IFT-7028 Conception et simulation de systèmes intelligents pour l'industrie 4.0

Simulation et Optimisation du Séchage de bois via l'Apprentissage par Renforcement

Ayoub.E	*Réjean.D*	*Vincent.C*	*Francis.T*
111 274 558	*903 187 234*	*111 160 754*	*111 288 372*

Projet de Session
Présenté à Mr. Jonathan Gaudreault

Université Laval
Faculté de Sciences et Génie

Juin 2022

Table des matières

I. INTRODUCTION 4

II. DESCRIPTION DU PROBLÈME 6

III. APPROCHE PROPOSÉE 8

IV. PROTOCOLE D'EXPÉRIMENTATION 10

V. RÉSULTATS 16

VI. DISCUSSION 17

VII. CONCLUSION 19

VIII. RÉFÉRENCES 20

--- ***RÉSUMÉ*** ---

Ce projet de recherche fait en collaboration avec Syntell et PMP SOLUTIONS présente des résultats quant à l'application simultanée de la simulation à évènement discrets et l'apprentissage par renforcement au problème d'ordonnancement des opérations de séchage du bois industrielle dans le but de stabiliser les proportions des différents produits dans la cour sans compromettre le taux d'utilisation des séchoirs. Le but de cette politique était d'assurer un maximum de flexibilité au niveau des séchoirs étant donnée une forte variabilité des temps de séchage. Pour ce faire, l'équipe a monté un simulateur à évènements discrets (SED) basé sur les opérations d'une entreprise de transformation du bois ainsi que sur des données réelles fournies par une compagnie spécialiste dans le domaine du nom de PMP Solutions. Également, un agent d'allocation représenté par un réseau de neurones profond fût entrainé via de l'apprentissage par renforcement grâce à l'algorithme *Proximal Policy Optimization*, également connu sous le nom de PPO [10] couplé avec la technique d'ingénierie de *Reward Heuristic-Guided Reinforcement Learning* [11]. Le tout a généré des résultats initiaux très prometteurs. Toutefois, à mesure que la simulation des opérations de séchage du bois industrielle se complexifiait (modélisation des horaires de travail, séchage à l'air libre, chargement/déchargement de wagons au lieu de séchoirs) la convergence de l'algorithme s'allongeait, si bien que l'entraînement n'a pas pu être complété dans les délais requis. L'agent présenté offre donc, avec un niveau de confiance de 95%, des performances inférieures aux heuristiques développées par l'équipe. Un résumé étoffé de la méthodologie de recherche des espaces d'action et d'observation est également présentée. Finalement, une discussion sur les améliorations possibles sur la méthodologie ainsi qu'une proposition de prochaines étapes potentielles sont proposées.

I. INTRODUCTION

1.1) Mise en Contexte

Parmi les différents sujets étudiés en informatique et en recherche opérationnelle, les problèmes d'affectation occupent une place primordiale dans les études académiques car ceux-ci sont essentiels à l'optimisation de procédés industriels. En effet, contrairement aux problèmes de permutation où nous sommes intéressés à trouver un arrangement des intrants qui optimise une certaine fonction objective, un problème d'affectation cherchera plutôt à créer un "mapping" optimal entre un ensemble de ressources et un ensemble d'entités, de telle sorte qu'une ou plusieurs fonctions objectives soient optimisées et qu'un ensemble de contraintes soit respecté.

Afin de rendre le secteur industriel plus efficace, de nombreuses compagnies manufacturières dépensent massivement dans des modèles et des algorithmes d'optimisation qui pourraient améliorer l'optimalité des affectations générées car cela peut représenter des millions de dollars d'économies à travers le temps [1]. Malheureusement, rares sont les problèmes pouvant être résolus à l'échelle polynomiale car beaucoup d'entre eux sont classifiés comme NP-difficile. En d'autres termes, à mesure que nous augmentons la taille des instances du problème, plus il devient difficile d'obtenir des solutions optimales car l'espace de recherche devient beaucoup trop important.

Ainsi, les approches traditionnelles pour résoudre ce type de problèmes tournent principalement autour de l'utilisation des connaissances d'experts du domaine afin de produire des modèles et des heuristiques personnalisés. En d'autres mots, nous référons ici à des stratégies permettant une forte réduction de l'espace de recherche afin de trouver des solutions optimales rapidement. Alors que les méthodes exactes sont préférées pour les petites instances car elles garantissent une solution optimale, leurs consommations de ressources sous forme de temps et de calculs deviennent trop importantes à mesure que le problème gagne en complexité. Par conséquent, les professionnels décident souvent de troquer de l'optimalité contre de la consommation ressource via ces méthodes dites heuristiques qui permettent à des solveurs de sélectionner des variables et des valeurs de branchement pour filtrer l'espace de recherche beaucoup plus rapidement.

Il existe également d'autres méthodes pour optimiser l'efficacité des systèmes de production 4.0. La quête de méthode toujours moins chers et plus efficaces a entre autres laissée place à l'émergence de la simulation via des jumeaux numériques et l'optimisation via l'apprentissage par renforcement profond. Effectivement, plusieurs recherches démontrent que l'implémentation de systèmes utilisant ces méthodes ont l'avantage de performé de façon supérieure face aux méthodes conventionnelles et qu'elles réduisent la dépendance sur l'utilisation des connaissances d'experts, tous cela ayant un impact important sur le coût total des solutions d'optimisation pour les compagnies industrielles [2].

1.2) Application à l'industrie du bois

L'industrie de la transformation du bois représente l'une des pierres angulaires du développement social et économique des pays à travers le monde. En effet, le bois d'œuvre représente l'un des principaux intrants dans l'industrie de la construction résidentielle ainsi que dans de nombreuses fournitures du quotidien. De ce fait, une meilleure optimisation des procédés qui diminuera les coûts fixes d'entreprises de transformation du bois peut générer énormément d'utilité à l'économie mondiale. Par exemple, l'utilisation de la simulation et l'optimisation avec des jumeaux numériques pour cette industrie a été bien analysé par *Chabanet et al.* (2022) [3], où ceux-ci conclue que l'industrie de la transformation du bois profiterait grandement de l'utilisation de jumeaux numériques afin de pouvoir tester et optimiser divers plans de production, ainsi que de réduire les pertes d'informations nécessaire à prendre les bonnes décisions.

