TutoQueenOrTools

March 15, 2020

Si vous voulez faire tourner le code éxecuter cette commande :

[0]:
```
!pip install ortools
```

Pour pouvoir utiliser ortools il faut l'avoir télécharger grâce à la commande suivante (cmd ou annaconda prompt si vous utilisez anaconda): `pip install ortools`

Si ortools s'est bien installé vous pouvez directement passer à la partie **Or Tools** Sinon il peut y avoir plusieurs raisons (si vous utilisez anaconda je ne sais pas si ces problèmes correspondent à votre situation) :

- Vous n'avez pas installé Microsoft Visual Studio C++ : https://support.microsoft.com/fr-fr/help/2977003/the-latest-supported-visual-c-downloads
- pip a besoin d'être mis à jour, ouvrez un terminal (cmd) et entrez la commande : `python -m install --upgrade pip` Réessayez d'installer ortools.
- Il se peut que vous ayez une version 32-bits de python, pour vérifier ouvrez un terminal :
- Ouvrez un interpréteur python avec la commande : `py`
- Importez le module struct : `import struct`
- Tapez la ligne : `print(struct.calcsize("P")*8` 64 –> votre version supporte ortools. 32 –> votre version ne supporte pas ortools, il faut passer à la version 64 bits.

Pour passer à la version 64 bits suivez les étapes suivantes (je précise encore une fois je ne pense pas que cela s'applique si vous utilisez anaconda) : * Désinstallez Python (Touche Windows -> Cherchez 'Applications et fonctionnalités' –> Python –> Désinstaller) * Rendez vous sur le lien suivant : https://www.python.org/downloads/release/python-382/ allez en bas de la page et téléchargez "Windows x86-64 executable installer", cochez la case 'Add Python to PATH' puis séléctionnez l'installation conseillée * Une fois bien installé, mettez en place python pour votre éditeur de texte et installez ortools (en général sur la page de l'éditeur il y a un tuto). Pour ceux qui utilisent visual studio code (que je vous recommande) voici un tuto : https://code.visualstudio.com/docs/python/python-tutorial

Si vous avez encore des erreurs essayez de copier/coller le message d'erreur dans Google pour trouver une solution.

1 Or Tools

Pour commencer nous avons besoin d'importer 'cp_model' du module ortools

[0]:
```
from ortools.sat.python import cp_model
```

Pour créer un model à résoudre on utlise la ligne suivante :

[0]:
```
monModel = cp_model.CpModel()
```

Je vais prendre comme exemple le problème des reines. Pour cela j'ai besoin de créer une liste qui contiendra les coordonnées i (ligne) et j (colonne). Si les coordonnées de ma reine 1 sont (1,0) on aura `plateau[0]= [1,0]`

Il faut créer les variables sur lesquelles les contraintes vont s'appliquer, cela se fait avec la fonction `NewIntVar(*borneInf*,*bornSup*,*nom*)`.

Notez que les bornes sont incluses. Pour notre problème nous avons huit list de deux int, chacun compris entre 0 et 7.

J'utilise une compréhension de liste pour créer mon plateau (si vous ne savez pas ce que c'est je vous conseil fortement de regarder des tutos sur internet, c'est abstrait au début mais c'est très pratique une fois que l'on a compris le concepte)

[4]:
```
plateau = [[monModel.NewIntVar(0,7,f'Reine : {k} : i'),monModel.
 ↪NewIntVar(0,7,f'Reine : {k} : j')] for k in range(8)]
plateau
```

[4]:
```
[[Reine : 0 : i(0..7), Reine : 0 : j(0..7)],
 [Reine : 1 : i(0..7), Reine : 1 : j(0..7)],
 [Reine : 2 : i(0..7), Reine : 2 : j(0..7)],
 [Reine : 3 : i(0..7), Reine : 3 : j(0..7)],
 [Reine : 4 : i(0..7), Reine : 4 : j(0..7)],
 [Reine : 5 : i(0..7), Reine : 5 : j(0..7)],
 [Reine : 6 : i(0..7), Reine : 6 : j(0..7)],
 [Reine : 7 : i(0..7), Reine : 7 : j(0..7)]]
```

Si vous avez exécuté le code ci-dessus vous verrez que l'on a bien 8 liste de deux valaurs qui peuvent prendre des valeurs entre 0 et 7.

Pour appliquer des contraintes les méthodes `Add` et `AddAllDifferent` seront suffisantes pour les problèmes vus en TP. * `Add` : permet d'appliquer une contrainte de type boolean, par exemple si vous avez deux variables x et y de type int et que vous voulez que x < y il faut écrire `monModel.Add(x<y)` * `AddAllDifferent` : permet d'appliquer une contrainte de difference à une liste. Si je veux préciser que tous les éléments de maList doivent être différent on écrit `monModel.AddAllDifferent(maList)`

Pour le problème des reines les contraintes à poser sont les suivantes :

- Que toutes les coordonnées lignes et colonnes soient différentes. (Contraintes horizontales)
- Que la différence `i-k` et `|j-z|` des coordonnées `(i,j)` et `(k,z)` ne soit pas égale (Contraintes diagonales)

[0]:
```
#Contraintes horizontales
col = [plateau[i][0] for i in range(8)]
```

```
row = [plateau[i][1] for i in range(8)]
monModel.AddAllDifferent(col)
monModel.AddAllDifferent(row)




#Contraintes diagonales
for i in range(8):
  for j in range(i+1,8):
    monModel.Add(plateau[i][0] - plateau[j][0] != plateau[i][1] - plateau[j][1])
    monModel.Add(plateau[i][0] - plateau[j][0] != -(plateau[i][1] -␣
 ↪plateau[j][1]))
```

Maintenant que toutes mes contraintes sont appliqués le code suivant permet de lancer le solver.

[0]:
```
solver = cp_model.CpSolver()
status = solver.Solve(monModel)
```

Si notre problème est résolvable on récupère les coordonnées dans une liste :

[7]:
```
results = []
if status==cp_model.FEASIBLE:
  for i in range(8):
      results.append([solver.Value(plateau[i][0]),solver.Value(plateau[i][1])])
else:
  print('Problème non résolvable')
print(results)
```

```
[[1, 7], [0, 3], [2, 0], [4, 5], [5, 1], [6, 6], [3, 2], [7, 4]]
```

Puis on affiche les résultats :

[8]:
```
if results:
  for i in range(8):
    for j in range(8):
      if [i,j] in results:
        print('X',end=' ')
      else:
        print('0',end=' ')
    print()
```

```
0 0 0 X 0 0 0 0
0 0 0 0 0 0 0 X
X 0 0 0 0 0 0 0
0 0 X 0 0 0 0 0
0 0 0 0 0 X 0 0
0 X 0 0 0 0 0 0
0 0 0 0 0 0 X 0
```

0 0 0 0 X 0 0 0