Parallele Suchalgorithmen für integrale Bäume

Viola Campos

Hochschule RheinMain, Wiesbaden

Abstract

Ziel des Masterprojekts ist die Entwicklung eines Algorithmus zur Suche nach integralen Bäumen, also nach Bäumen mit einem ganzzahligen Spektrum. Insbesondere werden die Möglichkeiten parallelen bzw. verteilten Rechnens untersucht.

1. Einleitung

Die algebraischen Graphentheorie untersucht graphentheoretische Probleme anhand der algebraischen Eigenschaften assoziierter Matrizen. Beispielsweise lassen sich Aussagen über die Struktur eines Graphen treffen mit Hilfe der Eigenwerte seiner Adjazenzmatrix.

Diese Arbeit beschäftigt sich mit der Suche nach Bäumen, deren Adjazenzmatrix ausschließlich ganzzahlige Eigenwerte besitzt. 2009 veröffentlichte A. E. Brouwer eine komplette Liste solcher integraler Bäume bis zu einer Größe von 50 Knoten [1]. Ziel dieses Projekts war die Entwicklung eines Algorithmus, der Brouwers Ergebnisse überprüft und die Liste auf größere Bäume erweitert.

2. Grundlagen

Im Rahmen dieser Arbeit werden folgende Definitionen und Bezeichnungen genutzt.

Definition 1 (Graph). *Ein* Graph *ist ein Tupel $G = (V, E)$ mit Knotenmenge $V \neq \emptyset$ und Kantenmenge E. Dabei ist E in* ungerichteten *Graphen eine Teilmenge aller 2-elementigen Teilmengen von V, in* gerichteten *Graphen eine Teilmenge aller Paare (i, j) die durch das kartesische Produkt $V \times V$ entstehen. Der Graph heisst* endlich*, falls $\#V < \infty$. Enthält der Graph keine Schlingen oder Mehrfachkanten, so heißt er* einfach *oder* schlicht*.*

- *Der* Grad *eines Knotens ist definiert als die Zahl aller mit diesem Knoten inzidenten Kanten.*
- *Ein* Weg *der Länge m in G von v nach w ist eine endliche Menge von Knoten $v = w_0, w_1, \ldots, w_m = w$, so dass w_{k-1} und w_k adjazent sind für $1 \leq k \leq m$.*
- *Ein Graph heißt* zusammenhängend*, falls je zwei beliebige Knoten von G durch einen Weg in G verbunden sind.*

Email address: viola.campos@hs-rm.de (Viola Campos)

- *Der* Abstand $d(v, w; G)$ *zwischen zwei Knoten v und w ist die Länge des kürzesten Weges, der v und w in G verbindet.*
- *Der* Durchmesser D *von* $G = (V, E)$ *ist die größte Distanz in G:* $D(G) = \max_{v,w \in V} d(v, w; G)$.

Definition 2 (Teilgraph). *Ein Graph* $G' = (V', E')$ *heißt Teilgraph oder Subgraph von* $G = (V, E)$*, wenn seine Knotenmenge* V' *Teilmenge von* V *und seine Kantenmenge* E' *Teilmenge von* E *ist, also* $V' \subseteq V$ *und* $E' \subseteq E$ *gilt.*

Gilt für einen Teilgraphen G' zusätzlich, dass E' alle Kanten zwischen den Knoten in V' enthält, die auch in E vorhanden sind, so bezeichnet man G' als den durch V' *induzierten Teilgraphen* von G. Für eine Knotenmenge $W \subseteq V$ bezeichnet man mit $G \setminus W$ den induzierten Teilgraphen, der aus $G = (V, E)$ durch Weglassen der Knoten aus W und aller mit diesen Knoten inzidenten Kanten entsteht.

Definition 3 (Baum). *Ein* Baum *ist ein zusammenhängender, kreisfreier, einfacher Graph. Ist ein Knoten des Baumes als dessen Wurzel definiert, spricht man von einem gewurzelten Baum.*

Bei den Bäumen, die in dieser Arbeit betrachtet werden, handelt es sich um ungerichtete, nicht gewurzelte Bäume.

