Meta-heuristics for Combinatorial Optimization

Jin-Kao Hao

LERIA
University of Angers
2, Boulevard Lavoisier
F- 49045 Angers Cedex 01

phone: (+33) 2 41 73 50 76
email: hao@info.univ-angers.fr
web: www.info.univ-angers.fr/pub/hao

PLAN

1. **Combinatorial optimization**
2. **Review of resolution methods**
3. **Local search methods**
4. **Evolutionary methods**
5. **Hybrid methods**
6. **Application examples**
7. **Conclusions**

Part One

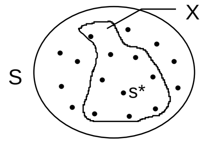
Review of Resolution Methods for Combinatorial Optimization

Combinartorial Optimization

Minimization

Given a couple (S,f) where
- S a finite set of solutions or configurations (search space)
- f: $S \rightarrow R$ a cost function (or objective)

find $s^* \in X \subseteq S$ such that $f(s^*) \leq f(s)$ for each element $s \in X$ (feasible space)

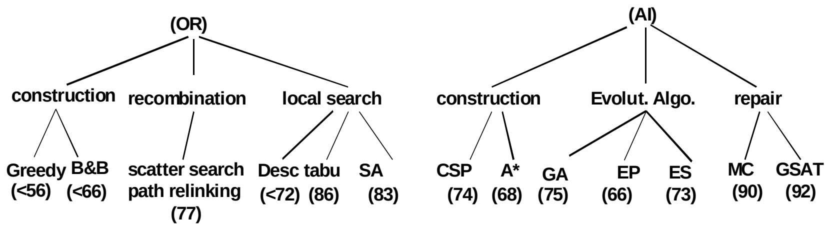


Example: Traveling Salesman Problem TSP

Remarks :
- For maximization, one replaces "$f(s^*) \leq f(s)$" by "$f(s^*) \geq f(s)$"
- In practical situations, neither S or f is necessarily given (modeling)
- Most of important optimization problems are NP-hard.

J.K. Hao, Université d'Angers, France

Resolution Methods



Four main approaches :

1. **Construction**: step-by-step instantiation of variables according to a static or dynamic order (branch & bound, CSP, greedy methods...)
2. **Local search**: iterative transition of complete configurations by local changes (descent, simulated annealing, tabu search, min-conflicts...)
3. **Evolution**: evolution of a population of solutions by « genetic » operators (selection, crossover, mutation), ex: genetic algorithms, scatter search...
4. **Hybrid**: combination of different approaches (evolution + LR, evolution + construction...)

J.K. Hao, Université d'Angers, France

Resolution Methods

Construction approach

- step-by-step instantiation of variables according to a static or dynamic order.
- if complete (or exact), then exponential complexity
- examples: branch & bound, CSP, greedy algorithms...

Local search or neighborhood search

- iterative repairs of complete configurations by local changes
- only sub-optimal solutions
- examples: descent, simulated annealing, tabu search, min-conflicts...

Remark : It's possible to integrate local search within an exhaustive search

Evolution approach

- evolution of a population of solutions by « genetic » operators (selection, crossover, mutation)
- only sub-optimal solutions
- examples: genetic algorithms, scatter search...

J.K. Hao, Université d'Angers, France

Local Search

Basic Notions

- Neighborhood function N: $S \rightarrow 2^S$, for each $s \in S$, $N(s) \subset S$
- $s \in S$ is a local optimum w.r.t. N if $s' \in N(s)$, $f(s) \leq f(s')$ (minimization)

General procedure

Step 1 (initialization)

a) choose an initial solution $s \in S$
b) $s^* \leftarrow s$ (i.e. record the best solution found so far)

step 2 (choice & termination)

a) choose $s' \in N(s)$
b) $s \leftarrow s'$ (i.e. replace s by s')
c) terminate et return the best solution found if the stop test is verified

step 3 (update)

a) $s^* \leftarrow s$ if $f(s) < f(s^*)$
b) go to 2

Remark: Meta-heuristics are different according to the strategy used at step 2

Resolution Methods

Pure descent

step 2 (choice & termination)

a) choose $s' \in N(s)$ such that $f(s') < f(s)$
b) $s \leftarrow s'$ (i.e. replace s by s')
c) terminate if $\forall s' \in N(s)$, $f(s') > f(s)$

