Minimum Quartett Inconsistency $\in$ FPT

Viktor Engelmann

RWTH Aachen University

28. Januar 2008

Inhalt

- Einführung

Inhalt

- Einführung
- Polynomieller Algorithmus für Spezialfall

Inhalt

- Einführung
- Polynomieller Algorithmus für Spezialfall
- Hauptsatz

Inhalt

- Einführung
- Polynomieller Algorithmus für Spezialfall
- Hauptsatz
- Primitiver FPT-Algorithmus

Inhalt

- Einführung
- Polynomieller Algorithmus für Spezialfall
- Hauptsatz
- Primitiver FPT-Algorithmus
- Laufzeitverbesserungen

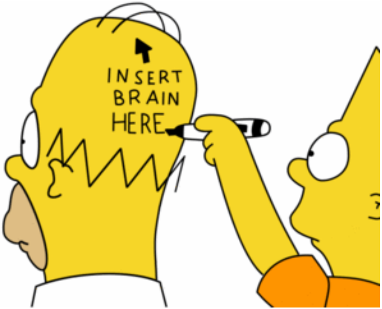
Inhalt

- Einführung
- Polynomieller Algorithmus für Spezialfall
- Hauptsatz
- Primitiver FPT-Algorithmus
- Laufzeitverbesserungen
- Verwandte Probleme

Inhalt

- Einführung
- Polynomieller Algorithmus für Spezialfall
- Hauptsatz
- Primitiver FPT-Algorithmus
- Laufzeitverbesserungen
- Verwandte Probleme
- Fragen?

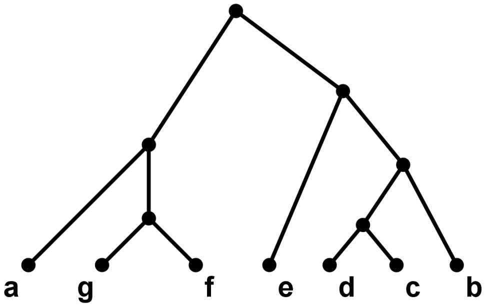
Einführung



Einführung

Gesucht

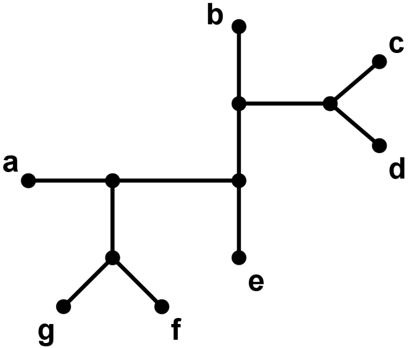
Ein bin. Evolutionsbaum



Einführung

Gesucht

Ein bin. Evolutionsbaum



Einführung

Gesucht

Ein bin. Evolutionsbaum

Gegeben

- Die Menge der Blätter **S** (z.B. heute lebende Spezies)

Einführung

Gesucht

Ein bin. Evolutionsbaum

Gegeben

- Die Menge der Blätter **S** (z.B. heute lebende Spezies)
- Eine vollständige Menge von *Topologien* **T** - Intuitiv etwa Verwandschaftsgrade der Spezies

Einführung

Gesucht

Ein bin. Evolutionsbaum

Gegeben

- Die Menge der Blätter **S** (z.B. heute lebende Spezies)
- Eine vollständige Menge von *Topologien* **T** - Intuitiv etwa Verwandschaftsgrade der Spezies
- Vollständig heisst: für **jede** Teilmenge von S mit Größe 4 (für jedes *Quartett*) eine Topologie - $|T| \in O(n^4)$

Einführung

Gesucht

Ein bin. Evolutionsbaum

Gegeben

- Die Menge der Blätter **S** (z.B. heute lebende Spezies)
- Eine <u>vollständige</u> Menge von *Topologien* **T** - Intuitiv etwa Verwandschaftsgrade der Spezies
- Vollständig heisst: für **jede** Teilmenge von S mit Größe 4 (für jedes *Quartett*) eine Topologie - $|T| \in O(n^4)$
- ***Problem:*** Topologien evtl. untereinander inkompatibel

Einführung

Gesucht

Ein bin. Evolutionsbaum

Gegeben

- Die Menge der Blätter **S** (z.B. heute lebende Spezies)
- Eine vollständige Menge von *Topologien* **T** - Intuitiv etwa Verwandschaftsgrade der Spezies
- Vollständig heisst: für **jede** Teilmenge von S mit Größe 4 (für jedes *Quartett*) eine Topologie - $|T| \in O(n^4)$
- ***Problem:*** Topologien evtl. untereinander inkompatibel $\rightarrow$ Dann passt kein Baum zu **allen** Topologien

Einführung

Gesucht

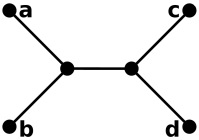
Ein bin. Evolutionsbaum, der min. viele Topologien verletzt

Gegeben

- Die Menge der Blätter **S** (z.B. heute lebende Spezies)
- Eine vollständige Menge von *Topologien* **T** - Intuitiv etwa Verwandschaftsgrade der Spezies
- Vollständig heisst: für **jede** Teilmenge von S mit Größe 4 (für jedes *Quartett*) eine Topologie - $|T| \in O(n^4)$
- ***Problem:*** Topologien evtl. untereinander inkompatibel $\rightarrow$ Dann passt kein Baum zu **allen** Topologien

Was sind Topologien?

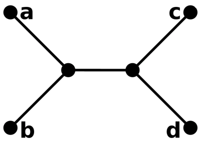
- Form:



(Im Text auch $[ab|cd]$)

Was sind Topologien?

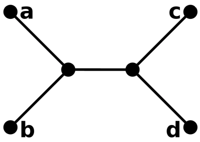
- Form:



  (Im Text auch $[ab|cd]$)
- Bedeutung: a, b und c, d liegen "nah" beieinander

Was sind Topologien?

- Form:



(Im Text auch $[ab|cd]$)
- Bedeutung: a, b und c, d liegen ”nah” beieinander
- Mögliche Topologien für das Quartett $\{a, b, c, d\}$?

