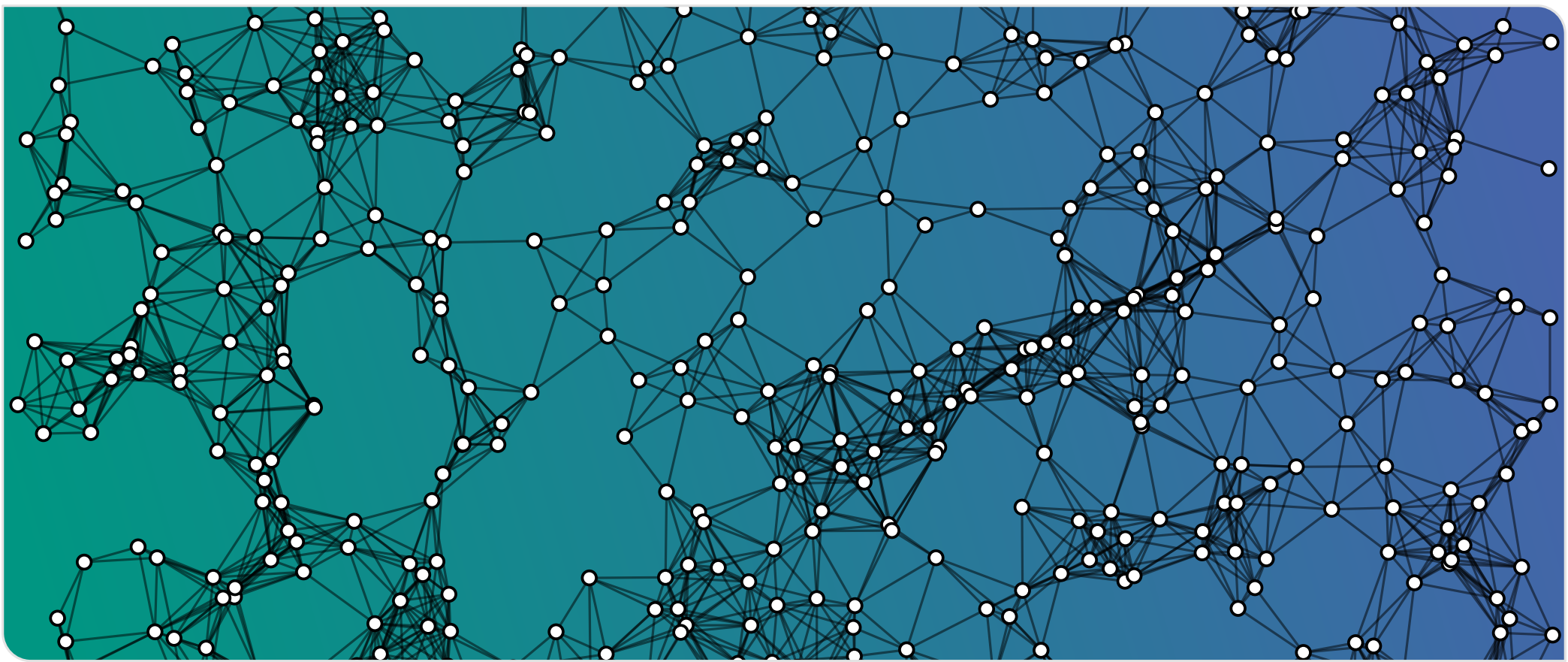
Karlsruher Institut für Technologie

# Parametrisierte Algorithmen

## Übung 6



# Heute

**Übungsblatt 5**

**Übungsblatt 6**

# Heute

**Übungsblatt 5**

- Sterne
- LONGEST CYCLE
- DOMINATING SET auf speziellen Graphen

**Übungsblatt 6**

# Heute

## Übungsblatt 5

- Sterne
- LONGEST CYCLE
- DOMINATING SET auf speziellen Graphen

## Übungsblatt 6

- Verschiedenes zu Baumweite
- dominierende Mengen
- Baumweite planarer Graphen

# Heute

## Übungsblatt 5

- Sterne
- Longest Cycle
- Dominating Set auf speziellen Graphen

## Übungsblatt 6

- Verschiedenes zu Baumweite
- dominierende Mengen
- Baumweite planarer Graphen

## Und sonst?

# Heute

## Übungsblatt 5

- Sterne
- Longest Cycle
- Dominating Set auf speziellen Graphen

## Übungsblatt 6

- Verschiedenes zu Baumweite
- dominierende Mengen
- Baumweite planarer Graphen

## Und sonst?

- Wie rechnet man Baumweite aus?