En analysant en détails cette industrie, de nombreuses étapes sont nécessaires afin de convertir des tronçons d'arbres fraichement coupés vers des produits de bois ayant une bonne qualité revendable sur le marché. Ainsi, les opportunités d'optimisation sont multiples. Puisque nous sommes très limités dans le temps, nous avons décidé de se concentrer uniquement sur l'optimisation du procédé de séchage des paquets de bois venant de la scierie et actuellement stocké dans la cour des produits sciés / cour du séchage à l'air libre. Donc, d'un point de vue général, notre problème peut se résumer à comment ordonner de façon optimale les opérations de notre cour de séchage afin d'optimiser les fonctions objectives visées par le système. Par exemple, lorsqu'une station de séchage accéléré se libère, quel ensemble de produits présents dans la cour doit être affecté à la station de séchage accéléré afin de maintenir un ratio d'inventaire de produit stable?

1.3) Méthodologies Actuelles

Ce problème de planification et l'ordonnancement des opérations de séchage du bois d'œuvre est bien analysé par de nombreux groupes d'experts, tous proposant des solutions d'optimisation ingénieuse se basant sur l'usage de modèle d'optimisation. Par exemple, *Marier et al.* (2021) [4] ont proposé un modèle de programmation mixtes en nombres entiers afin de générer des patrons de chargements de produits sur demande afin de minimiser les retards de commande. Similairement, *Gaudreault et al.* (2010) [5] ainsi que *Huka et al.* (2021) [6] proposent quant à eux des heuristiques de recherche, tandis que *Gaudreault et al.* (2011) [7] ont opté pour un modèle de programmation par contrainte. Il est donc commun dans l'industrie de créer des modèles choisissant quelles stratégies choisir à partir d'une liste de patrons de chargements préétablis, dans le but d'optimiser la ou les fonctions objectives. Également, l'autre choix considéré par les professionnels est la génération de patrons sur demande en fonction des produits disponibles dans la cour. Souvent, ces modèles d'optimisation gagnent en complexité rapidement car de multiples variables de décision doivent être considérées, telles que le choix de procédé de séchage, les dimensions et la hauteur des produits, ainsi que le choix de produits issus de la cour.

II. DESCRIPTION DU PROBLÈME

2.1) Problématique

Donc plus précisément, notre projet consiste en résoudre une version simplifiée du problème d'allocation de paquets de bois destinés aux opérations de séchage, tout cela à travers une **méthode d'apprentissage par renforcement** qui permettrait la création d'un agent optimisant une fonction objective désirée. Grâce à l'usage de la simulation à évènement discrets, notre agent serait en mesure d'apprendre à optimiser des fonctions telles que le **respect de certaines proportions de produits** variant dynamiquement à travers la simulation. Également, celui-ci peut apprendre à optimiser d'autres paramètres telles que l'affectation optimale de produits aux stations de séchage en choisissant la meilleure règle de chargement à appliquer parmi une liste de 13 règles fournies par *PMP Solutions*, une compagnie spécialisée dans le domaine.

Pour bien comprendre la problématique, voici donc un aperçu simplifié des différentes étapes d'une entreprise œuvrant dans la transformation du bois: une fois que les tronçons d'arbres sont coupés de la forêt, ils sont transportés dans la cour des tronçons bruts de l'entreprise, où les troncs possédants chacun des caractéristiques différentes passent à travers des stations de sciage, générant ainsi une multitude de produits différents comme il est démontré à la figure 2. Ces produits sont par la suite séchés grâce à deux options: le séchage à l'air libre et le séchage au sein de stations. Le séchage à l'air libre est un processus lent qui est souvent combiné avec des temps réduits de stations de séchage accéléré car le but est de faire descendre l'humidité du bois de 60% à 20% en moyenne, tout dépendant de la nature, des conditions, et des caractéristiques du bois traités. *(Voir figure 1)*

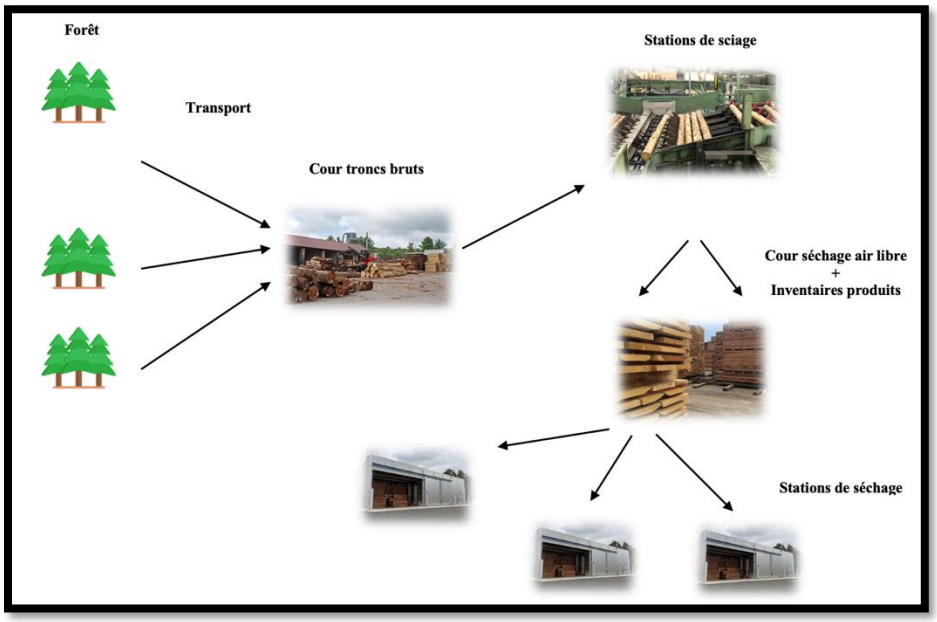

Figure 1 : Étapes simplifiées de la transformation du bois

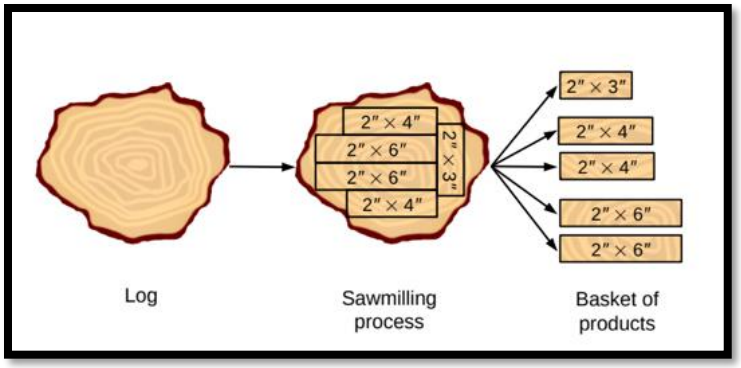

Figure 2 *:* Génération de produits à partir de troncs bruts

De multiples facteurs doivent être prise en compte pour optimiser les opérations de séchage des paquets de bois encore humide. En effet, puisque le séchage est fait en lots, il est nécessaire que seulement des produits ayant un temps de séchage estimé similaire via un processus de séchage commun soient regrouper dans le même lot. Ce temps de séchage estimé varie quant à lui en fonction de dizaines de facteurs, telles que le type de bois, la largeur, l'épaisseur, le temps de l'année, le temps passé au séchage à l'air libre, et ainsi de suite. Également, les produits doivent idéalement être assemblé sous forme de lot qui minimise le volume libre restant dans la station de séchage, un facteur clé permettant d'optimiser l'usage et la productivité des séchoirs [8].