Motivation

- Baum aus solchen Topologien eindeutig rekonstruierbar

Motivation

- Baum aus solchen Topologien <u>eindeutig</u> rekonstruierbar
- Aus kleineren Topologien <u>nicht</u> eindeutig rekonstruierbar

Motivation

- Baum aus solchen Topologien <u>eindeutig</u> rekonstruierbar
- Aus kleineren Topologien <u>nicht</u> eindeutig rekonstruierbar
- Solche Topologien können mit geringer Fehlerwahrscheinlichkeit aus Proteinsträngen gelesen werden

Präzisierung des Problems

- Menge von inkompatiblen Topologien =: *Konflikt* - Bsp.?

Präzisierung des Problems

- Menge von inkompatiblen Topologien =: *Konflikt* - Bsp.?
- **Problem (genauer)**: minimale Anzahl Topologien ändern, sodass T keine Konflikte mehr enthält

Einordnung in Komplexitätsklassen

- **NP**
  Nondet: *k* Topologien ändern, Baum generieren
  Konsistenz prüfen

Einordnung in Komplexitätsklassen

- **NP**
  Nondet: *k* Topologien ändern, Baum generieren
  Konsistenz prüfen
- **NPC**
  MQI $\geq_P$ Quartett Compatibility

Einordnung in Komplexitätsklassen

- **NP**
  Nondet: k Topologien ändern, Baum generieren
  Konsistenz prüfen
- **NPC**
  MQI $\geq_P$ Quartett Compatibility $\geq_P$ Betweenness

Einordnung in Komplexitätsklassen

- **NP**
  Nondet: k Topologien ändern, Baum generieren
  Konsistenz prüfen
- **NPC**
  MQI $\geq_P$ Quartett Compatibility $\geq_P$ Betweenness
  $\geq_P$ HyperColor-2

Einordnung in Komplexitätsklassen

- **NP**
  Nondet: k Topologien ändern, Baum generieren
  Konsistenz prüfen
- **NPC**
  MQI $\geq_P$ Quartett Compatibility $\geq_P$ Betweenness $\geq_P$ HyperColor-2 $\geq_P$ 3-SAT

Einordnung in Komplexitätsklassen

- **NP**
  Nondet: k Topologien ändern, Baum generieren
  Konsistenz prüfen
- **NPC**
  MQI $\geq_P$ Quartett Compatibility $\geq_P$ Betweenness $\geq_P$ HyperColor-2 $\geq_P$ 3-SAT
- **APX**
  Approximierbar

Einordnung in Komplexitätsklassen

- **NP**
  Nondet: k Topologien ändern, Baum generieren
  Konsistenz prüfen
- **NPC**
  MQI $\geq_P$ Quartett Compatibility $\geq_P$ Betweenness $\geq_P$ HyperColor-2 $\geq_P$ 3-SAT
- **APX**
  Approximierbar mit Güte n^2

Einordnung in Komplexitätsklassen

- **NP**
  Nondet: *k* Topologien ändern, Baum generieren
  Konsistenz prüfen
- **NPC**
  MQI $\geq_P$ Quartett Compatibility $\geq_P$ Betweenness $\geq_P$ HyperColor-2 $\geq_P$ 3-SAT
- **APX**
  Approximierbar mit Güte *n* - Wu, You, Lin (2006)

Einordnung in Komplexitätsklassen

- **NP**
  Nondet: k Topologien ändern, Baum generieren
  Konsistenz prüfen
- **NPC**
  MQI $\geq_P$ Quartett Compatibility $\geq_P$ Betweenness
  $\geq_P$ HyperColor-2 $\geq_P$ 3-SAT
- **APX**
  Approximierbar mit Güte n - Wu, You, Lin (2006)
- **FPT**
  Mit k = Anzahl verletzter Topologien $O(3,561^k \cdot n + n^4)$

Überblick

- Algorithmus von Gascuel & Berry (2000):
  Baum aus konsistenten Topologien in $O(n^4)$ konstruierbar

Überblick

- Algorithmus von Gascuel & Berry (2000):
  Baum aus konsistenten Topologien in $O(n^4)$ konstruierbar
- Topologien in Konflikten korrigieren

Überblick

- Algorithmus von Gascuel & Berry (2000):
  Baum aus konsistenten Topologien in $O(n^4)$ konstruierbar
- Topologien in Konflikten korrigieren
- Konflikte finden

Überblick

- Algorithmus von Gascuel & Berry (2000):
  Baum aus konsistenten Topologien in $O(n^4)$ konstruierbar
- Topologien in Konflikten korrigieren
- Konflikte finden
- Große Konflikte $\Leftrightarrow$ kleine Konflikte (Größe 3)

Überblick

- Algorithmus von Gascuel & Berry (2000):
  Baum aus konsistenten Topologien in $O(n^4)$ konstruierbar
- Topologien in Konflikten korrigieren
- Konflikte finden (Polynomiell)
- Große Konflikte $\Leftrightarrow$ kleine Konflikte (Größe 3)

Überblick

- Algorithmus von Gascuel & Berry (2000):
  Baum aus konsistenten Topologien in $O(n^4)$ konstruierbar
- Topologien in Konflikten korrigieren
- Konflikte finden (Polynomiell)
- Große Konflikte $\Leftrightarrow$ kleine Konflikte (Größe 3)
- ***Bounded-Searchtree-Algorithmus***

Überblick

- Algorithmus von Gascuel & Berry (2000):
  Baum aus konsistenten Topologien in $O(n^4)$ konstruierbar
- Topologien in Konflikten korrigieren
- Konflikte finden (Polynomiell)
- Große Konflikte $\Leftrightarrow$ kleine Konflikte (Größe 3)
- ***Bounded-Searchtree-Algorithmus***
  - Max. k Topologien ändern $\Rightarrow$ Höhe k

Überblick

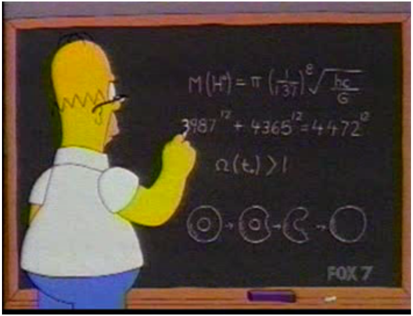
- Algorithmus von Gascuel & Berry (2000):
  Baum aus konsistenten Topologien in $O(n^4)$ konstruierbar
- Topologien in Konflikten korrigieren
- Konflikte finden (Polynomiell)
- Große Konflikte $\Leftrightarrow$ kleine Konflikte (Größe 3)
- ***Bounded-Searchtree-Algorithmus***
  - Max. k Topologien ändern $\Rightarrow$ Höhe k
  - Wenige Änderungen in einem Konflikt möglich

