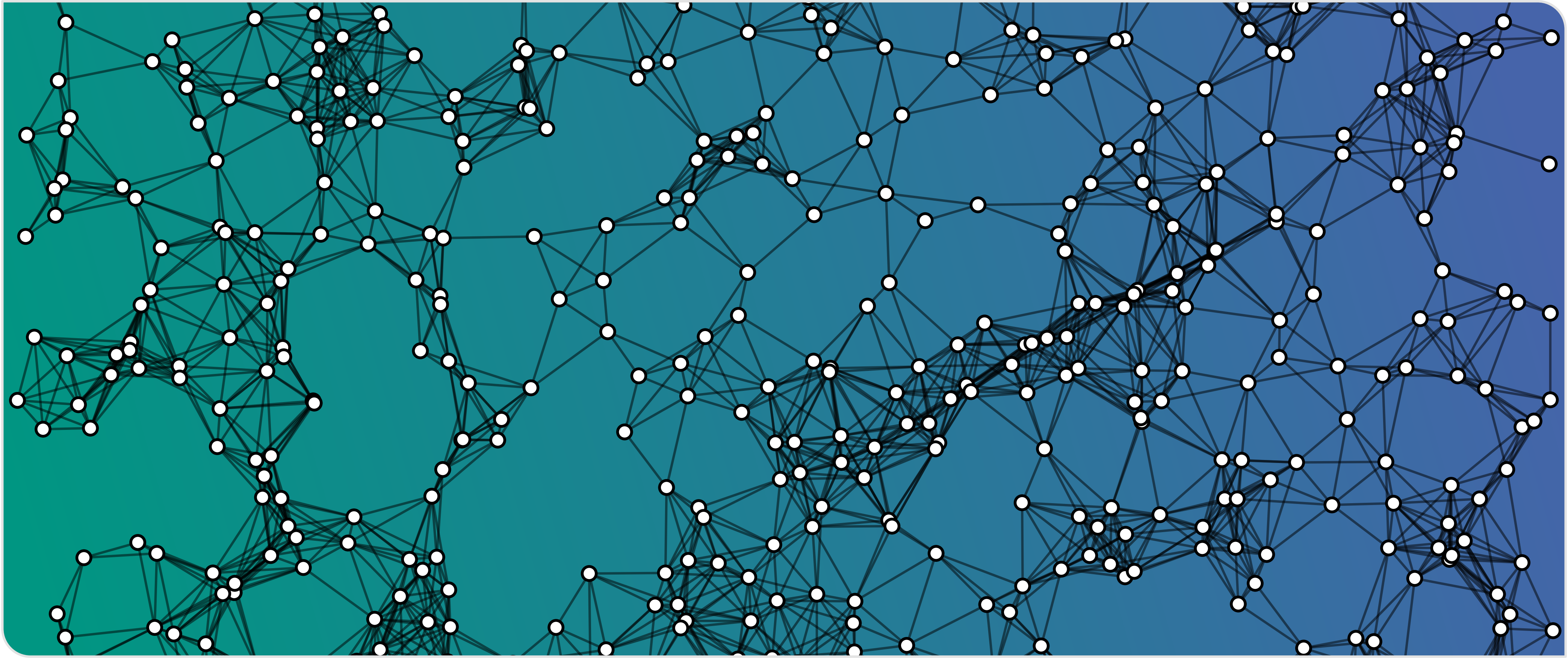
Karlsruher Institut für Technologie

Parametrisierte Algorithmen

Übung 3



Heute

Übungsblatt 2

Übungsblatt 3

Außerdem

- Lineare Programme & Lenstra
  - Aufgaben

Übungsblatt 2

d-Hitting Set

Gegeben $\{S_1, \ldots, S_n\}$ mit $|S_i| \leq d$, finde ein Hitting Set H mit $|H| \leq k$ und $H \cap S_i \neq \emptyset$ für alle S_i

Gib einen Algorithmus mit Laufzeit $(d - 0,658)^k \cdot n^{O(1)}$ an

Übungsblatt 2

d-Hitting Set

Gegeben $\{S_1, \ldots, S_n\}$ mit $|S_i| \leq d$, finde ein Hitting Set H mit $|H| \leq k$ und $H \cap S_i \neq \emptyset$ für alle S_i

Gib einen Algorithmus mit Laufzeit $(d - 0,658)^k \cdot n^{O(1)}$ an

- **IV**: $T_{d-\text{Hitting}}(n, k)$ ist in $(d - 0,658)^k \cdot n^{c+2d}$ lösbar

