Faculté des sciences et de génie

Optimisation Combinatoire
IFT-7020
Hiver 2020

Optimisation Combinatoire : Projet de recherche

Identification d'influenceurs dans les processus épidémiques à l'aide d'apprentissage par renforcement

présenté à

M. Claude-Guy Quimper

par

Matricule	*Nom*
111 073 283	Yoan Chamberland
111 184 507	Catherine Villeneuve

Université Laval
1 mai 2020

1. Introduction

L'étude des dynamiques de diffusion dans les réseaux de contacts permet de mieux saisir comment une maladie infectieuse peut se propager au sein d'une population en considérant l'*ensemble des interactions entre les individus*. En effet, à l'aide de modèles théoriques issus de l'épidémiologie mathématique et de notions tirées de la théorie des graphes, il est possible d'identifier les quelques individus essentiels à partir desquels l'ensemble de la dynamique collective du réseau peut être représentée. Au sein d'un processus épidémique de durée t, ces *influenceurs* peuvent être déclinés en trois catégories [1]:

1. Les « **super-propagateurs** », correspondant aux individus qui, s'ils sont infectés en premier, maximiseraient l'espérance du nombre de personnes infectées lorsque $t \to \infty$
2. Les « **super-bloqueurs** », correspondant aux individus qui, s'ils sont immunisés ou isolés en premier, minimiseraient l'espérance du nombre de personnes infectées lorsque $t \to \infty$
3. Les « **sentinelles** », correspondant aux individus les plus susceptibles d'être infectés en premier lors de l'éclosion d'une épidémie quelconque

Le présent rapport s'intéresse particulièrement à la seconde catégorie, en tentant de résoudre le problème suivant : *Étant donné un processus épidémique à temps discret se propageant sur un graphe G, quels sont les k individus qui doivent être immunisés en premier afin de minimiser la propagation de la maladie?*

Nous nous intéresserons particulièrement aux processus épidémiques dit *irréversibles*, c'est-à-dire qui atteignent éventuellement un état d'équilibre. C'est le cas des processus épidémiques modélisant la transmission des maladies dont on peut développer une immunité à la suite d'une première infection, tel que la varicelle ou la rougeole. Plus précisément, nous considérerons une version discrétisée du modèle épidémique compartimental *Susceptible-Infected-Recovered* (SIR), tel que présenté dans [13]. Ce processus aléatoire, de même que les autres composantes de notre problème, seront décrits à la section 2.

La formulation d'une instance et d'une solution valide pour le problème de l'immunisation optimale dépend de si l'on souhaite considérer une cardinalité k fixée ou non. Par conséquent, nous proposerons un modèle distinct pour chacune de ces deux variantes. Ces derniers, qui seront construits à partir de la théorie derrière l'état de l'art actuel [4], seront présentés à la section 3. Puisque nos modèles ne peuvent être résolus de manière exacte que sur de petites instances, l'approche favorisée pour résoudre le problème de l'immunisation optimale est d'avoir recours à une *heuristique vorace*. Nous proposerons une nouvelle manière de construire une telle heuristique à l'aide d'une politique de sélection basée sur l'*apprentissage par renforcement profond*. Elle sera présentée en détails à la section 3.3.1. Nous comparerons sa performance à celles de trois algorithmes présents dans la littérature, soit CI, Highest Degree et PageRank. Nous considérerons des instances de différentes tailles générées à partir du modèle de graphe aléatoire Barabási–Albert. Nos résultats révèleront que notre approche est prometteuse, notamment pour la résolution de notre premier modèle, où les algorithmes d'apprentissage parviendront à surpasser l'état de l'art actuel.

2. Description du problème

2.1 Identification de l'ensemble minimal de sommets à immuniser

Cette première façon de formuler le problème ne considère pas la cardinalité k de l'ensemble d'invidus à immuniser comme fixée et est équivalente à la formulation du problème de la *percolation optimale* dans un graphe, qui revient à identifier l'ensemble minimal de sommets à retirer du réseau pour le fragmenter en différentes composantes déconnectées entre elles [4]. Une représentation visuelle de la percolation optimale est présentée à la figure 1.

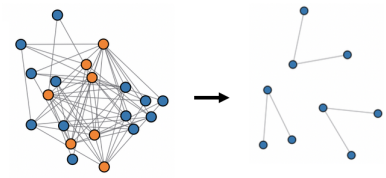

Figure 1 – Solution au problème de la percolation optimale pour un graphe de 20 sommets de type Barabási–Albert. Le graphe de droite illustre les connections restantes lorsque le sous-ensemble de sommets minimal (ceux en orange) est retiré du graphe.

La première stratégie de prévention employée afin de limiter la propagation d'une maladie infectieuse est de *briser* le réseau de contacts entre les individus. De cette manière, l'épidémie peut être plus facilement contenue en la limitant seulement à certaines communautés. La figure 1 illustre bien ce principe; il est beaucoup plus aisé de limiter la propagation d'une épidémie et d'intervenir efficacement dans le réseau de droite que dans celui de gauche.

Le problème de la percolation optimale est purement structurel et ne dépend d'aucun processus aléatoire. La meilleure stratégie pour le résoudre consiste à s'attaquer aux cycles du réseau et à retirer tous les arcs des sommets qui peuvent en briser le plus, jusqu'à ce que le graphe obtenu ne soit constitué que d'un ensemble d'arbres et de sommets isolés. Par conséquent, la solution du problème de l'immunisation optimale en considérant strictement la structure d'un graphe $G = (V, E)$ donné correspond donc au sous-ensemble $S \subseteq V$ de cardinalité k minimale tel que S parvient à briser tous les cycles.

