Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen
Faculté des Sciences
Départementd'Informatique

Mémoire de fin d'études

pour l'obtention du diplôme de Master en Informatique

Option: *Système d'Information et de Connaissances (S.I.C)*

Thème

Algorithme de Djikstra: réalisation et étude de complexité

Réalisé par :

- **FETTOUHI Mohamed Amine**

Présenté le 03 Juillet 2022 devant le jury composé de :

- Mme Iles Nawel (Présidente)
- Mme Chaouche Ramdane Lamia (Examinatrice)
- Mme Gaouar Lamia (Encadreur)

Année universitaire : 2021 / 2022

Remerciements

Je remercie tout d'abord ALLAH tout puissant qui par sa grâce nous a permis d'arriver au bout de nos efforts en nous donnant la santé, la force, le courage , sans lui rien de tout cela n'aurait pu être .

Je présente également mes remerciements aux membres de jury pour avoir accepté d'évaluer ce travail.

Je remercie Dr. GAOUAR Lamia mon encadreur d'avoir bien dirigé ce travail, avec ses judicieux conseils dont elle a fait preuve durant l'élaboration de notre étude.

Je réserve les derniers remerciements à toute ma famille et mes amis

Dédicaces

Ce travail est dédie

à l'âme de mon père, que Dieu ait pitié de lui

à ma tendre mère qui a toujours été une source

inépuisable d'amour et d'encouragements.

à mes soeurs et à mes fréres.

à mes ami(e)s intimes.

à tous mes professeurs.

à tous ceux que j'aime.

Mohamed Amine FETTOUHI

Table des matières

Liste des figures

Liste des tableaux

Glossaire

Terme	Définition	page
Cytoplasme	Contenu d'une cellule vivante compris entre la membrane et le Noyau	17/21
Eucaryote	Organisme vivant caractérisé par le fait que la majorité du matériel génétique cellulaire est contenu dans un noyau constitué par une enveloppe nucléaire Les Animaux, les Champignons et les Plantes sont des eucaryotes	20
Mycetozoaire	Champignon qui à une certaine époque de son développement, est susceptible de mouvements et a été comparé à un animal	17
Pseudopode	Prolongement cytoplasmique d'une cellule, d'un organisme unicellulaire, servant à la locomotion, à la phagocytose	18/31

Introduction générale

Les problèmes d'optimisation occupent actuellement une place importante dans la communauté des ingénieurs et des scientifiques. En effet, ce type de problèmes intervient dans plusieurs domaines, tels que la conception des systèmes mécaniques, traitement d'images et l'électronique. Le problème du plus court chemin est l'un des problèmes algorithmiques les plus élémentaires, importants et bien étudiés. Il apparaît dans d'innombrables applications pratiques (réseaux routiers, connexions de circuits, lastructure d'ADN d'un organisme…). La stratégie de base pour le résoudre dans les graphes G = (V (G), E(G)) avec longueurs de bord non négatif est l'algorithme de Dijkstra. Ce travail évolue autour de la problématique du plus court chemin avec pour référence, une base fondamentale, l'algorithme de Djikstra. Il vise également une vision nouvelle de cette problématique influencée par ce nouveau monde post-covid dans lequel nous évoluons aujourd'hui. L'intérêt de ce travail est l'utilisation de nouvelles techniques inspirées par la nature, comme les capteurs biologiques fournis par le mystérieux organisme : le Physarum polycephalum

L'objectif de ce mémoire est d'étudier le problème plus court chemin sur la base de l'algorithme de Djikstra, tout en proposant une nouvelle vision par le biais du Physarum polycephalum. Nous proposons également la complexité temporelle comme métrique d'évaluation de l'efficacité. Notre mémoire est organisé comme suit :

Dans le premier chapitre, nous présenterons les aspects fondamentaux de l'algorithme de Dijkstra pertinents pour notre étude. Dans Le deuxième chapitre, nous présenterons une description complète du Physarum Polycephalum (blob) et démontrerons sa place dans le domaine du problème de plus court chemin. Dans le dernier chapitre, nous calculons la complexité temporelle de l'algorithme de Djikstra, comme métrique de l'efficacité de l'algorithme.

CHAPITRE I

Algorithme de Djikstra

Chapitre I : Algorithme de Djikstra

1-Introduction :

Le problème de plus court chemin est un problème d'optimisation bien connu qui fait l'objet d'études approfondies depuis des décennies. Parmi les méthodes sophistiquées utilisées pour choisir les chemins : l'algorithme de Djikstra. Djikstra est une méthode simple mais puissante. Dans ce chapitre nous présentons l'algorithme de Djikstra , et leur principes, avec une présentation générale du problème classique du plus court chemin.

2-La notion de la théorie des graphes :

2-1-Application de la théorie des graphes :

La théorie des graphes a ses applications dans divers domaines de l'ingénierie :

- Génie électrique : Les concepts de la théorie des graphes soit largement utilisés dans la conception des connexions de circuits. Les types ou organisations de connexions sont nommées topologies, tels que : étoile, pont, série et topologies parallèles.

- Informatique : Théorie des graphes est utilisée pour l'étude des algorithmes. Par exemple :

- Algorithme de **Kruskal**.
- Algorithme de **Prim**.
- Algorithme de **Djikstra** .

- Réseaux informatique : Les relations entre les ordinateurs interconnectés dans le réseau suivent les principes de la théorie des graphes.

- Sciences : La structure moléculaire et de la structure chimique d'une substance, la structure d'ADN d'un organisme, sont représentés par les graphiques.

- Linguistique : L'arbre d'analyse syntaxique d'une langue et la grammaire d'une langue utilise des graphiques.

- Généralités : Routes entre les villes peuvent être représentées à l'aide de graphiques. Décrivant l'information hiérarchique ordonnée, comme arbre peut être utilisé comme un

type spécial de graphique appelé arbre . [16]

2-2-Les graphes :

Les graphes sont des concepts mathématiques utilisés pour modéliser des relations binaires entre des objets d'un même ensemble. Ils sont fréquemment utilisés pour modéliser des systèmes qui se présentent sous la forme d'un réseau.
Il existe deux types de graphes : les graphes orientés et les graphes non orientés.

2-2-1-Graphe orienté :

Un graphe orienté G est un couple (N;A) avec N un ensemble dont les éléments sont appelés nœuds et A un ensemble dont les éléments sontdes couples ordonnés de nœuds appelés arcs.

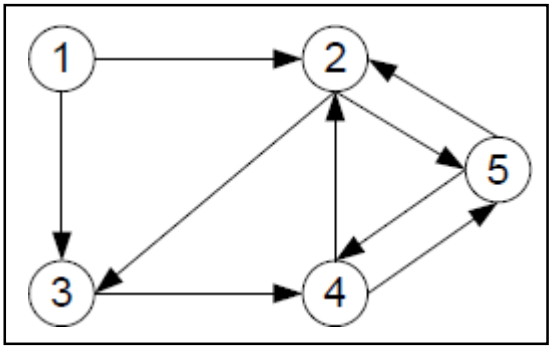

FIGURE 1. 1 - Exemple d'un graphe orienté

2-2-2-Graphe non orienté :

Un graphe non orienté G est un couple (N;A) avec N un ensemble dont les éléments sont appelés nœuds et A un ensemble dont les éléments sont des paires de nœuds appelées arêtes. [5]

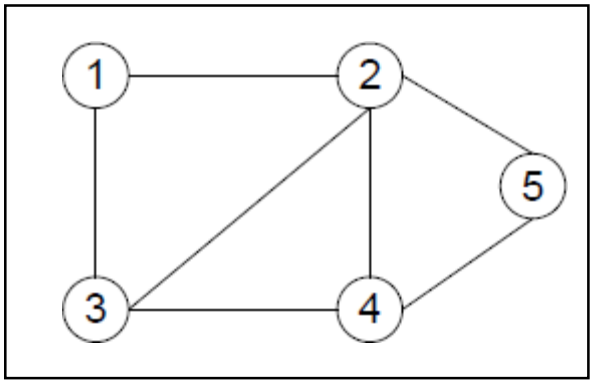

FIGURE 1. 2 - Exemple d'un graphe non orienté

3-Problème classique du plus court chemin :

Le problème du plus court chemin pour les graphes statiques déterministes a fait l'objet de nombreuses études. Il est devenu un problème classique de la théorie des graphes . L'objectif est de calculer le plus court chemin entre deux nœuds d'un graphe statique déterministe. Dans cette optique, les notions de fonction de coût d'un chemin et du plus court chemin sont définies. [10]

4-Algorithme de Djikstra :

Edsger Wybe Djikstra 1930-2002**,** informaticien néerlandais réputé pour avoir écrit plus de 1300 manuscrits au stylo-plum a proposé en 1959 un algorithme (nommé algorithme de Djikstra) qui permet de déterminer le plus court chemin entre deux sommets d'un graphe connexe pondéré. L'algorithme de Djikstra est basé sur l'observation suivante : une fois que nous déterminons le chemin le plus court vers un sommet v, alors les chemins qui vont de v à chacun de ses sommets adjacents pourraient être le plus court chemin vers chacun de ces sommets voisins. L'algorithme de Djikstra est un algorithme de programmation dynamique glouton, il visite toutes les solutions possibles.

4-1-Idée de Djikstra :

Djikstra est un algorithme de recherche de chemin basé sur la stratégie itérative gourmande. Chaque fois qu'il choisit le point le plus proche et marque le point, puis trouve le point le plus proche à partir des points de gauche jusqu'à ce que le chemin le plus court de tous les points ait été trouvé. En facteur, l'algorithme de Djikstra est de former un arbre de chemin en fonction de la longueur du chemin. De cette façon, le chemin le plus court sera obtenu. [9]

4-2-Conditions pour appliquer l'algorithme de Djikstra :

- Pas de longueur négative
- Arc ou arête
- Nombre de sommets fini
- Une source (et accessoirement une cible) définie.

4-3-Processus de l'algorithme de Djikstra :

L'algorithme de Djikstra prend en entrée un graphe orienté pondéré par des réels positifs et un sommet source. Il s'agit de construire progressivement un sous-graphe dans lequel

sont classés les différents sommets par ordre croissant de leur distance minimale au sommet de départ. La distance correspond à la somme des poids des arêtes empruntées.

Sommet visité : Un sommet pour lequel nous avons déterminé le chemin le plus court. Une fois que nous avons défini un sommet comme VISITE, cela est définitif, et nous ne reviendrons plus sur ce sommet.

Sommet marqué : Un sommet pour lequel un chemin a été trouvé. Nous marquons ce sommet comme CANDIDAT pour le chemin le plus court.

Au départ, on considère que les distances de chaque sommet au sommet de départ sont infinies sauf pour le sommet de départ pour lequel la distance est de 0. Le sous-graphe de départ est l'ensemble vide.