Übungsblatt 2

d-Hitting Set
Gegeben $\{S_1, \ldots, S_n\}$ mit $|S_i| \leq d$, finde ein Hitting Set H mit $|H| \leq k$ und $H \cap S_i \neq \emptyset$ für alle S_i

Gib einen Algorithmus mit Laufzeit $(d - 0,658)^k \cdot n^{O(1)}$ an

- **IV**: $T_{d-\text{Hitting}}(n, k)$ ist in $(d - 0,658)^k \cdot n^{c+2d}$ lösbar
- **IA**: $d = 2$ ist Vertex Cover aus der Vorlesung

Übungsblatt 2

d-Hitting Set
Gegeben $\{S_1, \ldots, S_n\}$ mit $|S_i| \leq d$, finde ein Hitting Set H mit $|H| \leq k$ und $H \cap S_i \neq \emptyset$ für alle S_i

Gib einen Algorithmus mit Laufzeit $(d - 0,658)^k \cdot n^{O(1)}$ an

- **IV**: $T_{d-\text{Hitting}}(n, k)$ ist in $(d - 0,658)^k \cdot n^{c+2d}$ lösbar
- **IA**: $d = 2$ ist Vertex Cover aus der Vorlesung

- $T^{\text{Disjoint}}_{d+1-\text{Hitting}}(n, k)$

$T^{\text{Disjoint}}_{d+1-\text{Hitting}}(n, k)$: gegeben Lösung X der Größe $k + 1$, gibt es Lösung H der Größe k mit $H \cap X = \emptyset$

Übungsblatt 2

d-Hitting Set
Gegeben $\{S_1, \ldots, S_n\}$ mit $|S_i| \leq d$, finde ein Hitting Set H mit $|H| \leq k$ und $H \cap S_i \neq \emptyset$ für alle S_i

Gib einen Algorithmus mit Laufzeit $(d - 0,658)^k \cdot n^{O(1)}$ an

- **IV**: $T_{d-\text{Hitting}}(n, k)$ ist in $(d - 0,658)^k \cdot n^{c+2d}$ lösbar
- **IA**: $d = 2$ ist Vertex Cover aus der Vorlesung

$T^{\text{Disjoint}}_{d+1-\text{Hitting}}(n, k)$: gegeben Lösung X der Größe $k + 1$, gibt es Lösung H der Größe k mit $H \cap X = \emptyset$

- $T^{\text{Disjoint}}_{d+1-\text{Hitting}}(n, k)$
  - Entspricht einer $T_{d-\text{Hitting}}(n', k)$ Instanz
  - Nach Induktions Voraussetzung in $(d - 0,658)^k \cdot n^{c+2d}$ lösbar

Übungsblatt 2

d-Hitting Set
Gegeben $\{S_1, \ldots, S_n\}$ mit $|S_i| \leq d$, finde ein Hitting Set H mit $|H| \leq k$ und $H \cap S_i \neq \emptyset$ für alle S_i

Gib einen Algorithmus mit Laufzeit $(d - 0,658)^k \cdot n^{O(1)}$ an

- **IV**: $T_{d-\text{Hitting}}(n, k)$ ist in $(d - 0,658)^k \cdot n^{c+2d}$ lösbar
- **IA**: $d = 2$ ist Vertex Cover aus der Vorlesung

$T^{\text{Disjoint}}_{d+1-\text{Hitting}}(n, k)$: gegeben Lösung X der Größe $k+1$, gibt es Lösung H der Größe k mit $H \cap X = \emptyset$

- $T^{\text{Disjoint}}_{d+1-\text{Hitting}}(n, k)$
  - Entspricht einer $T_{d-\text{Hitting}}(n', k)$ Instanz
  - Nach Induktions Voraussetzung in $(d - 0,658)^k \cdot n^{c+2d}$ lösbar

Aus Vorlesung
Disjoint in $\alpha^k \cdot n^c \Rightarrow$ Ursprünglich in $k(1+\alpha)^k \cdot n^{c+1}$

- $T_{d+1-\text{Hitting}}(n, k)$

Übungsblatt 2

d-Hitting Set
Gegeben $\{S_1, \dots, S_n\}$ mit $|S_i| \leq d$, finde ein Hitting Set H mit $|H| \leq k$ und $H \cap S_i \neq \emptyset$ für alle S_i

Gib einen Algorithmus mit Laufzeit $(d - 0,658)^k \cdot n^{O(1)}$ an

- **IV**: $T_{d-\text{Hitting}}(n, k)$ ist in $(d - 0,658)^k \cdot n^{c+2d}$ lösbar
- **IA**: $d = 2$ ist Vertex Cover aus der Vorlesung