Definition 4 (Adjazenzmatrix). *Für einen Graphen* $G = (V, E)$ *mit n Knoten ist die Adjazenzmatrix* $A(G) = [a_{ij}]$ *definiert als*

$$A_{ij} = \begin{cases} 1 \text{ falls } (i, j) \in E \\ 0 \text{ sonst} \end{cases} \qquad 0 \leq i < n \text{ und } 0 \leq j < n$$

Die nach Größe geordnete Folge aller Eigenwerte der Adjazenzmatrix eines Graphen G bezeichnet man als *Spektrum* von G.

Definition 5 (Integrale Graphen und Bäume). *Einen Graphen, dessen Spektrum nur ganzzahlige Eigenwerte enthält, nennt man* integral*. Handelt es sich bei dem Graphen um einen Baum, spricht man von einem* integralen Baum*.*

Für bipartite Graphen gilt, dass für jeden Eigenwert θ von G auch $-\theta$ ein Eigenwert mit der gleichen Multiplizität ist. Da es sich bei jedem Baum um einen bipartiten Graph handelt, sind die Spektren von Bäumen symmetrisch bezüglich der 0.

2.1. Bezeichnungen für Bäume

Analog zu [1] werden in dieser Arbeit zur Bezeichnung von Bäumen folgende Symbole verwendet.

- K_1 ist der einzelne Knoten. K_2 ist der Baum mit zwei Knoten.
- $K_{1,m}$ ist der Stern mit $m + 1$ Knoten.
- Ist G ein Baum, bezeichnet SG die Unterteilung von G mit einem neuen Knoten in der Mitte jeder Kante von G.

- Sind G und H gewurzelte Bäume, dann entsteht $G \sim H$ durch die Verbindung der beiden Wurzeln mit einer neuen Kante.
- $C(G_1 \dots G_s)$ bezeichnet den Baum, der aus s gewurzelten Bäumen $G_1 \dots G_s$ entsteht, wenn deren Wurzelknoten durch einen neuen Knoten verbunden werden. $C(mH)$ bezeichnet die Verbindung von m Kopien von H durch einen neuen Knoten.
- Sind G und H gewurzelte Bäume, dann ist $G * H$ der gewurzelte Baum, der durch die Vereinigung der beiden Wurzelknoten entsteht.
- Gewurzelte Bäume $T(n_k, \dots, n_1)$ sind induktiv wie folgt definiert:

  $T()$ ist K_1 und $T(n_k, \dots, n_1) = C(n_k T(n_{k-1}, \dots, n_1))$ für $k > 0$. Beispielsweise ist $T(m)$ gleich $K_{1,m}$ und $T(m, 1)$ entspricht $S\,K_{1,m}$.

3. Methoden

Um einen Eindruck der Komplexität des Problems zu bekommen, wurde im ersten Schritt eine Brute-Force Suche nach integralen Bäumen in Python implementiert. Diese erzeugt alle nicht-isomorphen Bäume mit einer bestimmten Knotenanzahl, was mit Hilfe eines Algorithmus von Richmond et al. [2] in linearer Zeit möglich ist. Danach wird für jeden Baum dessen Spektrum berechnet und auf Ganzzahligkeit überprüft. Algorithmus 1 zeigt das Vorgehen.

Algorithm 1: Brute-Force Implementierung

Input : n: Anzahl der Knoten
Output: resultSet: Liste aller integralen Bäume mit n Knoten

```
/* Initialisierung                                                    */
resultSet ← {}
candidateSet ← createAllNonisomorphicTrees(n)
for each tree ∈ candidateSet do
    s ← computeSpectrum(tree)
    if isIntegral(s) then
        resultSet ← resultSet ∪ {tree}
```

Die empirisch ermittelten Werte für die Laufzeit des Brute-Force Algorithmus lagen bei $t \approx 10 * e^{1.1n}$ in Abhängigkeit von der Knotenanzahl n. Die Überprüfung aller Bäume mit 50 Knoten hätte also $\approx 10^{12}$ Jahre benötigt. Betrachtet man die Anzahl aller nicht-isomorphen Bäume mit 50 Knoten, ist offensichtlich, dass 10^{17} Bäume [3] zu viele sind, um jeden davon einzeln auf integrale Spektren zu überprüfen.