# Heute

## Übungsblatt 5

- Sterne
- Longest Cycle
- Dominating Set auf speziellen Graphen

## Übungsblatt 6

- Verschiedenes zu Baumweite
- dominierende Mengen
- Baumweite planarer Graphen

## Und sonst?

- Wie rechnet man Baumweite aus? (in der echten Welt)

# Heute

## Übungsblatt 5

- Sterne
- Longest Cycle
- Dominating Set auf speziellen Graphen

## Übungsblatt 6

- Verschiedenes zu Baumweite
- dominierende Mengen
- Baumweite planarer Graphen

## Und sonst?

- Wie rechnet man Baumweite aus? (in der echten Welt)
- Evaluation

# Heute

## Übungsblatt 5

- Sterne
- Longest Cycle
- Dominating Set auf speziellen Graphen

## Übungsblatt 6

- Verschiedenes zu Baumweite
- dominierende Mengen
- Baumweite planarer Graphen

## Und sonst?

- Wie rechnet man Baumweite aus? (in der echten Welt)
- Evaluation (zuerst)

# Mehr zu Baumweite

**Definition:** (VL)
*G chordal* $\Leftrightarrow$ *G* hat keinen induzierten Kreis der Länge 4 oder mehr

# Mehr zu Baumweite

**Definition:** (VL)
*G chordal* $\Leftrightarrow$ *G* hat keinen induzierten Kreis der Länge 4 oder mehr

**Behauptung:**
Baumweite von chordalen Graphen lässt sich in $n^{O(1)}$ entscheiden

# Mehr zu Baumweite

**Definition:** (VL)
*G chordal* $\Leftrightarrow$ *G* hat keinen induzierten Kreis der Länge 4 oder mehr

**Behauptung:**
Baumweite von chordalen Graphen lässt sich in $n^{O(1)}$ entscheiden

**Satz:** Sei *G* chordal. Dann gibt es einen sogenannten *simplizialen* Knoten $v \in V(G)$ dessen Nachbarschaft eine Clique bildet.

# Mehr zu Baumweite

**Definition:** (VL)
*G chordal* $\Leftrightarrow$ *G* hat keinen induzierten Kreis der Länge 4 oder mehr

**Behauptung:**
Baumweite von chordalen Graphen lässt sich in $n^{O(1)}$ entscheiden

**Satz:** Sei *G* chordal. Dann gibt es einen sogenannten *simplizialen* Knoten $v \in V(G)$ dessen Nachbarschaft eine Clique bildet. (Beweis auf Blatt 7)

# Mehr zu Baumweite

**Definition:** (VL)
*G chordal* $\Leftrightarrow$ *G* hat keinen induzierten Kreis der Länge 4 oder mehr

**Behauptung:**
Baumweite von chordalen Graphen lässt sich in $n^{O(1)}$ entscheiden

**Satz:** Sei $G$ chordal. Dann gibt es einen sogenannten *simplizialen* Knoten $v \in V(G)$ dessen Nachbarschaft eine Clique bildet. (Beweis auf Blatt 7)

**Algorithmus:**

# Mehr zu Baumweite

**Definition:** (VL)
*G chordal* $\Leftrightarrow$ *G* hat keinen induzierten Kreis der Länge 4 oder mehr

**Behauptung:**
Baumweite von chordalen Graphen lässt sich in $n^{O(1)}$ entscheiden

**Satz:** Sei *G* chordal. Dann gibt es einen sogenannten *simplizialen* Knoten $v \in V(G)$ dessen Nachbarschaft eine Clique bildet. (Beweis auf Blatt 7)