2.2 Identification des k meilleurs sommets à immuniser

Lorsqu'on s'intéresse à identifier les k meilleurs sommets à immuniser pour une cardinalité k préalablement fixée, on peut voir ce problème comme une variante de la *maximisation de l'influence* dans les réseaux sociaux [10]. Dans ce problème, on demande, pour un processus aléatoire à temps discret, un graphe $G = (V, E)$ et un paramètre $k \in \mathbb{N}^+$ donné, d'identifier le sous-ensemble de k sommets d'influence maximale. On défini l'influence d'un sous-ensemble de sommets S, dénotée $\sigma(S)$, comme étant l'espérance du nombre de sommets influencés à l'équilibre lorsque S constitue l'ensemble des sommets actifs de départ. Kempe *et al.* démontrent dans [10] qu'il s'agit d'un problème NP-Difficile pour deux processus de diffusion simples, soit les modèles LTM (*Linear Threshold Model*) et ICM (*Independent Cascade Model*). Il est possible de démontrer que le problème est également NP-Difficile pour un processus de diffusion plus complexe - tel qu'un processus épidémique - à l'aide d'une réduction sur l'un ou l'autre de ces modèles [11].

L'adaptation du problème de la maximisation de l'influence au problème de l'immunisation optimale consiste simplement à définir S comme l'ensemble des sommets immunisés au départ et à chercher à minimiser $\sigma(S)$ plutôt qu'à le maximiser. On peut considérer que la source de l'épidémie est fixe ou aléatoire. La solution du problème est donc l'ensemble $S \subseteq V$ de cardinalité k parvenant à minimiser $\sigma(S)$.

2.3 Le modèle *Susceptible-Infected-Recovered* (SIR)

Soit $G = (V, E)$ un graphe non orienté dont l'ensemble des sommets est $V = \{1, ..., n\}$ et l'ensemble des arêtes est $E \subset V \times V$. Le couple $(i, j) \in E$ est interprété comme une arête allant de i à j.

À chaque instant t, un individu $i \in V$ peut être dans l'un des trois états suivants : susceptible ($\mathcal{S}$), infecté ($\mathcal{I}$) ou immunisé ($\mathcal{R}$). L'état du noeud i au temps t est représenté par la variable $x_i^t \in \{\mathcal{S}, \mathcal{I}, \mathcal{R}\}$. On assume qu'à l'instant t_0, il existe un certain triplet ($\mathcal{S}_0$, $\mathcal{I}_0$, $\mathcal{R}_0$) d'individus infectés, susceptibles et immunisés au départ.

Soit $E_i = \{ j \,|\, (i,j) \in E\}$ le voisinage d'un sommet i. À chaque épisode, un individu infecté i peut d'abord contaminer chacun de ses voisins $j \in E_i$ avec une probabilité $T_{ij} \in (0,1]$, puis guérit avec une probabilité $r_i \in (0,1]$. Une fois remis de la maladie, l'individu est considéré comme immunisé ($\mathcal{R}$) et ne peut plus la transmettre. C'est l'équivalent d'être retiré du graphe. La probabilité p_{ij} qu'un individu infecté i transmette directement la maladie à j avant que i ne soit immunisé est donnée par

$$p_{ij} = \frac{T_{ij}}{T_{ij} + (1 - T_{ij})r_i} \tag{1}$$

Soit $\partial_i = \{ j \,|\, (i,j) \in E \wedge x_j^{t-1} = \mathcal{I}\}$ l'ensemble des voisins infectés d'un individu i à l'instant $t-1$. La probabilité $p(x_i^t = \mathcal{I} | x_i^{t-1} = \mathcal{S}, \partial_i)$ qu'il devienne infecté à l'instant t considérant l'état de son voisinage est donnée par

$$p(x_i^t = \mathcal{I} | x_i^{t-1} = \mathcal{S}, \partial_i) = 1 - \prod_{j \in \partial_i} p_{ij} \tag{2}$$

c'est donc dire que la probabilité que l'état d'un sommet i passe de susceptible au temps $t-1$ à infecté au temps t dépend uniquement de l'état au temps $t-1$ du voisinage de i, mais ne dépend pas de l'état des autres sommets du graphe. Une fois infecté, la probabilité que le sommet i le demeure au temps $t+1$ est simplement donnée par $1 - r_i$.

3. Approche proposée

Nous formulerons d'abord un modèle construit à partir de la description du problème présentée à la section 2.2, c'est-à-dire que nous chercherons à identifier l'ensemble minimal de sommets S permettant de briser tous les cycles d'un graphe G. Ce modèle sera décrit à la section 3.1. Nous formulerons par la suite à la section 3.2 un modèle qui permet de résoudre le problème pour un budget k préalablement fixé. Enfin, la section 3.3 présentera les différents algorithmes qui seront utilisés pour résoudre le problème. Nos modèles sont construits à partir de la théorie derrière l'état de l'art actuel, soit l'heuristique présentée dans [4].

3.1 Modèle 1

Il existe plusieurs manières de représenter les liens entre les sommets d'un graphe G. La plus intuitive est la matrice d'adjacence $\mathcal{A}$ de dimension $n \times n$ dont l'entrée (i,j) vaut 1 si $(i,j) \in E$ et 0 autrement. On peut également représenter la géométrie de G à l'aide de la *matrice sans épine* (*non-backtracking matrix*) [5], dite $\mathcal{B}$, une matrice carrée asymétrique moins connue de dimension $2E \times 2E$ dont la définition formelle est la suivante:

$$\mathcal{B}_{k\to\ell, i\to j} = \delta_{i,\ell}(1 - \delta_{j,k}) \tag{3}$$

où $\delta_{i,j} = 1$ si $i = j$, 0 autrement. La matrice $\mathcal{B}$ possède de nombreuses propriétés spectrales intéressantes. Entre autres, l'étude de la deuxième plus grande valeur propre a notamment permis de développer des algorithmes de détection de communautés [8] [9]. Pour le problème de l'immunisation optimale, la valeur propre *maximale* de $\mathcal{B}$, dite λ_{max}, est particulièrement d'intérêt. En effet, lorsque qu'un graphe G donné n'est constitué que d'un ensemble d'arbres binaires et de sommets isolés, alors $\lambda_{max} = 0$ [7]. On peut utiliser cette information afin de définir une variante de $\mathcal{B}$, dite $\mathcal{M}$, dont la définition est la suivante [4] :

$$\mathcal{M}_{k\to\ell, i\to j} = n_i \mathcal{B}_{k\to\ell, i\to j} \tag{4}$$

où n_i est une variable binaire de valeur 1 lorsque le sommet i n'est pas immunisé et 0 autrement. La matrice $\mathcal{M}$ est donc tout simplement une matrice sans épine dérivée de $\mathcal{B}$ représentant la géométrie d'un graphe obtenu en retirant tous les sommets i tel que $n_i = 0$.

