Modèle d'hybridation pour la résolution de CSP

ROADEF 2008, Clermont-Ferrand, France

Tony LAMBERT

LIFO, Rue Léonard de Vinci 45067 ORLEANS

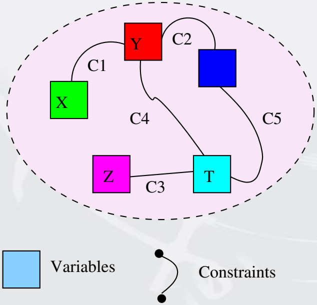
Mercredi 27 février 2008

Constraint programming process

Formulate the problem with constraints as a CSP

- constraint : a relation on variables and their domains
- Constraint Satisfaction Problem (CSP) : a set of constraints together with a set of variable domains

CSP model



- $\mathcal{X} = \{x_1, \ldots, x_n\}$ set of n variables,
- $\mathcal{D} = \{D_{x_1}, \ldots, D_{x_n}\}$ set of n domains,
- $\mathcal{C} = \{c_1, \ldots, c_m\}$ set of m constraints.

Problems are modeled as CSPs (X, D, C)

Some variables to represent objects

$(X = \{X_1, \ldots, X_n\})$

Domains over which variables can range

$(D = D_1 \times \ldots \times D_n)$

Some constraints to set relation between objects

$$C_1 : X \leq Y * 3$$
$$C_2 : Z \neq X - Y$$
$$\ldots$$

CSP solving

A solution

- Given a search space $\mathcal{S} = D_{x_1} \times \ldots \times D_{x_n}$
- an assignment s is a solution if :
  - $s \in \mathcal{S}$ and
  - $\forall c \in \mathcal{C}, s \in c$

Solving a CSP can be :

- compute whether the CSP has a solution (satisfiability)
- find A solution
- find ALL solutions
- find optimal solutions (global optimum)
- find A good solution (local optimum)

Outline

First approach

Complete Methods

- Search space : $D_{x_1} \times \cdots \times D_{x_n}$
- Enumerate all assignments

Backtracking

- search (backtrack)
- Select variables
- Split / enumeration

Constraint propagation : reducing domains

Generally :

- reduce domains using constraint and domains
- → reduce the search space

Generic domain reduction :

- given a constraint C over $x_1, \ldots, x_n$ with domains $D_1, \ldots, D_n$
- select a variable x_i reduce its domain
- delete from D_i all values for x_i that do not participate in a solution of C

Constraint propagation

- constraint propagation mechanism : repeatedly reduce domains
- replace a CSP by a CSP which is :
  - equivalent (same set of solutions)
  - “smaller” (domains are reduced)

Second Approach

Incomplete methods

- heuristics algorithms
- Metaheuristics

two families

- Local search
  - Simulated annealing [Kirkpatrick et al, 1983]
  - Tabu Search [Glover, 1986]
  - ...
- Evolutionary Algorithms
  - Genetic Algorithms [Holland, 1975]
  - Genetic programming [Koza, 1992]
  - ...

Incomplete methods

Definitions

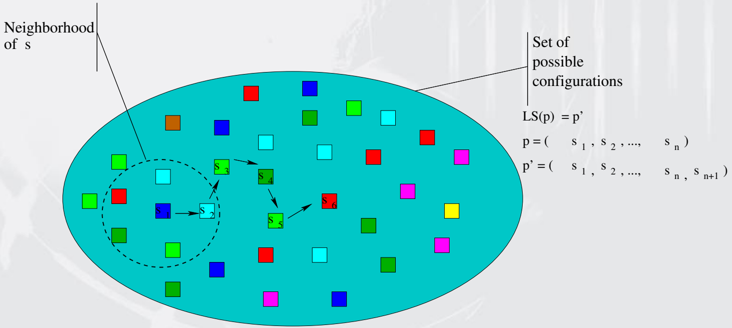
- Explore a $D_1 \times D_2 \times \cdots \times D_n$ search space
- Move from neighbor to neighbor (resp. generation to generation) thanks to an evaluation function
  - Intensification
  - Diversification