Par exemple, le temps nécessaire pour sécher des produits à base de chêne rouge avec des dimension 4 x 4 à une humidité optimale (7 % dans ce cas-ci) peut être deux à trois fois supérieur à celui nécessaire pour sécher des produits à base d'érable 4 x 4 [8]. De plus, d'autres facteurs s'ajoutent à cela, tels que le fait que le processus de séchage doit être beaucoup plus doux pour les produits à base de chêne rouge afin d'éviter les défauts de séchage. Un bon séchage optimise donc l'allocation des produits de sortes à ce que chaque produit dans un lot obtient le temps de séchage juste afin qu'il atteigne une humidité optimale sans subir de défauts. Ainsi, des produits de bois possédant la même épaisseur et la même teneur en humidité peuvent souvent être séchés ensemble de manière très économique. La demande actuelle des différents produits est généralement la fonction objective clé qui assure la viabilité de l'entreprise à travers la génération de revenu et la minimisation des inventaires de produits finis [8].

III. APPROCHE PROPOSÉE

Au sein de ce problème d'affectation, nous considérons donc comme ressources nos stations de séchages et les produits issues de la scierie actuellement en attente dans l'inventaire / séchage à l'air libre en tant que nos entités qui doivent être assignées à ces ressources, le tout afin d'optimiser la fonction objective qu'est la stabilité de l'inventaire et en respectant les contraintes de notre cour. De ce fait, bien que la fonction objective reliée à la satisfaction de la demande et des ordres soit la plus commune dans les recherches, nous avons décidé de nous concentrer sur l'optimisation de la stabilité de notre inventaire afin de découvrir les éléments affectant le plus celui-ci. Également, à la place de générer des affectations sur demande lors du séchage, nous avons décidé de suivre la pratique commune de choisir parmi une liste de 13 règles de chargement préétablies par PMP Solutions afin que notre problème soit le plus réaliste possible.

3.1) Simulation à évènements discrets (SED)

Ainsi, l'une des premières composantes clés à la création et l'entrainement de notre agent allocateur est la génération d'un environnement imitant les extrants d'une scierie, où des produits possédant des caractéristiques différentes seront dirigés de façon statistique vers la cour d'inventaire et de séchage à l'air libre afin que notre agent puisse avoir accès à un **état de son environnement** à chaque itération temporelle de notre simulation. En effet, lorsqu'une station se libère à un temps t_i et que notre agent doit assigner un ensemble de produits {p_1, p_2, p_3, ...} via le choix d'une règle, l'agent prendrait donc en intrant cet état, supplémenté par d'autres paramètres afin de produire les allocations nécessaires. L'apprentissage se fera à travers la récompense (reward) reçu par notre agent, une valeur calculée à chaque assignement et qui se base sur le choix de la ou les fonctions objectives retenues ainsi que le respect des contraintes induites.

Figure 3: Exemple d'évènements discrets caractérisant la dynamique de notre simulation

Une particularité de la simulation à évènements discrets (SED) est que les états du modèle environnemental changent non pas continuellement, mais seulement à des points discrets et de façon stochastique dans le temps. Par conséquent, ce sont donc des évènements tels que la sortie de produits de la scierie, la disponibilité d'un camion-chargeur, ou le fait qu'une station de séchage devient libre qui guident la dynamique de notre système. Bien-sûr, divers paramètres discrétionnaires tels que la quantité de stations de séchage, le nombre de camions-chargeurs, ou la complexité de la fonction de variation des proportions à respecter peuvent tous être ajusté afin de faire varier le degré de complexité.

3.2) Agent d'allocation "Deep-RL"

Comme dit précédemment, à chaque évènement où une station de séchage se libère, notre agent d'allocation doit générer une action précisant quelle règle choisir afin de remplir le wagon qui sera diriger vers la station de séchage en question. L'utilisation de l'apprentissage par renforcement profond fût un choix guidé par le désir d'apprendre une politique de décision et non approximer via l'apprentissage supervisé, qui est d'ailleurs largement dépendant de la qualité et de la distribution sous-jacente des données d'apprentissage, une ressource très difficile et coûteuse à générer. De plus, dans le cas de l'apprentissage supervisé, même si les solutions optimales aux instances sont utilisées dans la phase d'entrainement, la capacité de généraliser sera probablement inférieure à un agent entrainé via l'apprentissage par renforcement car l'agent RL sera en mesure d'explorer des états divers et leurs récompenses respectives [9].

De ce fait, la politique que suivra l'agent sera paramétrisée par un réseau de neurones de type "perceptron multicouches" (MLP) et l'entrainement se fera via l'algorithme d'Open AI appelé *Proximal Policy Optimization*, également connu sous le nom de PPO [13]. D'ailleurs, il est bien important de comprendre que par politique, nous parlons d'un mapping entre l'espace d'états et l'espace d'actions de notre agent. Donc, à chaque itération de notre simulation où nous avons la combinaison d'une station de séchage libre, un camion-chargeur disponible et du bois dans la cour, notre agent va prendre en intrant l'état actuel de l'environnement, pour ensuite générer en extrant une action correspondant **à un choix parmi les 13 règles permises**. Bref, la simulation déplacera un chargement de bois correspondant à la règle choisie par notre agent.

