**Test de conformité des machines de Mealy sous restrictions d'entrée.**

**Résumé.** Nous présentons une méthode de test de conformité en boîte grise pour des réseaux de machines de Mealy interconnectées. Cette approche traite le scénario où toutes les interfaces du composant à tester sont observables, mais où ses entrées sont sous le contrôle d'autres composants en boîte blanche. Nous établissons de nouvelles conditions pour une détection complète des fautes, en exploitant de manière innovante les répétitions au travers des exécutions ramifiées de la machine composite. Enfin, nous fournissons une évaluation expérimentale de notre approche sur des compositions en cascade atteignant jusqu'à mille états, et démontrons qu'elle surpasse notablement les techniques de test en boîte noire existantes.

**I. Introduction/Motivation**

Dans cet article, nous proposons une approche de test en boîte grise pour des réseaux de machines de Mealy interconnectées. Nous abordons le scénario où toutes les communications du composant à tester peuvent être observées, mais où certaines de ses entrées sont contrôlées par d'autres parties du système en boîte blanche. La méthode présentée s'inscrit dans le cadre du test de conformité des machines à états finis (FSMs).[3,6]

Dans sa variante la plus étudiée, le problème de test de conformité pour les FSMs concerne les FSMs déterministes et à entrées complètes, c'est-à-dire les machines de Mealy.[15,20,25,8,7,22] Dans ce contexte, nous considérons une machine de Mealy M entièrement connue (la spécification) et une boîte noire B, pour laquelle nous connaissons seulement une limite k sur le nombre d'états. L'objectif est de concevoir une suite de tests afin de déterminer si la boîte noire B est conforme (équivalente) à M.

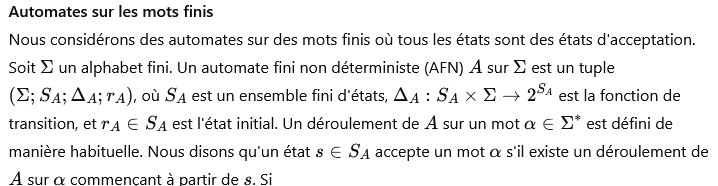
Le test de conformité basé sur les FSM est un domaine de recherche actif, et de nombreuses techniques existent dans la littérature (voir l'enquête [6], ou [24]). La principale motivation de ces techniques est la vérification des systèmes réactifs, pour lesquels les FSM constituent un modèle approprié. Malgré sa simplicité, le formalisme FSM est utilisé dans des domaines très variés, offrant une large gamme d'applications pour le test basé sur les FSM [3]. Une autre application notable du test de conformité réside dans l'apprentissage des automates [5] et les procédures dérivées, telles que le test de boîte noire [16]. Dans le cadre du « professeur minimalement adéquat » introduit par Angluin [1], ces techniques nécessitent un oracle d'équivalence dans leur application. Cependant, ces oracles sont souvent impossibles à obtenir dans les systèmes en boîte noire. En pratique, les requêtes d'équivalence sont donc simulées à l'aide de différentes stratégies de test [10]. De plus, il existe une relation bien connue entre l'inférence de modèles et le test de conformité (voir [2]), qui s'étend même aux techniques d'apprentissage des automates plus récentes, ne nécessitant pas d'oracles d'équivalence [23, 26].

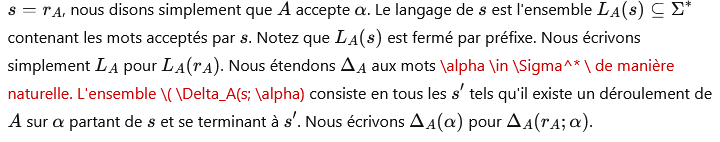
En réalité, cependant, les systèmes réactifs ne consistent que rarement en une structure monolithique unique, mais plutôt en de plus petits composants interagissant entre eux. Les techniques existantes basées sur les FSM, développées pour les systèmes en boîte noire, ne sont pas adaptées à ce contexte, car elles souffrent du problème d'explosion des états et atteignent rapidement leurs limites. Par conséquent, il existe un besoin de méthodes en boîte grise capables d'exploiter les informations sur les composants internes connus et leurs communications. Quelques travaux notables sur le test de conformité dans cette direction existent [19, 18, 17], mais il s'agit encore d'un domaine relativement peu exploré.

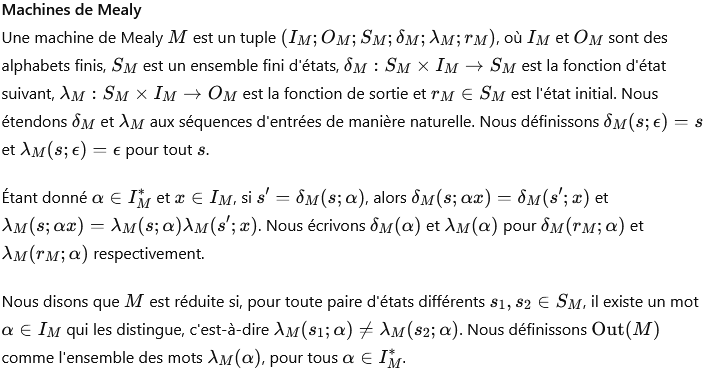
Nous considérons un scénario où toutes les interfaces du composant à tester B sont observables, mais où ses entrées sont contrôlées par d'autres composants connus du système. Un exemple typique se produit lorsque B est le composant terminal d'une composition en cascade de machines de Mealy B∘HB \circ HB∘H, comme illustré à la figure 3. La méthode de « comptage des états » [17], l'une des principales approches pour cette situation, consiste à traiter B comme une machine de Mealy partiellement spécifiée, c'est-à-dire une machine où certaines transitions sont manquantes. Cette réduction repose sur une construction classique pour la minimisation des composants par Kim et Newborn [12], qui entraîne une explosion exponentielle de la taille du problème. Cependant, des travaux récents ont montré que cette construction coûteuse n'est pas nécessaire pour optimiser les composants [14], et que des techniques plus économiques peuvent être utilisées.

