Génération de texte sous contraintes pour mesurer des performances de lecture : Une nouvelle approche basée sur les diagrammes de décisions multivalués

Alexandre Bonlarron, Aurelie Calabrese, Pierre Kornprobst, Jean-Charles Régin

Génération de texte sous contraintes pour mesurer des performances de lecture : Une nouvelle approche basée sur les diagrammes de décisions multivalués

A. Bonlarron[1,3], A. Calabrèse[2], P. Kornprobst[1], J.-C. Régin[3]

[1] Université Côte d'Azur, Inria, France
[2] Aix Marseille Univ, CNRS, LPC, Marseille, France
[3] Université Côte d'Azur, I3S, France

Alexandre.Bonlarron@inria.fr

Résumé

Mesurer les performances de lecture est l'une des méthodes les plus utilisées en clinique ophtalmologique pour juger de l'efficacité des traitements, des procédures chirurgicales ou des techniques de rééducation. Cependant, l'utilisation des tests de lecture est limitée par le faible nombre de textes standardisés disponibles. Pour le test MNREAD, qui est l'un des tests de référence pris comme exemple dans ce papier, il ne comporte que deux jeux de 19 phrases en français. Ces phrases sont difficiles à écrire car elles doivent respecter des règles de différentes natures (e.g., liées à la grammaire, la longueur, le lexique et l'affichage). Ils sont aussi difficile à trouver : Sur un échantillon de plus de trois millions de phrases issues d'ouvrages de la littérature jeunesse, seulement quatre satisfont les critères du test de lecture MNREAD. Pour obtenir davantage de phrases, nous proposons une approche originale de génération de texte qui prenne en compte l'ensemble des règles dès la génération. Notre approche est basée sur les Multi-valued Decision Diagrams (MDD). Nous représentons le corpus par des n-grammes et les différentes règles par des MDD, puis nous les combinons à l'aide d'opérateurs, notamment des intersections. Les résultats obtenus montrent que cette approche est prometteuse, même si certains problèmes demeurent comme la consommation mémoire ou la validation a posteriori du sens des phrases. En 5-gramme, nous engendrons plus de 4000 phrases qui respectent les critères MNREAD et proposons ainsi facilement une extension d'un jeu de 19 phrases au test MNREAD.

Mots-clés

Génération de texte, texte standardisé, tests de lecture, test MNREAD, basse vision, diagrammes de décisions multivalués, contraintes.

Abstract

Measuring reading performance is one of the most widely used methods in ophthalmology clinics to judge the effectiveness of treatments, surgical procedures, or rehabilitation techniques. However, reading tests are limited by the small number of standardized texts available. For the MNREAD test, which is one of the reference tests used as an example in this paper, there are only two sets of 19 sentences in French. These sentences are challenging to write because they have to respect rules of different kinds (e.g., related to grammar, length, lexicon, and display). They are also tricky to find : out of a sample of more than three million sentences from children's literature, only four satisfy the criteria of the MNREAD reading test. To obtain more sentences, we propose an original approach to text generation that considers all the rules at the generation stage. Our approach is based on Multi-valued Decision Diagrams (MDD). First, we represent the corpus by n-grams and the different rules by MDDs, and then we combine them using operators, notably intersections. The results obtained show that this approach is promising, even if some problems remain, such as memory consumption or a posteriori validation of the meaning of sentences. In 5-gram, we generate more than 4000 sentences that meet the MNREAD criteria and thus easily provide an extension of a 19-sentence set to the MNREAD test.

Keywords

Text generation, standardized text, reading tests, MNREAD test, low vision, multivalued decision diagrams, constraints.

1 Introduction

Les performances de lecture sont devenues l'une des mesures cliniques les plus utilisées pour juger de l'efficacité des traitements, des procédures chirurgicales ou des techniques de rééducation [22, 9]. Ces performances de lecture sont mesurées à partir du temps nécessaire pour lire des textes standardisés, i.e., conçus pour être équivalents en termes de longueur, d'affichage et de linguistique. Ce besoin d'avoir des textes équivalents est la clé pour éviter tout biais lié aux textes eux-mêmes. D'autre part chaque texte ne doit être présenté qu'une seule fois aux sujets ou patients, ceci pour éviter d'introduire un biais de mémorisation et assurer une mesure précise.

Parmi les différents tests existants [19], le test MNREAD [11, 12] est probablement l'un des tests de lecture standardisés les plus utilisés au monde pour mesurer les performances de lecture dans des contextes cliniques mais aussi de recherche, chez les personnes ayant une vision normale ou faible (basse vision [3]). Un test MNREAD est composé de phrases standardisées imprimées en 19 tailles par pas de 0,1 logMAR (une unité angulaire qui permet de quantifier l'acuité visuelle). L'objectif est d'évaluer comment la performance de lecture dépend de la taille de la police [4]. Il est disponible en 19 langues, avec un nombre de tests variable en fonction de la langue (e.g., cinq anglais, deux en français). Étant donné que des mesures répétées sont nécessaires dans de nombreuses applications de MNREAD, les communautés scientifiques et médicales ont besoin d'un grand nombre de jeux de tests, notamment pour éviter les biais de mémorisation. Ceci est impossible aujourd'hui car le nombre de phrases MNREAD est largement insuffisant.

Cependant, accroître le nombre de phrases MNREAD est un problème difficile à cause des règles multiples que les phrases doivent respecter. Ecrire ces phrases ou les trouver dans des corpus n'est clairement pas la solution car elles sont très spécifiques. Pour résoudre ce problème, qui est un problème dominé par des règles, une première idée serait d'utiliser des méthodes d'apprentissage profond (e.g., GPT, Bert) qui sont en plein essor [6]. Cependant, elles pourraient s'avérer difficile à mettre en oeuvre dans notre cas pour les deux raisons suivantes : (i) elles nécessitent de larges bases d'apprentissage, mais nous ne disposons que d'un faible nombre de phrases MNREAD d'origine, et (ii) à notre connaissance, l'introduction de règles (contraintes) dans les réseaux profonds pose des limitations, mais c'est un sujet qui évolue vite actuellement [14, 20].