Remark:

Decisions to be taken

- First improvement or best improvement
- how are neighbors evaluated rapidly at each iteration (special data structures)

Local optimum & remedy

- stop once a local optimum is found
- random re-run
- acceptation of non-improvement neighbors

Local Search

Simulated annealing

Step 2 (choice & termination)

1. choose *randomly* s' $\in$ N(s)
2. if f(s') $\leq$ f(s), then accept s, otherwise accept s with probability p(Δf,T)
3. terminate if stop condition is verified (max nb of iterations…)

Remark:

- **Decision to be taken**
  1. how to determine the probability p(Δf,T)
  2. how to evaluate rapidly the neighbors (move values) at each iteration

- a search method based (partially) on randomness (exploration > exploitation)

Local Search

Tabu search (TS)

step 2 (choice & termination)

1. choose the *best* neighbor s' $\in$ N(s) i.e. $\forall$ s" $\in$ N(s), f(s') $\leq$ f(s") (*tabu list* to prevent the search from *cycling*)
2. s $\leftarrow$ s', even if f(s') > f(s)
3. terminate if stop condition is verified

Remark:

- **decision to be taken**
  - what to record in tabu list
  - how to determine the length (tabu tenure) of tabu list (dynamic or static)
  - how to evaluate rapidly the neighbors (move values) at each iteration

- randomness is not essential: exploitation > exploration

Local Search

Other local search methods

Variable Neighborhood Search (VNS): a set of (nested) neighborhood relations are alternatively used during the search process

GRASP: a hybrid method combining greedy construction and local search

Local Search (summary)

Descent:

choose an improving neighbor $s' \in N(s)$ i.e. $f(s') < f(s)$
fast but stops at the first local optimum found

Simulated annealing:

choose randomly $s' \in N(s)$; if $f(s') \leq f(s)$ then accept s'
otherwise accept s' probability $p(\Delta f,T)$

Tabu search:

choose the best neighbor $s' \in N(s)$, accept s' even if $f(s') > f(s)$ (*tabu list* to prevent the search from cycling)

➢ Simualted annealing and tabu search don't stop at the first local optimum encountered

Local Search: performance

Theory

convergence proof in certain cases (SA, probabilistic TS…) under strong conditions

Practice

- ➢ strong experimental results for numerous hard problems
- ➢ *adaptation* is necessary:
  - • problem encoding (configuration and search space)
  - • neighborhood relations
  - • constraint handling
  - • data structures

Improvement

- ➢ hybrid with genetic algorithms
- ➢ hybrid with construction approaches

Evolutionary Approach

Basic concepts

- ➢ evolution of a set of configurations (notion of *population*)
- ➢ evolution operators (selection, recombination and mutation)

General procedure

step 1 : (*initialization*)
choose a set of initial configurations (population)

step 2 : (*evolution*)
application of recombination and mutation operators

step 3 : (*update*)
re-organization of the population (e.g. elimination of bad configurations from the population)

Remarks:

- ➢ different schools: genetic algorithms, evolutionary strategies, evolutionary programming
- ➢a general and powerful framework for algorithm design

J.K. Hao, Université d'Angers, France

Simple Genetic Algorithms (Holland 75)

Main features:

- universal problem representation based on binary encoding (binary strings)
- random genetic operators (mutaion and crossover)

Crossover: exchange of sub-strings between two indivuduals (monopoint, bi-points, uniforme...)

parents

1 1 0 1 | 0 1
X
0 1 0 0 | 1 0

children

1 1 0 1 1 0
0 1 0 0 0 1

Mutation: random modification of bit values of a new configuration

1 1 0 1 *1* 0 → 1 1 0 1 *0* 0

Remarks:

Therory of schemata (building blocks):

- the number of short and good schemas (building blocks) increases with the search,
- the performance of a configuration is an indicator of the performance of all the schemas represented by this particular schema (implicit //)