Au cours de chaque itération, on choisit en dehors du sous-graphe un sommet de distance minimale et on l'ajoute au sous-graphe (il devient un sommet visité). Ensuite, on met à jour les distances des sommets voisins de celui ajouté (les sommets sont marqués). La mise à jour s'opère comme suit : la nouvelle distance du sommet voisin est le minimum entre la distance existante et celle obtenue en ajoutant le poids de l'arc entre sommet voisin et sommet ajouté à la distance du sommet ajouté

On continue ainsi jusqu'à compléter entièrement le sous-graphe (ou jusqu'à sélection du sommet d'arrivée). [15]

4-4-Pseudo code algorithme de Djikstra :

```
Données : Un graphe orienté pondéré G = (X, A, W) et un sommet s ∈ X
Résultat : Le plus court chemin de s vers tous les autres sommets de G
// V : Tableau stockant les étiquettes des sommets de G
Initialiser V à +∞
V[s] = 0
// P : Tableau permettant de retrouver la composition des chemins
Initialiser P à 0
P[s] = s
répéter
    // Recherche du sommet x non fixé de plus petite étiquette
    V_min = +∞
    pour y allant de 1 à N faire
        si y non marqué et V[y] < V_min alors
            x ← y
            V_min ← V[y]
    // Mise à jour des successeurs non fixés de x
    si V_min < +∞ alors
        Marquer x
        pour tout successeur y de x faire
            si y non marqué et V[x] + W[x, y] < V[y] alors
                V[y] = V[x] + W[x, y]
                P[y] = x
jusqu'à V_min = +∞
```

L'algorithme présenté ci-dessus permet aussi, avec le tableau , souvent appelé tableaudes prédécesseurs, de retrouver la composition du plus court chemin, et les sommets intermédiaires du chemin.

Exemple :

Le graphe ci-dessous représente le réseau routier d'une région qui prend en compte le sens de la circulation, chaque arc représente une route à sens unique dont le poids est la distance en kilomètre entre deux sommets. Quel est l'itinéraire le plus court qui relie A à H ?

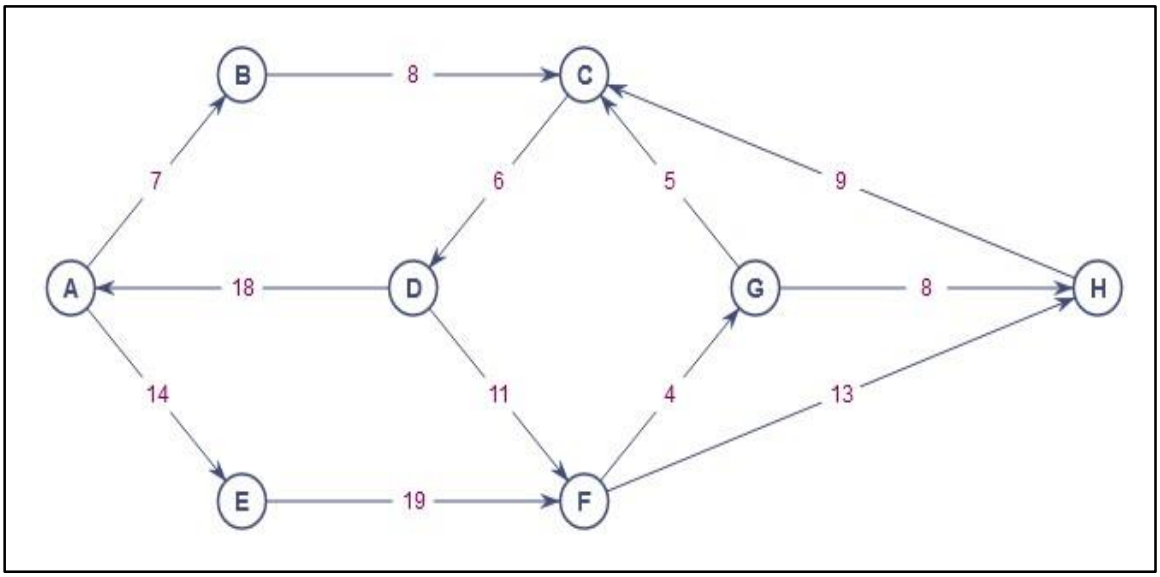

FIGURE 1. 3 – Exemple d'un graphe orienté pondéré

Pour faciliter la recherche du plus court chemin il est commode de présenter les résultats dans un tableau :

A	B	C	D	E	F	G	H	Sommets sélectionnés	Commentaires
0	∞	∞	∞	∞	∞	∞	∞	A(0)	On affecte le poids 0 au sommet origine (A) et on attribue provisoirement un poids ∞ aux autres sommets. Le sommet A de poids 0 est selectionné.
	7	∞	∞	14(A)	∞	∞	∞	B(7)	Le sommet A de poids 0 ayant été selectionné, on marque provisoirement les sommets B (0+7<∞) et E (0+14<∞) adjacents à A. On selectionne le sommet B de poids minimal. Le prédécesseur de B est A.
		15(B)	∞	14(A)	∞	∞	∞	E(14)	B de poids 7 ayant été selectionné, on marque provisoirement le sommet C(7+8<∞) adjacent à B. On selectionne le sommet E de poids minimal. Le prédécesseur de E est A.
		15(B)	∞		33(E)	∞	∞	C(15)	E de poids 14 ayant été selectionné, on marque provisoirement le sommet F (14+19<∞) adjacent à E. On selectionne le sommet C de poids minimal. Le prédécesseur de C est B.
			21(C		33(E)	∞	∞	D(21)	C de poids 15 ayant été selectionné, on marque provisoirement le sommet D (15+6<∞) adjacent à C. On selectionne le sommet D de poids minimal. Le prédécesseur de D est C.
					~~33(E)~~ 33(D)	∞	∞	F(32)	D de poids 21 ayant été selectionné, on marque provisoirement le sommet F (21+11<33) adjacent à D. On selectionne le sommet F de poids minimal. Le prédécesseur de F est D.
						36(F)	45(F)	G(36)	F de poids 32 ayant été selectionné, on marque provisoirement les sommets G (32+4<∞) et E (32+13<∞) adjacent à F. On selectionne le sommet G de poids minimal. Le prédécesseur de G est F.
							~~45(F)~~ 44(G)	H(44)	G de poids 36 ayant été selectionné, on marque provisoirement le sommet H (36+8<45). On selectionne le sommet H de poids minimal. Le prédécesseur de H est G.

TABLE 1.1 – Processus de l'algorithme de Djikstra avec explication détaillée .

L'algorithme de Djikstra fournit les longueurs des plus courts chemins du sommet origine aux différents sommets.

- Pour déterminer le plus court chemin du sommet origine à un sommet x, il suffit de remonter la liste des prédécesseurs en partant du sommet x.
- Le sommet final H étant marqué, pour lire la chaîne de poids minimal, on part de H eton remonte la chaîne en suivant les prédécesseurs. H←G←F←D←C←B←A. [24]

Le trajet le plus court pour aller de **A** à **H** est **A - B - C - D - F - G - H**, la distance parcourue est **44 km**.

5-Avantages et inconvénients de l'algorithme de Djikstra :

L'algorithme de Djikstra est fondé sur un parcours en largeur. L'algorithme ne s'applique pas aux graphes avec des poids négatifs. Étant donné que Djikstra suit une approche gourmande, une fois qu'un nœud est marqué comme visité, il ne peut pas être reconsidéré même s'il existe un autre chemin avec moins de coût ou de distance. Ce problème ne se pose que s'il existe un poids ou une arête négative dans le graphique. Mais l'algorithme de Bellman-Ford permet de résoudre le problème. L'algorithme de Floyd-Warshall calcule des plus courts chemins entre tous les sommets dans un graphe où les poids peuvent être négatifs.

L'algorithme de Djikstra a une complexité temporelle linéaire, il peut donc être facilement utilisé pour de gros problèmes. Il est utile pour trouver la distance la plus courte, il est donc également utilisé dans Google Maps et le calcul du trafic.

En contrepartie, il est incapable de gérer les poids négatifs. Il suit une sorte d'approche aveugle donc il y a une perte de temps.

6-Applications de l'algorithme de Djikstra :

L'algorithme de Djikstra a de nombreuses applications:
Services de cartographie numérique comme Google Maps : Supposons que vous voulez voyager d'une ville à une autre ville. Vous utilisez Google Maps pour trouver l'itinéraire le plus court. Comment cela fonctionnera-t-il? Supposons que la ville dans laquelle vous vous trouvez soit le sommet source et que votre destination soit un autre sommet.

Il y aura encore beaucoup de villes entre votre destination et votre point actuel. Supposons que ces villes soient des sommets intermédiaires. La distance et le trafic entre deux villes peut être supposé être le poids entre chaque paire de vérités. Google maps va appliquer l'algorithme Djikstra sur le graphique de la ville et trouver le chemin le plus court.

Désigner des serveurs : Dans un réseau, l'algorithme de Djikstra peut être utilisé pour trouver le chemin le plus court vers un serveur pour transmettre des fichiers parmi les nœuds sur un réseau.

Routage IP : L'algorithme de Djikstra peut être utilisé par les protocoles de routage d'état de lien pour trouver le meilleur chemin pour acheminer les données entre les routeurs. [18]

7-Extension de l'algorithme de Djikstra :

L'algorithme de Djikstra fournit efficacement un chemin optimal pour passer d'un nœud de départ à un nœud cible dans un graphique pondéré. Mais quand une personne estime un chemin à suivre dans une situation réelle (par exemple lorsque vous conduisez entre deux points différents dans une ville), le temps de conduite le plus courtn'est pas toujours obtenu. Par conséquent, utiliser l'algorithme de Djikstra pour simuler le comportement des pilotes n'est pas réaliste.

Pour réaliser des simulations plus réalistes, il faut utilisé une extension probabiliste de l'algorithme de Djikstra basée sur plusieurs approches différentes:

- Dans la plupart des situations, les conducteurs ne connaissent pas la longueur exacte des voies.
- Ce qui est encore plus convaincant, c'est que même si le conducteur connaîtl'itinéraire le plus court, il peut préférer choisir un itinéraire différent
- La livraison rapide contribue au changement climatique. Il y a toujours un coût environnemental caché qui n'apparait pas à la surface (pour réduire l'impact écologique des milliard de colis livrés chaque année , Amazon expérimente les livraisons de colis par vélos , drones et même à pied qu'on les appelle les walkers).
- Pendant la pandémie de COVID-19, le facteur temps n'est plus important. Les gens ont passé beaucoup de temps chez eux pendant la quarantaine**.** [7]

8-Conclusion :

Ce premier chapitre est consacré à l'introduction des concepts de base de la théorie des graphes et du problème de plus court chemin. Nous nous sommes particulièrement penchés sur une présentation détaillée de l'algorithme de Djikstra dans le but de mettre en lumière la problématique suivante : les notions de fonction de coût d'un chemin et du plus court chemin sur lesquelles se basent initialement l'algorithme de Djikstra en particulier, et la problématique du plus court chemin en générale, sont-elles toujours les mêmes en vue des bouleversements récents que connais le monde ?

En effet, les contraintes causées par la pandémie du COVID, la guerre en Ukraine et ses répercussions sur le prix des carburants ainsi que la problématique de l'acheminement du blé à travers la mer noire ou encore le dérèglement climatique redéfinissent la notion même de plus court chemin. Dans un souci écologique, nous nous tournons vers une nouvelle solution innovante inspirée de la nature pour reconsidérer la problématique du plus court chemin, que l'on présente sous le nom de BLOB.

CHAPITRE II

Physarum Polycephalum (BLOB)

Chapitre II: Physarum Polycephalum (BLOB)

1-Introduction :

La nouvelle méthode d'optimisation des réseaux de générations est un sujet émergent qui peut être appliqué dans de nombreux domaines, y compris le développement de la médecine biologique, la communication sociale et les transports locaux. Ces questions difficiles sont considérées comme très difficiles à résoudre. En fait, les scientifiques ont exploité beaucoup de technologies de bio-intelligence pour résoudre différents types de problèmes d'optimisation des réseaux. . Les études récemment conduites ont constaté que le Physarum Polycephalum montre une excellente intelligence dans la construction du réseau biologique. Ce chapitre donne une présentation générale de cet être mystérieux, nous discutons également sa probable place dans le problème classique de plus court chemin.