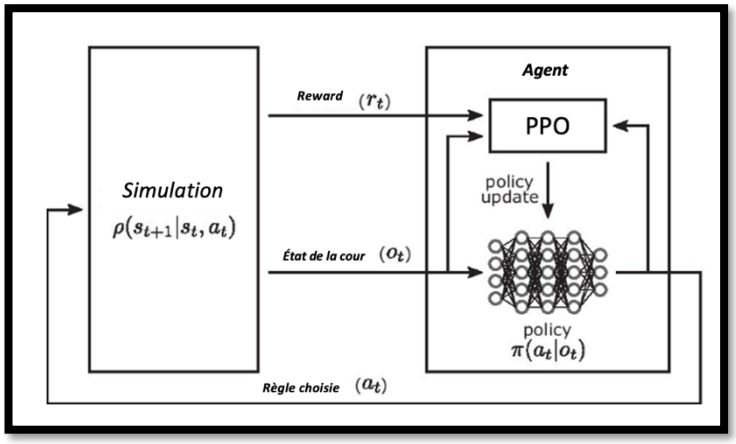

Figure 4: Interaction entre l'algorithme PPO et la simulation de la cour à bois

L'état de notre simulation est composé des produits dans la cour, les quantités de produits dans l'inventaire, l'heure et la journée actuelle, ainsi qu'une estimation du temps restant aux différents séchoirs avant que ceux-ci se libèrent. Toutes les informations de l'état sont entre 0 et 1 grâce à une normalisation min/max, champs par champs, avec des bornes définies logiquement selon les valeurs possibles plutôt que dynamiquement avec les valeurs présentes. La fonction objective, qui guide la récompense donnée à notre agent, est dans un premier

temps quantifié par deux paramètres clés à l'inventaire : l'intervalle de proportions désirées entre nos différentes règles (lignes vertes sur la *Figure 5*), ainsi qu'une valeur temporelle dictant la quantité approximative de chaque règle voulue à travers le temps (ligne rouge sur la *figure 5*). Si notre agent dévie de l'intervalle, l'agent est puni au carré en fonction de la distance par rapport à l'intervalle voulue. Si notre agent ne fonctionne pas assez efficacement en flux poussé et qu'une quantité est supérieure à la quantité désirée à travers le temps, la distance au carré par rapport à celle-ci est aussi ajoutée à la punition de l'agent. Pour chaque intervalle respecté, c'est à dire que notre agent a su maintenir les paramètres clés de l'inventaire entre des valeurs critiques, la récompense obtenue est de 5. De ce fait, la récompense maximale pour une batch parfaite est de 13, puisque cela implique que les 13 règles furent respectées à la lettre. (*Voir figure 5).* Voici l'équation de notre fonction objective utilisé, au début du projet, pour calculer le coût total des décisions prises par notre agent lors d'une réplication.

$$X_{rt} : \textit{Nombre de chargement de la règle } r \textit{ dans la cour au temps } t$$
$$Y_{rt} : \textit{Nombre de chargement de la règle } r \textit{ voulus dans la cour au temps } t$$
$$Z_{rt} : \textit{Proportion de chargement de la règle } r \textit{ voulus dans la cour au temps } t$$
$$\sum_{t=0}^{T} \sqrt{abs(X_{rt} - Y_{rt})} + \sum_{t=0}^{T} \text{abs}(X_{rt} - Z_{rt})$$

Trois conditions pouvant être perçues comme des contraintes sont nécessaires afin que l'agent puisse demander l'état de la simulation afin de performer une allocation, puis recevoir un score. En effet, il doit d'abord y avoir du bois dans la cour, puis un camion-chargeur de disponible, et finalement un wagon de disponible afin de charger les produits de bois sur celui-ci. Les règles formant les possibles actions de notre agent sont responsables non seulement pour guider comment un wagon entrant dans une station de séchage sera rempli, mais également le temps nécessaire au camion-chargeur de performer les déplacements requis, puis le temps de séchage total nécessaire au wagon en question. Ce temps de séchage est directement utilisé pour calculer la prochaine itération où cette station se libérera, un paramètre enregistré dans la simulation comme évènement à venir.

Figure 5 : (A) Exemple de réplication où la police de chargement de type 1 n'optimise pas du tout le management de l'inventaire. (B) Exemple de réplication où la police de chargement de type 1 optimise bien l'inventaire. (C) Exemple où l'agent a bien optimisé les 13 règles, de ce fait obtenant une récompense qui converge asymptotiquement vers la valeur 13.

IV. PROTOCOLE D'EXPÉRIMENTATION

4.1) Prétraitement des données

Les données nécessaires à la réalisation de ce projet nous ont été fournies par PMP Solutions. Celles-ci contenaient la production hebdomadaire en pied-mesure-planche (PMP) de chacun des 121 produits de sapin et d'épinette sortant de la scierie. Cette production varie selon la planification actuelle de la scierie. Ainsi, le fichier fourni contenait les productions pour la planification sapin où environ 75% de la production est composée de sapin et la planification épinette qui comprend à son tour 75% d'épinette. Une troisième planification 50/50 a été calculée comme étant la moyenne des deux pour chaque produit selon les directions de François de PMP Solutions. Les différentes règles de séchage ont également été données tout comme les différents produits qui y sont associés. Dans certains cas, des produits ne correspondaient à aucune règle. Selon les indications qui nous ont été fournies, ceux-ci ne sont pas envoyés aux séchoirs, donc ils ont pu être retirés de notre problème.

Un prétraitement des données a été effectué sur celles-ci pour les rendre utiles à notre projet. En effet, la simulation fonctionne en suivant les règles de séchage. Il convient alors de faire ressortir toute l'information pertinente de chacune de celles-ci.

- Avec la production de chaque produit, il est possible de calculer la production de chaque règle en sortie de scierie. Celles-ci sont ensuite utilisées pour déterminer les proportions de la capacité de stockage initiale.

- Avec l'épaisseur des produits de chaque règle, le volume pouvant entrer dans chacun des quatre séchoirs disponibles a été établi par règle. Pour les règles d'un pouce d'épaisseur, ce volume est de 155 milliers de PMP pour deux séchoirs et de 175 milliers de PMP pour les deux autres. Pour ce qui est des règles de deux pouces d'épaisseur, ces valeurs étaient de 215 et 245 milliers de PMP respectivement.

- Avec l'essence des produits, les temps de séchage ont pu être ajoutés : 44 heures pour les règles contenant de l'épinettes et 100h pour celles contenant du sapin.

