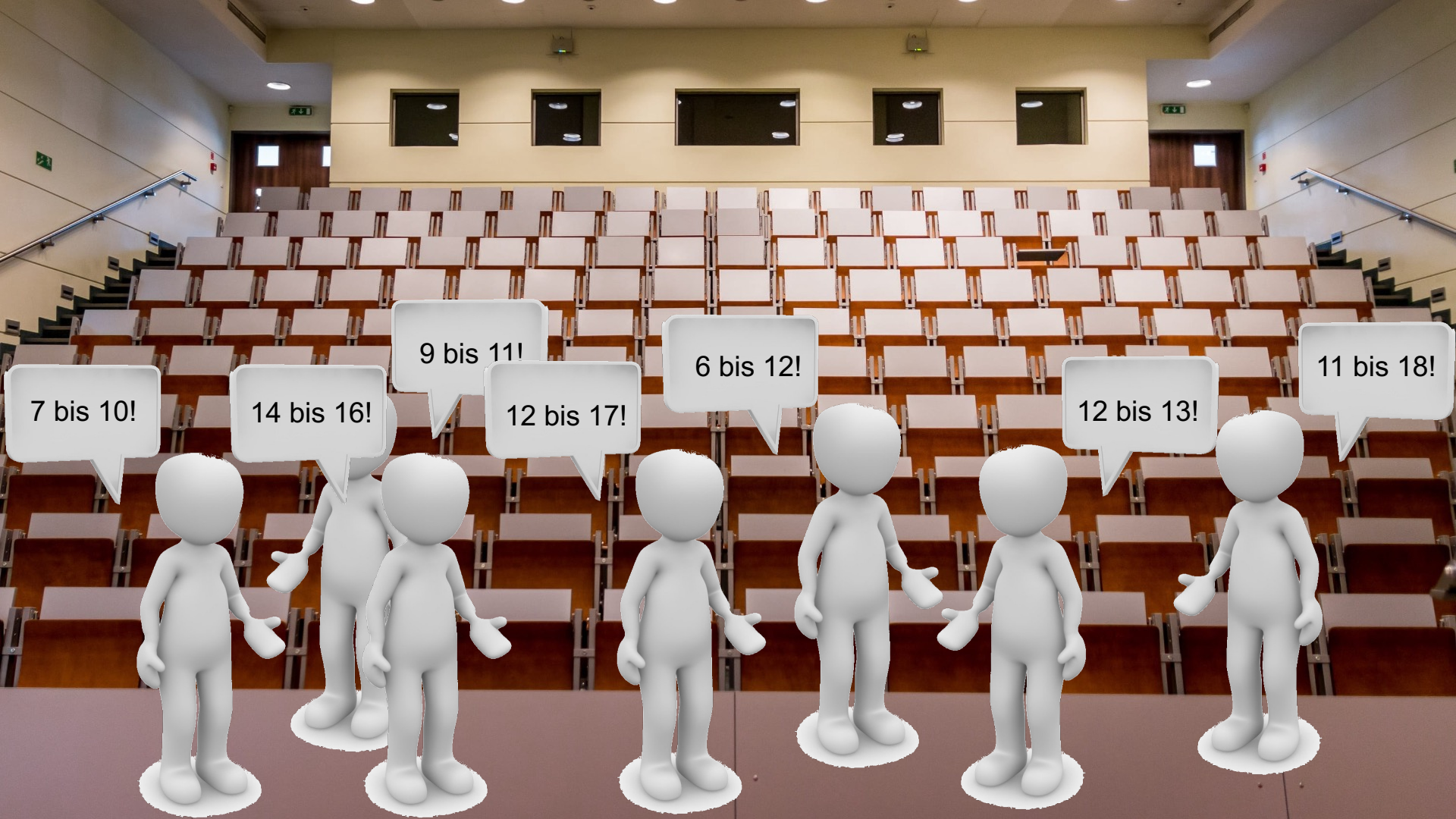
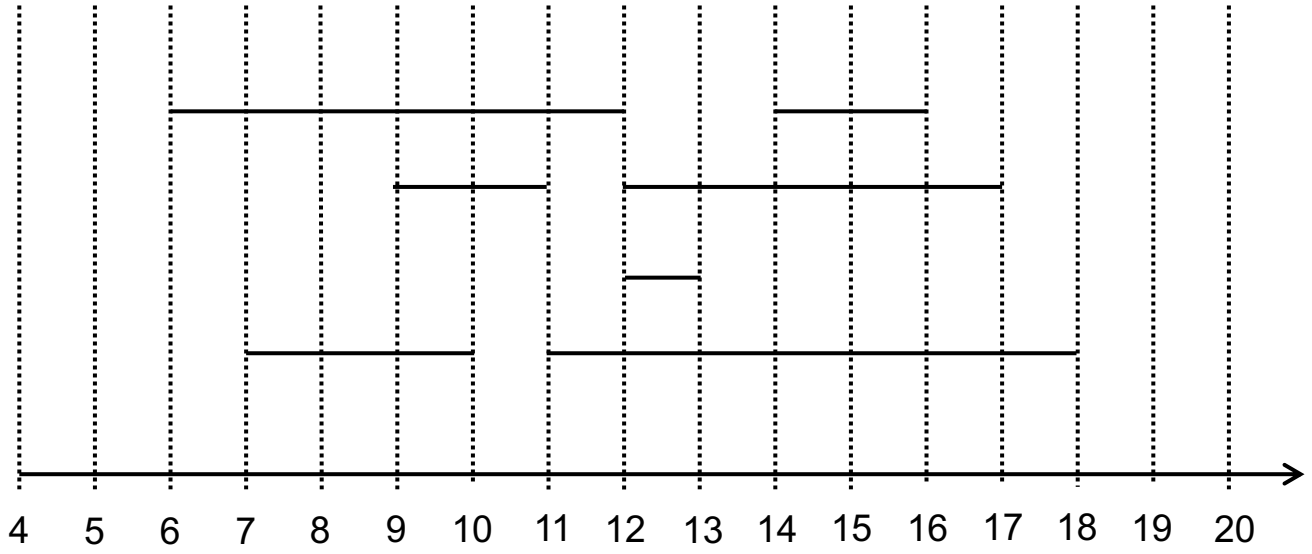
Technische Universität Braunschweig

Algorithmen und Datenstrukturen 2 – Übung #1

Phillip Keldenich
26.04.2023


7 bis 10!
14 bis 16!
9 bis 11!
12 bis 17!
6 bis 12!
12 bis 13!
11 bis 18!

Hörsaal-Belegung



Hörsaal-Belegung – Das Problem

Gegeben

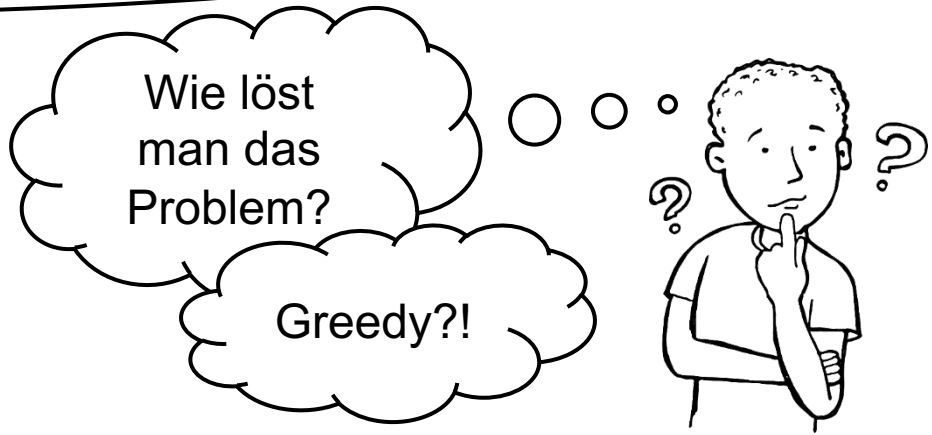
Menge von Intervallen $\mathcal{I} = \{I_1 = [s_1, e_1), \dots, I_n = [s_n, e_n)\}$

Gesucht

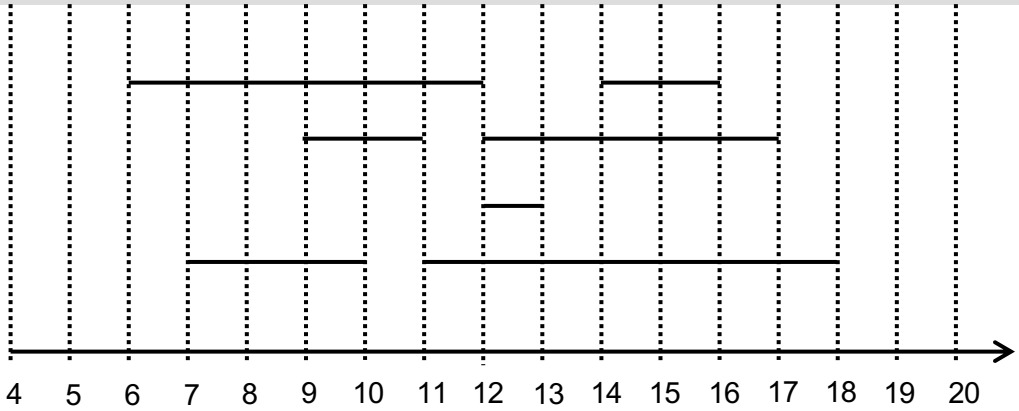
Teilmenge $\mathcal{I}' \subseteq \mathcal{I}$ mit den folgenden Eigenschaften

1. $\forall I_i, I_j \in \mathcal{I}' : I_i \cap I_j = \emptyset$
2. $\mathcal{I}'$ ist größtmöglich

Die ausgewählten Intervalle müssen disjunkt sein.



Hörsaal-Belegung – Strategien



Welche Greedyfunktion?

Kürzeste zuerst?

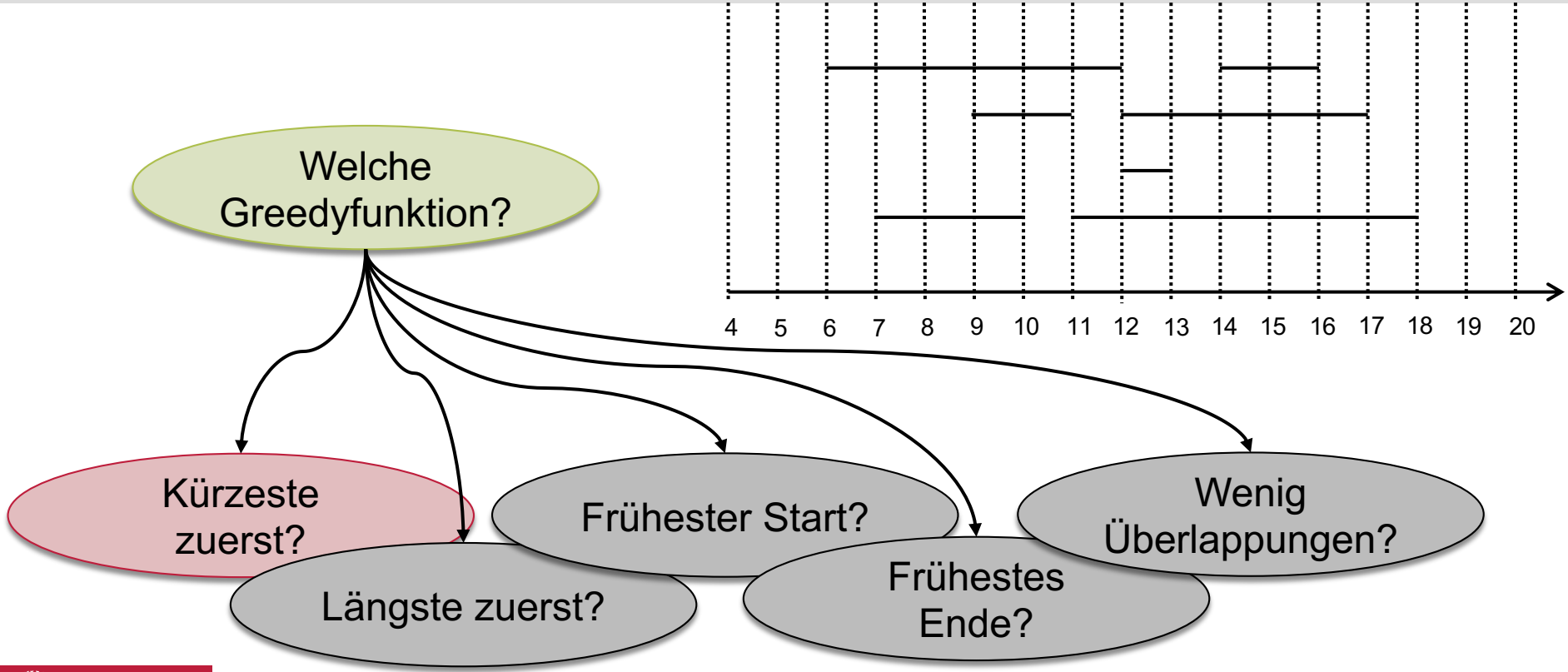
Längste zuerst?

Frühester Start?

