**Rapport de ERO**

**Au commencement du projet nous décidons de nous orienté vers la création du classe GRAPH possédant une liste de NODE lié entre eux par des LINKS. Cette méthode avait l’avantage d’être simple à créer et à utiliser. Avec un VISITOR on a pu lancer nos premiers algorithmes de recherche de chemin le plus court possible pour le drone. Le graphe était non-orienté car le sens de circulation ou le nombre de voies de nous intéresse pas dans le cas du drone. Nous analysions différents survols du graphe et ne le retenions que si le temps d’analyse était plus court. Ce mode d’analyse est rapide pour de petit graphe mais pour de graphe plus conséquent requiert plus de temps et ne garanti pas d‘avoir trouvé la meilleure méthode.**

**Nous somme donc revenu en arrière et utilisons désormais Jupyter dont le bénéfice est d’avoir des résultats plus visuels. De plus nous avons changé la méthode de détection de chemin le plus cours en passant par des graphiques Eulériens. Un graphique Eulérien est un graphe où il est possible de parcourir toute les arrêtes une fois et une seule. Cette propriété est donc fortement intéressante dans notre cas puisqu’il nous permet de gagner beaucoup de temps lors du survol du drone et lors du déneigement. Notre priorité c’est donc reporté a trouver le cycle Eulérien. Tous les graphes ne sont pas forcément Eulérien mais comme nous allons le voir plus tard, tout les quartiers de Montréal, la position des rues et des croisements forment des graphes Eulérien.**

**Pour notre graphe de test (METTRE GRAPH DE TEST ICI) nous choisissons un graphe non Eulérien. Nous créons une fonction qui permet de transformer les graphes non Eulérien en graphe Eulérien. Le résultat est donc le graphe ci-dessous (GRAPHE EULERIEN).**

**Pour la partie de la déneigeuse lors de la première tentative nous avons redessiné un nouveau graphe orienté cette fois ci. (GRAPHE ORIENTE) Nous considérons qu’une déneigeuse effectue dans un seul sens une rue à double sens mais que par conséquent elle double le temps mis pour déneiger la zone. De plus une rue a sens unique, dû à la présence de neige, est bloqué et donc nos machines peuvent l’emprunter dans n’importe quel sens. Voici le résultat de notre premier essais (GRAPHE PARCOURU). Nous voyons qu’il est illogique et qu’une solution évidente pourrait-être : (GRAPHE COULEUR). Nous avons omis l’ajout de différent type de machine qui sont spécialisés certaines pour de grandes surfaces et d’autre pour des trottoirs.**

**Dans notre deuxième tentative sur Jupyter nous rajoutons ces paramètres et obtenons ce résultat : (GRAPHE JUPYTER DENEIGEMENT)**

**Ensuite pour notre modèle de coût nous pensons qu’il nous faut 2 drones pour notre graphe de test pour le parcourir efficacement et ensuite lancé nos équipes de déneigement a l’assaut des rues. De plus nous avons constitué un objet paramétrable très simplement nous permettant en fonction du nombre de machines et personnes déployé dans la municipalité le coût d’une opération. Il prend en compte le salaire horaire d’un ouvrier, le coût d’exploitation par heure d’une machine et le temps de déploiement des équipes. C’est un outil très basique qui ne prends pas en compte les casses, pannes, augmentations de salaire, primes, salaire différents selon les postes. Ce n’est qu’un début d’estimation des coûts mais il nous semble plutôt correcte dans l’idée.**

**Sur une semaine nous avons simuler un départ de peu de personne pour de légères chute de neige, le lendemain l’envoi de personnes sur les cites stratégiques à protéger de la neige pour préparer des fortes chutes qui s’annoncent pour les 2 jours a venir et une mission lourde où beaucoup de personnel est dehors pour charger et évacuer la neige. Ce cumul d’activité engendre des coûts que nous pouvons comparer au budget annuel que la commune de Montréal. Nos résultats sont pour une semaine le coûts est de 2.150.400$ avec une dépense annuel de 177.000.000$ pour Montréal nous voyons que pour notre modèl théorique il nous manque des coût que nous n’avons pas pris en compte.**