- Avec les longueurs des produits, les temps de chargement ont été approximés. En effet, selon les données fournies, un chargement contient, selon leur longueur, entre 90 et 160 paquets. Ainsi, considérant qu'un paquet prend en moyenne 5 minutes à déplacer, le temps de chargement varie entre 7.5 et 13 heures. Pour associer un temps de chargement à une règle, la longueur moyenne d'une règle a été calculée à partir de la longueur de chaque produit et de leur proportion. Puis, il a été établi qu'une longueur de 16 pieds prenait 7.5 heures alors qu'une longueur de 6 pieds prenait 13h. Enfin, l'équation suivante a été utilisée :

$$Temps\ de\ chargement = 13h + \left(\frac{longueur\ moyenne - \ 6\ pieds}{16\ pieds - 6\ pieds}\right) \times (7.5h - 13h)$$

4.2) Simulateur

Le simulateur fût construit avec l'outil Simpy du langage de programmation Python. Ce choix fut orienté afin d'avoir une intégration plus facile avec l'agent d'allocation (décrit plus loin) qui est aussi facilement utilisable sous Python. Le simulateur contient donc les différentes classes suivantes :

- La classe principale du simulateur est « EnvSimpy ». Celle-ci représente l'environnement globale de la cour à bois et contient des objets des classes décrivant la scierie, la cours à bois, les camions-chargeurs et les séchoirs. Elle s'occupe aussi de la gestion des appels aux événements, l'enregistrement des journaux d'événements et le calcul des indicateurs et métriques.

- La classe « Emplacement » permet de surclasser les méthodes des « Ressources » de Simpy afin de faciliter notre gestion des différents endroits où nous avons une limite d'espace de disponible (sortie de la scierie, wagons du séchoir, séchoirs).

- La classe « Loader » permet de gérer les déplacements des camions-chargeurs. Cette classe n'est pas directement basée sur Simpy. En effet, ce dernier a besoin d'un état à un temps « t » afin de retourner la meilleure action possible qui sera faite à ce même temps « t ». La gestion de cette classe en dehors de « Simpy », nous donne plus de liberté et de contrôle sur ce qui est fait à ce temps « t » et dans quel ordre exactement facilitant ainsi l'interaction avec l'agent d'allocation.

- Le fichier « Temps.py » permet de gérer les horaires de travail pour la scierie et les camions-chargeurs en fournissant les fonctions de calcul nécessaires pour transformer un temps nécessaire pour une action en un temps correspondant en fonction de l'horaire de travail et vice-versa.

Les principaux éléments considérés par la simulation sont les suivants :

- Le bois sort de la scierie à une vitesse définie lors du prétraitement des données, sous la forme d'un chargement prêt à être placé dans un wagon. Si la cours est pleine, la scierie est bloquée et attend que la cour se libère pour sortir du nouveau bois.

- Du moment où le bois sort de la scierie, celui-ci est prêt à être transporté par un camion-chargeur. Également, son temps de séchage se réduit à chaque semaine selon un paramètre pour simuler le séchage à l'air libre. D'autres paramètres (par essences) permettent de contrôler le moment à partir duquel on considère que le chargement commence à se détériorer car il a été trop longtemps dans la cour.

- Les camions-chargeurs et la scierie possède un horaire (8h à minuit, lundi au vendredi) alors que les stations de séchages fonctionnent 24h/24.

- Les camions-chargeurs remplissent des wagons à la porte des séchoirs selon le temps de chargement pré-établi. Lorsque le séchage est terminé le wagon dans le séchoir sort et celui à la porte entre dans le séchoir. Un temps de déchargement est considéré avant de pouvoir utiliser à nouveau le wagon qui vient de sortir, mais la gestion des camions-déchargeurs n'est pas modélisée (on considère qu'il y en a toujours un de disponible).

- Possibilité de nuancer les sorties de la scierie selon les planifications fournies par François de *PMP Solutions*, c'est-à-dire 25/75, 50/50, 75/25 (épinettes/sapins).
- Tous les temps et vitesses sont affectés par un paramètre permettant de mettre un peu de bruits dans les valeurs. Une loi triangulaire est utilisée avec la valeur prévue comme moyenne et les paramètres permettent d'accepter un certain pourcentage de différences de chaque côté de la moyenne.
- Initialisation de la cour afin d'avoir un environnement intéressant dès le départ de la simulation (évite la période transitoire).
- Produire les indicateurs nécessaires à l'analyse des résultats.

Bref, un résumé des principaux paramètres discrétionnaires de la simulation sont présentés dans le Tableau 1. Ceux-ci représentent le plus fidèlement possible la réalité tout en conservant une cour dont notre heuristique de comparaison est capable de garder le contrôle.

Paramètre de Simulation	Valeur habituelle	Description
Règles	13 règles avec leurs détails	Importation des données d'un fichier csv contenant les informations des règles que l'on veut utiliser dans la simulation.
Horaires	8 à minuit, lundi au vendredi	Horaire de travail pour la scierie et les camions-chargeurs/déchargeurs. Les séchoirs fonctionnent 24h/24.
Details Événements	False	Permet de conserver un log plus détaillé des événements qui se produisent. Aide pour trouver les problèmes, mais ralenti considérablement la simulation.
Nb Step Simulation Nb Step Simulation Test	64*100	Nombre d'actions effectuées par l'agent dans chacune des réplications lors de l'entraînement ou de la validation.
Capacité Sortie Sciage	100	Nombre de chargements possible dans la cour avant de considérer la cour pleine et de bloquer la scierie.
Temps séchage à l'air libre Ratio séchage à l'air libre Max séchage à l'air libre	7*24 0.1*(12/52) 0.3	Fréquence de mise à jour du temps de séchage suite à la considération du séchage à l'air libre ainsi que le facteur de réduction utilisé et le maximum de réduction possible si le bois reste longtemps dans la cour.
Durée Détérioration Épinette Durée Détérioration Sapin	30*24 4*30*24	Temps avant lequel on considère que le bois de chaque essence commence à se dégrader.
Variation production scierie Variation temps séchage Var. temps déplacement des camions	0.1	Pourcentage de variation en plus ou en moins des temps et vitesses des différents éléments de la simulation.
Facteur de sortie de la scierie Facteur du temps de chargement	0.35 0.85	Facteurs permettant de ne pas modifier les chiffres réels fournis par le client, mais qui permettent d'avoir une simulation traitable par une heuristique ou un agent moindrement efficace. Les facteurs compensent pour certaines simplifications du modèle (nb heures travaillées, aucun camion en surtemps, 4 séchoirs identiques)
Objectif stable en PMP	215 000*4*2.5	Objectif à long terme qu'on cherche à atteindre. L'hypothèse est que si nous avons 2.5 fois la capacité de nos séchoirs, la cour est sous contrôle.
Ratio Sapin / Épinette	50/50	Permet de changer la proportion de sapin et d'épinettes qui sort de la scierie.