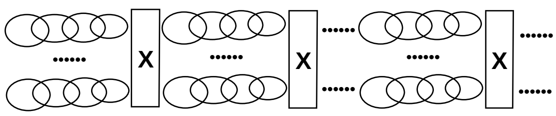
Decision to be taken:

- how to define the *fitness* function
- how to find a compromis between exploitation and exploration

Evolutionary Approach

In practice

- **Specialization:**
  - specialized encoding *adapted* to each problem (e.g. permutation for TSP)
  - specialized evolution operators based on the specialized encoding
- **Hybrid:**
  - with construction approaches
  - with local search



Evolutionary Approach: Performance

Theory

- convergence proof for some cases under strong conditions

Practice

- weak results for combinatorial optimization with simple GA (blind mutation and crossover)
- competitive results with *specialized* GA
  - problem specific encoding
  - problem specific evolution operators integrating problem knowledge
- very competitive results with hybrid GA
  - hybrid with construction methods (eg, greedy)
  - hybrid with local search: *GLS - Genetic Local Search*

Remark:

- adaptation is indispensable

Hybrid Genetic Algorithms

Basic idea

combine 2 complementary methods: *global* search and *local* search

General procedure

step 1 (initialization)

a) generate a population of configurations P
b) **apply a local search to each configurations of the population P**

step 2 (evolution and termination)

a) choose **p1** and **p2** in P
b) generate a configuration e by a recombination of **p1** and **p2**
c) **improve e by local search**
d) Insert the improved **e** in the population
e) terminate and return the best solution found when stop condition is verified

step 3 (*update*)

a) re-organization de la population (eg, elimination of bad configurations from the population)

Hybrid Genetic Algorithms

Two hybrid schemas

- GA + descent ("*Memetic algorithms*"):
  the configurations in the population are local optima

- GA + tabu search (simulated annealing):
  the configurations of the population are improved for a fixed number of iterations before a crossover

Design principles

- Local search operator should be efficient
- Crossover should be problem specific

Adaptation of meta-heuristics

Problem solving with meta-heuristics

- problem modeling
- choice of a meta-heuristic according to
  - the solution quality required
  - the availability of problem knowledge
  - the know-how...
- adaptation of the chosen meta-heuristic to the problem
  - configuration (search space)
  - neighborhood and evaluation function
  - search operators and constraint handling
  - data structures...

Performance evaluation (*benchmarking* whenever possible)

- the quality of the best solution found
- search profile (time v.s. quality plot)
- efficiency, *i.e.* efforts (computing time, number of iterations) necessary to reach the best solutions
- robustness

Other Related Important Issues

Fundamental issues

- Characterizing the search space
  - density of states
  - distribution of local optima
  - metrics of the difficulty of search instance...

- Studying the behaviors of meta-heuristic algorithms
  - landscapes of search process
  - relations « heuristics v.s. instances »
  - experimental comparisons...

Meta-heuristics based general purpose solvers

- language-based approaches
- library-based approaches
- primitive-based approaches

J.K. Hao, Université d'Angers, France

Conclusions

Strong points
- general and applicable to a large class of problems
- possibility of time-quality compromise
- preferred application domains: large and strongly constrained combinatorial optimization problems

« Weak points »
- only local optimum
- adaptation indispensable
- difficult to predict the performance (quality and time)

Performance
- **theory**: convergence proof in certain cases under strong conditions
- **practice**: depends on each adaptation (problem encoding, integration of problem knowledge, constraint handling, data structures…)

Perspectives
- General meta-heuristics based problem solvers will be available
- Cooperation and combination between meta-heuristics and other resolution methods
- More and more applications will be solved

J.K. Hao, Université d'Angers, France

Part Two

Application Examples

Application examples

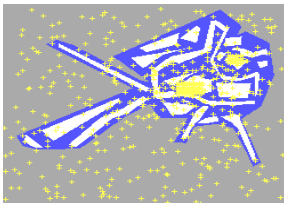
Applications

- antenna positioning in cellular phone networks (Esprit 4)
- frequency assignment in cellular phone networks (France Telecom)
- photograph scheduling of satellite SPOT5 (Application CNES)
- Migration and evolution of telecom equipment (Bouygues Telecom)
- Sports league scheduling
- Progressive party problem