Überblick

- Algorithmus von Gascuel & Berry (2000):
  Baum aus konsistenten Topologien in $O(n^4)$ konstruierbar
- Topologien in Konflikten korrigieren
- Konflikte finden (Polynomiell)
- Große Konflikte $\Leftrightarrow$ kleine Konflikte (Größe 3)
- ***Bounded-Searchtree-Algorithmus***
  - Max. k Topologien ändern $\Rightarrow$ Höhe k
  - Wenige Änderungen in einem Konflikt möglich
  - $\Rightarrow$ beschränkte Verzweigung

Gascuel und Berry

Polynomieller Algorithmus für Spezialfall $k = 0$



Algorithmus von Gascuel und Berry (2000)

1. Starte mit einer bel. Topologie, füge die $n - 4$ anderen Spezies sukzessive ein:

Algorithmus von Gascuel und Berry (2000)

1. Starte mit einer bel. Topologie, füge die $n-4$ anderen Spezies sukzessive ein:
2. Schreibe ”Wegweiser” für jeden Knoten an die inzidenten Kanten (je eine bel. Spezies, die in dieser Richtung liegt)

Algorithmus von Gascuel und Berry (2000)

1. Starte mit einer bel. Topologie, füge die $n-4$ anderen Spezies sukzessive ein:
2. Schreibe ”Wegweiser” für jeden Knoten an die inzidenten Kanten (je eine bel. Spezies, die in dieser Richtung liegt)
3. Zum Einfügen der Spezies h starte an bel. innerem Knoten

Algorithmus von Gascuel und Berry (2000)

1. Starte mit einer bel. Topologie, füge die $n - 4$ anderen Spezies sukzessive ein:
2. Schreibe "Wegweiser" für jeden Knoten an die inzidenten Kanten (je eine bel. Spezies, die in dieser Richtung liegt)
3. Zum Einfügen der Spezies h starte an bel. innerem Knoten
4. An aktuellem Knoten betrachte die Wegweiser z.B. a, f, e und die Topologie t für $\{a, f, e, h\}$ (z.B. $t = [af|eh]$)

Algorithmus von Gascuel und Berry (2000)

1. Starte mit einer bel. Topologie, füge die $n-4$ anderen Spezies sukzessive ein:
2. Schreibe ”Wegweiser” für jeden Knoten an die inzidenten Kanten (je eine bel. Spezies, die in dieser Richtung liegt)
3. Zum Einfügen der Spezies h starte an bel. innerem Knoten
4. An aktuellem Knoten betrachte die Wegweiser z.B. a, f, e und die Topologie t für $\{a, f, e, h\}$ (z.B. $t = [af|eh]$)
5. Gehe über die Kante, an der der Wegweiser e steht

Algorithmus von Gascuel und Berry (2000)

1. Starte mit einer bel. Topologie, füge die $n - 4$ anderen Spezies sukzessive ein:
2. Schreibe "Wegweiser" für jeden Knoten an die inzidenten Kanten (je eine bel. Spezies, die in dieser Richtung liegt)
3. Zum Einfügen der Spezies h starte an bel. innerem Knoten
4. An aktuellem Knoten betrachte die Wegweiser z.B. a, f, e und die Topologie t für $\{a, f, e, h\}$ (z.B. $t = [af|eh]$)
5. Gehe über die Kante, an der der Wegweiser e steht
6. Wenn man ein Blatt erreicht oder umkehren muss, füge die Spezies an der letzen durchlaufenen Kante ein, stop sonst gehe zu 4

Algorithmus von Gascuel und Berry (2000)

Laufzeit

- **Einfügen einer Spezies:**
  nur innere Knoten werden passiert (max. 1 mal)
  es ex. max. n innere Knoten $\Rightarrow O(n)$

Algorithmus von Gascuel und Berry (2000)

Laufzeit

- **Einfügen einer Spezies:**
  nur innere Knoten werden passiert (max. 1 mal)
  es ex. max. n innere Knoten $\Rightarrow O(n)$
- Einfügen von $O(n)$ Spezies: $O(n^2)$

Algorithmus von Gascuel und Berry (2000)

Laufzeit

- **Einfügen einer Spezies:**
  nur innere Knoten werden passiert (max. 1 mal)
  es ex. max. n innere Knoten $\Rightarrow O(n)$
- Einfügen von $O(n)$ Spezies: $O(n^2)$
- $O(n^4)$ Topologien werden eingelesen also insges. $O(n^4)$

Hauptsatz

$\exists$ Große Konflikte $\Leftrightarrow$ $\exists$ Kleine Konflikte

Hauptsatz

Definition
Eine Menge von Topologien T heißt

Hauptsatz

Definition

Eine Menge von Topologien T heißt

- ***tree-like***, wenn ein Baum existiert, der alle $t \in T$ erfüllt und T vollständig ist

Hauptsatz

Definition

Eine Menge von Topologien T heißt

- ***tree-like***, wenn ein Baum existiert, der alle $t \in T$ erfüllt und T vollständig ist
- ***tree-consistent***, wenn sie Teilmenge einer *tree-like* Menge von Topologien ist

Hauptsatz

Definition

Eine Menge von Topologien T heißt

- ***tree-like***, wenn ein Baum existiert, der alle $t \in T$ erfüllt und T vollständig ist
- ***tree-consistent***, wenn sie Teilmenge einer *tree-like* Menge von Topologien ist

tree-like $\Rightarrow$ tree-consistent

Hauptsatz

Definition

Eine Menge von Topologien T heißt

- ***tree-like***, wenn ein Baum existiert, der alle $t \in T$ erfüllt und T vollständig ist
- ***tree-consistent***, wenn sie Teilmenge einer *tree-like* Menge von Topologien ist

$$\begin{array}{ccc} \text{tree-like} & \Rightarrow & \text{tree-consistent} \\ \Updownarrow & & \\ \forall a, b, c, d, e & & \\ [ab|cd] \to [ae|cd] \vee [ab|ce] & & \\ \text{(substitution-property)} & & \end{array}$$