3.1. Interlacing

Um dennoch alle integralen Bäume bis zu 50 Knoten zu berechnen, muss der Suchraum eingeschränkt werden. Dazu lässt sich folgende Tatsache nutzen [4]:

Definition 6 (Interlacing). *Seien θ und η geordnete Folgen reeller Zahlen $\theta_1 \geq \dots \geq \theta_n$ und $\eta_1 \geq \dots \geq \eta_m$ mit $m < n$. Die zweite Folge* interlaced *die erste, wenn gilt:*

$$\theta_i \geq \eta_i \geq \theta_{n-m+i}, \text{ für } i = 1, \dots, m.$$

Gilt $m = n - 1$, folgt:

$$\theta_1 \geq \eta_1 \geq \theta_2 \geq \eta_2 \geq \cdots \geq \eta_m \geq \theta_n$$

Sei x ein Knoten eines Graphen G und $G \setminus x$ der Graph der nach Entfernung des Knotens x aus G entsteht. Besitzt $G \setminus x$ zwei Eigenwerte, die echt zwischen zwei aufeinander folgenden ganzen Zahlen $a \in \mathbb{Z}$ und $a + 1$ liegen, dann hat auch G einen Eigenwert zwischen a und $a + 1$ und damit kein ganzzahliges Spektrum. Handelt es sich bei G um einen Baum, zerfällt $G \setminus x$ in unzusammenhängende Komponenten oder einen Wald. Das Spektrum eines solchen unzusammenhängenden Graphen ist die Vereinigung der Spektren der einzelnen Komponenten, $G \setminus x$ hat also zwei Eigenwerte zwischen a und $a + 1$, wenn dies schon für eine Komponente oder die Vereinigung einiger Komponenten gilt.
Damit lassen sich neue Bäume folgendermassen erzeugen [1]:

- Schrittweise werden alle 2-, 3-, ..., n-elementigen Bäume ausgehend vom Baum mit einem Knoten durch Hinzufügen einzelnder Knoten erzeugt.
- Wurde bereits eine Isomorphie eines Baumes untersucht, wird er nicht weiter betrachtet
- Erfüllt ein Baum die oben beschriebene Abbruchbedingung, wird er verworfen und keine weiteren Bäume aus ihm erzeugt.
- Alle übrigen Bäume werden auf ganzzahlige Spektren untersucht.

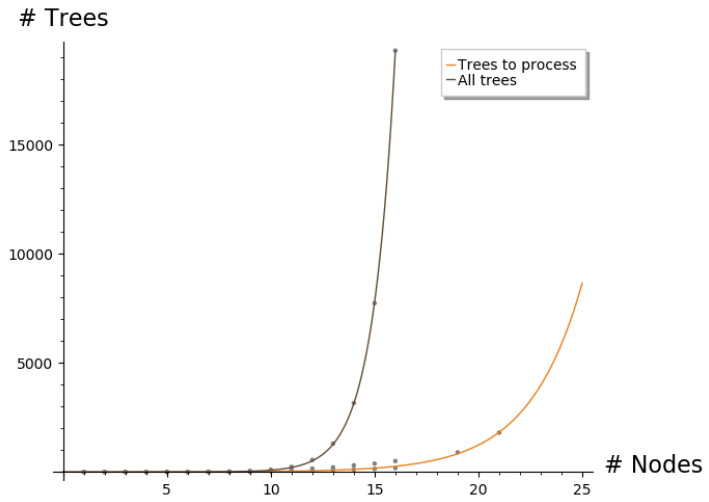


Abbildung 1: Anzahl zu untersuchender Bäume mit n Knoten (gelb) im Vergleich zu allen Bäumen (braun)

Um Mehrfachberechnungen auszuschließen, muss für jeden neu erzeugten Baum überprüft werden, ob bereits ein zu ihm isomorpher Baum bearbeitet wurde. Dazu wird eine kanonische Darstellung der Bäume definiert, so dass isomorphe Bäume auf dieselbe kanonische Darstellung abgebildet werden. Für jeden neu erzeugten Baum wird dessen kanonische Darstellung berechnet und überprüft, ob er in der Liste bereits bearbeiteter Bäume enthalten ist.

3.2. *Berechnung der Graphenspektren*

Profiling der Funktion zeigt, dass die Berechnung der Spektren der untersuchten Bäume den mit Abstand rechenintensivsten Teil des Algorithmus darstellt, sie beansprucht rund 93% der Rechenzeit. Ausgehend von dieser Beobachtung wird hier im nächsten Schritt nach weiteren Optimierungsmöglichkeiten gesucht.