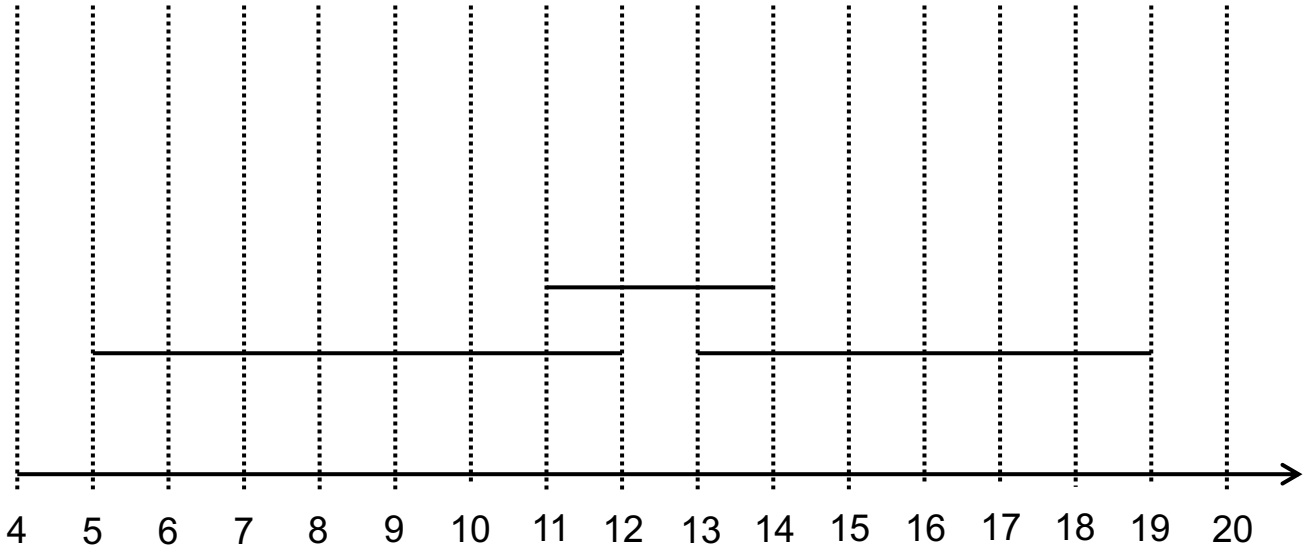
Frühestes Ende?

Wenig Überlappungen?

Hörsaal-Belegung – Strategien



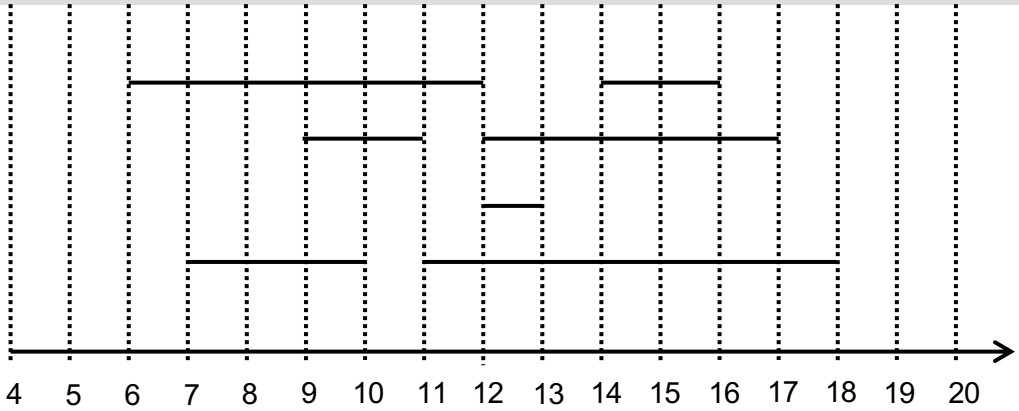
Hörsaal-Belegung – Kürzeste zuerst



ALG = 1

OPT = 2

Hörsaal-Belegung – Strategien



Welche Greedyfunktion?

Kürzeste zuerst?

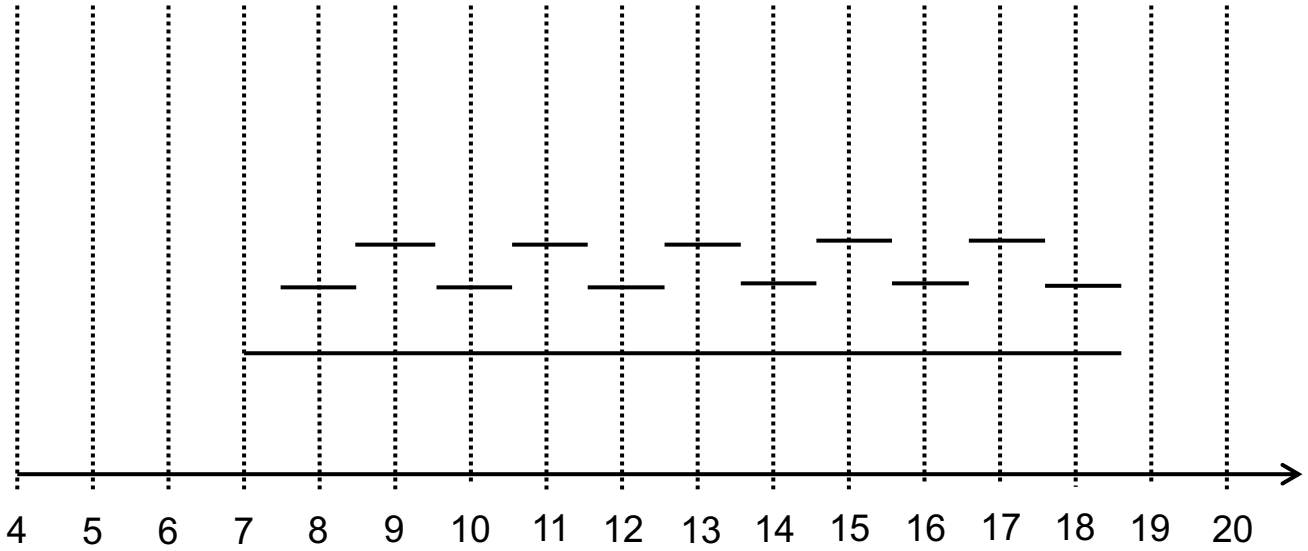
Längste zuerst?

Frühester Start?

Frühestes Ende?

Wenig Überlappungen?

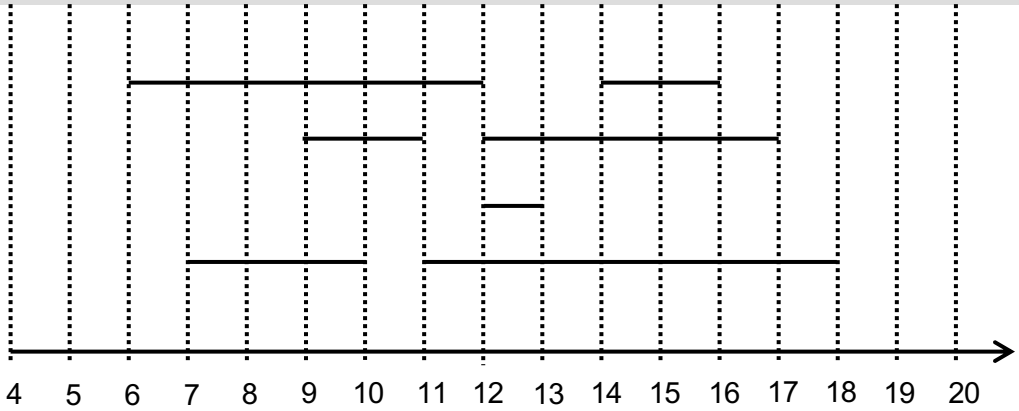
Hörsaal-Belegung – Frühester Start/Längstes Intervall



ALG = 1

OPT = 11

Hörsaal-Belegung – Strategien



Welche Greedyfunktion?

Kürzeste zuerst?

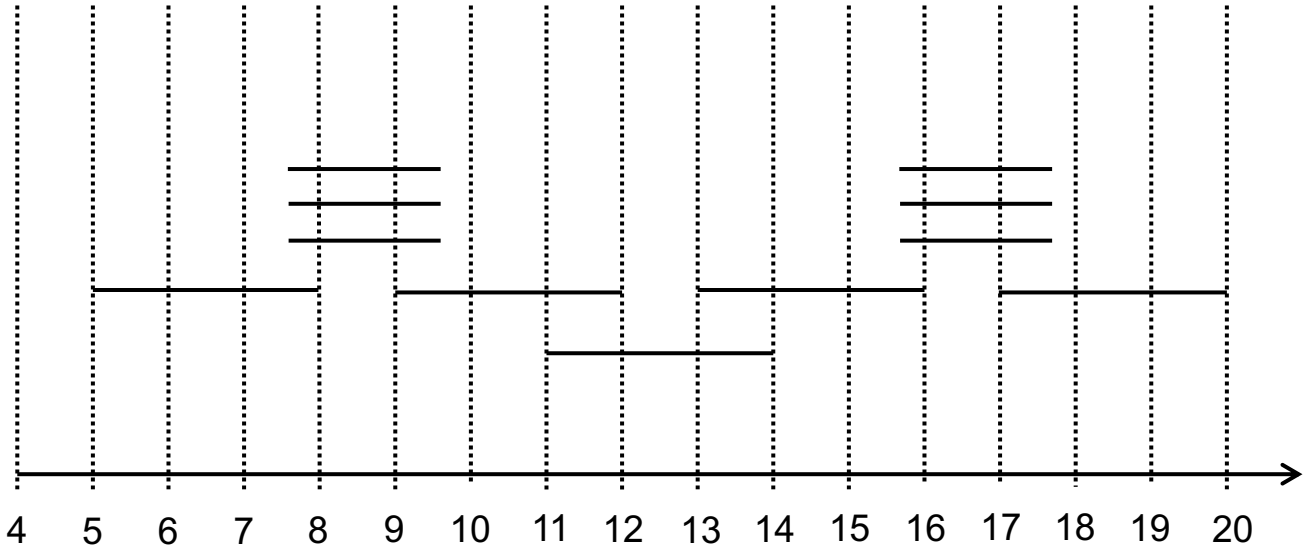
Längste zuerst?

Frühester Start?

Frühestes Ende?

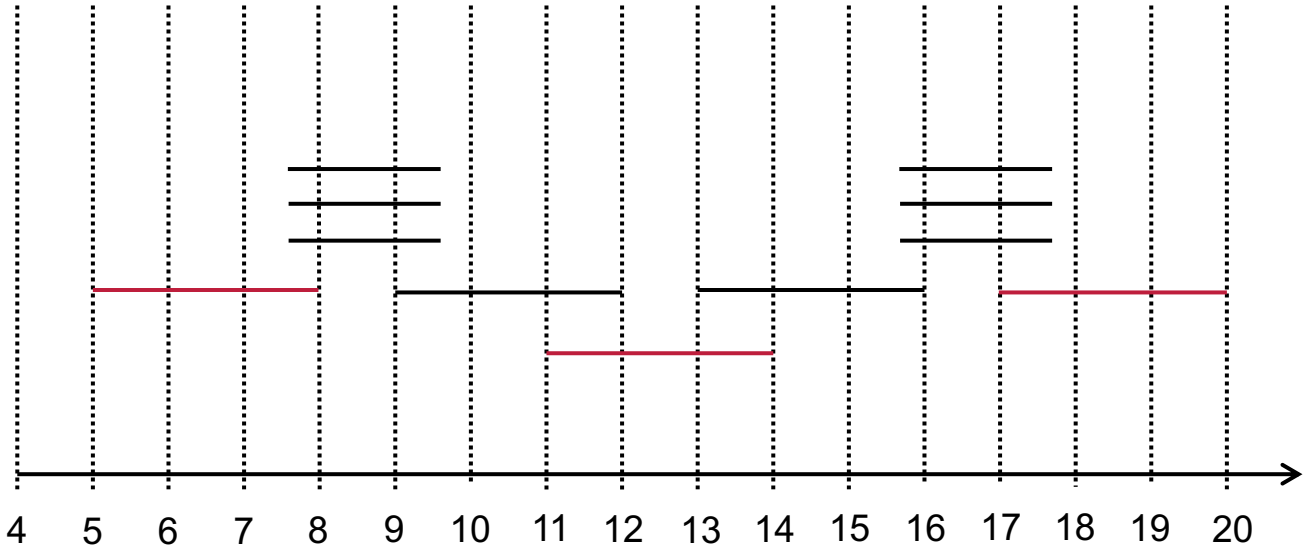
Wenig Überlappungen?

Hörsaal-Belegung – Wenig Überlappungen



ALG = 3

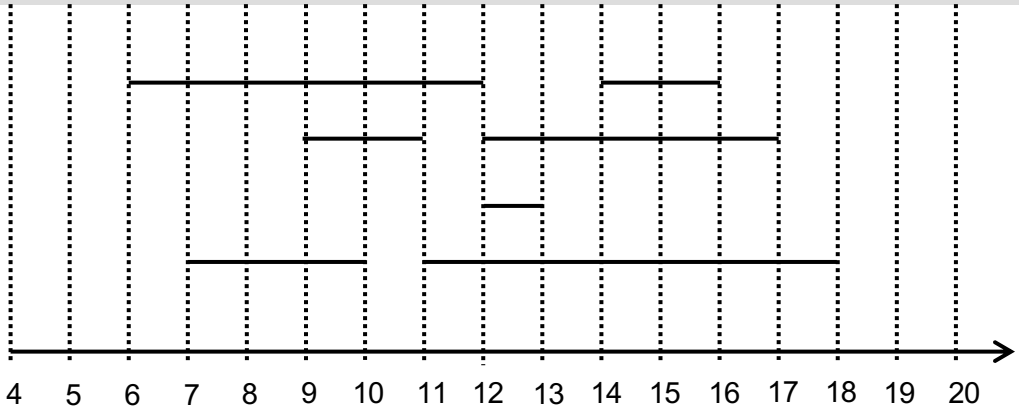
Hörsaal-Belegung – Wenig Überlappungen



ALG = 3

OPT = 4

Hörsaal-Belegung – Strategien



Welche Greedyfunktion?

Kürzeste zuerst?