**Algorithmus:**

- finde simplizialen Knoten *v*

# Mehr zu Baumweite

**Definition:** (VL)
*G chordal* $\Leftrightarrow$ *G* hat keinen induzierten Kreis der Länge 4 oder mehr

**Behauptung:**
Baumweite von chordalen Graphen lässt sich in $n^{O(1)}$ entscheiden

**Satz:** Sei $G$ chordal. Dann gibt es einen sogenannten *simplizialen* Knoten $v \in V(G)$ dessen Nachbarschaft eine Clique bildet. (Beweis auf Blatt 7)

**Algorithmus:**

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)

# Mehr zu Baumweite

**Definition:** (VL)
*G chordal* $\Leftrightarrow$ *G* hat keinen induzierten Kreis der Länge 4 oder mehr

**Behauptung:**
Baumweite von chordalen Graphen lässt sich in $n^{O(1)}$ entscheiden

**Satz:** Sei $G$ chordal. Dann gibt es einen sogenannten *simplizialen* Knoten $v \in V(G)$ dessen Nachbarschaft eine Clique bildet. (Beweis auf Blatt 7)

**Algorithmus:**

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

# Was wenn $G$ nicht chordal ist?

Karlsruher Institut für Technologie

**Algorithmus:**

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

# Was wenn $G$ nicht chordal ist?

**Algorithmus:**

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

# Was wenn $G$ nicht chordal ist?

**Algorithmus:**

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

**Satz:** Jeder Graph hat ein Eliminationsschema, mit dem sich die korrekte Baumweite bestimmen lässt.

# Was wenn $G$ nicht chordal ist?

**Algorithmus:**

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

**Satz:** Jeder Graph hat ein Eliminationsschema, mit dem sich die korrekte Baumweite bestimmen lässt.

Wieso?

# Was wenn $G$ nicht chordal ist?

**Algorithmus:**

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

**Satz:** Jeder Graph hat ein Eliminationsschema, mit dem sich die korrekte Baumweite bestimmen lässt.

Wieso?

**Chordalweite**

- $\text{chordalwidth}(G) = \min\{\omega(G') \mid G \subseteq G' \text{ und } G' \text{ ist chordal}\}$
- $\Rightarrow \text{tw}(G) = \text{chordalwidth}(G) - 1$

# Was wenn $G$ nicht chordal ist?

## Algorithmus:

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

**Satz:** Jeder Graph hat ein Eliminationsschema, mit dem sich die korrekte Baumweite bestimmen lässt.

Wieso?

**Chordalweite**

- $\text{chordalwidth}(G) = \min\{\omega(G') \mid G \subseteq G' \text{ und } G' \text{ ist chordal}\}$
- $\Rightarrow \text{tw}(G) = \text{chordalwidth}(G) - 1$

## Baumweite lösen:

# Was wenn $G$ nicht chordal ist?

## Algorithmus:

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

**Satz:** Jeder Graph hat ein Eliminationsschema, mit dem sich die korrekte Baumweite bestimmen lässt.

Wieso?

**Chordalweite**

- $\text{chordalwidth}(G) = \min\{\omega(G') \mid G \subseteq G' \text{ und } G' \text{ ist chordal}\}$
- $\Rightarrow \text{tw}(G) = \text{chordalwidth}(G) - 1$

## Baumweite lösen:

- heuristisch: *min-degree* heuristic, *min-fill-in* heuristic

# Was wenn $G$ nicht chordal ist?

## Algorithmus:

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

**Satz:** Jeder Graph hat ein Eliminationsschema, mit dem sich die korrekte Baumweite bestimmen lässt.

Wieso?

**Chordalweite**

- $\text{chordalwidth}(G) = \min\{\omega(G') \mid G \subseteq G' \text{ und } G' \text{ ist chordal}\}$
- $\Rightarrow \text{tw}(G) = \text{chordalwidth}(G) - 1$

## Baumweite lösen:

- heuristisch: *min-degree* heuristic, *min-fill-in* heuristic
- heuristisch (fancy)

