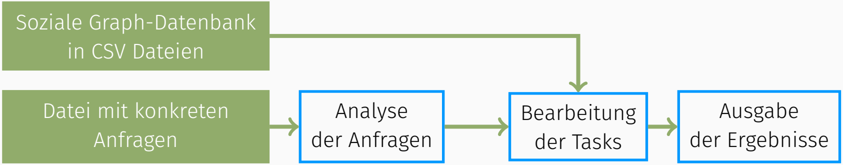
HOCHPERFORMANTE ANALYSEN IN GRAPH-DATENBANKEN

Moritz Kaufmann, Tobias Mühlbauer, Manuel Then,
Andrey Gubichev, Alfons Kemper, Thomas Neumann

5. März 2015

Technische Universität München

Der Wettbewerb



4 Anfragetypen

1. Kürzeste Wege finden
2. Top-K größten Interessengruppen von Personen finden, die nach einem Tab *b* geboren wurden
3. K-Hop Nachbarschaften von Knoten analysieren
4. Die zentralsten Knoten finden

Unser Fokus

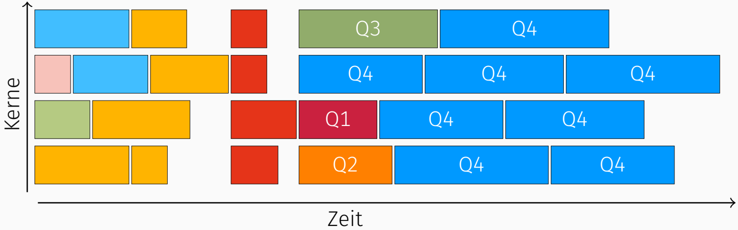
Parallelisierung

Effiziente Breitensuchen

Schranken zur Suchraumbegrenzung

Parallele Anfragebearbeitung

Aufteilung: **Laden → Indexe erstellen → Anfragen**
+ Dateien aufteilen

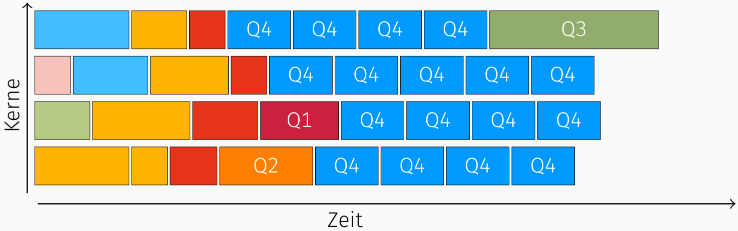


Probleme: *Weiße Flächen bedeuten Ineffizienz!*

- Unterschiedliche Laufzeiten
- Ineffizienz bei den Barrieren

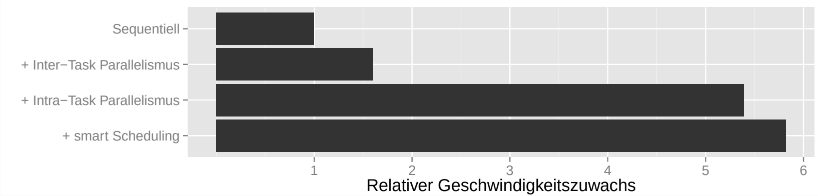
Parallele Anfragebearbeitung

Besser: + Q4 parallelisieren
+ Feingranulare Taskabhängigkeiten



Probleme: Priorisierung

Parallele Anfragebearbeitung



Aber weshalb nicht Faktor 8?

- I/O skaliert nicht linear
- Mehrere Threads teilen sich den CPU-Cache

Breitensuche Literatur

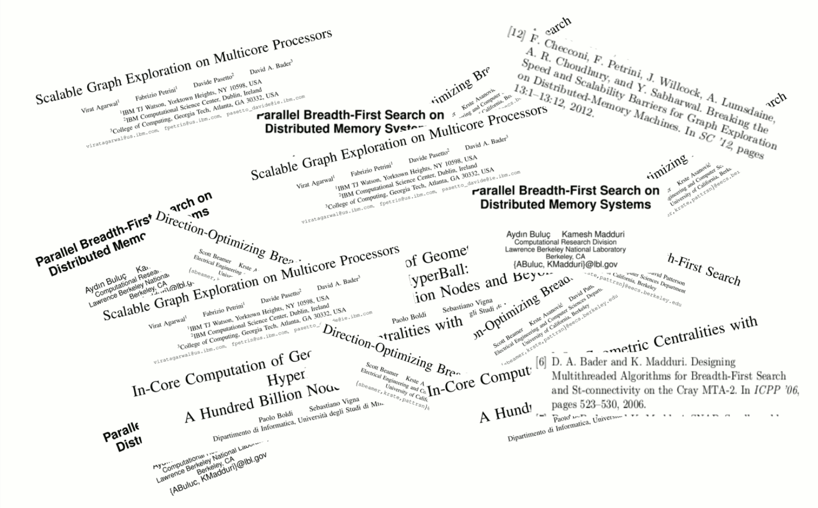
Anfragetyp 2: Eine Breitensuchen pro Interesse

