Universität Konstanz
Fachbereich Informatik & Informationswissenschaft
David Boetius

Software Projekt, Gruppe 2.1
Sommersemester 19
June 12, 2019

Milestone 2 StressMinimization Benchmark

Einführung

In den benchmarks wurde einzig die Ausführungszeit in Millisekunden gemessen (Median der Ausführungszeit bei in der Regel 20 Runden Warmup und 50 Runden für den Benchmark, für die Benchmarks mit mehr als 1000 Knoten wurde hiervon abgewichen).
Der reservierte Arbeitsspeicher in der VM nach der Ausführung der warmup Runden wurde auch gemessen, gibt aber nicht wieder wie hoch der Speicherbedarf für die eigentliche Ausführung des Algorithmus wirklich war. Der in VANTED angezeigte verwendete Arbeitspeicher während der Ausführung war typischerweise etwa halb so groß, wie der von der VM als total available memory angegebene Arbeitsspeicher.
Alle Messungen wurden auf einem iMac Ende 2013, Intel i5, 4 Kerne, 16 GB Arbeitsspeicher ausgeführt.
Gebenchmarks wurden Sternen Graphen, Rad Graphen (Sternen Graphen, bei denen die äußeren Knoten zusätzlich zu einem Kreis verbunden wurden), vollständige Graphen, Pfade, sowie Barabais Albert Netzwerke, Watts Strogatz Netzwerke und Sierpinsky Dreiecke.
Barabais Albert Netzwerke und Watts Strogatz Netzwerke wurden gewählt um Graphen zu benchmarken, die eine gewisse Ähnlichkeit zu bestimmten nicht künstlich erzeugten Graphen haben.

Asymptotische Performance

Abbildung 1 zeigt die Laufzeiten für Watts-Strogatz und Barabasi-Albert Netzwerke mit Knotenanzahlen zwischen 100 und 1000 an.
Es zeichnet sich die quadratische Laufzeitklasse der Berechnung ab. Abbildung **??** zeigt für diese Graphen die respektiven Kantenanzahlen. Mit dieser zusätzlichen Information können wir in der ersten Abbildung den Unterschied zwischen den beiden Netzwerken erklären. Hier spielt die Berechnung der Distanzen ein, die von der Anzahl der Kanten abhängt.
In den Abbildungen kommen für einzelne Knotenzahlen mehrere Laufzeiten vor, weil für diese in verschiedenen Benchmark-Suits Messwerte vorlagen.

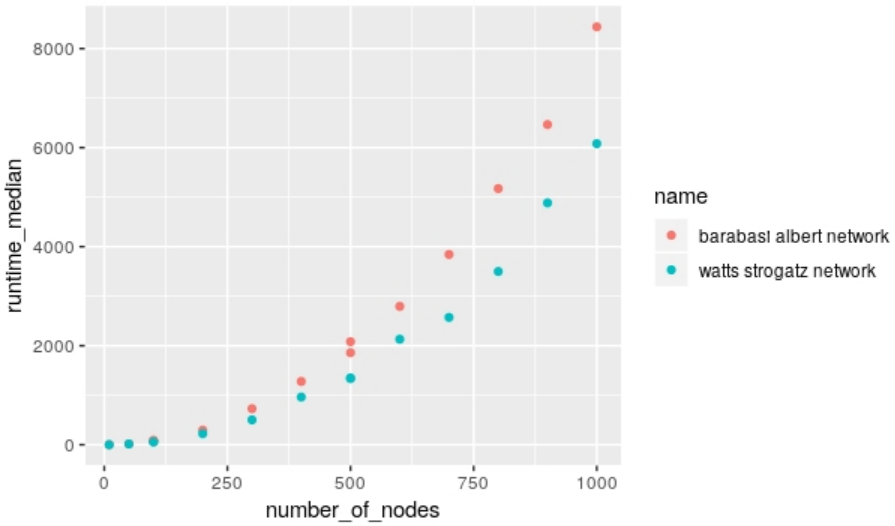

Figure 1: Laufzeiten der Ausführung des Stress Minimization Prozesses auf Watts-Strogatz und Barabasi-Albert Netzwerken für $n \in [100, 1000]$