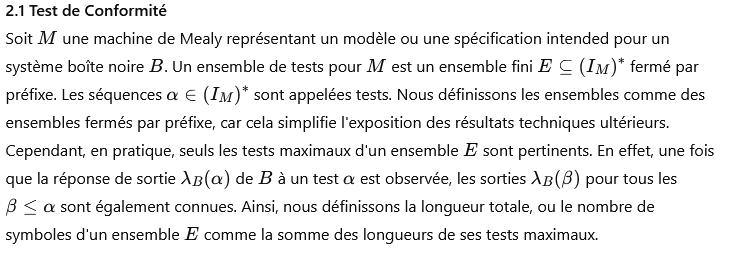
Notre principale contribution dans cet article est une généralisation de la méthode de « comptage des états », qui évite la construction de Kim-Newborn. Pour y parvenir, nous développons un formalisme permettant de raisonner sur les exécutions entrelacées dans des systèmes avec des branches universelles. Cela nous permet d'établir de nouvelles conditions suffisantes pour une détection complète des fautes dans le cadre de la boîte grise. Nous proposons deux algorithmes de test utilisant cette théorie nouvellement introduite et montrons expérimentalement qu'ils sont capables de gérer des compositions atteignant jusqu'à mille états, alors que les données expérimentales sur des exemples de taille raisonnable sont inexistantes pour les méthodes à l'état de l'art [19, 17]. De plus, nous montrons une relation pratique entre la tâche de test en boîte grise et le problème classique de l'inclusion de langage entre automates non déterministes (NFA) [13], ainsi que le problème de réduction d'états pour les NFA [11].

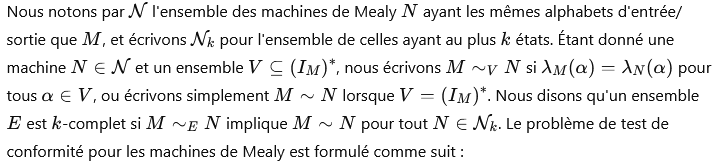
**2. Préliminaires**  
Notation Générale Étant donné un alphabet X, nous notons X∗ l'ensemble des mots finis de longueur arbitraire sur X. Nous utilisons ϵ\epsilonϵ pour désigner le mot vide, et pour un mot α\alphaα, ∣α∣|\alpha|∣α∣ représente sa longueur. Nous écrivons (α<β)(\alpha < \beta)(α<β) ou α≤β\alpha \leq \betaα≤β lorsque α\alphaα est un préfixe (strict) de β\betaβ.

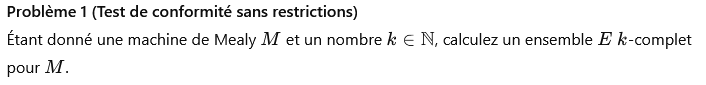




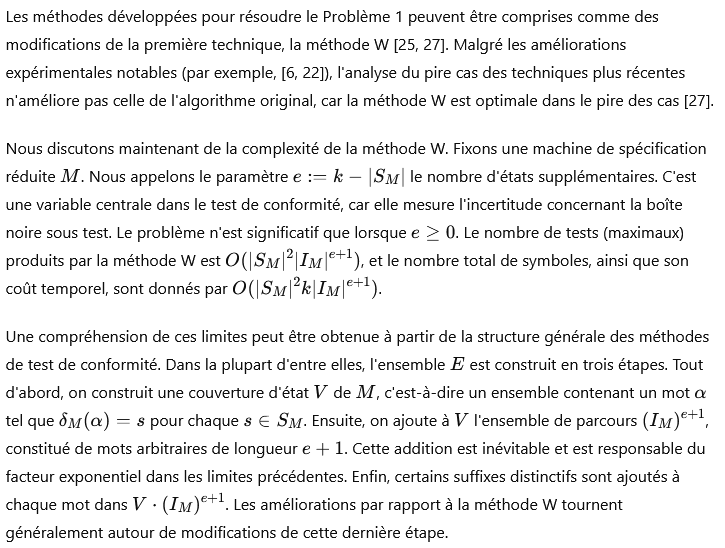


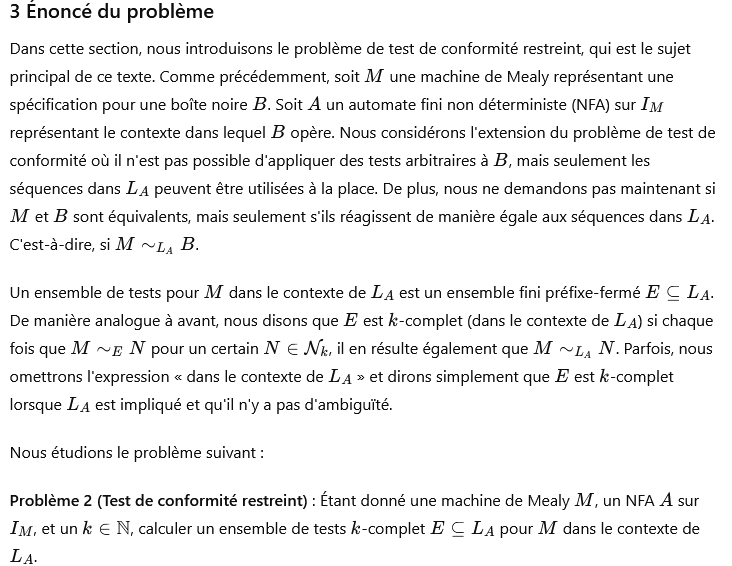






Il y a trois paramètres principaux à optimiser dans ce problème : le temps d'exécution, le nombre de tests maximaux dans l'ensemble E, et le nombre de symboles. Les deux derniers objectifs sont importants car un ensemble peut être utilisé sur plusieurs systèmes black box après sa construction, ou ces systèmes black box peuvent être lents à exécuter. Ainsi, pour certaines applications, il peut être intéressant de développer un algorithme plus lent qui aboutit à des ensembles plus petits. Nous adoptons la convention selon laquelle les ensembles produits par les algorithmes de test de conformité sont retournés en listant leurs tests maximaux. Dans ces circonstances, le coût temporel de tels algorithmes est trivialement limité par la longueur totale des ensembles qu'ils construisent.





Comme mentionné dans l'introduction, notre motivation pour cette tâche réside dans le problème de test en boîte grise où le composant soumis à l'essai a des interfaces observables, mais des entrées incontrôlables. Au cours de cet article, nous nous concentrons sur le cas particulier suivant.