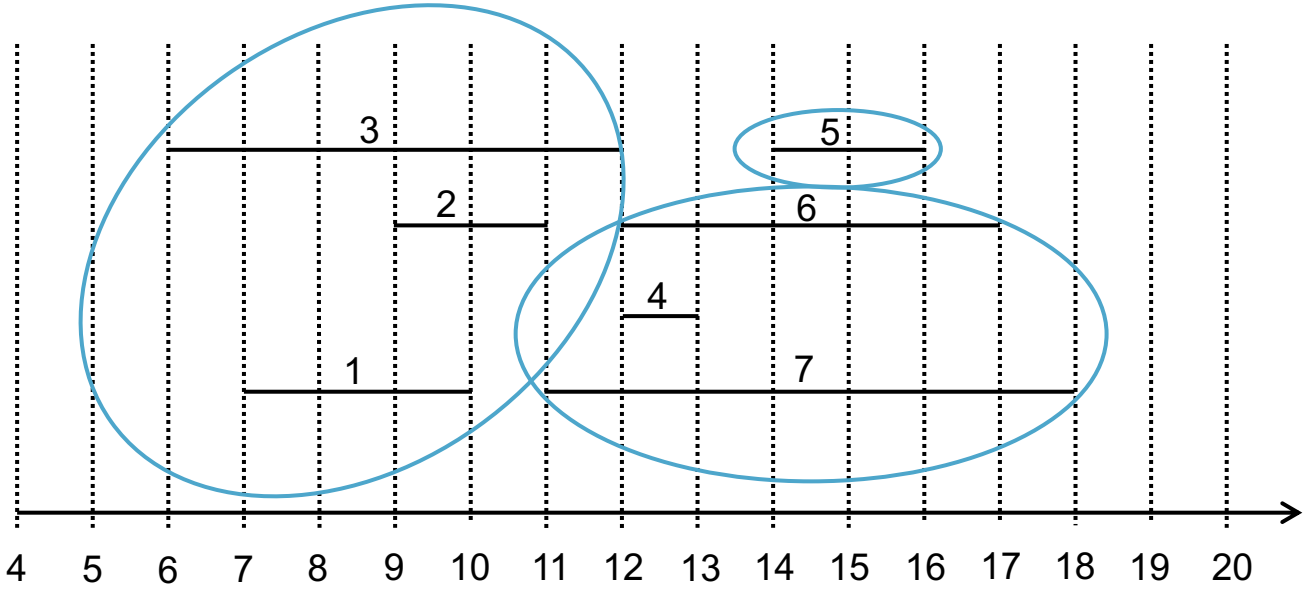
Längste zuerst?

Frühester Start?

Frühestes Ende?

Wenig Überlappungen?

Hörsaal-Belegung – Frühestes Ende



ALG = 3

OPT = 3?

Hörsaal-Belegung – Frühestes Ende

Greedy-Algorithmus: Frühestes Ende

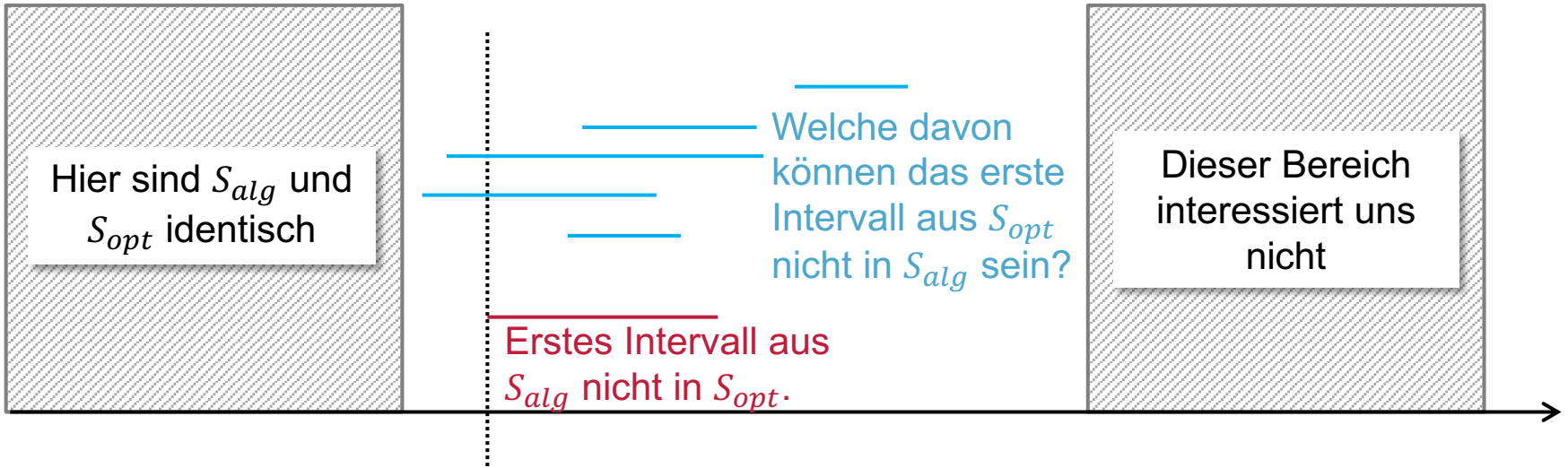
1. Sortiere $\{1, \dots, n\}$ nach e_i aufsteigend; erhalte Permutation $\pi(1), \dots, \pi(n)$
2. $S \coloneqq \{\pi(1)\}$
3. $e \coloneqq e_{\pi(1)}$
4. **for** $k = 2$ **to** n **do**
5. **if** $(s_{\pi(k)} \geq e)$ **then**
6. $S \coloneqq S \cup \{\pi(k)\}$
7. $e \coloneqq e_{\pi(k)}$
8. **return** S

Theorem
Der Algorithmus löst das Hörsaal-Problem optimal in Zeit $O(n \log n)$

Beweis

Annahme: Algorithmus 1 sei nicht optimal.

Sei S_{alg} die Lösung von Algorithmus 1 und S_{opt} eine optimale Lösung.



Beweis

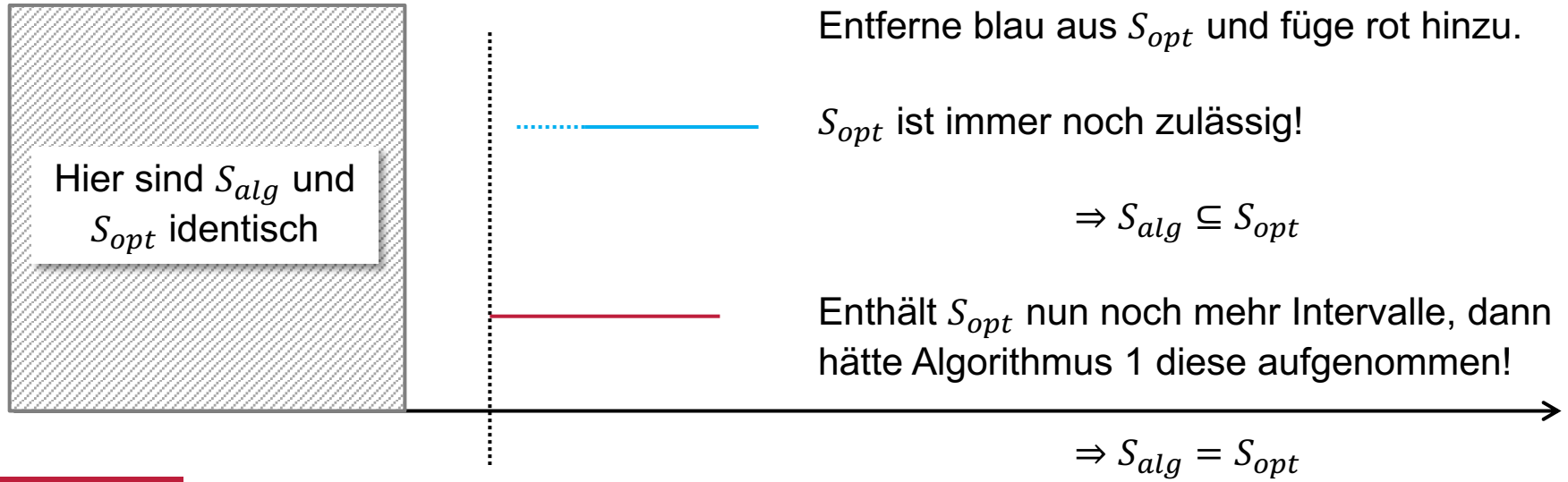
Annahme: Algorithmus 1 sei nicht optimal.

Sei S_{alg} die Lösung von Algorithmus 1 und S_{opt} eine optimale Lösung.

Beweis

Annahme: Algorithmus 1 sei nicht optimal.

Sei S_{alg} die Lösung von Algorithmus 1 und S_{opt} eine optimale Lösung.



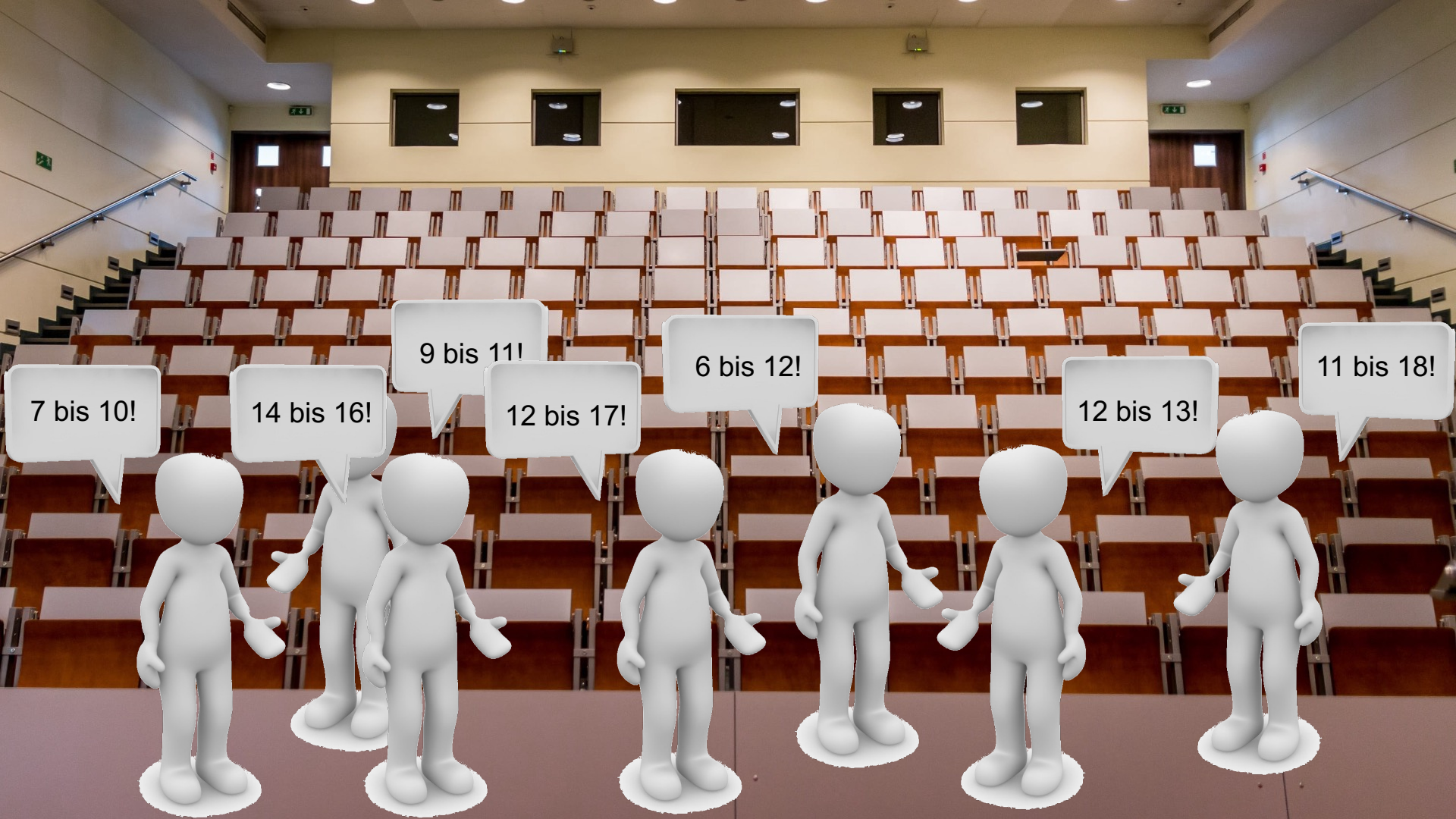
Entferne blau aus S_{opt} und füge rot hinzu.

