Réunion du 14/06

14/06/2024

Variable utilisées

10 premières variables

```
les 1e variables avec le plus de domaine: ['5']
les 2e variables avec le plus de domaine: ['3']
les 3e variables avec le plus de domaine: ['1']
les 1 variable avec le plus de contraint: ['3', '5', '6']
```

20 premières variables

```
les 1e variables avec le plus de domaine: ['5']
les 2e variables avec le plus de domaine: ['3']
les 3e variables avec le plus de domaine: ['1']
les 1 variable avec le plus de contraint: ['3', '16']
```

30 premières variables

```
les 1e variables avec le plus de domaine: ['5']
les 2e variables avec le plus de domaine: ['3']
les 3e variables avec le plus de domaine: ['1']
les 1 variable avec le plus de contraint: ['3', '16']
les 2 variable avec le plus de contraint: ['1', '2', '5', '6', '9', '11', '13', '15', '19', '20']
```

Optimisation du temps

Problème : trop de variables booléennes

```
choix= []
for k in range (1,i+1):
    choix.append(model.NewBoolVar(f"{lists} et {j} et {k} "))
dernier=model.NewBoolVar('dernier')
```

Création de i+1 booléens pour chaque tuple dans la contrainte

```
for k in range (i):
    model.Add(dico_variable[var[k]]==int(list[k])).OnlyEnforceIf(choix[k])
    model.Add(dico_variable[var[k]]!=int(list[k])).OnlyEnforceIf(choix[k].Not())

  model.AddBoolAnd(choix[:-1]).OnlyEnforceIf(dernier)
  model.Add(dernier==1).OnlyEnforceIf(choix[:-1])
  model.AddImplication(dernier,choix[i-1])
```

(i*2)+3 contraintes pour chaque tuple dans la contrainte

i = nombre de variables qui est concerné par la contrainte
var= nom des variables
list= valeurs à assigner
choix= variables booléennes

Pour 10 premières variables (1 contrainte avec 3 variables de 1050 tuples) :

10+1050*(3+1)= 4210 variables

1050*(3+3)= 9450 contraintes

Optimisation du temps

1 variable booléenne pour chaque contrainte

```
#Identify the variables related to the constraints
condition=model.NewBoolVar(f'{constraint.get("name")}')
```

2 contraintes pour chaque contrainte

```
for j in range (len(lists)):
    list=lists[j].split(' ')
    intlists.append([int(l) for l in list])
    intlists2.append([int(l) for l in list[:-1]])

model.AddAllowedAssignments([dico_variable[v] for v in var],intlists).OnlyEnforceIf(condition)
model.AddForbiddenAssignments([dico_variable[v] for v in var[:-1]],intlists2).OnlyEnforceIf(condition.Not())
```

Pour 10 premières variables (1 contrainte avec 3 variable de 1050 tuples) :

10+1= 11 variables

2= 2 contraintes

Optimisation du temps: Résultat

Pour 10 première variables = 1 512 000 combinaison

V1

10 premières variables avec contraintes:

```
Status = OPTIMAL
Number of solutions found: 756000
1014.8853089809418
```

10 premières variables avec contraintes + assignation de la variable 5 :

```
Status = OPTIMAL
Number of solutions found: 18000
13.856397151947021
```

V2

10 premières variables avec contraintes:

```
Status = OPTIMAL
Number of solutions found: 756000
solution :118.77489185333252
```

10 premières variables avec contraintes + assignation de la variable 5 :

```
Status = OPTIMAL
Number of solutions found: 18000
solution :1.502542495727539
```

Optimisation du temps: Résultat

Pour 10 première variables = 1 512 000 combinaison

assignation des variables 5 et 3

```
Status = OPTIMAL
Number of solutions found: 720
solution :0.08507800102233887
```

assignation des variables 5,3 et 1:

```
Status = OPTIMAL
Number of solutions found: 80
solution :0.03693437576293945
```

assignation des variables 5,3 et 6:

```
Status = OPTIMAL
Number of solutions found: 720
solution :0.0834505558013916
```

Recommandation

```
les 1e variables avec le plus de domaine: ['5']
les 2e variables avec le plus de domaine: ['3']
les 3e variables avec le plus de domaine: ['1']
les 1 variable avec le plus de contraint: ['3', '5', '6']
```

Optimisation du temps: Résultat

Pour 20 première variables = 2 322 432 000 combinaison

assignation des variables 5 et 3

```
Status = OPTIMAL
Number of solutions found: 411264
solution :60.4783718585968
```

assignation des variables 5,3 et 1:

```
Status = OPTIMAL
Number of solutions found: 44928
solution :6.1828954219818115
```

assignation des variables 5,3 et 16:

```
Status = OPTIMAL
Number of solutions found: 103680
solution :14.787682294845581
```

Recommandation

```
les 1e variables avec le plus de domaine: ['5']
les 2e variables avec le plus de domaine: ['3']
les 3e variables avec le plus de domaine: ['1']
les 1 variable avec le plus de contraint: ['3', '16']
```

assignation des variables 5,3,1,16:

```
Status = OPTIMAL
Number of solutions found: 10368
solution :1.332580327987671
```

Optimisation du temps: Résultat

Pour 30 première variables = 4 013 162 496 000 combinaison

Recommandation

```
les 1e variables avec le plus de domaine: ['5']
les 2e variables avec le plus de domaine: ['3']
les 3e variables avec le plus de domaine: ['1']
les 1 variable avec le plus de contraint: ['3', '16']
les 2 variable avec le plus de contraint: ['1', '2', '5', '6', '9', '11', '13', '15', '19', '20']
```

assignation de 8 variables:

```
Status = FEASIBLE
Number of solutions found: 693111
solution :125.97974920272827
```

assignation de 12 variables:

```
Status = OPTIMAL
Number of solutions found: 331776
solution :56.831883668899536
```

Aide à l'utilisateur : type de recommandation

Filtrage collaboratif :

- Recommande des produits en fonction de la similarité avec des utilisateurs
- Problème : nombreuses données (utilisateur et produit)/ Achat anonyme/ nouveau produit

Basé sur le contenu :

- Recommandation basée sur les caractéristiques du produit (type,nationalité…)
- Problème : Information sur le produit pas facile à obtenir/ pas suffisantes pour des recommandation adaptés

Basé sur les connaissance :

- Des informations sur les préférences de l'utilisateur en posant des questions/ ou lui demander de faire des choix

Aide à l'utilisateur : Basé sur les connaissance

Type de recommandation pour la configuration interactive

Complétion:
- Compléter la configuration de l'utilisateur

Prochaine attribut:
- Recommander un attribut en fonction de la configuration partielle courante

Valeur pour un attribut:
- Recommander une valeur à un attribut en fonction des possibilité sur le configuration

Bibliographie

OR-TOOLS: https://developers.google.com/optimization

Configuration de voitures + recommandation: Apprentissage de préférences en espace combinatoire et application à la recommandation en configuration interactive de P.F.GIMENEZ