NP-hard problems

- graph coloring and T-coloring
- multidimensional Knapsack
- constraint satisfaction CSP and MCSP
- SAT and Max-SAT

Antenna Positioning in Mobile Radio Networks



Urban network: 50 km x 46 km
568 potential sites
56792 reception test points (RTP)
17393 Service test points (STP)
6652 Traffic test point (TTP) for 2988.08 'erlang'

Working area

- o a set S of service test points (STP, **blue** points): quality threshold of radio signal (-82 dBm→2W 'incar',-90 dBm→8W 'outdoor',...),
- o a set $T \subset S$ of traffic test points (TTP, white points): traffic estimation in erlang,
- o a set of candidate sites for positioning antennas

Radio wave propagation model

- o for each site, the radio signal received by each other STP

Antennas

- o different types: omni-directional , large and small directional
- o parameter: power (26 à 55 dBm), azimut (0° à 359°), tilt (-15° à 0°)
- o number of transmitteurs (TRX): 1 to 7 according to the traffic to serve

Antenna Positioning in Mobile Radio Networks

Determine

- ➢ a subset of sites among the candidate sites,
- ➢ for each selected site, the number and the type of antennas,
- ➢ for each antenna, the value of each of the three parameters (power, tilt et azimut).

Imperative constraints

- ➢ *Cover*: all the STP must be covered by at least one antenna,
- ➢ *One component cell*: the STP served by an antenna form a single connected component (cell),
- ➢ *Hand-over*: each cell must have points covered by neighboring cells.

Objectives

- ➢ minimize the number of selected sites,
- ➢ minimize the interference level,
- ➢ maximize the traffic supported by the network (traffic hold),
- ➢ maximize yield of the transmitters of each antenna (traffic yield).

Remarks:

- o high number of combinations for the choice of feasible positioning,
- o high computational complexity to verify constraint satisfaction and objective evaluation
- o high demand of memory resource (200 to 500 MB for each instance)

Antenna Positioning in Mobile Radio Networks

Context

- Some studies for micro networks (indoor), almost nothing for large size networks (STORMS, Esprit 2)
- European ESPRIT 4 ARNO (Algorithms for Radio Network Optimisation)
- Heuristic methods:
  - simulated annealing, tabu search, genetic approach, neural network.
  - problem specific heuristics

A heuristic approach (Vasquez & Hao 00a)

- Constraint based pre-processing to reduce the search space
- Tabu search based optimization to search for feasible solutions
- Post-optimization by local refinement of parameters

Remarks:

- Hard even for finding feasible solution,
- A single heuristic or meta-heuristic is not sufficient to tackle the problem.

Antenna Positioning: Experimental Results

Networks	Potential Sites	Min. sites	Min. Cells	STP
Green Field				
Highway	250	25	75	29954
Rural	320	22	65	72295
Small urban area	568	24	70	17393
High traffic urban area	244	21	61	48512
Extension				
Highway	250	25	75	29954
Rural	320	47	140	80854
Small urban area	568	63	189	42492
High traffic urban area	244	113	337	48512

Antenna Positioning: Experimental Results

Small urban network: 50 km x 46 km

- 568 potential sites
- 56792 reception test points (RTP)
- 17393 service test points (STP)
- 6652 traffic test points (TTP) for 2988.08 'erlang'

Constraints:

- o constraints to be satisfied: cover, one component and hand-over.

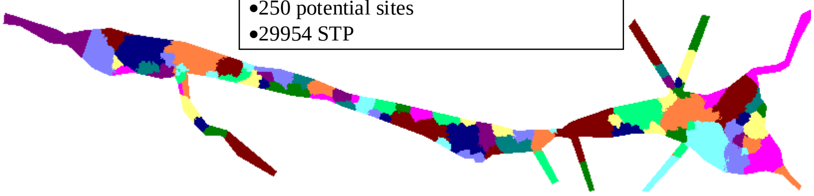
Objectives:

- o number of sites/antennas: 34 / 52 (35 small directional, 17 large directional)
- o potential interference: very low
- o traffic hold: 85%
- o traffic yield: very good

Remark: It is difficult to find feasible solution without pre-processing.