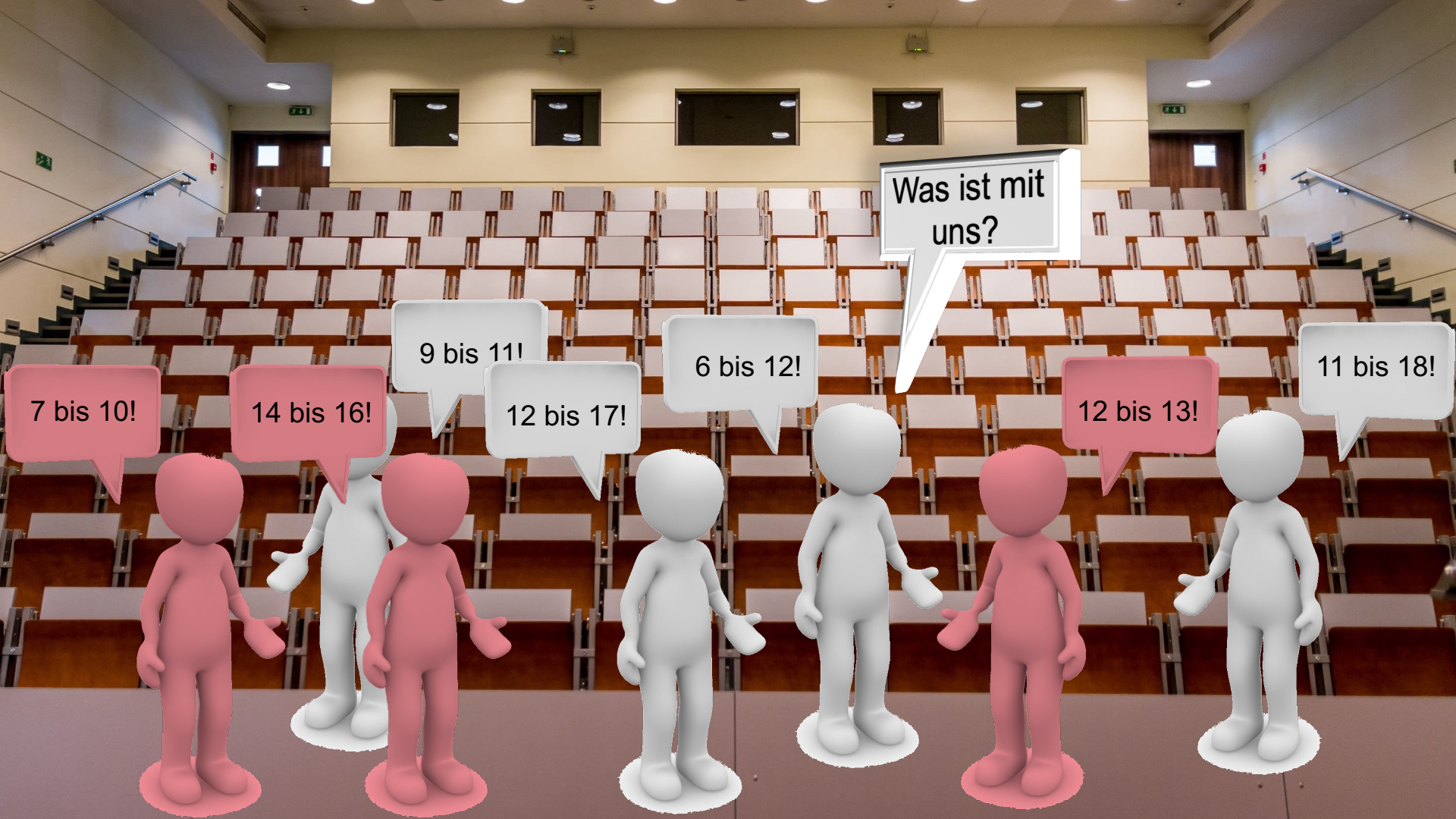
S_{opt} ist immer noch zulässig!

$$\Rightarrow S_{alg} \subseteq S_{opt}$$

Enthält S_{opt} nun noch mehr Intervalle, dann hätte Algorithmus 1 diese aufgenommen!

$$\Rightarrow S_{alg} = S_{opt}$$


7 bis 10!
14 bis 16!
9 bis 11!
12 bis 17!
6 bis 12!
12 bis 13!
11 bis 18!


7 bis 10!
14 bis 16!
9 bis 11!
12 bis 17!
6 bis 12!
Was ist mit uns?
12 bis 13!
11 bis 18!

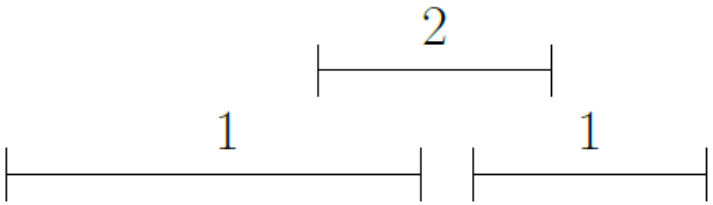
Hörsaal-Belegung – Eine Variante

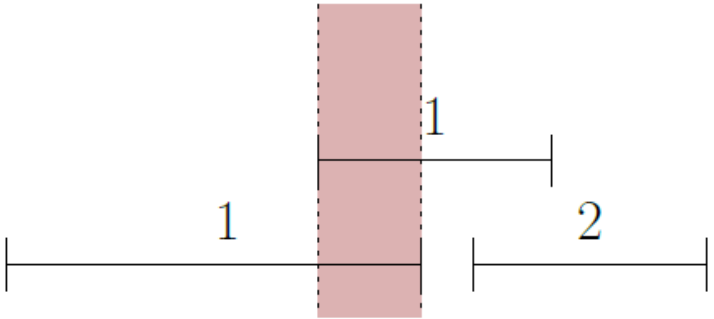
Gegeben

Menge von Intervallen $\mathcal{I} = \{I_1 = [s_1, e_1), \dots, I_n = [s_n, e_n)\}$

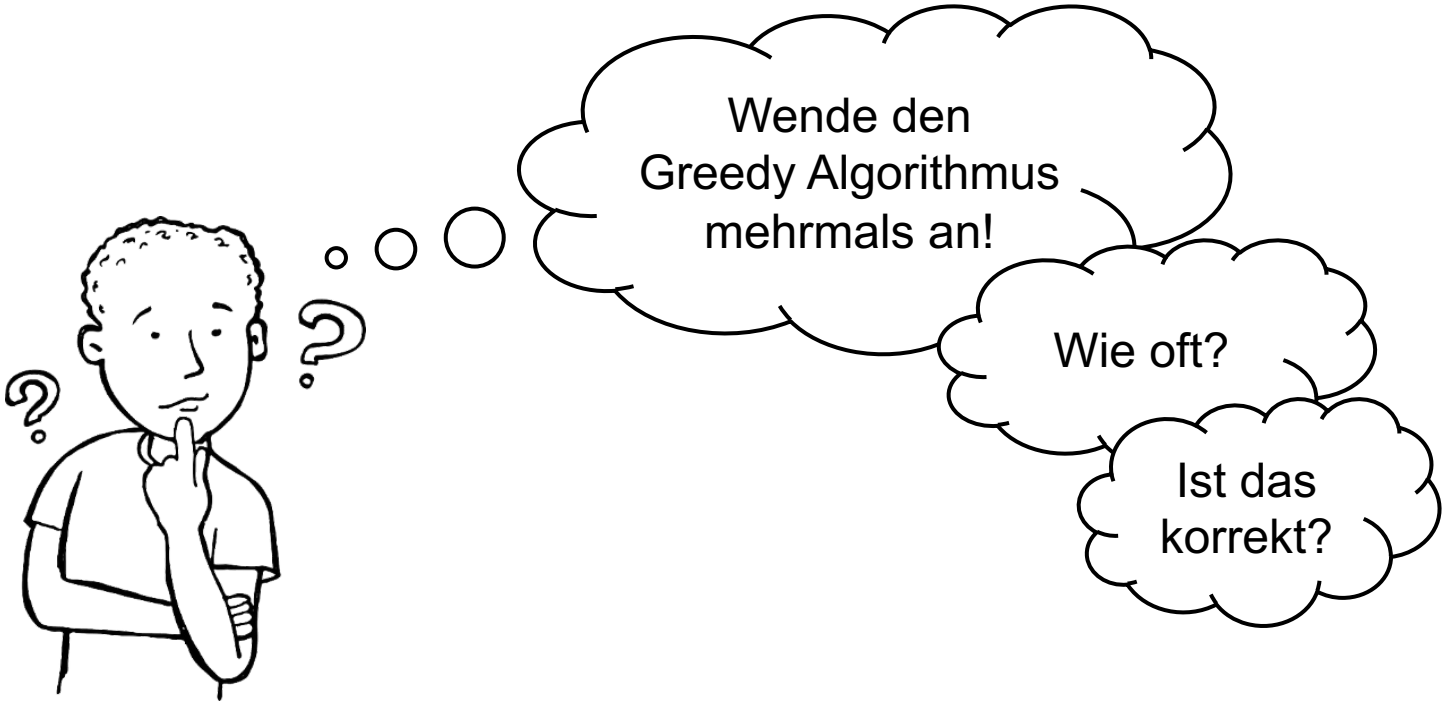
Gesucht

Die kleinste Zahl $k \in \mathbb{N}$, sodass eine Funktion $f: I \to \{1, \dots, k\}$ existiert und für je zwei Intervalle I und I' gilt $I \cap I' \neq \emptyset \Rightarrow f(I) \neq f(I')$

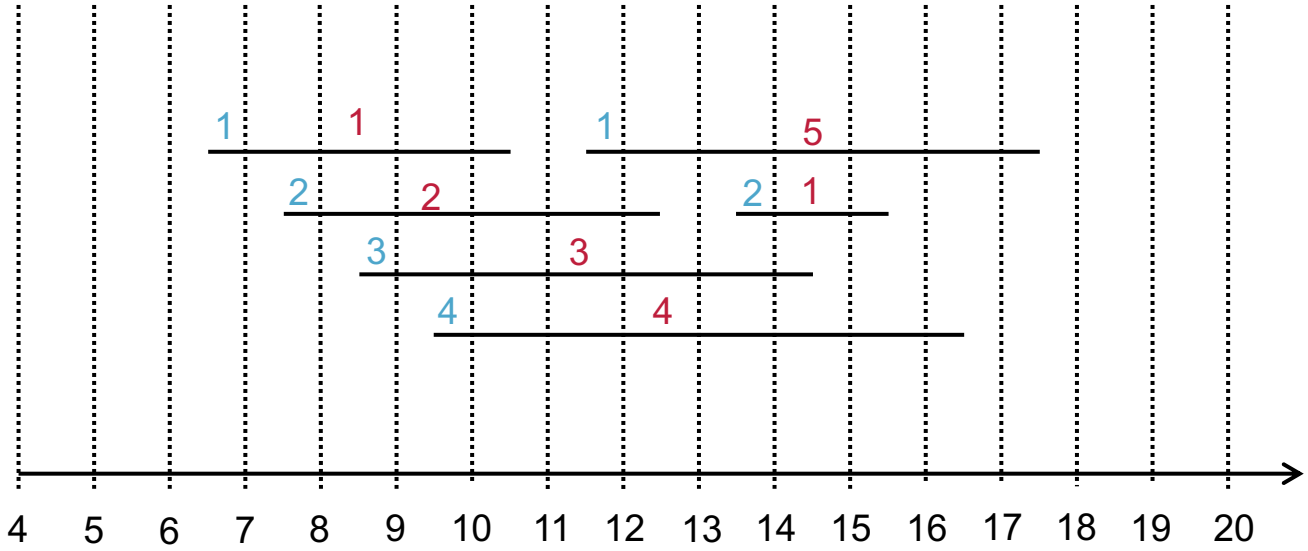




Hörsaal-Belegung – Eine Variante



Hörsaal-Belegung – Eine Variante

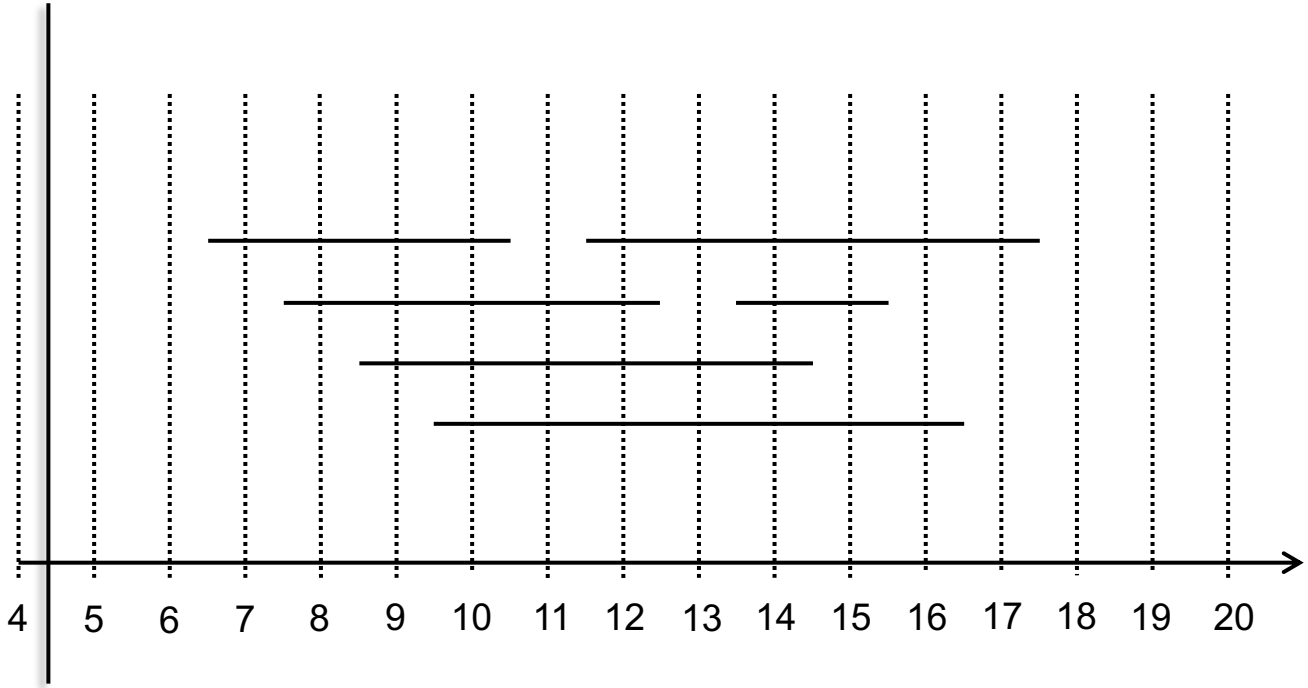


ALG = 5

OPT = 4

Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

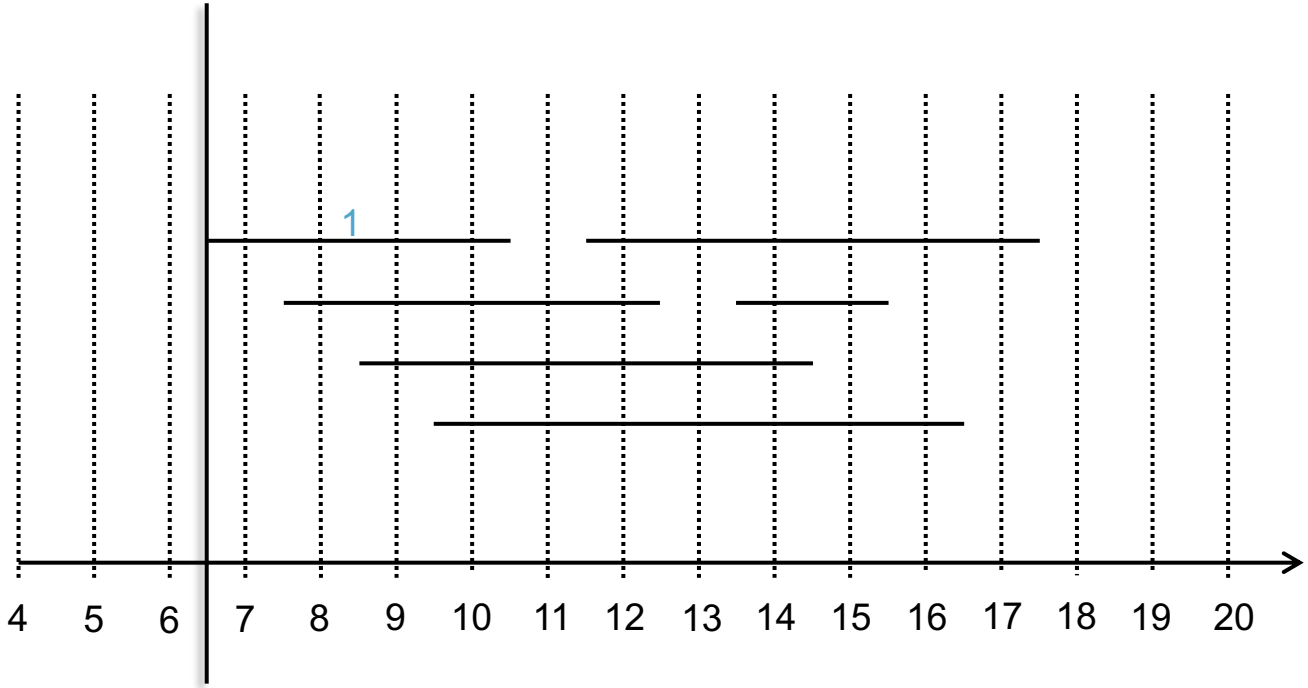
Hörsaal-Belegung – Eine Variante



Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bewege eine Linie von links nach rechts