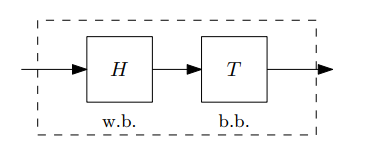
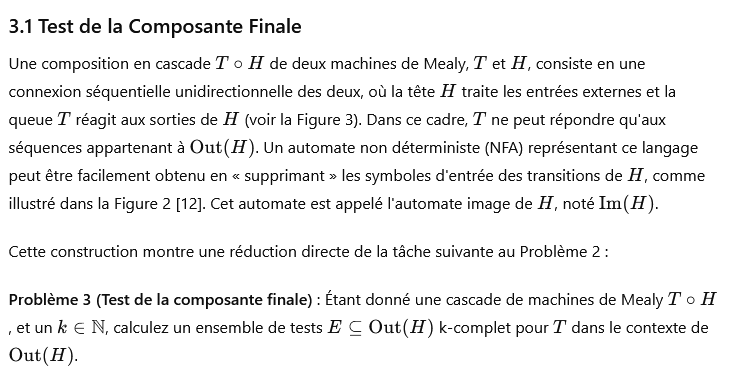
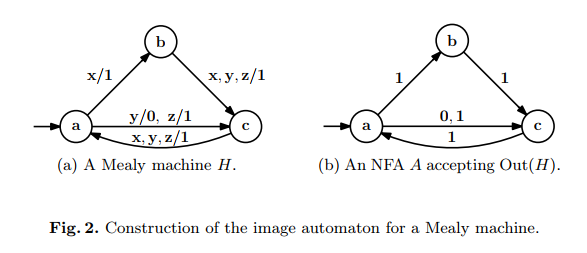


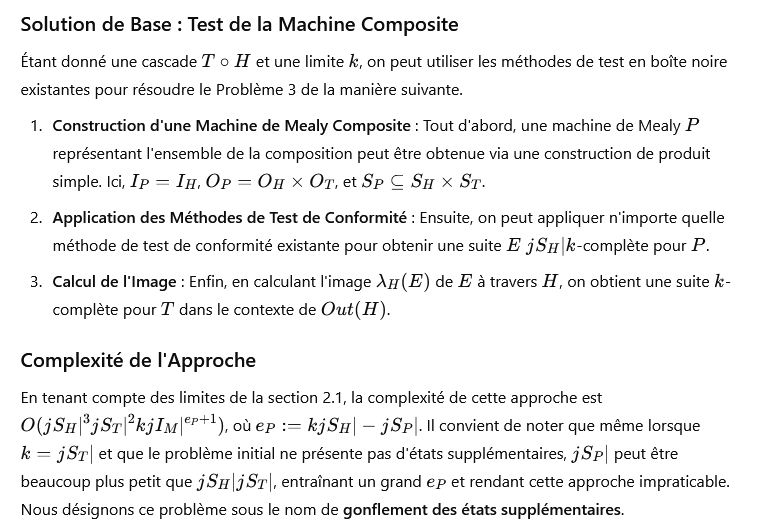
Fig. 1. Une composition en cascade de machines de Mealy





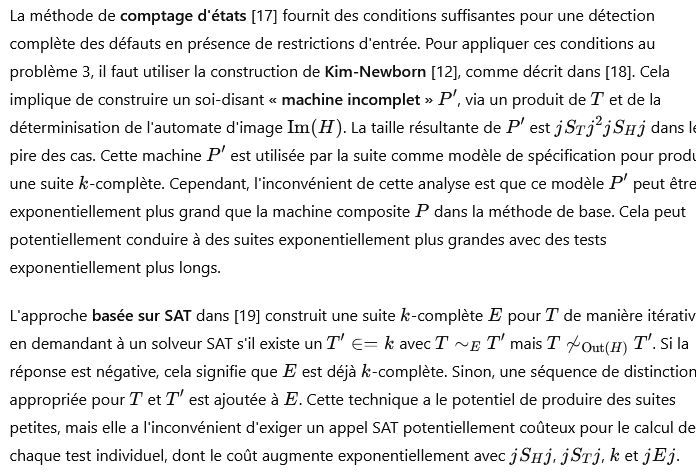
Pour simplifier la discussion, nous utiliserons ce cas particulier de test de composant pour motiver notre problème principal. Cependant, des formes plus générales de test de composant, où les interfaces sont observables, peuvent également être abordées via le Problème 2, car il existe des réductions polynomiales transformant ces scénarios en compositions en cascade [28, 14].

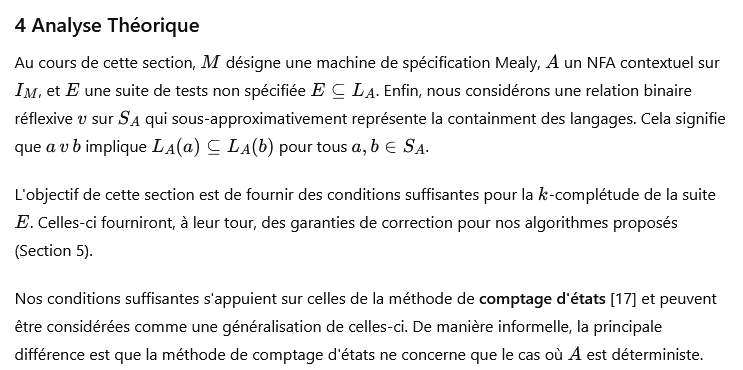
Nous allons maintenant donner un aperçu des solutions existantes pour le problème de test de la composante finale.

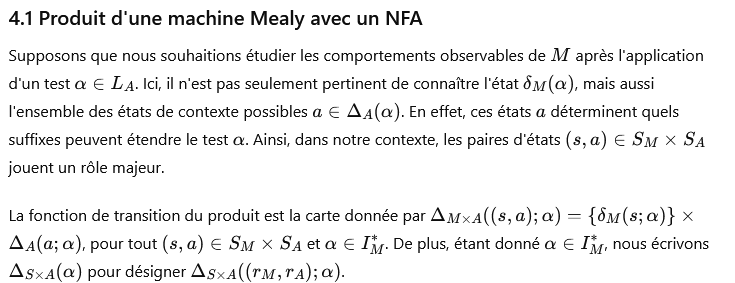


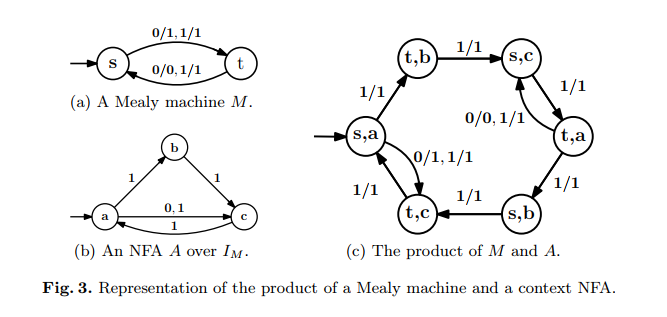
**Travaux Connexes** À ce jour, il existe deux approches principales proposées pour le problème de test de la queue, qui visent à surmonter le gonflement des états supplémentaires de la méthode précédente. Ce sont la **méthode de comptage d'états** [17] et une technique plus récente basée sur la **SAT** [19].