Properties :

- focus on some “promising” parts of the search space
- does not answer to unsat. problems
- no guaranteed
- “fast” to find a “good” solution

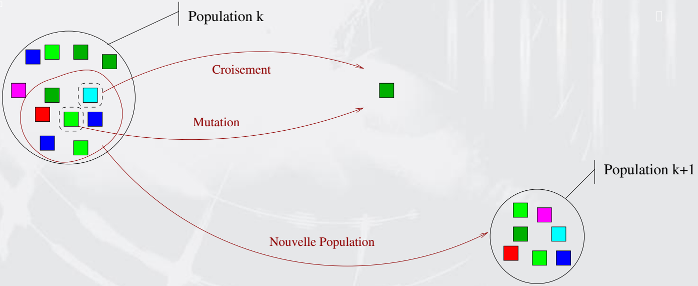
Local search

- Search space : set of possible configurations
- Tools : neighborhood and evaluation function



Genetic algorithms

- Search space : set of possible configurations
- Tools : population, crossing,mutations, and evaluation function



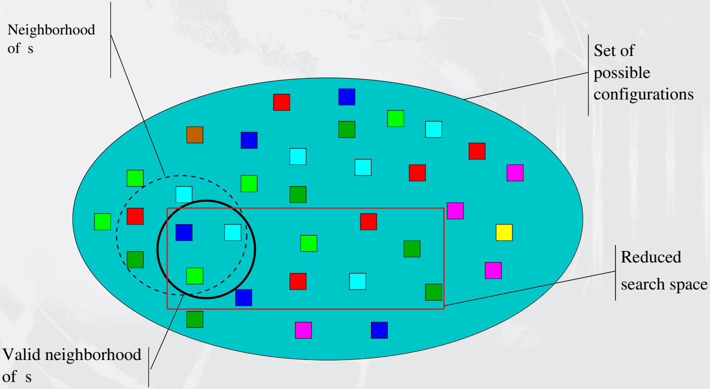
Hybridization : getting the best of the both

- Efficiency : faster complete solver
- Quality : better solutions (better optimum)
- Generally :
  - Ad-hoc systems (designed from scratch)
  - Dedicated to a class of problems
  - Master-slave approaches (LS for CP, CP for LS)

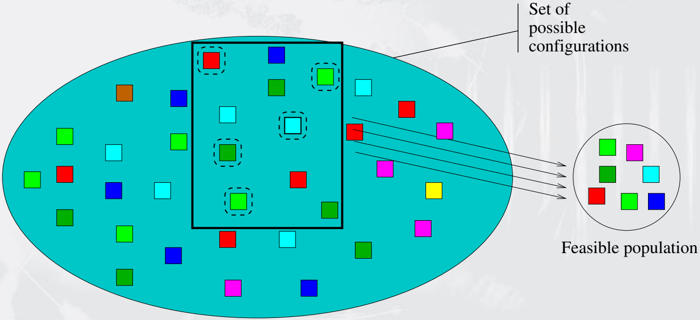
Hybridization : Overview

- **complete and incomplete methods combinations**
  - collaborative combinations
  - integrative combinaisons
    - Incorporate a complete method in metaheuristics
    - Incorporate a metaheuristic in complete methods

CP for LS


Neighborhood
of s
Set of
possible
configurations
Reduced
search space
Valid neighborhood
of s

CP for GA



Hybridization : getting the best of the both

- Idea :
  - fine grain hybridization
  - finer strategies
  - every technique at the same level
  - one algorithm squeleton
  - easier to modify, extend, compare, ...