$T^{\text{Disjoint}}_{d+1-\text{Hitting}}(n, k)$: gegeben Lösung X der Größe $k+1$, gibt es Lösung H der Größe k mit $H \cap X = \emptyset$

- $T^{\text{Disjoint}}_{d+1-\text{Hitting}}(n, k)$
  - Entspricht einer $T_{d-\text{Hitting}}(n', k)$ Instanz
  - Nach Induktions Voraussetzung in $(d - 0,658)^k \cdot n^{c+2d}$ lösbar

Aus Vorlesung
Disjoint in $\alpha^k \cdot n^c \Rightarrow$ Ursprünglich in $k(1+\alpha)^k \cdot n^{c+1}$

- $T_{d+1-\text{Hitting}}(n, k)$
  - in $k(1 + d - 0,658)^k \cdot n^{c+2d+1} \leq (1 + d - 0,658)^k \cdot n^{c+2(d+1)}$ lösbar

Übungsblatt 2

Vertex Cover

Implementiere Suchbaum für Vertex Cover

Übungsblatt 2

Vertex Cover

Implementiere Suchbaum für Vertex Cover

Suchen erst nach einem VC der Größe 1, dann $2, 3, \ldots$

Übungsblatt 2

Vertex Cover

Implementiere Suchbaum für Vertex Cover

Suchen erst nach einem VC der Größe 1, dann $2, 3, \ldots$

Reduktionsregeln

- Grad 1 Knoten
- Knoten mit Grad $> k$
- Dominanzregel
- Kern größer als k^2

Übungsblatt 2

Vertex Cover
Implementiere Suchbaum für Vertex Cover

Suchen erst nach einem VC der Größe 1, dann $2, 3, \ldots$

Reduktionsregeln

- Grad 1 Knoten
- Knoten mit Grad $> k$
- Dominanzregel
- Kern größer als k^2

Verzweigungsregeln

- Knoten vom Grad 2 oder 3
- Knoten von größerem Grad

Übungsblatt 2

Vertex Cover
Implementiere Suchbaum für Vertex Cover

Suchen erst nach einem VC der Größe 1, dann $2, 3, \ldots$

Reihenfolge der Regeln hat großen Einfluss auf die Laufzeit

Reduktionsregeln

- Grad 1 Knoten
- Knoten mit Grad $> k$
- Dominanzregel
- Kern größer als k^2

Verzweigungsregeln

- Knoten vom Grad 2 oder 3
- Knoten von größerem Grad

Übungsblatt 2

Vertex Cover

Implementiere Suchbaum für Vertex Cover

Suchen erst nach einem VC der Größe 1, dann $2, 3, \ldots$

Reihenfolge der Regeln hat großen Einfluss auf die Laufzeit

Regeln mit dem kleinsten Verzweigungsvektor zu erst

Reduktionsregeln

- Grad 1 Knoten
- Knoten mit Grad $> k$
- Dominanzregel
- Kern größer als k^2

Verzweigungsregeln

- Knoten vom Grad 2 oder 3
- Knoten von größerem Grad

Übungsblatt 2

Vertex Cover

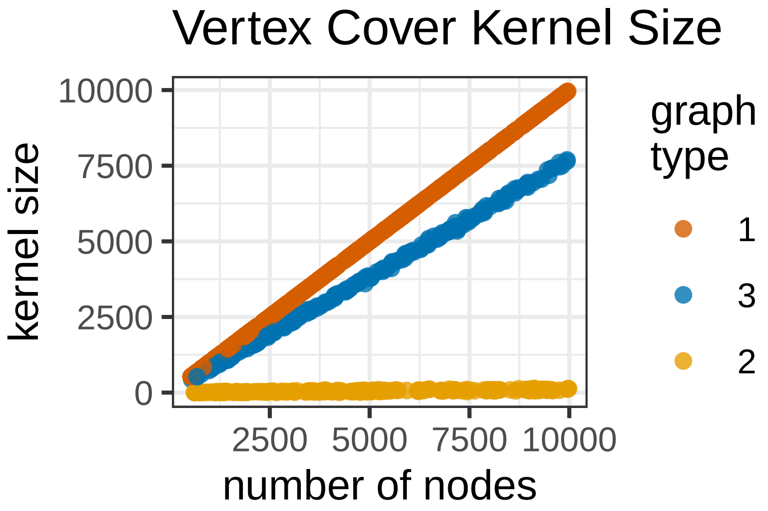
Implementiere Suchbaum für Vertex Cover

Suchen erst nach einem VC der Größe 1, dann $2, 3, \ldots$

Reihenfolge der Regeln hat großen Einfluss auf die Laufzeit

Regeln mit dem kleinsten Verzweigungsvektor zu erst

Der Graph aber auch...