Hörsaal-Belegung – Eine Variante

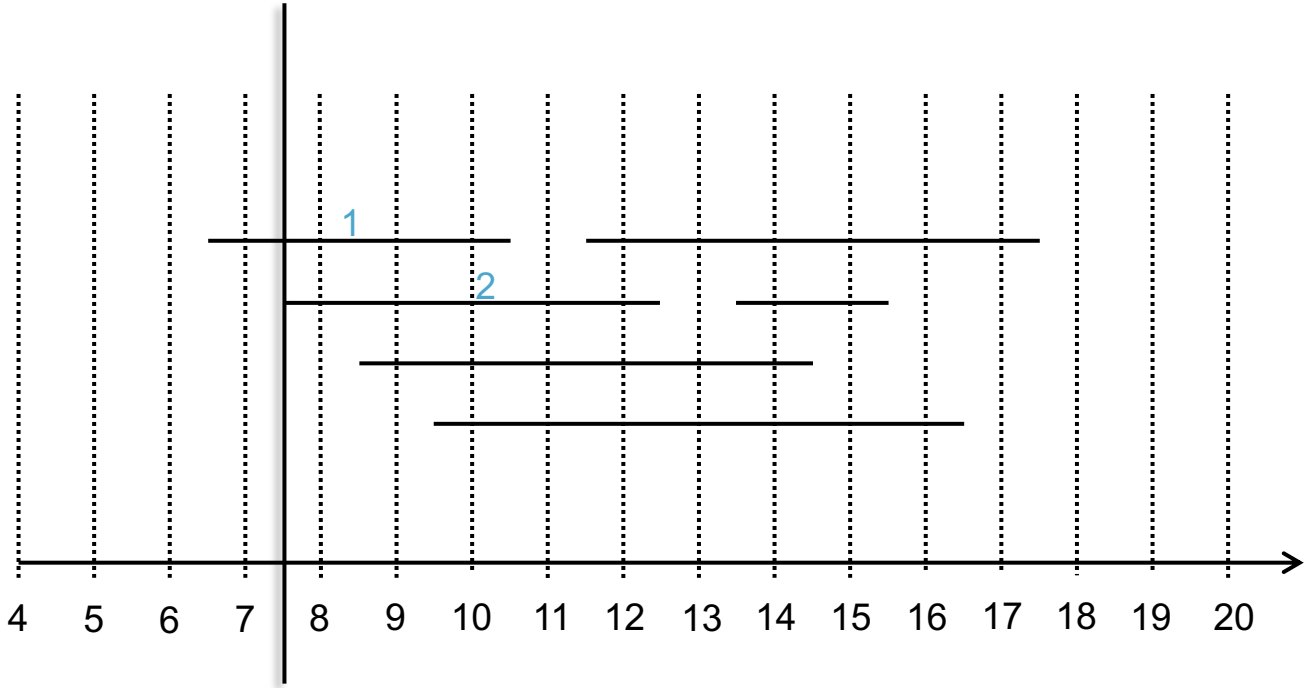


Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bei Erreichen eines

- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Hörsaal-Belegung – Eine Variante

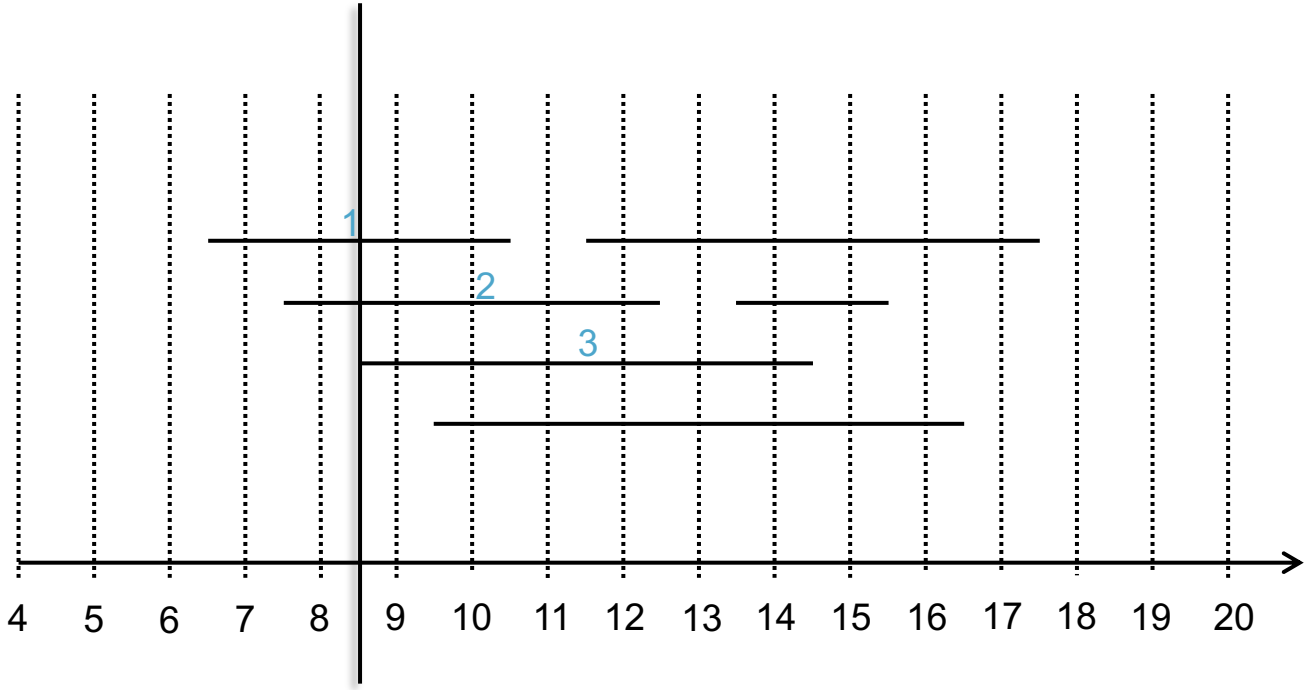


Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bei Erreichen eines

- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Hörsaal-Belegung – Eine Variante

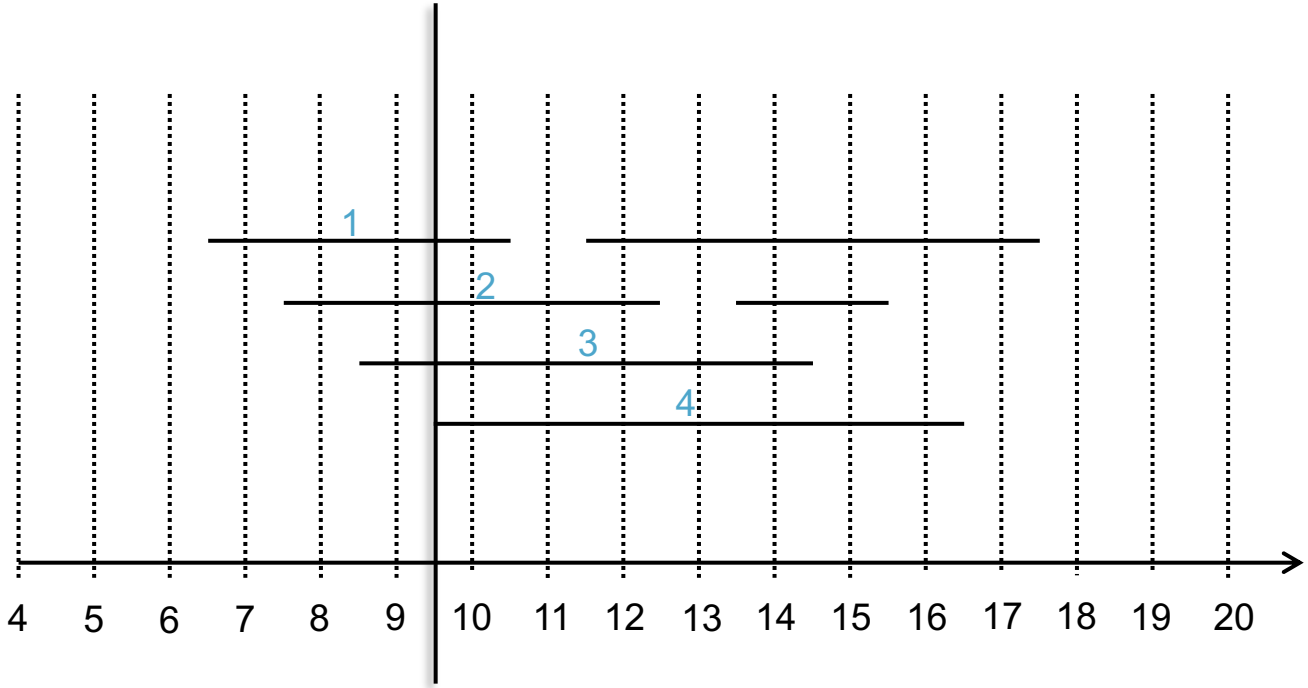


Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bei Erreichen eines

- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Hörsaal-Belegung – Eine Variante

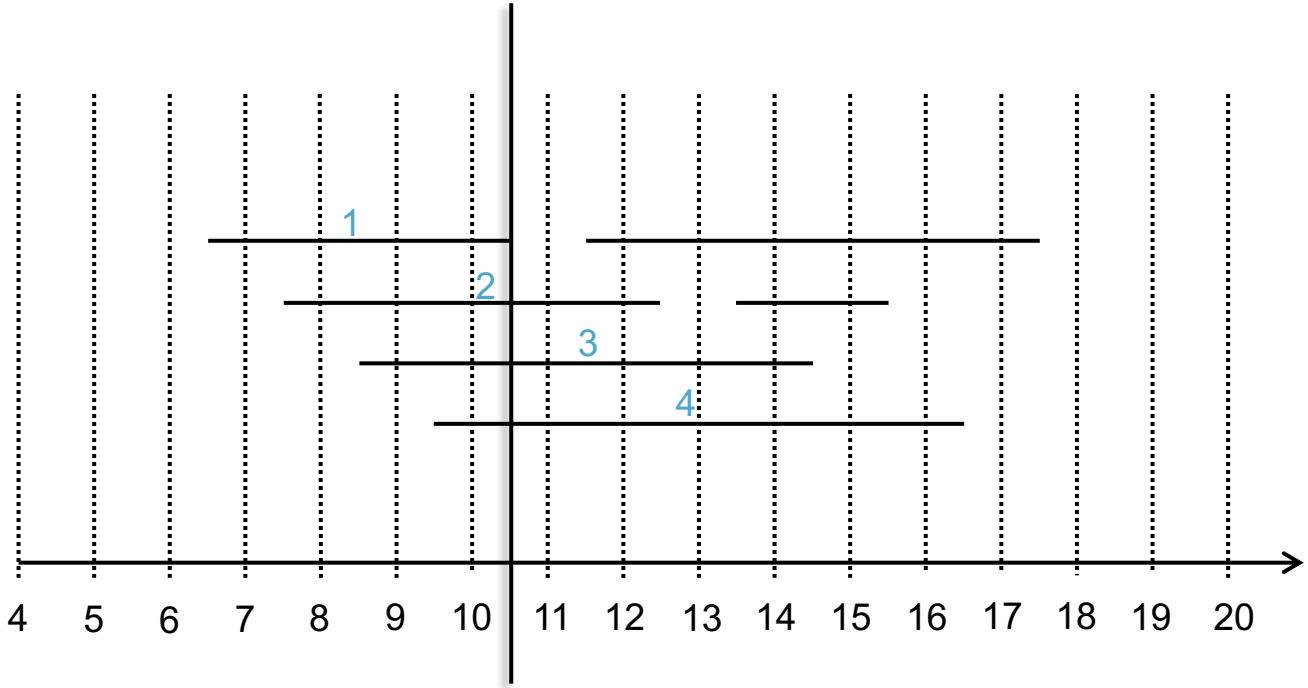


Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bei Erreichen eines

- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Hörsaal-Belegung – Eine Variante