Für die Überprüfung der beschriebenen Abbruchbedingung lässt sich ein divide-and-conquer-Ansatz nutzen. Da das Spektrum eines nicht verbundenen Graphen aus der Vereinigung der Spektren seiner Komponenten besteht, genügt es, für einen Graphen $G \setminus x$ zu bestimmen, ob für eine seiner Komponenten oder die Vereinigung einiger Komponenten zwei Eigenwerte echt zwischen zwei ganzen Zahlen liegen. In diesem Fall erfüllt auch $G \setminus x$ diese Bedingung und muss nicht weiter betrachtet werden.

Konkret wird für die Überprüfung der Abbruchbedingung ein möglichst zentraler Knoten des Baumes entfernt und die Spektren der entstandenen Komponenten bestimmt. Große Teilbäume werden rekursiv weiter geteilt. Dabei ist zu beachten, dass jedes Entfernen eines Knotens das Spektrum des Graphen ändert, entsprechend müssen bei mehrfachem Teilen $k + 1$ Eigenwerte zwischen zwei aufeinander folgenden Ganzzahlen liegen, wobei k die Anzahl der entfernten Knoten bezeichnet.

Um die Spektren derselben Teilkomponenten nicht mehrfach zu berechnen, wird ein Lookup-Table aus bereits berechneten Teilspektren aufgebaut, in dem Spektren nur noch nachgeschlagen werden müssen. Die maximale Größe der Teilbäume im Lookup-Table wird dynamisch festgelegt, ein hoher Wert bedeutet einen hohen Speicherbedarf für die Tabelle aber auch weniger Rekursionsschritte, mehr direkte Treffer im Cache und somit geringeren Zeitaufwand. Umgekehrt verbessert ein niedriger Wert einerseits den Speicherbedarf, führt andererseits aber dazu, dass weniger Bäume direkt aufgrund ihrer Teilspektren verworfen werden können. Abbildung 2 vergleicht die Laufzeiten der Spektrenberechnung mit und ohne die beschriebenen Optimierungen.

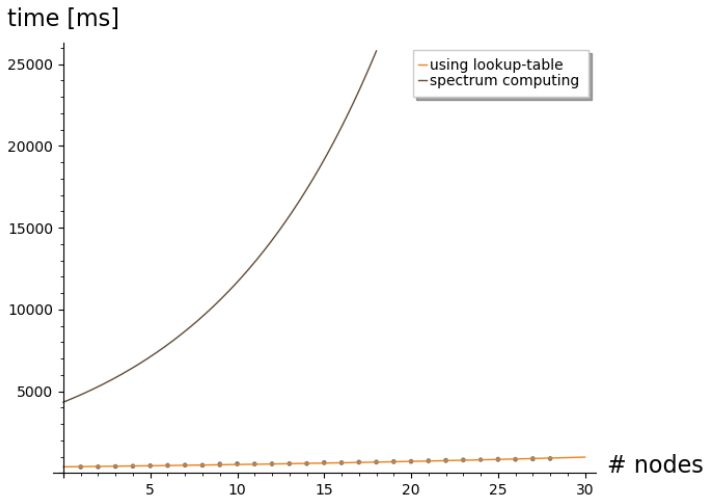


Abbildung 2: Vergleich des Zeitaufwand für die Spektrenberechnung.

3.3. Speicherbedarf

Der Speicherbedarf des Algorithmus wächst mit der Größe der untersuchten Bäume, da für die Überprüfung auf Isomorphie bei der Erzeugung neuer Bäume intern eine komplette Liste aller bisher erzeugten Bäume im Speicher gehalten werden muss. Um den Speicherbedarf zu minimieren werden die Bäume als möglichst kompakte (in Bezug auf den benötigten Speicherplatz) Datenobjekte repräsentiert. Listen von Bäumen, die nicht notwendigerweise komplett im Speicher vorgehalten werden müssen, werden als Dateien im System abgelegt und bei Bedarf sequentiell wieder eingelesen und direkt verarbeitet.