Reduktionsregeln

- Grad 1 Knoten
- Knoten mit Grad $> k$
- Dominanzregel
- Kern größer als k^2

Verzweigungsregeln

- Knoten vom Grad 2 oder 3
- Knoten von größerem Grad

Übungsblatt 2

Vertex Cover
Implementiere Suchbaum für Vertex Cover

Suchen erst nach einem VC der Größe 1, dann $2, 3, \ldots$

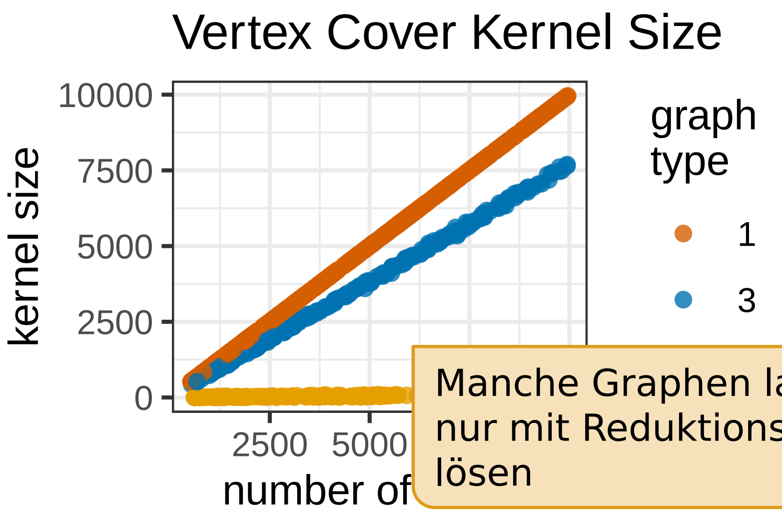
Reduktionsregeln
- Grad 1 Knoten
- Knoten mit Grad $> k$
- Dominanzregel
- Kern größer als k^2

Verzweigungsregeln
- Knoten vom Grad 2 oder 3
- Knoten von größerem Grad

Reihenfolge der Regeln hat großen Einfluss auf die Laufzeit

Regeln mit dem kleinsten Verzweigungsvektor zu erst

Der Graph aber auch...



Manche Graphen lassen sich nur mit Reduktionsregeln lösen

Übungsblatt 3

Übungsblatt 3

Odd Subgraph (Kernbildung)

- Gibt es einen Subgraphen mit k Kanten, indem jeder Knoten einen ungeraden Grad hat?

Übungsblatt 3

Odd Subgraph (Kernbildung)

- Gibt es einen Subgraphen mit k Kanten, indem jeder Knoten einen ungeraden Grad hat?

Edge Clique Cover (Kernbildung)

- Gibt es k Cliquen, die den ganzen Graphen überdecken?

Übungsblatt 3

Odd Subgraph (Kernbildung)

- Gibt es einen Subgraphen mit k Kanten, indem jeder Knoten einen ungeraden Grad hat?

Edge Clique Cover (Kernbildung)

- Gibt es k Cliquen, die den ganzen Graphen überdecken?

Anwendungen von Lenstra (ILPs)

- Gibt es eine Teilmultimenge der Größe genau b? (Parameter ist Anzahl an verschiedenen Zahlen)
- Können n Jobs auf m Maschienen zugewiesen werden, sodass alle Maschinen in maximal k Zeit fertig sind?

Übungsblatt 3

Odd Subgraph (Kernbildung)

- Gibt es einen Subgraphen mit k Kanten, indem jeder Knoten einen ungeraden Grad hat?

Edge Clique Cover (Kernbildung)

- Gibt es k Cliquen, die den ganzen Graphen überdecken?

Anwendungen von Lenstra (ILPs)

- Gibt es eine Teilmultimenge der Größe genau b? (Parameter ist Anzahl an verschiedenen Zahlen)
- Können n Jobs auf m Maschienen zugewiesen werden, sodass alle Maschinen in maximal k Zeit fertig sind?