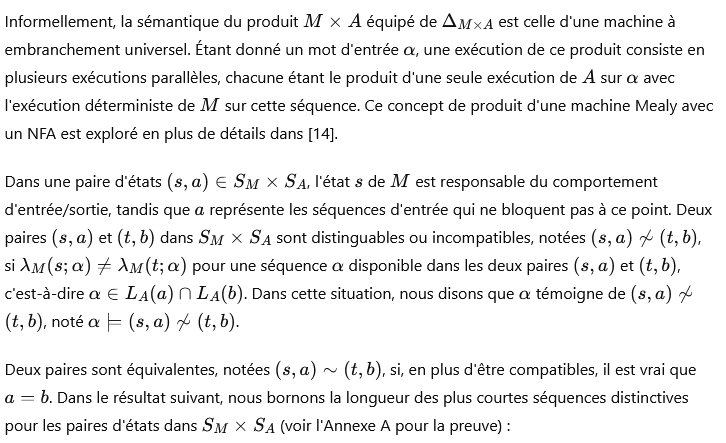
Cependant, chacune de ces techniques rencontre des problèmes importants dans leurs analyses de complexité, et il y a un manque de données expérimentales sur leurs performances en dehors de très petits exemples (compositions ne dépassant pas dix états au total).

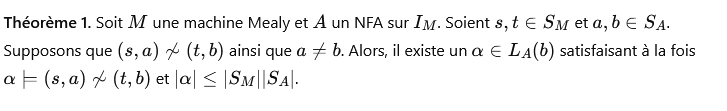