3.4. Parallelisierung

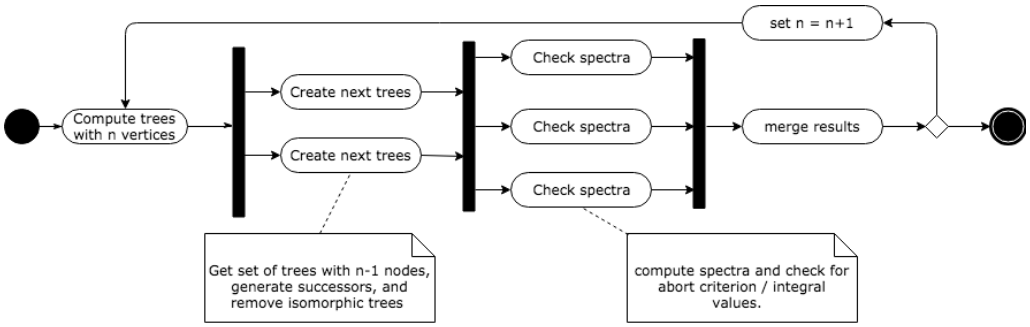


Abbildung 3: Schematische Darstellung der parallelen Verarbeitung.

Um den Algorithmus zu parallelisieren, wurde eine Pipeline aufgebaut, in der einzelne Verarbeitungsschritte von unterschiedlichen Prozessen ausgeführt werden. Jeder Prozess verarbeitet dabei die Ergebnisse seiner Vorgänger und gibt seine Ergebnisse an nachfolgende Prozesse weiter. Durch die schrittweise Verarbeitung lassen sich rechenintensive von weniger rechenintensiven Schritten trennen und die Anzahl der parallelen Prozesse für jeden Schritt dynamisch annpassen. Um den Kommunikations- und Synchronisationsaufwand zwischen den Prozessen möglichst gering zu halten, wird eine gewisse Redundanz in Kauf genommen und ursprünglich globale Datenstrukturen wie die Tabelle mit den vorberechneten Teilspektren für jeden Prozess aufgebaut. Abbildung 3 zeigt das schematische Vorgehen.

Ein Problem dieses Ansatzes ist, dass durch die fehlende Synchronisierung zwischen den parallel arbeitenden Prozessen einzelne Schritte redundant erledigt werden. Werden neue Bäume parallel erzeugt, können nicht mehr alle isomorphen Bäume erkannt werden, so dass derselbe Baum mehrfach überprüft wird. Der Mehraufwand, der abhängig von der Anzahl der erzeugenden Prozesse durch redundant erzeugte Bäume entsteht, wird in Abbildung 4 dargestellt. Wie die Kurve zeigt, steigt die Zahl der zusätzlich zu überprüfenden Bäume anfangs mit der Prozessanzahl stark an, sie konvergiert dann aber gegen einen Maximalwert.

4. Ergebnisse

Mit Hilfe der vorgenommenen Optimierungen war es möglich, eine vollständige Liste aller integralen Bäume bis 64 Knoten zu berechnen. Die Laufzeit des Algorithmus konnte durch verschiedene Maßnahmen von $10e^{1.1n}$ für die naive Brute-Force Implementierung auf $0.29e^{0.25n}$ für