2-Le Physarum Polycephalum : un être stupéfiant

Le Physarum Polycephalum est une créature rampante et visqueuse apparut il y a environ 500 millions d'années.
Inclassable dans la catégorisation des espèces vivantes, il ne s'apparente ni à un végétal, ni à un champignon, ni à un animal, ni même à un minéral.

Toutefois, les biologistes lui ont trouvé une place et le considère comme un Mycetozoaire: une catégorie en marge de l'évolution puisqu'il s'agit d'une forme de vie primaire, qui n'est composée que d'une seule cellule et qui est apparue avant même les premiers animaux et végétaux. [20]

3-Premier découvert de Physarum Polycephalum :

Le Physarum Polycephalum est un organisme unicellulaire étonnant capable de retenir la localisation de sources de nourriture et d'établir entre elles un réseau optimal de transport des nutriments. MirnaKramar et Karen Alim, de l'institut Max-Planck à Gottingen, ont étudié le mécanisme qui lui permet de mémoriser ces informations . il se compose d'un réseau de tubes qui transportent le cytoplasme et les nutriments. A proximité d'une source de nourriture, un composé chimique libéré dans le cytoplasme interagit avec la membrane des tubes et en modifie le diamètre: les plus proches de la source en reçoivent davantage et s'élargissent, tandis que les autres rapetissent.

L'importance du flux encode la position d'intérêt. La découverte d'un nouveau réservoir de nourriture conduit à une mise à jour du réseau, qui maintient le souvenir des précédents sites et les connecte de façon optimale. [12]

Il a été découvert pour la première fois en 1973, au Texas. Une habitante, découvre effrayée, une monstrueuse masse gluante dans son jardin. Prise de panique, elle appelle la police ainsi que les pompiers. Ces derniers, pour se débarrasser et détruire cet ORNI (Objet Rampant Non Identifié) usent de plusieurs techniques : feu, eau, allant même jusqu'à tirer dessus.. mais sans succès... Plusieurs heures après, la masse a doublé de volume !

Bien qu'il soit unicellulaire, le Physarum est capable de grossir, et est susceptible de couvrir une surface d'au moins 10 m^2 ! Cette immense cellule est composée de milliers de noyaux qui se développent à l'intérieur d'une même membrane. Lorsqu'il est affamé, il forme des réseaux d'excroissances que l'on appelle des Pseudopodes.

4-Pourquoi il est difficile de classer le Physarum Polycephalum ?

Si les biologistes ont eu des difficultés pour le classer, c'est que le Physarum emprunte à la fois, des caractéristiques des mondes animal et végétal, mais également des champignons :

- il produit des pigments comme une plantes.
- il se déplace et se nourrit comme un animal.
- il se reproduit par spores comme un champignon.

Ces caractéristiques rendent le BLOB attractif et inspirant dans la formulation de nouvelles techniques pour la problématique du plus court chemin.

5-Cycle de vie du BLOB :

Les scientifiques l'on surnommé « BLOB » en référence à un film d'horreur américain de 1958 : « Danger planétaire » en français et dont le titre original est « The BLOB ».Ce film mêle horreur et science-fiction. Réalisé par Irvin S. Yeaworth Jr, Russell S. Doughten Jr. ayant pour acteur principal Steve McQueen.

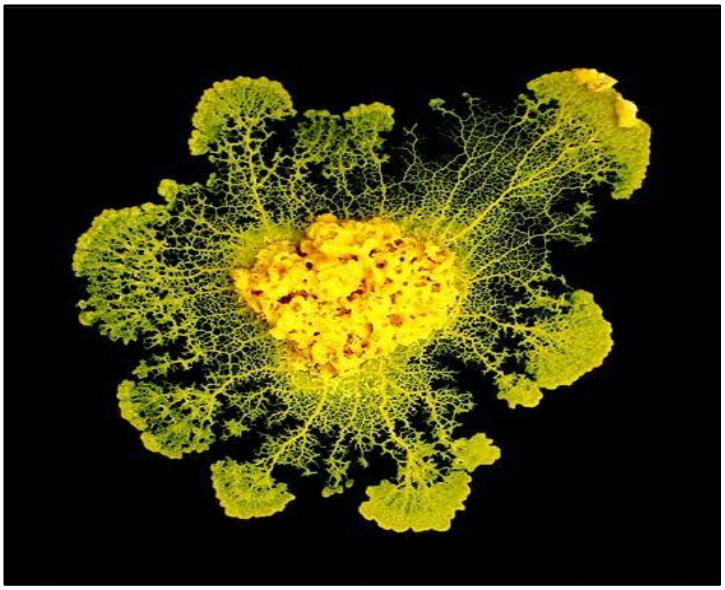

FIGURE 2.1 – Physarum Polycephalum (diamètre : environ 10 centimètres)

La principale phase végétative de P. Polycephalum est le plasmode. Ce plasmode est constitué de réseaux de veines protoplasmiques et de nombreux noyaux. C'est au cours de cette étape que l'organisme cherche de la nourriture. Le plasmodium entoure sa nourriture et sécrète des enzymes pour la digérer.

Si les conditions environnementales entrainent la dessiccation du plasmode lors de l'alimentation ou de migration, il se formera alors une sclérote. Le sclérote est multi-nucléées constitué de tissus très renforcés servant de stade de dormance assurant ainsi la protection de Physarum pendant de longues périodes. Une fois les conditions favorables revenues, le plasmode réapparaît pour poursuivre sa quête de nourriture.

Quand les réserves alimentaires sont épuisées, le plasmode cesse de se nourrir et commence sa phase de reproduction. Des sporanges se forment dans le plasmode, la méiose se produit au sein de ces structures et les spores se forment. Les sporanges se forment habituellement à l'air libre pour que les spores soient transmises par le vent. Les spores peuvent rester quiescentes pendant des années. Toutefois, lorsque les conditions environnementales sont favorables à la croissance, les spores germent et libèrent soit des cellules flagellées soit amiboïdes (stade mobiles); les cellules fusionnent ensuite pour former un nouveau plasmode.

FIGURE 2.2 Physarum Polycephalum dans son habitat naturel.

6-Caractéristiques hors du commun:

Visible à l'œil nu, P. Polycephalum est généralement de couleur jaune, se nourrissant de spores de champignons, de bactéries et autres microbes. *P. Polycephalum* est l'un des microbes eucaryotes**.** Le plus facile à cultiver in vitro (du papier absorbant humide et des flocons d'avoine suffisent), et a été utilisé comme organisme modèle pour de nombreuses études sur les mouvements amiboïdes et la motilité cellulaire. [22]

6-1-Une composition unicellulaire :

Les chercheurs estiment que le nombre total de cellules dans le corps humain est d'environ 30.000 milliards. Ces cellules sont indispensables pour assurer nos fonctions vitales. Et bien le BLOB, avec sa cellule unique, est capable d'assurer toute les fonctions : la vue, l'odorat, l'ouïe, mais aussi les fonctions digestives et respiratoires !

6-2-Une résistance à toute épreuve :

Il ne craint ni le feu, ni l'eau.

Il a la capacité de cicatriser en quelques secondes et peut être découpé en morceaux. Chaque morceau devient alors un clone indépendant.

Quand ils sont compatibles, les BLOBs ont même la capacité de fusionner entre eux. [20]

6-3-Un système de reproduction atypique :

En effet, il existe 720 genres sexuels chez le BLOB, ce qui signifie que contrairement à la majorité des êtres vivants sur terre, le BLOB n'a pas une chance sur deux de trouver un congénère compatible, mais 719 chances sur 720 ! Une particularité qui accroît drastiquement ses capacités de reproduction. En opposition avec cette masse visqueuse et peu ragoutante, les spores du BLOB sont particulièrement magnifiques, à tel point que certaines personnes les collectionnent.

6-4-Déplacement :

Le déplacement du **BLOB** est lié à un courant cytoplasmique appelé « shuttle streaming » en anglais, évoquant le va-et-vient d'une navette (shuttle). Ce shuttle streaming est caractérisé par un changement de direction d'avant en arrière du flux de cytoplasme, avec un intervalle de temps d'environ deux minutes. À l'intérieur des plasmodes, la force motrice est générée par la contraction suivie de la relaxation de couches membraneuses probablement constituées d'actine (de type *filament d'actine* associé à la contraction). La couche de filaments crée un gradient de pression grâce auquel le cytoplasme .s'écoule à l'intérieur du plasmode.

Le plasmode sécrète un mucus qui le protège contre la dessiccation mais a aussi un rôle répulsif qui lui évite d'explorer deux fois la même piste. Cette mémoire spatiale externalisée lui permet de se déplacer à 1 cm/h et peut atteindre les 4 cm/heure s'il est affamé. Il peut même ramper sur l'eau. [4]

6-5-Calcul :

Andrew Adamatzky de l'université de Bristol montre comment il est possible d'orienter ou de cliver un plasmode en utilisant la lumière ou des sources de nourriture. Dans la mesure où des plasmodes réagissent toujours de la même manière aux mêmes stimuli, Adamatzky suggère que Physarum Polycephalum constitue « un modèle idéal pour de

futurs outils de bioinformatique

6-6-Nutrition :

Une équipe de l'université Toulouse-III-Paul-Sabatier montre que Physarum Polycephalum est capable de choisir le régime le plus adapté à son métabolisme lorsqu'il est mis en présence de nombreuses sources de carbone et d'azote différentes.

Science participative : Sont détaillées sous cette appellation les expériences réalisées par des personnes non professionnelles.

- Le BLOB préfère l'avoine au blé et au sarrasin, et ne semble pas apprécier le maïs.
- Le BLOB est connu pour détester le sel, il a par contre de l'attrait pour des épices comme le curcuma ou le paprika, moins pour d'autres comme le cacao ou la cannelle, le thé, ou le sucre.
- Le BLOB semble apprécier spécifiquement le gingembre.
- Le BLOB semble préférer les noix, et noisettes, aux grains de grenades. L'aciditéde la grenade pourrait en être la cause.
- Les huiles essentielles pourraient avoir un effet répulsif, peut-être par leur tropforte concentration. [21]

6-7-Anticipation des stimuli :

En générant de façon répétée des *stimuli* de chaud et de froid à Physarum Polycephalum et ce avec 60 minutes d'intervalle, des biophysiciens de l'université de Hokkaidō découvrent que le plasmode peut anticiper ces *stimuli* en y réagissant même quand ceux-ci sont absents. Ils montrent également que ces résultats peuvent être obtenus en appliquant les stimuli avec un intervalle de 30 ou 90 minutes. [11]

6-8-Une quasi immortalité :

Audrey Dussutour (chercheuse au CNRS) est une des rares française à étudier les BLOBs en laboratoire. Elle nous apprend qu'au bout de quelques mois, le BLOB se fatigue.
Pour le régénérer, il suffit alors de le placer à la lumière et de le faire sécher.
Il se rétracte et perd 70% de ses protéines et peut rester dans cet état plusieurs années sans bouger. Pour le « réveiller », il suffit de le mettre en contact avec de l'eau et le BLOB retrouve une seconde jeunesse. Audrey Dussutour travaille avec la même souche depuis

environ 10 ans.

Fait remarquable : cette souche provient d'un laboratoire américain qui l'utilisait depuis près de soixante ans.

6-9-Le BLOB : un être intelligent et doté d'une personnalité :

Cet organisme ne cesse d'étonner la science. En effet, les biologistes nous apprennent que les BLOBs sont dotés d'une intelligence. Cela va à contre courant de la pensée scientifique, qui a toujours cru que l'intelligence était liée aux neurones et donc au cerveau. Force est de constater que le BLOB, qui est je le rappelle unicellulaire et ne possède pas de cerveau, est capable de non seulement résoudre des labyrinthes, mais est aussi habile pour retenir des informations et même les transmettre.