Hauptsatz

Definition

Eine Menge von Topologien T heißt

- ***tree-like***, wenn ein Baum existiert, der alle $t \in T$ erfüllt und T vollständig ist
- ***tree-consistent***, wenn sie Teilmenge einer *tree-like* Menge von Topologien ist

$$\begin{array}{ccc} \text{tree-like} & \Rightarrow & \text{tree-consistent} \\ \Updownarrow & & \Downarrow \\ \forall a, b, c, d, e & & \forall t_1, t_2, t_3 \in T \\ [ab|cd] \rightarrow [ae|cd] \vee [ab|ce] & & \{t_1, t_2, t_3\} \\ \text{(substitution-property)} & & \text{tree-consistent} \end{array}$$

Hauptsatz

Definition

Eine Menge von Topologien T heißt

- ***tree-like***, wenn ein Baum existiert, der alle $t \in T$ erfüllt und T vollständig ist
- ***tree-consistent***, wenn sie Teilmenge einer *tree-like* Menge von Topologien ist

$$
\begin{array}{ccc}
\text{tree-like} & \Rightarrow & \text{tree-consistent} \\
\Updownarrow & & \Downarrow \\
\forall a, b, c, d, e & \Leftarrow & \forall t_1, t_2, t_3 \in T \\
[ab|cd] \rightarrow [ae|cd] \vee [ab|ce] & \text{für vollst } T & \{t_1, t_2, t_3\} \\
\text{(substitution-property)} & & \text{tree-consistent}
\end{array}
$$

Hauptsatz

Widerspruchsbeweis: Es gelte
$\neg\forall a, b, c, d, e : [ab|cd] \to [ab|ce] \vee [ae|cd]$

Hauptsatz

Widerspruchsbeweis: Es gelte
$\neg\forall a, b, c, d, e : [ab|cd] \rightarrow [ab|ce] \vee [ae|cd]$
$\Leftrightarrow \neg\forall a, b, c, d, e : \neg[ab|cd] \vee [ab|ce] \vee [ae|cd]$

Hauptsatz

Widerspruchsbeweis: Es gelte
$\neg\forall a, b, c, d, e : [ab|cd] \rightarrow [ab|ce] \vee [ae|cd]$
$\Leftrightarrow \neg\forall a, b, c, d, e : \neg[ab|cd] \vee [ab|ce] \vee [ae|cd]$
$\Leftrightarrow \exists a, b, c, d, e : [ab|cd] \wedge \neg[ab|ce] \wedge \neg[ae|cd]$

Hauptsatz

Widerspruchsbeweis: Es gelte
$\neg\forall a, b, c, d, e : [ab|cd] \rightarrow [ab|ce] \vee [ae|cd]$
$\Leftrightarrow \neg\forall a, b, c, d, e : \neg[ab|cd] \vee [ab|ce] \vee [ae|cd]$
$\Leftrightarrow \exists a, b, c, d, e : [ab|cd] \wedge \neg[ab|ce] \wedge \neg[ae|cd]$
$\Leftrightarrow \exists a, b, c, d, e : [ab|cd] \wedge ([ac|be] \vee [ae|bc]) \wedge ([ac|de] \vee [ad|ce])$
weil T vollständig

Hauptsatz

Widerspruchsbeweis: Es gelte

$\neg\forall a, b, c, d, e : [ab|cd] \rightarrow [ab|ce] \vee [ae|cd]$

$\Leftrightarrow \neg\forall a, b, c, d, e : \neg[ab|cd] \vee [ab|ce] \vee [ae|cd]$

$\Leftrightarrow \exists a, b, c, d, e : [ab|cd] \wedge \neg[ab|ce] \wedge \neg[ae|cd]$

$\Leftrightarrow \exists a, b, c, d, e : [ab|cd] \wedge ([ac|be] \vee [ae|bc]) \wedge ([ac|de] \vee [ad|ce])$

weil T vollständig

- $[ab|cd], [ac|be], [ac|de]$

Hauptsatz

Widerspruchsbeweis: Es gelte
$\neg\forall a, b, c, d, e : [ab|cd] \rightarrow [ab|ce] \vee [ae|cd]$
$\Leftrightarrow \neg\forall a, b, c, d, e : \neg[ab|cd] \vee [ab|ce] \vee [ae|cd]$
$\Leftrightarrow \exists a, b, c, d, e : [ab|cd] \wedge \neg[ab|ce] \wedge \neg[ae|cd]$
$\Leftrightarrow \exists a, b, c, d, e : [ab|cd] \wedge ([ac|be] \vee [ae|bc]) \wedge ([ac|de] \vee [ad|ce])$
weil T vollständig

- $[ab|cd], [ac|be], [ac|de]$ Inkonsistent

Hauptsatz

Widerspruchsbeweis: Es gelte
$\neg\forall a, b, c, d, e : [ab|cd] \rightarrow [ab|ce] \vee [ae|cd]$
$\Leftrightarrow \neg\forall a, b, c, d, e : \neg[ab|cd] \vee [ab|ce] \vee [ae|cd]$
$\Leftrightarrow \exists a, b, c, d, e : [ab|cd] \wedge \neg[ab|ce] \wedge \neg[ae|cd]$
$\Leftrightarrow \exists a, b, c, d, e : [ab|cd] \wedge ([ac|be] \vee [ae|bc]) \wedge ([ac|de] \vee [ad|ce])$
weil T vollständig

- $[ab|cd], [ac|be], [ac|de]$ Inkonsistent
- $[ab|cd], [ac|be], [ad|ce]$ Inkonsistent (analog)
- $[ab|cd], [ae|bc], [ac|de]$ Inkonsistent (analog)
- $[ab|cd], [ae|bc], [ad|ce]$ Inkonsistent (analog)