Constraint propagation framework

Can be seen as a fixed point of application of reduction functions

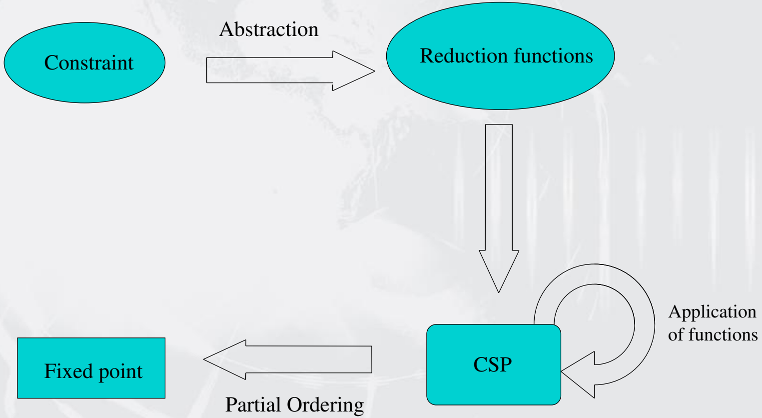
- reduction function to reduce domains or constraints
- can be seen as an abstraction of the constraints by reduction functions

Chaotic iteration

- Compute a limit of a set of functions [Cousot and Cousot 77]
- monotonic and inflationary functions in a generic algorithm to achieve consistency [Apt 97]

Abstract model K.R. Apt [CP99]

For propagation (based on chaotic iterations)



Partial ordering and functions

Partial Ordering

Given a CSP $(\mathcal{X}, \mathcal{D}, \mathcal{C})$

- $\mathcal{P}(\mathcal{D})$: all possible subset from $\mathcal{D}$
- $\sqsubseteq$: subset relation $\supseteq$

$\Longrightarrow (\mathcal{P}(\mathcal{D}), \sqsubseteq)$ is a partial ordering

Functions

Given a set F of functions on $\mathcal{D}$, every $f \in F$ is :

- inflationary : $x \sqsubseteq f(x)$
- monotonic : $x \sqsubseteq y$ implies $f(x) \sqsubseteq f(y)$
- idempotent : $f(f(x)) = f(x)$

$\Longrightarrow$ Every sequence of elements $d_0 \sqsubseteq d_1 \sqsubseteq ...$ with $d_j = f_{i_j}(d_{j-1})$ stabilizes to a fix point.

A Generic Algorithm to reach fixpoint

for **constraint propagation** : ordering on size of domains

work on a CSP

$F = \{$ set of propagation functions$\}$

$X =$ initial CSP

$G = F$

While $G \neq \emptyset$

  choose $g \in G$

  $G = G - \{g\}$

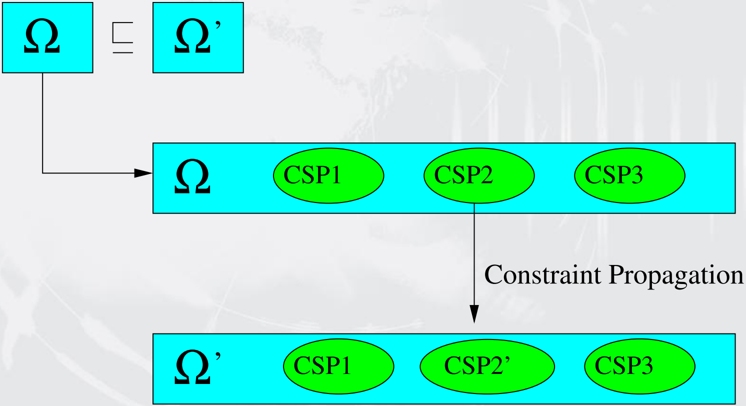
  $G = G \cup update(G, g, X)$

  $X = g(X)$

EndWhile

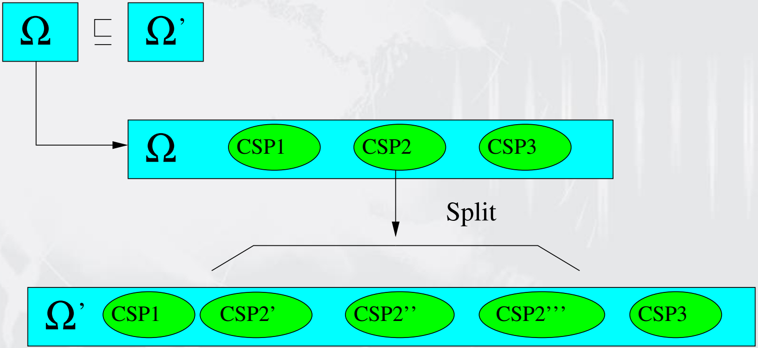
Theoretical model for CSP solving

Reduction : by constraint propagation :