Doté d'une mémoire spatiale, le BLOB est capable de ne pas repasser deux fois au même endroit. Afin d'éviter de réitérer des erreurs, il est également capable d'apprendre de ses congénères, en fusionnant. Trouver le plus court chemin dans un labyrinthe est un jeu d'enfant pour le BLOB. A tel point que certains chercheurs travaillent actuellement sur des « bio-ordinateurs » à base de Physarum .
Les BLOBs pourraient ainsi remplacer des circuits électroniques, avec l'avantage de pouvoir s'adapter en temps réel à leur environnement.

La société PhyChips utilise actuellement le BLOB pour concevoir des algorithmes pour les intégrer à leurs puces.

6-10-Apprentissage et mémorisation :

Des chercheurs toulousains ont montré qu'il pouvait apprendre. Et cela depuis qu'il se trouve sur Terre, soit 500 millions d'années, bien avant qu'un cerveau digne de ce nom y ait vu le jour. Les membres du Centre de recherches en cognition animale de la Ville rose, qui bichonnent plusieurs exemplaires de ces êtres primitifs, avaient déjà démontré qu'ils pouvaient communiquer et avait donc une sorte de mémoire.

Deux chercheuses allemandes viennent d'enfoncer le clou sur les capacités du blob à stocker des souvenirs. Dans une étude parue sur Proceedings of the National Academy of Sciences, elles démontrent qu'en fonction de la présence de nutriments, les petits tubes qui constituent son système veineux vont voir leur diamètre augmenter ou rétrécir, « imprimant ainsi l'emplacement » de la nourriture. Il a ainsi la capacité de stocker et lire des souvenirs en fonction de l'organisation et du diamètre de ses tubes.

Et ce serait une substance circulant dans le réseau veineux qui signalerait la présence de nourriture à un certain endroit pour qu'il puisse réagir et agir en fonction. De quoi ouvrir pour ces scientifiques des pistes sur les nouveaux matériaux intelligents, bio inspirés.[20]

7-Physarum et le réseau du métro de Tokyo :

Comment optimiser un réseau de transport et de distribution ? Que ce soit dans le domaine de l'énergie, des télécommunications ou encore celui des transports en commun, c'est une question que se posent quotidiennement les informaticiens et mathématiciens spécialisés en réseaux. Ils se creusent sans arrêt les méninges pour concevoir de nouveaux algorithmes capables de fournir des solutions adaptées. C'est que la question n'est pas simple : pour une situation donnée, quelle est la configurationla mieux adaptée ? Et si nous posions la question à un micro-organisme ? Voilà l'intrigante idée qu'ont eue, en 2010, des chercheurs japonais et britanniques. [1]

FIGURE 2.3 - Le réseau ferré du grand Tokyo.

L'équipe anglo-nipponne a soumis le problème de l'optimisation du réseau de métro de

Tokyo à ***Physarum Polycephalum*** , un micro-organisme unicellulaire qui, c'est le moins que l'on puisse dire, ne brille pas par son intelligence. Quelle forme ce réseau doit-il avoir pour garantir une efficacité de distribution maximale pour un coût minimal ? Le problème est loin d'être simple : il faut tenir compte des densités de population, de la géographie (reliefs, fleuves etc.) et de la localisation des principaux points névralgiques.Il faut que le réseau soit à la fois robuste et flexible, pour pouvoir proposer des itinéraires alternatifs en cas de défauts ponctuels : variations du trafic, pannes matérielles, suicides ou accidents. Dites-vous bien que des ordis tournent à plein temps pour recracher des solutions potables à ce genre de problèmes. Et pourtant, en l'espace d'une journée, *Physarum* a élaboré un réseau aussi performant que celui proposé par les ingénieurs japonais ! Les résultats ont été publiés en 2010, dans la revue Science.

Physarum Polycephalum est en effet habitué à résoudre un problème similaire : pour chercher de la nourriture et l'acheminer là où il en a besoin, il s'organise en réseau de petits tubes gluants. Pour des raisons évidentes, il ne s'épuise pas à construire un réseau aléatoire, au contraire : fort de ses millions d'années d'expérience, il est passé maitre dans l'art d'optimiser son réseau de distribution alimentaire. De façon générale, le réseau de *Physarum* est très réactif et dynamique : les branches changent sans cesse de forme et de taille, pour s'adapter aux variations des stocks de nourriture, et le réseau entier peut se déplacer de plusieurs centimètres en l'espace d'une heure ! Des chercheursavaient déjà exhibé ses talents dans une expérience où le champignon devait trouver le chemin le plus court pour relier deux points de nourriture dans un labyrinthe :

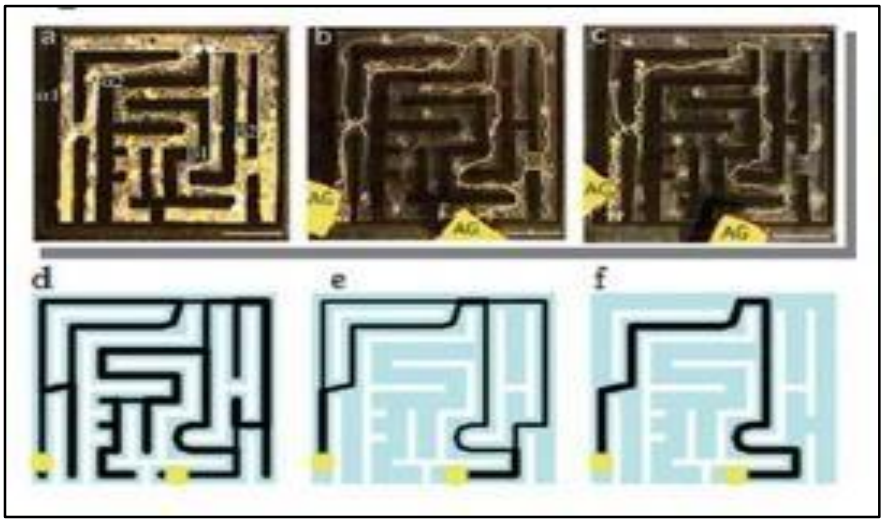

FIGURE 2.4 - Physarum dans un labyrinthe

Concrètement, voici comment les chercheurs ont procédé pour reproduire, à l'échelle de *Physarum* , la problématique du réseau de Tokyo, qui relie la capitale aux 36 villes voisines : ils ont disposé 37 points de nourriture, correspondants aux 36 villes et Tokyo, dans une boite de Petri, en essayant de respecter tant bien que mal la géographie de la région. Puis, ils ont implanté *Physarum* au point correspondant à Tokyo (le point jaune sur les photographies ci-dessous) et observé comment la moisissure se développait.

Celle-ci se plait d'ordinaire dans l'obscurité, aussi, pour simuler les contraintes géographiques, les chercheurs ont éclairé certains points que *Physarum* a soigneusement évités. Dans les premières heures de l'expérience, *Physarum* a exploré son environnement en tissant un réseau fin et dense de petites branches, appelées plasmodia, sur toute la surface de la boite. Ce faisant, il a repéré les 36 autres points de nourriture. Puis, pour ne pas perdre son énergie futilement, Physarum a réduit au minimum les branches du réseau qui n'étaient quasiment pas utilisées (elles ont toutefois été conservées, pour pouvoir être facilement réactivées en cas de défaillance localisées) tout en renforçant les "tronçons" les plus utiles. Le résultat est visible dans la série de photographies suivante :

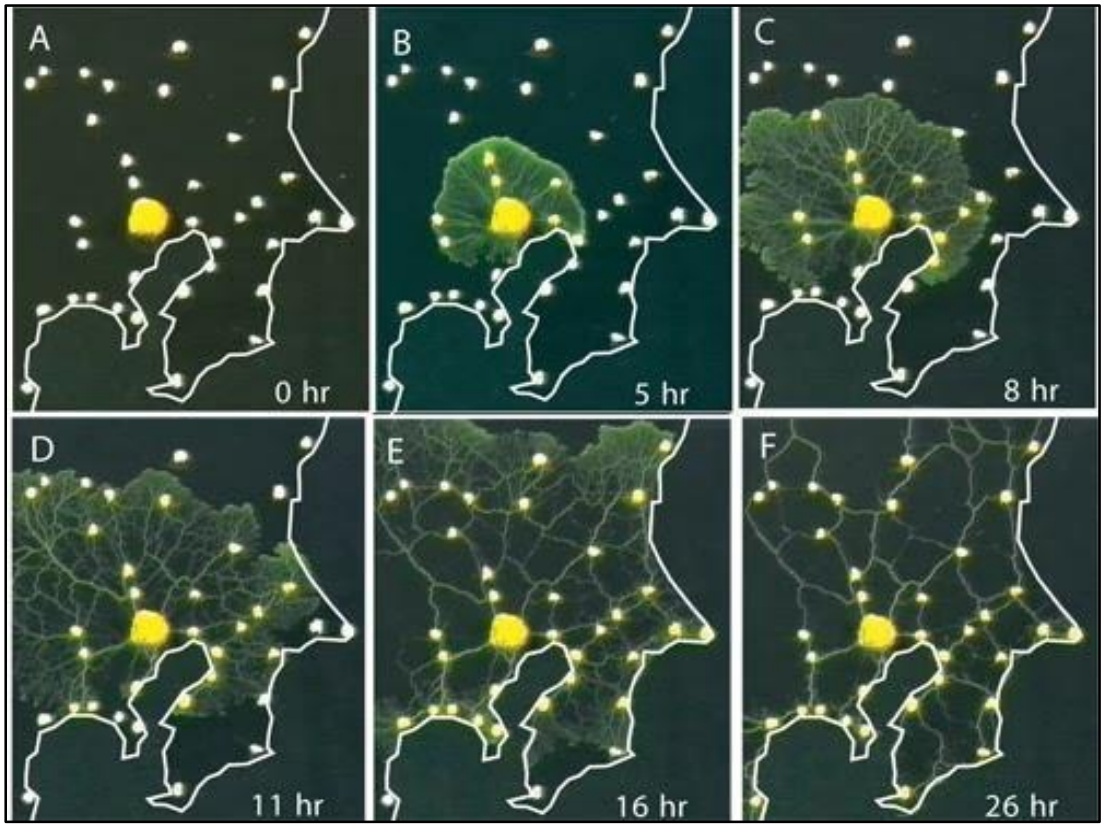

FIGURE 2.5 L'évolution du réseau de Physarum dans une boite de Petri

représentant la région de Tokyo.