Hauptsatz

Widerspruchsbeweis: Es gelte
$\neg\forall a, b, c, d, e : [ab|cd] \rightarrow [ab|ce] \vee [ae|cd]$
$\Leftrightarrow \neg\forall a, b, c, d, e : \neg[ab|cd] \vee [ab|ce] \vee [ae|cd]$
$\Leftrightarrow \exists a, b, c, d, e : [ab|cd] \wedge \neg[ab|ce] \wedge \neg[ae|cd]$
$\Leftrightarrow \exists a, b, c, d, e : [ab|cd] \wedge ([ac|be] \vee [ae|bc]) \wedge ([ac|de] \vee [ad|ce])$
weil T vollständig

- $[ab|cd], [ac|be], [ac|de]$ Inkonsistent
- $[ab|cd], [ac|be], [ad|ce]$ Inkonsistent (analog)
- $[ab|cd], [ae|bc], [ac|de]$ Inkonsistent (analog)
- $[ab|cd], [ae|bc], [ad|ce]$ Inkonsistent (analog)

$\square$

Hauptsatz

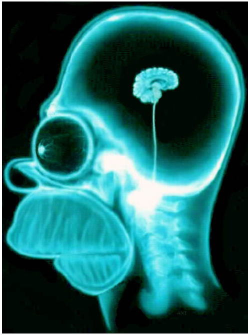
Definition

Eine Menge von Topologien T heißt

- ***tree-like***, wenn ein Baum existiert, der alle $t \in T$ erfüllt und T vollständig ist
- ***tree-consistent***, wenn sie Teilmenge einer *tree-like* Menge von Topologien ist

tree-like	$\Rightarrow$	tree-consistent
$\Updownarrow$		$\Downarrow$
$\forall a, b, c, d, e$	$\Leftarrow$	$\forall t_1, t_2, t_3 \in T$
$[ab\|cd] \rightarrow [ae\|cd] \vee [ab\|ce]$	für vollst T	$\{t_1, t_2, t_3\}$
(substitution-property)		tree-consistent

Primitiver FPT-Algorithmus



Primitiver FPT-Algorithmus

Suche nach Menge der Konflikte C:

Primitiver FPT-Algorithmus

Suche nach Menge der Konflikte C:

Es gibt $O(n^4)$ Quartetts und damit $O((n^4)^3) = O(n^{12})$ 3-elementige Mengen von Quartetts

Primitiver FPT-Algorithmus

Behebung der Konflikte:

Primitiver FPT-Algorithmus

Behebung der Konflikte:

- Wähle einen Konflikt $c := \{t_1, t_2, t_3\}$ aus C

Primitiver FPT-Algorithmus

Behebung der Konflikte:

- Wähle einen Konflikt $c := \{t_1, t_2, t_3\}$ aus C
- Ändere eine Topologie t_i aus c
  3 Topologien $\cdot$ je 2 mögl. Änderungen $\Rightarrow$ 6 Branches

Primitiver FPT-Algorithmus

<u>Behebung der Konflikte:</u>

- Wähle einen Konflikt $c := \{t_1, t_2, t_3\}$ aus C
- Ändere eine Topologie t_i aus c
  3 Topologien · je 2 mögl. Änderungen $\Rightarrow$ 6 Branches
- Passe C an die Änderung an

Primitiver FPT-Algorithmus

Behebung der Konflikte:

- Wähle einen Konflikt $c := \{t_1, t_2, t_3\}$ aus C
- Ändere eine Topologie t_i aus c
  3 Topologien $\cdot$ je 2 mögl. Änderungen $\Rightarrow$ 6 Branches
- Passe C an die Änderung an
  z.B. generiere C neu $O(n^{12})$

Primitiver FPT-Algorithmus

Behebung der Konflikte:

- Wähle einen Konflikt $c := \{t_1, t_2, t_3\}$ aus C
- Ändere eine Topologie t_i aus c
  3 Topologien $\cdot$ je 2 mögl. Änderungen $\Rightarrow$ 6 Branches
- Passe C an die Änderung an
  z.B. generiere C neu $O(n^{12})$
- Führe Rekursion mit neuem C und $k - 1$ aus

Primitiver FPT-Algorithmus

Behebung der Konflikte:

- Wähle einen Konflikt $c := \{t_1, t_2, t_3\}$ aus C
- Ändere eine Topologie t_i aus c
  3 Topologien $\cdot$ je 2 mögl. Änderungen $\Rightarrow$ 6 Branches
- Passe C an die Änderung an
  z.B. generiere C neu $O(n^{12})$
- Führe Rekursion mit neuem C und $k-1$ aus

Suchbaum mit Höhe k, Verzweigung 6
Vor jedem Rekursionsaufruf $O(n^{12}) \Rightarrow$ insges. $O(6^k \cdot n^{12})$

Primitiver FPT-Algorithmus

<u>Gesamte Laufzeit</u>

- Suche nach Konflikten: $O(n^{12})$

Primitiver FPT-Algorithmus

Gesamte Laufzeit

- Suche nach Konflikten: $O(n^{12})$
- Behebung der Konflikte: $O(6^k \cdot n^{12})$

Primitiver FPT-Algorithmus

Gesamte Laufzeit

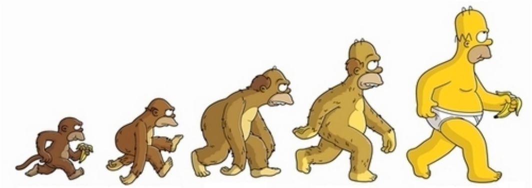
- Suche nach Konflikten: $O(n^{12})$
- Behebung der Konflikte: $O(6^k \cdot n^{12})$
- Generierung des Baums: $O(n^4)$

Primitiver FPT-Algorithmus

Gesamte Laufzeit

- Suche nach Konflikten: $O(n^{12})$
- Behebung der Konflikte: $O(6^k \cdot n^{12})$
- Generierung des Baums: $O(n^4)$
- Zusammen: $O(n^{12} + 6^k \cdot n^{12} + n^4) = O(6^k \cdot n^{12} + n^{12})$
  also ein FPT-Algorithmus

Laufzeitverbesserungen



Laufzeitverbesserungen

<u>Lemma 1:</u>
Sei T eine konsistente Menge von Topologien und $t \notin T$ eine Topologie, die min. eine Spezies enthält, die in T nicht vorkommt, dann ist $T \cup \{t\}$ konsistent

Laufzeitverbesserungen

Lemma 1:
Sei T eine konsistente Menge von Topologien und $t \notin T$ eine Topologie, die min. eine Spezies enthält, die in T nicht vorkommt, dann ist $T \cup \{t\}$ konsistent