$\rightarrow$ 10.000 Breitensuchen pro Anfrage

Anfragetyp 4: Eine Breitensuche pro Knoten

$\rightarrow$ 1 Millionen Breitensuchen pro Anfrage

Breitensuche Literatur



Breitensuche Literatur

Anfragetyp 2: Eine Breitensuchen pro Interesse

$\rightarrow$ 10.000 Breitensuchen pro Anfrage

Anfragetyp 4: Eine Breitensuche pro Knoten

$\rightarrow$ **1 Millionen Breitensuchen pro Anfrage**

Breitensuche Literatur

Anfragetyp 2: Eine Breitensuchen pro Interesse

$\rightarrow$ 10.000 Breitensuchen pro Anfrage

Anfragetyp 4: Eine Breitensuche pro Knoten

$\rightarrow$ **1 Millionen Breitensuchen pro Anfrage**

Parallelisierung hat immer Zusatzkosten!

Parallelisierung einzelner Breitensuchen unnötig um Kerne auszulasten.

Breitensuche Literatur

Suche: Optimierung von Breitensuchen *ohne Parallelisierung* auf großen Graphen.

Liste:

Wikipedia BFS

```
for (NodeId start : graph) {
  queue<NodeId> frontier, next;     frontier.push(start);
  vector<bool> seen(graph.size());  seen[start] = true;

  while (!frontier.empty()) {
    for (v : frontier) {
      for (NodeId n : neighbors(v)) {
        if (seen[n]) continue;
        next.push(n);
        seen[n] = true;
        do_something_with(n);
    } }

    handle_round_end();
    frontier = next;
    next.clear();
} }
```

Wikipedia BFS

```
for (NodeId start : graph) {
  queue<NodeId> frontier, next;      frontier.push(start);
  vector<bool> seen(graph.size());  seen[start] = true;

  while (!frontier.empty()) {
    for (v : frontier) {
      for (NodeId n : neighbors(v)) {
        if (seen[n]) continue;
        next.push(n);
        seen[n] = true;
        do_something_with(n);
    } }

    handle_round_end();
    frontier = next;
    next.clear();
} }
```

Langsam!

Problem(e):

Zufällige Speicherzugriffe

→ Cache misses

→ CPU muss warten

Fix: Manuelles Prefetching? Sortierung der Nachbarn? Sortierung der Queues? ...

Multi-Source BFS

Einsicht: Zufällige Zugriffsmuster *können nicht vermieden* werden.

Idee: Mehrere Breitensuchen innerhalb einer Iteration ausführen.

```
queue<NodeId> frontier, next;
vector<bool> seen(graph.size());

if (!seen[n]) {
  ...
}
```

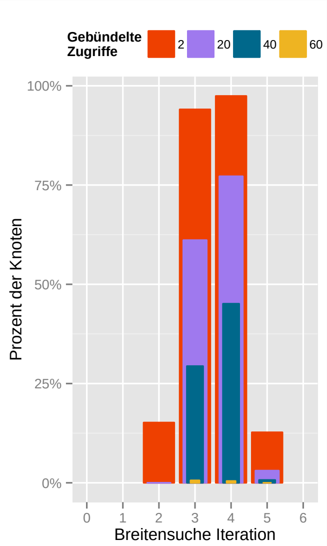
```
queue<NodeId, BfsSet> frontier, next;
vector<BfsSet> seen(graph.size());

cur_seen = seen[n];
for (b: activeBfs) {
  if (!cur_seen.contains(b)) {
    ...
  }
}
```

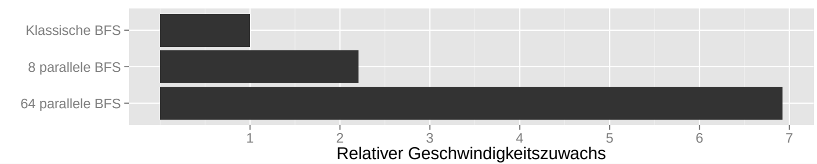
MS-BFS: Weshalb funktioniert das?