# Was wenn $G$ nicht chordal ist?

## Algorithmus:

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

**Satz:** Jeder Graph hat ein Eliminationsschema, mit dem sich die korrekte Baumweite bestimmen lässt.

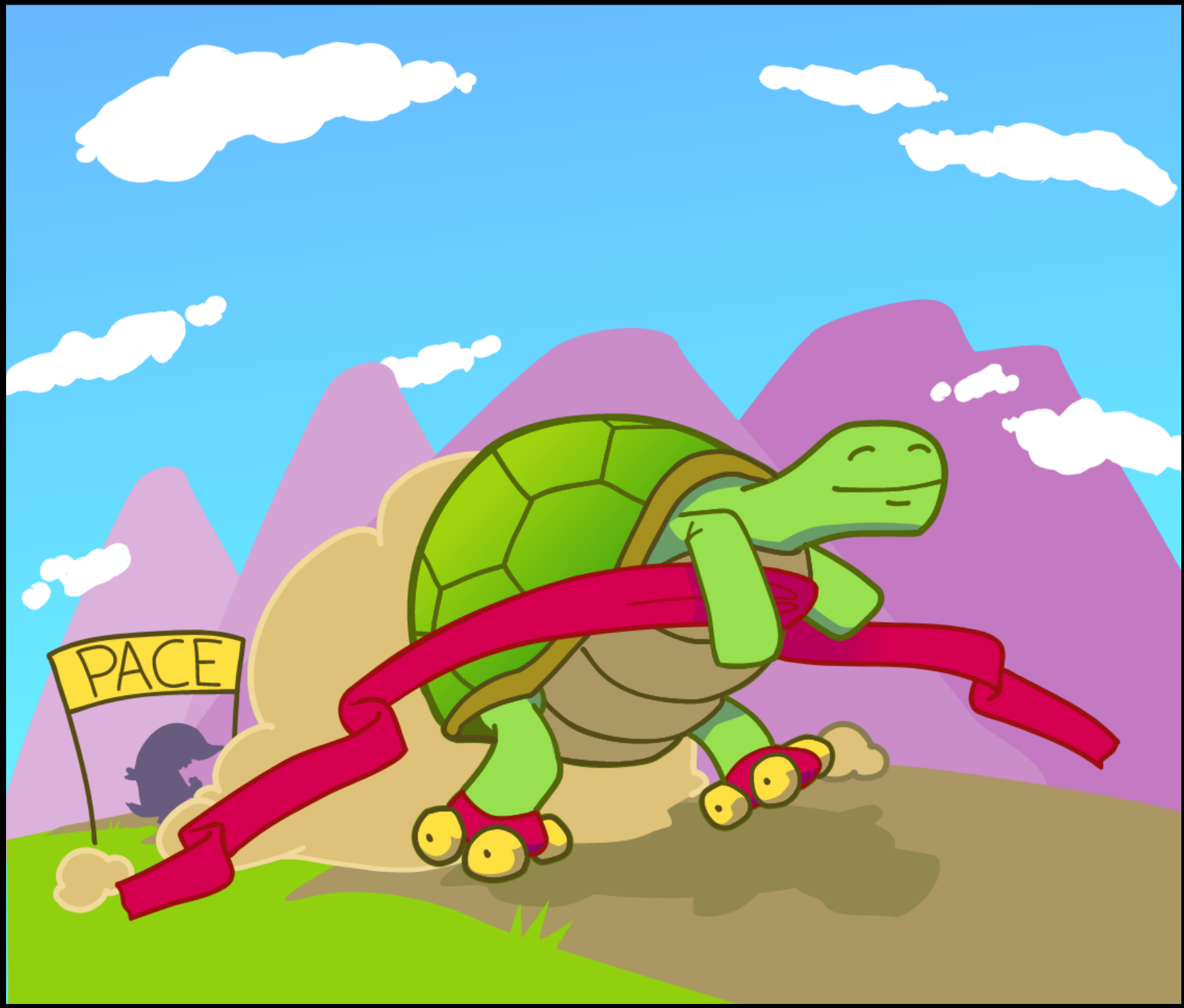
Wieso?