Tableau 1 : Résumé des principaux paramètres discrétionnaires de la simulation

4.3) Opérateur Deep-RL

Notre agent a pu interagir avec l'environnement décrit plus haut via le Framework *gym* d'OpenAI et la librairie d'algorithmes d'apprentissage par renforcement profond *stable_baselines*3. L'algorithme interagi avec la simulation en envoyant une action puis en recueillant l'observation de l'état de la cour ainsi qu'un *Reward* (tel que décrit en 3.2). De plus, il est à noter l'observation de l'état du système ainsi que le Reward furent déterminés

de manière itérative et en ordre croissant de complexité au fur et à mesure que la simulation se complexifiait et devenait réaliste (*Tableau 2, 4.2*). Il fallait donc tenir compte de plus en plus de facteurs pouvant venir influencer la prise de décision de l'agent. Ce qui fût un défi de taille considérant qu'il fallait minimiser l'espace d'observation tout en s'assurant que l'algorithme ait accès à toute l'information et la rétroaction nécessaire pour apprendre une politique intéressante.

Cela étant dit, l'observation de l'état finale comprend le ratio épinette/sapin de la scierie, le nombre de chargement pour chaque règle dans la cour (X_r), le jour, l'heure et les temps de séchage restant dans les séchoirs (*Tableau 2*). De son côté, le *Reward* fut, pour les mêmes raisons, tout aussi difficile à fixer. L'équipe a finalement opté pour la formule présentée en *3.2* qui permet de donner un *Reward* après chaque action de l'agent avec le système dans le but de converger plus rapidement et d'éviter le problème du *Sparse Reward*. Toutefois, l'équipe a dû utiliser la méthode de l'*Heuristic-Guided Reinforcement Learning* [14], car l'agent avait beaucoup de difficulté à apprendre les horaires des camions-chargeurs ainsi que la gestion des séchoirs la fin de semaine. Il a donc fallu se servir de l'heuristique de gestion de l'horaire (présentée en *4.4)*, pour guider l'agent. Cela amène donc les modifications suivantes à la formule de *Reward* initiale décrite en *3.2* :

$$\lambda : 0.75$$

$$A_{mt} : Action\ retourné\ par\ le\ model\ m\ au\ temps\ t$$

$$A_{ht} : Action\ retourné\ par\ le\ l'heuristique\ h\ au\ temps\ t$$

$$\sum_{r=0}^{R} \sqrt{abs(X_{rt} - Y_{rt})} + \sum_{r=0}^{R} \text{abs}(X_{rt} - Z_{rt}) + \sum_{r=0}^{R} (1-\lambda) * (A_{mt} = A_{ht}) \qquad \forall\ t\ \in T$$

Simulation	**Observation de l'état**	***Reward***
Ajout des camions-chargeurs	X_r et ratio sapin/épinette	$\sum_{t=0}^{R} (X_r - Z_r)^2 + \sum_{t=0}^{R} (X_r - Z_r)^2 \qquad \forall\ t\ \in T$
Ajout des horaires de travail	Ajout du jour et de l'heure	$\sum_{t=0}^{R} abs(X_r - Z_r) + \sum_{t=0}^{R} (X_r - Z_r)^2 \qquad \forall\ t\ \in T$
Ajout des wagons de chargements	Ajout du temps restant dans les séchoirs	$\sum_{r=0}^{R} \sqrt{abs(X_{rt} - Y_{rt})} + \sum_{r=0}^{R} \text{abs}(X_{rt} - Z_{rt}) + \sum_{r=0}^{R} (1-\lambda) * (A_{mt} = A_{ht}) \qquad \forall\ t\ \in T$

Tableau 2 : Évolution de l'observation de l'état et du Reward

Comme dit précédemment, l'agent fût entrainé via l'algorithme PPO, considéré comme un type de méthode dite *Policy Gradient*, c'est-à-dire qu'elle cherche à paramétriser la politique en fonction du *Reward* obtenus via descente du gradient, et *Model-free*, c'est-à-dire qu'elle peut apprendre une politique sans avoir à se baser sur les probabilités de transitions spécifiques au problème à résoudre. L'algorithme apprend donc par essais et erreur où, à chaque étape, une mise à jour de la stratégie existante est performée via la modification de certains paramètres internes du réseau de neurones, mais tout en gardant une faible variance afin de garantir que la

politique mise à jour n'est pas trop différente de l'ancienne politique. Finalement, la méthode PPO est dite *On-Policy* c'est-à-dire qu'elle essaie d'améliorer directement la politique après avoir interagis avec la simulation (tel qu'expliqué en *3.2*).

Une fois l'espace d'action, l'espace des observations et le *Reward* déterminés, une recherche des hyperparamètres optimaux pour l'entraînement final du modèle a eu lieu. Celle-ci fût effectuée sur des GPUs de type Nvidia k80. La recherche fût effectuée en parallèle sur les ordinateurs de tous les membres de l'équipe pendant approximativement 10 heures. La recherche fut donc aléatoire afin d'assurer une parallélisation non biaisée de celle-ci. Au total, près de 50 combinaisons furent étudiées et l'échantillonnage fut un entraînement d'environ 5 ans d'activités dans la simulation. La combinaison d'hyperparamètres ayant engendré le meilleur *Reward* après ledit entrainement d'échantillonnage est présentée au *tableau 3*.

Type	Intervalles de recherche	Valeur
Batch size	[8, 16, 32, 64]	8
No. of epoch	[5, 10, 15, 20]	15
λ	[0.25, 0.5, 075]	0.75
Learning rate	[1e-5 : 1e-2]	0.65e-2

Tableau 3 : Intervalles de recherche et valeur sélectionnées d'hyperparamètres

4.4) Baseline

Trois heuristiques ont été développées pour être comparées à l'agent ainsi que pour s'assurer d'avoir des paramètres de simulations où il est réaliste de garder le contrôle de la cour. Voici un résumé des heuristiques:

1) Heuristique aléatoire : Piger une règle au hasard parmi les règles ayant du bois prêt à être transporté.

2) Pile la plus élevée : Choisi l'action qui représente la règle qui dépasse le plus l'intervalle de proportions de produits désirée dans la cour (Ligne verte sur les graphiques tel que présenté à la figure 5).

3) Gestion horaire et pile : Même heuristique que la pile la plus élevée, mais s'assure de ne pas faire entrer d'épinette le vendredi après 15h car celle-ci sortirait durant la fin de semaine et nous n'avons pas suffisamment de temps pour préparer le prochain wagon avant la fin de semaine.

Dans le cas où nous aurions réalisé un travail complet sur une version non-simplifié du problème avec une fonction objective reliée à la satisfaction de la demande et des ordres de clients, un bon heuristique utilisé par les professionnels académiques aurait été celui proposé par *Gaudreault et al.* (2010) [5], car celui-ci s'applique à la version de notre problème, c'est à dire lorsque des règles de chargement prédéfinies sont fournies.