Viele große Graphen haben folgende Eigenschaften:

- **Kleiner Durchmesser:** Alle Knoten sind sehr nah beieinander
- Die Anzahl der Nachbarn pro Knoten folgt dem **power law**
- Beispiele: Soziale Netzwerke, Webgraphen, Kommunikationsgraphen



MS-BFS: Messung



Gut skalierbar bis zu $64 * 8 = 512$ Breitensuchen pro Kern mit SIMD!

Bisherige Benchmarks (Graph500) decken diesen Use-Case nicht ab.

The More the Merrier:
Efficient Multi-Source Graph Traversal

Suchraumeinschränkung (Q2)

Aufgabe: Finden der *Top-K* größten Interessengruppen von Personen, die nach einem Tag *b* geboren wurden.

Beispiel: k=3, b=Februar 1995

Aktuelle Top-K Werte

Interesse	Größte CC
Fußball	9920
Brasilien	8137
Wandern	4219

Suchraumeinschränkung (Q2)

Aufgabe: Finden der *Top-K* größten Interessengruppen von Personen, die nach einem Tag *b* geboren wurden.

Beispiel: k=3, b=Februar 1995

Aktuelle Top-K Werte

Interesse	Größte CC
Fußball	9920
Brasilien	8137
Wandern	4219

Behelfsindexe

Interesse	Jüngste Person	Personen
Reisen	Mai 1994	12519
Ski fahren	Mai 1996	712
DB	Juni 1990	5
...	...	...

Suchraumeinschränkung (Q2)

Aufgabe: Finden der *Top-K* größten Interessengruppen von Personen, die nach einem Tag *b* geboren wurden.

Beispiel: k=3, b=Februar 1995

Aktuelle Top-K Werte

Interesse	Größte CC
Fußball	9920
Brasilien	8137
Wandern	4219

Behelfsindexe

Interesse	Jüngste Person	Personen
Reisen	~~Mai 1994~~	12519
Ski fahren	Mai 1996	712
DB	Juni 1990	5
...	...	...

Suchraumeinschränkung (Q2)

Aufgabe: Finden der *Top-K* größten Interessengruppen von Personen, die nach einem Tag *b* geboren wurden.

Beispiel: k=3, b=Februar 1995

Aktuelle Top-K Werte

Interesse	Größte CC
Fußball	9920
Brasilien	8137
Wandern	4219

Behelfsindexe

Interesse	Jüngste Person	Personen
Reisen	~~Mai 1994~~	12519
Ski fahren	Mai 1996	~~712~~
DB	Juni 1990	5
…	…	…

Suchraumeinschränkung (Q2)

Aufgabe: Finden der *Top-K* größten Interessengruppen von Personen, die nach einem Tag *b* geboren wurden.

Beispiel: k=3, b=Februar 1995

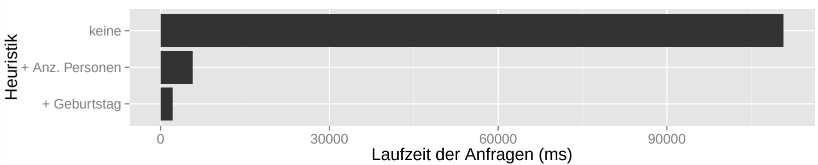
Aktuelle Top-K Werte

Interesse	Größte CC
Fußball	9920
Brasilien	8137
Wandern	4219

Behelfsindexe

Interesse	Jüngste Person	Personen
Reisen	~~Mai 1994~~	12519
Ski fahren	Mai 1996	~~712~~
DB	~~Juni 1990~~	~~5~~
...	...	...

Suchraumeinschränkung (Q2): Effizienz



Suchraumeinschränkung (Q4)

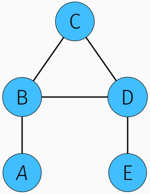
Aufgabe: Finden der k zentralsten Personen im Graph. Zentralität wird bestimmt über die Distanz zu allen anderen Personen.

Gesucht: Eine untere Schranke für die Distanzsumme!

oder: Eine obere Schranke wieviele Knoten mit Distanz k erreicht werden.

Aufgabe: Finden der k zentralsten Personen im Graph. Zentralität wird bestimmt über die Distanz zu allen anderen Personen.

Gesucht: Eine untere Schranke für die Distanzsumme!
oder: Eine obere Schranke wieviele Knoten mit Distanz k erreicht werden.

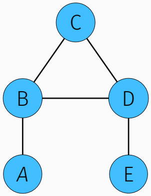


$$\begin{aligned}
DSUM(A) &= d(B) + d(C) + d(D) + d(E) \\
&= 1 + 2 + 2 + 3 = 8 \\
DSUM(A) &= 1 * (|R_1| - |R_0|) + 2 * (|R_2| - |R_1|) + 3 * (|R_3| - |R_2|) \\
&= 1 * (2 - 1) + 2 * (4 - 2) + 3 * (5 - 4) \\
&= 1 * 1 + 2 * 2 + 3 * 1 = 8
\end{aligned}$$

Suchraumeinschränkung (Q4)

Aufgabe: Finden der k zentralsten Personen im Graph. Zentralität wird bestimmt über die Distanz zu allen anderen Personen.