Beweis Fall 1 t enthält genau 1 solche Spezies

- OBdA $t = [ab|cd]$ und d ist diese Spezies

Laufzeitverbesserungen

Lemma 1:
Sei T eine konsistente Menge von Topologien und $t \notin T$ eine Topologie, die min. eine Spezies enthält, die in T nicht vorkommt, dann ist $T \cup \{t\}$ konsistent

Beweis Fall 1 t enthält genau 1 solche Spezies

- OBdA $t = [ab|cd]$ und d ist diese Spezies
- Betrachte Baum B für T (ex, da T konsistent)

Laufzeitverbesserungen

Lemma 1:
Sei T eine konsistente Menge von Topologien und $t \notin T$ eine Topologie, die min. eine Spezies enthält, die in T nicht vorkommt, dann ist $T \cup \{t\}$ konsistent

Beweis Fall 1 t enthält genau 1 solche Spezies

- OBdA $t = [ab|cd]$ und d ist diese Spezies
- Betrachte Baum B für T (ex, da T konsistent)
- d einfügen in der Kante an c

Laufzeitverbesserungen

Lemma 1:
Sei T eine konsistente Menge von Topologien und $t \notin T$ eine Topologie, die min. eine Spezies enthält, die in T nicht vorkommt, dann ist $T \cup \{t\}$ konsistent

Beweis Fall 1 t enthält genau 1 solche Spezies

- OBdA $t = [ab|cd]$ und d ist diese Spezies
- Betrachte Baum B für T (ex, da T konsistent)
- d einfügen in der Kante an c
- $\rightarrow$ Baum, der alle Topologien von $T \cup \{t\}$ erfüllt

Korollar 1:
Jede 2-elementige Menge von Topologien (über verschiedenen Quartetts) ist konsistent

Lemma 2:
Wenn in drei Topologien $\{t_1, t_2, t_3\}$ mehr als 5 Spezies vorkommen, sind die Topologien konsistent

Laufzeitverbesserungen

Lemma 2:
Wenn in drei Topologien $\{t_1, t_2, t_3\}$ mehr als 5 Spezies vorkommen, sind die Topologien konsistent
Beweis:

- **Fall 1:** $\exists$ Spezies, die nur in 1 Topol. (OBdA t_1) vorkommt

Laufzeitverbesserungen

Lemma 2:

Wenn in drei Topologien $\{t_1, t_2, t_3\}$ mehr als 5 Spezies vorkommen, sind die Topologien konsistent

Beweis:

- **Fall 1:** $\exists$ Spezies, die nur in 1 Topol. (OBdA t_1) vorkommt
  - Korollar 1: $\{t_2, t_3\}$ konsistent

Laufzeitverbesserungen

Lemma 2:

Wenn in drei Topologien $\{t_1, t_2, t_3\}$ mehr als 5 Spezies vorkommen, sind die Topologien konsistent

Beweis:

- **Fall 1:** $\exists$ Spezies, die nur in 1 Topol. (OBdA t_1) vorkommt
  - Korollar 1: $\{t_2, t_3\}$ konsistent
  - Lemma 1: $\{t_1, t_2, t_3\}$ konsistent

Laufzeitverbesserungen

Lemma 2:
Wenn in drei Topologien $\{t_1, t_2, t_3\}$ mehr als 5 Spezies vorkommen, sind die Topologien konsistent
Beweis:

- **Fall 1:** $\exists$ Spezies, die nur in 1 Topol. (OBdA t_1) vorkommt
  - Korollar 1: $\{t_2, t_3\}$ konsistent
  - Lemma 1: $\{t_1, t_2, t_3\}$ konsistent
- **Fall 2:** Jede Spezies kommt in min. 2 Topol. vor. Abzählungsargument: jede Spezies kommt in **genau** 2 Topol. vor, je 2 Topol. haben genau 2 Spezies gemeinsam

Laufzeitverbesserungen

Lemma 2:
Wenn in drei Topologien $\{t_1, t_2, t_3\}$ mehr als 5 Spezies vorkommen, sind die Topologien konsistent
Beweis:

- **Fall 1:** $\exists$ Spezies, die nur in 1 Topol. (OBdA t_1) vorkommt
  - Korollar 1: $\{t_2, t_3\}$ konsistent
  - Lemma 1: $\{t_1, t_2, t_3\}$ konsistent
- **Fall 2:** Jede Spezies kommt in min. 2 Topol. vor. Abzählungsargument: jede Spezies kommt in **genau** 2 Topol. vor, je 2 Topol. haben genau 2 Spezies gemeinsam
  - D.h. $q_1 = \{a, b, c, d\}, q_2 = \{a, b, e, f\}, q_3 = \{c, d, e, f\}$

Laufzeitverbesserungen

Lemma 2:
Wenn in drei Topologien $\{t_1, t_2, t_3\}$ mehr als 5 Spezies vorkommen, sind die Topologien konsistent
Beweis:

- **Fall 1:** $\exists$ Spezies, die nur in 1 Topol. (OBdA t_1) vorkommt
  - Korollar 1: $\{t_2, t_3\}$ konsistent
  - Lemma 1: $\{t_1, t_2, t_3\}$ konsistent
- **Fall 2:** Jede Spezies kommt in min. 2 Topol. vor. Abzählungsargument: jede Spezies kommt in **genau** 2 Topol. vor, je 2 Topol. haben genau 2 Spezies gemeinsam
  - D.h. $q_1 = \{a, b, c, d\}, q_2 = \{a, b, e, f\}, q_3 = \{c, d, e, f\}$
  - Für alle 27 mögl. Topol. zu diesen Quartetts ex. Baum

Laufzeitverbesserungen

Suche nach Konflikten

Laufzeitverbesserungen

Suche nach Konflikten

- Lemma 2 $\Rightarrow$ ein Konflikt beinhaltet genau 5 Spezies

Laufzeitverbesserungen

Suche nach Konflikten

- Lemma 2 $\Rightarrow$ ein Konflikt beinhaltet genau 5 Spezies
- $\Rightarrow$ es genügt, die 5-elementigen Mengen von Spezies zu durchlaufen ($O(n^5)$) und die $\binom{5}{4} = 5$ Topologien über den aktuellen Spezies auf Konsistenz zu prüfen