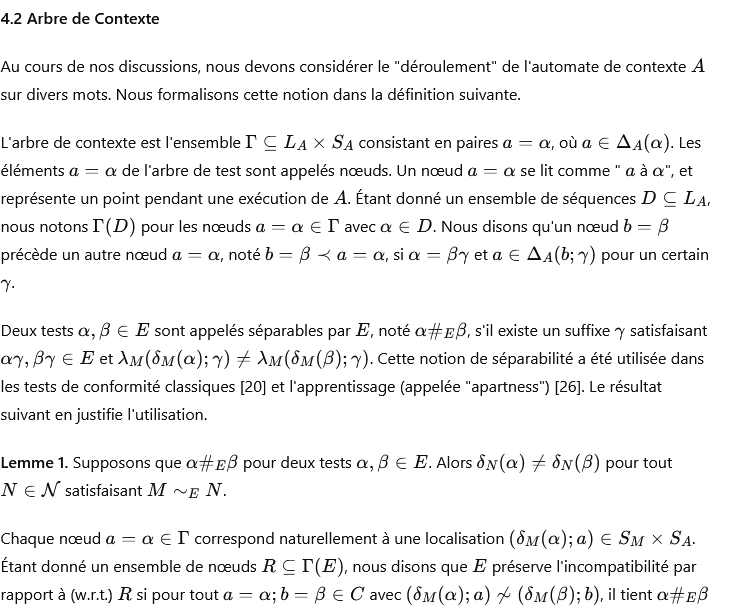


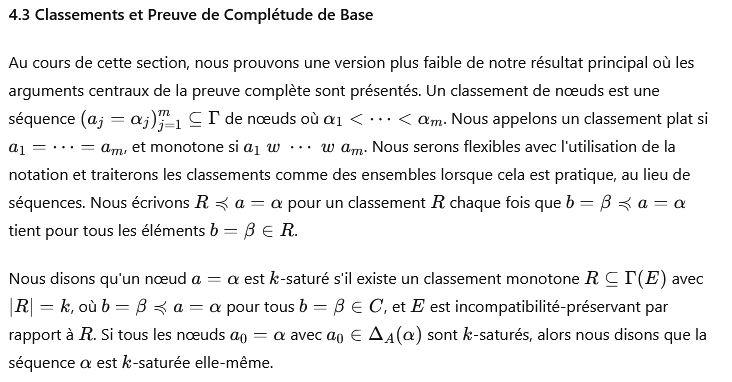


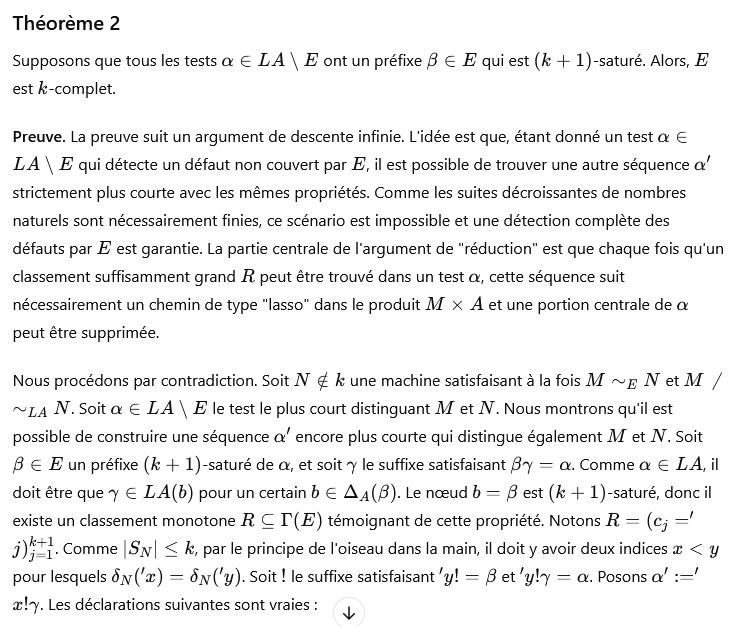


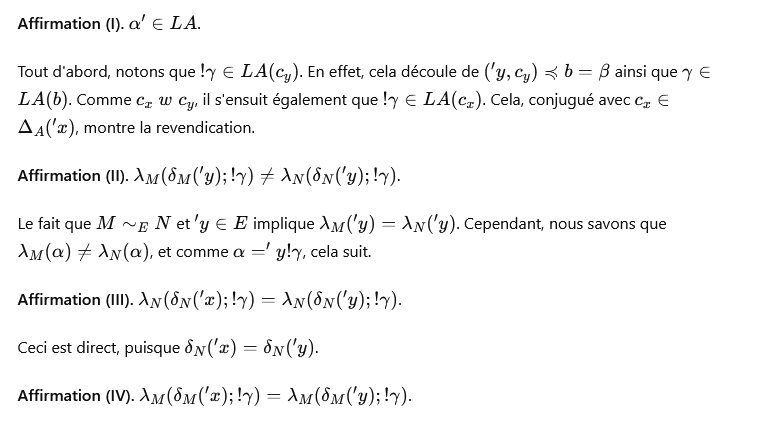


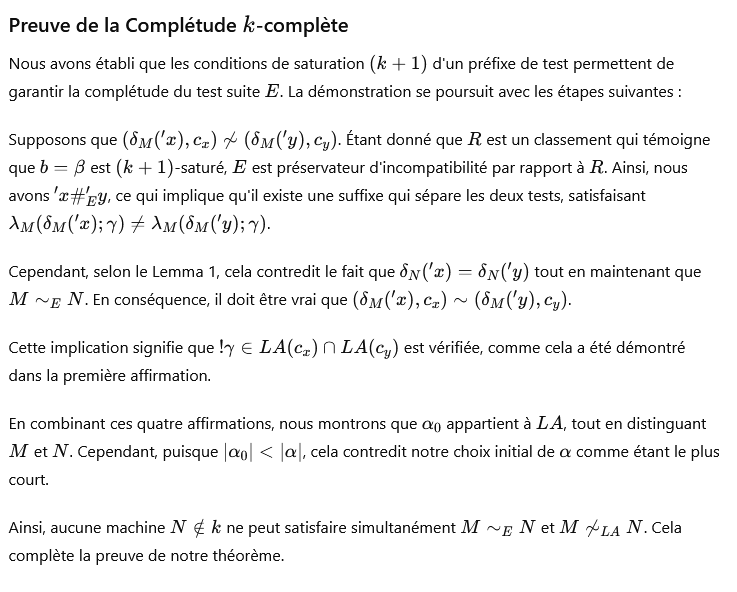


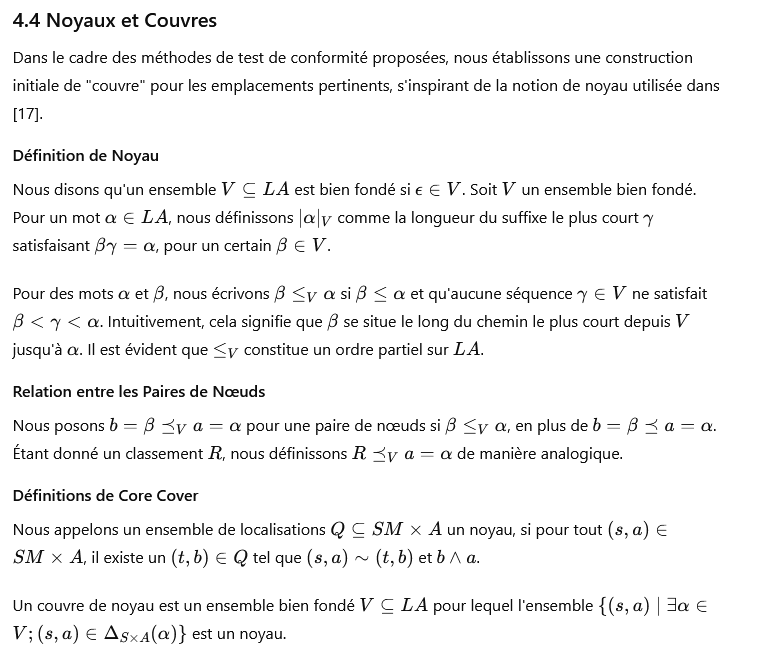


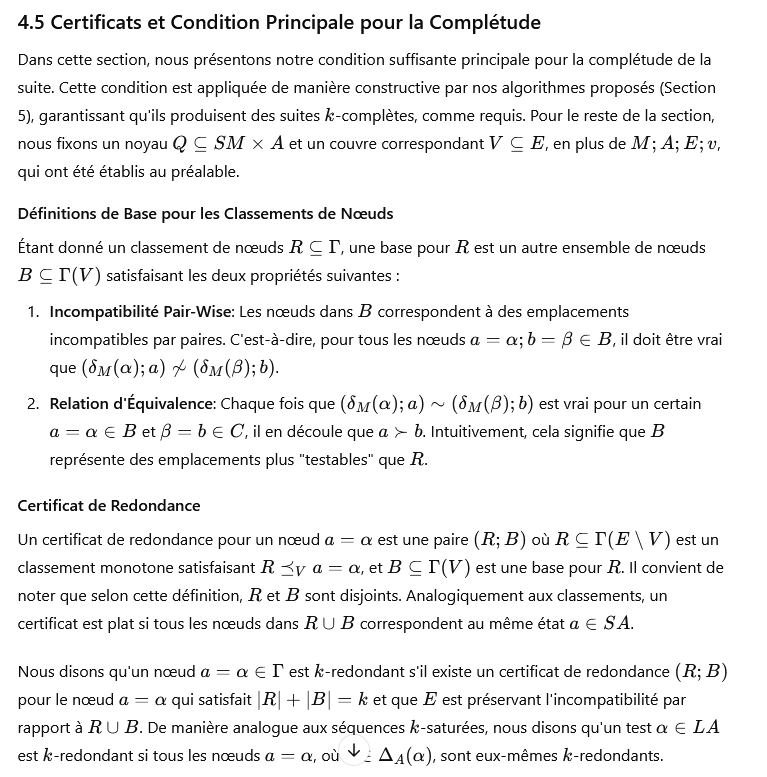


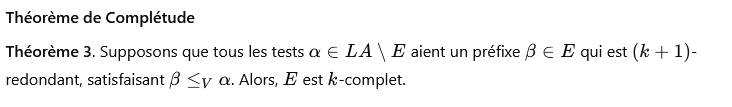