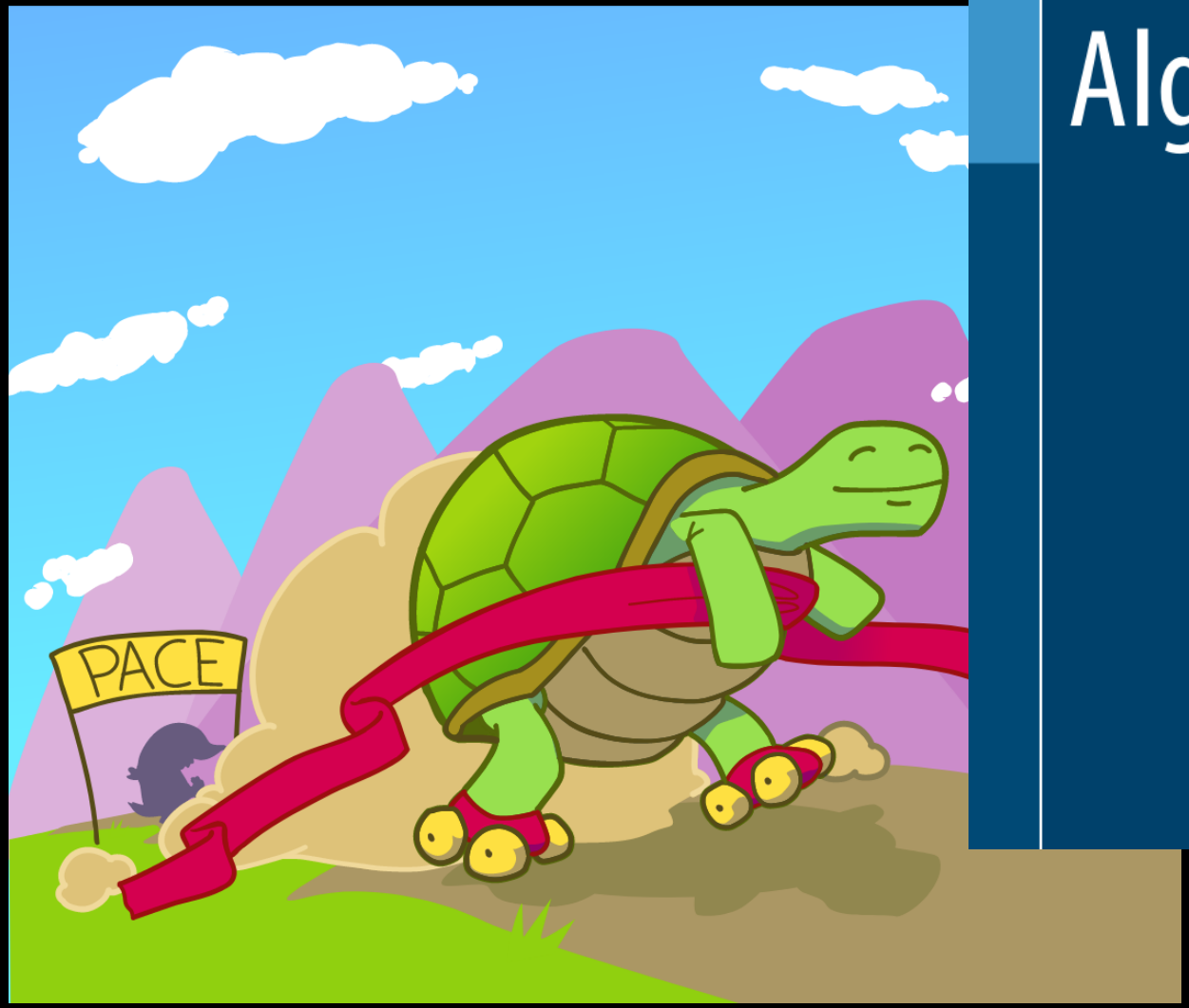
**Chordalweite**

- $\text{chordalwidth}(G) = \min\{\omega(G') \mid G \subseteq G' \text{ und } G' \text{ ist chordal}\}$
- $\Rightarrow \text{tw}(G) = \text{chordalwidth}(G) - 1$