Une autre idée est de concevoir des approches plus ad-hoc dédiées aux tests de lectures [5, 15, 13]. En particulier, dans [13], les auteurs, qui sont aussi les créateurs du test MNREAD, proposent une approche basée sur des modèles de phrases (*templates*) et capable de générer des millions de phrases en procédant à une marche aléatoire. Cependant, cette méthode semi-automatique présente plusieurs inconvénients majeurs : (i) elle repose sur des modèles de phrases qui doivent être créés manuellement (i.e., des séquences d'espaces réservés, chacun contenant une liste de mots possibles qui s'intègrent dans la phrase à ce moment-là) ; (ii) elle fonctionne en deux étapes (i.e., la création de la phrase suivie de la sélection de la phrase) impliquant des calculs supplémentaires et un temps d'exécution plus long ; (iii) elle ne peut pas être étendue facilement à d'autres langues, comme par exemple pour le français où l'on doit tenir compte des accords et conjugaisons.

Pour remédier à ces limitations, nous proposons dans cet article de modéliser le problème de génération de textes standardisés comme un problème d'optimisation sous contraintes. L'objectif est d'avoir une approche plus automatique (sans besoin de définir de modèle de phrase *a priori*), qui produisent uniquement des phrases respectant les règles (sans avoir à les vérifier *a posteriori*), et enfin qui puissent être généralisable (i.e., avec d'autres règles ou dans d'autres langues). Pour cela, nous proposons une approche originale basée sur les diagrammes de décision multivalués (en anglais *multi-valued decision diagrams*, MDD). Les MDD ont déjà été utilisés avec succès dans de nombreux autres problèmes comme la génération de musique [21] ou de poèmes [17]. D'autres utilisations des MDD pour des problèmes d'optimisations peuvent être consultés dans l'ouvrage de référence de Bergman et al. [2], ou dans la thèse plus récente de Perez [16]. Les MDD sont une généralisation des diagrammes de décision binaire (BDD) [1]. Ce sont des structures de données permettant de calculer et stocker des solutions (des n-uplets) d'un problème à l'aide d'un graphe orienté acyclique (DAG). Utiliser les MDD dans notre contexte offre trois avantages principaux : (i) Les MDD permettent de représenter un grand nombre de solutions dans une structure compressante, ce qui va être crucial étant donné la taille des données que nous allons devoir manipuler, (ii) Les MDD permettent de décomposer le problème en définissant autant de MDD que de règles que le texte doit respecter. (iii) Il est possible d'effectuer efficacement des opérations entre MDD permettant ainsi de combiner différentes règles [18].

Pour sa construction, un MDD dans sa forme ordonné est structuré en couches où chaque couche représente une variable ordonnée X_i. Il y a deux noeuds particulier dans le MDD : la racine (`root`) et le noeud true terminal (`tt`). Chaque chemin entre le noeud `root` et `tt` forme un n-uplets de label qui correspond à une affectation valide des variables associées à chaque couche. Le label l_{a_k} de l'arc a_k appartenant à un chemin quelconque entre la couche $i-1$ et i est l'affectation de la variable X_i tel que $X_i = l_{a_k}$. Un chemin $p = (a_1, a_2, a_3, \ldots, a_r)$ existe dans le MDD si et seulement si l'assignation des variables X_i=l_{a_i} est valide pour tout i. Ainsi calculer un MDD contenant r couches revient à calculer les r-uplets valides de la fonction $f : \{0...d\}^r \mapsto \{true, false\}$. Un exemple de MDD de type somme est donné dans la Fig. 1.

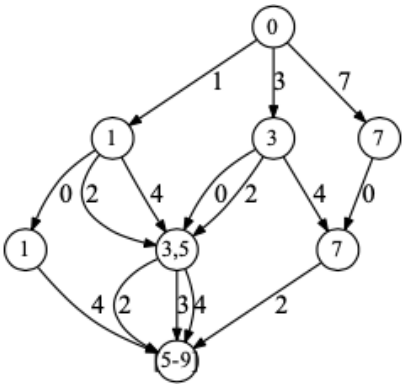

FIGURE 1 – Exemple de MDD représentant l'ensemble de $x_1 + x_2 + x_3 \in [5, 9]$. Pour chaque variable x_i, le domaine $D(.)$ est $D(x_1) = \{1, 3, 7\}$, $D(x_2) = \{0, 2, 4\}$, $D(x_3) = \{2, 3, 4\}$. Par exemple, $(7, 0, 2)$ appartient à l'ensemble des solutions appartenant au MDD.

Le plan de cet article est le suivant. Dans la section 2 nous montrons comment modéliser le problème de génération de texte sous contraintes avec des MDD. Pour l'illustrer, on se focalise dans ce papier sur la génération de phrases de type MNREAD dont on commencera par rappeler les règles MNREAD, puis on expliquera comment construire les MDD associés, et enfin comment on peu les intersecter. Les résultats sont présentés dans la section 3 dans laquelle on montre le potentiel de notre méthode. Nous terminons en section 4 par une discussion.

2 Méthode

2.1 Caractérisation des phrases MNREAD : définition des règles

Afin d'être standardisées, toutes les phrases MNREAD doivent obéir à cinq règles $(\mathcal{R}_i)_{i=0..4}$ explicitées par les créateurs du test (voir [13] pour plus de détails) :

- règles grammaticales : des phrases en forme déclarative, sans ponctuation, pas de noms propres ($\mathcal{R}_0$)
- règles lexicales : des phrases construites uniquement à partir des 3000 lemmes (forme canonique) des mots les plus fréquents dans les manuel de niveau CE2, comme défini dans le lexique Manulex [10] ($\mathcal{R}_1$)
- règles de longueur : des phrases 9 à 15 mots ($\mathcal{R}_3$) et 59 caractères avec les espaces et sans compter le point final ($\mathcal{R}_2$)
- règles d'affichage : des phrases qui peuvent s'afficher sur trois lignes, avec une justification gauche-droite et des espace inter-mots compris entre des valeurs seuils strictes, proportionnelles à la taille de l'espace standard pour la police donnée ($\mathcal{R}_4$).