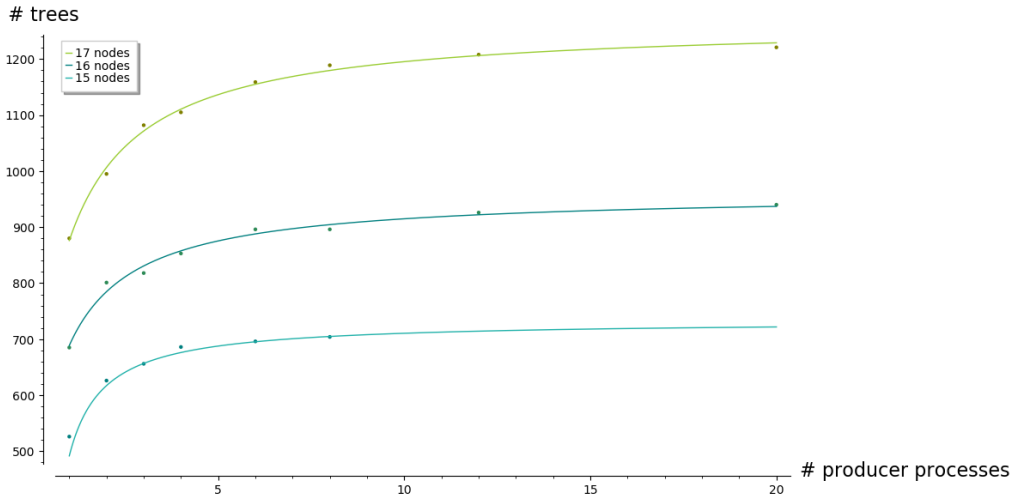


Abbildung 4: Mehraufwand durch parallele Erzeugung neuer Bäume. Dargestellt wird die Anzahl der zu überprüfenden Bäume in Abhängigkeit von der Anzahl der erzeugenden Prozesse.

die parallelisierte Version reduziert werden. Tabelle 1 gibt einen Überblick über die Laufzeiten verschiedener Varianten des Algorithmus.

Bis 50 Knoten entsprechen die Ergebnisse der Tabelle aus [1], im Bereich zwischen 50 und 60 Knoten wurden mit $T(1)*T(4,4)*T(2,1,3)*T(1,13,1)$ und $T(2)*T(3,4)*T(2,3,1)*T(1,13,1)$ zwei bisher unbekannte integrale Bäume mit 59 Knoten gefunden. Sie werden in Abbildung 5 dargestellt.

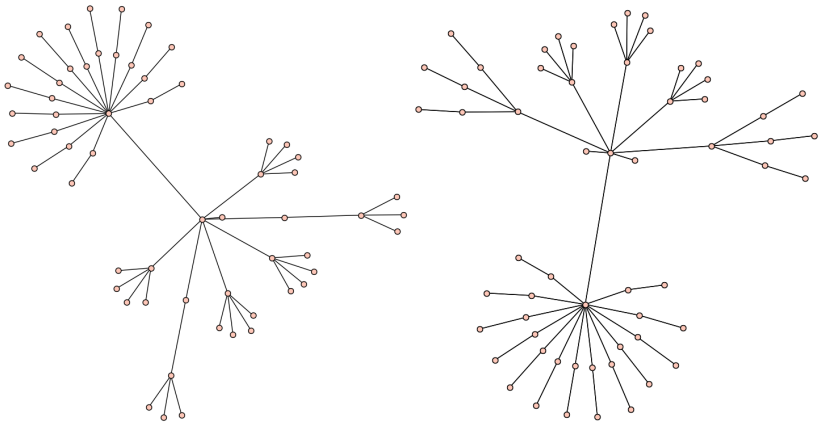
# Knoten	10	20	30	40	50	60
Brute Force	0.6 s	7 Std	30 J.	10^6J.	10^{10}J.	
Interlacing	0.7 s	2 min	4.4 Std	27 Tage	11 J.	
Dynamische Spektrenberechnung	1.9 s	51 s	23 min	10.5 Std	12 Tage	
Parallelisiert (6 Cores a 2.2GHz)	3.5 s	43 s	9 min	106 min	21.6 Std	9 Tage

Tabelle 1: Laufzeiten unterschiedlicher Varianten des Algorithmus

4.1. Häufigkeitsverteilung von Teilgraphen

Die für die Berechnung der Spektren genutzten Caches mit vorberechneten Teilspektren ermöglichen die Ermittlung von Trefferquoten. Das Programm zählt, welche Teilbäume wie oft nachgeschlagen werden, um so eventuelle Auffälligkeiten in der Verteilung der Komponenten zu erkennen.

Betrachtet man alle Teilbäume, scheint die Verteilung wie erwartet. Es gibt mehr Treffer bei Bäumen mit kleiner Knotenzahl und innerhalb der Bäume mit gleicher Knotenzahl werden Bäume mit kleinem Durchmesser häufiger nachgeschlagen. Abbildung 6 zeigt die Verteilung der Komponenten für alle Bäume mit 21 bis 23 Knoten.



