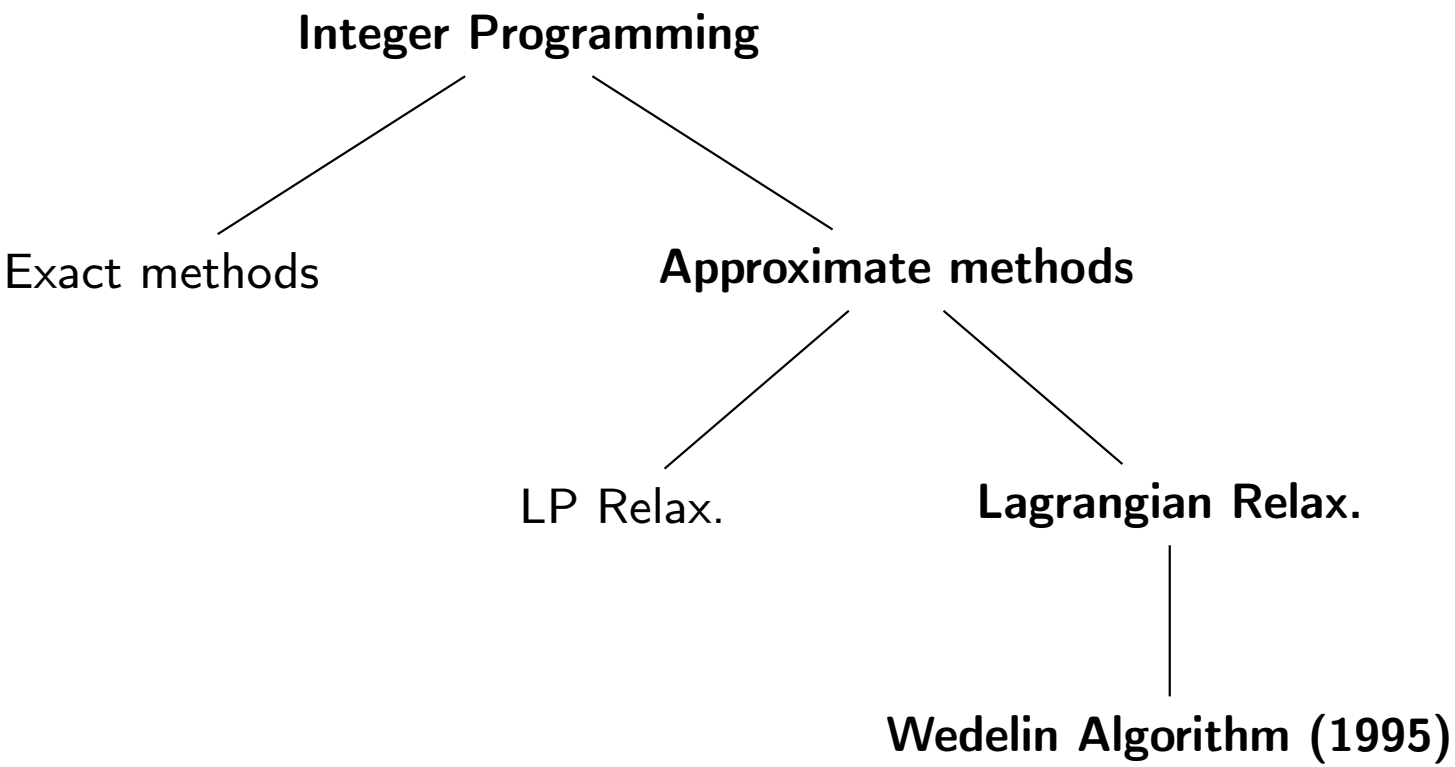
Improving Wedelin's Heuristic with Sensitivity Analysis for Set Partitioning Problem

Sara Maqrot [1], Simon De Givry [1], Marc Tchamitchian [2]
Gauthier Quesnel [1]

[1] UR 875, MIAT-INRA Toulouse France
[2] UR 767, Ecodeveloppement, INRA-Avignon France

July 2018

Outline


Integer Programming
Exact methods
Approximate methods
LP Relax.
Lagrangian Relax.
Wedelin Algorithm (1995)