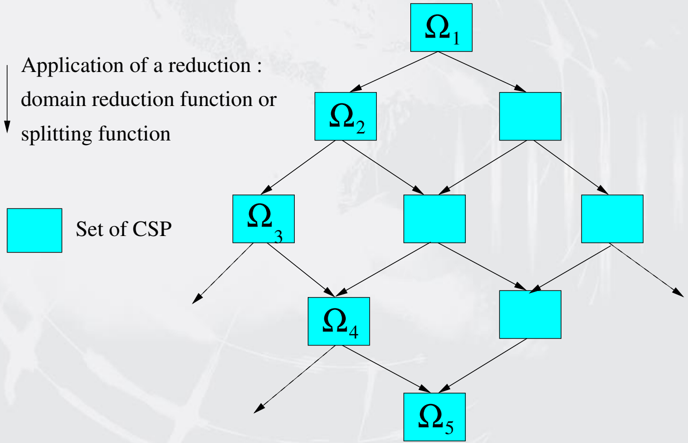
Theoretical model for CSP solving

Reduction by domain splitting :



Theoretical model for CSP solving

Partial ordering :



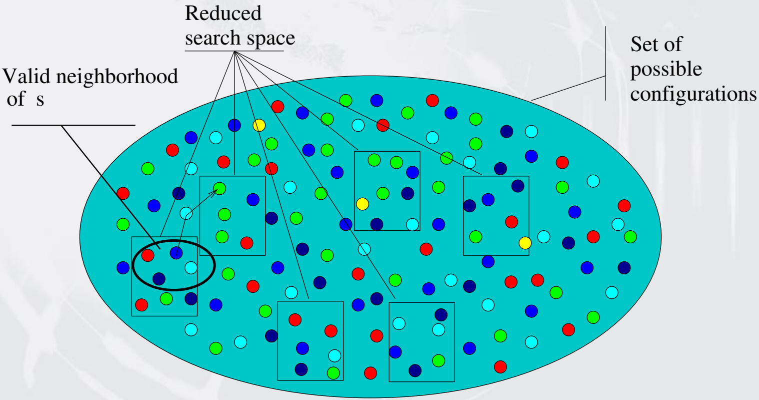
➡ Terminates with a fixed point : set of solutions or inconsistent CSP

Motivation

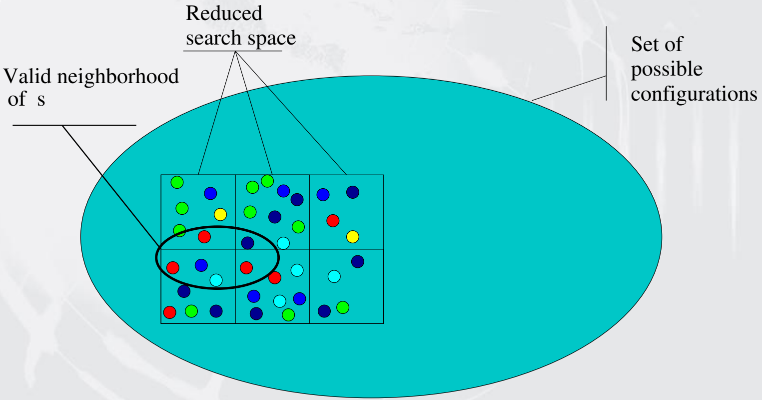
- But : GA + CP + LS ?