Closest String (Suchbaum)

- Gibt es einen String, der zu allen anderen Distanz maximal k hat?
  - Nicht verwechseln mit der Variante, die wir gleich kennenlernen

Übungsblatt 3

Odd Subgraph (Kernbildung)

- Gibt es einen Subgraphen mit k Kanten, indem jeder Knoten einen ungeraden Grad hat?

Edge Clique Cover (Kernbildung)

- Gibt es k Cliquen, die den ganzen Graphen überdecken?

Anwendungen von Lenstra (ILPs)

- Gibt es eine Teilmultimenge der Größe genau b? (Parameter ist Anzahl an verschiedenen Zahlen)
- Können n Jobs auf m Maschienen zugewiesen werden, sodass alle Maschinen in maximal k Zeit fertig sind?

Closest String (Suchbaum)

- Gibt es einen String, der zu allen anderen Distanz maximal k hat?
  - Nicht verwechseln mit der Variante, die wir gleich kennenlernen

Kontaktvermeidung (Kernbildung)

- Lösche k Knoten aus dem Graphen, sodass er anschließend Maximalgrad d hat. (Parameter ist $k+d$)

Lenstras Theorem

Theorem (ohne Beweis)
Für $A \in \mathbb{Z}^{m \times n}$ und $b \in \mathbb{Z}^m$ kann die Frage, ob es ein $x \in \mathbb{N}^n$ mit $Ax \leq b$ gibt in $O(n^{2,5n}|A, b|)$ entschieden werden, wobei $|A, b|$ die Länge der Binärkodierung für die Instanz bezeichnet.

Lenstras Theorem

Theorem (ohne Beweis)
Für $A \in \mathbb{Z}^{m \times n}$ und $b \in \mathbb{Z}^m$ kann die Frage, ob es ein $x \in \mathbb{N}^n$ mit $Ax \leq b$ gibt in $O(n^{2,5n}|A, b|)$ entschieden werden, wobei $|A, b|$ die Länge der Binärkodierung für die Instanz bezeichnet.

Folgerungen

- ILP (als Entscheidungsproblem) mit Parameter n = Anzahl der Variablen, auch *Dimension* genannt, ist in FPT

Lenstras Theorem

Theorem (ohne Beweis)
Für $A \in \mathbb{Z}^{m \times n}$ und $b \in \mathbb{Z}^m$ kann die Frage, ob es ein $x \in \mathbb{N}^n$ mit $Ax \leq b$ gibt in $O(n^{2,5n}|A,b|)$ entschieden werden, wobei $|A,b|$ die Länge der Binärkodierung für die Instanz bezeichnet.

Folgerungen

- ILP (als Entscheidungsproblem) mit Parameter n = Anzahl der Variablen, auch *Dimension* genannt, ist in FPT
- Anzahl der Ungleichungen geht nur polynomiell in Laufzeit ein
- Größe der Zahlen geht nur logarithmisch in Laufzeit ein

Lenstras Theorem

Theorem (ohne Beweis)
Für $A \in \mathbb{Z}^{m \times n}$ und $b \in \mathbb{Z}^m$ kann die Frage, ob es ein $x \in \mathbb{N}^n$ mit $Ax \leq b$ gibt in $O(n^{2,5n}|A, b|)$ entschieden werden, wobei $|A, b|$ die Länge der Binärkodierung für die Instanz bezeichnet.

Folgerungen

- ILP (als Entscheidungsproblem) mit Parameter n = Anzahl der Variablen, auch *Dimension* genannt, ist in FPT
- Anzahl der Ungleichungen geht nur polynomiell in Laufzeit ein
- Größe der Zahlen geht nur logarithmisch in Laufzeit ein

Metatheorem
Ein parametrisiertes Problem mit Parameter k, dass sich als ILP mit $f(k)$ vielen Variablen darstellen lässt, ist in FPT.

Beispiel: Closest String

Problem: Closest String (*k* wird unser Parameter sein)

Gegeben k Strings $s_1, \dots, s_k \in \Sigma^n$ und $D \in \mathbb{N}$. Gibt es einen String $s \in \Sigma^n$, der von jedem s_i Hamming Distanz maximal D hat?

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F

Beispiel: Closest String

Problem: Closest String (k wird unser Parameter sein)
Gegeben k Strings $s_1, \ldots, s_k \in \Sigma^n$ und $D \in \mathbb{N}$. Gibt es einen String $s \in \Sigma^n$, der von jedem s_i Hamming Distanz maximal D hat?

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F
s	B	B	E	F	G	D	C	G	B	B	C

Beispiel: Closest String

Problem: Closest String (k wird unser Parameter sein)
Gegeben k Strings $s_1, \dots, s_k \in \Sigma^n$ und $D \in \mathbb{N}$. Gibt es einen String $s \in \Sigma^n$, der von jedem s_i Hamming Distanz maximal D hat?