## Baumweite lösen:

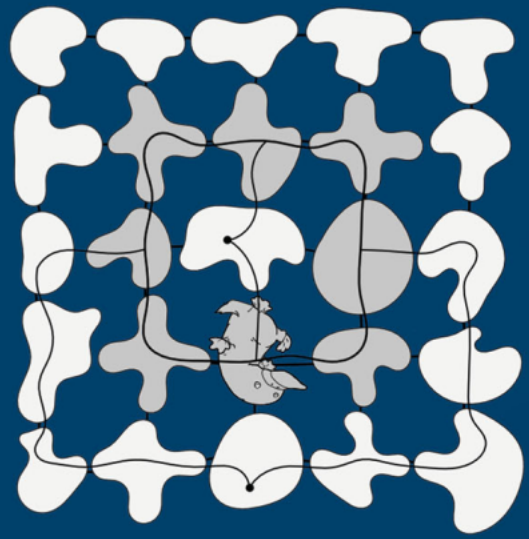
- heuristisch: *min-degree* heuristic, *min-fill-in* heuristic
- heuristisch (fancy)
- heuristisch (exakt)


PACE

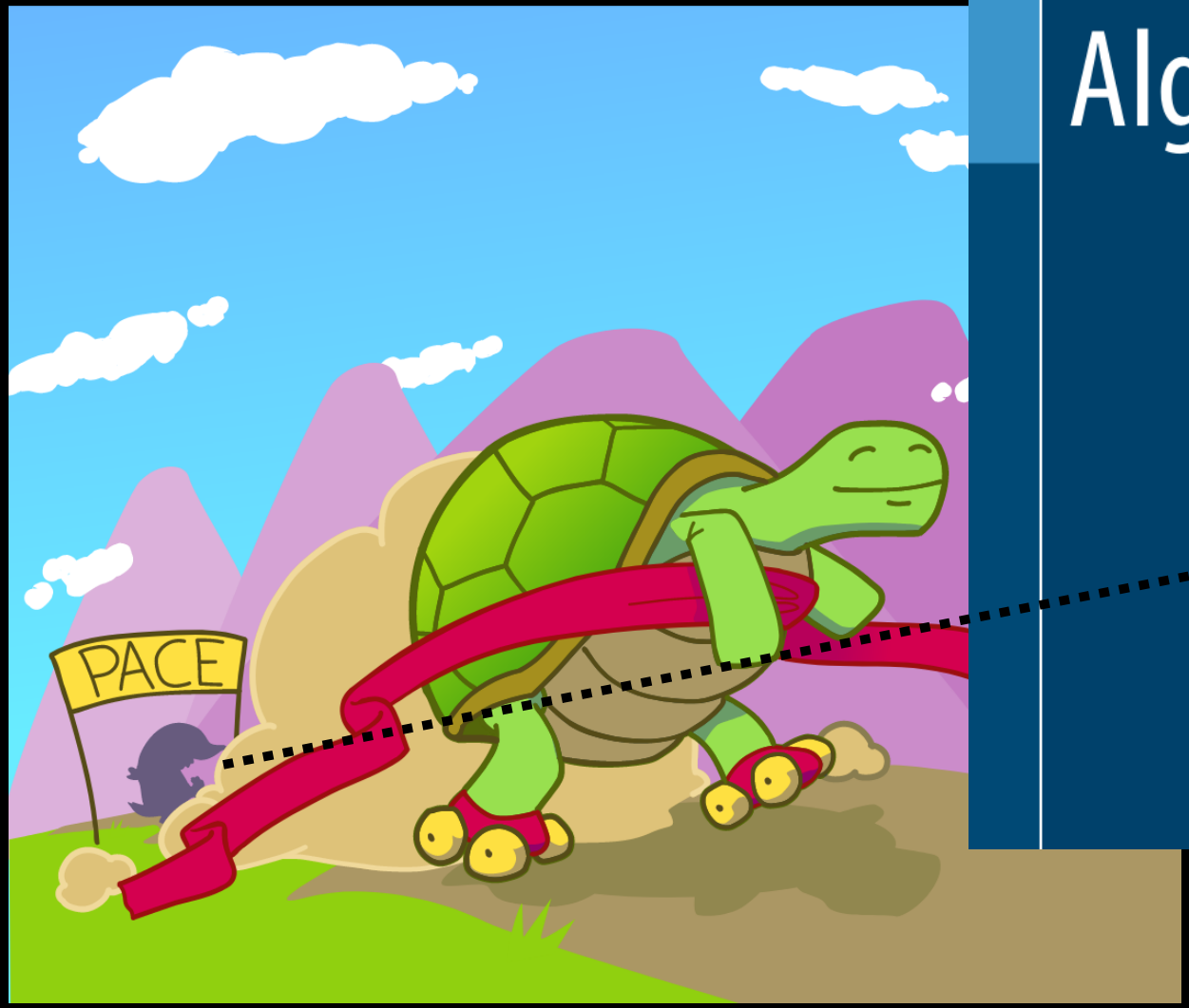



Marek Cygan · Fedor V. Fomin
Łukasz Kowalik · Daniel Lokshtanov
Dániel Marx · Marcin Pilipczuk
Michał Pilipczuk · Saket Saurabh