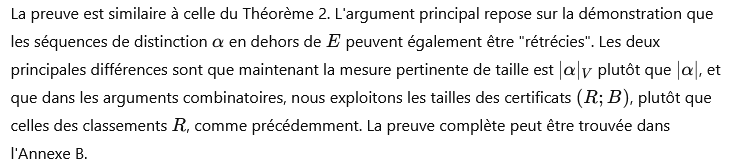


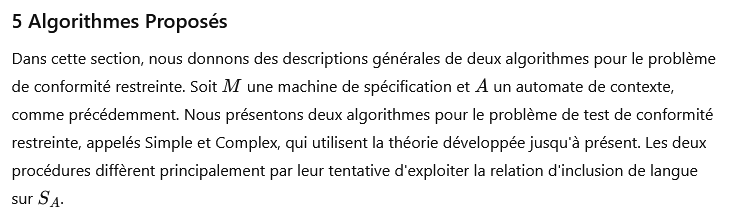


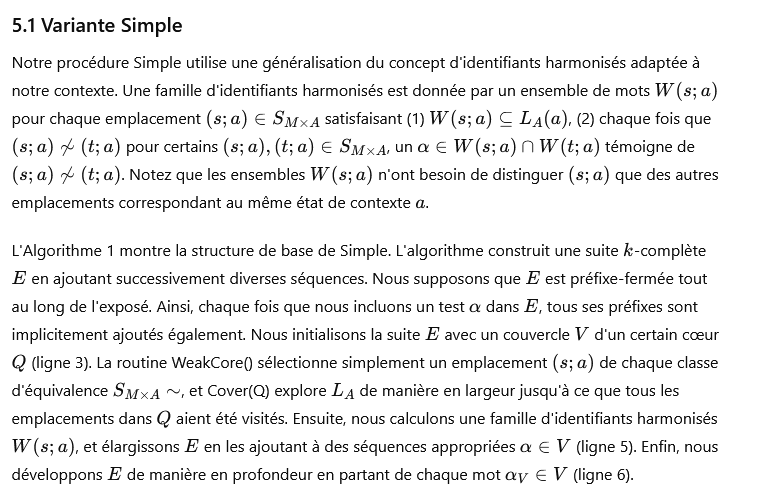




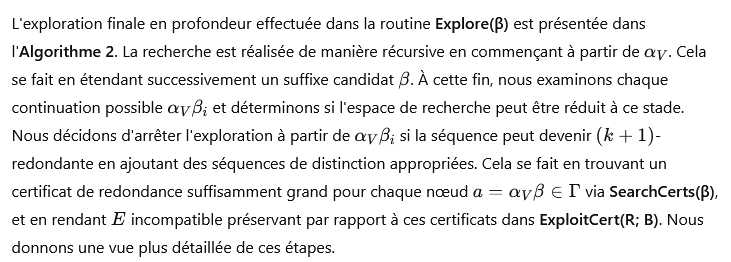


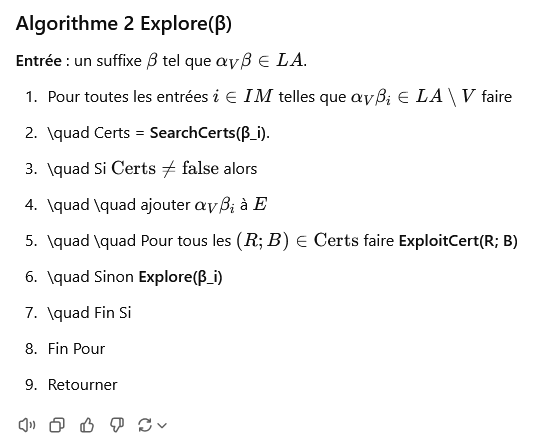


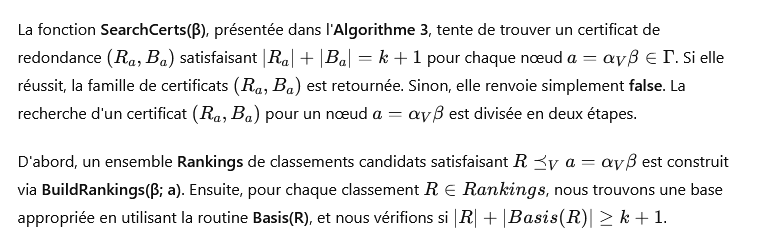


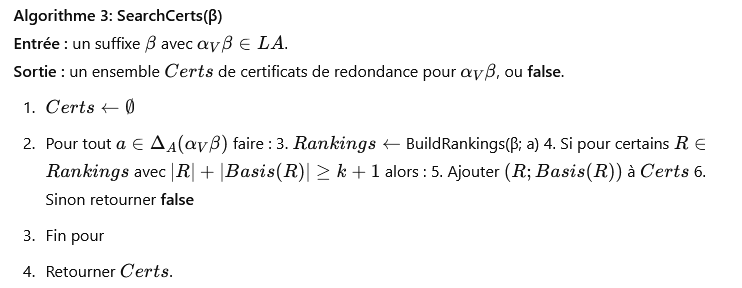