Antenna Positioning: Experimental Results

A highway network: 39km x 168km
•250 potential sites
•29954 STP



Constraints:

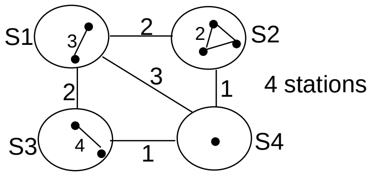
- ➢ all constraints satisfied: cover, local connectivity and hand-over.

Objectives:

- ➢ number of sites / antennas: 58 / 103
- ➢ 1 omni-directional, 67 small directional, 35 large directional
- ➢ interference: very low
- ➢ traffic hold: high
- ➢ traffic yield: very high

Frequency Assignment in Mobile Radio Networks

The problem



Given

1. n stations {S1, S2...Sn}
2. traffics bi (i ∈ {1...n}), i.e. the number of frequencies required by each station
3. interference constraint defined by a matrix M[n,n]:
   - "co-station": $| f_{i,g} – f_{i,h}| \geq M[i,i], \forall\ i \in \{1...n\}, \forall\ g,h \in \{1..Ti\}, g \neq h$
   - "adjacent stations":
     $| f_{i,g} – f_{j,h}| \geq M[i,j], \forall i,j \in \{1...n\},\ i \neq j,\ \forall g \in \{1..bi\}, \forall h \in \{1..bj\}$

to find a frequency assignment

such that the interference is minimized with k fixed frequencies (interference is measured in terms of violated constraints)

Remark: this is a generalized graph coloring problem (set T-coloring)

Frequency Assignment in Mobile Networks

Resolution methods

- Simulated annealing (Duque-Anton et al. 93, CNET 95)
- Neural network (Kunz 91)
- Genetic algorithms (Crompton et al. 94)
- Graph coloring algorithm (Gamst 86, CNET 95)

Tabu and genetic algorithms

(Dorne & Hao 95, Hao & Dorne 96, Renaud & Camanida 97, Hao et al. 98)

- constraint handling
- specialized crossover

Frequency Assignment in Mobile Networks

A tabu algorithm

- *configuration*: any complete frequency assignment respecting the traffic and co-station constraints

S1		S2			S3		S4
f11	f12	f21	f22	f23	f31	f32	f41

- *Neighborhood N: S* $\rightarrow 2^S$ **:** s are s' neighbors only if they are different by the value of a conflicting station
- *tabu list*: recorded attribute = <station, old_freq_value>
- *tabu tenure*: a randomized linear function of the size of the neighborhood
- *incremental evaluation of configurations*: move values using special data structures (a matrix Δ of w x k)
- *aspiration*: accept any move leading to a configuration better than the best found so far

J.K. Hao, Université d'Angers, France

Affectation de fréquences dans les réseaux radio-mobiles

Un algorithme tabou

étape 1 (initialisation)

- choisir une solution initiale $s \in S$
- mémoriser la meilleure solution trouvée $s^* \leftarrow s$
- initialiser les structures de données (liste tabou, matrice Δ...)

étape 2 (choix et terminaison)

- choisir un des meilleurs voisins non tabou $s' \in N(s)$ tq $\forall\, s'' \in N(s)$, $f(s') < f(s'')$
- $s \leftarrow s'$ (même si s' est moins performant que s)
- terminer si max_itér est effectué (ou si s n'est plus améliorée pendant max_itér)

étape 3 (mise à jour)

- $s^* \leftarrow s$ si $f(s) < f(s^*)$
- rendre le dernier mouvement tabou pendant k itérations
- mettre à jour d'autres structures de données (matrice Δ...)
- aller à l'étape 2

J.K. Hao, Université d'Angers, France

Frequency Assignment in Mobile Networks

Test data (artificial and real data) (from France Telecom R&D)

- nb of frequencies per station: 1 to 4
- separation distance for co-station frequencies: 2 to 4
- separation distance for adjacent station: 1 to 3
- of large size, up to
  - 1 000 integer variables
  - 64 values per variable
  - 35 000 constraints

Results

- tabu and genetic algorithms dominate largely other approaches (greedy, CP…)
- Random crossover play marginal role.