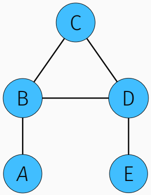
Gesucht: Eine untere Schranke für die Distanzsumme!
oder: Eine obere Schranke wieviele Knoten mit Distanz k erreicht werden.



R_0	R_1	R_2	R_3
$\{A\}$	$\{A, B\}$	$\{A, B, C, D\}$	$\{A, B, C, D, E\}$
$\{B\}$	$\{B, A, C, D\}$	$\{B, A, C, D, E\}$	-
$\{C\}$	$\{C, B, D\}$	$\{C, B, D, A, E\}$	-
$\{D\}$	$\{D, B, C, E\}$	$\{D, B, C, E, A\}$	-
$\{E\}$	$\{E, D\}$	$\{E, D, C\}$	$\{E, D, B, C, A\}$

Suchraumeinschränkung (Q4)

Ansatz: Die Menge der Personen die von A mit $k+1$ Schritten erreichbar sind, ist durch die Vereinigung der Personen, die seine Nachbarn in k Schritten erreichen können, beschränkt.



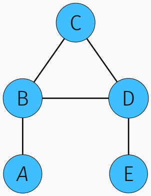
R_0	R_1	R_2	R_3
$\{A\}$	$\{A, B\}$	$\{A, B, C, D\}$	$\{A, B, C, D, E\}$
$\{B\}$	$\{B, A, C, D\}$	$\{B, A, C, D, E\}$	-
$\{C\}$	$\{C, B, D\}$	$\{C, B, D, A, E\}$	-
$\{D\}$	$\{D, B, C, E\}$	$\{D, B, C, E, A\}$	-
$\{E\}$	$\{E, D\}$	$\{E, D, C\}$	$\{E, D, B, C, A\}$

Suchraumeinschränkung (Q4)

Annahme: Die Mengen sind unabhängig.

$\rightarrow$ Es gilt $|A \cup B| = |A| + |B|$.

$\rightarrow$ Nur die Größe muss gespeichert werden.



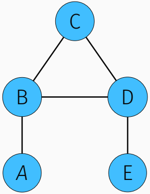
R_0	R_1	R_2	R_3
$\{A\}$	$\{A, B\}$	$\{A, B, C, D\}$	$\{A, B, C, D, E\}$
$\{B\}$	$\{B, A, C, D\}$	$\{B, A, C, D, E\}$	-
$\{C\}$	$\{C, B, D\}$	$\{C, B, D, A, E\}$	-
$\{D\}$	$\{D, B, C, E\}$	$\{D, B, C, E, A\}$	-
$\{E\}$	$\{E, D\}$	$\{E, D, C\}$	$\{E, D, B, C, A\}$

Suchraumeinschränkung (Q4)

Annahme: Die Mengen sind unabhängig.

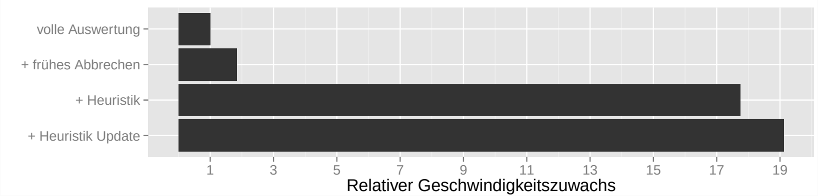
$\rightarrow$ Es gilt $|A \cup B| = |A| + |B|$.

$\rightarrow$ Nur die Größe muss gespeichert werden.



$R_0 \approx \|R_0\|$	$R_1 \approx \|R_1\|$	$R_2 \approx \|R_2\|$
1	2	$6 \rightarrow 5$
1	4	$15 \rightarrow 5$
1	3	$11 \rightarrow 5$
1	4	$15 \rightarrow 5$
1	2	$6 \rightarrow 5$

Suchraumeinschränkung (Q4): Effizienz



Zusammenfassung

→ MS-BFS mit einer neuen Sicht auf Graphalgorithmen
→ Heuristiken sind für Top-K Anfragen enorm wichtig

Graphanalysesysteme sind (leider) sehr weit von handgeschriebenen Programmen entfernt!

→ Haben wir die passenden Benchmarks?
→ Haben wir die passenden Abstraktionen?