Un exemple de phrase MNREAD respectant ces règles est donné dans la Fig. 2.

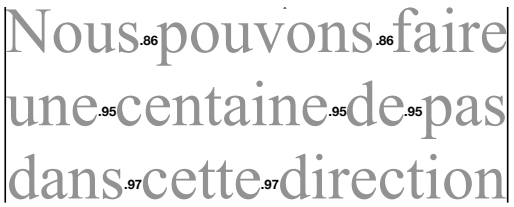

FIGURE 2 – Affichage d'une phrase MNREAD type respectant les contraintes $\mathcal{R}_3 \ldots \mathcal{R}_4$. Chaque phrase du test est affichée avec une justification à gauche et à droite sur trois lignes de texte qui tiennent dans une boîte dont la largeur est égale à 17,3 fois la taille d'une lettre standard en police Times-Roman. La largeur des espaces entre les mots (indiquée comme multiple de la largeur normale des espaces) doit être comprise entre 0,80 et 1,25 (adapté de [13]).

Les règles $(\mathcal{R}_i)_{i=0..4}$ sont nécessaires mais non suffisantes pour appartenir au test. En effet, remarquons qu'il y a également des règles implicites, plus qualitatives et plus difficile à formaliser mais qui semblent "évidentes", comme des règles sémantiques (la phrase doit avoir un sens et respecter la congruence) ou syntaxiques (la phrase doit avoir une structure "simple"). Une phrase considérée de type MNREAD sera donc une phrase qui respecte toutes les règles précédemment énoncées, qui a du sens et enfin qui ressemble aux phrases officielles du test.

2.2 Définition d'un ensemble de n-grammes

Dans l'esprit, les MDD vont décrire tous les chemins possibles entre mots pour construire des phrases. Cependant, afin d'avoir plus de chance d'obtenir des phrases dont la syntaxe est correcte, l'idée est de ne considérer que des recombinaisons de bouts de phrases existantes. Plus précisement, étant donné un corpus constitué d'un ensemble de phrases écrites par des auteurs (dont la syntaxe et le sens sont donc corrects), l'idée est de les décomposer en n-grammes (sous-séquence de n mots construite à partir d'une phrase donnée) que l'on va chercher à recombiner tout en respectant les règles MNREAD.

En pratique, comme il est fait classiquement, nous allons créer un dictionnaire de n-grammes à partir d'un corpus constitué d'un ensemble de livres. Ce qui fait la spécificité de notre problème, c'est que nous allons ignorer un certain nombre de phrases (et donc de n-grammes) afin de prendre en compte dès le début les règles grammaticales $\mathcal{R}_0$ (e.g., les phrases interrogatives vont être ignorées, ceci afin de ne pas rajouter des n-grammes correspondant à des tournures interrogatives qui ne sont pas souhaitées dans notre application). Il y a ensuite une série de transformations à l'aide d'expressions régulières qui peuvent être appliquées aux phrases du corpus pour obtenir davantage de n-grammes utilisables (e.g., les points de suspensions, deux points et point-virgules sont considérées comme des points finaux ; les guillemets sont ignorés). Enfin, une série de réécriture est appliquée aux abréviations de statuts usuels de manière à retrouver leurs formes non abrégées (e.g., M. est réécrit Monsieur).

2.3 Construction des MDD : des règles aux contraintes

Pour résoudre notre problème, il s'agit de formaliser les règles MNREAD dans le paradigme MDD. Chaque MDD va résoudre un sous-problème correspondant à une règle. Ensuite, il s'agira d'effectuer des opérations (intersections) entre ces MDD pour obtenir les solutions finales. Pour cela, tous les MDD doivent avoir la même taille, i.e., le même nombre de couches, ce qui correspond aux nombres de mots que notre solution doit comporter. Les labels des arcs seront des informations en rapport avec les mots. Cette section présente comment chaque MDD est construit. La Fig. 3 montre une représentation de ces MDD dans un cadre simplifié.

MDD_U (associé à tous les mots du corpus). Ce MDD est le MDD universel associé au corpus. Ce MDD contient toutes les séquences de mots d'une taille 9 à 15 qui peuvent être écrites à partir des mots appartenant au corpus. L'ensemble des solutions qu'il contient est égal au produit car-

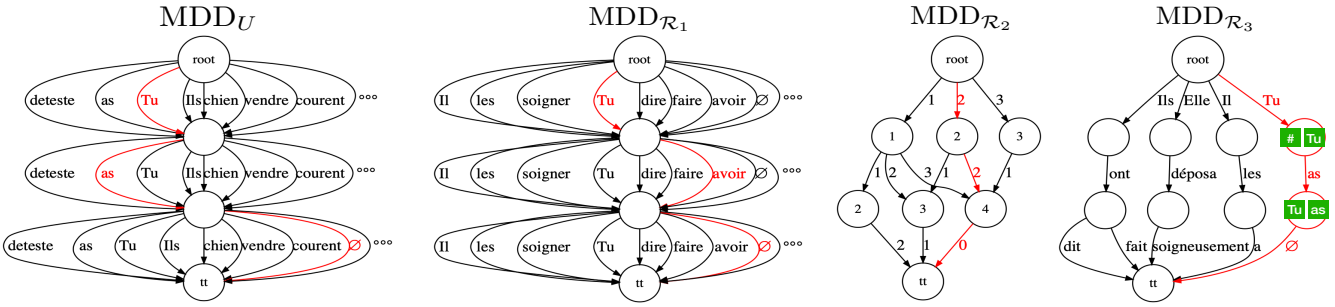

FIGURE 3 – Illustrations des différents MDD dans un cadre simplifié avec quatre couches de noeuds (ceci à des fins d'illustration). Le MDD$_{\mathcal{R}_4}$ n'est pas représenté car il y aurait besoin de plus de couches pour en montrer le principe (en particulier la réinitialisation de la somme), mais conceptuellement il suit le même principe que le MDD$_{\mathcal{R}_2}$. Dans chaque MDD, un chemin est indiqué en rouge. Ce chemin correspond à une même phrase. Cette phrase sera donc une solution pour notre problème, comme résultats de l'intersection des MDD. Remarquons que pour des raisons de lisibilité, les états de MDD$_{\mathcal{R}_3}$ n'ont été représentés que pour la solutions mise en évidence, en supposant que l'on utilise des 2-grammes (en vert).

tésien des domaines de toutes les couches. On complétera ces séquences avec le mot vide, le mot qui ne peut être suivi que par le mot vide. Ainsi, toutes les séquences ont une taille 15. Dans ce MDD, il n'y a pas d'**état** et chaque **arc** est un mot.