V. RÉSULTATS

Afin de comparer de façon uniforme les résultats, nous avons regardé les résultats des heuristiques et de notre meilleur modèle final, tous sur des simulations de la même nature (cour moyennement facile à gérer). Vingt réplications ont été menées avec chaque algorithmes et nous avons utilisé un niveau de confiance à 95%. Le *Reward* moyen est utilisé comme métrique de comparaison puisque celui-ci prend déjà les différents objectifs en compte selon leur importance. L'utilisation des séchoirs est aussi comparée puisque c'est une préoccupation importante par les gestionnaires de cour à bois.

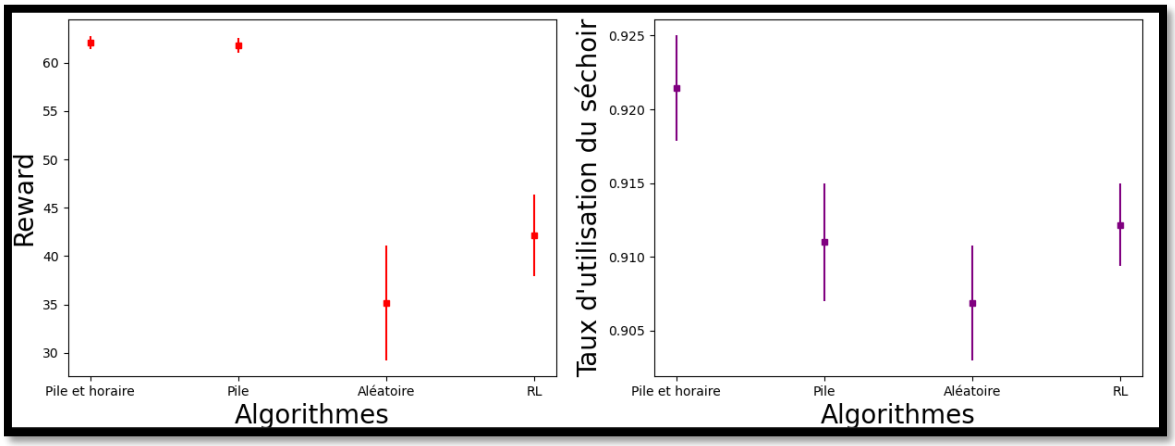

Figure 6 : Intervalles de confiances (95%) du reward et de l'utilisation du séchoir des différents algorithmes

On constate, sans grande surprise, que les heuristiques ont des meilleurs résultats que l'aléatoire puisqu'ils sont directement développés dans le but de répondre à l'objectif principal. On voit que l'intervalle de confiance de l'heuristique qui considèrent le temps semble légèrement supérieure à celle qui ne le considère pas, mais puisqu'il y a chevauchement des intervalles, nous ne pouvons pas conclure qu'elle est meilleure. Cependant, on peut considérer que l'ajout de la gestion du temps ne nuit pas à l'heuristique afin de répondre à la fonction objective. En regardant l'intervalle de confiance de l'utilisation du séchoir, on peut conclure avec 95% de confiance que d'éviter d'envoyer de l'épinette en fin de journée le vendredi mène à une meilleure utilisation des séchoirs. Cela démontre bien que cette heuristique est un bon point de comparaison pour notre algorithme.

On constate que l'intervalle de confiance de notre meilleur modèle chevauche celui de l'aléatoire dans la partie supérieure. En fait, un niveau de confiance de 70% permettrait d'éviter ce chevauchement. Nous sommes donc convaincus que notre agent a commencé à apprendre à bien gérer la cour, mais qu'il a manqué de temps d'entraînement pour maximiser ses résultats. En effet, notre agent a convergé très vite avec la simulation de départ, mais il fut difficile de le faire converger avec la simulation finale qui est plus complexe nous laissant donc peu de temps pour un entraînement final long et complet.

VI. DISCUSSION

Des modèles d'apprentissage par renforcement comme celui-ci, appliqués à des versions non-simplifiées du problème peuvent servir à automatiser une bonne partie de la prise de décision d'usine 4.0 car les allocations optimales générées par des agents entrainés en simulation peuvent servir directement à générer les ordres de contrôle aux différentes machines telles que les camions-chargeurs. Bien-sûr, à quel degré l'environnement de la simulation utilisé est comparable au vrai système modélisé reste la variable clé dans les systèmes d'automatisation du genre. En ce sens, nous avons convergé avec des résultats similaires à l'heuristique sur la première version de la simulation, nous encourageant grandement pour la suite des travaux. Cependant, au fur et à mesure où l'on rendait la simulation plus réaliste (horaires, wagons, séchage à l'air libre), l'algorithme devenait de plus en plus couteux et difficile à entrainer. Tellement, que nous en sommes rendus à un point où il n'apprenait plus vraiment dans un temps d'entraînement gérable dans les limites des temps alloués pour le projet.

Différentes solutions ont été essayées afin d'avoir de meilleurs résultats. En ce sens, le premier test fut d'entrainer l'algorithme en le récompensant ou le pénalisant selon s'il prenait la même décision que l'heuristique, le tout dans le but de lui donner une intuition de départ. Il fut surprenant de constater que nous n'avons pas eu de gain significatif lors de cette tentative. Nous avons tout de même conservé l'idée en ajoutant une petite récompense dans la fonction objective (*Heuristic guided RL*). Nous avons aussi testé l'idée d'utiliser seulement 4 règles au lieu de 13 afin de simplifier le problème et d'ajouter des informations supplémentaires dans l'état pour l'aider dans sa prise de décision. Bien que ces changements ne nous aient pas apporté de gains significatifs, nous croyons encore qu'il reste de la place pour l'optimisation des informations passées à notre agent.

Finalement, la modification de la fonction objective fut le changement qui a apporté le plus grand gain. En effet, nous avons réduit les pénalités pour pénaliser moins fort lors de l'apprentissage en enlevant le carré dans la pénalité (*Tableau 2*). Nous avons remarqué que l'agent avait souvent tendance à conserver très bas le niveau d'inventaire de certains produits puisqu'il était aussi payant pour lui de respecter le niveau souhaité d'inventaire (ligne rouge, *Figure 5*) que les proportions souhaitées selon l'inventaire en cours (lignes vertes, *Figure 5*). Puisqu'il était prioritaire de conserver les proportions, nous avons fait la racine carrée pour le respect de l'inventaire souhaité. Ces deux modifications nous ont permis de revenir à un agent qui réussit à apprendre, mais malheureusement, puisque les entraînements sont très longs, le temps a manqué pour finaliser un modèle vraiment performant.