Wedelin Algorithm

- The original algorithm [Wedelin, 1995] has been designed for IP of the following form :

$$\min_{x \in \{0,1\}^n} \{c^T x | Ax = b\}, \qquad A \in \{0,1\}^{m \times n}, b \in \mathbb{N}^m \tag{LP}$$

- The basic idea is to consider the problem's Lagrangian relaxation

$$\min_{x \in \{0,1\}^n} \{c^T x - \pi^T (Ax - b)\} \tag{LR}$$

where π constitutes the Lagrangian multipliers

The idea

- Lagrangian relaxation.

$$\min_{x\in\{0,1\}^n}\{c^T x|\mathbf{Ax}=\mathbf{b}\}\quad\text{(LP)}\qquad\Longleftrightarrow\qquad\max_{\pi\in\mathbb{R}^n}[\min_{x\in\{0,1\}^n}\{c^T x-\pi^T(\mathbf{Ax}\text{ - }\mathbf{b})\}]\quad\text{(D)}$$

- It is trivial to find the minimum $\hat{x}$ once the value of π has been fixed to $\hat{\pi}$

$$\hat{\pi}^T b+\min_{x\in\{0,1\}^n}(c^T-\hat{\pi}^T A)x$$

- Obviously, $\hat{x}$ is optimal iff

$$\hat{x}_i=\begin{cases}1 & \text{if}\quad (c^T-\hat{\pi}^T A)_i<0,\\ 0/1 & \text{if}\quad (c^T-\hat{\pi}^T A)_i=0,\\ 0 & \text{if}\quad (c^T-\hat{\pi}^T A)_i>0.\end{cases}$$

The idea

- Lagrangian relaxation.

$$\min_{x\in\{0,1\}^n}\{c^T x|\mathbf{Ax}=\mathbf{b}\} \quad \text{(LP)} \quad \Longleftrightarrow \quad \max_{\pi\in\mathbb{R}^n}[\min_{x\in\{0,1\}^n}\{c^T x-\pi^T(\mathbf{Ax}\text{ - }\mathbf{b})\}] \quad \text{(D)}$$

- It is trivial to find the minimum $\hat{x}$ once the value of π has been fixed to $\hat{\pi}$