MDD$_{\mathcal{R}_1}$ (associé à $\mathcal{R}_1$, des phrases utilisant un sous-ensemble de lemmes). Ce MDD va nous servir à modéliser le vocabulaire du lexique autorisé.

Dans ce MDD, il n'y a pas de dépendance entre les variables pour l'affectation des valeurs. Il n'y a donc pas de nécessité de définir une notion d'état qui permettrait de statuer sur la validité de l'existence des arcs. Ce MDD a donc $r + 1$ noeuds pour r variables.

Un **arc** correspond à un lemme appartenant au vocabulaire autorisé (Manulex). Chaque variable est libre de prendre n'importe quelle valeur du domaine. Dans la couche i, entre le noeud de la couche i et $i + 1$, il y a 3000 arcs sortant, i.e., un arc pour chaque mot du lexique.

Le MDD associé à $\mathcal{R}_1$ est très facile à construire. Il ne peut pas être réduit. Le MDD$_{\mathcal{R}_1}$ est un MDD universel. Une solution de ce MDD est une suite de lemmes d'une taille égale au nombre de couches du MDD : entre 9 et 15.

Nous faisons appel à la bibliothèque treetagger pour passer d'un mot à son lemme [23]. Remarquons que ce travail consistant à restreindre le vocabulaire des phrases avec un lexique aurait pu être réalisé au niveau des pré-traitements sur le corpus. L'intérêt de vérifier $\mathcal{R}_1$ avec un MDD se justifie dans notre méthodologie car nous voulons profiter de la modularité et de la robustesse de l'approche MDD pour ainsi être en mesure de changer à volonté le lexique autorisé des phrases. Remarquons que, MDD$_{\mathcal{R}_1}$ n'est pas nécessairement un sous-ensemble de MDD$_U$ car ils sont construits à partir de données différentes (au niveau corpus) et la nature des données qu'ils contiennent est aussi différentes. Le MDD$_{\mathcal{R}_1}$ contient seulement des lemmes (e.g., masculin singulier pour un nom, infinitif pour un verbe) construits à partir du lexique Manulex [10]. A l'inverse, le corpus peut lui être construit à partir de livre très spécifiques. En effet, un livre (même jeunesse) sur les volcans risque de ne pas utiliser le mot "tomate" qui appartient bien pourtant au lexique Manulex, et *a fortiori* au MDD$_{\mathcal{R}_1}$.

MDD$_{\mathcal{R}_2}$ (associé à $\mathcal{R}_2$, des phrases de 59 caractères). Ce MDD est le MDD responsable de modéliser les phrases dont la somme des caractères des mots est égale à 59 caractères. Cela revient à imposer une contrainte de somme [24] sur les phrases. Nous voulons definir un MDD dont la somme des labels des arcs qui composent toutes solutions est égale à 59. C'est un MDD de type somme (voir l'exemple de la Fig. 1).

Un **état** est représenté par un entier s égal à la valeur de la somme partielle de tous les arcs appartenant aux chemins qui mènent au noeud associé à l'état. Un état est valide si et seulement si de la couche 0 à $r - 1$ si $0 \leqslant s \leqslant 59$, et si à la couche r, $s = 59$; Un **arc** est le nombre de caractères d'un mot. Une solution du MDD est une suite de 15 entiers ayant pour somme 59 caractères.

MDD$_{\mathcal{R}_3}$ (associé à $\mathcal{R}_3$, des phrases de 9 à 15 mots). Ce MDD est le MDD de base qui fait le lien avec le dictionnaire de n-grammes. Il va contenir toutes les combinaisons de mots et indirectement de n-grammes possibles conduisant à produire des phrases.

Un **état** est un n-gramme. Dans notre construction, les n-grammes des premières et dernières couches doivent correspondre à des n-grammes de début et fin de phrase. Ceci permet d'éliminer une partie des phrases n'aboutissant pas et une partie de celles qui laissent ouvertement sur leur fin.

Un **arc** est un mot, un n-gramme dans un état est donc un historique des n derniers arcs qui appartenant au chemin entre `root` et le noeud ayant pour état un n-gramme donné. Le mot vide est un mot possible si il ne peut être suivi que par le noeud `tt` ou un mot vide. Cette astuce permet soit de construire un MDD contenant des phrases de taille différente mais aussi à ajouter des couches de manières à ce que tout les MDD fassent la même taille pour une composition.

