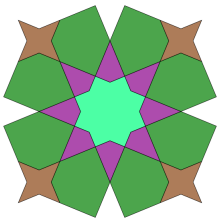
TD d'Algorithmique et structures de données
Tables de hachage

Équipe pédagogique Algo SD

1 Identification de polygones



Dans cet exercice, on dispose d'un ensemble de polygones, stockés dans un vecteur. Chaque polygone est stocké sous la forme d'une suite de points, stockée elle aussi sous forme d'un vecteur (inclu par exemple dans une classe *Polygone*).

On cherche à trouver quels polygones sont identiques aux transformations suivantes près :

— translations ;

— rotations.

On souhaite avoir en sortie un vecteur de vecteurs de polygones.

Une telle opération permet par exemple de réaliser des coloriages intéressants sur des mosaïques arabes, en coloriant les polygones identiques d'une même couleur.

On suppose disposer d'un opérateur de comparaison (coûteux) sur la classe polygone répondant si deux polygones sont identiques. Il n'est donc pas nécessaire de le re-programmer.

1. Proposez un algorithme simple ; quel est son coût au pire cas ?
2. En remarquant que deux polygones identiques contiennent forcément le même nombre de sommets, proposez un nouvel algorithme.
3. Comment pourrait-on continuer à améliorer les choses ?

2 Élimination des recouvrements

On dispose en entrée d'un vecteur de segments. Parmi eux, certains peuvent se recouvrir. On cherche à éliminer toutes les parties communes et à obtenir un autre vecteur de segments en sortie.

Par exemple, les segments (0,0) – (3,3) et (1,1) – (4,4) se recouvrent sur (1,1) – (3,3). En enlevant la partie qui leur est commune on obtient ainsi les segments (0,0) – (1,1) et (3,3) – (4,4).

On cherche comme dans l'exercice précédent à éviter un coût quadratique.

1. En remarquant que deux segments se recouvrent uniquement s'ils sont sur la même droite, proposez une clef pour une table de hachage.

   On regarde maintenant **uniquement les segments alignés sur une même droite**.

   On considère que chaque segment démarre sur son point minimal et s'arrête sur son point maximal. Étant donné un point p et l'ensemble des segments S, on définit :

$$c(p) = |\{s \in S | \text{s démarre en } p\}| - |\{s \in S | \text{s s'arrête en } p\}|$$

2. Calculer et stocker c.
3. Conclure pour chaque droite grâce à un balayage (sur les points). On utilisera un compteur comptant pour chaque point le nombre de segments sur lequel celui-ci se trouve.
4. Quel est le coût au pire cas de votre algorithme ?