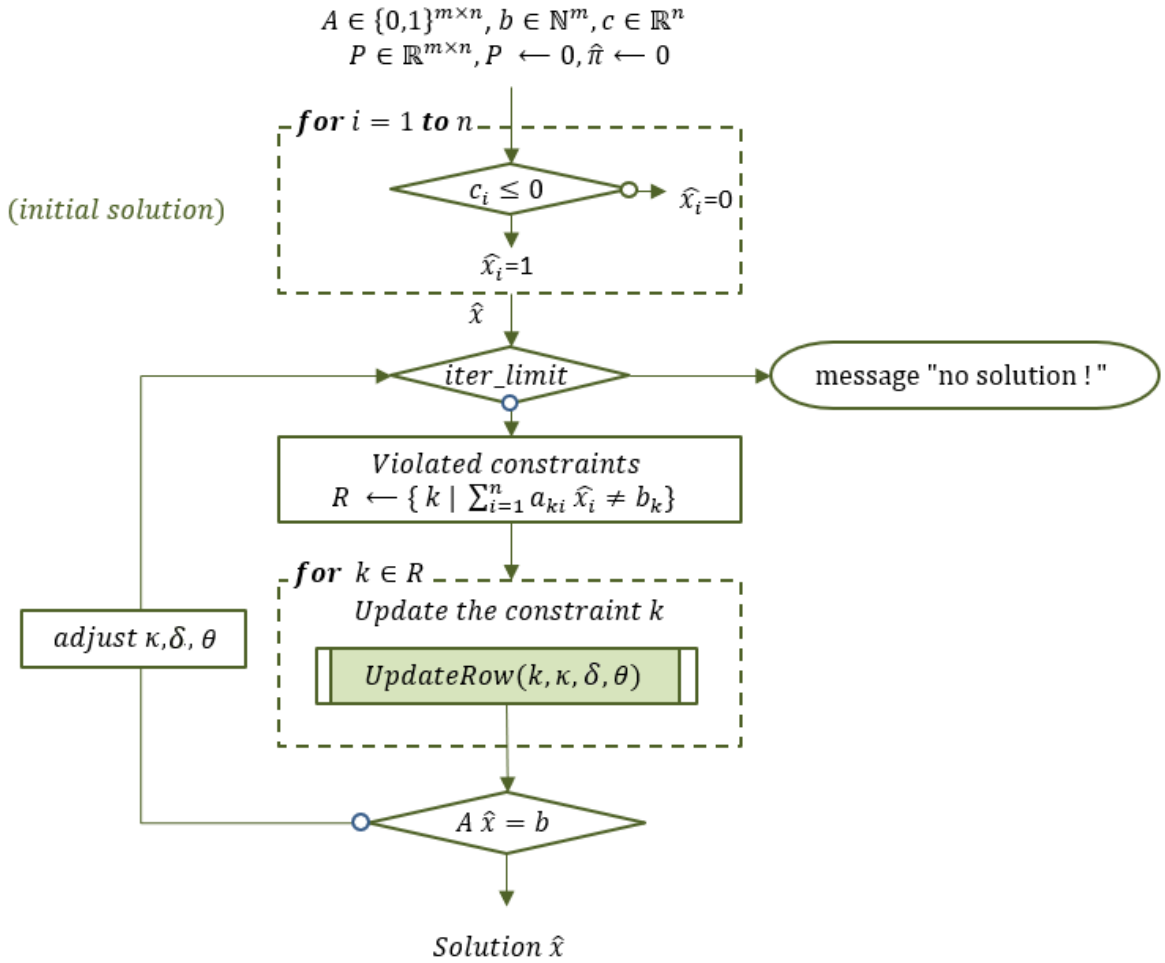
$$\hat{\pi}^T b+\min_{x\in\{0,1\}^n}(c^T-\hat{\pi}^T A)x$$

- Obviously, $\hat{x}$ is optimal iff

$$\hat{x}_i=\begin{cases}1 & \text{if} \quad (c^T-\hat{\pi}^T A)_i<0,\\ 0/1 & \text{if} \quad (c^T-\hat{\pi}^T A)_i=0,\\ 0 & \text{if} \quad (c^T-\hat{\pi}^T A)_i>0.\end{cases}$$

The goal is to manipulate π to achieve a solution $\hat{x}$ with a good value for (LR) which is feasible for (LP).

Basic algorithm



UpdateRow algorithm

Index k of the row to update

$\hat{\pi}, P$

$\kappa \in [0,1[, l \geq 0, et\ \theta \in [0,1]$

Le I be the support of the k^{th} equation $Ax = b$

$I \leftarrow \{ i \mid a_{ki} \neq 0 \}$

for $i \in I$

Decrease the influence of local preferences

$p_{ki} \leftarrow \theta\, p_{ki}$

for $i \in I$

Calculate reduced costs

$r_i \leftarrow c_i - \sum_{h \in H} |a_{hi}|\, \hat{\pi}_h - \sum_{h \in H} |a_{hi}|\, p_{hi}$

Sort columns increasingly by reduced costs

$r_{\phi(1)} \leq r_{\phi(2)} \leq \ \ldots \leq r_{\phi(|I|)}$

Update LAgrangian multipliers

$\hat{\pi}_k \leftarrow \hat{\pi}_k + \frac{1}{2}(r_{\phi(b_k)} + r_{\phi(b_k+1)})$

Update variables and preferences

$l \leftarrow (\kappa/(1-\kappa))(r_{\phi(b_k)} - r_{\phi(b_k+1)}) + \delta$

for $j = 1\ to\ b_k$ ***do***

$\hat{x}_{\phi(j)} \leftarrow 1, p_{k\phi(j)} \leftarrow p_{k\phi(j)} + l$

for $j = b_k + 1\ to\ |I|$ ***do***

$\hat{x}_{\phi(j)} \leftarrow 0, p_{k\phi(j)} \leftarrow p_{k\phi(j)} - l$

$\hat{x}, \hat{\pi}, P$

Example :

$$C = [3 \quad 8 \quad 6 \quad 5 \quad 4 \quad 2 \quad 3 \quad 2 \quad 2 \quad 3 \quad 10 \quad 8 \quad 6 \quad 8 \quad 10 \quad 5]$$