MDD$_{\mathcal{R}_4}$ (associé à $\mathcal{R}_4$, affichage des phrases). Ce MDD est aussi un MDD de type somme comme le MDD$_{\mathcal{R}_2}$. L'idée est ici aussi de modéliser une somme sur les éléments des phrases, mais cette fois-ci sur la taille (la largeur) des caractères au sens d'une police de caractère définie.
Initialement, un **état** est représenté par un entier s la valeur de la somme partielle de tous les arcs appartenant aux chemins qui mènent au noeud associé à l'état. On peut aussi définir un intervalle de somme valide dans lequel un état doit appartenir pour modéliser la taille des espace appartenant à un intervalle défini. Il y a un noeud entre chaque paire d'arcs, un arc correspond à un mot dans notre modèle. C'est au niveau du noeud, et donc dans l'état que les espaces doivent être gérés.
Un **arc** est la taille d'un mot dans sa police d'écriture.
Dans $\mathcal{R}_4$, il y a trois lignes, ce qui signifie qu'il faut en réalité vérifier trois sommes différentes, une pour chaque ligne, pour ce faire on ajoute à la définition d'un état, trois variables booléenne, pour connaître quelle est la ligne qui est entrain d'être calculée. Un état est valide de 0 à r-1 si la somme s est inférieur à taille maximale d'une ligne avec des petits espacements. De plus, si la taille des espaces appartient à l'encadrement des espaces de $\mathcal{R}_4$ l'état est valide, la variable booléenne associée à la ligne courante devient vrai et enfin la somme est remise à zéro. Au niveau de la couche r, il faut aussi vérifier que les variables booléennes associées à la ligne 1 et 2 soient vraies.

2.4 Opérations entre MDD : calcul des solutions

Pour obtenir les solutions qui respectent toutes les contraintes, on calcule le MDD final comme l'intersection à la volée des MDD définis précédemment. Cette opération d'intersection à la volée et toutes les autres opérations sur les MDD repose sur un algorithme associé à l'opérateur générique APPLY $\bigoplus$ défini dans [18, 7]. Tout l'intérêt de cette intersection à la volée est de permettre de réaliser cette opération de conjonction de contraintes efficacement sur les MDD.
Le MDD final MDD$_F$ se calcule de la façon suivante :

$$\begin{aligned}
\text{MDD}_{I_1} &= \text{MDD}_U \bigcap \text{MDD}_{\mathcal{R}_1},\\
\text{MDD}_{I_2} &= \text{MDD}_{I_1} \bigcap \text{MDD}_{\mathcal{R}_2},\\
\text{MDD}_{I_3} &= \text{MDD}_{I_2} \bigcap \text{MDD}_{\mathcal{R}_3},\\
\text{MDD}_F &= \text{MDD}_{I_3} \bigcap \text{MDD}_{\mathcal{R}_4}.
\end{aligned}$$

3 Résultats

Le modèle décrit dans la Section 2 est implémenté en Java 17 dans un solveur de MDD (MDDLib) développé au sein de l'équipe de recherche. Les expériences ont été réalisées sur une machine Ubuntu 18.04 utilisant un CPU Intel(R) Xeon(R) Gold 5222 @ 3.80GHz et 188 GB de RAM.

3.1 Constitution de corpus de n-grammes

Pour constituer notre corpus de n-grammes, nous sommes partis d'un ensemble de 658 livres appartenant à la catégorie jeunesse. Ce choix est motivé par le fait vouloir générer in fine des phrases de structure simple et utilisant un lexique simple (voir $\mathcal{R}_1$). Pour se convaincre de la pertinence de notre approche qui consiste à générer des phrases plutôt que d'en chercher des existantes, nous avons d'abord regardé combien de phrases étaient de type MNREAD. Sur les 3165037 phrases contenues dans le corpus brut, seulement quatre phrases étaient de bonnes candidates pour être des phrases MNREAD (voir Tableau 1).

Je couvris la tête de mes mains pour me protéger du serpent
Je lançai de grands coups de pied pour tenter de me libérer
Mais il n'y aura pas assez de place pour emmener le sorcier
Elle poussa un petit cri lorsque la chose bougea de nouveau

TABLE 1 – Les quatre phrases de type MNREAD trouvées dans le corpus d'origine de 658 livres.

Partant de ces phrases, nous avons suivi trois étapes pour définir notre corpus de n-grammes : (i) Les quatre phrases identifiées dans le corpus brut (Tab. 1) ont été retirées. L'idée est de partir d'un corpus ne contenant aucune phrase de type MNREAD pour éviter tout biais (garder ces phrases serait donner des bouts de solution valides) et pour montrer comment on peut générer des phrases "ex-nihilo" ; (ii) Seules les phrases respectant la règle $\mathcal{R}_0$ ont été retenues ; (iii) Les phrases retenues sont décomposées en n-grammes (pour un n donné). L'ensemble des n-grammes obtenus pour toutes les phrases retenues définit notre corpus de référence.
En pratique, nous allons aussi considérer des sous-ensembles de n-grammes obtenus à partir de pourcentages de phrases retenues. La relation entre le nombre de n-grammes (uniques) et le pourcentage de phrases utilisées pour les obtenir, est présentée dans la Fig. 4 et la Table 2. Par exemple, si l'on choisit 50% des phrases du corpus, cela nous donne 2.8 millions de 5-grammes. On peut y voir la relation non-linéaire entre le nombre de n-grammes et le pourcentage de phrases considérées. Plus le nombre de grammes augmente plus le nombre de n-grammes uniques est élevé. Enfin, ces résultats suggèrent que le nombre de n-grammes pourrait être facilement augmenté par l'accroissement de la taille du corpus (il n'y a pas "d'effet plateau" à ce stade).

3.2 Analyse des performances

Le Tableau 3 présente une analyse des performances de notre approche pour différentes valeurs de n-grammes, aussi bien du point de vue computationnel que du nombre de solutions obtenues. Les pourcentages de corpus pour chaque valeur de n-gramme ont été choisi de façon à obtenir un nombre de solutions du même ordre de grandeur afin de pouvoir les comparer.
Nous pouvons faire quelques observations générales : Le MDD$_{I_1}$ est un MDD facile à calculer, dans la mesure où il n'y a pas d'état, il s'agit de filtrer les valeurs des domaines de chaque couches.
Pour le MDD$_{I_2}$, cela revient à calculer un MDD somme, ce qui est habituellement assez rapide, mais il faut tout de

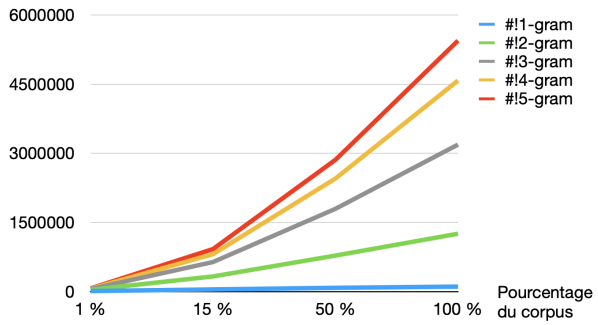

FIGURE 4 – Nombre de n-grammes uniques en fonction du pourcentage du corpus considéré et pour différents grammages.