Photograph Scheduling of an Earth Observation Satellite

The problem

Given:

- a set P of candidate photographs, mono or stereo, to be scheduled on the next day;
- a “profit” associated with each candidate photo (aggregation of certain criteria);
- a “size” associated with each candidate photo (memory required to record the photo);
- different possibilities for realizing each photo:
  - three for each mono photo (any of the three cameras: front, middle and rear),
  - one single possibility for each stereo photo (front and rear simultaneously)
- a set of imperative binary and ternary constraints;
- recording capacity (knapsack) constraint.

To determine a subset $P' \subseteq P$ such that:

- the total profit of the photos in P’ is maximized while all the constraints are satisfied.

Remark:

can be formulated as a logic-constrained 0/1 knapsack problem

Photograph Scheduling of an Earth Observation Satellite

Resolution method

- exact algorithm (Verfaillie et al. 96)
- greedy algorithms (Agnèse et al. 95)
- tabu search algorithm (Agnèse et al. 95)

A new tabu search algorithm (Vasquez & Hao 00b)

- a logic-constrained knapsack problem
- a tabu algorithm integrating:
  - a relaxation technique for handling the memory constraint
  - add-drop-repair neighborhood
  - a fast neighborhood evaluation technique
  - a dynamic technique for dynamic tabu list management

Tight upper bounds (Vasquez & Hao 00c)

- based on graph partitioning, dynamic programming and tabu search

Photograph Scheduling of an Earth Observation Satellite

Benchmarks (defined by the French Space Agency CNES, available on the web)

- two different types
  without memory constraint (known optima)
  with memory constraint (unknown optima except for one instance)
- of large size, up to
  900 integer variables
  17 000 constraints (binary, ternary and knapsack)

Experimental results

- for instances without memory constraint: optimal in about two minutes solution
- for instances with memory constraint:
  - several seconds to reach the best known results
  - better results in several minutes

Progressive Party Problem

The problem

Given:

- 29 crews, each having a fixed occupation size;
- 13 host boat, each having a receiving capacity;
- T (fixed or not fixed) time periods, during which the party is organized;

with the following constraints:

- each guest crew moves to a different host boat at each time period;
- two crews meet each other at most once;
- for each time period, the capacities of the host boats must be respected

- to find, for a fixed number T of time periods, an assignment of the 29 crews to the 13 boats satisfying all the constraints

- to find assignment maximizing the number T of time periods

Remarks:

- the initial problem with T = 6, more difficult when T increases,
- no solution for T > 8 until 1996, T= 10 an upper bounds
- heterogeneous, non-binary constraints

Progressive Party Problem

Resolution methods

- Integer linear programming: failure for T<6 (Brailsford et al. 96)
- constraint programming 1: T = 6 and 7 in 20-30 mn, failure for T ≥ 8 (Smith et la 96)
- constraint programming 2 : T < 7 in several sec., T = 8 in several hours, failure for T>9 (MIC 97)

Heuristic approach with local search (Galinier & Hao 98)

- a CSP formulation of the problem:
  * var. = a couple (crew, period) and dom. = the set of the host boats D = {1...13}
  * constraints: nary
- two neighborhoods:
  * N1: change the value (host boat) of a conflicting variable (crew, period)
  * N2: exchange the values of two 2 variables of the same period
    (at least one is in conflict)
- local search algorithms (descent, metropolis, tabu)

Progressive Party Problem

Results

- for T≤9, solutions in 5 seconds
- failure for T = 10 (one constraint violated)

Remarks:

- solutions up to T = 9 with both N1 and N2
- N2 more efficient (solutions until T= 8 with descent)
- no significant difference for meta-heuristics used (simulated annealing and tabu search)

- problem for T = 10 remains an OPEN problem

=> until T = 9, the problem is simple for local search

Sports League Scheduling Problem

The problem

Given:

• n teams (n even), n-1 weeks, n/2 periods,
• each team plays one game per week,
• each team plays no more two games per period,
• each game is played exactly one

To schedule the tournament satisfying all the constraints

	week 1	week 2	week 3	week 4	week 5	week 6	week 7
Period 1	**1 vs 2**	**1 vs 3**	**5 vs 8**	**4 vs 7**	**4 vs 8**	**2 vs 6**	**3 vs 5**
Period 2	**3 vs 4**	**2 vs 8**	**1 vs 4**	**6 vs 8**	**2 vs 5**	**1 vs 7**	**6 vs 7**
Period 3	**5 vs 6**	**4 vs 6**	**2 vs 7**	**1 vs 5**	**3 vs 7**	**3 vs 8**	**1 vs 8**
Period 4	**7 vs 8**	**5 vs 7**	**3 vs 6**	**2 vs 3**	**1 vs 6**	**4 vs 5**	**2 vs 4**

Example of valid scheduling involving 8 teams

Sports League Scheduling Problem

Resolution methods

- ILP with cardinality constraint (Cplex): $n \leq 12$ (McAloon et al. 97)
- CP with "difference" constraint (ILOG Solver): $n \leq 14$ (McAloon et al. 97)
- CP with new filtering (propagation) algorithms (ILOG Solver): $n \leq 24$ (Régin 98)
- CP with problem transformation and constraint propagation (ILOG Solver): $n \leq 40$ (Régin 99)

Tabu search (Hamiez & Hao 00)

• a CSP formulation of the problem CSP
• a graph based construction for initial solution
• a swap neighborhood
• dynamic tabu tenure

Sports League Scheduling Problem

Results

- solutions up to $n = 40$
- for $n \leq 20$, solutions in several seconds

Remarks:

- initial solution is important
- the way to handle constraints is important
- diversification indispensable

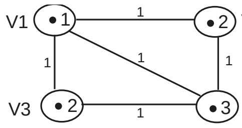
- open problem for $n > 40$

Graph Coloring

k-coloring (satisfaction)

- given a graph G=<V,E>, find a function c: V $\rightarrow$ {1,2...k} such that

$\forall$ (Vi,Vj)$\in$E, |c(Vi) - c(Vj)| > 0 (2 adjacent nodes are colored with different colors)



3-colorable

Coloring (optimization)

- determine the smallest k for which the graph is k-colorable (chromatic number)

Remarks:

1) NP-hard, many applications (frequency assignment, timetabling…), one of the three target problems of the "2nd DIMACS Implementation Challenge" (92-93)

2) k-coloring can be solved via optimization:

* a configuration = a complete assignment of the k colors to the nodes
* the cost function (to be minimized) = the number of edges with its two end-points colored the same color

2) coloring can be solved as a series of k-coloring problems(with decreasing k)

J.K. Hao, Université d'Angers, France

Graph T-coloring

T-coloring

- given a graph G=<V,E> and a set Tij of positive integers including the value 0 for each (Vi,Vj)$\in$E, find a function c: V $\rightarrow$ {1,2...k} such that:

$\forall$ (Vi,Vj)$\in$E, | c(Vi) - c(Vj) |$\notin$Tij

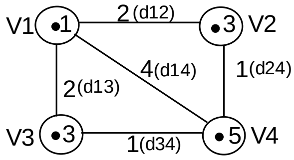
(the distance between the colors of 2 adj. nodes must be different from those of Tij)

Remarks:

- if Tij = {0,1...dij}, then $\forall$ (Vi,Vj)$\in$E, | c(Vi) - c(Vj) | $\geq$ dij

- "T-Span" of a T-coloring: the distance between the maximal and the minimal colors
- "T-Span" of a graph G: the smallest «T-span » of all the T-colorings de G

T-coloring



T-span = 4

J.K. Hao, Université d'Angers, France

Graph Coloring

Set T-coloring

With each V_i, one associates:

- an integer $b_i \geq 1$, the number of colors required by V_i and
- a set T_i of positive integers including the value 0 (forbidden values for adj nodes)