# Parameterized Algorithms

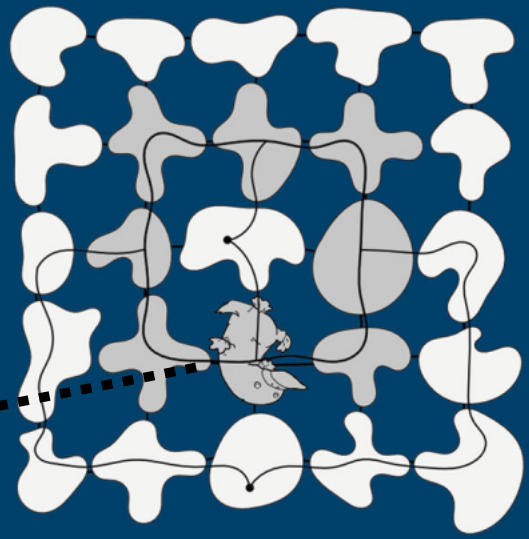


Springer




Marek Cygan · Fedor V. Fomin
Łukasz Kowalik · Daniel Lokshtanov
Dániel Marx · Marcin Pilipczuk
Michał Pilipczuk · Saket Saurabh

# Parameterized Algorithms



Springer

# Was wenn $G$ nicht chordal ist?

## Algorithmus:

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

**Satz:** Jeder Graph hat ein Eliminationsschema, mit dem sich die korrekte Baumweite bestimmen lässt.

Wieso?

**Chordalweite**

- $\text{chordalwidth}(G) = \min\{\omega(G') \mid G \subseteq G' \text{ und } G' \text{ ist chordal}\}$
- $\Rightarrow \text{tw}(G) = \text{chordalwidth}(G) - 1$

## Baumweite lösen:

- heuristisch: *min-degree* heuristic, *min-fill-in* heuristic
- heuristisch (fancy)
- heuristisch (exakt)

# Was wenn $G$ nicht chordal ist?

Karlsruher Institut für Technologie

## Algorithmus:

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

**Satz:** Jeder Graph hat ein Eliminationsschema, mit dem sich die korrekte Baumweite bestimmen lässt.

Wieso?

**Chordalweite**

- $\text{chordalwidth}(G) = \min\{\omega(G') \mid G \subseteq G' \text{ und } G' \text{ ist chordal}\}$
- $\Rightarrow \text{tw}(G) = \text{chordalwidth}(G) - 1$

## Baumweite lösen:

- heuristisch: *min-degree* heuristic, *min-fill-in* heuristic
- heuristisch (fancy)
- heuristisch (exakt)

} Paper aus PACE Challenge anschauen

# Was wenn $G$ nicht chordal ist?

Karlsruher Institut für Technologie

## Algorithmus:

- finde simplizialen Knoten $v$
- sei $(T, \{X_t\})$ Baumzerlegung von $G \setminus \{v\}$ (rekursiv)
- $(T, \{X_t\})$ mit zusätzlichem Blatt-Bag $N[v]$ verbunden zu Bag $X_t$ mit $X_t \subseteq N(v)$ ist Baumzerlegung von $G$ :)

**Satz:** Jeder Graph hat ein Eliminationsschema, mit dem sich die korrekte Baumweite bestimmen lässt.

Wieso?