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \quad b = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$\boldsymbol{k = 1}$ $\boldsymbol{(1^{st}\ constraint)}$

$I \leftarrow \{ i \mid a_{1i} \neq 0 \} = \{1,2,3,4\}$

$r_1 = 3 - (2 \times 0) - (2 \times 0) = 3$
$r_2 = 8 - (2 \times 0) - (2 \times 0) = 8$
$r_3 = 6 - (2 \times 0) - (2 \times 0) = 6$
$r_4 = 5 - (2 \times 0) - (2 \times 0) = 5$

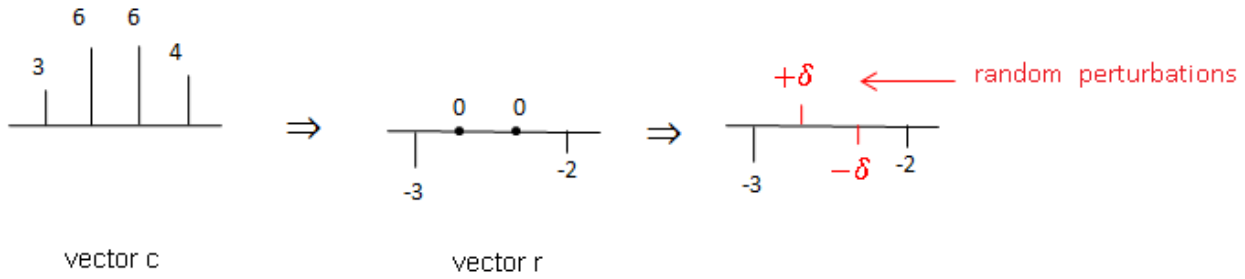
$3 < 5 < 6 < 8$

$\hat{x} = [1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]$

$\hat{\pi}_1 \leftarrow \hat{\pi}_1 + \frac{1}{2}(3 + 5) = 4$

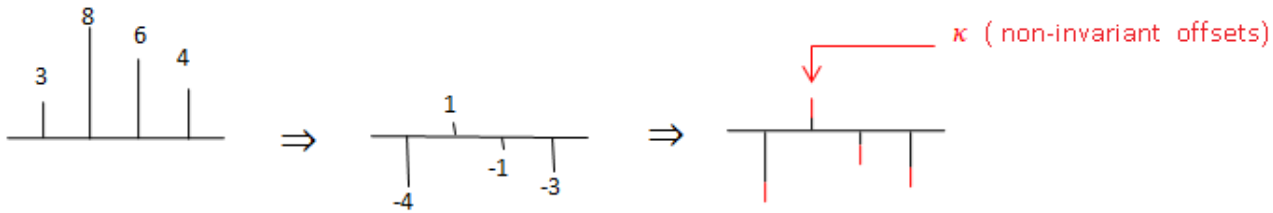
Parameters

Parameter δ



Parameter κ

(κ_{min}, κ_{step} & κ_{max})



Parameters

Parameter α

$$\kappa_{new} \leftarrow \begin{cases} \kappa_{min} & \text{if} \quad i \leq w \\ \kappa_{old} + \kappa_{step}(\frac{|R|}{m})^{\alpha} & \text{otherwise} \end{cases}$$

where :

R :	set of violated constraints
m :	total number of constraints
i :	number of iterations performed
w :	number of warmup iterations

Parameter θ gives some control over the history of the preference matrix P

$\theta = 0$	reset the preference values
$\theta = 1$	keeps the full preference history

General problems

The original algorithm has been designed for IP of the following form :

$$\min_{x \in \{0,1\}^n} \{c^T x | Ax = b\}, \quad A \in \{0,1\}^{m \times n}, b \in \mathbb{N}^n.$$

Handling more general problems [Bastert et al., 2010]