	1%	10%	50%	100%
#!phrase	5947	59484	297405	594800
#!1-gram	11279	40004	81792	108231
#!2-gram	39747	244947	781699	1255730
#!3-gram	56254	451096	1796797	3189270
#!4-gram	63885	559297	2453949	4578657
#!5-gram	70069	630035	2860526	5444025

TABLE 2 – Caractéristiques du corpus : nombre de phrase, de mots (1-gram), de k-grammes ($k = 2..5$) uniques pour un pourcentage de corpus donné.

3-gramme avec 1% corpus					
MDD	nodes	arcs	sols	t	mem
I_1	16	91620	6.10^{56}	0m1s	2Go
I_2	712	3532517	$1,9.10^{42}$	0m23s	11Go
I_3	187079	104636	1601119	5m9s	36Go
F	11518	15928	9899	0m4s	2Go

4-gramme avec 15% corpus					
MDD	nodes	arcs	sols	t	mem
I_1	16	285975	2.10^{64}	0m1s	2Go
I_2	760	11403386	6.10^{44}	0h10m	12Go
I_3	356013	581504	2179577	2h47	39Go
F	18660	24387	10478	0m4s	2Go

5-gramme avec 100% corpus					
MDD	nodes	arcs	sols	t	mem
I_1	16	534225	1.10^{68}	0m1s	2Go
I_2	778	21639890	$1,3.10^{46}$	0h41m	16Go
I_3	529294	816078	1225086	10h6m	64Go
F	16459	13043	4335	0m2s	2Go

TABLE 3 – Analyse de performance. Pour chaque MDD intermédiaire (MDD_{I_i}) jusqu'au MDD final (MDD_F), ce tableau donne le nombre de noeuds, nombre d'arêtes, nombre de solutions, le temps de calcul de l'intersection, et la mémoire du processus. Trois résultats sont présentés avec différentes valeurs de n-grammes.

même dans le calcul de l'intersection à la volé : itérer sur le domaine du MDD précédant, pour une couche donné le cardinal du domaine est égale à (#arcs/#couches) de MDD_{I_1}. En effet, dans MDD_{I_1} tout les mots sont autorisés et ce pour toutes les couches à supposer qu'ils appartiennent au lexique.

Le MDD_{I_3} est long et coûteux en mémoire à calculer, c'est à ce moment là du calcul que nous pouvons dire que nous manipulons des phrases. Jusque là, c'était simplement des séquences de mots quelconques d'un nombre de caractères fixés et d'un vocabulaire restreint. En revanche, le MDD_F est très rapide à calculer, (quelques secondes à chaque fois) et peu coûteux en mémoire. Cependant, ceci est contextuel à sa position dans l'intersection. Le plus gros du calcul a déjà était effectué dans les intersections précédentes, un grand nombre d'arcs formant des solutions non valides ont déjà été filtrées dans le calcul de MDD_{I_3}.

Plus précisément, pour le MDD_{I_3}, cette croissance de la mémoire et du temps de calcul s'explique par deux raisons : (i) Nous calculons un grand nombre de phrases inutiles, car nous devons attendre d'atteindre les dernières couches pour vérifier qu'elles disposent d'un k-gramme de fin phrase. C'est à ce moment que nous savons si le chemin construit (i.e., la phrase) sera relié à `tt`. Remarquons que le nombre de n-grammes de début et de n-gramme de fin de phrase est fonction du corpus d'entrée, alors que le nombre "milieu de phrase" (séquences de plusieurs n-grammes formant un morceau de phrase) est fonction de la combinatoire possible des n-grammes entre eux pour n fixé. Plus généralement, il faudrait être en mesure d'anticiper la validité des états appartenant aux futures couches pour une couche donnée dans le MDD ; (ii) Il s'avère que ce $\text{MDD}_{\mathcal{R}_3}$ a une compression médiocre. En effet, le nombre de solutions est du même ordre de grandeur que le nombre d'arcs du MDD. Cette décompression à la construction est la conséquence l'utilisation de l'état n-gramme qui est discriminant en fonction des n derniers arcs visités.

Concernant la mémoire utilisée dans l'implémentation actuelle, il nous faut tout de même plus de 64 Go en 5-gramme pour effectuer nos calculs sur un corpus relativement petit. Notre corpus fait quelques centaines de Mo, sachant qu'il n'est plus inhabituel de voir des corpus manipulés d'une taille de l'ordre du gigaoctet, voire de la centaine de gigaoctets. Ceci est à nuancer par le fait que la constitution de notre corpus est un facteur important de la qualité des phrases engendrées. Il n'y a pas d'intérêt à le faire grossir sans avoir de justification. D'où notre choix de restreindre à des livres de littérature jeunesse dont on fait l'hypothèse d'y trouver des phrases simples.

Enfin, comme on peut le voir dans le Tableau 3, si nous souhaitons obtenir plus de phrases, soit on augmente la taille du corpus, soit on diminue la taille des n-grammes. En effet, on produit un nombre équivalent de phrase en utilisant 1% du corpus en 3-gramme qu'en utilisant 100% en 5-gramme.

3.3 Analyse des phrases

Parmi les phrases que nous sommes capable d'engendrer, il y a des phrases dites admissibles, i.e., dont la syntaxe et

Bien que je ne veux pas que les yeux de ce que ça me plaira
A l'autre bout de la place au fond de la voiture de ma mère
Allez le dire à mon père que je n'ai pas envie de me marier

TABLE 4 – Trois exemples de phrases en 3, 4 et 5 grammes respectivement.