**Chordalweite**

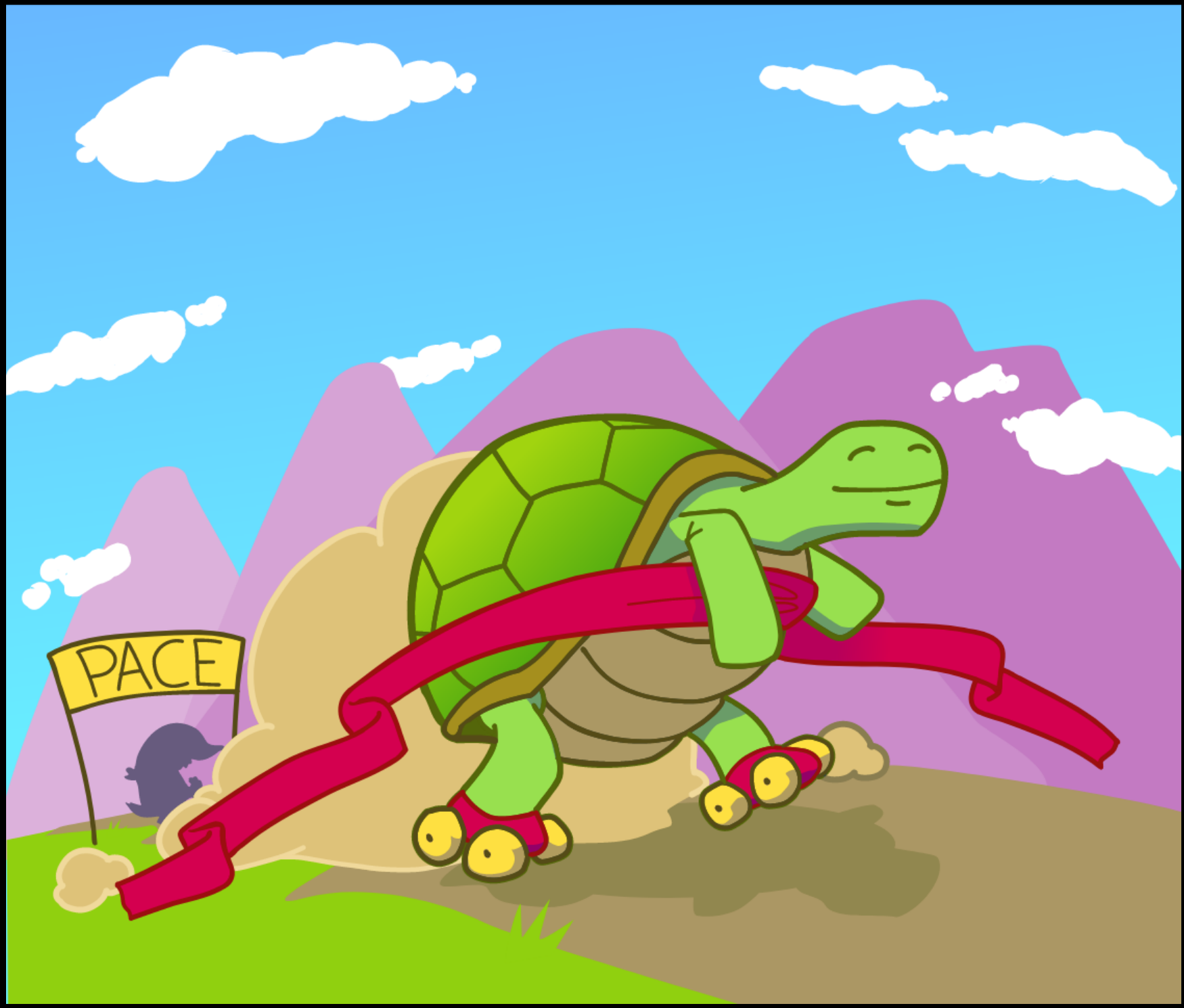
- $\text{chordalwidth}(G) = \min\{\omega(G') \mid G \subseteq G' \text{ und } G' \text{ ist chordal}\}$
- $\Rightarrow \text{tw}(G) = \text{chordalwidth}(G) - 1$

## Baumweite lösen:

- heuristisch: *min-degree* heuristic, *min-fill-in* heuristic
- heuristisch (fancy)
- heuristisch (exakt)

Paper aus PACE Challenge anschauen

Positive-Instance Driven Dynamic Programming for Treewidth (ESA'17, best paper); Heuristic computation of exact treewidth (SEA'22); https://github.com/mabseher/htd


PACE