Densely Connected Biclusters

Stefan Dresselhaus, Thomas Pajenkamp

Parallele Algorithmen und Datenverarbeitung
Programmierprojekt im Wintersemester 2013/14

Universität Bielefeld – Technische Fakultät

6. Februar 2014

Inhaltsverzeichnis

1 Zielsetzung des Projekts

Im Rahmen dieses Programmierprojekts wurde ein Programm entworfen und entwickelt, um Densely Connected Biclusters, im weiteren DCB, in einem biologischen Netzwerk zu ermitteln. Bei DCB handelt es sich um Teilgraphen eines Netzwerks, dessen Knoten untereinander hoch vernetzt sind und Objekte mit ähnlichen Eigenschaften repräsentieren.
Die Suche nach DCB ist ein NP-schweres Problem [Dum29]. Da mit einem geeigneten Algorithmus jedoch voneinander unabhängige Lösungspfade einzeln verfolgt werden können, ist das Problem gut für eine parallele Berechnung geeignet, wodurch die Gesamtlaufzeit stark reduziert werden kann.

1.1 Densely Connected Biclusters

Ausgangsbasis ist ein ungerichteter ungewichteter Graph $G = (V, E)$, dessen Knoten mit p Attributen versehen sind. Jedem Knoten n ist zu jedem Attribut i ein numerischer Wert a_{ni} zugewiesen.
Ein DCB $D_k = (V_k, E_k)$ ist ein Teilgraph von G, der durch die Paramter $\alpha \in [0, 1]$, $\delta \in \mathbb{N}$ und $\omega \in \mathbb{R}^p$ beschränkt wird und die folgende drei Eigenschaften erfüllt.

- Der Teilgraph ist zusammenhängend.
- Die Dichte des Teilgraphen unterschreitet einen Schwellenwert α nicht, also $\frac{2 \cdot |E_k|}{|V_k|(|V_k|-1)} \geq \alpha$.
- Für mindestens δ Attribute liegen die Werte aller Knoten des Teilgraphen höchstens ω_i auseinander. Anders ausgedrückt

$$\delta \leq \left| \left\{ 1 \leq k \leq p \;\middle|\; \omega_k \geq \left(\max_n a_{nk} - \min_n a_{nk} \right) \right\} \right|.$$

2 Wahl der Programmiersprache

Imperativ: Gefahr unerwünschter wechselseitiger Beeinflussung, Gefahr Verklemmung bei Kommunikation
Funktional: Garantiert keine Nebenwirkung
Haskell: Pakete zur einfachen Parallelisierung, kaum Änderung des sequentiellen Codes nötig

zweischneidiges Schwert: Lazy-Evaluation → 1) nicht zu viele „Sparks" auf einmal 2) baut große Datenstrukturen, anstatt direkt zu reduzieren, wo das Gesamtergebnis immer benötigt wird (keine Parallelisierungs-Problem, muss man sich aber mit auseinandersetzen)

3 Der Algorithmus

Von Pseudocode rüberkopieren, eventuell anpassen an Details der Programmierung zur besseren Effizienz.

4 Ausführung und Auswertung

Amdahls Gesetz, Minskys Vermutung
Nach jedem Erweiterungsschritt: Sammeln und Aufgaben neu verteilen → Kommunikation

5 Fazit

Literatur

[Dum29] Dummy. „Dummy. Dummy“. In: *Dummy* (26. Okt. 1929).