1. Integer variables $x \in \mathbb{N}^n$
2. **Inequalities** $Ax \leq b$
3. **Negative coefficients** $A \in \{-1, 0, 1\}^{m \times n}$
4. General coefficients $A \in \mathbb{Z}^{m \times n}$

Parameter tuning

Wedelin algorithm includes numerous (integer, real and categorical) parameters :

parameter	type	domain
limit	integer	$[1, +\infty[$
w	integer	$[0, +\infty[$
θ	real	$[0, 1]$
δ	real	$[0, +\infty[$
κ_{min}	real	$[0, \kappa_{max}[$
κ_{step}	real	$[0, +\infty[$
κ_{max}	real	$[\kappa_{min}, \infty[$
α	real	$[0, +\infty[$
$constraint_order$	categorical	none,reversing, random
...	...	...

The choice of these parameters depends on the problem to solve.

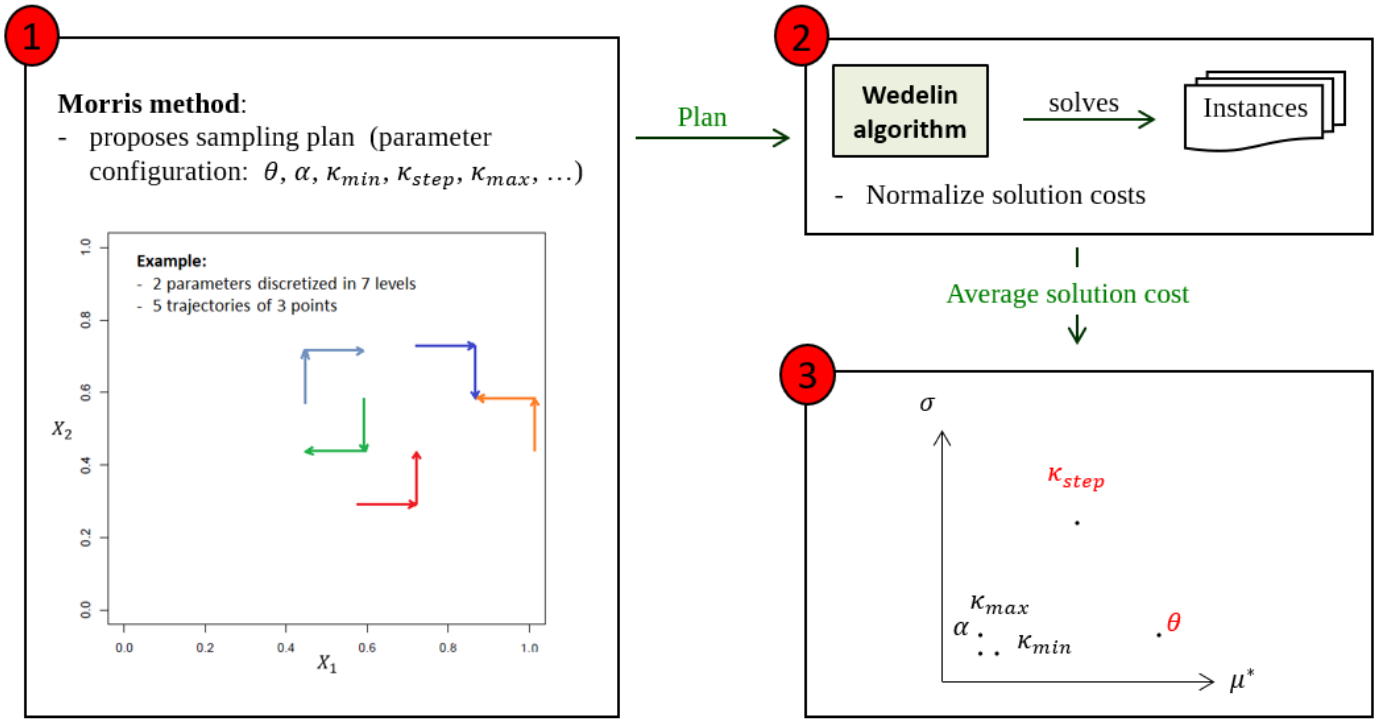
To avoid tuning all the parameters, we choose those with a great influence on the results, and then we tune them to find the best configuration values.