Beobachtung

- nur (Un)gleichheit innerhalb jeder Spalte relevant
- es gibt äquivalente Instanz mit $\Sigma = [k]$

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F

Beispiel: Closest String

Problem: Closest String (k wird unser Parameter sein)
Gegeben k Strings $s_1, \ldots, s_k \in \Sigma^n$ und $D \in \mathbb{N}$. Gibt es einen String $s \in \Sigma^n$, der von jedem s_i Hamming Distanz maximal D hat?

Beobachtung

- nur (Un)gleichheit innerhalb jeder Spalte relevant
- es gibt äquivalente Instanz mit $\Sigma = [k]$

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F

↓

s_1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	1	2	2	1	1	2	1	2	2	2
s_3	2	2	3	1	2	2	1	1	2	1	3

Beispiel: Closest String

Problem: Closest String (*k* wird unser Parameter sein)
Gegeben k Strings $s_1, \ldots, s_k \in \Sigma^n$ und $D \in \mathbb{N}$. Gibt es einen String $s \in \Sigma^n$, der von jedem s_i Hamming Distanz maximal D hat?

Beobachtung

- nur (Un)gleichheit innerhalb jeder Spalte relevant
- es gibt äquivalente Instanz mit $\Sigma = [k]$
- Anzahl verschiedener Spalten nur von k abhängig ($\leq k!$)

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F

s_1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	1	2	2	1	1	2	1	2	2	2
s_3	2	2	3	1	2	2	1	1	2	1	3

Beispiel: Closest String

Problem: Closest String (*k* wird unser Parameter sein)
Gegeben k Strings $s_1, \dots, s_k \in \Sigma^n$ und $D \in \mathbb{N}$. Gibt es einen String $s \in \Sigma^n$, der von jedem s_i Hamming Distanz maximal D hat?

Beobachtung

- nur (Un)gleichheit innerhalb jeder Spalte relevant
- es gibt äquivalente Instanz mit $\Sigma = [k]$
- Anzahl verschiedener Spalten nur von k abhängig ($\leq k!$)

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F

s_1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	1	2	2	1	1	2	1	2	2	2
s_3	2	2	3	1	2	2	1	1	2	1	3

Formulierung als ILP

- reduziere Lösung auf: Wie oft wurde für den Spalten-Typ t der Wert v gewählt?

Beispiel: CLOSEST STRING

Problem: CLOSEST STRING (k wird unser Parameter sein)
Gegeben k Strings $s_1, \dots, s_k \in \Sigma^n$ und $D \in \mathbb{N}$. Gibt es einen String $s \in \Sigma^n$, der von jedem s_i Hamming Distanz maximal D hat?

Beobachtung

- nur (Un)gleichheit innerhalb jeder Spalte relevant
- es gibt äquivalente Instanz mit $\Sigma = [k]$
- Anzahl verschiedener Spalten nur von k abhängig ($\leq k!$)

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F

s_1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	1	2	2	1	1	2	1	2	2	2
s_3	2	2	3	1	2	2	1	1	2	1	3
s	2	1	3	1	2	1	2	1	2	1	2

Formulierung als ILP

- reduziere Lösung auf: Wie oft wurde für den Spalten-Typ t der Wert v gewählt?

Beispiel: Closest String

Problem: Closest String (*k wird unser Parameter sein*)
Gegeben k Strings $s_1, \dots, s_k \in \Sigma^n$ und $D \in \mathbb{N}$. Gibt es einen String $s \in \Sigma^n$, der von jedem s_i Hamming Distanz maximal D hat?

Beobachtung

- nur (Un)gleichheit innerhalb jeder Spalte relevant
- es gibt äquivalente Instanz mit $\Sigma = [k]$
- Anzahl verschiedener Spalten nur von k abhängig ($\leq k!$)

Formulierung als ILP

- reduziere Lösung auf: Wie oft wurde für den Spalten-Typ t der Wert v gewählt?

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F

s_1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	1	2	2	1	1	2	1	2	2	2
s_3	2	2	3	1	2	2	1	1	2	1	3
s	2	1	3	1	2	1	2	1	2	1	2

0×1	2×2	0×3
2×1	1×2	0×3
0×1	1×2	1×3
2×1	1×2	0×3
1×1	0×2	0×3

Beispiel: Closest String

Problem: Closest String (*k* wird unser Parameter sein)
Gegeben k Strings $s_1, \dots, s_k \in \Sigma^n$ und $D \in \mathbb{N}$. Gibt es einen String $s \in \Sigma^n$, der von jedem s_i Hamming Distanz maximal D hat?