Le lendemain, le réseau construit par *Physarum* ressemblait comme deux gouttes d'eau au vrai réseau de Tokyo et semblait montrer la même efficacité dans l'acheminement et la distribution de la nourriture :

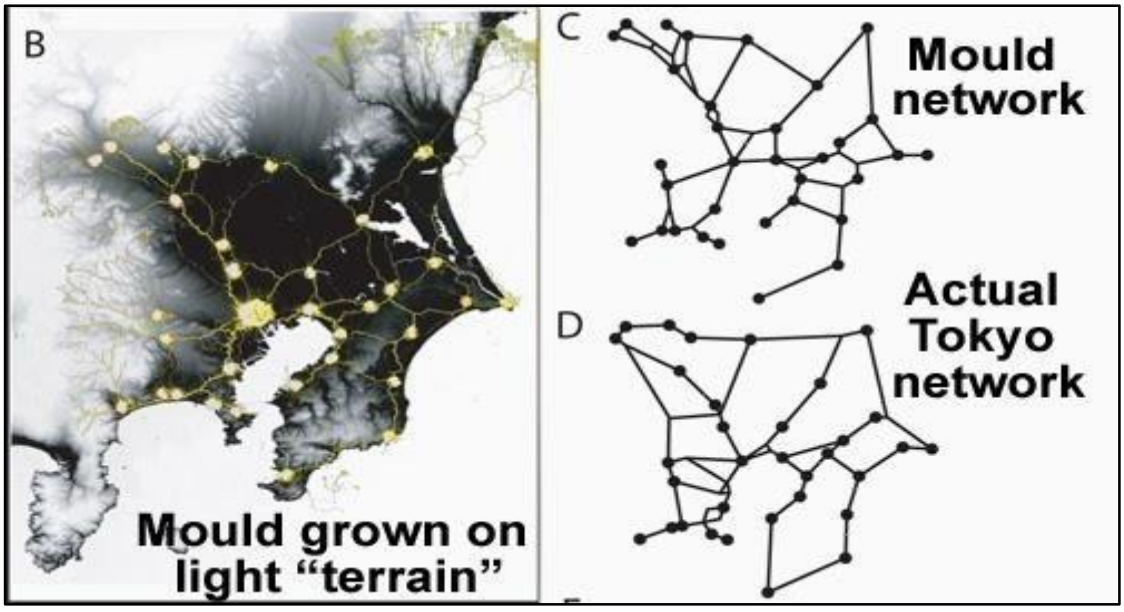

FIGURE 2.6 - (B) Extension du BLOB avec contraintes lumineuses pour reproduire les contraintes de terrain (C) Réseau produit par le BLOB (D) Actuel réseau de Tokyo.

7-1-L'algorithme Physarum et les problèmes d'optimisation :

Comme on ne peut décemment pas créer de nouveaux réseaux en s'appuyant sur un résultat visuel dans une boite de Petri, si bon fût-il, et qu'on ne peut pas non plus s'amuser à mettre en place un réseau de rails ou de câbles pour ensuite retirer ceux qui ne servent à rien, les scientifiques ont essayé de reproduire la stratégie de Physarum dans un algorithme informatique (une suite d'instructions élémentaires) extrêmement simple. Leur modèle mathématique va se révéler incroyablement efficace : ils parviennent non seulement à reproduire le réseau, mais à l'améliorer ! Comme le micro- organisme, l'algorithme commence par quadriller l'espace de façon aléatoire, puis renforce ou affaiblit les branches selon leur degré d'utilité. Puis, l'algorithme choisit deux villes au hasard et calcule le flux de liquide (dans ce cas, la nourriture) à travers le tube qui les relie. Pour cela, il se base sur des équations très simples de la mécanique des fluides, qui relient les proportions d'un tube (sa longueur et son diamètre) à la pression aux extrémités du tube. Enfin, l'algorithme procède à des ajustements : si le flux est important, le diamètre du tube est augmenté, sinon, il est diminué. Si la nouvelle configuration

augmente l'efficacité globale du réseau, les changements sont conservés

sinon l'opération est répétée avec deux nouvelles villes choisies au hasard. Le calcul, simple et rapide, permet une économie de ressources informatiques conséquente. Voici ce qu'on peut lire dans l'article de Paloma Bertrand consacré à ce sujet sur le site :

Universcience

« *Cet algorithme est simple et beau* », constate Bertrand Maury du Laboratoire de mathématiques de l'université Paris-Sud, « *et la démarche est originale. S'inspirer de ce que fait une petite bestiole qui obéit à des mécanismes élémentaires pour élaborer un modèle capable de résoudre en temps réel des problèmes complexes, c'est assez osé. Mais leurs résultats, si l'on en croit la publication de* Science, *semblent concluants.* ». [13]

8-Modéliser le réseau routier au Royaume-Uni et aux États-Unis avec L'algorithme Physarum :

Après le réseau du métro Tokyoïte, Andy Adamatzky et Jeff Jones, deux informaticiens de l'Université de Bristol, ont utilisé l'algorithme *Physarum* pour modéliser le réseau autoroutier en Angleterre. Dans leur étude, seules les 10 plus grandes villes du Royaume-Uni sont considérées. La moisissure virtuelle entame sa marche victorieuse à Londres bien évidemment. L'analyse des résultats montre que Physarum se débrouille plutôt bien encore une fois : le réseau obtenu ressemble au véritable réseau de routes, à la différence notable de l'autoroute reliant l'Angleterre à l'Ecosse. En introduisant des perturbations, les chercheurs observent comment le réseau pourrait être reconfiguré en cas de catastrophe naturelle ou d'accident majeur.

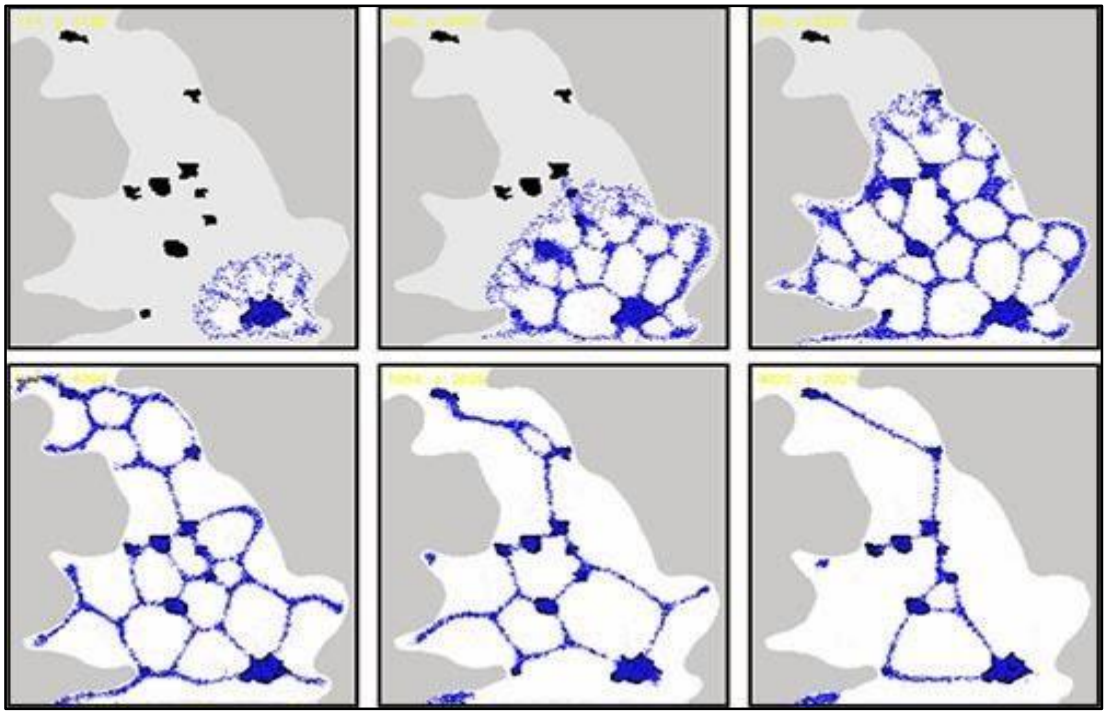

FIGURE 2.7 - Algorithme Physarum recréant le réseau routier anglais.

Les deux chercheurs se sont aussi amusés à modéliser le réseau des autoroutes aux États-Unis, en utilisant la méthode de l'expérience nipponne. La moisissure commence cette fois son périple à New-York avant de coloniser le territoire entier, en reliant les points de nourriture. Le résultat est assez sympathique :

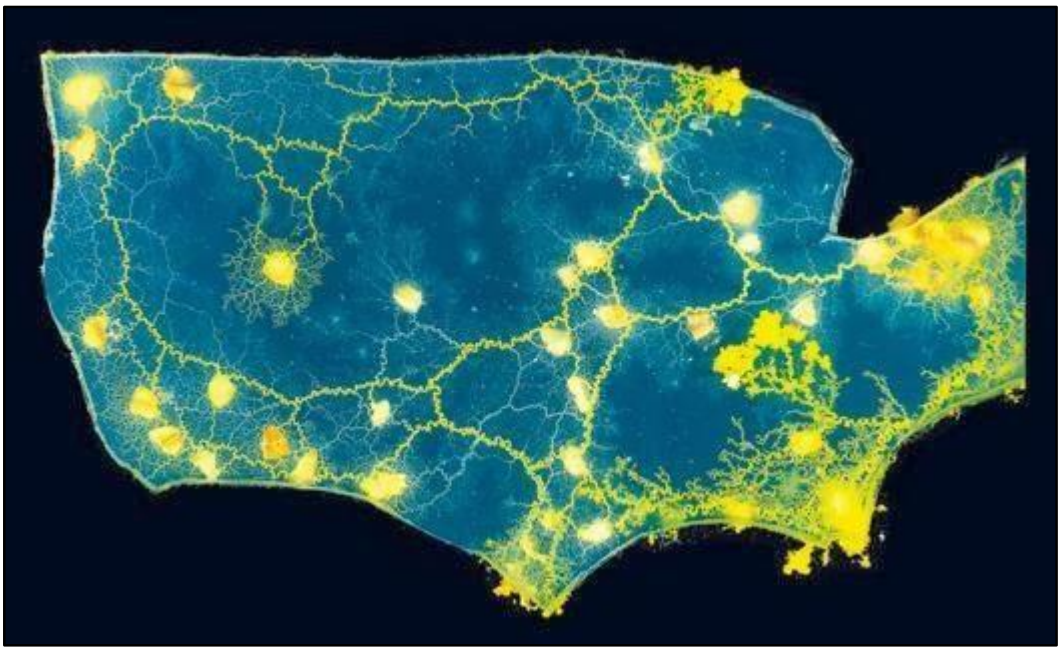

FIGURE 2.8 – Les aventures de Physarum aux U.S.A

Puis, ils ont utilisé l'algorithme *Physarum* pour modéliser diverses situations. Dans la vidéo ci-dessous, on peut voir comment *Physarum* , larguée à New-York, envahit progressivement le territoire des États-Unis, où chaque grande ville est un centre d'approvisionnement. Le micro-organisme relie une à une les grandes villes et finit par établir un réseau stable et robuste, qui laisse cependant une immense parcelle complètement à l'abandon.

Cette fois, ***Physarum*** est réparti de façon uniforme sur le territoire et doit s'adapter aux sources de nourriture, distribuées de façon irrégulière : abondantes dans les villes, elles sont au contraire très limitées dans les "campagnes". Au cours de la simulation, *Physarum* consomme rapidement les ressources rurales et le réseau se simplifie en se concentrant sur les grandes villes. Ces deux simulations ne reflètent pas tout à fait la réalité du réseau routier nord-américain, mais les résultats pourraient être exploités pourcréer de futurs réseaux. [1]

Aujourd'hui, ***Physarum Polycephalum*** est utilisé dans l'analyse de réseaux de transports dans de nombreux pays. Les résultats sont également transposés dans des domaines où se posent des problèmes d'optimisation des réseaux de flux.

