AutoExpe.jl : Ne coder que les méthodes de résolution

Zacharie Alès

▶ **To cite this version:**

Zacharie Alès. AutoExpe.jl : Ne coder que les méthodes de résolution. 23ème congrès annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision, INSA Lyon, Feb 2022, Villeurbanne - Lyon, France. hal-03595255

HAL Id: hal-03595255
https://hal.science/hal-03595255

Submitted on 3 Mar 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

`AutoExpe.jl` : Ne coder que les méthodes de résolution

Zacharie Ales[1,2]

[1] UMA, ENSTA Paris, Institut Polytechnique de Paris, Palaiseau, France.
[2] CEDRIC, Conservatoire National des Arts et Métiers, Paris, France.
zacharie.ales@ensta-paris.fr

Mots-clés : *package Julia, automatisation, expérimentations numériques.*

1 Introduction

`AutoExpe.jl` [1] est un package julia permettant d'automatiser la réalisation d'expérimentations numériques et la génération de tableaux de résultats afin de se concentrer sur l'essentiel : l'implémentation des méthodes de résolution et la comparaison de leurs performances.

Dans ce cadre, une expérimentation consiste à appliquer plusieurs *algorithmes* à des *instances* d'un problème d'optimisation (P) en faisant éventuellement varier des *paramètres*. Afin d'illustrer ces différentes notions, considérons un exemple.

2 Application au sac à dos multiple

Soit un problème de sac à dos multiple comportant m sacs de capacité K ainsi que n objets de poids $p \in \mathbb{R}^n$ et de valeur $v \in \mathbb{R}^n$. L'objectif consiste à trouver une solution maximisant la valeur des objets mis dans les sacs tout en respectant leur capacité. Ceci peut être modélisé par un PLNE comportant des variables binaires x_{ij} égales à 1 si et seulement si l'objet i est mis dans le sac j :

$$(P) \begin{cases} \max & \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n} v_{ij}x_{ij} & \\ \text{s.c.} & \displaystyle\sum_{i=1}^{n} p_i x_{i,j} \leq K & j \in \{1,...,m\} \\ & \displaystyle\sum_{j=1}^{m} x_{i,j} \leq 1 & i \in \{1,...,n\} \\ & x_{ij} \in \{0,1\} & i \in \{1,...,n\},\ j \in \{1,...,m\} \end{cases} \tag{1}$$

Supposons que l'on souhaite comparer les performances de deux heuristiques pour ce problème :

- **heuristique** (H_1) : met aléatoirement des objets dans les sacs ;
- **heuristique** (H_2) : considère les objets par ratio $\frac{v}{w}$ décroissant et place toujours un objet dans un des sacs les plus rempli.

Nous souhaitons observer l'évolution des performances de ces deux méthodes en fonction du nombre de sacs m et de leur capacité K afin d'obtenir automatiquement un tableau de résultats tel que celui représenté en Table 1.

Dans ce contexte, une *instance* est un couple de vecteurs (p, v) et nous faisons varier deux *paramètres* : m et K.

1. `https://github.com/ZacharieALES/AutoExpe.jl`

K	m	Objective			Time (ms)	
		(H_1)	(H_2)		(H_1)	(H_2)
10	1	61	**492**	(+88%)	81.36	30.72
	2	172	**780**	(+79%)	0.05	0.03
	3	171	**1021**	(+84%)	0.04	0.03
20	1	160	**533**	(+70%)	0.03	0.03
	2	249	**799**	(+68%)	0.03	0.04
	3	339	**1025**	(+67%)	0.03	0.03
30	1	157	**568**	(+71%)	0.04	0.03
	2	269	**846**	(+69%)	0.03	0.03
	3	378	**1091**	(+66%)	0.03	0.04

TAB. 1 – Table générée automatiquement par `AutoExpe.jl`. Chaque entrée correspond à une valeur moyenne sur l'ensemble des instances considérées.

3 Fonctionnalités

La réalisation d'expérimentations de ce type est très courante en recherche opérationnelle et nécessite généralement l'implémentation de tâches telles que :

1. l'application de toutes les méthodes de résolution sur toutes les instances pour toutes les combinaisons de paramètres ;
2. la sauvegarde des résultats : choix du format, lecture, écriture ;
3. l'affichage de logs : avancement, messages d'erreurs, ... ;
4. la gestion des interruptions : sauvegardes régulières, reprise de l'expérimentation sans recalcules inutiles ;
5. la génération de tableaux de résultats LaTeX partiels pendant l'expérimentation, puis complets.

Chacune de ces tâches ne pose aucune difficulté théorique mais nécessite plus ou moins de temps pour être implémentée. Le package `AutoExpe.jl` les automatise, réduisant ainsi le développement d'une expérimentation à :

— l'implémentation des méthodes de résolution ;
— l'écriture d'un fichier JSON décrivant l'expérience : liste les méthodes, les instances et les paramètres considérés ;
— l'écriture d'un fichier JSON par table de résultats souhaitée.

Le gain de temps engendré par l'usage de cette librairie peut permettre d'aller plus loin dans les expérimentation, par exemple en testant un plus grand nombre de variantes des méthodes de résolution (*e.g.*, variations de paramètres tels que ceux de CPLEX ou Gurobi). Un second avantage est de faciliter la manipulation de tableaux de résultats LaTeX (ajout, suppression ou changement de l'ordre des lignes ou des colonnes, résultats présentés par instance ou moyennés, ...).

Une limitation actuelle du package est qu'il ne peut être utilisé que pour des méthodes de résolution implémentées en Julia. Cependant, il devrait à terme être utilisable quel que soit le langage et ne nécessiter que des connaissances basiques en Julia.