Beobachtung

- nur (Un)gleichheit innerhalb jeder Spalte relevant
- es gibt äquivalente Instanz mit $\Sigma = [k]$
- Anzahl verschiedener Spalten nur von k abhängig ($\leq k!$)

Formulierung als ILP

- reduziere Lösung auf: Wie oft wurde für den Spalten-Typ t der Wert v gewählt? $\rightarrow$ Variable $x_{t,v}$

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F

$\downarrow$

s_1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	1	2	2	1	1	2	1	2	2	2
s_3	2	2	3	1	2	2	1	1	2	1	3
s	2	1	3	1	2	1	2	1	2	1	2

0×1	2×2	0×3
2×1	1×2	0×3
0×1	1×2	1×3
2×1	1×2	0×3
1×1	0×2	0×3

Beispiel: Closest String

Problem: Closest String (k wird unser Parameter sein)
Gegeben k Strings $s_1, \dots, s_k \in \Sigma^n$ und $D \in \mathbb{N}$. Gibt es einen String $s \in \Sigma^n$, der von jedem s_i Hamming Distanz maximal D hat?

Beobachtung

- nur (Un)gleichheit innerhalb jeder Spalte relevant
- es gibt äquivalente Instanz mit $\Sigma = [k]$
- Anzahl verschiedener Spalten nur von k abhängig ($\leq k!$)

Formulierung als ILP

- reduziere Lösung auf: Wie oft wurde für den Spalten-Typ t der Wert v gewählt? $\rightarrow$ Variable $x_{t,v}$
- wähle ein Zeichen für jede Spalte von Typ t:

$$\sum_{v \in \Sigma} x_{t,v} = \text{Anzahl Spalten von Typ } t$$

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F

s_1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	1	2	2	1	1	2	1	2	2	2
s_3	2	2	3	1	2	2	1	1	2	1	3
s	2	1	3	1	2	1	2	1	2	1	2

0×1	2×2	0×3
2×1	1×2	0×3
0×1	1×2	1×3
2×1	1×2	0×3
1×1	0×2	0×3

Beispiel: Closest String

Problem: Closest String (*k* wird unser Parameter sein)
Gegeben k Strings $s_1, \ldots, s_k \in \Sigma^n$ und $D \in \mathbb{N}$. Gibt es einen String $s \in \Sigma^n$, der von jedem s_i Hamming Distanz maximal D hat?

Beobachtung

- nur (Un)gleichheit innerhalb jeder Spalte relevant
- es gibt äquivalente Instanz mit $\Sigma = [k]$
- Anzahl verschiedener Spalten nur von k abhängig ($\leq k!$)

Formulierung als ILP

- reduziere Lösung auf: Wie oft wurde für den Spalten-Typ t der Wert v gewählt? $\rightarrow$ Variable $x_{t,v}$
- wähle ein Zeichen für jede Spalte von Typ t:

$$\sum_{v \in \Sigma} x_{t,v} = \text{Anzahl Spalten von Typ } t$$

- Distanz $\leq D$ für jedes s_i: $\displaystyle\sum_{t \in \text{Typen}} \sum_{\substack{v \in \Sigma \\ v \neq t[i]}} x_{t,v} \leq D$

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F

$\downarrow$

s_1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	1	2	2	1	1	2	1	2	2	2
s_3	2	2	3	1	2	2	1	1	2	1	3
s	2	1	3	1	2	1	2	1	2	1	2

0×1	2×2	0×3
2×1	1×2	0×3
0×1	1×2	1×3
2×1	1×2	0×3
1×1	0×2	0×3

Beispiel: Closest String

Problem: Closest String (*k* wird unser Parameter sein)
Gegeben k Strings $s_1, \dots, s_k \in \Sigma^n$ und $D \in \mathbb{N}$. Gibt es einen String $s \in \Sigma^n$, der von jedem s_i Hamming Distanz maximal D hat?