Application :

1. Sensitivity Analysis (Morris Method) [Morris, 1991]
   $\Rightarrow$ parameters with great influence on the results.
2. Tuning the found parameters
   - Rgenoud (Genetic Optimization Using Derivatives) [Mebane Jr and Sekhon, 2011]
   - paramILS (Iterated Local Search) [Hutter et al., 2007]

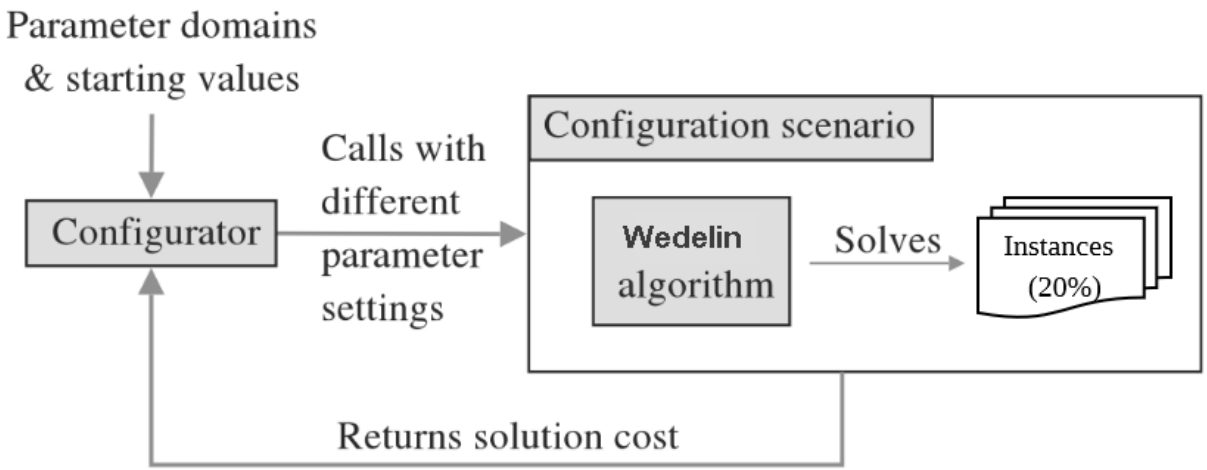
Procedure

1. Choose 20% of a collection of problem instances
2. Apply sensitivity analysis



Procedure

3. Find the best parameter configuration



4. Apply the parameter setting found on the entire collection of instances (100%)

Experimental results

- We have implemented a C++ parallel version of the Wedelin algorithm.
- The solver named baryonyx (https://github.com/quesnel/baryonyx)
- Two execution modes :
  1. Solver mode.
  2. Optimizer mode (in parallel according to the number of processors).
- Constraint order : none / reversing / random
- Initial solution :
  - default : using cost values in the original objective function
  - random : using random values.
  - best : using best solution found if available (default otherwise).

Experimental results

- We compare baryonyx
  1. by default
  2. with paramILS parameter settings
  3. with Rgenoud parameter settings

- with
  1. vehicle scheduling algorithm [Borndörfer, 1998]
  2. 4-flip neighborhood local search algorithm [Umetani, 2017]
  3. hybrid mathematical programming solver LocalSolver 7.5 [Benoist et al., 2011]
  4. exact solver IBM ILOG cplex 12.7

- on a collection of problem instances :
  Set Partitioning Problem (SPP)

Experimental results

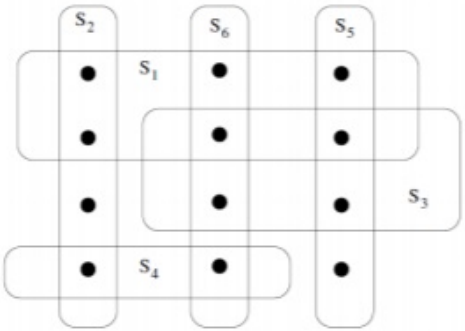
Set Partitioning Problem (SPP)

A set partitioning problem determines how the items x in one set can be partitioned into smaller subsets S. All items must be contained in one and only one partition with a minimum cost.