(a) $(4, 3, 2^5, 1^{13}, 0^{19}, -1^{13}, -2^5, -3, -4)$ (b) $(4, 3, 2^4, 1^{17}, 0^{13}, -1^{17}, -2^4, -3, -4)$

Abbildung 5: Neue integrale Bäume und ihre Spektren

Betrachtet man nur integrale Bäume, fällt auf, dass die bisher gefundenen Bäume bis 62 Knoten alle in Komponenten zerfallen, die entweder selbst wieder ein integrales Spektrum besitzen oder einen sternförmigen Aufbau haben. Abbildung 7 zeigt die Verteilung, Tabelle 2 gibt einen Überblick über die aufgetretenen Teilbäume. Auch alle in [1] gelisteten größeren integralen Bäume lassen sich entsprechend zerlegen. Aus dieser Beobachtung lässt sich folgende Vermutung formulieren:

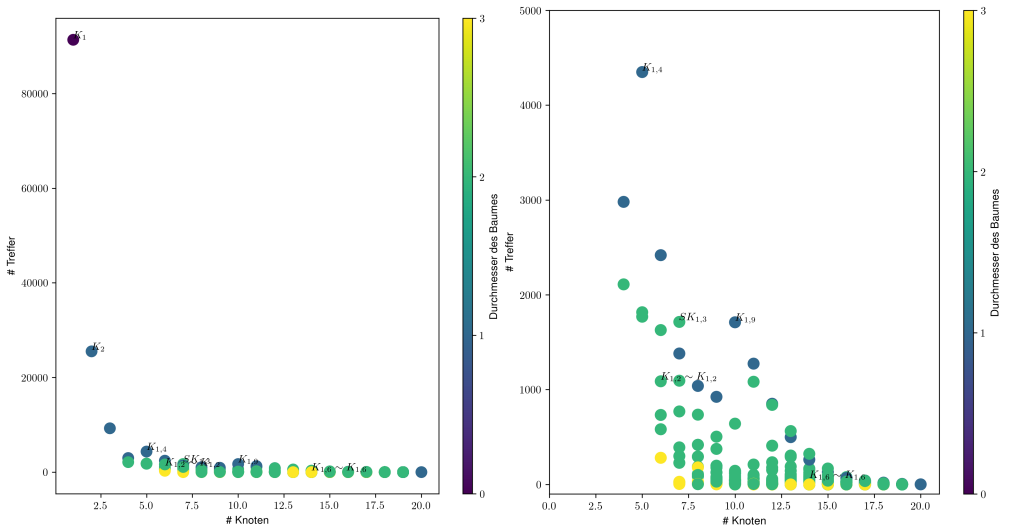
Vermutung 7. *Alle integralen Bäume lassen sich als Kombinationen aus kleineren integralen und sternförmigen Bäumen konstruieren.*

Wenn die Vermutung stimmt, müsste ein Algorithmus zur Suche integraler Bäume nur solche Kombinationen überprüfen, was die Suche deutlich beschleunigen würde.

5. Fazit

Aufbauend auf einer Idee aus [1] konnte im Rahmen dieser Arbeit eine Liste der integralen Bäume bis 64 Knoten berechnet werden.

Eine Frage, die nicht abschließend geklärt werden konnte, ist, ob diese Liste tatsächlich vollständig ist. Das Verfahren erzeugt Bäume iterativ indem bereits untersuchte Bäume jeweils um einen neuen Knoten erweitert werden. Erfüllt ein Baum die in Kapitel 3.1 beschriebene Abbruchbedingung, wird er verworfen und keine weiteren Bäume aus diesem Baum erzeugt. Aufgrund der beschriebenen Interlacing-Eigenschaft kann aus ihm im nächsten Schritt kein integraler Baum konstruiert werden. Würden neue Bäume nur durch einmaliges Verbinden bekannter Komponenten durch einen zusätzlichen Knoten konstruiert werden, wäre der Algorithmus korrekt und vollständig. Da die Bäume aber schrittweise aufgebaut werden, ist es durchaus möglich, nach $n > 1$ Schritten aus einem solchen 'Abbruchbaum' wieder einen integralen Baum zu konstruieren. Zu klären wäre, ob jeder integrale Baum so schrittweise aufgebaut werden kann, dass



(a) Alle Teilbäume

(b) Ausschnitt bis 5000 Treffer

Abbildung 6: Verteilung der Komponenten aller untersuchter Bäume mit 21 bis 23 Knoten. Häufig auftretende integrale Bäume sind mit ihrer Bezeichnung annotiert.

er von unserem Algorithmus gefunden wird, oder ob es tatsächlich vorkommen kann, dass die verwendete Abbruchbedingung im Suchbaum alle Wege zu einem integralen Baum abschneidet.