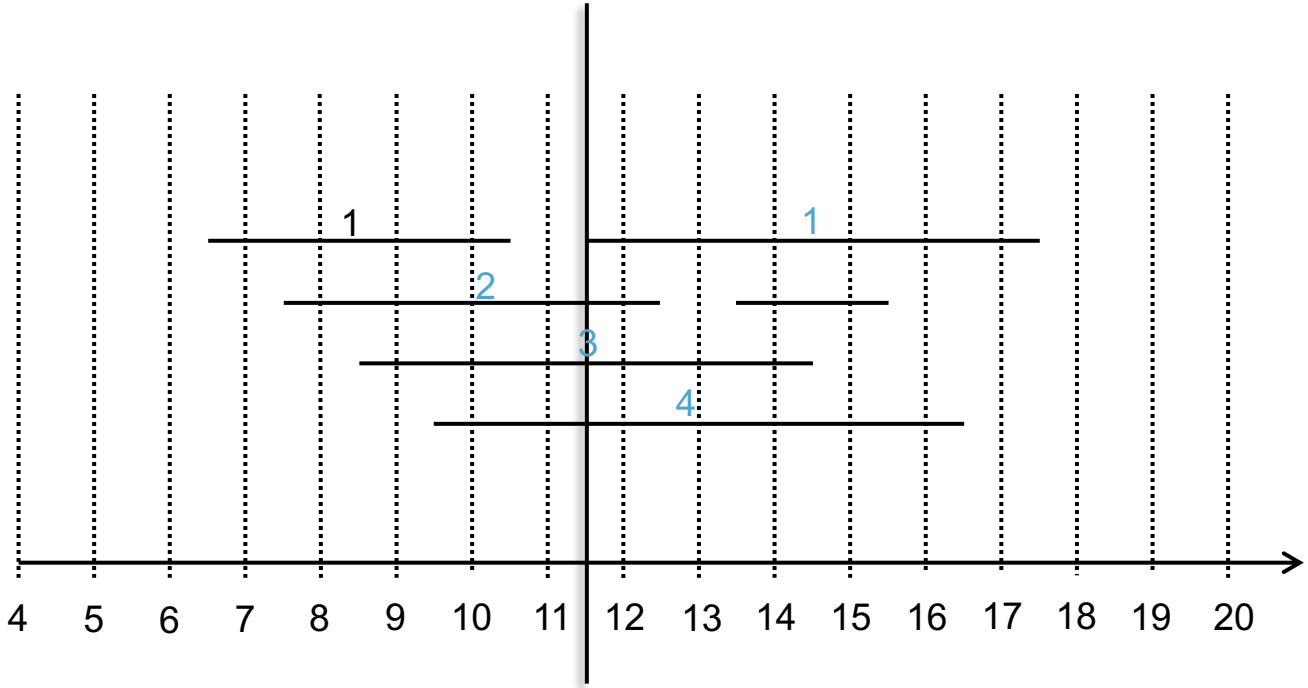


Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bei Erreichen eines

- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Hörsaal-Belegung – Eine Variante

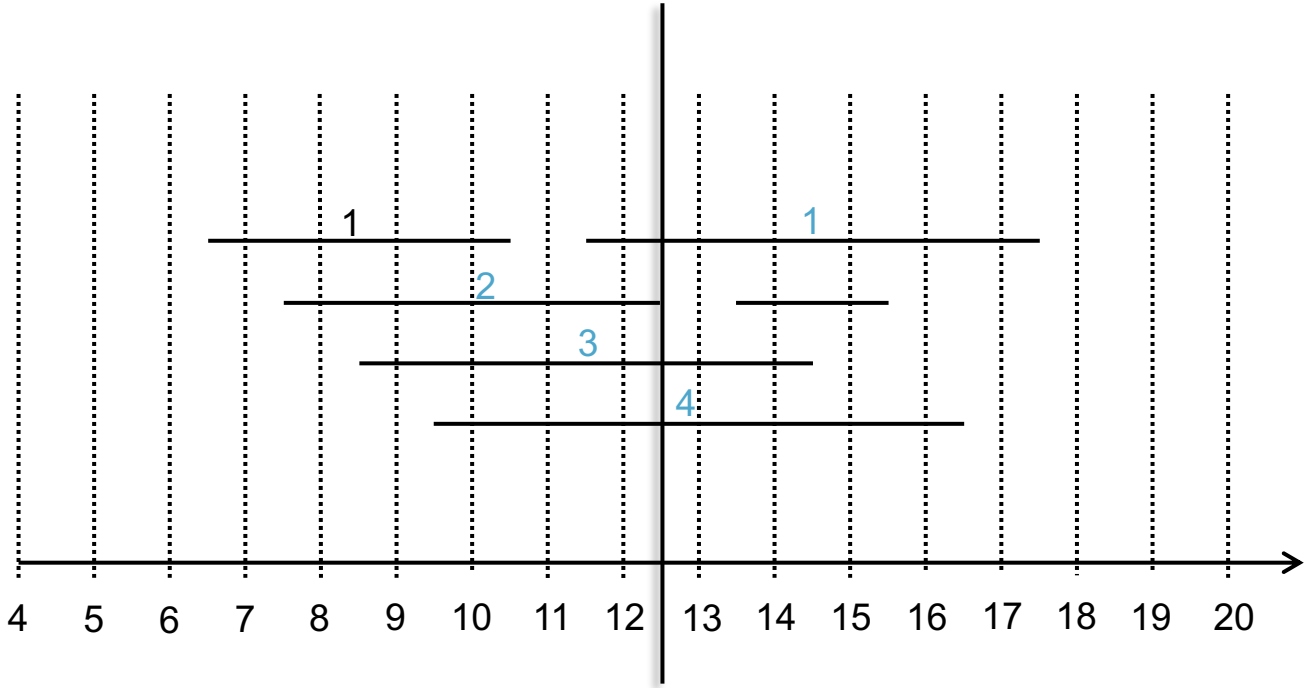


Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bei Erreichen eines

- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Hörsaal-Belegung – Eine Variante

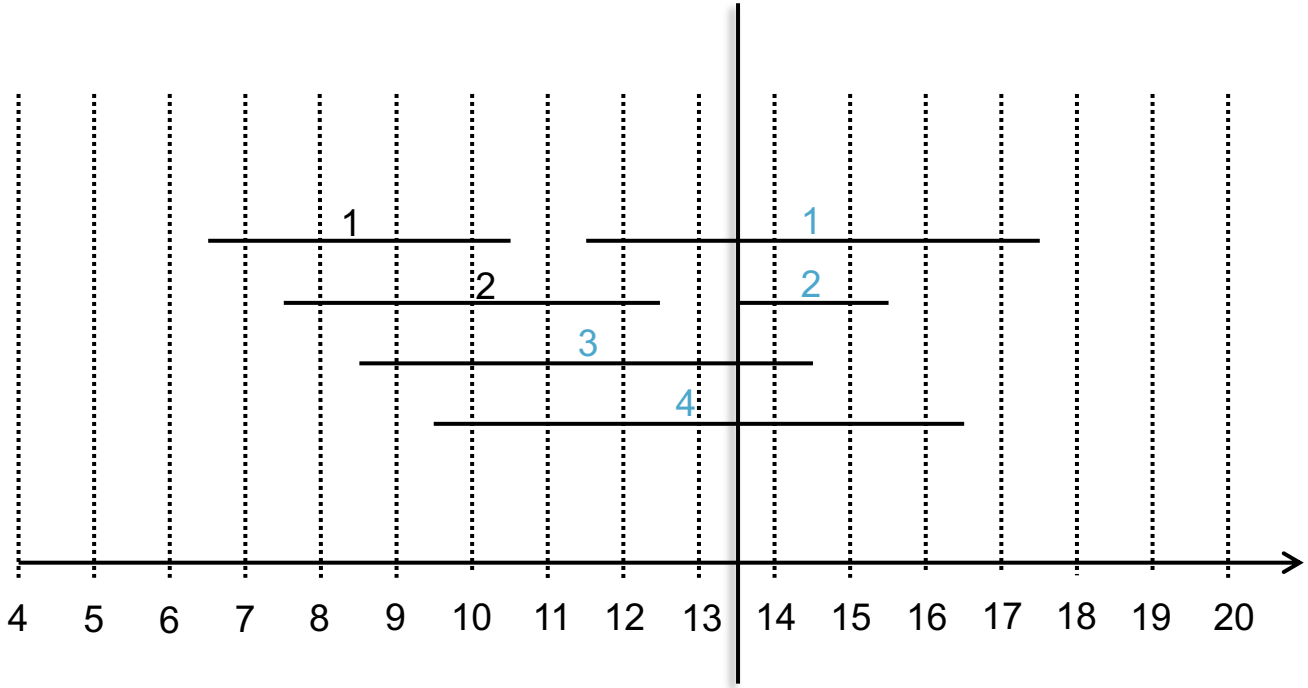


Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bei Erreichen eines

- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Hörsaal-Belegung – Eine Variante

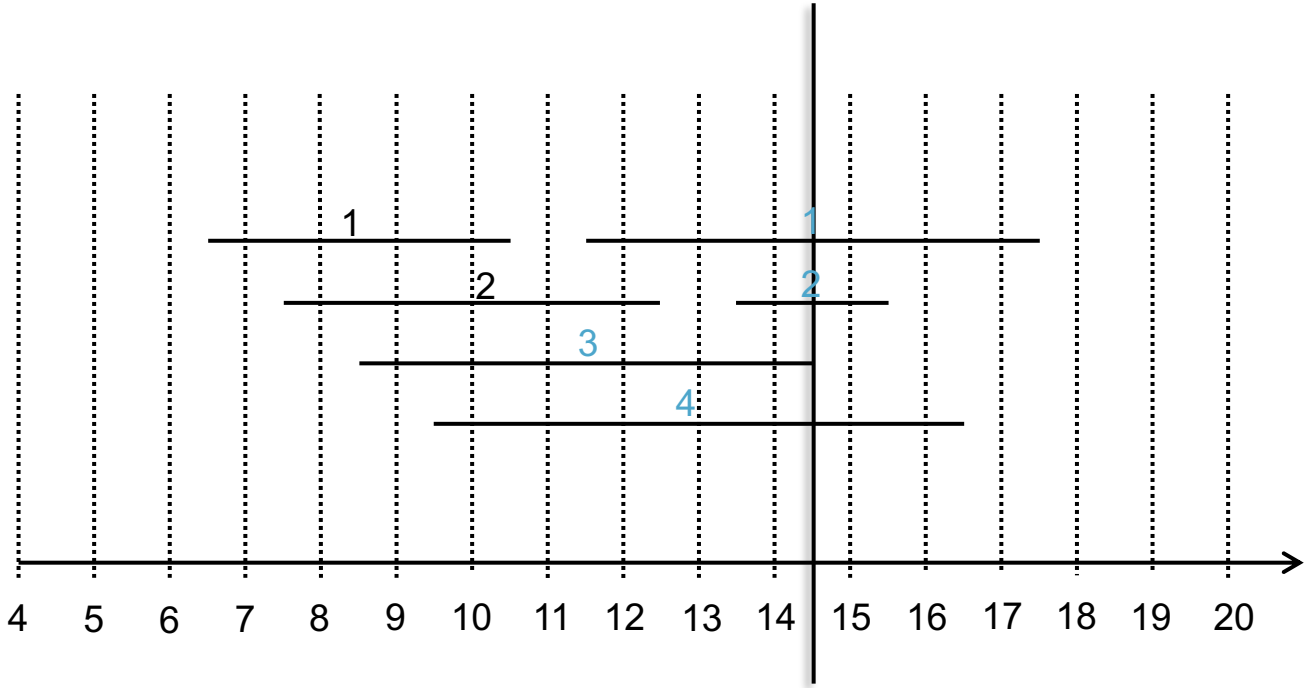


Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bei Erreichen eines

- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Hörsaal-Belegung – Eine Variante

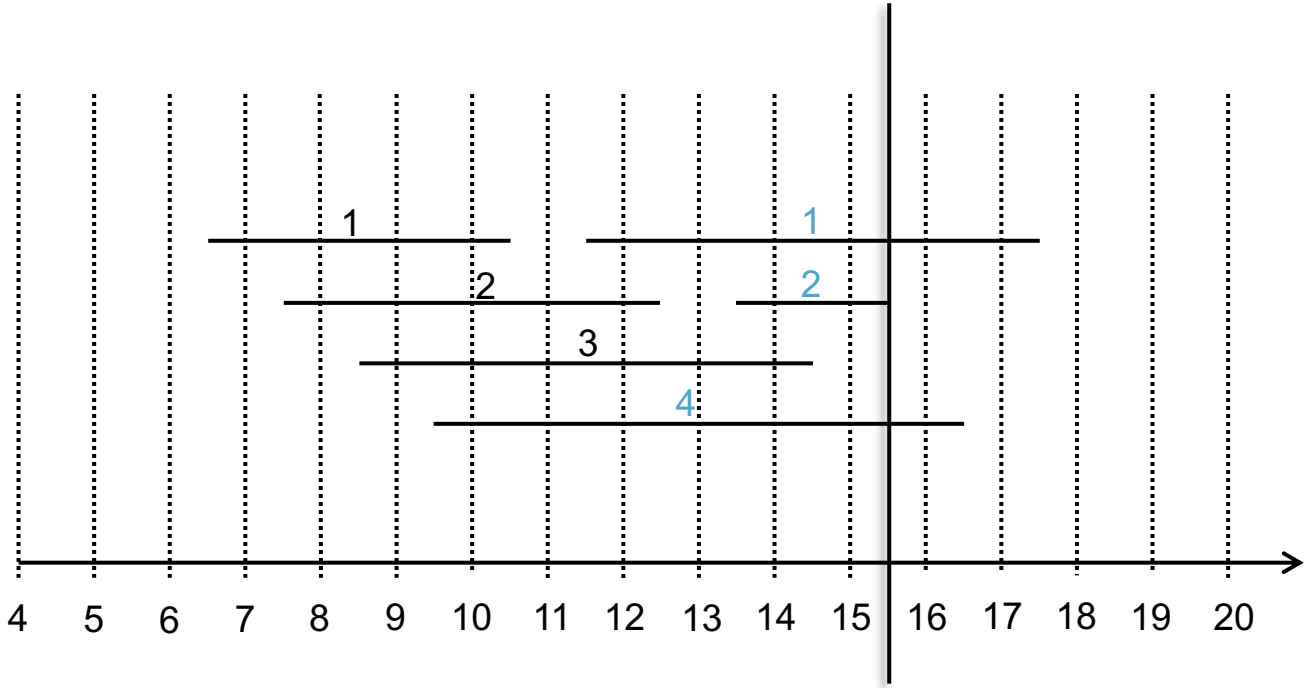


Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bei Erreichen eines

- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Hörsaal-Belegung – Eine Variante