9-Utilisation de BLOB pour concevoir un robot biologique :

Au premier abord, les myxomycètes ne soulèvent guère d'enthousiasme, mais ces protistes classés parmi les moisissures intéressent énormément les scientifiques. Ainsi, des chercheurs britanniques viennent d'annoncer qu'ils vont utiliser l'un de ces myxomycètes pour concevoir le tout premier robot biologique. Une telle avancée ne sera pas sans conséquence sur le domaine de la robotique algorithmique. Des scientifiques de l'University of the West of England (UWE) à Bristol (Royaume-Uni) ont présenté le «plasmobot», un robot biologique amorphe non basé sur le silicium. L'élément de base du robot est le plasmodium (la phase végétative) de Physarum Polycephalum , un myxomycète fréquent dans les forêts et les jardins du Royaume-Uni. Andy Adamatzky, du département d'informatique de l'UWE et chef du projet, déclare que les travaux précédents de l'équipe ont démontré les capacités de calcul du plasmodium. «Pour la plupart des gens, un ordinateur est un appareil disposant de logiciels conçus pour effectuer certaines tâches. Mais le plasmodium de ce myxomycèteest un organisme naturel qui

dispose de ses propres capacités de calcul.» «Il se déplace et cherche des sources de nourriture, et lorsqu'il en trouve, il ramifie vers elles plusieurs pseudopodes de protoplasme», ajoute le professeur Adamatzky. «Le plasmodium peut résoudre des problèmes complexes, par exemple calculer le chemin le plus court entre deux points et autres expressions logiques.» Lors d'expériences antérieures, l'équipe a réussi à faire transporter des objets au plasmodium. «Lorsqu'on le nourrit de flocons d'avoine, il produit des tubes qui oscillent et le font se déplacer dans une direction donnée, transportant avec lui des objets», souligne le professeur Adamatzky. «Nous pouvons aussi utiliser des stimulus lumineux ou chimiques pour le faire avancer dans une certaine direction.» Le plasmobot peut détecter des objets et les atteindre, et aussi transporter de petits objets dans des directions programmées, confirme le chercheur de l'UWE. «Les robots auront des entrées et sorties parallèles, un réseau de détecteurs et la puissance de calcul d'un superordinateur. Le plasmobot sera contrôlé par un gradient spatial de lumière, des champs électromagnétiques et les caractéristiques du substrat sur lequel il sera placé.» Le professeur Adamatzky souligne que le plasmobot sera «un robot amorphe totalement programmable et contrôlable, embarquant un ordinateur massivement parallèle». Ces derniers travaux serviront de base à d'autres études sur la façon de tirer parti des puissantes capacités de calcul de ce myxomycète. «Nous n'en sommes qu'aux toutes premières étapes de la compréhension de la façon dont le potentiel du plasmodium peut être utilisé. Dans les années qui viennent nous devrions pouvoir nous en servir, par exemple, pour apporter jusqu'à sa cible une petite quantité de substance chimique, en le guidant par la lumière, ou pour faciliter l'assemblage de microcomposants pour des machines», constate le professeur Adamatzky. L'étape suivante consisterait à utiliser les capacités du plasmodium à l'intérieur de notre corps, par exemple pour apporter des médicaments à un endroit donné. «Nous pourrions aussi avoir des milliers de petits ordinateurs faits de plasmodium, vivant à la surface de notre peau et conduisant des tâches de routine, libérant notre cerveau pour d'autres choses», imagine le professeur Adamatzky. «Beaucoup de scientifiques considèrent que c'est un développement possible de l'informatique amorphe, mais il reste totalement théorique à l'heure actuelle.» [25]

10-Un modele pour concevoir des portes logiques inspirées du BLOB :

Des chercheurs de l'Université Democritus de Thrace et de l'Université de l'Ouest de l'Angleterre ont développé un modèle de conception de portes logiques qui s'inspire en partie du comportement de P. Polycephalum . Leur papier, initialement publié sur arXiv,

sera bientôt publié dans le *Journal international d'informatique non conventionnelle* .

"Notre travail visait à concevoir un modèle basé sur des automates cellulaires (AC) moins compliqué pour simuler les capacités de calcul de P. Polycephalum , " Karolos-Alexandros Tsakalos, un doctorat étudiant qui a mené l'étude, a déclaré TechXplore. "L'objectif ultime était de concevoir des algorithmes bio-inspirés plus efficaces pour résoudre des problèmes de calcul difficiles."

L'étude menée par Tsakalos et ses collègues s'appuie sur les travaux antérieurs de l'équipe sur les outils de calcul inspirés du Physarum et les techniques d'apprentissage automatique. La nouvelle technique des chercheurs pour concevoir des portes logiques incarne les principes des automates cellulaires (AC), une classe de modèles discrets souvent utilisés pour résoudre l'informatique, problèmes de mathématiques et de physique. Les fonctionnalités de l'AC ont été combinées avec des techniques d'apprentissage automatique, conduisant à un modèle informatique robuste qui reflète le comportement de P. Polycephalum .

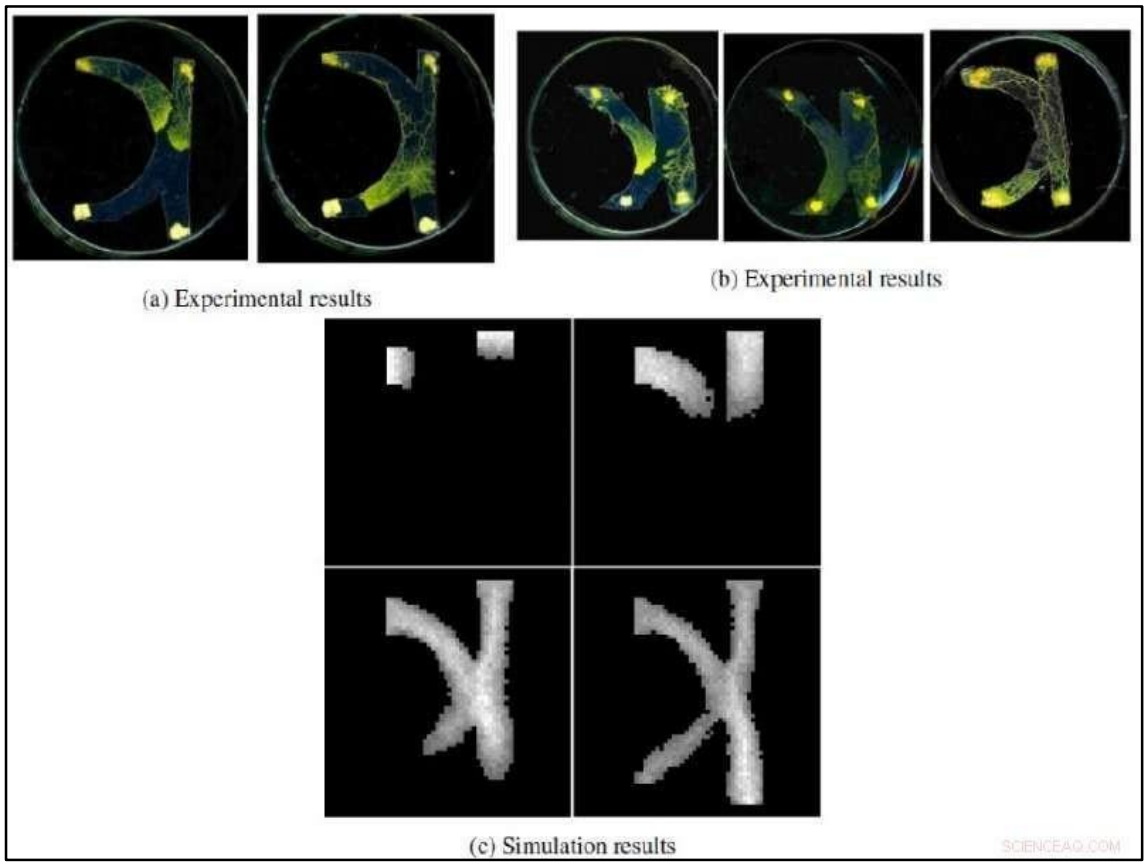

FIGURE 2.9 – Résultats de l'expérience de P. Polycephalum dans la conception de portes logiques .

{1, 1} → {1, 1} transformation du modèle proposé sur la porte P1.

"Notre modèle utilise l'apprentissage par renforcement au sein de chaque zone locale où des règles sont appliquées afin d'apprendre quel est le chemin approprié vers la destination finale, " Nikolaos Dourvas, un autre doctorat étudiant impliqué dans l'étude, a déclaré TechXplore. « Le principal avantage par rapport à ceux développés précédemment est sa simplicité, sa capacité à apprendre et à fournir des résultats stochastiquement différents, comme il a été découvert dans les expériences biologiques réelles."

La méthode simple introduite par Tsakalos, Dourvas, et leurs collègues peuvent être utilisés pour modéliser le comportement d'une variété d'organismes vivants. Dans leur étude, les chercheurs ont appliqué P. Polycephalum et testé ses performances dans la conception de portes logiques dans un environnement simulé, où le modèle devait identifier des chemins minimaux dans des labyrinthes contenant des sources de nourriture.

"La réalisation la plus significative de cette étude est la simulation réussie du comportement et donc des capacités de calcul de Physarum Polycephalum , à l'aide d'un modèle informatique, " Dr Michail-Antisthénis I. Tsompanas, un chercheur de l'Université de l'Ouest de l'Angleterre impliqué dans l'étude, a déclaré TechXplore. "Ce modèle s'inspire du parallélisme inhérent aux automates cellulaires, mais leur capacité à fournir des simulations adéquates de phénomènes physiques complexes est encore enrichie par la stochasticité des automates d'apprentissage et les capacités d'apprentissage correspondantes."

La technique de calcul bio-inspirée imaginée par Tsakalos, Dourvas, Tsompanas et leurs collègues se sont avérés très performants, modéliser efficacement les portes logiques dans de nombreux scénarios simulés. À l'avenir, leur modèle pourrait être appliqué à une variété de problèmes mathématiques et informatiques très complexes. Il pourrait également être adapté pour reproduire le comportement d'autres organismes vivants et phénomènes biologiques.

"Nous envisageons que le modèle bio-inspiré proposé puisse servir d'outil efficace dans d'autres études pour modéliser le comportement des autres, encore plus complexe, organismes vivants et résoudre des problèmes similaires représentés graphiquement, " Prof. Georgios Ch. Sirakoulis, un chercheur de l'Université Démocrite de Thrace qui a mené l'étude, a déclaré TechXplore. [26]

11-Conclusion :

Au final, l'étude de cet organisme unicellulaire donne de nouvelles idées pour un problème qui fait sécher les informaticiens depuis quelques décennies : calculer de manière décentralisée le plus court chemin dans un graphe. La décentralisation exige, ici, que chaque nœud effectue une partie du calcul en se basant uniquement sur l'information dont il dispose localement. Un algorithme d'inspiration biologique est un exemple assez spectaculaire d'algorithme naturel, dont l'étude laisse paraître de possibles solutions. Afin d'évaluer une telle solution, nous proposons comme métrique la complexité temporelle plus en détail au chapitre suivant.

CHAPITRE III

Développement de l’algorithme de Djikstra

Chapitre III : Développement de l'algorithme de Djikstra

1-Introduction :

Le développement d'une solution innovante sur la base du BLOB dépasse le cadre de ce travail. Notre principal objectif est de présenter une vision nouvelle de la problématique du plus court chemin à travers la redéfinition du monde qui nous entoure bouleversé par les récents évènements cités précédemment. A cet effet, nous revenons aux fondamentaux avec une implémentation de l'algorithme de Djikstra présentée dans ce chapitre. Nous discutons par la suite la possible utilisation de la complexité temporelle comme métrique d'évaluation détaillée dans une étude.

2-Environnement de développement :

2-1-Le langage de programmation java :

Le langage Java est un langage de programmation orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy, cofondateur de Sun Microsystems. Java a été officiellement présentée le 23 mai 1995 au SunWorld. La société Oracle racheta alors la société Sun en 2009, ce qui explique pourquoi ce langage appartient désormais à Oracle. La particularité et l'intérêt de Java réside dans sa portabilité entre les différents systèmes d'exploitation tels que Unix, Windows, ou MacOS. Un programme développé en langage Java, peut ainsi s'exécuter sur toutes les plateformes, grâce à ses Framework associés visant à garantir cette portabilité .

