**LOG8470**

**Vérification de la fiabilité et la sécurité**

**Automne 2018**

TP1

Model checking - iSpin

Présenté par :

Ylan Luu (1684719)

David Tremblay (1748125)

Amine Belaid (1673289)

1er novembre 2018

École Polytechnique de Montréal

Département de Génie Informatique et Logiciel



# Introduction

En classe, la théorie vue jusqu’à présent englobe la théorie de la logique temporelle linéaire et la vérification de propriétés de model checking. Comme premier travail pratique, nous devons simuler un système avionique composé de modules IMA interconnectés par un réseau de communication AFDX gérant les systèmes embarqués dans des avions civils modernes afin de valider les apprentissages théoriques faits en classe. Des modules seront modélisés à l’aide du langage de programmation PROMELA. De plus, des expressions LTL pour représenter les requis de ces systèmes seront établies et par la suite vérifiées au sein du logiciel iSPIN.

# Présentation des résultats

**Les modules IMA**

**1.**

Le code qui modélise nos modules IMA associés à cette sous-question peut être retrouvé dans le fichier LOG8470\_TP1\_Q1 et exécuté dans le logiciel iSPIN. Il est important de noter que nous avions tout de suite implémenté un comportement en cas de panne à exécuter pour chaque fonction puisqu’il n’y avait pas de requis dictant le contraire dans cette sous-question.

**2.** Considérons les modules X et Y. Si X correspond au module A de notre modèle, alors Y correspond au module B de ce modèle, et vice versa.

1. Les deux fonctions ne peuvent accéder aux ressources du module en même temps.

**⃞ (( critiqueX → ¬ critiqueY))**

**Propriété de sûreté**

1. Une fonction active (non en panne) doit pouvoir continuellement accéder aux

ressources du module.

**⃞ (panneX ⋃ (¬panneX ^ ◇critiqueX))**

**Propriété de vivacité**

1. L’une des propriétés importantes de l’approche IMA est d’assurer la non propagation d’une faille d’une fonction à l’autre. En d’autres, une faille dans une fonction ne doit pas modifier le comportement normal des autres modules.

**⃞ ((panneX ^ ¬panneY) → ◇critiqueY))**

**Propriété de sûreté & vivacité**

**3.**

Lorsque nous avons exécuté les propriétés présentées ci-haut dans le programme iSPIN, et les avons adaptées à Promela de façon à ce que celles prennent en considération tous deux des cas où X = A et Y = B, et vice versa. Chacune des trois a été validée et aucune n’a soulevé d’erreurs au sein du programme. Nous pouvons alors conclure que nos propriétés sont valides.

**4.**

Comme expliqué à la sous-question 1, dès le début de l’écriture de notre programme, nous avions déjà pris en compte tous les requis nécessaires, incluant un comportement pour le modèle à adopter en cas de panne d’une fonction. Le code complet pour la modélisation IMA se trouve, comme indiqué précédemment, dans le fichier LOG8470\_TP1\_Q1 et peut être exécuté au sein du programme iSpin. Ce qui différencie le code de la sous-question 1 de cette sous-question est l’ajout des canaux qui indiquent dans quel état le processus se trouve (en panne ou non). Dans le cas d’une panne, un timeout est exécuté pour débloquer les processus et assurer qu’on ne bloque pas.

**La communication réseau**

**1.**

Le code répondant à cette question se trouve dans le fichier LOG8470\_TP1\_Q2.

**2.**

Nous considérons les variables suivantes dans notre modèle:

* chan\_cc: Le contrôleur de communication
* chan\_com: Le commutateur
* m\_x: un message, ayant dans le contexte de notre code x une des valeurs suivantes:
  + 1: Un message du ECS
  + 2: Un message du FCS (critique)
  + 3: Un message du LGS (critique)
* m\_c: un message critique, ayant dans le contexte de notre code c une des valeurs suivantes:
  + 2: Un message du FCS
  + 3: Un message du LGS

Les propriétés suivantes suivent celles tirées des notes à ce sujet:

1. Tout message reçu par la destination est nécessairement produit par l’un des sources.

**⃞ [¬(m\_x ∈ chan\_com) ⋃ ◇(m\_x ∈ chan\_cc)]**

**Propriété de vivacité**

1. Un des requis importants des systèmes avioniques est que le réseau ne doit jamais perdre des messages de haut niveau de criticité.