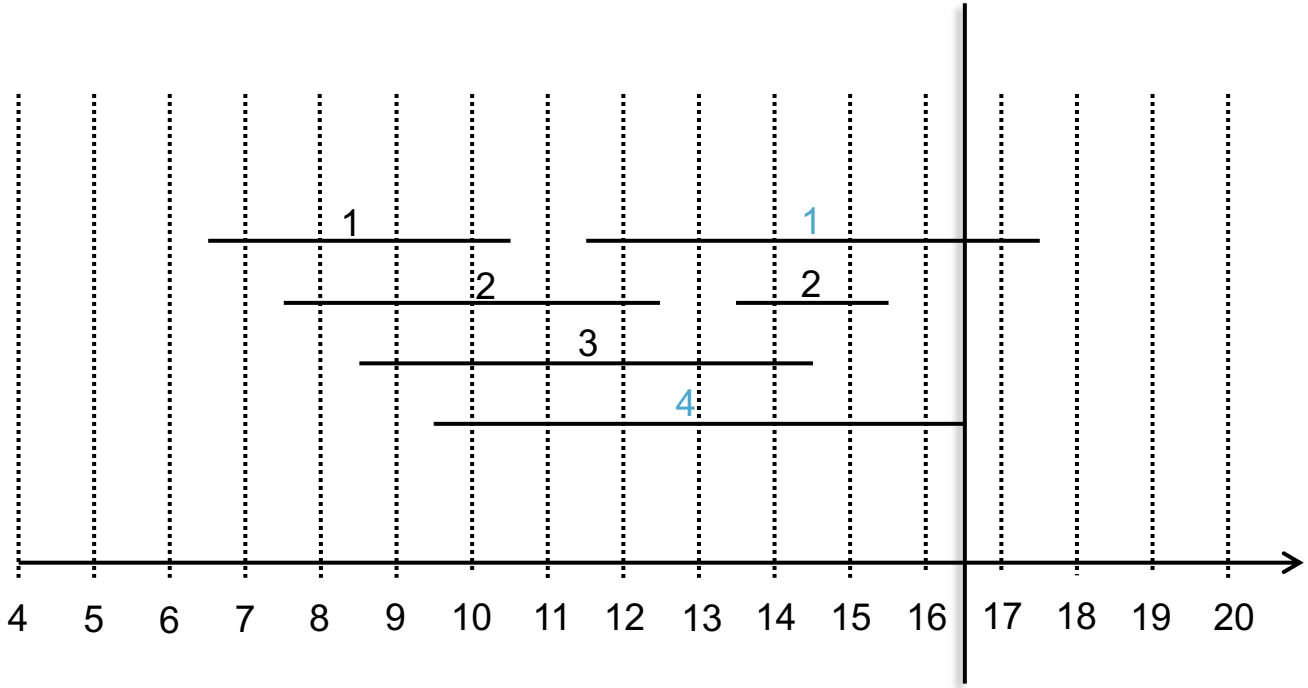


Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bei Erreichen eines

- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Hörsaal-Belegung – Eine Variante

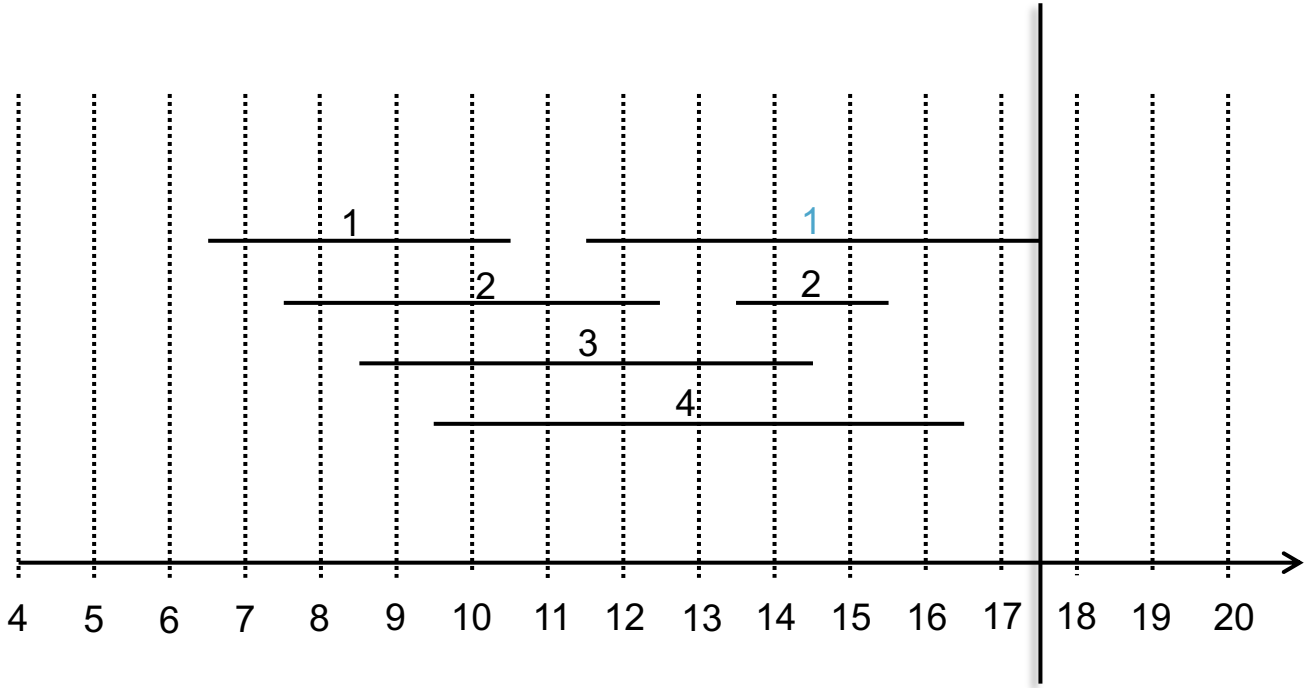


Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bei Erreichen eines

- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Hörsaal-Belegung – Eine Variante



Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

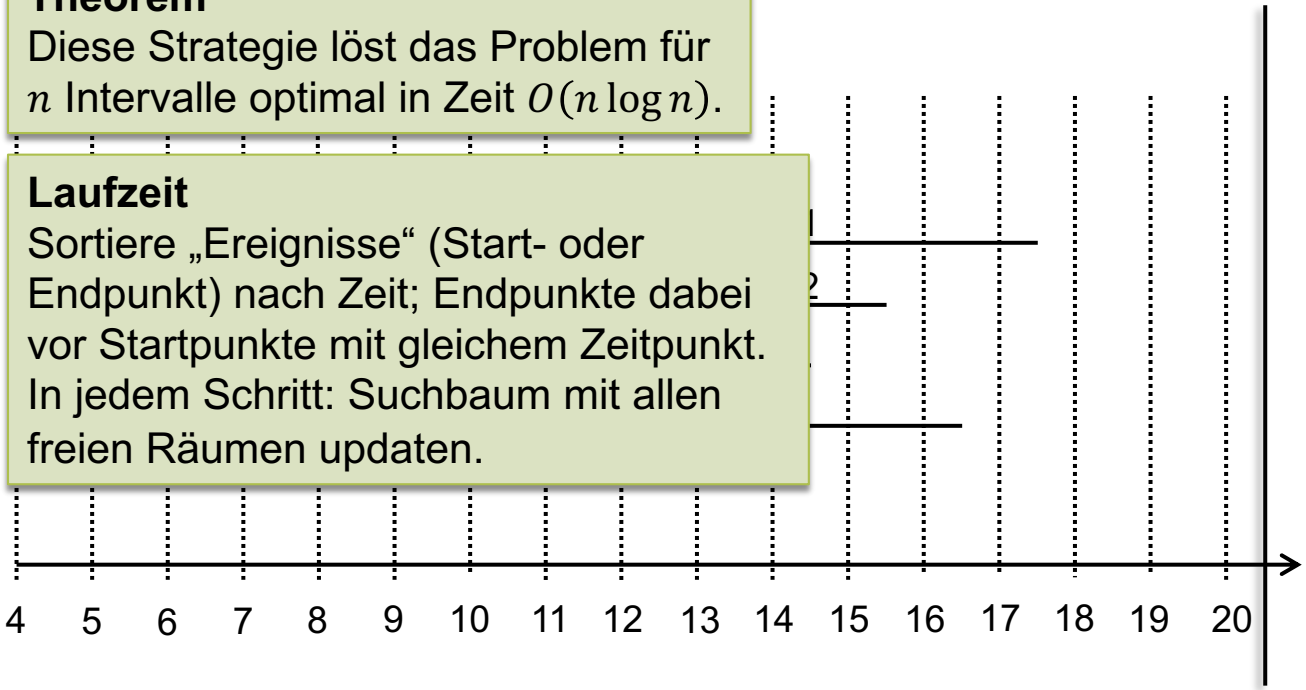
Bei Erreichen eines

- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Hörsaal-Belegung – Eine Variante

Theorem
Diese Strategie löst das Problem für n Intervalle optimal in Zeit $O(n \log n)$.

Laufzeit
Sortiere „Ereignisse“ (Start- oder Endpunkt) nach Zeit; Endpunkte dabei vor Startpunkte mit gleichem Zeitpunkt. In jedem Schritt: Suchbaum mit allen freien Räumen updaten.



Wichtig ist nicht nur das Ende, sondern auch der Start eines Intervalls!

Bei Erreichen eines
- **Startpunktes:**
  Weise kleinste Raumnummer zu
- **Endpunktes:**
  Gib Raum frei

Beweis

Wie viele Räume braucht man mindestens?

Sei

- $\chi := \max_i(|\{I \in \mathcal{I} : i \in I\}|)$,
- opt der Wert einer optimalen Lösung,
- alg der Wert der Lösung unserer Strategie

Maximale Anzahl an Intervallen, die sich gleichzeitig überlappen.

