Constraint Programming Project

Le projet CP comporte deux problèmes à traiter, pour lesquels il faudra rendre sous la forme d'une **archive zip ou tgz** :

- un petit rapport au format pdf indiquant pour chaque problème :
  - un rapide descriptif du problème
  - un modèle PyCSP^3 (avec possibles variantes), accompagnée de sa description
  - les problèmes rencontrés, et les choix adoptés, lors de la modélisation
  - les tests expérimentaux effectués (sur la base de différents jeux de donnés), avec notamment un tableau indiquant les performances obtenues avec différentes options de résolution
- le(s) fichier(s) PyCSP^3
- les fichiers de données originaux (au format JSON)

Condition de réalisation du projet :

- Travail réalisé seul(e) ou en binôme
- Langues possibles pour le rapport : français ou anglais
- Date de rendu : fin Mars / début Avril (date à confirmer)

1 Frequency Allocation

Il s'agit du problème décrit en section 2 de :

> Pascal Van Hentenryck, Laurent Michel, Laurent Perron, Jean-Charles Régin:
> *Constraint Programming in OPL*. PPDP 1999: 98-116
> https://www.researchgate.net/publication/2607766_Constraint_Programming_in_OPL

Il faudra :

- convertir l'instance (données dans l'article) au format JSON
- convertir le modèle proposé dans l'article en PySCP^3
- introduire l'objectif minimisant le nombre de fréquences différentes utilisées
- tester l'instance, avec Ace

2 Test Scheduling Problem

Il s'agit du problème 073 sur CSPLib: https://www.csplib.org/Problems/prob073/, qui fut l'objet du "Industrial Modelling Challenge" à la conférence CP2015. Trois fichiers JSON de test sont fournis :

- t10-example.json (optimum 11)
- t20m10r3-1.json (optimum 1876)
- t40m10r3-2.json (optimum 1725)

mais il est souhaité qu'une campagne expérimentale concerne davantage d'instances (disponibles sur le site).

Le modèle présent sur le site peut difficilement être exploité car le langage OPL est spécialisé. On pourra néanamoins chercher à utiliser des contraintes en intension, en extension et/ou NoOverlap.