**⃞ [ m\_c ∈ chan\_com ⇒ ◇ (m\_c ∈ chan\_cc) ]**

**Propriété de sûreté**

n étant le même message de chaque côté de l’équation LTL.

**3.** Nous n’avons pas été en mesure d’interpréter sur Promela la première propriété, car nous n’avons pas réussi à vérifier si un élément donné est présent dans une chaîne. Pour ce qui est de la seconde propriété, notre tentative de valider que l’ensemble des messages m\_c qui été envoyés par le CC soient reçus par le commutateur n’a pas réussi, et nous ne pouvons donc pas valider cette propriété.

**4.** Malgré notre tentative de tri des messages transmis au processus du commutateur, de façon à ce que les messages FCS et LGS y soient placés en priorité, nous ne sommes toujours pas en mesure de faire valider la propriété. Nous sommes incapables de déterminer s’il s’agit d’une erreur dans la définition de notre processus de tri au niveau du commutateur, ou bien celle de la propriété à valider.

# Difficultés rencontrées

La première difficulté rencontrée dans ce travail pratique a tout d’abord été de se familiariser avec le logiciel iSPIN afin d’y trouver les fonctionnalités nécessaires au bon déroulement du travail demandé. La documentation mise en ligne sur le site Moodle du cours, la démonstration du logiciel réalisée durant la période de laboratoire ainsi que des recherches sur internet nous ont permis de pallier cette difficulté. Ensuite, le langage PROMELA était surtout vu de manière théorique dans le cours jusqu’à présent et le langage est assez différent des langages de programmation auxquels nous avions l’habitude. Une relecture des notes de cours ainsi que l’outil de vérification de syntaxe dans iSPIN ont été les outils utilisés pour modéliser nos modules correctement en PROMELA. La traduction de propriétés en LTL était déjà développée dans des exercices de travaux dirigés et dans le mini-contrôle 1 alors ceci n’a pas trop causé de problème en théorie, toutefois leur interprétation en Promela est plus ardue, au point que celle-ci s’est avérée impossible pour nous au cours de la deuxième partie de ce travail. La modélisation des modules IMA s’est déroulée sans trop de problèmes. Par contre, la modélisation de la communication réseau a été beaucoup plus ardue, étant donné que seules les opérations effectuées sur les chaînes sont visibles dans iSPIN, et de ce fait, aussi bien dans la première partie du travail que la seconde (dans cette dernière, ces difficultés se sont avérées largement plus apparentes), il est difficile d’être certain de la fiabilité de ces modèles incomplets, de même qu’il est difficile de les déboguer via la console de la simulation sur iSPIN. La gestion des canaux était aussi un sujet abordé surtout de manière théorique et avons donc dû implémenter nos premiers canaux mais ceci était simple. L’utilisation de timeout et la simulation de systèmes tombant en panne ont aussi nécessité quelques recherches pour être implémentés. Finalement, nous n’étions pas tout à fait certains de la manière de différencier les types de messages afin de les trier dans le commutateur. Nous avons opté pour une valeur de message différent par type de message afin de facilement les stockés dans leur tampon respectif.

# Conclusion

En conclusion, ce laboratoire nous a permis d’utiliser le programme iSPIN pour modéliser des systèmes formels afin de vérifier des propriétés de model checking. Nous avons aussi appris comment vérifier des propriétés LTL dans un modèle ainsi que comment obtenir une trace d’exécution d’un modèle sous forme de graphe grâce au logiciel iSPIN. Nous avons pu aussi utiliser le langage Promela de façon plus concrète et l’utilisation de canaux et de la fonction timeout du langage. De plus, en ayant été introduit une mise en situation, il a été possible de mieux comprendre l’utilité de la matière apprise de manière théorique en classe. D’après la matière étudiée en classe suite à la théorie mise en pratique dans ce travail pratique, nous nous attendons à mettre en pratique la théorie qui en ce qui concerne les réseaux de Petri.