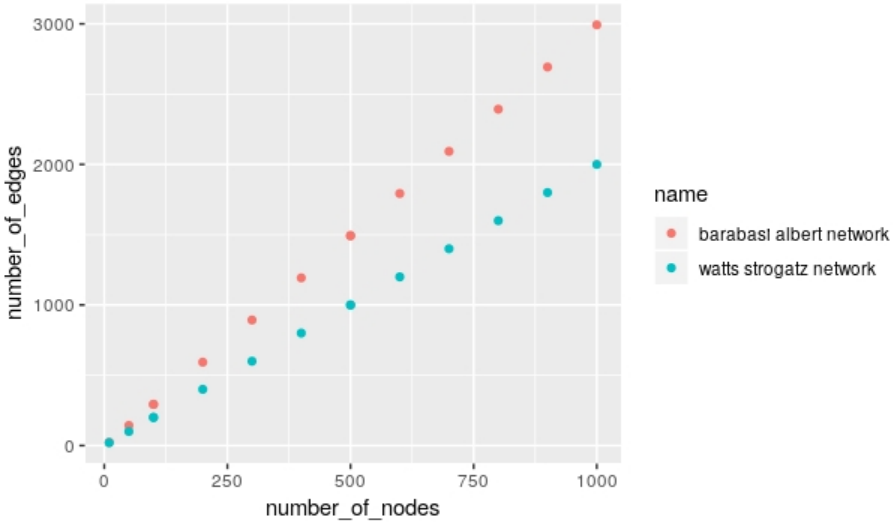

Figure 2: Kantenanzahlen von Watts-Strogatz und Barabasi-Albert Netzwerken

Performance für verschiedene Typen von Graphen

In den Abbildungen **??** und **??** sehen wir die Laufzeiten des Stress Minimization Algorithmus für verschiedene Typen von Graphen: Stern Graphen, Rad Graphen, vollständige Graphen, Pfade und wiederum Watts-Strogatz und Barabasi-Albert Natzwerken.
Für Pfade lag die Ausführungszeit immer wesentlich höher als bei den anderen Graphen, so dass kein Benchmark mehr mit $n = 500$ ausgeführt wurde. Die Ausführungszeit des Algorithmus hängt gegenwärtig also auch wesentlich von anderen Faktoren als nur der Knotenzahl und der Kantenzahl ab. Für diese Graphen führt der Algorithmus sehr viele Iterationen aus, wobei das berechnete Layout einen sehr geringen Stress (≈ 0) aufweißt. Durch den geringen Stress erfüllen die Veränderungen des Stress in den Itterationsschritten immernoch das Kriterium $\frac{stress_{prev} - stress_{new}}{stress_{prev}} > 1 \cdot 10^{-4}$ aber die absolute Änderung im Stress ist gering.
Ansonsten können wir wiederum sehen, dass auch die Kantenanzahl signifikanten Einfluss auf die Ausführungszeit haben kann (vollständiger Graph mit $n = 500$) und wir können die tatsächlichen Ausführungszeiten abschätzen.