$$\text{Minimize} \quad z = \sum_{j \in N} c_j x_j$$

$$\text{subject to} \quad \sum_{j \in N} a_{ij} x_j = 1, \quad \forall\, i \in M$$

$$x_j \in \{0, 1\}, \quad \forall\, j \in N$$



Solution : S_2, S_5, S_6

SPP : Sensitivity analysis of benchmark instances

81 benchmark instances of two families of Set Partitioning Problem

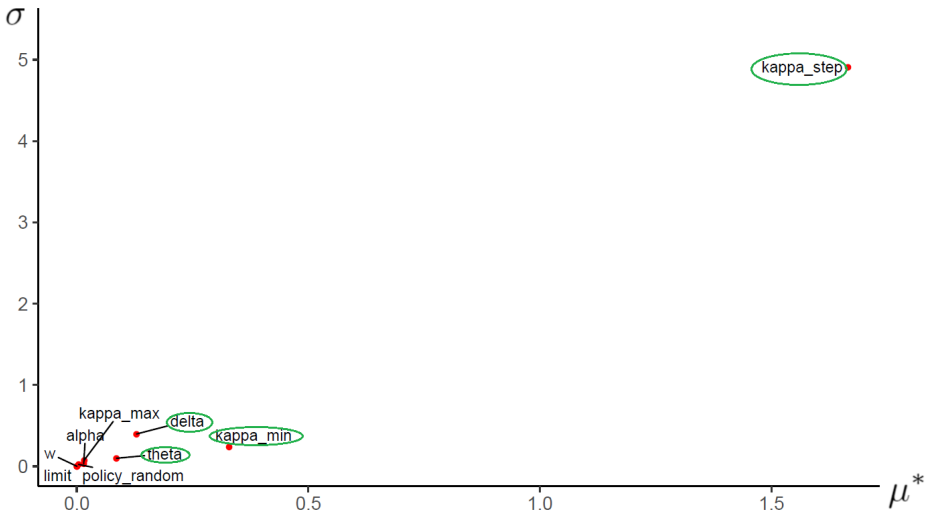
9 parameters of baryonyx solver :

parameter	type	domain
limit	integer	$[100, 10000]$
w	integer	$[0, 100]$
θ	real	$[0, 1]$
δ	real	$[10^{-3}, 10^{-1}]$
κ_{min}	real	$[0, 0.5]$
κ_{step}	real	$[10^{-4}, 10^{-2}]$
κ_{max}	real	$[0.6, 1]$
α	real	$[0, 2]$
$policy_{random}$	real	$[10^{-4}, 1 - 10^{-4}]$

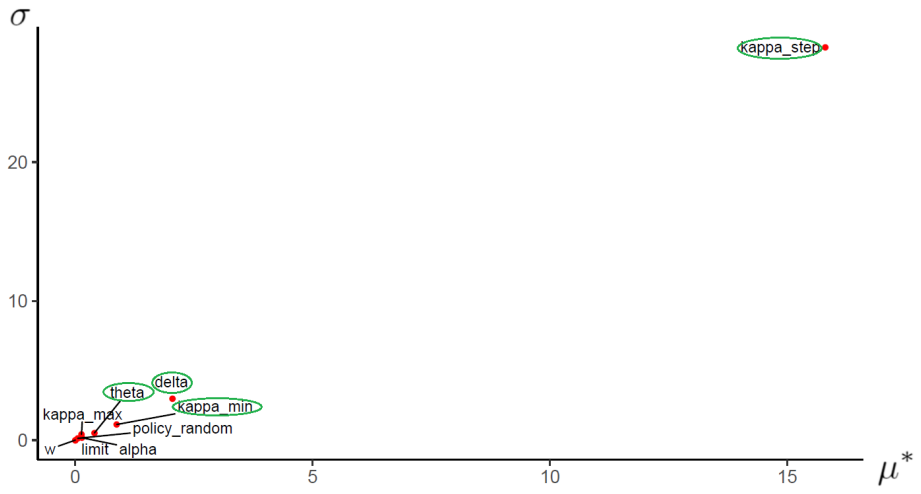
Morris analysis :
50 trajectories
$50 \times (1 + 9) = 500$ simulations of the model, 60 seconds for each simulation (parallel execution).

