→ 1.5.11.4. Algorithmus:

1). Berechne Streifen von links nach rechts sortiert in Zeit $O(n)$, wobei $n = m + \ell$.
Gesamtzahl der Ecken, durch Mischen der Eckenfolgen von P und Q.
(siehe D&C für CH.)

2). Berechne für jeden Streifen R Trapezoide $P \cap R$ und $Q \cap R$. Das geht auch in $O(n)$, da wir Eckenfolgen (siehe 1)) von links nach rechts sortiert haben.

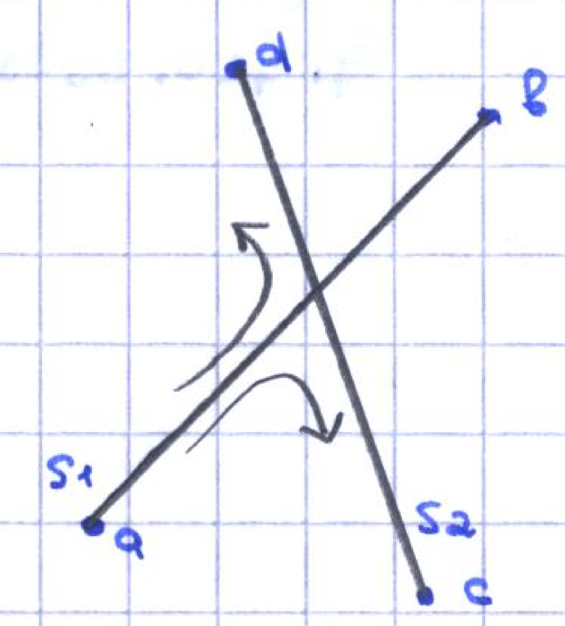
3) Berechne für jeden Streifen R $P \cap Q \cap R := (P \cap R) \cap (Q \cap R)$
Zeit $O(1)$ pro Streifen.

4). Zusammen kleben der Teile aus 3)
Insbesondere Eliminierung von Ecken, die nicht zur Ausgabe gehören.
Laufzeit: $O(n)$ (Durchlaufen der Senkrechten von links nach rechts).
→ Eckenfolge von $P \cap Q$ gg. Uhrzeigersinn.

→ 1.5.11.5. Implementierungsdetails:

Teilprobleme:

- Test ob Segment s_1 ein anderes schneidet.
  $s_1 = \overline{a,b}$, $s_2 = \overline{c,d}$
  Gerade durch (a,b) schneidet (c,d) (1) und Gerade durch (c,d) schneidet (a,b) (2)
  $\Leftrightarrow s_1 \cap s_2 \neq \emptyset$
  (1) orientation$(a,b,c) \neq$ orientation(a,b,d) oder beide 0
  (2) orientation$(c,d,a) \neq$ orientation(c,d,b) oder beide 0.
- falls $s_1 \cap s_2 = \emptyset \Rightarrow$ Bestimmung der relativen Lage von s_1 und s_2 durch weitere Orientierungstests.
- falls $s_1 \cap s_2 \neq \emptyset \Rightarrow$ Bestimmung der Schnittpkte (anal. Geometrie)
- Sonderfälle: $s_1 = s_2$.

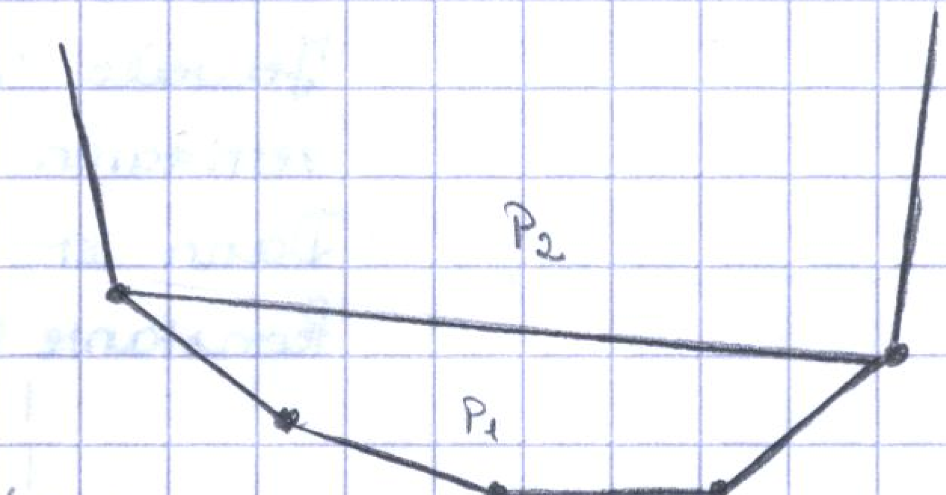


1.5.12. Satz: Der Schnitt $P \cap Q$ von zwei konvexen Polygonen P und Q kann in Zeit $O(n)$ berechnet werden, wobei n = Summe der Ecken von P und Q.
Mischschritt

1.5.13. Fragen:

1) Geht das schneller, wenn alle Polygone im Voraus geg. sind, für die man evtl. Schnittoperationen ausführt.

2). Existiert Algorithmus, der output-sensitiv ist?
Dh. Laufzeit hängt von Größe der Ausgabe ab.
→ siehe nächster Abschnitt über konvexe Polygone.



1.5.14. Divide & Conquer - Alg. für Schnitt von Halbebenen.

Sei S Menge von Halbebenen.

```
INTERSECT(S)
if |S| = 1 then
    return S;
else
    teile S in zwei gleich große Teile S1 und S2
    P ← INTERSECT(S1);
    Q ← INTERSECT(S2);
    return P ∩ Q   // wie oben beschrieben, möglicherweise offen → Übung.
```

Ein unbeschr. Polygon besteht aus einem beschr. Pol. P_1 und unbeschr. Pol. $P_2 \Rightarrow P = P_1 \cup P_2$
Berechne $P_1 \cap Q_1$ mit D&C und Mischritt von oben $\Rightarrow O(n \log n)$
Berechne $P_2 \cap Q_1$ und $P_1 \cap Q_2$ als Schnitt von Polygon mit drei Geraden in Zeit $3 \cdot O(n \log n) = O(n \log n)$. Anschließend berechne $P_2 \cap Q_2$ als Schnitt von 6 Geraden in $O(9) = O(1)$
$\Rightarrow$ Gesamtlaufzeit: $O(n \log n)$.

1.5.15 Laufzeit: $T(n) = \begin{cases} c_0, & |S| = 1 \\ c_1 n + 2 \cdot T(n/2), & |S| > 1 \end{cases}$

$\Rightarrow T(n) = O(n \log n)$
Alternative zu Dualitätsalgorithmus

1.5.16. Frage: Welcher Algorithmus ist in welchen Fällen schneller? Wahrscheinlich Divide & Conquer