2. Connections

On considère une grille carrée sur laquelle un nombre fini de cases sont accessibles. On place un drapeau **Départ** et un drapeau **Arrivée** sur deux cases accessibles de la grille.

Établissons une méthode qui permet de déterminer, en fonction des cases accessibles et des positions des drapeaux, s'il est possible de relier le drapeau **Départ** au drapeau **Arrivée** par un chemin qui passe une et une fois par chaque case accessible ? On autorise uniquement les déplacements horizontaux et verticaux (pas en diagonale).

Il pourra être intéressant d'établir un algorithme qui prend en paramètre la grille et résout le problème.

Pistes de recherche (à explorer dans l'ordre, ou pas !)

1. *Plusieurs drapeaux **Départ**/**Arrivée**.* On suppose qu'il a désormais n drapeaux **Départ** et autant de drapeaux **Arrivée** disposés sur la grille. L'objectif est de tracer n chemins, chacun reliant un drapeau **Départ** à un drapeau **Arrivée**, de sorte que chaque case accessible soit parcourue par un unique chemin.
2. *Drapeaux de couleur.* En plus du point précédent, on associe une couleur à chaque drapeau et on impose que, pour être reliés, deux drapeaux doivent être de la même couleur.
3. *Cases de couleur.* On marque certaines cases accessibles par une couleur et on impose que ces cases soient parcourues par un chemin reliant des drapeaux de leur couleur.
4. *Cases « doubles ».* On marque certaines cases accessibles comme étant « doubles » et on impose que ces cases soient parcourues par exactement deux chemins (au lieu d'un seul).
5. *Tunnels.* On ajoute des tunnels. Un tunnel est une paire de cases **Entrée**/**Sortie** (non nécessairement adjacentes) telle que un chemin arrivant sur la case **Entrée** doit être interrompu et reprendre depuis la case **Sortie**.
6. *Grille triangulaire.* On rempace la grille carrée par une grille triangulaire.
7. … les limites de votre imagination !

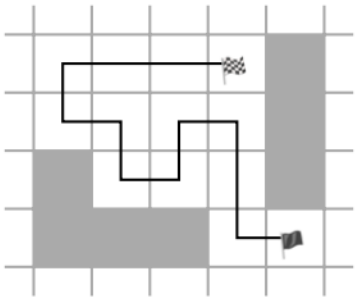

Fig. 1. – Grille qui admet un chemin

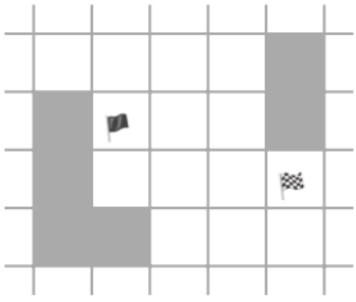

Fig. 2. – Grille qui n'admet pas de chemin