Описание алгоритма за $O(n^4)$

Алгоритм принимает множество S из n непересекающихся отрезков. P - множесто концов отрезков S, является МТОП.

1. Строится множество L, прямых, определяющихся любой парой $p_i, p_j \in P$.

   Данная операция требует $O(n^2)$ времени, $|L| = 2n^2 - n$.

   Для дальнейшей обработки множество точек P сохраняется. Для каждого отрезка запоминаются индексы его концов. Контейнер занимает $2n$ памяти.

2. Строится упорядочение прямых $A(L)$ с помощью инкрементального алгоритма. Его сложность для m прямых есть $O(m^2)$. Соответственно в данном случае потребуется $O(n^4)$.

   Результатом работы данного алгоритма является РСДС, занимающий линейную память от числа ребер, вершин и граней. Так как это число для $A(L)$ квадратично зависит от количества прямых в L, то расход по памяти на данном этапе алгоритма достигает $O(n^4)$.

   Для каждого ребра сохраняется информация об отрезках из S, на точках которых была построена прямая, частью которой является данное ребро.

   Также каждое ребро помечается булевым флагом, определяющим лежит оно между задающих его точек или с одной из сторон.

3. Произвольно выбирается грань f РСДС. Производится упорядочивание P по возрастанию полярного угла относительно любой внутренней точки грани, обозначим точку q. Упорядоченное множество P обозначим через P_u.

   Введем понятие «правильность»: f - грань РСДС, q - внутренняя точка f. Назовем отрезок правильным относительно f, если его точки являются соседними в P_u, построенном относительно q, а тройка $p_i q p_j$ образовывает левый поворот, в преположении, что p_i находится в P_u раньше. Если в P_u всего один отрезок, то он правильный.

   После сортировки все отрезки проверяются на правильность. Обозначим через R количество правильных отрезков.

   P_u и R вместе образуют статус.

   Сортировка занимает $O(nlog(n))$, проверка на правильность - $O(n)$.

4. Производится обход РСДС (например в ширину), начиная с грани f. На каждом шаге, зная «перешагиваемое» ребро, имеем информацию об индексах i и j, порождающих его точек, информацию о положении ребра относительно этих точек. Также имеем информацию об отрезке/отрезках, частью которого/которых являются точки p_i, p_j.

   Этой информации и статуса достаточно, чтобы внести изменение в статус:

   (a) Подсчитываем количество правильных отрезков, содержащих p_i, p_j, до «перешагивания». Обозначим через r_b.

   (b) Вносим изменение в P_u (меняем или не меняем точки p_i, p_j местами).

   (c) Подсчитываем количество правильных отрезков, содержащих p_i, p_j, после «перешагивания». Обозначим через r_a.

(d) Перевычислим $R : R = R - r_b + r_a$.

На внесение изменений в порядок точек в контейнере и количество правильных отрезков требуется константное время. Таким образом, обработка каждой грани требует $O(1)$ времени и памяти.

5. Обход продолжается до тех пор пока не найдется грань, в которой $R = n$, или пока не останется непосещенных граней. В первом случае ответом алгоритма - «да» с предоставлением любой точки внутри найденной грани, во втором случае ответ - «нет».

   Обход требует $O(N^4)$ времени.

Источники:

1. de Berg, Mark; Cheong, Otfried; van Kreveld, Marc; Overmars, Mark (2008). Computational Geometry, Algorithms and Applications (3rd ed.). Springer. pp. 172–177. ISBN 978-3-540-77973-5.