Beobachtung: $opt \geq \chi$

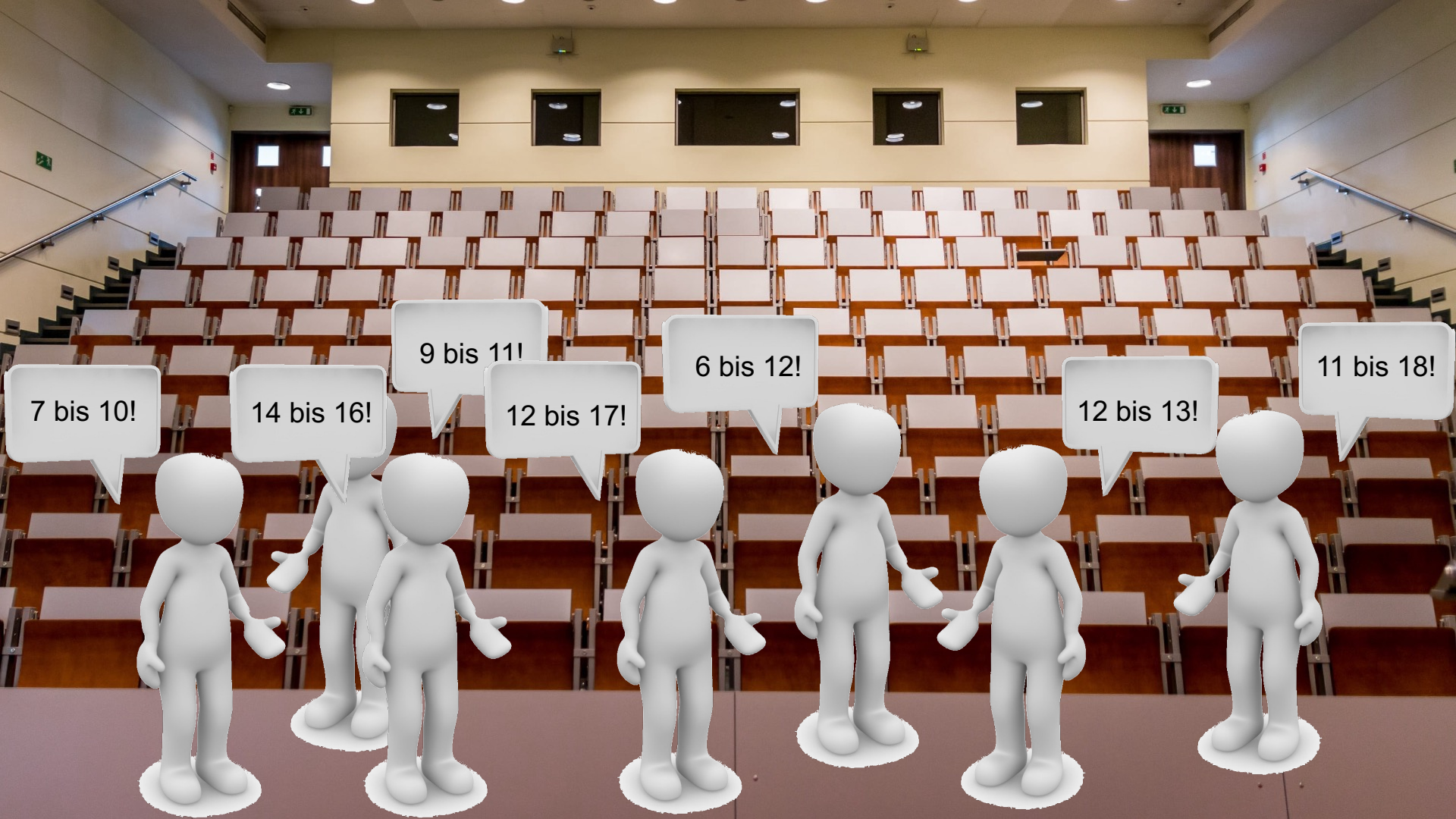
Wir zeigen: $alg \leq \chi$

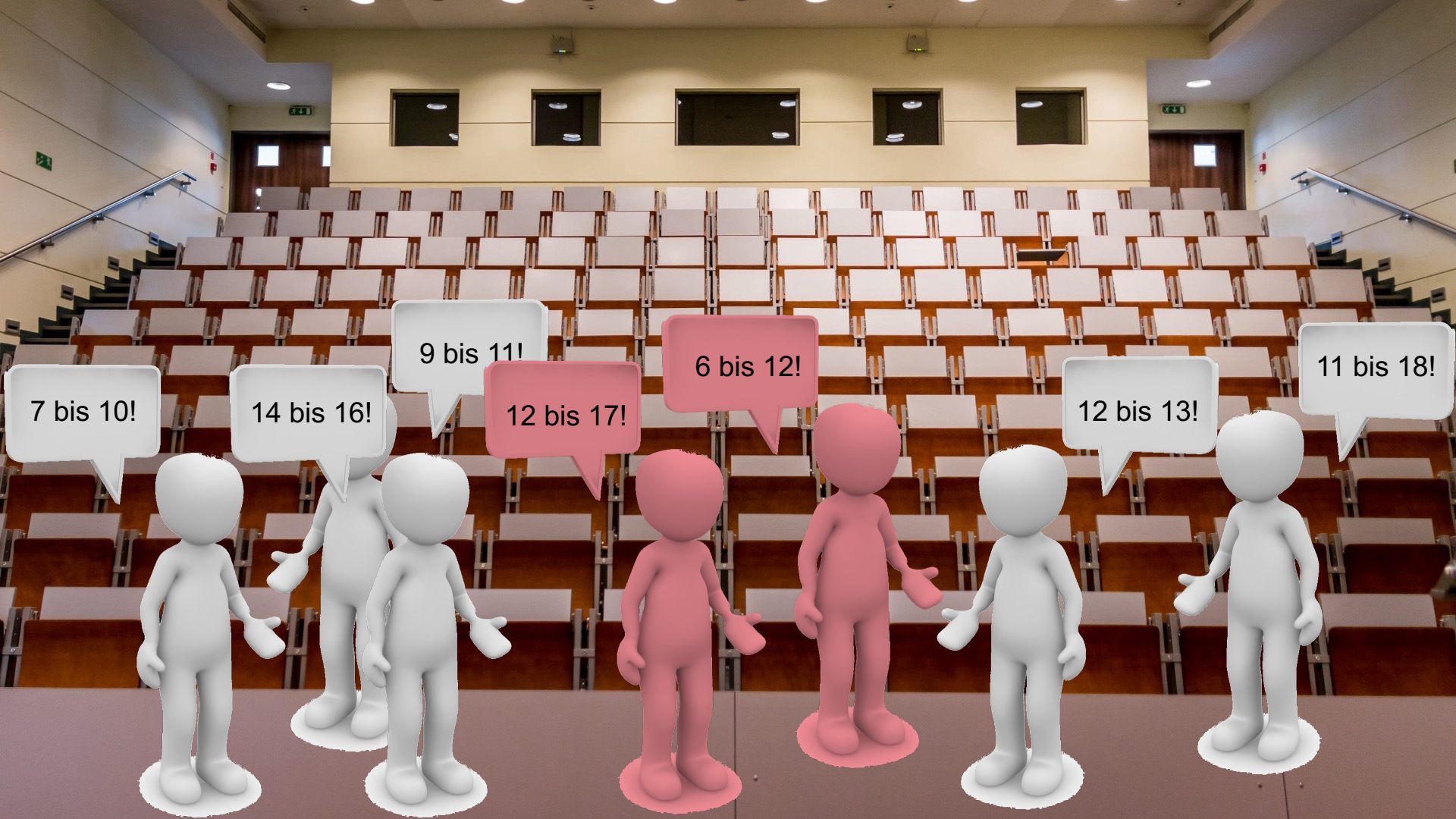
Beweis

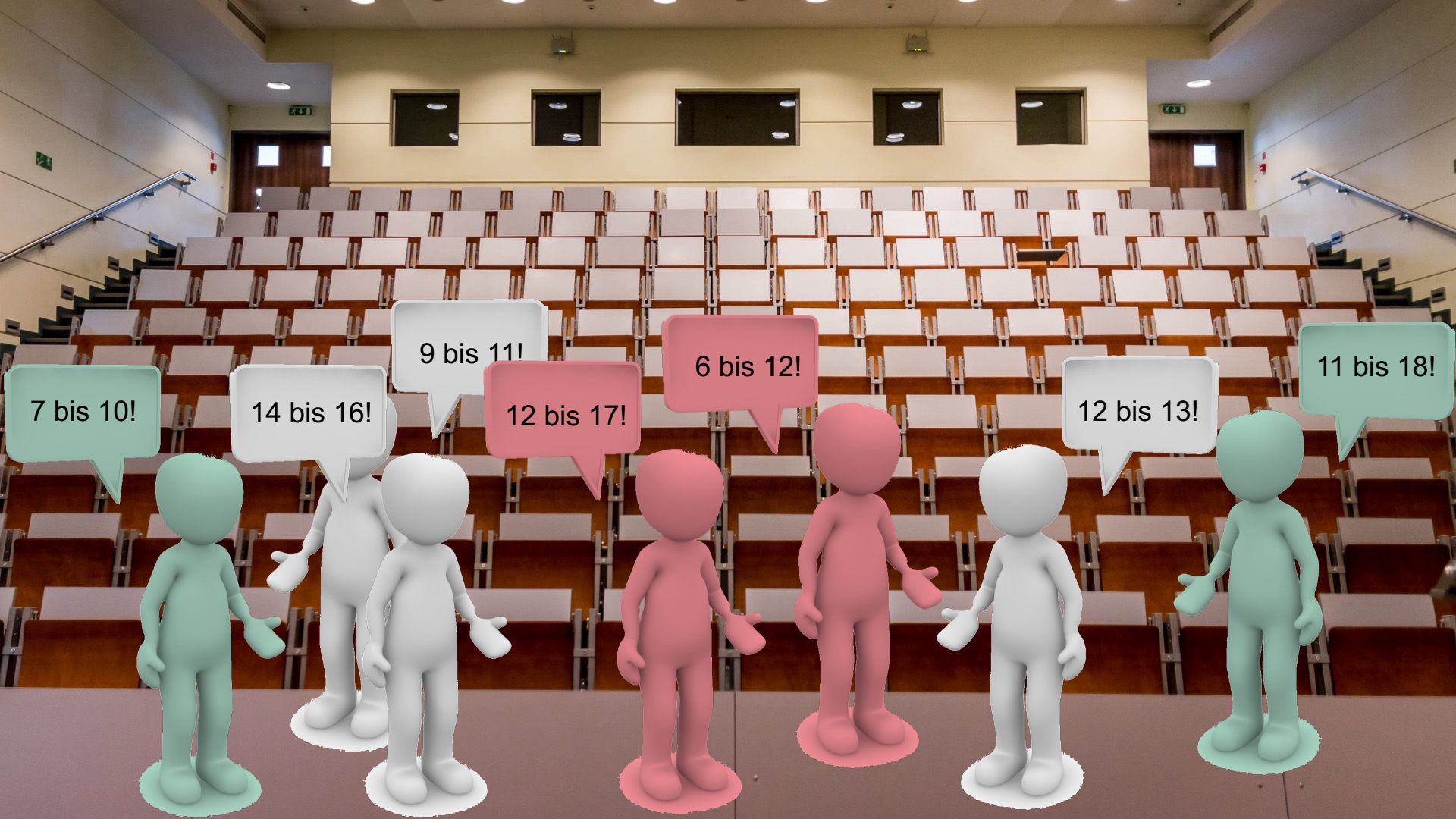
Angenommen, wir brauchen k Räume, d.h. $alg = k$.

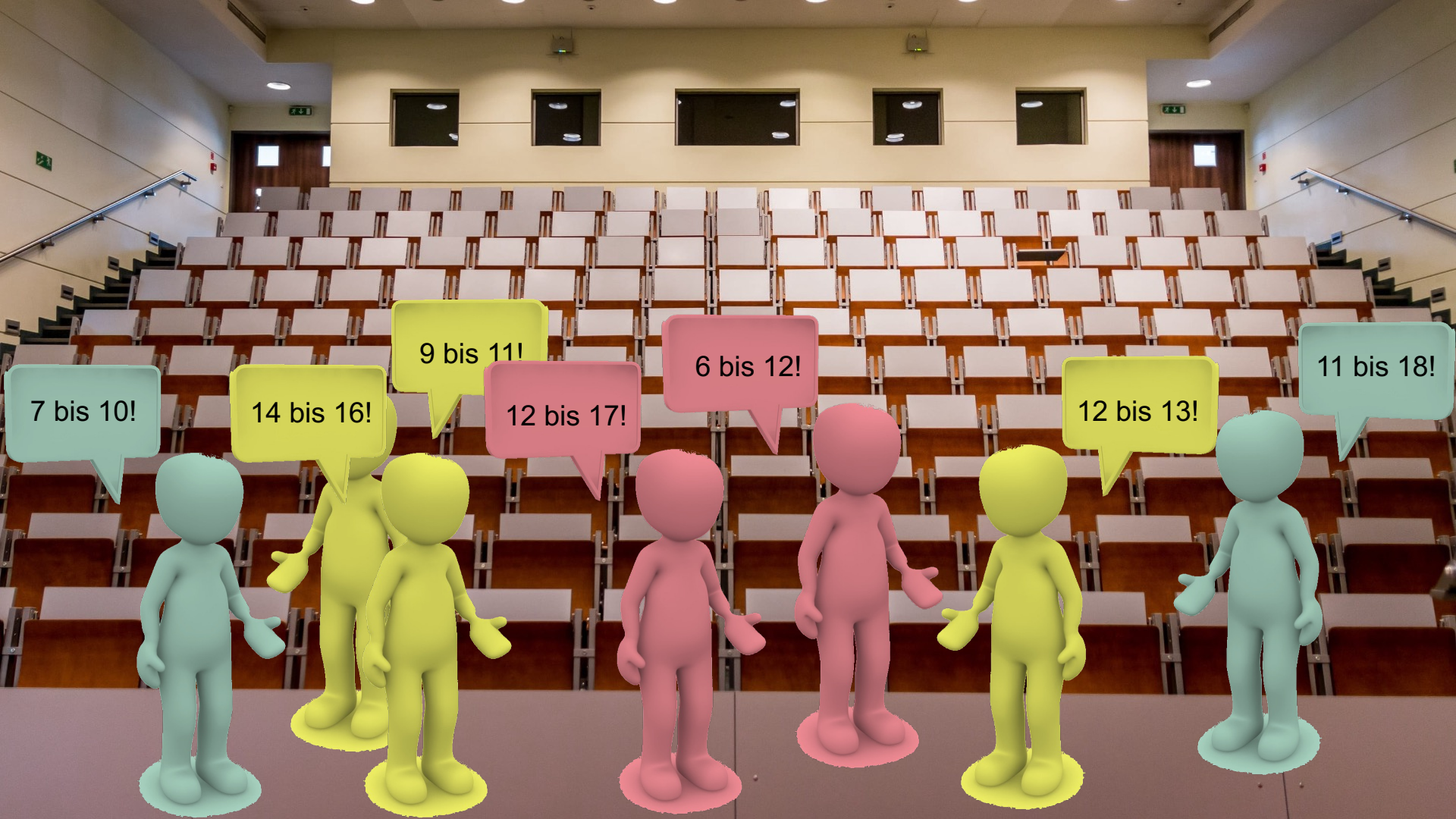
Betrachte den ersten Zeitpunkt, zu dem wir Raum k benutzen.
Das passiert nur beim Startpunkt eines neuen Intervalls I.
Was muss noch gelten?

- Alle anderen Räume $1, \dots, k-1$ sind in Gebrauch!
- D.h. I überlappt $k-1$ andere Intervalle an seinem Startpunkt.
- Endpunkte vor Startpunkten zum gleichen Zeitpunkt bearbeitet: echte Überlappung!


7 bis 10!
14 bis 16!
9 bis 11!
12 bis 17!
6 bis 12!
12 bis 13!
11 bis 18!


7 bis 10!
14 bis 16!
9 bis 11!
12 bis 17!
6 bis 12!
12 bis 13!
11 bis 18!


7 bis 10!
14 bis 16!
9 bis 11!
12 bis 17!
6 bis 12!
12 bis 13!
11 bis 18!


7 bis 10!
14 bis 16!
9 bis 11!
12 bis 17!
6 bis 12!
12 bis 13!
11 bis 18!

Verallgemeinerung

Hier vorgestellte Probleme sind Spezialfälle von berühmten Graphproblemen:

- Independent Set (unabhängige Menge bzw. stabile Menge)
  - **Gegeben:** ungerichteter Graph $G = (V, E)$
  - **Gesucht:** möglichst großes $S \subseteq V$ mit $\forall \{v, w\} \subseteq S: vw \notin E$ (keine Kanten in S)
- Graphfärbung
  - **Gegeben:** ungerichteter Graph $G = (V, E)$
  - **Gesucht:** Färbung $c: V \rightarrow \{1, \ldots, k\}$ mit $c(v) = c(w) \Rightarrow vw \notin E$
    - D.h. keine einfarbigen Kanten
    - Möglichst wenige Farben (möglichst kleines k)
  - Äquivalent: Möglichst wenige Independent Sets $S_i, 1 \leq i \leq k$ mit $\bigcup_{i=1}^{k} S_i = V$.
- Hier betrachtet auf so genannten Intervallgraphen
- Auf allgemeinen Graphen: Wohl kein effizienter Algorithmus für optimale Lösungen…