- C'est l'un des langages de programmation les plus populaires au monde.
- Il est facile à apprendre et simple à utiliser.
- C'est open-source et gratuit.
- C'est sécurisé, rapide et puissant.
- Il a un énorme soutien de la communauté (des dizaines de millions de développeurs).

2-2-L'environnement NetBeans :

NetBeans est un environnement de développement intégré (EDI), placé en open source par Sun en juin 2000 sous licence CDDL (Common Développent and Distribution License) et GPLv2. En plus de Java. NetBeans permet la prise en charge native de divers langages tels le C, le C, le JavaScript, le XML, le Groovy, le PHP et le HTML,ou d'autres (dont Python et Ruby) par l'ajout de greffons. Il offre toutes les facilités d'un IDE moderne (éditeur avec coloration syntaxique, projets multi-langage, re factoring, éditeur graphique d'interfaces et de pages Web). NetBeans est disponible sous Windows, Linux, Solaris, Mac OS X ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle Java).

NetBeans constitue par ailleurs une plateforme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)).[14]

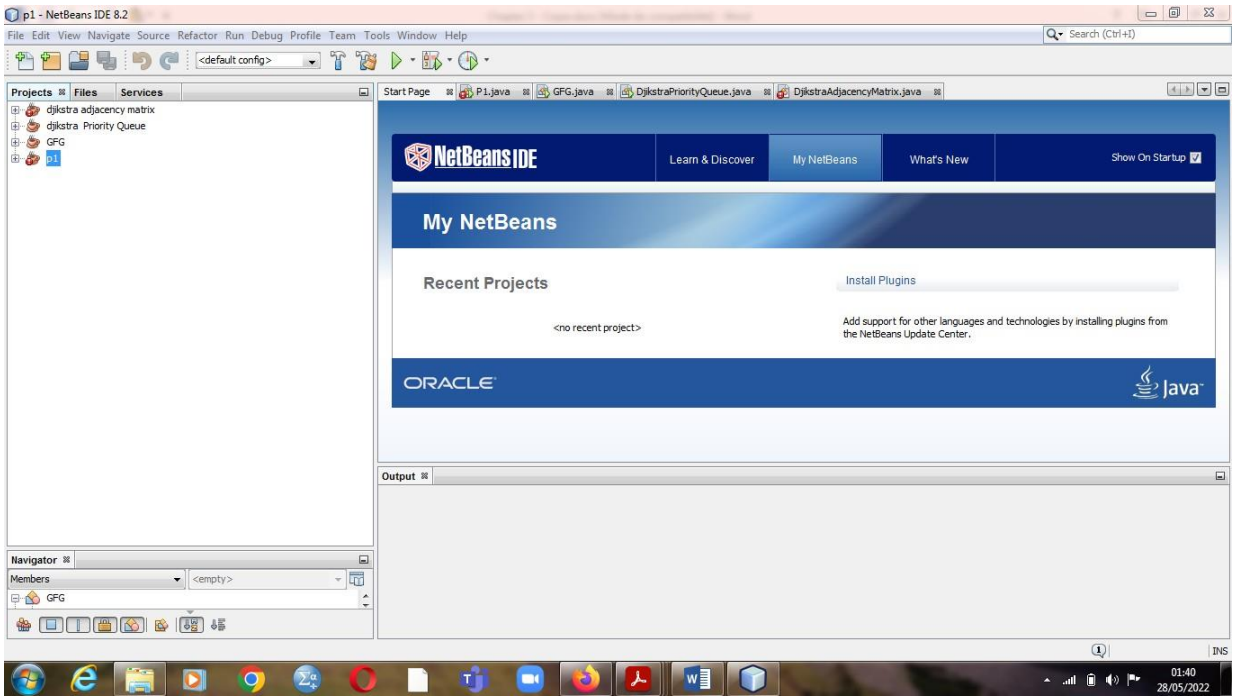

FIGURE 3. 1 - Page de démarrage de NetBeans

3-Complexité d'un algorithme :

La notion de complexité formalise la notion d'efficacité d'un programme : capacité à fournir le résultat attendu dans un temps minimal

Elle consiste en la détermination des ressources nécessaires en temps de calcul,mémoire.

Le calcul de la **complexité** d'un algorithme permet de mesurer sa **performance**. Il existe deux types de complexité :

- **complexité spatiale** : permet de quantifier l'utilisation de la **mémoire**
- **complexité temporelle** : permet de quantifier la **vitesse** d'exécution. [3]

3-1-Complexité temporelle :

L'objectif d'un calcul de complexité algorithmique temporelle est de pouvoir comparer l'efficacité d'algorithmes résolvant le même problème. Dans une situation donnée, cela permet donc d'établir lequel des algorithmes disponibles est le plus optimal.

Réaliser un calcul de complexité en temps revient à compter le nombre d'opérations élémentaires (affectation, calcul arithmétique ou logique, comparaison…) effectuées par l'algorithme.

Puisqu'il s'agit seulement de comparer des algorithmes, les règles de ce calcul doivent être indépendantes :

- du langage de programmation utilisé ;
- du processeur de l'ordinateur sur lequel sera exécuté le code ;
- de l'éventuel compilateur employé.

Par soucis de simplicité, on fera l'hypothèse que toutes les opérations élémentaires sont à égalité de coût, soit 1 « unité » de temps.

*Exemple : y = x / 5 : 1 division + 1 affectation = 2 « unité*s »

La complexité en temps d'un algorithme sera exprimé par une fonction, notée *T* (pour *Time*), qui dépend :

- de la taille des données passées en paramètres : plus ces données seront volumineuses, plus il faudra d'opérations élémentaires pour les traiter. On notera *n* le nombre de données à traiter.
- de la donnée en elle même, de la façon dont sont réparties les différentes valeurs qui la constituent.

Cette remarque nous conduit à préciser un peu notre définition de la complexité en temps. En toute rigueur, on peut en effet distinguer deux formes de complexité en temps :

- la complexité dans le **meilleur des cas** : c'est la situation la plus favorable, *par exemple : recherche d'un élément situé à la première position d'une liste*

- la complexité dans le **pire des cas** : c'est la situation la plus défavorable, par exemple : recherche d'un élément dans une liste alors qu'il n'y figure pas On calculera le plus souvent la complexité dans le pire des cas, car elle est la pluspertinente. Il vaut mieux en effet toujours envisager le pire.

3-2-Ordre de grandeur :

Pour comparer des algorithmes, il n'est pas nécessaire d'utiliser la fonction *T*, mais seulement l'ordre de grandeur asymptotique, noté *O* (« grand O »). Une fonction T(n) est en O(f(n)) (« en grand O de f(n)«) si : $\exists n0 \in N, \exists c \in R+, \forall n \in R+, n \geq n0 \Rightarrow |T(n)| \leq c|f(n)|$
Autrement dit :
T(n) est en O(f(n)) s'il existe un seuil n_0 à partir duquel la fonction T est toujours dominée par la fonction f, à une constante multiplicative fixée c près

Exemples :

- T1(n)=7=O(1)
- T2(n)=12n+5=O(n)
- T3(n)=4n2+2n+6=O(n2)
- T4(n)=2+(n−1)×5=O(n) . [23]

3-3-Classes de complexité :

O	Type de complexité
$O(1)$	constante
$O(log(n))$	logarithmique
$O(n)$	linéaire
$O(n \times log(n))$	quasi-linéaire
$O(n^2)$	quadratique
$O(n^3)$	cubique
$O(2^n)$	exponentielle
$O(n!)$	factorielle

TABLE 3.1 – Les Classes de Complexité

n $f(n)$	$\lg n$	n	$n \lg n$	n^2	2^n	$n!$
10	0.003 μs	0.01 μs	0.033 μs	0.1 μs	1 μs	3.63 ms
20	0.004 μs	0.02 μs	0.086 μs	0.4 μs	1 ms	77.1 years
30	0.005 μs	0.03 μs	0.147 μs	0.9 μs	1 sec	8.4×10^{15} yrs
40	0.005 μs	0.04 μs	0.213 μs	1.6 μs	18.3 min	
50	0.006 μs	0.05 μs	0.282 μs	2.5 μs	13 days	
100	0.007 μs	0.1 μs	0.644 μs	10 μs	4×10^{13} yrs	
1,000	0.010 μs	1.00 μs	9.966 μs	1 ms		
10,000	0.013 μs	10 μs	130 μs	100 ms		
100,000	0.017 μs	0.10 ms	1.67 ms	10 sec		
1,000,000	0.020 μs	1 ms	19.93 ms	16.7 min		
10,000,000	0.023 μs	0.01 sec	0.23 sec	1.16 days		
100,000,000	0.027 μs	0.10 sec	2.66 sec	115.7 days		
1,000,000,000	0.030 μs	1 sec	29.90 sec	31.7 years		

TABLE 3.2 - Comparaison en pratique du temps de calcul qu'impliquent les différentes complexités, en supposant qu'une opération s'exécute en une nanoseconde.

4-Complexité de l'algorithme de Djikstra :

Maintenant, calculons le temps d'exécution de l'algorithme de Djikstra en utilisant une file d'attente de priorité min-heap binaire comme la frange. Soit M et N le nombre d'arêtes et le nombre de sommets dans le graphique respectivement.(En théorie des graphes, M et N font référence à l'ensemble des bords d'un graphe et à l'ensemble des sommets d'un graphique ; M et N sont les tailles de ces ensembles.)

```
    // Methode 1
    // Algorithme de Djikstra
    public void dijkstra(List<List<Node> > adj, int src)
    {
            this.adj = adj;

            for (int i = 0; i < V; i++)
                dist[i] = Integer.MAX_VALUE;

            // Ajouter un nœud source à la file d'attente de priorité
            pq.add(new Node(src, 0));

    // Distance à la source est 0
            dist[src] = 0;

            while (settled.size() != V) {

                //Terminer la vérification de l'état lorsque
                //la file d'attente prioritaire est vide, retourner
                if (pq.isEmpty())
                    return;

                // Supression du nœud de distance minimale
                // de la file d'attente prioritaire
                int u = pq.remove().node;

               //Ajout du nœud dont la distance est finalisée
               if (settled.contains(u))

                    // Continuer l'exécution des sauts de mots clés
                    // pour la vérification suivante
                    continue;

                //Nous n'avons pas besoin d'appeler e_Neighbors(u)
                // si u est déjà présent dans l'ensemble réglé
               settled.add(u);

               e_Neighbours(u);
        }
}
```

FIGURE 3. 2 – Capture d'écran d'une partie de code de l'algorithme de Djikstra .

1- Ajouter tous les N sommets du graphique à un min-heap binaire prend un total de O(N) temps.

2- La mise à jour de la valeur de priorité d'un sommet dans la file d'attente de priorité peut être effectuée avec un mini-tas binaire en le supprimant, puis en le lisant avec une valeur de priorité mise à jour. Si nous conservons une carte de hachage des sommets et de leurs indices dans le tableau binaire min-heap et supposons que l'opération de mise de la carte de hachage prend un temps constant, alors nous pouvons supprimer et lire un vertex à un min-heap binaire en O(log N) temps. Cela arrive à un sommet au plus une fois pour chacun de ses voisins. Ainsi, cela se produira tout au plus M fois au cours d'une série entière de l'algorithme de Djikstra. Puisque chaque mise à jour de valeur prioritaire prend O(log N) temps, le total de toutes les mises à jour de valeur prioritaire prend O(M log N).