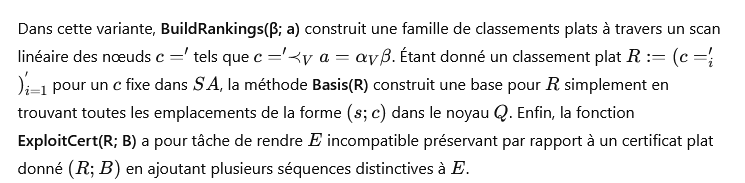


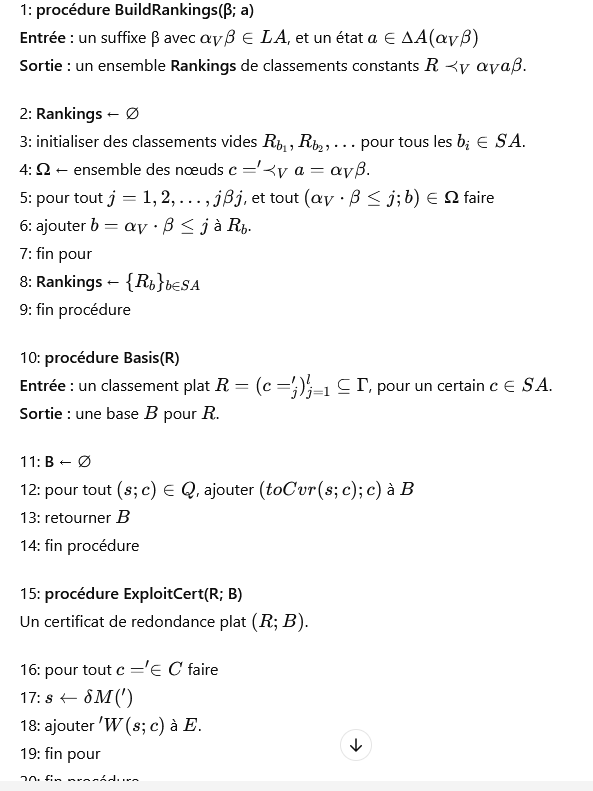


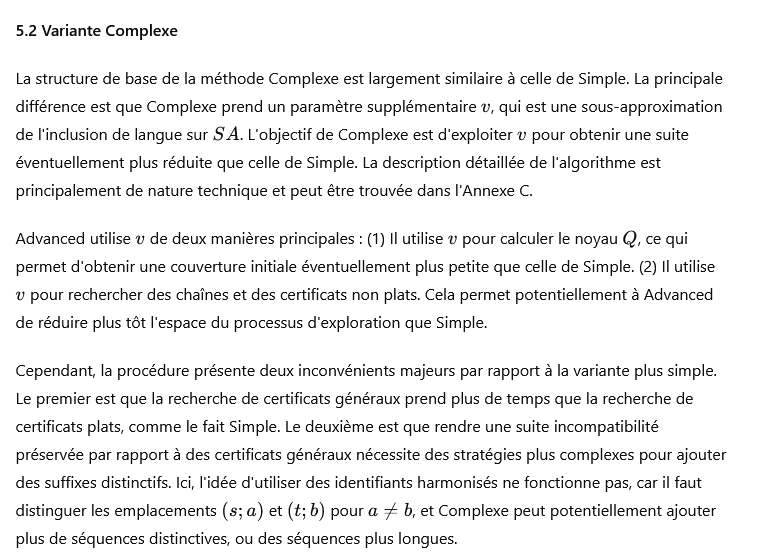


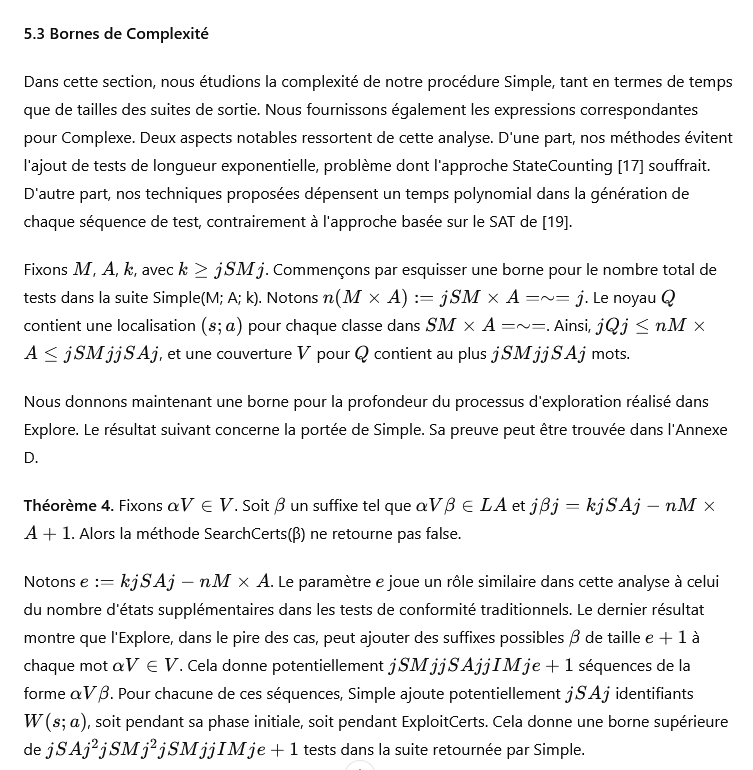


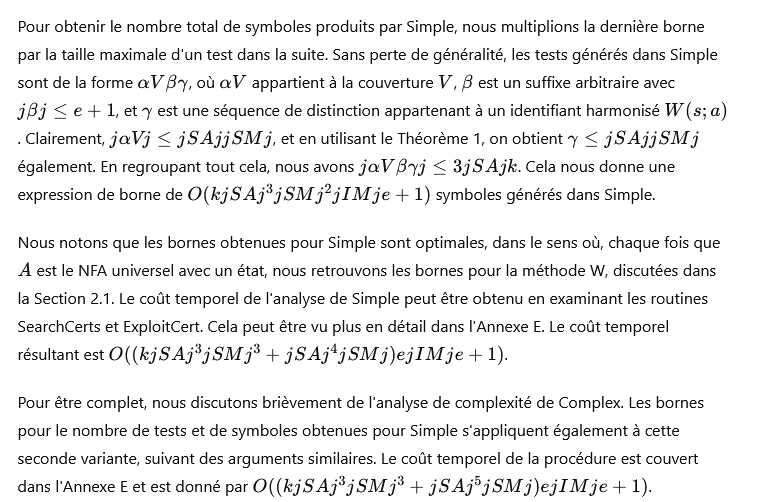


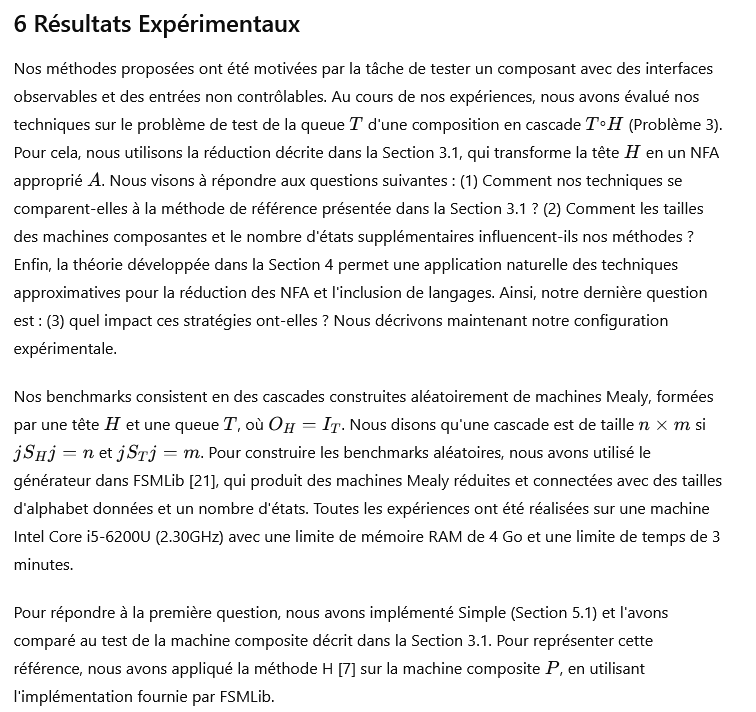


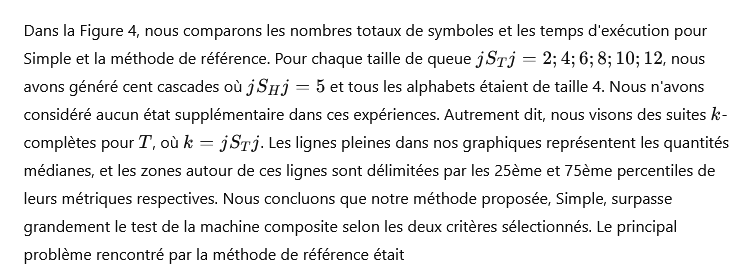


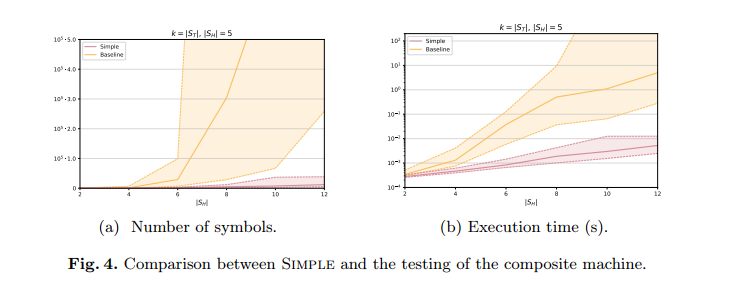




