Verbesserungen

Levin Nemesch, Joshua Sangmeister

03. Februar 2021

Algorithm Engineering - Projekt

ILP-Formulierung

Ansätze:

- Alternative ILP-Formulierung
- Alternativer exakter Algorithmus: Reduzierung auf MaxClique
- ILP mit neuer Separationsheuristik
- ILP-Formulierung mit stärkeren Constraints

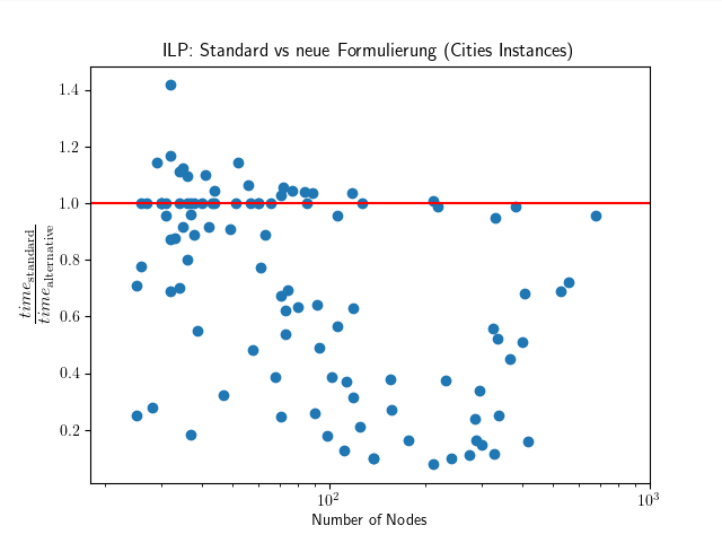
Alternative ILP-Formulierung - Idee

Wie verhalten sich weniger, aber dafür größere Constraints?

- Laut Vorlesung nicht gut...
- Bisherige Formulierung: Jeder Konflikt ein Constraint
- Alternative: Fasse für jedes Label Konflikte zusammen

$$\sum_{x_{c_j} \in C_i} x_{c_j} \leq |C_i|(1 - x_i) \qquad \forall \text{Label } i \text{ with } C_i \text{ as conflicts of i}$$

Alternative ILP-Formulierung - Ergebnisse



$\longrightarrow$ Laufzeit auf mittelgroßen Instanzen schlechter, mehr Timeouts (30%) auf größeren

Reduzierung auf MaxClique

- Beobachtung: Optimale Lösung des Labeling-Problems bildet Clique maximaler Größe im komplementären Konfliktgraphen, da dieser Label ohne Konflikte verbindet
  - Komplementärer Graph G': In G nicht-adjazente Knoten sind adjazent in G' und umgekehrt

Reduzierung auf MaxClique

- Beobachtung: Optimale Lösung des Labeling-Problems bildet Clique maximaler Größe im komplementären Konfliktgraphen, da dieser Label ohne Konflikte verbindet
  - Komplementärer Graph G': In G nicht-adjazente Knoten sind adjazent in G' und umgekehrt
- Aufbauen des komplementären Konfliktgraphen, lösen mittels Maximum-Clique-Algorithmus, Rücktransformation der Lösung
  - Algorithmus: MCQD, unter GNU General Public License veröffentlichte Implementation eines Papers von Konc & Janežič (An improved branch and bound algorithm for the maximum clique problem, 2007)

Reduzierung auf MaxClique

- Als exakter Algorithmus: Gut auf kleinen Instanzen, aber extrem langsam schon auf mittleren Instanzen
- Bei vorzeitigem Abbruch als Heuristik verwendbar, aber auch da schlechter als Simulated Annealing oder Gurobi mit timeout
- Benötigt Adjazenzmatrix, Platzverbrauch macht größere Instanzen unberechenbar

$\longrightarrow$ Ungeeignet! Vielleicht doch lieber ILP verbessern...

ILP mit Separationsheuristik

- Zu Beginn nur Punkt-Constraints einfügen, keine Konflikt-Constraints
- Separationsheuristik:

```
sort candidates descending by number of conflicts
foreach candidate with >=1 conflict:
    find max clique in this candidates conflicts
    add cut for all members of this clique
    remove conflicts of clique from graph
    break if "enough" cuts were generated
```

ILP mit Separationsheuristik

- Zu Beginn nur Punkt-Constraints einfügen, keine Konflikt-Constraints
- Separationsheuristik:

```
sort candidates descending by number of conflicts
foreach candidate with >=1 conflict:
    find max clique in this candidates conflicts
    add cut for all members of this clique
    remove conflicts of clique from graph
    break if "enough" cuts were generated
```

$\longrightarrow$ deutlich langsamer

ILP mit stärkeren Constraints

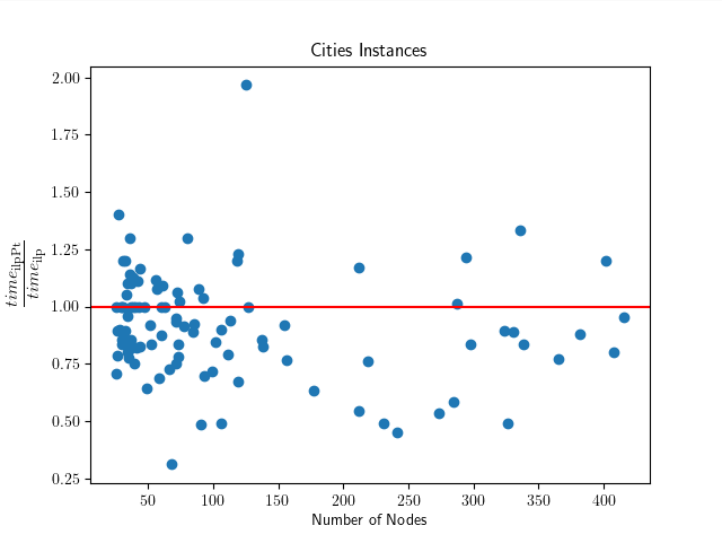
- Beobachtung 1: einfach zu ermitteln, ob Punkt in einem anderen Label liegt
- Beobachtung 2: wenn Punkt in Label liegt, hat dieses Konflikte mit allen Kandidaten des Punktes

ILP mit stärkeren Constraints

- Beobachtung 1: einfach zu ermitteln, ob Punkt in einem anderen Label liegt
- Beobachtung 2: wenn Punkt in Label liegt, hat dieses Konflikte mit allen Kandidaten des Punktes

$\longrightarrow$ Simple Verbesserung: Bei Generierung der Punkt-Constraints direkt alle Kandidaten mit einbeziehen, die den Punkt umschließen und auf die dadurch abgedeckten paarweisen Konflikte der Label verzichten
$\Longrightarrow$ stärkere Ungleichungen

ILP mit stärkeren Constraints



Yay! Eine Verbesserung! (zumindest meistens...)

ILP mit stärkeren Constraints

Welche Instanzen profitieren eher?