# LOG3430 - Méthodes de test et de validation du logiciel

## Laboratoire 2

### Fonctions logiques

Département de génie informatique et de génie logiciel
École Polytechnique de Montréal


Hiver 2021

# 1 Introduction

Dans ce laboratoire, vous allez tester deux fonctions logiques pour classifier les messages. Après vous allez ajouter les fonctionnalités additionnelles dans votre système et implémenter les testes d'interactions correspondants.

# 2 Objectifs

Les objectifs généraux de ce laboratoire sont :

1. Pratiquer la conception des tests pour les fonctions logiques.

# 3 Fonctions logiques

Une fonction logique prend en entrée un nombre $n$ de clauses et sort une seule sortie booléenne. Les expressions logiques peuvent être dérivées de divers artefacts logiciels, y compris le code, les spécifications et les automates finis. Dans le cas de notre système de filtrage de Spam, on va utiliser une fonction logique pour classifier les messages. Pendant la classification, nous sommes intéressés aux questions suivantes :

- Le message courant est-il Spam ?
- Est-ce que l'utilisateur a un long historique de communication ?
- Quel est le niveau de confiance envers l'utilisateur ?
- Quel est le niveau de confiance envers les groupes d'utilisateur ?

En considérant ces questions on peut proposer la fonction logique suivante. Le message va être classifié comme spam, si :

- (le message courant est spam)
  ET
- (l'historique de communication est moins qu'un mois ET le niveau de confiance de l'utilisateur est moins que 60 OU le niveau de confiance du groupe de l'utilisateur est moins 70)
  OU
- (l'historique de communication est moins qu'un mois ET le niveau de confiance de l'utilisateur est moins que 75 ET le niveau de confiance du groupe de l'utilisateur est moins que 70)

On peut définir les variables suivantes :

- S = vrai, si le message est classifié comme Spam.
- P : vrai, si le message courant est classifié comme Spam selon le vocabulaire créé.
- H : vrai, si le temps entre la date du premier message vu et la date du dernier message vu est est inférieur à un mois.
- T1 : vrai, si le niveau de Trust de l'utilisateur est moins que 60.
- T2 : vrai, si le niveau de Trust moyen des groupes de l'utilisateur est moins que 70.
- T3 : vrai, si le niveau de Trust de l'utilisateur est plus que 75.

Finalement, l'équation logique pour classifier les messages est :

$$S = P * (H * T1 + T2) + H * T2 * \neg T3 \tag{1}$$

En forme DNF, on va utiliser l'équation :

$$S = P + \neg T3 * T2 \tag{2}$$

Vous pouvez remarquer qu'il existe une limitation pour les jeux de tests pour ces fonctions : T1 et T3 ne peuvent être vraies en même temps.

## Les tâches

Votre tâche sera d'écrire les jeux des tests pour les fonctions logiques (1) et (2) pour différents critères de couvertures.

Du point de vue des tests ACC, on veut tester chaque clause quand la clause détermine le sortie de prédicat (sortie de la fonction logique). Dans ce cas, la clause qu'on teste va être la clause majeur $c_i$ et les autres - les clauses mineurs $c_j$. Considérez cela comme le fait de mettre différents membres d'une équipe en charge de l'équipe. Nous ne savons pas s'ils peuvent être des leaders efficaces jusqu'à ce qu'ils essaient [1]. La question la plus importante est de savoir si les clauses mineures $c_j$ doivent avoir les mêmes valeurs lorsque la clause majeure $c_i$ est vraie que lorsque $c_i$ est fausse. La résolution de cette ambiguïté mène a trois types d'ACC : GACC, CACC et RACC.

Les critères de couverture des clauses actives (ACC) visent à s'assurer que les clauses majeures affectent leurs prédicats. Un critère complémentaire à ACC, l'ICC garantit que la modification d'une clause majeure qui ne devrait pas affecter le prédicat n'affecte pas, en fait, le prédicat. Il y a deux types d'ICC : GICC, RICC.

Certains critères de couverture sont basés sur des prédicats écrits dans la forme DNF. Notamment les critères IC, PIC et VNS.

Vous devez **implémenter par un code** la manière de trouver les critères *RACC, RICC et VNS*. Par la suite, il faut montrer que les jeux de tests obtenus couvrent bien les critères en expliquant pourquoi comme vu en cours. Pour les autres critères, vous n'avez qu'à donner les jeux de tests avec une explication (donc, sans code).

### Tests ACC

Pour la fonction (1) proposez les jeux de tests qui satisfont les critères GACC, CACC et RACC.

### Tests ICC

Pour la fonction (1) proposez les jeux de tests qui satisfont les critères GICC et RICC.

### Tests DNF

Pour la fonction (2) proposez les jeux de tests qui satisfont les critères IC, PIC et VNS.

1. Ammann, Paul, and Jeff Offutt. Introduction to software testing. Cambridge University Press, 2016.http://lib.sgu.edu.vn:84/dspace/bitstream/TTHLDHSG/2791/1/Introduction%20to%20Software%20Testing.pdf

**Créer un fichier et nommez le *criteres.py*** pour mettre vos fonctions (les 3 fonctions de critères + les deux fonctions $S$ à tester). Comme pour le premier devoir, **créer un fichier *main.py/sh*** et mettez y toutes les commandes à exécuter pour effectuer la tâche.

# 4 Remarques

— Vous n'avez donc que deux fichiers à produire : *criteres.py* et *main.py/sh*, le reste se passe dans le rapport.
— Pour tous les critères, expliquez en quoi vos jeux de tests couvrent ces critères (comme vu en cours).

# 5 Livrables attendus

Les livrables suivants sont attendus :

— Un rapport pour le laboratoire : 12 points. Rapport doit contenir :
  — Jeux de tests demandés pour les fonctions logiques. Il faut inclure toutes les étapes intermédiaires pour la création des tests (consulter les exemples donnés en cours).
— Le dossier COMPLET contenant le projet : 8 points. **N'oubliez pas de bien commenter votre code**. Un code peu ou pas commenté sera pénalisé.

Le tout à remettre dans une seule archive **zip** avec le titre matricule1_matricule2_matricule3_lab1.zip sur Moodle. Seulement une personne d'équipe doit remettre le travail.

Le rapport doit contenir le titre et numéro du laboratoire, les noms et matricules des coéquipiers ainsi que le numéro du groupe.

# 6 Information importante

1. Consultez le site Moodle du cours pour la date et l'heure limites de remise des fichiers (deux semaines après la première séance du laboratoire).
2. Un retard de [0,24h] sera pénalisé de 10%, de [24h, 48h] de 20% et de plus de 48h de 50%.
3. Aucun plagiat n'est toléré. Vous devez soumettre juste le code, qui était fait par les membres de votre équipe.