3- Si nous conservons une carte de hachage des sommets avec leurs valeurs prioritaires, alors accéder à la valeur prioritaire d'un sommet est également une opération de temps constant. Ceci rend le calcul de la valeur de priorité mise à jour d'un sommet une opération de temps constant. Comme nous n'avons qu'à calculer les valeurs de priorité mises à jour O(M) fois, tous les calculs de mise à jour prennent un total de O(M) temps.

4- Notez que chaque sommet est retiré de la frange exactement une fois, et jamais lu à la frange (à l'exclusion des mises à jour des valeurs prioritaires). L'algorithme de Djikstra supprime uniquement de la file d'attente prioritaire N fois, et chaque suppression prend O(log N) temps pour un total de O(N log N) temps pour tous les retraits de vertex.

5- Vérifier si la file d'attente prioritaire est vide est une opération de temps de constaint et se produit O(N) fois (une fois juste avant chaque sommet est supprimé de la file d'attente de priorité). Ainsi, tous les contrôles vides prennent un total de O (N) temps.

6- Itérer par les voisins d'un sommet peut être fait dans le temps proportionnel au degré de ce sommet (le nombre de voisins qu'il a) avec une liste de contiguïté. Par conséquent itérer sur tous les sommets voisins au cours d'une série de l'algorithme de Djikstra prend O (M) temps.

En additionnant ces temps d'exécution, nous avons O(M log N) pour toutes les mises à jour de valeur prioritaires et O(N log N) pour supprimer tous les vertices (il y a aussi d'autres termes additifs O(M) et O(N), mais ils sont dominés par ces deux termes). Cela signifie que le temps d'exécution de l'algorithme de Djikstra utilisant un min-heap binaire comme une file d'attente prioritaire est O((M +N)logN).

Un graphique dirigé est faiblement connecté si le remplacement de tous ses bords dirigés par des bords non dirigés produit un graphique non dirigé connecté. Si nous supposons que le graphique d'entrée est faiblement connecté, alors le graphique doit avoir au moins N - 1 bords. Cela implique que N est en O(M), ce qui signifie que nous pouvons simplifier le temps d'exécution ci-dessus à O(M logN).

5-Conclusion :

Dans ce chapitre nous avons présenté notre implémentation de l'algorithme de Djikstra en langage Java en utilisant une file de priorité, ensuite nous avons proposé la complexité temporelle comme une métrique pour évaluer cette solution .

Conclusion générale

L'objectif de ce travail a été l'étude de l'algorithme de Djikstra avec d'autres critères pris en compte qui n'avaient pas existé auparavant au moment de la première publication de l'algorithme en 1959.

L'accent a particulièrement été mise sur l'utilisation des techniques bio-inspirés. Un choix justifié par la nature même du problème qui en réalité formulé sous forme d'un ensemble d'objectifs.

Parmi les organismes vivants, nous avons présenté le Physarum polycephalum dont l'intelligence surnaturelle est utilisée pour résoudre les problèmes du type "trouver le chemin le plus court possible".

Nous avons utilisé les derniers travaux, le calcule de la complexité de l'algorithme de Djikstra pour mesurer sa performance et sa optimalité .

Ce travail ouvre diverses perspectives.

• Un premier point concerne Adoption l'algorithme de Djikstra comme base fondamentale pour résoudre les problèmes de " plus court chemin" avec la nécessité de définir des nouvelles critères et de tenir en compte les changements actuels (la préservation de l'environnement, période post-covid …)

- Un deuxième point concerne La nécessité de simuler les organismes dans leur vie dans la nature pour résoudre les problèmes quotidiens dans lesquels les êtres humains tombent, de sorte que les machines et les robots deviennent biologiques.

• Un troisième point concerne l'objectif d'un calcul de complexité algorithmique temporelle est de pouvoir comparer l'efficacité d'algorithmes résolvant le même problème. Dans une situation donnée, cela permet donc d'établir lequel des algorithmes disponibles est le plus optimal.

Ceci étant dit, ce n'est qu'un rayon de lumière parmi tant d'autres. Il faudrait maintenant aller au-delà, car il reste beaucoup à découvrir pour donner un plus grand élan à ce genre d'études .

Bibliographie

1) Andrew Adamatzky et Jeff Jones, « Road planning with slime mould: If Physarum built motorways it would route M6/M74 through Newcastle» , Université de l'Ouest de l'Angleterre, Royaume-Uni.
2) Andrew Adamatzky, « Steering plasmodium with light: Dynamical programming of *Physarum*machine » [archive], arXiv, 6 août 2008.
3) Anthony Labarre , « Structures de données et algorithmes fondamentaux » , Année académique 2020–2021,université Gustave Eiffel.
4) Audrey Dussutour, « *Tout ce que vous avez voulu savoir sur le blob sans jamais le demander »*,Éditions Équateur Sciences, avril 2017, p. 87
5) Didier Müller, « Introduction à la théorie des graphes », CAHIER NO 6,comission romande de mathématique , France, 2011.
6) Ian Stewart , « Les mathématiques du vivant ou La Clé des mystères del'existence » , Éditions Flammarion,2016
7) José L. Galán-García , Gabriel Aguilera-Venegas, María Á. Galán-García, Pedro Rodríguez-Cielos, «A new Probabilistic Extension of Dijkstra's Algorithm to simulate more realistic traffic flow in a smart city»,Applied Mathematics and Computation 267 (2015) 780–789
8) M.Rigo, « Théorie des graphes,Deuxièmes bacheliers en science mathématiques », Université de Liége, 2010.
9) Mingjun Wei et Yu Meng , «Research on the optimal route choice based on improved Dijkstra» 2014 IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA), 2014.
10) N. Deo et C.Y. Pang, « Shortest path algorithms : Taxonomy and Annotations Networks », 14:275 _ 323, 1984 .
11) Tetsu Saigusa, Atsushi Tero, Toshiyuki Nakagaki et Yoshiki Kuramoto, « Amoebae anticipateperiodic events », *Physical Review Letters*, vol. 100, 2008.
12) Torcq Théo, « Mémoriser sans cerveau », *Pour la Science*, 2021/5 (N° 523 - mai), p. 13a-13a. DOI :10.3917
13) Toshiyuki Nakagaki, Hiroyasu Yamada, Agota Toth ,"Path finding bytube morphogenesis in an amoeboid Organism» ,2001.

14) Zroug Rihab, « Développement et implémentation d'un algorithme génétique pour la détection de communautés dans les réseaux sociaux», MEMOIRE de MASTER Informatique Décisionnel et Optimisation ,universitéE MOHAMED BOUDIAF - M'SILA ,2019

Webographie

15) https://complex-systems-ai.com/recherche-de-chemin-theorie-des-graphes/algorithme-de-dijkstra/.Dernière visite : 01/01/2022 .

16) http://www.w3ii.com/fr/graph_theory/graph_theory_introduction.html. Dernière visite : 18/03/2022.

17) https://fr.acervolima.com/pourquoi-l-algorithme-de-dijkstra-echoue-t-il-sur-les-poids-negatifs/. Dernière visite : 18/03/2022.

18) https://www.scaler.com/topics/data-structures/dijkstra-algorithm/. Dernière visite : 18/03/2022.

19) http://sweetrandomscience.blogspot.com/2014/01/le-metro-de-tokyo-et-les-routes-des.html.Dernière visite : 01/01/2022 .

20) https://peakd.com/francostem/@yann85/le-physarum-polycephalum-le-blob.Dernière visite : 01/01/2022

21) https://blob.ytterbium.eu/category/physarum-polycephalum.html.Dernière visite : 18/03/2022.

22) https://www.techno-science.net/glossaire-definition/Physarum-polycephalum.html#ref_1.Dernière visite : 01/01/2022 .

23) https://info.blaisepascal.fr/nsi-complexite-dun-algorithme.Dernière visite : 05/06/2022

24) http://yallouz.arie.free.fr/terminale_cours/graphes/graphes.php?page=g3 Dernière visite : 10/06/2022

25) https://cordis.europa.eu/article/id/31207-british-scientists-use-mould-to-create-biological-robot/fr. Dernière visite : 10/06/2022

26) http://fr.scienceaq.com/Electronics/1001078721.html. Dernière visite : 10/06/2022

Résumé

Le problème de plus court chemin est un problème d'optimisation bien connu. Parmi les méthodes sophistiquées utilisées pour choisir les chemins : l'algorithme de Djikstra. Djikstra est une méthode simple et puissante qui tient compte de plusieurs critères. Ces critères ont changé aujourd'hui en raison des circonstances actuelles. En fait, les scientifiques ont exploité beaucoup de technologies de bio-intelligence pour résoudre ces types de problèmes d'optimisation des réseaux y compris le physarum polycephalum (BLOB). Des études récentes ont prouvé que cet être mystérieux montre une excellente intelligence dans la construction du réseau biologique.

Ce projet de fin d'études vise à présenter une nouvelle vision de la problématique du plus court chemin à travers la redéfinition du monde qui nous entoure bouleversé par les récents évènements (les contraintes causées par la pandémie du COVID, la guerre en Ukraine et ses répercussions sur le prix des carburants…) .à la fin nous avons présenté une implémentation de l'algorithme de Djikstra avec sa complexité temporelle comme métrique d'évaluation .

Mots clés : plus court chemin, optimisation, l'algorithme de Djikstra, le physarum polycephalum (BLOB), complexité temporelle .

Abstract

The shortest path problem is a well-known optimization problem. Among the sophisticated methods used to choose paths: the Djikstra algorithm. Djikstra is a simple and powerful method that takes into account several criteria. These criteria have changed today because of the current circumstances. In fact, scientists have exploited many bio-intelligence technologies to solve these types of network optimization problems including Physarum Polycephalum (BLOB). Recent studies have shown that this mysterious being shows excellent intelligence in the construction of the biological network.

This graduation project aims to present a new vision of the problem of the shortest path through the redefinition of the world that surrounds us upset by recent events (the constraints caused by the COVID pandemic, the war in Ukraine and its impact on fuel prices…). at the end we presented an implementation of the Djikstra algorithm with its time complexity as an evaluation metric.

Keywords : The shortest path , optimization , Djikstra algorithm , Physarum Polycephalum (BLOB) , time complexity

الملـخـص

مشكلة المسار الأقصر هي مشكلة تحسين معروفة. من بين الأساليب المتطورة المستخدمة لاختيار المسارات: خوارزمية جيكسترا. جيكسترا هي طريقة بسيطة وقوية تأخذ بعين الاعتبار العديد من المعايير. لقد تغيرت هذه المعايير اليوم بسبب الظروف الحالية. في الواقع، استغل العلماء العديد من تقنيات الذكاء الحيوي لحل هذه الأنواع من مشاكل تحسين الشبكة بما في ذلك: Physarum Polycephalum (BLOB) حيث أظهرت الدراسات الحديثة أن هذا الكائن العجيب يملك ذكاء ممتازًا في بناء الشبكة البيولوجية.

يهدف مشروع التخرج هذا إلى تقديم رؤية جديدة لمشكلة أقصر مسار من خلال إعادة تعريف الظروف التي تحيط بنا المتأثرة من الأحداث الأخيرة (الآثار الناجمة عن جائحة COVID ، والحرب في أوكرانيا وتأثيرها على أسعار الوقود...). في النهاية قدمنا تنفيذ خوارزمية جيكسترا مع تعقيدها الزمني كمقياس للتقييم.

الكلمات المفتاحية : المسار الأقصر ، تحسين ، خوارزمية جيكسترا ، Physarum Polycephalum (BLOB) ، التعقيد الزمني