Phrases admissibles
L'homme tourna la tête en direction de la voiture de police
Le dragon de métal tourna lentement la tête vers le plafond

Problème de sens
Prends place à côté de son mari pour sa douceur et sa grâce
J'avoue que je n'ai pas le temps de me réchauffer plus tard
La pluie formait un ruisseau de plus en plus insupportables

Problème de syntaxe
Vous lui attrapez la main et vous aide à monter sur son dos

TABLE 5 – Exemples de phrases en 5-grammes de différents types : (i) deux phrases admissibles, (ii) trois phrases ayant un problème de sens, (iii) une phrase ayant un problème de syntaxe.

Le policier passa la main dans sa poche et se mit à compter
Le policier passa la main dans sa poche et se mit à marcher
Le policier passa la main dans la poche arrière de mon jean
Le policier passa la main dans la poche de son jean déchiré
Le policier passa la main dans la poche de sa nouvelle robe

TABLE 6 – Exemples de phrases engendrées en 5-grammes ayant le même début de phrase.

J'essaie de changer de sujet avant de la perdre pour de bon
Tuez tous ceux qui se trouvaient sur le dessus de son siège
Dis à mon père que je n'ai pas envie de me les mettre à dos
L'homme avait tiré de sa poche un gros rat gris qui dormait
Posez une main sur le bras de son mari et le roi de ce pays
J'imagine qu'on ne peut pas savoir tant qu'on ne l'a pas vu
Pour ma part je me tourne et me dirige vers le bout du quai
Prends place à côté de son mari pour sa douceur et sa grâce
Très peu de choses arrivent sans que je le voie de mes yeux
Or sa mère lui avait demandé de reculer du bord de la route
J'espère que je n'ai pas vu mes parents depuis plus d'un an
Allez le dire à mon père que je n'ai pas envie de me marier
J'avoue que je n'ai pas le temps de me réchauffer plus tard
Vous seriez surpris de ce que venait de dire le vieil homme
Cette fois encore le jeune homme ne se décidait pas à poser
J'ai du mal à croire que tu me dises que tu ne m'aimes plus
Rien qui ressemble à une petite maison en bois et en pierre
Ceux qui ont perdu la vie à cause de ce que nous avons fait
Comme il regrettait à présent de ne pas se noyer de chagrin

TABLE 7 – Proposition d'un jeu de 19 phrases engendrées à partir de 5-grammes.

le sens sont corrects, mais aussi des phrases dont le sens est curieux, ou bien des phrases dont la syntaxe n'est pas correcte (voir Tab. 5).

On engendre des phrases qui respectent des règles mais on a tendance à engendrer toutes les variantes d'une même amorce phrase (voir Tab. 6). Ces variantes peuvent être toutes de bonne qualité comme on peut le voir avec l'amorce : "Le policier passa la main dans sa poche". Cependant, dans l'optique de former un jeu de phrases pour test, remarquons que nous devrons garantir une variabilité pour éviter d'avoir dans le même jeu de phrases des bouts de phrases similaires.

On constate alors que plus on diminue la taille des n-grammes plus la qualité grammaticale et sémantique des phrases diminuent. (voir Tab. 4)

Dans le Tableau 7, on présente un jeu de 19 phrases que nous avons sélectionnées parmis les 4000 solutions obtenues. Hormis le fait que ces phrases devront faire l'objet d'une validation expérimentales, une prochaine analyse consistera à estimer le pourcentage de phrases admissibles parmis les phrases générées (en terme sémantique et de syntaxe).

4 Conclusion

Ces résultats préliminaires montrent qu'il est possible de modéliser les règles du test MNREAD avec succès dans le paradigme MDD et d'être en mesure de prendre en comptes ces règles dès la génération. Comme nous l'avons montré, il est maintenant possible d'engendrer un grand nombre de phrases : plusieurs milliers en 5-grammes.

Par rapport à l'approche de Mansfield et al. [13], notre approche présente plusieurs avantages. D'abord, notre approche ne nécessite pas un travail manuel préliminaire long et fastidieux de construction de structures de phrases admissibles. A l'inverse, notre approche ne requiert que la constitution d'un corpus de phrases. Aussi en profitant de la combinatoire inhérente à l'utilisation de n-grammes, notre approche permet beaucoup plus facilement d'engendrer des phrases d'une plus grande diversité. Tout l'intérêt de notre approche est aussi qu'elle est *a priori* facilement généralisable dans différentes langues. Nous avons pu l'appliquer français (où les accords et les conjugaisons sont plus complexes qu'en anglais), là où la méthode de Mansfield [13] profite de cette flexibilité de l'anglais qui permet de construire plus facilement des *templates*. Il est toutefois difficile de nous comparer en terme de temps de calcul d'autant plus que l'entrée de nos méthodes sont très différentes (*templates* de phrases vs n-grammes). La méthode proposée par Mansfield et al. consiste à engendrer puis tester ; à l'inverse, notre méthode à pour esprit d'éviter de construire autant que possible des phrases non valides (même si certaines contraintes nous oblige à construire la phrase jusqu'au dernier mot pour s'assurer de sa validité). La composition des résolutions des sous-problèmes résulte en une convergence vers l'ensemble exhaustif de solutions du problème pour un corpus donné.

Notre méthode produit des phrases majoritairement cor-

rectes syntaxiquement, même si cette syntaxe peut-être lourde ou maladroite. Ceci est intéressant car, alors que rien n'est imposé en terme de syntaxe, une structure syntaxique en général correcte est obtenue grâce à la combinaisons n-grammes. Par contre, notre méthode ne peut aucunement garantir le sens des phrases. Ceci étant dit, à partir de ces milliers de phrases respectant toutes les contraintes MNREAD, il nous a été facile d'identifier un nouveau jeu de 19 phrases candidates ce qui, en soit, est déjà un résultat très important.