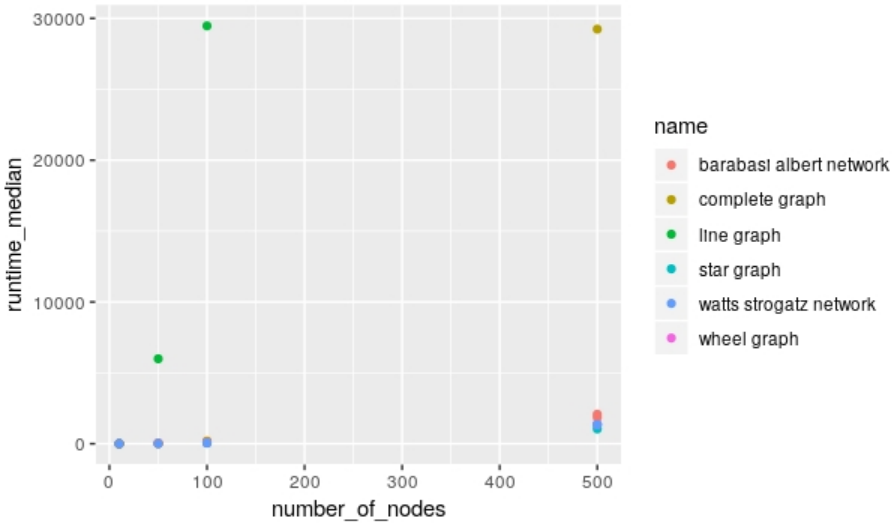

Figure 3: Laufzeiten für verschiedene Graphtypen nach Knotenzahl

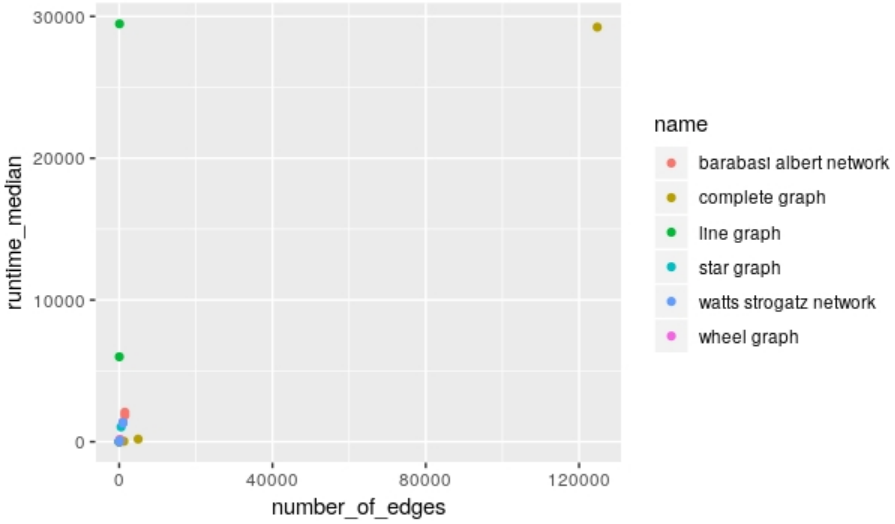

Figure 4: Laufzeiten für verschiedene Graphtypen nach Kantenzahl

Sierpisky Dreiecke: Faktoren und Speicherprobleme

Abbildung **??** zeigt Laufzeiten für Sierpinsky Dreiecke mit Rekursionstiefen zwischen 3 und 8 an.
Hieraus können wir beispielhaft ablesen, in welchen Bereichen die Laufzeiten für Graphen mit mehr als 1000 Knoten liegen.
Während die Aufführungszeit für ein Sierpinsky Dreieck mit etwa 1000 Knoten circa bei 20 Sekunden lag, stieg dieser Wert für die weiteren Rekursionstiefen auf etwa 4 Minuten ($n \approx 3000$), bzw. 50 Minuten ($n \approx 10000$) an.

Für das Sierpinsky Dreieck mit Rekursionstiefe 9 ($n = 29526$) stürzte das Programm ab, weil der Heap-Speicherplatz (ca. 4 GB) nicht ausreichte. Hier wurde die maximale Größe des Heap erreicht, der für die VM eingestellt war.

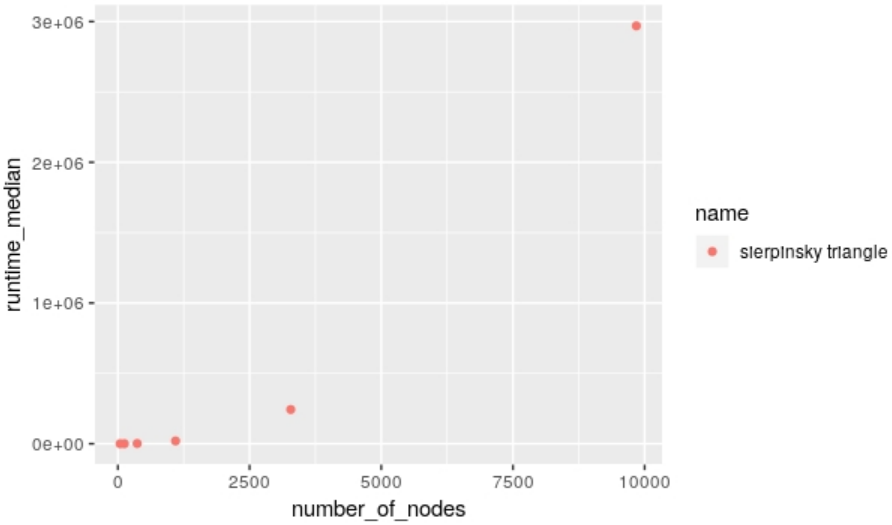

Figure 5: Laufzeiten für Sierpinsky Dreiecke bis zur Tiefe 8