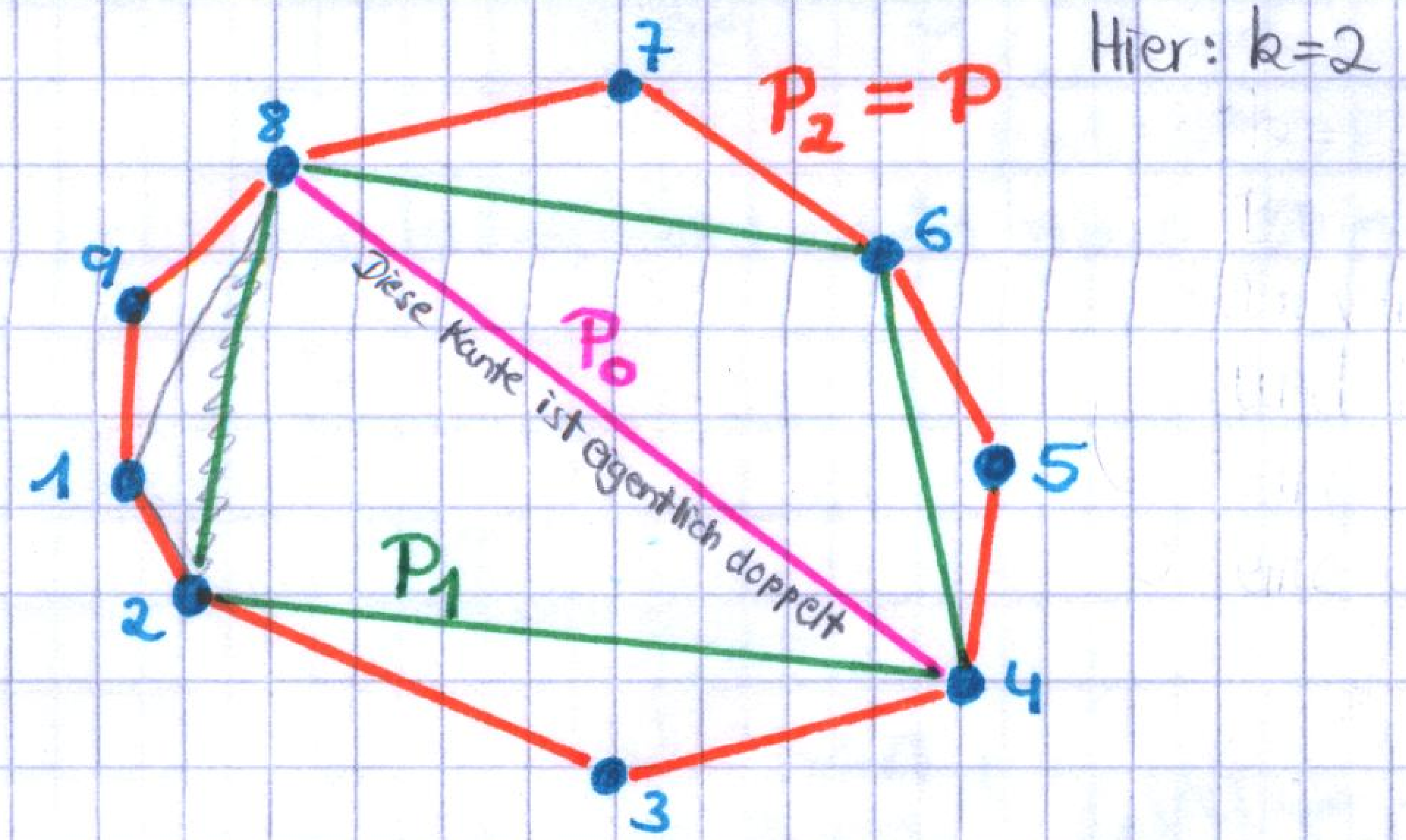
2. Konvexe Polygone

2.1 Hierarchische Darstellung

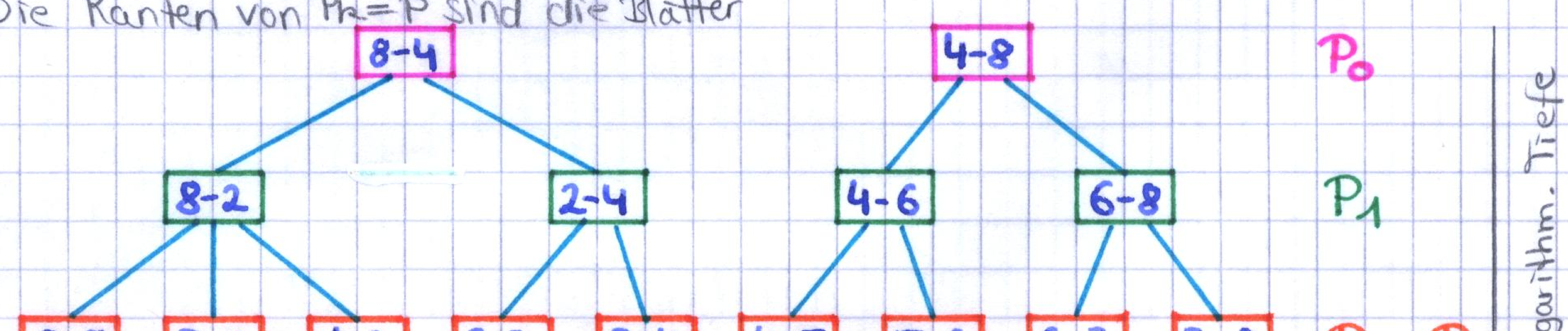
- **Idee:**
  Investiere Zeit und Platz $O(n)$ in Aufbau einer Struktur um nachfolgende Operationen billiger zu machen.
  Lohnt sich nur, wenn mit demselben Polygon viele Operationen ausgeführt werden.

- **Definition:**
  Sei P konvexes Polygon mit n Ecken.
  Eine Folge $P_0, P_1, \dots, P_k$ konvexer Polygone heißt hierarchische Darstellung von P, wenn gilt:
  - P_0 hat ≤ 4 Ecken
  - $P_k = P$
  - P_{i-1} entsteht aus P_i $(1 \leq i \leq k)$ durch Entfernen einiger Ecken:
    - von je 3 aufeinanderfolgenden Ecken von P_i wird mind eine entfernt
    - es werden nie 4 aufeinanderfolgende Ecken entfernt.



- Diese Def sichert **2 Eigenschaften:**
  (∗∗) • Beim Übergang von $P_{i+1} \rightarrow P_i$ $(0 \leq i \leq k-1)$ verlieren wir einen konstanten Bruchteil der Ecken (mind $1/3$, höchstens [redacted] $3/4$)
  - P_{i+1} ist ähnlich zu P_i, da nie mehr als 3 aufeinanderfolgende Ecken entfernt werden

