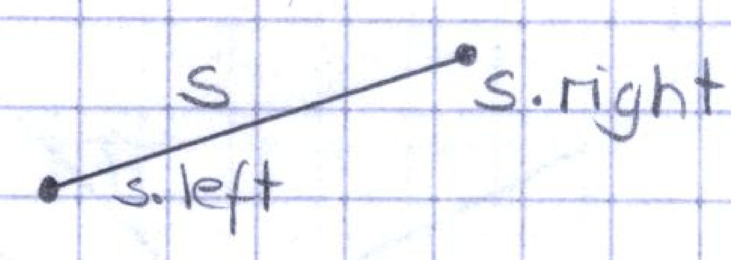
- **Algorithmus:**

SWEEP(S)

- X-Struktur $X \leftarrow \emptyset$ — $O(1)$
- Y-Struktur $Y \leftarrow \emptyset$ — $O(1)$
- double xpos $\leftarrow -\infty$ // xpos ist die Position der SL — $O(1)$
- forall $s \in S$ do
  - X.insert (s.left)
  - X.insert (s.right)

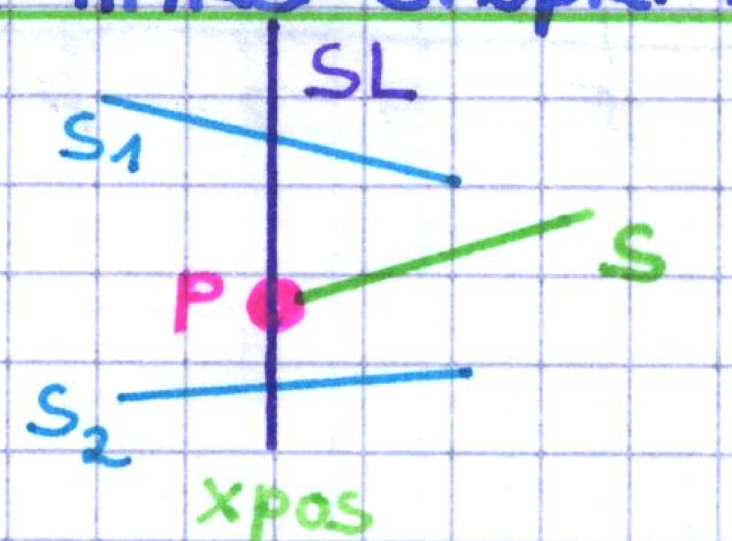
  od — $O(n \log n)$



- while ($X \neq \emptyset$) {
  - $p \leftarrow X$.findmin()

    xpos $\leftarrow$ p.xcoord() // schiebt SL zum Pkt p

    Fallunterscheidungen gemäß der verschiedenen Events:
  - switch (p) {
  - case linker Endpkt von s: { 6



Y.insert (s)

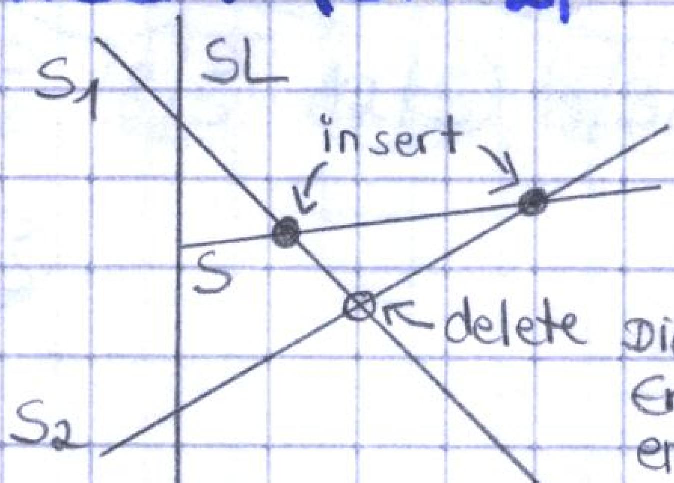
$s_1 \leftarrow Y$.succ (s) — s_1 exist nicht immer

$s_2 \leftarrow Y$.pred (s) — s_2 —"—

X.delete ($s_1 \cap s_2$) — $s_1 \cap s_2$ —"—

X.insert ($s_1 \cap s$) — $s_1 \cap s$ —"—

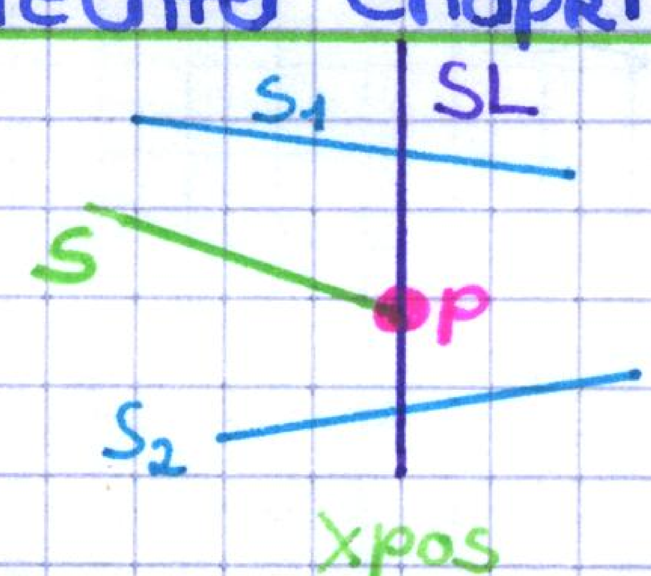
X.insert ($s \cap s_2$) — $s \cap s_2$ —"—



Dieser Schnittpkt wird jetzt gelöscht um Platz zu sparen. Er wird aber später (wenn die SL nahe genug ist) wieder entdeckt.

}

- case rechter Endpkt von s: { 4



Geht nicht !! Weil man in der Y-Strukt. nur Vorgänger & Nachf. kennt. Und wenn s weg ist, weiß man nicht mehr dass es zw. s_1 und s_2 war, dass man diese beiden jetzt neu "verbinden" muss.

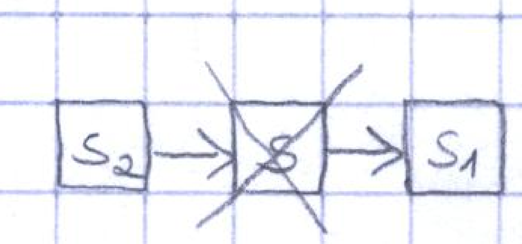
Y.delete(s)
$s_1 \leftarrow Y$.succ(s_2)
$s_2 \leftarrow Y$.pred(s_1)

$s_1 \leftarrow Y$.succ (s)

$s_2 \leftarrow Y$.pred (s)

Y.delete (s)

X.insert ($s_1 \cap s_2$) — $s_1 \cap s_2$ liegt recht von SL !!



}

- case Schnittpkt von s' und s'': { 8