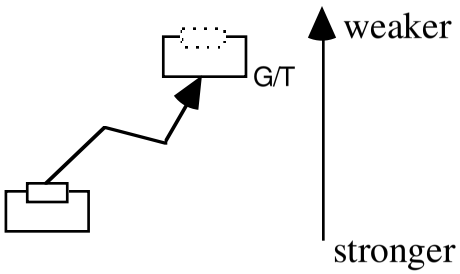
conflict-free hierarchy decomposition

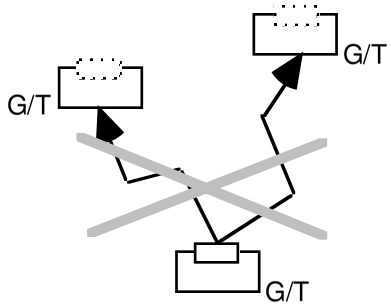


constraint network = a directed acyclic graph (nodes ≈ cells)

- defines partial ordering of cells





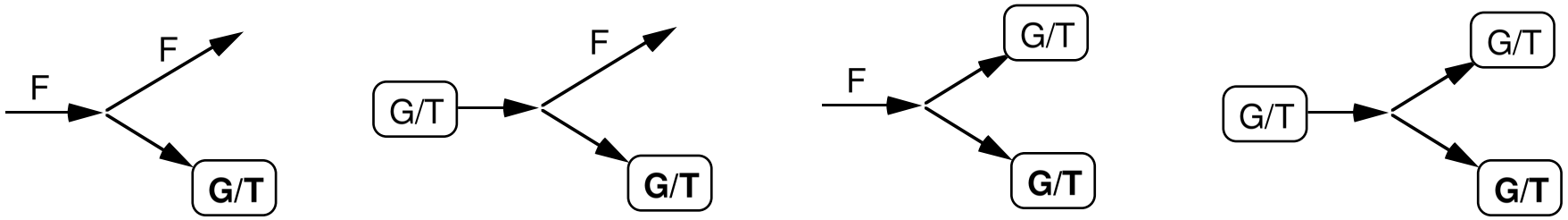


© Roman Barták 1997

Justification (sketch)

dissatisfaction of a constraint is not caused by satisfaction of a weaker (or equally preferred) constraint

Note: dissatisfied constraint can only be in generative or test cell



- remaining questions:
  - ?? constraint hierarchy -> constraint network ??
  - ?? effectiveness of transformation ??

© Roman Barták 1997

Constraint Planners

- incremental construction of constraint networks

Simple Planner

- keeps all equally preferred constraints in one cell
- corresponds to refining method

Advanced Planner

- keeps up the network as structured as possible
- exploits more local propagation

Algorithm (schema):

1) add constraint as a functional cell
2) if not possible, add it as a generative/test cell
3) add necessary edges/change orientation
4) if necessary, join cells

© Roman Barták 1997

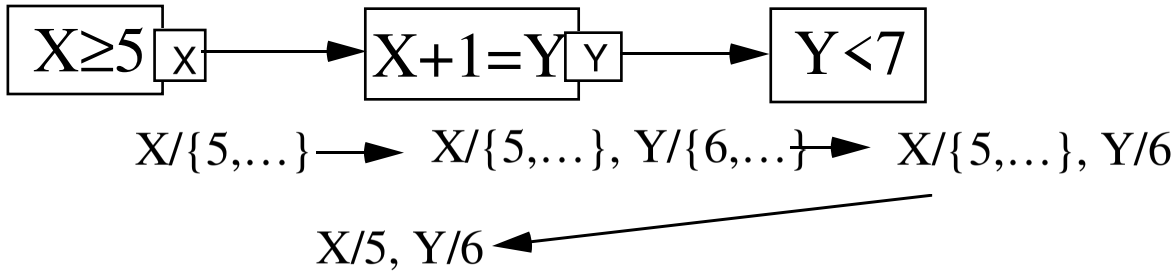
Constraint Executors

- traverse constraint network and propagate values (valuations)

Algorithm (schema):

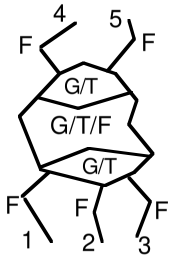
1) sort topologically the cells
2) propagate values through cells
   2.1) if the values of input variables are changed, re-test satisfaction of influenced constraints

Example:

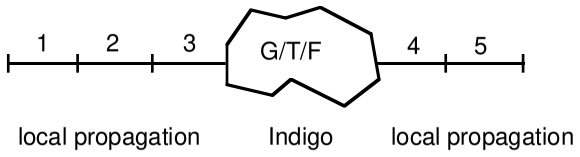


smart ordering of cells

scheme of constraint network



scheme of constraint cells ordering



© Roman Barták 1997

Final notes

Further development:

- constraint planning
  - tighten bindings to theory
  - more general planning algorithm
  - cross-platform implementation of core planner in Java
- constraint execution
  - smart orderings

---

papers on constraint planning and PROLOG source codes of above mentioned algorithms are available online at URL:

`http://kti.ms.mff.cuni.cz/~bartak/`

© Roman Barták 1997