SPP : Sensitivity analysis of benchmark instances

SPP [K.L.Hoffman, 1993]



Telebus [Borndörfer, 1998]



Parameters with a significant effect on results : κ_{step}, κ_{min}, δ and θ.

SPP : Tuning barynyx's parameters for benchmark instances

Rgenoud :

- Population Size : 10
- Stopping criterion : number of generations (5 generations)
- Time elasped : 2 days (for each family of instances)

ParamILS :

- Prior discretization
- Stopping criterion : turner time (12000s)

SPP : Computational results of benchmark instances

limit time x core nb	Born. 2h×1	Umetani 1h×1	cplex 1h×1	cplex 60s×10	localsolver 60s×10	baryonyx 60s×10	baryonyx^{ILS} 60s×10	baryonyx^{rgd} 60s×10
SPP instances								
Dist. to the optimum								
sppaa01-06 (6)	0.00% (6)	1.00% (6)	0.00% (6)	0.00% (6)	7.34% (5)	0.63% (6)	0.13% (6)	0.24% (6)
sppus01-04 (4)	0.00% (4)	0.04% (4)	0.00% (4)	0.00% (4)	0.25% (3)	0.25% (3)	0.25% (3)	0.04% (4)
sppkl01-02								
sppnw01-43 (45)			0.00% (45)	0.00% (45)	9.38% (45)	0.19% (45)	0.04% (45)	0.05% (45)
Telebus instances								
Dist. to the best solution found								
v0417-0421 (6)	0.00% (6)	0.00% (6)	0.00% (6)	0.00% (6)	0.04% (6)	0.22% (6)	0.03% (6)	0.05% (6)
v1616-1621 (6)	0.00% (6)	0.05% (6)	0.00% (6)	0.00% (6)	6.75% (6)	1.29% (6)	1.61% (6)	0.28% (6)
t0415-0421 (7)	1.88% (7)	0.95% (6)	0.00% (7)	0.67% (7)	∞ (0)	0.39% (7)	0.13% (7)	0.04% (7)
t1716-1722 (7)	5.08% (7)	6.82% (7)	16.53% (7)	124.20% (7)	105.05% (3)	6.21% (7)	0.41% (7)	1.87% (7)

(n) : number of solved instances

∞ : no feasible solution was obtained within the time limit

Conclusions and Perspectives

Conclusions

- ✓ Implementation of generalized wedelin algorithm (baryonyx solver)
- ✓ Automatic tuning of baryonyx's parameters.
- ✓ Comparison with other methods on SPP shown good performances of baryonyx

Perspectives

- Continue the algorithm generalization study :
  - Integer variables
  - Real coefficients
  - Quadratic objective function
- Test baryonyx's performance on more general problems.


Bastert, O., Hummel, B., and de Vries, S. (2010).
A generalized Wedelin heuristic for integer programming.
INFORMS Journal on Computing, 22(1) :93–107.


Benoist, T., Estellon, B., Gardi, F., Megel, R., and Nouioua, K. (2011).
Localsolver 1. x : a black-box local-search solver for 0-1 programming.
4OR : A Quarterly Journal of Operations Research, 9(3) :299–316.


Borndörfer, R. (1998).
Aspects of set packing, partitioning, and covering.


Hutter, F., Hoos, H. H., and Stützle, T. (2007).
Automatic algorithm configuration based on local search.
In *Aaai*, volume 7, pages 1152–1157.


Mebane Jr, W. R. and Sekhon, J. S. (2011).
Genetic optimization using derivatives : the rgenoud package for R.
Journal of Statistical Software, 42(11) :1–26.


Morris, M. D. (1991).
Factorial sampling plans for preliminary computational experiments.
Technometrics, 33(2) :161–174.


Umetani, S. (2017).
Exploiting variable asso. to conf. efficient local search algo. in large-scale binary integer prog.
European Journal of Operational Research.


Wedelin, D. (1995).
An algorithm for large scale 0/1 integer programming with application to airline crew scheduling.
Annals of operations research, 57(1) :283–301.