Dans l'éventualité où le modèle ne rencontre toujours pas la performance de l'heuristique, nous envisagions d'ajouter des hyperparamètres pour les différents objectifs du *Reward* afin d'effectuer une recherche d'hyperparamètres pour trouver la bonne nuance entre les pénalités et les récompenses. Nous aurions aussi continué l'optimisation des informations passées à notre agent. Le tout, en lui passant d'avantage d'indicateurs

de performances représentant, par exemple, à quel point le niveau des règles dans la cour s'approche du niveau optimal. L'hypothèse étant de lui passer ces indicateurs au lieu de l'état réel de la cour il aurait été capable de faire un lien plus rapide entre l'action, l'observation et le *Reward,* convergeant ainsi plus rapidement vers une solution jugée idéale.

Si au contraire, le modèle performerait très bien, des composantes primordiales à ajouter à ce modèle, seraient d'ajouter d'autres fonctions objectives (telles que de la demande pour chaque produit à travers la simulation, l'utilisation des séchoirs et de la scierie), les variations dues aux saisons et les bris d'équipements. Également, la génération d'affectation aux stations de séchages pourrait être faite de façon sur demande (par paquets ou arrimes), contrairement aux choix de règles qui font abstraction d'une bonne partie des composantes de la simulation. Cela deviendrait donc un problème d'optimisation complexe avec un espace de recherche devenant très large à la moindre augmentation du nombre de produits différents contenus dans l'état de la simulation à chaque itération. De plus, utiliser une architecture d'apprentissage profond avec beaucoup plus de capacité serait quelques choses qui aurait comme impact une amélioration importante des capacités de notre ou nos agents.

Enfin, le fait que les résultats n'aient pas été à la hauteur de nos attentes témoigne de la complexité d'entraîner un algorithme d'apprentissage par renforcement. Les principaux défis rencontrés lors du projet furent la conception d'une observation et d'un Reward permettant à l'algorithme d'apprendre les allocations optimales en fonction du temps et des horaires. Ce casse-tête est complexe, car l'observation doit être statique en espace tout en contenant un minimum de bruit. De plus, le Reward doit être suffisamment abondant pour renforcir une bonne politique, mais pas trop pour éviter d'obtenir un algorithme avare et d'ainsi hypothéquer la recherche. Bref, l'apprentissage par renforcement reste une avenue intéressante, mais coriace pour résoudre le problème d'allocation de chargements aux séchoirs dans une cour à bois.

VII. CONCLUSION

Nous avons initié ce projet dans le but d'étudier la possibilité d'appliquer l'apprentissage par renforcement jumelé à l'outil de la simulation au problème d'ordonnancement des opérations de séchage du bois industrielle afin de stabiliser dynamiquement les proportions des différents produits dans une cour d'inventaire. Malgré de nombreux défis, nous avons produit des résultats pour les instances où nous avons gardé la complexité du problème à son plus bas. Mais plus nous rajoutions des variables rendant le problème plus complexe, plus l'entrainement devenait long et l'impact sur la performance de l'agent se faisant ressentir. En raison du temps et des ressources limités, nous avons été limités dans notre capacité à analyser plus en détail comment améliorer les capacités de notre agent. Il se peut également que le choix d'une fonction objective différente de celle utilisé classiquement dans l'analyse de ce problème fût un mauvais choix. Malgré cela, tel qu'illustré à la Figure 5, il a été possible de converger rapidement vers une solution idéale lors des premières itérations du problème (Tableau 2), quand l'algorithme n'avait pas à tenir compte des variables supplémentaires tels que les horaires de travail pour prendre des décisions. Enfin, tel que mentionné dans la discussion, il pourrait être intéressant de pousser la recherche plus loin, soit en complexifiant le problème d'affectation en faisant fi des règles ou bien en ajoutant d'autres objectifs au problème.

Nous remercions l'aide précieuse de Mr. François Léger de PMP Solutions (https://pmpsolutions.ca/en/) ainsi que de notre professeur Mr. Jonathan qui nous ont aidé à converger vers un problème simplifié mais réaliste qui était tout de même faisable en tant que projet de session.

VIII. RÉFÉRENCES

[1] Job Scheduling in High Performance Computing, Yuping Fan. Illinois Institute of Technology. https://arxiv.org/pdf/2109.09269.pdf

[2] Marcel Panzer & Benedict Bender (2021): Deep reinforcement learning in production systems: a systematic literature review, International Journal of Production Research, DOI: 10.1080/00207543.2021.1973138

[3] Sylvain Chabanet, Hind Bril El-Haouzi, Michael Morin, Jonathan Gaudreault & Philippe Thomas (2022): Toward digital twins for sawmill production planning and control: benefits, opportunities, and challenges, International Journal of Production Research, DOI: 10.1080/00207543.2022.2068086

[4] Philippe Marier, Jonathan Gaudreault, Thomas Noguer (2021. *Kiln Drying Operations Scheduling with Dynamic Composition of Loading Patterns*. Forest Products Journal Vol. 71, No. 2.

[5] Gaudreault, J., P. Forget, J. M. Frayret, A. Rousseau, S. Lemieux, and S. D'Amours. 2010. Distributed operations planning in the softwood lumber supply chain: Models and coordination. Int. J. Ind. Eng.: Theory Appl. Practice 17(3):168–189.

[6] Maria Anna Huka, Christian Rindler, Manfred Gronalt. *Scheduling and loading problem for multiple, identical dry kilns.* URL: https://link.springer.com/content/pdf/10.1007/s10696-020-09386-4.pdf

[7] Gaudreault, J., J. M. Frayret, A. Rousseau, and S. D'Amours. 2011. Combined planning and scheduling in a divergent production system with co-production: A case study in the lumber industry. Comput. Operations Res. 38:1238–1250.

[8] Rasmussen, E.F. 1961. Dry kiln operator's manual. Agric. Handb. 188. Washington, DC: U.S. Department of Agriculture. 197 p. *https://www.fpl.fs.fed.us/documnts/usda/ah188/chapter05.pdf*

[9] Neural Combinatorial Optimization With Reinforcement Learning. Irwan Bello, Hieu Pham, Quoc V, Le, Mohammad Norouzi, Samy Bengio.Google Brain. URL: *https://arxiv.org/pdf/1611.09940.pdf*

[10] Proximal Policy Optimization Algorithms, John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov. URL: https://arxiv.org/pdf/1707.06347.pdf

[11] Ching-An Cheng, Andrey Kolobov, Adith Swaminathan. *Heuristic-Guided Reinforcement Learning*. URL : *https://openreview.net/pdf?id=HipwnJKnp3*