Puisque $\lambda_{max}(\mathcal{B})$ est toujours simple et positive [6] et que $\mathcal{M} \leq \mathcal{B}$, si $\mathcal{M} \neq \mathcal{B}$, alors $\lambda_{max}(\mathcal{M}) < \lambda_{max}(\mathcal{B})$.

Par conséquent, en retirant graduellement les sommets qui diminuent le plus $\lambda_{max}(\mathcal{M})$, on obtient éventuellement une matrice $\mathcal{M}'$ telle que $\lambda_{max}(\mathcal{M}') = 0$ et le problème est alors résolu. Puisque nous cherchons à immuniser un nombre minimal k de sommets, le problème d'optimisation peut être formulé de la manière suivante :

$$\text{(P1)} = \begin{cases} \textbf{minimiser} & k \\ \text{sujet à} & k = \sum_{i \in V}(1 - n_i) \\ & \mathcal{M}_{k\to\ell,i\to j} = n_i \mathcal{B}_{k\to\ell,i\to j} \quad \forall (k,\ell),(i,j) \in E \\ & \mathcal{B}_{k\to\ell,i\to j} = \delta_{i,\ell}(1 - \delta_{j,k}) \quad \forall (k,\ell),(i,j) \in E \\ & \lambda_{max}(\mathcal{M}) < \epsilon \end{cases} \tag{5}$$

où $\epsilon \in (0, 1)$. Puisque $\lambda_{max}(\mathcal{M})$ est calculé à l'aide d'une méthode numérique, le fait de contraindre cette valeur à être arbitrairement proche de 0 permettra de prévenir d'éventuelles instabilités et de toujours parvenir à une solution. Pour nos expérimentations, nous fixerons $\epsilon = 1 \cdot 10^{-4}$.

Notre modèle nécessite la résolution de $\Theta(E^2)$ contraintes. Il contient également $\Theta(E^2)$ variables, soit une variable $k \in [0, n]$, une variable variable réelle $\epsilon \in (0, 1)$, n variables binaires n_i et $8E^2$ variables binaires pour représenter les éléments des matrices $\mathcal{B}$ et $\mathcal{M}$. Cela constitue un total de $\Theta(E^2)$ valeurs.

3.2 Modèle 2

Les résultats empiriques de [1] et analytiques de [2] suggèrent que la solution au problème de l'immunisation optimale pour un certain graphe G serait davantage déterminée par sa topologie que par la dynamique de propagation. En effet, il semblerait que pour minimiser le nombre de personnes infectées lors d'un processus épidémique, peu importe la virulence de la maladie, l'essentiel serait de briser le plus possible le réseau de contacts entre les individus. C'est donc dire que les sommets minimisant la valeur de $\lambda_{max}(\mathcal{M})$ du modèle précédent constitueraient toujours une bonne solution.

Nous formulerons donc une adaptation de notre premier modèle pour traiter la situation où on souhaite identifier les k meilleurs sommets à immuniser pour une cardinalité k préalablement fixée :

$$\text{(P2)} = \begin{cases} \textbf{minimiser} & \lambda_{max}(\mathcal{M}) \\ \text{sujet à} & k = |S| \\ & |S| = \sum_{i \in V}(1 - n_i) \\ & \mathcal{M}_{k\to\ell,i\to j} = n_i \mathcal{B}_{k\to\ell,i\to j} \quad \forall (k,\ell),(i,j) \in E \\ & \mathcal{B}_{k\to\ell,i\to j} = \delta_{i,\ell}(1 - \delta_{j,k}) \quad \forall (k,\ell),(i,j) \in E \end{cases} \tag{6}$$

Ce modèle est de complexité équivalente à celle du premier. Bien qu'il s'agisse d'une manière entièrement statique de considérer le problème, nous pensons qu'elle peut potentiellement rivaliser avec des modèles qui cherchent directement à minimiser $\sigma(S)$, soit l'espérance du nombre de sommets ayant été infectés au moins une fois à l'équilibre d'un processus épidémique. En effet, le calcul de $\sigma(S)$ étant en lui-même un problème NP-Difficile [14], on ne peut pas espérer pouvoir utiliser cette stratégie sur de grandes instances.

3.3 Processus de résolution

L'énumération des solutions candidates permet de résoudre de manière exacte le problème pour les modèles décrits aux sections 3.1 et 3.2. Il s'agit de la seule stratégie permettant d'obtenir le véritable optimum avec certitude. Pour un graphe de n sommets, cela correspond à l'évaluation

en pire cas de $\binom{n}{0} + \binom{n}{1} + ... + \binom{n}{n} = 2^n$ solutions candidates pour le premier modèle et $\binom{n}{k}$ solutions candidates pour le deuxième. La taille de l'espace des solutions, autant pour le premier modèle que pour le second, rend rapidement impossible la résolution du problème de manière exacte sur des instances réelles de taille considérable. Par exemple, pour le premier modèle, un graphe de seulement 30 sommets nécessite déjà l'exploration en pire cas de plus d'un milliard de solutions potentielles. Il importe alors de s'orienter vers des *heuristiques de recherche* permettant d'obtenir une solution satisfaisante en un temps raisonnable.

L'heuristique traditionnellement utilisée pour le problème de l'immunisation optimale est d'avoir recours à un *algorithme vorace*. Son fonctionnement est simple ; à chaque itération, un classement des sommets est effectué à partir d'un calcul préalablement déterminé, puis le sommet en tête du classement est ajouté à la solution partielle. L'algorithme prend fin lorsque toutes les contraintes du problème sont respectées.