- **Balancierter Baum der hierarchischen Darstellung:**
  Die Kanten von P_i sind die Blätter und Knoten im Baum $(0 \leq i \leq k)$
  Die Kanten von $P_k = P$ sind die Blätter



Kein binärer Baum !

- **Lemma:**
  - Ein balancierter Baum einer hierarchischen Darstellung eines konv. Polygons mit n Ecken kann in Zeit $O(n)$ berechnet werden. ← mit Mastermethode $(\log_4 4 = 1)$
  - Benötigt Platz $O(n)$ $\#\text{Knoten} \leq n + \frac{n}{3} + \frac{n}{9} + \frac{n}{27} + \dots = \sum_{v=0}^{n} \left(\frac{1}{3}\right)^v = \frac{(1/3)^{n+1} - 1}{1/3 - 1} = -3/2\,(1/3)^{n+1} + 3/2 = 3/2 - 1/2\,(1/3)^n = 1/2\,\underbrace{(3 - (1/3)^n)}_{\leq 3} \leq 3/2 \Rightarrow O(1 \cdot n) = O(n)$
  - Die Tiefe dieser Darstellung ist $O(\log_2 n)$ max 4 Verzweigungen

Beweis: Folgt aus (∗∗)

$\log_2 n = \frac{\log_{10} n}{\log_{10} 2}$

$\log_4 n = \frac{\log_{10} n}{\log_{10} 4} = \frac{\log_{10} 2 \cdot \log_2 n}{\log_{10} 4} = \frac{\log_{10} 2}{\log_{10} 4} \cdot \log_2 n = \underbrace{\log_4 2}_{\text{Konst.}} \cdot \log_2 n \Rightarrow \log_4 n = O(\log_2 n)$