Dans la suite de ce travail, nous souhaitons donc considérer la modélisation d'autres contraintes, non pas associées à de nouvelles règles mais associées à la syntaxe et au sens. Tout l'objectif est d'être capable d'obtenir des phrases qui intègrent ces contraintes dès la génération pour éviter la phase de vérification a posteriori. Pour y parvenir, une idée sera de considérer des travaux qui tendent à montrer que cette notion d'acceptabilité d'une phrase est corrélée à la fréquence des mots dans une langue et donc indirectement, entre autres, aux probabilités qui peuvent être calculées à partir de modèles de langue comme celui des n-grammes [8].

Aussi nous avons choisi d'utiliser des MDD déterministes, mais nous envisageons d'utiliser des MDD non-deterministes comme perspectives d'amélioration du modèles notamment pour les $\text{MDD}_{\mathcal{R}_4}$ et $\text{MDD}_{\mathcal{R}_2}$ et ainsi, reconsidérer l'ordre de l'intersection proposée dans l'optique d'améliorer les performances.

Pour finir, nous voulons insister sur l'une des grandes forces de cette approche qui est sa modularité. En fonction des besoins, on va pouvoir modifier comme on le souhaite les contraintes ou en rajouter. Ainsi, les perspectives à plus long terme de ce travail sont de voir comment adapter notre approche dans d'autres problématiques de génération de texte sous contraintes, comme pour la dyslexie ou l'apprentissage de langues.

Références

[1] S.B. Akers. Binary decision diagrams. *IEEE Transactions on Computers*, C(27) :509–516, June 1978.

[2] D. Bergman, A. A. Cire, W.-J. Hoeve, and J. Hooker. *Decision Diagrams for Optimization*. Springer Publishing Company, Incorporated, 1st edition, 2016.

[3] R.R.A. Bourne, S.R. Flaxman, T. Braithwaite, M.V. Cicinelli, A. Das, J.B. Jonas, J. Keeffe, J. Kempen, J. Leasher, H. Limburg, K. Naidoo, K. Pesudovs, S. Resnikoff, A. Silvester, G.A. Stevens, N. Tahhan, T.-Y. Wong, H.R. Taylor, R. Bourne, and P. Sieving. Magnitude, temporal trends, and projections of the global prevalence of blindness and distance and near vision impairment : a systematic review and meta-analysis. *The Lancet Global Health*, 5(9) :e888–e897, 2017.

[4] A. Calabrese, C. Owsley, G. McGwin, and G.E. Legge. Development of a reading accessibility index using the MNREAD acuity chart. *JAMA Ophthalmol.*, 134(4) :398–405, April 2016.

[5] M.D Crossland, G.E. Legge, and S. C. Dakin. The development of an automated sentence generator for the assessment of reading speed. *Behavioral and Brain Functions : BBF*, 4 :14–14, 2007.

[6] Z. Hu, Z. Yang, X., R. Salakhutdinov, and E. P. Xing. Toward controlled generation of text. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, page 1587–1596. PMLR, 2017.

[7] V. Jung and J.-C. Régin. Efficient operations between MDDs and constraints. In *Proceedings of the 19th International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, 2022.

[8] J. H. Lau, A. Clark, and S. Lappin. Grammaticality, acceptability, and probability : A probabilistic view of linguistic knowledge. *Cognitive science*, 41(5) :1202–1241, 2017.

[9] G. E. Legge. *Psychophysics of Reading in Normal and Low Vision*. CRC Press, 2 edition, 2021.

[10] B. Lété, L. Sprenger-Charolles, and P. Colé. Manulex : A grade-level lexical database from french elementary-school readers. *Behavior Research Methods, Instruments & Computers*, 36 :166–176, 2004.

[11] J.S. Mansfield, S.J. Ahn, G.E. Legge, and A. Luebker. A new reading-acuity chart for normal and low vision. *Ophthalmic and Visual Optics/Noninvasive Assessment of the Visual System Technical Digest*, 3 :232–235, 1993.

[12] J.S. Mansfield and G.E. Legge. The mnread acuity chart. In *Psychophysics of reading in normal and low vision*, page 167–191. Taylor and Francis, 2006.

[13] S. Mansfield, N. Atilgan, A. Lewis, and G. Legge. Extending the mnread sentence corpus : Computer-generated sentences for measuring visual performance in reading. *Vision research*, 158 :11–18, 2019.

[14] P. Márquez-Neila, M. Salzmann, and P. Fua. Imposing hard constraints on deep networks : Promises and limitations. *CoRR*, abs/1706.02025, 2017.

[15] J.-L. P., D. Paillé, and T. Baccino. A new sentence generator providing material for maximum reading speed measurement. *Behav Res*, 47 :055–1064, 2015.

[16] G. Perez. *Diagrammes de décision : contraintes et algorithmes*. PhD thesis, Université Côte d'Azur, 2017.

[17] G. Perez and J.-C. Régin. MDDs : Sampling and probability constraints. In *Proceedings of the International Conference on Principles and Practice of Constraint Programming*, page 226–242, 2017.

[18] G. Perez and J.-C. Régin. Efficient operations on MDDs for building constraint programming models. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2015.

[19] W. Radner. Reading charts in ophthalmology. *Graefe's Archive for Clinical and Experimental Ophthalmology*, 255(8) :1465–1482, August 2017.

[20] S. N. Ravi, T. Dinh, V. S. Lokhande, and V. Singh. Explicitly imposing constraints in deep networks via conditional gradients gives improved generalization and faster convergence. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01) :4772–4779, 2019.

[21] P. Roy, G. Perez, J.-C. Régin, A. Papadopoulos, F. Pachet, and M. Marchini. Enforcing structure on temporal sequences : the Allen constraint. In *International conference on principles and practice of constraint programming*, page 786–801. Springer, 2016.

[22] G. S. Rubin. Measuring reading performance. *Vision Research*, 90(C) :43–51, September 2013.

[23] H. Schmid. Probabilistic part-ofispeech tagging using decision trees. In *New methods in language processing*, page 154, 2013.

[24] M. Trick. A dynamic programming approach for consistency and propagation for knapsack constraints. *Annals of Operations Research*, 118 :73–84, 2003.