L'approche que nous proposons est de développer de nouvelles stratégies voraces pour le problème de l'immunisation optimale à l'aide d'*apprentissage par renforcement profond*. Elle sera détaillée à la section 3.3.1. Ces nouvelles heuristiques seront comparées à celles déjà présentes dans la littérature, qui seront brièvement résumées à la section 3.3.2.

3.3.1 Développement d'heuristiques à l'aide d'apprentissage par renforcement profond

Un algorithme d'apprentissage par renforcement est un processus de contrôle stochastique à temps discret pouvant être résumé à l'aide du scénario typique suivant [20] : à chaque épisode t, un agent reçoit l'état actuel d'un système et la récompense associée à son état précédent. Il détermine ensuite la prochaine action à effectuer, qui est envoyée au système. Le système effectue alors une transition vers un nouvel état et le cycle est répété jusqu'à ce que le processus de contrôle prenne fin. L'objectif d'un algorithme d'apprentissage par renforcement est d'apprendre, par essais-erreurs, un « bon comportement », c'est-à-dire une *politique* lui permettant de maximiser la récompense cumulative à la fin d'un processus de contrôle donné.

Plus formellement, un algorithme d'apprentissage par renforcement est généralement défini à l'aide d'un processus de décision markovien (MDP) [19], soit un tuple $(\mathcal{S}, \mathcal{A}, T, R)$ où $\mathcal{S}$ est l'espace d'états, $\mathcal{A}$ est l'espace d'actions, $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ est la fonction de transition entre les états et $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ est la fonction de récompense, associant une valeur réelle à une transition donnée.

Il est possible de définir un algorithme de minimisation vorace dans un graphe G à l'aide de ce formalisme. La définition de chacune des composantes du MDP est alors la suivante [17]:

- **État :** Un état $s_t \in \mathcal{S}$ est une séquence d'actions (sommets) dans le graphe. Les sommets dans s_t sont ceux faisant partie de la solution partielle à l'instant t.
- **Action :** Une action $a_t \in \mathcal{A}$ correspond à un sommet $u \in V \backslash s_t$
- **Transition :** Une transition correspond à l'ajout du sommet a_t à la solution partielle, de façon à obtenir $s_{t+1} \leftarrow s_t \cup \{a_t\}$
- **Récompense :** Soit $f(S)$ la fonction objectif considérée. La fonction de récompense $r(s_t, a_t)$ est définie par le changement dans la fonction objectif à la suite d'une transition, soit :

  $$r(s_t, a_t) = f(s_t) - f(s_{t+1})$$

  Pour notre problème, nous considérerons la fonction de récompense $r(s_t, a_t) = -1$ pour le premier modèle et $r(s_t, a_t) = \lambda_{max}(\mathcal{M}_t) - \lambda_{max}(\mathcal{M}_{t+1})$ pour le deuxième.

La politique $\pi(s) : \mathcal{S} \rightarrow \mathcal{A}$ est la stratégie déterminant quel sommet ajouter à la solution partielle considérant l'état actuel de la fonction objectif. Soit $Q^{\pi}(s,a) : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ la *fonction d'évaluation* de $\pi(s)$ déterminant la pertinence d'ajouter un sommet à la solution partielle. Les heuristiques présentées à la section 3.3.2 sont des exemples de telles fonction d'évaluation. La politique vorace est toujours la même, soit :

$$\pi(s) = \underset{a \in \mathcal{A}}{\operatorname{argmax}} Q^{\pi}(s,a)$$

L'objectif d'un algorithme d'apprentissage par renforcement est d'identifier la *meilleure* politique, c'est-à-dire celle permettant de minimiser le plus possible la fonction objectif. Soit S_{π} la solution obtenue à l'aide de la fonction d'évaluation propre à une politique π. Il s'agit donc d'identifier la politique $\pi^* \in \Pi$ telle que

$$\pi^* = \min_{\pi \in \Pi} f(S_{\pi})$$

La difficulté de cette tâche réside dans l'identification de la fonction d'évaluation optimale $Q^*(s,a)$. L'approche que nous proposons pour ce faire est d'adapter à notre problème l'algorithme S2V-DQN (*structure2vec-Deep Q Network*) présenté dans [17]. Notre politique de sélection est la suivante :

$$\pi(s) = \begin{cases} \operatorname{argmax}_{a \in \mathcal{A}} Q(s,a,\Theta) & \text{avec probabilité } 1-\epsilon \\ rand(a \in V \backslash s) & \text{autrement} \end{cases} \tag{7}$$

où Θ correspond à l'ensemble des poids d'un réseau de neurones (un perceptron multicouche). C'est donc dire que la fonction d'évaluation $Q(s,a,\Theta)$ est *paramétrée* par un réseau de neurones profond ayant, on l'espère, appris à approximer $Q^*(s,a)$. Ce perceptron multicouche accepte entre autres en entrée $|V \backslash s|$ vecteurs p-dimensionnels produits en sortie par une adaptation de $structure2vec$ [18], un réseau de neurones en mesure de transformer un graphe $G = (V,E)$ en une représentation vectorielle (*graph embedding*) telle que, $\forall i \in V$, un vecteur $\mu_i \in \mathbb{R}^p$ d'abord initialisé à 0, puis est actualisé itérativement de la matière suivante :

$$\mu_i^{(t+1)} \leftarrow ReLU\left(ReLU(\theta_1 x_i) + \theta_2 \sum_{j \in E_i} \mu_j^{(t)}\right) \tag{8}$$

où E_i correspond à l'ensemble des voisins du sommet i et $x_i = [1,1]$, $\forall i \in V$. Après T itérations, un vecteur μ_i contiendra donc les informations de tous les sommets accessibles à l'aide d'un chemin de longueur inférieure ou égale à T à partir de i. De plus, le réseau de neurones accepte également en entrée un vecteur $\phi \in \mathbb{R}^3$, représentant l'état actuel de la fonction objectif :