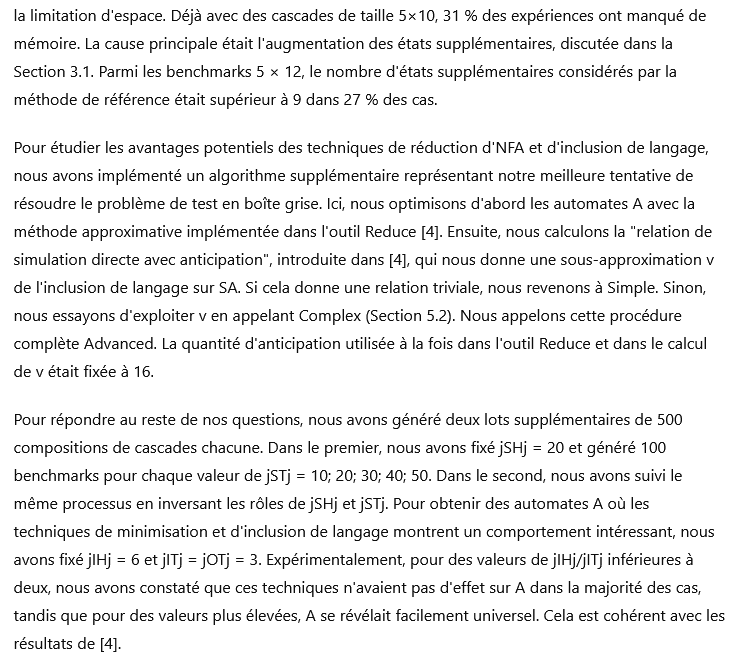


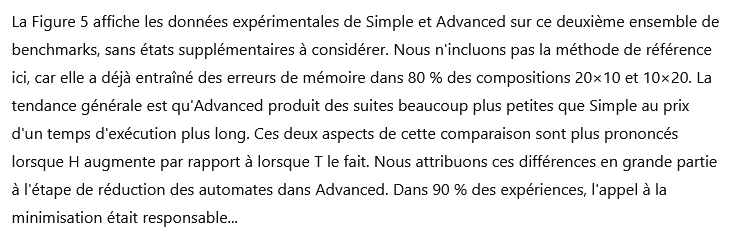






**Fig. 4.** Comparaison entre Simple et le test de la machine composite.





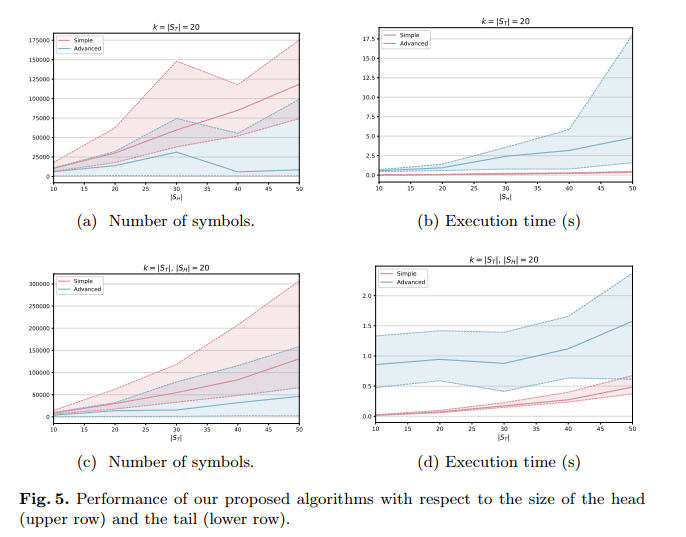


Fig. 5. Performance de nos algorithmes proposés en fonction de la taille de la tête (ligne supérieure) et de la queue (ligne inférieure).