Laufzeitverbesserungen

Suche nach Konflikten

- Lemma 2 $\Rightarrow$ ein Konflikt beinhaltet genau 5 Spezies
- $\Rightarrow$ es genügt, die 5-elementigen Mengen von Spezies zu durchlaufen ($O(n^5)$) und die $\binom{5}{4} = 5$ Topologien über den aktuellen Spezies auf Konsistenz zu prüfen
- Hierzu müssen je $\binom{5}{3} = 10$ Teilmengen der Größe 3 auf Konsistenz geprüft werden $\Rightarrow O(n^5 \cdot 10) = O(n^5)$

Laufzeitverbesserungen

Suche nach Konflikten

- Lemma 2 $\Rightarrow$ ein Konflikt beinhaltet genau 5 Spezies
- $\Rightarrow$ es genügt, die 5-elementigen Mengen von Spezies zu durchlaufen ($O(n^5)$) und die $\binom{5}{4} = 5$ Topologien über den aktuellen Spezies auf Konsistenz zu prüfen
- Hierzu müssen je $\binom{5}{3} = 10$ Teilmengen der Größe 3 auf Konsistenz geprüft werden $\Rightarrow O(n^5 \cdot 10) = O(n^5)$
- Man kann eine der Spezies beliebig festlegen und muss nur noch 4 andere Spezies wählen $\Rightarrow O(n^4)$

Laufzeitverbesserungen

Anpassen der Menge von Konflikten C

Idee: Berechne C nicht nach jeder Änderung neu, sondern

Laufzeitverbesserungen

Anpassen der Menge von Konflikten C

Idee: Berechne C nicht nach jeder Änderung neu, sondern

- Entferne die Konflikte, die behoben wurden

Laufzeitverbesserungen

Anpassen der Menge von Konflikten C

Idee: Berechne C nicht nach jeder Änderung neu, sondern

- Entferne die Konflikte, die behoben wurden
- Füge die Konflikte hinzu, die entstanden sind

Laufzeitverbesserungen

Anpassen der Menge von Konflikten C

Idee: Berechne C nicht nach jeder Änderung neu, sondern

- Entferne die Konflikte, die behoben wurden
- Füge die Konflikte hinzu, die entstanden sind

Dies ist in $O(n)$

Laufzeitverbesserungen

Anpassen der Menge von Konflikten C

Idee: Berechne C nicht nach jeder Änderung neu, sondern

- Entferne die Konflikte, die behoben wurden
- Füge die Konflikte hinzu, die entstanden sind

Dies ist in $O(n)$
$\Rightarrow$ Behebung der Konflikte in $O(6^k \cdot n)$

Laufzeitverbesserungen

Anpassen der Menge von Konflikten C in $O(n)$

- Die geänderte Topol. enthält 4 Spezies (das Quartett q)

Laufzeitverbesserungen

Anpassen der Menge von Konflikten C in $O(n)$

- Die geänderte Topol. enthält 4 Spezies (das Quartett q)
- Ein behobener bzw. entstandener Konflikt beinhaltet diese 4 Spezies und **eine** der anderen $O(n)$ Spezies (Lemma 2)

Laufzeitverbesserungen

Anpassen der Menge von Konflikten C in $O(n)$

- Die geänderte Topol. enthält 4 Spezies (das Quartett q)
- Ein behobener bzw. entstandener Konflikt beinhaltet diese 4 Spezies und **eine** der anderen $O(n)$ Spezies (Lemma 2)
- Es gibt nur $\binom{\binom{5}{4}}{3} = 10$ Teilmengen der Größe 3 über den Topologien für 5 Spezies

Laufzeitverbesserungen

Anpassen der Menge von Konflikten C in $O(n)$

- Die geänderte Topol. enthält 4 Spezies (das Quartett q)
- Ein behobener bzw. entstandener Konflikt beinhaltet diese 4 Spezies und **eine** der anderen $O(n)$ Spezies (Lemma 2)
- Es gibt nur $\binom{\binom{5}{4}}{3} = 10$ Teilmengen der Größe 3 über den Topologien für 5 Spezies
- also können nur $10n \in O(n)$ Konflikte entstehen oder behoben werden

Laufzeitverbesserungen

Anpassen der Menge von Konflikten C in $O(n)$

- Die geänderte Topol. enthält 4 Spezies (das Quartett q)
- Ein behobener bzw. entstandener Konflikt beinhaltet diese 4 Spezies und **eine** der anderen $O(n)$ Spezies (Lemma 2)
- Es gibt nur $\binom{\binom{5}{4}}{3} = 10$ Teilmengen der Größe 3 über den Topologien für 5 Spezies
- also können nur $10n \in O(n)$ Konflikte entstehen oder behoben werden
- Hieran sieht man auch, dass $|C| \leq 10 \cdot n \cdot k$

Laufzeitverbesserungen

Behebung der Konflikte:

- Nicht jede Änderung in einem Konflikt c behebt den Konflikt

Laufzeitverbesserungen

Behebung der Konflikte:

- Nicht jede Änderung in einem Konflikt c behebt den Konflikt
- $\Rightarrow$ irgendwann muss wieder ein $t \in c$ geändert werden

Laufzeitverbesserungen

Behebung der Konflikte:

- Nicht jede Änderung in einem Konflikt c behebt den Konflikt
- $\Rightarrow$ irgendwann muss wieder ein $t \in c$ geändert werden
- $\Rightarrow$ nur Branches benutzen, in denen c behoben wird

Laufzeitverbesserungen

Behebung der Konflikte:

- Nicht jede Änderung in einem Konflikt c behebt den Konflikt
- $\Rightarrow$ irgendwann muss wieder ein $t \in c$ geändert werden
- $\Rightarrow$ nur Branches benutzen, in denen c behoben wird
- Es gibt
  1. 3 Möglichkeiten, c mit genau 1 Änderung zu beheben

Laufzeitverbesserungen

Behebung der Konflikte:

- Nicht jede Änderung in einem Konflikt c behebt den Konflikt
- $\Rightarrow$ irgendwann muss wieder ein $t \in c$ geändert werden
- $\Rightarrow$ nur Branches benutzen, in denen c behoben wird
- Es gibt
  1. 3 Möglichkeiten, c mit genau 1 Änderung zu beheben
  2. 5 Möglichkeiten, c mit genau 2 Änderungen zu beheben

Laufzeitverbesserungen

Behebung der Konflikte:

- Nicht jede Änderung in einem Konflikt c behebt den Konflikt
- $\Rightarrow$ irgendwann muss wieder ein $t \in c$ geändert werden
- $\Rightarrow$ nur Branches benutzen, in denen c behoben wird
- Es gibt
  1. 3 Möglichkeiten, c mit genau 1 Änderung zu beheben
  2. 5 Möglichkeiten, c mit genau 2 Änderungen zu beheben
  3. 5 Möglichkeiten, c mit genau 3 Änderungen zu beheben

Laufzeitverbesserungen

Behebung der Konflikte:

- Branchingvektor $(1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3)$
  Branchingzahl $4,396$

Laufzeitverbesserungen

Behebung der Konflikte:

- Branchingvektor $(1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3)$
  Branchingzahl $4,396$
- Manche der Branches überschneiden sich, drei 2er Branches, alle 3er Branches können ausgelassen werden

Laufzeitverbesserungen

Behebung der Konflikte:

- Branchingvektor $(1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3)$
  Branchingzahl $4,396$
- Manche der Branches überschneiden sich, drei 2er Branches, alle 3er Branches können ausgelassen werden
- Branchingvektor $(1, 1, 1, 2, 2)$
  Branchingzahl $3,561$

Laufzeitverbesserungen

Behebung der Konflikte:

- Branchingvektor $(1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3)$
  Branchingzahl $4,396$
- Manche der Branches überschneiden sich, drei 2er Branches, alle 3er Branches können ausgelassen werden
- Branchingvektor $(1, 1, 1, 2, 2)$
  Branchingzahl $3,561$

Also können die Konflikte in $O(3,561^k \cdot n)$ behoben werden

Laufzeitverbesserungen

Insgesamt:

- Konflikte suchen: $O(n^4)$

Laufzeitverbesserungen

Insgesamt:

- Konflikte suchen: $O(n^4)$
- Konflikte beheben: $O(3,561^k \cdot n)$

Laufzeitverbesserungen

Insgesamt:

- Konflikte suchen: $O(n^4)$
- Konflikte beheben: $O(3,561^k \cdot n)$
- Baum generieren $O(n^4)$

Laufzeitverbesserungen

Insgesamt:

- Konflikte suchen: $O(n^4)$
- Konflikte beheben: $O(3,561^k \cdot n)$
- Baum generieren $O(n^4)$

Zusammen $O(3,561^k \cdot n + n^4)$

Verwandte Probleme

Verwandte Probleme

- **Alle Bäume**, die maximal k Topologien verletzen, können generiert werden (damit kann z.B. der Benutzer aus mehreren gültigen Lösungen wählen z.B. mit Fachwissen) $O(3,561^k \cdot n^5 + n^4) \Rightarrow$ FPT

Verwandte Probleme

- **Alle Bäume**, die maximal k Topologien verletzen, können generiert werden (damit kann z.B. der Benutzer aus mehreren gültigen Lösungen wählen z.B. mit Fachwissen) $O(3,561^k \cdot n^5 + n^4) \Rightarrow$ FPT
- **Gewichtete Topologien**: Gewichte ignorieren, alle passenden Bäume generieren, diejenigen auswählen, die am besten zu den Gewichten passen - FPT

Verwandte Probleme

- **Alle Bäume**, die maximal k Topologien verletzen, können generiert werden (damit kann z.B. der Benutzer aus mehreren gültigen Lösungen wählen z.B. mit Fachwissen) $O(3,561^k \cdot n^5 + n^4) \Rightarrow$ FPT
- **Gewichtete Topologien**: Gewichte ignorieren, alle passenden Bäume generieren, diejenigen auswählen, die am besten zu den Gewichten passen - FPT
- **SparseMQI** Nicht für alle Quartetts sind Topologien gegeben: auch mit $k = 0$ NPC $\Rightarrow$ nicht APX, nicht FPT

Verwandte Probleme

- **Alle Bäume**, die maximal k Topologien verletzen, können generiert werden (damit kann z.B. der Benutzer aus mehreren gültigen Lösungen wählen z.B. mit Fachwissen) $O(3,561^k \cdot n^5 + n^4) \Rightarrow$ FPT
- **Gewichtete Topologien**: Gewichte ignorieren, alle passenden Bäume generieren, diejenigen auswählen, die am besten zu den Gewichten passen - FPT
- **SparseMQI** Nicht für alle Quartetts sind Topologien gegeben: auch mit $k = 0$ NPC $\Rightarrow$ nicht APX, nicht FPT
- **SparseMQI**, aber max. j Topologien fehlen $\Rightarrow$ max. 3^j mögl. Ergänzungen $\Rightarrow O(3^j(3,561^k n + n^4)) \Rightarrow$ FPT

Ausblick

- Problemkernreduktion

Ausblick

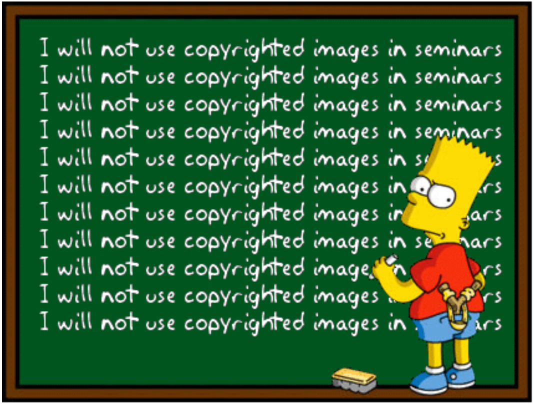
- Problemkernreduktion
- $k < n \Rightarrow$ Polynomiell

Ausblick

- Problemkernreduktion
- $k < n \Rightarrow$ Polynomiell
- $O^*((1+\varepsilon)^k)$

Fragen?

Kritik?


I will not use copyrighted images in seminars
I will not use copyrighted images in seminars
I will not use copyrighted images in seminars
I will not use copyrighted images in seminars

Tschüss, schönen Tag noch