Modèle	a
1	$\left[\frac{s_t}{\lvert V \rvert}, \frac{\lambda_{max}(\mathcal{B}) - \lambda_{max}(\mathcal{M})}{\lambda_{max}(\mathcal{B})}, 1\right]$
2	$\left[\frac{s_t}{k}, \frac{\lambda_{max}(\mathcal{B}) - \lambda_{max}(\mathcal{M})}{\lambda_{max}(\mathcal{B})}, 1\right]$

Table 1 – Vecteurs ϕ

À partir de toutes ces entrées, la fonction $Q(s,a,\Theta)$ est calculée en sortie de la manière suivante :

$$Q(s,a,\Theta) = \theta_3^{\top}\left[ReLU(\left[\theta_4 \sum_{u \in V} \mu_u^T, \theta_5 \mu_v^T\right], \phi\right] \tag{9}$$

La fonction d'évaluation $Q(s,a,\Theta)$ dépend donc d'un ensemble de cinq paramètres $\Theta = \{\theta_i\}_{i=1}^5$. Ces derniers seront appris par descente de gradient stochastique, de manière à minimiser l'erreur quadratique moyenne suivante :

$$((y - Q(s_t, a_t, \Theta))^2 \tag{10}$$

où $y = \sum_{i=0}^{n-1} r(s_{t+i}, a_{t+i}) - \max_{a \in \mathcal{A}} Q(s_{t+n}, a, \Theta)$. La variable $n \in \mathbb{N}$ ($n \geq 2$) est un hyperparamètre contrôlant la fréquence à laquelle les poids du réseau de neurones sont mis à jour. Le fait d'attendre n itérations avant d'actualiser Θ possède l'avantage de rendre les prédictions de l'algorithme plus robuste en le contraignant à considérer les récompenses futures [21]. Enfin, plutôt que d'être effectuée seulement à partir de l'instance actuelle, la mise à jour de Θ sera réalisée à partir d'un échantillon aléatoire des expériences précédentes. Le principe est le suivant ; un jeu de données E est rempli au fil de l'entraînement de manière à ce que, à l'épisode $t+n$ de la résolution d'une instance donnée, le tuple $(s_t, a_t, R_{t,t+n}, s_{t+n})$ soit ajouté à E, avec $R_{t,t+n} = \sum_{i=0}^{n-1} r(s_{t+i}, a_{t+1})$. Un échantillon aléatoire de B tuples est alors utilisé pour actualiser Θ à partir de l'équation 10. Le pseudocode de cet algorithme, de même que les valeurs des hyperparamètres qui ont été sélectionnées pour chacun de nos modèles, sont disponibles aux annexes B et C.

3.3.2 Heuristiques de recherche proposées dans la littérature

Nous comparerons notre approche avec les algorithmes voraces suivants :

1. **Collective Influence (CI) [4] :** C'est l'état de l'art actuel. Le classement des sommets est réalisé à partir de la métrique suivante :

$$CI_\ell(i) = (k_i - 1) \sum_{j \in Ball(i,\ell)} (k_j - 1) \tag{11}$$

   où k_i correspond au degré d'un sommet i et $Ball(i, \ell)$ correspond à l'ensemble des sommets dont le chemin le plus court à partir de i est de longueur ℓ. Nous fixerons $\ell = 1$.

2. **Highest Degree (HD) [15] :** On immunise itérativement les sommets de plus haut degré (nombre de voisins), jusqu'à ce que toutes les contraintes du problème soient respectées.

3. **PageRank (PR) [16] :** Proposé par Sergei Brin et Larry Page, cet algorithme est célèbre pour être à l'origine du moteur de recherche Google. PR assume que les sommets du graphe en entrée sont des sites internet et que les arêtes correspondent aux hyperliens entre eux. Pour chaque sommet, il calcule la probabilité que, si un individu naviguait aléatoirement d'hyperliens en hyperliens, il visiterait éventuellement ce site internet. Le sommet associé à la probabilité la plus élevée correspondrait donc à un site internet plus consulté que les autres. Bien que nos graphes soient des réseaux sociaux, on peut tout de même utiliser cette stratégie pour évalue l'importance de chaque sommet dans le cadre du problème de l'immunisation optimale.

4. Protocole d'expérimentation

4.1 Protocole d'entraînement des algorithmes d'apprentissage

Pour chacun de nos deux modèles, nos algorithmes d'apprentissage seront entraînés sur des graphes de différentes tailles se situant dans les intervalles $\{15-20, 40-50, 50-100, 100-200\}$. Pour chacun de ces cinq intervalles, nous entraînerons des réseaux de neurones sur le modèle de graphe aléatoire Barabási–Albert [22], de paramètre $m = 4$. Une brève de ce modèle est disponible à l'annexe A. Le fait d'entraîner nos agents sur des graphes aléatoires permettra de valider s'ils sont en mesure d'apprendre une heuristique généralisant à une seule et même distribution.

Puisque nos algorithmes d'apprentissage nécessitent d'importantes ressources computationnelles, les algorithmes entraînés sur les jeux de données $\{40-50, 50-100, 100-200\}$ seront initialisés à partir des algorithmes entraînés sur le jeu de données $\{15-20\}$. Nous imposerons également une limite de cinq jours d'entraînement, jusqu'à conccurence d'un million de graphes. L'entraînement des algorithmes qui auront résolu un million de fois le problème en moins de cinq jours sera tout simplement interrompu.

4.2 Cardinalités sélectionnées pour le modèle 2

Nous réaliserons des expérimentations en immunisant 15, 25 et 35 pourcents des sommets, en arrondissant ce ratio à l'entier le plus élevé.