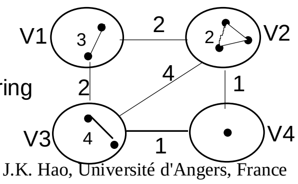
Color each node $V_i \in V$ with $b_i \geq 1$ colors with the following constraints:

- each pair of colors of a node V_i must have a distance different from those of T_i: $\forall V_i \in V$, $m, n \in \{1...b_i\}$, $m \neq n$, $| c(V_{im}) - c(V_{in}) | \notin T_i$,
- each pair of colors of 2 adjacent nodes V_i and V_j must have a distance different from those of T_{ij}: $\forall (V_i,V_j) \in E$, $m \in \{1...b_i\}$, $n \in \{1...b_j\}$, $|c(V_{im}) - c(V_{jn})| \notin T_{ij}$

Remarks:

- if $T_i = \{0,1...d_i\}$, then $\forall V_i \in V$, m,nÎ{1...bi}, $m \neq n$, $|c(V_{im}) - c(V_{in}) \geq d_i$
- frequency assignment is an application of set T-coloring

Set T-coloring



Graph Coloring

Resolution methods

- coloring
  - sequential construction: DSATUR (Brélaz 79), RLF (Recurs. largest First) (Leighton 79)
  - simulated annealing (Chams et al. 87, Johnson et al. 91), tabu (Hertz & de Werra 87, Dorne & Hao 98a)
  - hybrid (Fleurent & Ferland 94, Morgenstern 94, Costa et al. 95, Dorne & Hao 98b, Galinier & Hao 99)
  - scatter search (Hamiez & Hao 01)
- T-coloring
  - greedy dynamic ordering (Gamst 92)
  - tabu search and simulated annealing (Costa 93, Dorne & Hao 98a)
- set T-coloring
  - greedy dynamic ordering and tabu (Jiang 96, Dorne & Hao 98a)

Graph Coloring

Benchmarks

- ➢ coloring
  - • DIMACS benchmarks
  - • graphs containing up to 1000 nodes

- ➢T-coloring
  - • random graphs (available upon request)
    - + separation distance taken between 1 and 5
    - + number of colors per node taken between 1 and 5
  - • graphs containing up to 1 000 nodes (some 3 000 integer variables, 4 millions constraints) and 2 000 colors

Remark: no exact algorithm is able to color graphs of density 0.5 > 90 nodes

Graph Coloring

Results

- ➢ coloring
  - • algorithms based on tabu and hybrid dominate sequential algorithms
  - • Genetic local search hybrid improves on some best known results for large graphs

- ➢ (set) T-coloring
  - • local search heuristic algorithms dominate sequential algorithms

Remark:

➢ important to initialize a heuristic search with a sequential solution

Multidimensional Knapsack

The problem

$$\max c.x$$

subject to:

$$A.x \leq b \text{ and } x \in \{0,1\}^n$$

where $c \in N^n$, $A \in N^{mxn}$, $b \in N^m$

*Remark***:**

- many applications (cutting stock, cargo loading…),
- NP-hard, exact resolution limited to instances with $n < 90$ et $m < 5$.

Multidimensional Knapsack

Resolution method

- exact algorithm (B&B) (Shih 79)
- relaxation (Fréville & Plateau 93)
- Tabu (Glover & Kochenberger 97, Hanafi & Fréville 98) (very good results)
- genetic algorithms (Chu & Beasley 98)
- hybrid algorithm “simplex + tabu” (Vasquez & Hao 00d)

Test data

- benchmarks OR-LIB (n=500 and m=5 to 30)
- recent benchmarks (n=100 to 2500 and m = 15 to 100)

Results

- hybrid algorithm improves on all the best known results for the largest instances.

Conclusions

- Meta-heuristics are powerful tools for large scale combinatorial optimization problems
- Meta-heuristics offer possibilities to be combined with other resolution methods (both exact and heuristics)
- A good performance requires:
  - an appropriate formulation of the problem
  - “intelligent” adaptation of the chosen meta-heuristic:

    + integration of problem specific knowledge

    + efficient data structures