$\Longrightarrow$ Notion of sample $\neq$ generation
$\Longrightarrow$ Need to consider sample and individual at the same level : CSP level

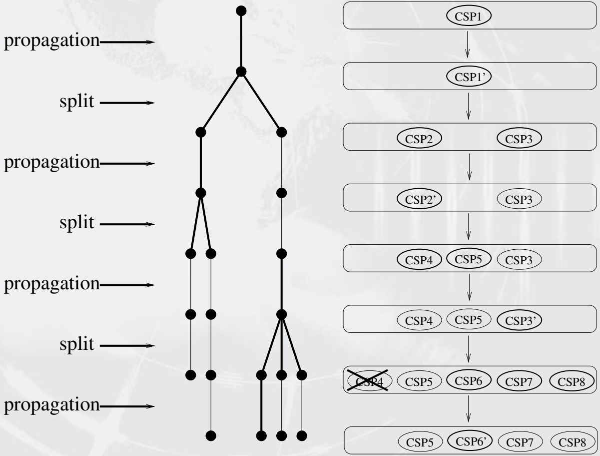
Motivation


Reduced
search space
Valid neighborhood
of s
Set of
possible
configurations

Motivation


Reduced
search space
Valid neighborhood
of s
Set of
possible
configurations

Search Tree


propagation
split
propagation
split
propagation
split
propagation
CSP1
CSP1'
CSP2
CSP3
CSP2'
CSP3
CSP4
CSP5
CSP3
CSP4
CSP5
CSP3'
CSP4
CSP5
CSP6
CSP7
CSP8
CSP5
CSP6'
CSP7
CSP8

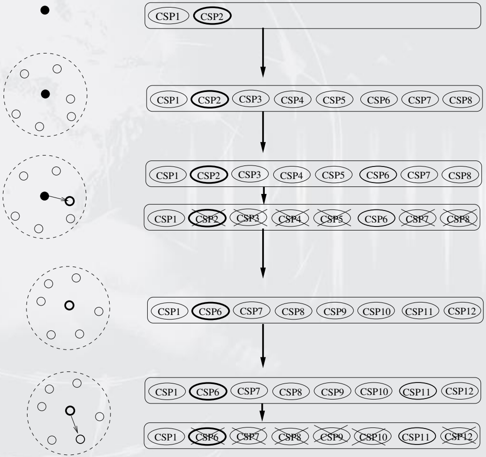
Example : Local Search

Generate new samples

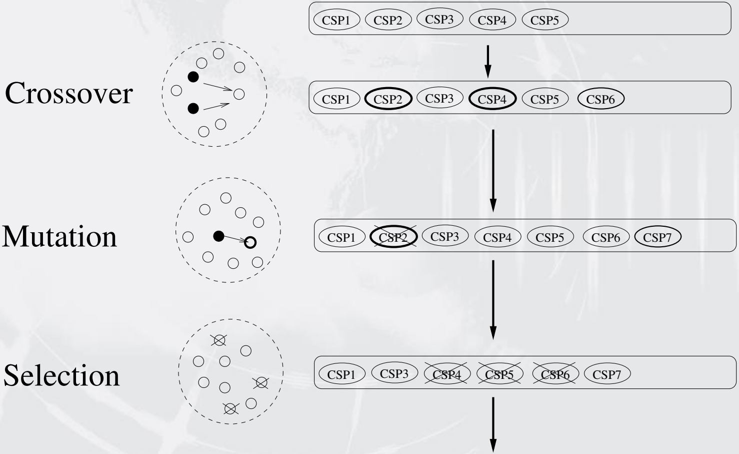
Chose a sample

Generate new samples

Chose a sample



Example : Genetic algorithms



Break up methods

Basic components

- reducing is a search component
- splitting is a search component
- generating a neighborhood is a search component
- moving to sample is a search component

Basic behaviours

- splitting and generating a neighborhood
- reducing, selecting and moving to sample