4.3 Comparaison de la performance des heuristiques

Soit OPT l'heuristique étant parvenue, parmi celles présentées à la section 3.3.2, à obtenir les meilleurs résultats pour une instance donnée G. Soit S la solution d'un algorithme quelconque. OPT sera déterminée à partir d'une moyenne sur un ensemble 100 graphes aléatoires pour chaque ensemble de tailles. Soit π l'heuristique basée sur l'apprentissage par renforcement profond. Nous comparerons les performances des heuristiques à partir de la métrique suivante :

$$\frac{f(\pi)}{f(OPT)} \tag{12}$$

De plus, pour le modèle 1, nous réaliserons également des simulations du processus SIR tel que décrit à la section 2.3 afin de comparer des solutions valides de même cardinalité. Nous calculerons $\sigma(S)$ à partir d'une moyenne calculée à partir de simulations sur les 16 combinaisons possibles de probabilités d'infections $T_{ij} = \tau \in \{0.3, 0.4, 0.5, 0.6\}$ et de guérison $r_i = \rho \in \{0.3, 0.4, 0.5, 1.0\}$. Nous réaliserons 1000 simulations pour chacune de ces combinaisons et la moyenne correspondra à la valeur de $\sigma(S)$. Puisque dans la réalité, l'éclosion d'une épidémie provient généralement d'une seule source (le « patient zéro »), nous réaliserons de multiples simulations en considérant, à chaque fois, une source sélectionné aléatoirement parmi les sommets qui ne sont pas immunisés. Ce choix nous permettra de déterminer quelle heuristique est meilleure pour identifier les individus à protéger, peu importe l'origine de la contagion. Évidemment, pour notre seconde métrique, nous comparerons des solutions de même cardinalité. Cela permettra d'identifier une meilleure heuristique pour notre premier modèle dans l'éventualité où plusieurs heuristiques produiraient une solution $S \subseteq V$ de cardinalité minimale, mais ne contenant pas exactement les mêmes sommets.

5. Résultats et discussion

5.1 Comparaison des approches proposées pour le modèle 1

Le tableau 2 présente la performance des algorithmes d'apprentissage pour le modèle 1, évaluée à partir de la métrique présentée à l'équation 12. Les données à partir desquelles ces ratios furent calculés sont disponibles aux tableaux D1 et D2 de l'annexe D. La meilleure heuristique disponible dans la littérature pour le premier modèle étant *Collective Influence*, le tableau 2 compare la performance de notre approche face à cet algorithme.

Tailles des données d'entraînement	**Taille des données de test**				
	15-20	40-50	50-100	100-200	400-500
15-20	0.996	1.000	0.998	1.017	XX
40-50		0.992	0.968	0.969	XX
50-100			0.974	0.963	XX
100-200				1.455	XX

Table 2 – Performance des algorithmes d'apprentissage machine selon la taille des graphes avec lesquels ils ont été entraînés pour la résolution du modèle 1

Dans presque tous les cas, la performance de notre approche est supérieure à celle de CI. Les agents entraînés sur de petits graphes parviennent également à bien généraliser sur des instances de plus grande taille. Les heuristiques apprises sur des graphes de taille 100-200 ne performent toutefois pas très bien. Cela peut s'expliquer par la limite de temps d'entraînement imposée (cinq jours). En effet, puisque des instances plus grandes nécessitent des ressources computationnelles plus importantes, les agents n'ont donc pas eu l'occasion de résoudre le problème suffisamment de fois pour parvenir à identifier une heuristique robuste. Puisque les résultats obtenus sur de petits

graphes sont déjà très bons, il vaut mieux alors réduire la taille des données d'entraînement. Pour améliorer la performance des heuristiques apprises sur les graphes de taille 100-200, il faudrait grandement augmenter le temps d'entraînement.

Également, pour chaque jeu de données de test, nous avons sélectionné le meilleur agent et nous avons comparé sa performance à celles des différentes heuristiques lors de simulations du processus SIR à l'aide du calcul de $\sigma(S)$. Nous avons comparé des solutions de même cardinalité. Le tableau 3 présente les résultats de cette étude.

Heuristique	**Taille des graphes**			
	15-20	40-50	50-100	100-200
Apprentissage	1.541	1.571	1.631	1.667
CI	1.645	1.623	1.677	1.685
HD	1.736	1.758	1.766	1.771
PR	1.745	1.755	1.776	1.793

Table 3 – Espérance du nombre de sommets infectés durant un processus SIR en fonction de la taille des graphes

Notre approche performe mieux que les autres en simulations pour des solutions valides du modèle 1 de même cardinalité. Cela suggère donc que l'heuristique apprise généralise mieux à différentes instances d'une même distribution et qu'elle parvient effectivement à surpasser les approches présentes dans la littérature.

5.2 Comparaison des approches proposées pour le modèle 2

Le tableau 4 présente la performance des algorithmes d'apprentissage pour le modèle 2, évaluée à partir de la métrique présentée à l'équation 2. Les données à partir desquelles ces ratios furent calculés sont disponibles aux tableaux D.3 et D.4 de l'annexe D. La meilleure heuristique disponible dans la littérature pour le premier modèle étant *Collective Influence*, le tableau 4 compare la performance de notre approche face à cet algorithme.

Taille des données d'entraînement	**Budget (%)**	**Taille des données de test**				
		15-20	40-50	50-100	100-200	400-500
15-20	15	1.000	1.713	2.274	3.22	
	25	0.983	1.166	1.583	4.973	
	35	0.791	1.918	1.877	10.272	
40-50	15		0.998	0.995	2.366	
	25		0.980	1.247	3.143	
	35		0.706	0.407	6.617	
50-100	15			0.988	1.368	
	25			1.009	2.467	
	35			0.220	0.125	
100-200	15				1.060	
	25				1.301	
	35				0.412	

Table 4 – Ratio d'approximation des algorithmes d'apprentissage machine selon la taille des graphes avec lesquels ils ont été entraînés pour la résolution du modèle 2

Ce résultat est plutôt surprenant, car l'espace des solutions pour le deuxième modèle est largement plus petit que celui du premier. Il est possible que le processus d'apprentissage ne soit pas optimal et que des modifications à l'architecture du réseau de neurones et/ou à la fonction de récompense

suffisent pour améliorer la performance de notre approche. Il se peut aussi que le problème de l'immunisation optimale avec une contrainte de budget se généralise moins bien que la variante sans budget. Cela impliquerait alors qu'une stratégie vorace permettant d'identifier les k meilleurs sommets pour une distribution donnée serait beaucoup plus difficile à identifier, car ces k meilleurs sommets ne partageraient pas forcément les mêmes propriétés topologiques d'un graphe à un autre.