Beobachtung

- nur (Un)gleichheit innerhalb jeder Spalte relevant
- es gibt äquivalente Instanz mit $\Sigma = [k]$
- Anzahl verschiedener Spalten nur von k abhängig ($\leq k!$)

Formulierung als ILP

- reduziere Lösung auf: Wie oft wurde für den Spalten-Typ t der Wert v gewählt? $\to$ Variable $x_{t,v}$
- wähle ein Zeichen für jede Spalte von Typ t:

$$\sum_{v \in \Sigma} x_{t,v} = \text{Anzahl Spalten von Typ } t$$

- Distanz $\leq D$ für jedes s_i: $\sum_{t \in \text{Typen}} \sum_{\substack{v \in \Sigma \\ v \neq t[i]}} x_{t,v} \leq D$
- $\to$ ILP mit „nur“ $k! \cdot k$ Variablen

s_1	A	B	C	F	G	D	C	G	E	B	A
s_2	B	B	D	A	G	D	B	G	B	E	C
s_3	B	A	E	F	C	A	C	G	B	B	F

$\downarrow$

s_1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	1	2	2	1	1	2	1	2	2	2
s_3	2	2	3	1	2	2	1	1	2	1	3
s	2	1	3	1	2	1	2	1	2	1	2

0×1	2×2	0×3
2×1	1×2	0×3
0×1	1×2	1×3
2×1	1×2	0×3
1×1	0×2	0×3

Aufgaben

Wie können die folgenden Probleme mit (I)LPs gelöst werden?

- HITTING-SET (einfach)
- HAMILTONKREIS, sowie TSP
- MINIMUM LINEAR ARRANGEMENT: Given a Graph $G = (V, E)$ find an ordering $\sigma : V \to \{1, \dots, n\}$ of the vertices that minimizes $\sum_{\{u,v\} \in E} |\sigma(u) - \sigma(v)|$.

Further reading: Integer programming in parameterized complexity: Five miniatures

https://doi.org/10.1016/j.disopt.2020.100596

Integer programming in parameterized complexity: Five miniatures☆

Tomáš Gavenčiak[a], Martin Koutecký[a,1], Dušan Knop[b,*,2]

[a] Computer Science Institute of Charles University, Faculty of Mathematics and Physics, Charles University, Malostranské náměstí 25, Praha, Czech Republic
[b] Department of Theoretical Computer Science, Faculty of Information Technology, Czech Technical University in Prague, Thákurova 9, Praha, Czech Republic

ARTICLE INFO

Article history:
Received 31 December 2018
Received in revised form 14 May 2020
Accepted 25 June 2020
Available online 20 July 2020

Keywords:
Graph coloring
Sum-coloring
Parameterized complexity

ABSTRACT

Powerful results from the theory of integer programming have recently led to substantial advances in parameterized complexity. However, our perception is that, except for Lenstra's algorithm for solving integer linear programming in fixed dimension, there is still little understanding in the parameterized complexity community of the strengths and limitations of the available tools. This is understandable: it is often difficult to infer exact runtimes or even the distinction between FPT and XP algorithms, and some knowledge is simply unwritten folklore in a different community. We wish to make a step in remedying this situation.

To that end, we first provide an easy to navigate quick reference guide of integer programming algorithms from the perspective of parameterized complexity. Then, we show their applications in three case studies, obtaining FPT algorithms with runtime $f(k)\text{poly}(n)$. We focus on:

- *Modeling*: since the algorithmic results follow by applying existing algorithms to new models, we shift the focus from the complexity result to the modeling result, highlighting common patterns and tricks which are used.
- *Optimality program:* after giving an FPT algorithm, we are interested in reducing the dependence on the parameter; we show which algorithms and tricks are often useful for speed-ups.
- *Minding the* $\text{poly}(n)$: reducing $f(k)$ often has the unintended consequence of increasing $\text{poly}(n)$; so we highlight the common trade-offs and show how to get the best of both worlds.

Specifically, we consider graphs of bounded neighborhood diversity which are in a sense the simplest of dense graphs, and we show several FPT algorithms for CAPACITATED DOMINATING SET, SUM COLORING, MAX-q-CUT, and certain other coloring problems by modeling them as convex programs in fixed dimension, n-fold integer programs, bounded dual treewidth programs, indefinite quadratic programs

☆ An extended abstract presenting preliminary results appeared in proceedings of IPEC 2018 [1] (Gavenčiak et al (2018)).
* Corresponding author.
E-mail addresses: gavento@kam.mff.cuni.cz (T. Gavenčiak), koutecky@iuuk.mff.cuni.cz (M. Koutecký), dusan.knop@fit.cvut.cz (D. Knop).

1 Partially supported by Charles University project UNCE/SCI/004, by the project 19-27871X of GA ČR, and by the Israel Science Foundation grant 308/18.
2 Supported by the OP VVV MEYS funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics".

https://doi.org/10.1016/j.disopt.2020.100596