Basic fonctions

Sampling $\mathcal{S}$:

$$\mathcal{P}(\langle X, C, \mathcal{P}(D_1) \times \cdots \times \mathcal{P}(D_n) \rangle)$$

$$\rightarrow \mathcal{P}(\langle X, C, \mathcal{P}(D_1) \times \cdots \times \mathcal{P}(D_n) \rangle)$$

$$\{\phi_1, \ldots, \phi_n\} \mapsto \{\phi_1, \ldots, \phi_n, \phi_{n+1}\}$$

s.t. $\exists \phi_i$ with $\phi_i \sqsubseteq \phi_{n+1}$

Basic fonctions

Reducing $\mathcal{R}$:

$$\mathcal{P}(\langle X, C, \mathcal{P}(D_1) \times \cdots \times \mathcal{P}(D_n)\rangle) \rightarrow \mathcal{P}(\langle X, C, \mathcal{P}(D_1) \times \cdots \times \mathcal{P}(D_n)\rangle)$$

$$\{\phi_1, \ldots, \phi_i, \ldots, \phi_n\} \mapsto \{\phi_1, \ldots, \phi'_i, \ldots, \phi_n\}$$

Where $\phi'_i = \emptyset$ or $\phi = \langle X, C, D_i\rangle$ and $\phi' = \langle X, C, D'_i\rangle$ s.t. $D'_i \subseteq D_i$.

Reduction functions (1)

Domain reduction (DR)

$\{\phi_1, \ldots, \phi_i, \ldots, \phi_n\} \rightarrow^{DR} \{\phi_1, \ldots, \phi'_i, \ldots, \phi_n\}$
Where $DR = \mathcal{R}^m$ with $m > 0$

Split (SP)

$\{\phi_1, \ldots, \phi_i, \ldots, \phi_n\} \rightarrow^{SP} \{\phi_1, \ldots, \phi_i^1, \ldots, \phi_i^m, \ldots, \phi_n\}$
Where $SP = \mathcal{S}^m\mathcal{R}$

Local Search (LS)

$\{\phi_1, \ldots, \phi_i, \ldots, \phi_n\} \rightarrow^{LS} \{\phi_1, \ldots, \phi'_i, \ldots, \phi_n\}$
Where $LS = \mathcal{S}^m\mathcal{R}^{m-1}$

Reduction functions (2) Genetic Algorithms

Crossover

$\{\phi_1, \ldots, \phi_n\} \rightarrow^{CR} \{\phi_1, \ldots, \phi_n, \phi_{n+1}\}$

Where $CR = \mathcal{S}$

Mutation

$\{\phi_1, \ldots, \phi_i, \ldots, \phi_n\} \rightarrow^{MU} \{\phi_1, \ldots, \phi_i', \ldots, \phi_n\}$

Where $MU = \mathcal{SR}$

Selection

$\{\phi_1, \ldots, \phi_n\} \rightarrow^{SE} \{\phi_1, \ldots, \phi_n'\}$

Where $SE = \mathcal{R}^m$.

Examples

Memetic Algorithms

nbc samples are generated then some are used in a local search process and finaly the population is reduced.

$$((S^{nbc}(S^{p}R^{p})^{nbmut})R^{taillepop})^{n}$$

- S^{nbc}: Generate Samples (crossover)
- $(S^{p}R^{p})^{nbmut}$: local search
- $R^{taillepop}$: Restrict population
- n: n times

Examples

Hybrid Algorithms

Local search, domains reduction, genetic algorithm and split are executed sequentially.

$$(LS^{n}\ DR^{*}\ GA^{m}\ SP^{l})^{*}$$

- LS^{n}: Local Search
- DR^{*}: Domain reduction
- GA^{m}: Genetic Algorithm
- SP^{l}: Split
- $(\ldots)^{*}$: Until it reached a solution/all solutions/inconsistency

Properties

- a strategy is a sequence of words $\in \{DR, SP, LS, SE, CR, MU\}$
- is it finite sequences ?
- need conditions to avoid loops

Conclusion

- A generic model for hybridizing complete (CP) and incomplete (LS and GA) methods
- Implementation of modules working on the same structure
- Complementarity of methods : hybridization

Modèle d'hybridation pour la résolution de CSP

ROADEF 2008, Clermont-Ferrand, France

Tony LAMBERT

LIFO, Rue Léonard de Vinci 45067 ORLEANS

Mercredi 27 février 2008