6. Conclusion

Dans ce rapport, de nouvelles approches basées sur l'apprentissage par renforcement profond pour la résolution du problème de l'immunisation optimale furent proposées. Nous avons considéré les deux variantes du problème, soit celle où on cherche à identifier l'ensemble minimal d'individus à immuniser et celle où on dispose d'un budget à respecter. Les fondements théoriques de l'état de l'art actuel s'étant révélés, dans de récents travaux, à être la meilleure façon de les modéliser, nous nous sommes donc basés sur cette théorie pour construire nos modèles. Nous avons comparé les heuristiques apprises par nos agents à trois algorithmes présents dans la littérature, soit Collective Influence (état de l'art), Highest Degree et PageRank. La performance des différentes heuristiques fut mesurée à partir de la fonction objectif de nos modèles et des résultats de simulations du processus épidémique *Susceptible-Infected-Recovered*. Nos expérimentations ont considéré des graphes Barabási–Albert dont la taille variait entre 15 et 200 sommets.

Pour la première variante du problème de l'immunisation optimale, notre approche surpasse l'état de l'art. Ce ne fut pas le cas toutefois pour la seconde, soit celle où on considère un budget à respecter. Ce résultat est surpenant, car l'espace des solutions pour notre premier modèle est beaucoup plus grand que celui du deuxième. Il serait pertinent de tenter de modifier l'architecture du réseau de neurones et de considérer différentes fonctions de récompense afin de tenter d'améliorer ce résultat. Des architectures plus récentes pourraient également permettre d'améliorer davantage les résultats obtenus pour le premier modèle. De plus, il serait intéressant de considérer des méthodes d'apprentissage nécessitant moins de ressources computationnelles lors de la période d'entraînement, afin de tenter d'apprendre des heuristiques généralisant à des graphes contenant plusieurs milliers de sommets. Enfin, des expérimentations sur d'autres modèles de graphes aléatoires permettraient de valider davantage la robustesse de notre approche. Dans l'optique où les résultats seraient concluants pour tous les modèles, des analyses sur de petits graphes pourraient être réalisées afin de déterminer si les sommets identifiés par nos agents partagent des propriétés topologiques qui pourraient permettre de formuler une nouvelle heuristique sans avoir à passer par l'étape d'apprentissage.

Références

[1] Holme, P. *Three faces of node importance in network epidemiology: Exact results for small graphs.* Physical Review E 96, 062305, 2017.

[2] Radicchi, F. and Castellano, C. *Fundamental difference between superblockers and superspreaders in networks.* Physical Review E 95, 012318, 2017.

[3] Kermack, W.O. and McKendrick, A.G. *A contribution to the mathematical theory of epidemics.* Proceedings of the Royal Society A 115, 1927.

[4] Morone, F. and Makse, H.A. *Influence maximization in complex networks through optimal percolation.* Nature, 2015.

[5] Hashimoto, K. *Zeta functions of finite graphs and representations of p-adic groups.* Advanced Studies in Pure Mathematics 15, 1989.

[6] Angel, O., Friedman, J. and Hoory, S. *The non-backtracking spectrum of the universal cover of a graph.* Transactions of the American Mathematical Society 367, 2015.

[7] Durfee, C. and Martin, K. *Distinguishing graphs with zeta functions and generalized spectra.* Linear Algebra and its Applications, 2015.

[8] Krzakala, F., Moore, C., Mossel, E., Neeman, J., Sly, A. Zdeborová, L. and Zhang, P. *Spectral redemption in clustering sparse networks.* PNAS, 2013.

[9] Newman, M. E. J. *Spectral methods for community detection and graph partitioning.* Physical Review E 88, 042822, 2013.

[10] Kempe, D., Kleinberg, J. and Tardos E. *Maximizing the spread of influence through a social network.* Proceeding of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2003.

[11] Pei, S., Wang, J., Morone, F. and Makse, H.A. *Influencer identification in dynamical complex systems.* Journal of Complex Networks, 2019.

[12] Wang, C., Chen, W., and Wang, Y. *Scalable influence maximization for independent cascade model in large-scale social networks.* Proceeding of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2010.

[13] Altarelli, F., Braunstein, A., Dall'Asta, L., Wakeling, J. R., and Zecchina, R. *Containing epidemic outbreaks by message-passing techniques.* Physical Review X 4, 021024, 2014.

[14] Shapiro, M. and Delgado-Eckert, E. *Finding the probability of infection in an SIR network is NP-Hard.* Mathematical Biosciences 240, 2012.

[15] Albert, R., Jeong, H. and Barabási, A.-L. *Error and attack tolerance of complex networks.* Nature, 2000.

[16] Brin, S. and Page, L. *The anatomy of a large-scale hypertextual web search engine.* Computer Networks and ISDN Systems 30, 1998.

[17] H. Dai, E. B. Khalil, Y. Zhang, B. Dilkina, and L. Song. *Learning combinatorial optimization algorithms over graphs.* 31st Conference on Neural Information Processing Systems (NIPS), 2017.

[18] Dai, Hanjun, Dai, Bo, and Song, Le. *Discriminative embeddings of latent variable models for structured data.* ICML, 2016.

[19] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, J. Pineau, et al.. *An introduction to deep reinforcement learning.* Foundations and Trends in Machine Learning, 2018.

[20] Szepesvári, C. *Algorithms for reinforcement learning.* Morgan and Claypool, 2009.

[21] Sutton, R.S. and Barto, A.G. *Reinforcement Learning: An Introduction.* MIT Press, 1998.

[22] Barabasi, A.L, and Albert, R. *Emergence of scaling in random networks.* Science, 1999.

Annexe A

Le modèle de Barabási-Albert

Le modèle de Barabási-Albert [22] est un algorithme qui permet de générer aléatoirement des réseaux dits « sans échelle » (*scale free network*). Il s'agit d'un modèle de graphes aléatoires très connu, censé encapsuler les mécanismes de croissance et d'attachement préférentiel (plus le degré d'un noeud est grand, plus il a de chances d'être connecté à de nouveaux individus) qu'on remarque dans les réseaux sociaux. L'algorithme générant un graphe Barabási–Albert de n sommets nécessite de fixer un paramètre m qui correspond, pour chaque sommet, au nombre d'arcs qui le connecteront préférentiellement aux sommets de plus haut degré.

Annexe B

Hyperparamètres des algorithmes d'apprentissage

Les algorithmes d'apprentissage nécessitent de sélectionner de nombreux hyperparamètres. Les valeurs choisies pour ceux directement mentionnés à la section 3.3.1 sont présentées dans le tableau B.1.

Nom	Valeur
p	64
T	5
n	2 pour les agents entraînés sur des graphes de taille 15-20, 40-50, 4 pour 50-100 et 5 pour 100-200
ϵ	La valeur d'ϵ n'est pas constante au fil de l'entraînement, de manière à laisser l'algorithme explorer davantage l'espace des solutions au départ. En effet, la valeur d'epsilon est d'abord initialisée à 1.0, puis est diminuée à chaque itération pendant 10 000 épisodes jusqu'à atteindre 0.05. La valeur d'epsilon est alors maintenue à 0.05 pour la suite de l'entraînement

Tab. B.1 – Valeurs choisies des hyperparamètres

L'architecture que nous avons utilisé pour le réseau de neurones paramétrant $Q(s,a,\Theta)$ est un perceptron multicouche, soit l'architecture la plus simple. Elle est organisée en trois parties, soit la couche d'entrée, les couches cachées et la couche de sortie. Nous avons fixé le nombre de couches cachées à 64. De plus, nous initialisons au début de l'entrâinement les poids Θ avec une valeur réelle déterminée aléatoirement à partir d'une distribution normale $\mathcal{N}(0,0.01)$. Les valeurs des autres hyperparamètres qu'il fallait fixer dans la librairie que nous avons utilisé sont présentées au tableau B.2.

Hyperparamètre	Valeur
learning rate	0.0001
momentum	0.9
batch size	64

Tab. B.2 – Hyperparamètres supplémentaires

Annexe C

Actualisation des poids du réseau de neurones

L'algorithme 1 présente la manière dont les poids Θ sont mis à jour, tel que décrit à la fin de la section 3.3.1.

Algorithme 1 : Actualisation des poids pour $Q(s, a, \Theta)$

```
Initialiser une mémoire M de capacité E
pour chaque graphe G du jeu de données d'entraînement faire
    s_1 → {}
    tant que les contraintes du modèle ne sont pas respectées faire
        Obtenir une représentation du graphe obtenu en retirant tous les sommets
         contenus dans s_t à partir de l'équation 8
        Sélectionner un sommet a_t ∈ V \ s_t à l'aide de l'équation 7
        Observer la récompense r(s_t, a_t)
        s_{t+1} → s_t ∪ {a_t}
        si t ≥ n alors
            Ajouter le tuple (s_{t−n}, a_{t−n}, R_{t−n,t}, s_t) à M
            Obtenir un échantillon B ∼ M
            Actualiser Θ par descente de gradient stochastique à partir de l'équation
             10 et de l'échantillon B
        fin
    fin
fin
return Θ
```

Annexe D

Résultats supplémentaires

Cette annexe contient des tableaux dont les données furent utilisées pour calculer les résultats présentés à la section 5.

Données d'entraînement	Données de test			
	15-20	40-50	50-100	100-200
15-20	7.95	20.17	31.63	62.61
40-50		20.02	30.7	59.68
50-100			30.88	59.31
100-200				89.57
400-500				

Tab. D.1 – Résultats obtnenus sur les données de tests avec les algorithmes d'apprentissage (méthode 1)

Heuristique	Taille des données			
	15-20	40-50	50-100	100-200
CI	8.03	20.17	31.7	61.56
HD	8.30	21.18	33.86	66.55
PR	8.59	22.31	35.43	69.25

Tab. D.2 – Résultats obtnenus sur les données de tests avec les heuristiques choisies (méthode 1)

Heuristique	**Budget (%)**	**Taille des données de test**			
		15-20	40-50	50-100	100-200
CI	15	3.470	3.380	3.420	3.140
	25	2.170	1.878	1.997	1.681
	35	0.814	0.660	0.676	0.136
HD	15	3.527	3.467	3.466	3.217
	25	2.279	1.918	2.062	1.796
	35	0.839	0.845	0.957	0.745
PR	15	3.532	3.448	3.556	3.255
	25	2.307	1.948	2.117	1.906
	35	1.227	0.996	1.338	1.068

Tab. D.3 – Résultats obtnenus sur les données de tests avec les heuristiques choisies (méthode 2)

Taille ds données d'entraînement	**Budget (%)**	**Taille des données de test**				
		15-20	40-50	50-100	100-200	400-500
15-20	15	3.472	5.792	7.779	10.100	
	25	2.133	2.191	3.163	8.359	
	35	0.644	1.266	1.269	1.397	
40-50	15		3.376	3.404	7.430	
	25		1.841	2.492	5.283	
	35		0.466	0.275	0.900	
50-100	15			3.381	4.296	
	25			2.016	4.147	
	35			0.149	0.017	
100-200	15				3.332	
	25				2.187	
	35				0.056	

Tab. D.4 – Résultats obtnenus sur les données de tests avec les algorithmes d'apprentissage (méthode 2)