Ebenfalls interessant wäre zu klären, ob integrale Bäume tatsächlich nur aus integralen und sternförmigen Komponenten aufgebaut sind. Wenn die Vermutung stimmt, könnte darauf aufbauend ein deutlich schnellerer Suchalgorithmus entwickelt werden.

Eine weitere mögliche Optimierung des Algorithmus wäre, bei der numerischen Berechnung der Eigenwerte gerundete Werte zu verwenden. Wenn sich Fehlerschranken finden lassen, die sicherstellen, dass ganzzahlige Eigenwerte trotz Rundungsfehler gefunden werden, können schnellere Verfahren verwendet werden, die angenäherte Eigenwerte mit ausreichender Genauigkeit berechnen.

Anhang A. Überblick über integrale Bäume

Abbildung A.8 zeigt eine grafische Darstellung aller integralen Bäume mit höchstens 62 Knoten.

Literatur

[1] A. E. Brouwer, Small integral trees, Electr. J. Comb. 15 (2008).

[2] B. Richmond, A. Odlyzko, B. D McKay, Constant time generation of free trees, SIAM Journal on Computing 15 (1986) 540–548.

[3] N. J. A. Sloane, Sequence a000055 in sloane's on-line encyclopedia of integer sequences, http://oeis.org/A000055, 2009.

[4] W. H. Haemers, Interlacing eigenvalues and graphs, Linear Algebra and its Applications 226-228 (1995) 593 – 616. Honoring J.J.Seidel.

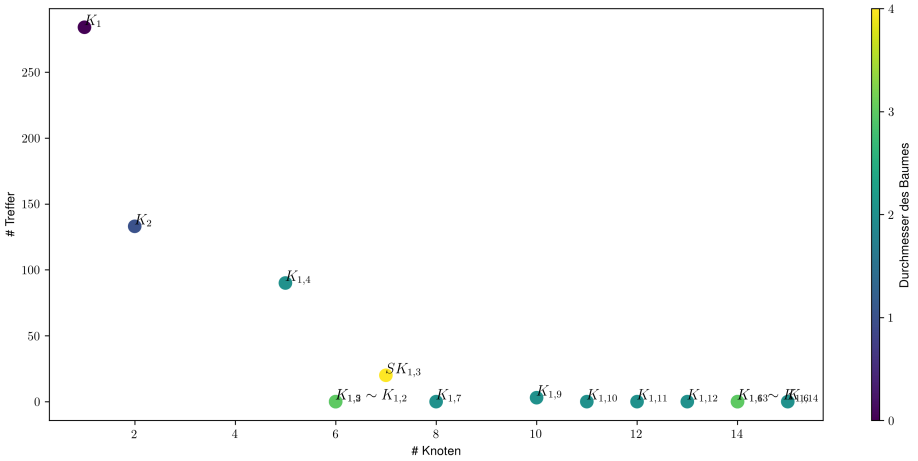


Abbildung 7: Verteilung der Komponenten integraler Bäume.

# Knoten	d	Bezeichnung	Spektrum	
1	0	K_1	0	integral
2	1	K_2	1	integral
5	2	$K_{1,4}$	$2, 0^3$	integral
6	3	$K_{1,2} \sim K_{1,2}$	$2, 1, 0^2$	integral
6	2	$K_{1,5}$	$\sqrt{5}, 0^4$	Stern
7	4	$SK_{1,3}$	$2, 1^2, 0$	integral
8	2	$K_{1,7}$	$\sqrt{7}, 0^6$	Stern
10	2	$K_{1,9}$	$3, 0^8$	integral
11	2	$K_{1,10}$	$\sqrt{10}, 0^9$	Stern
12	2	$K_{1,11}$	$\sqrt{11}, 0^{10}$	Stern
13	2	$K_{1,12}$	$\sqrt{12}, 0^{11}$	Stern
14	3	$K_{1,6} \sim K_{1,6}$	$3, 2, 0^{10}$	integral
14	2	$K_{1,13}$	$\sqrt{13}, 0^{12}$	Stern
15	2	$K_{1,14}$	$\sqrt{14}, 0^{13}$	Stern
17	2	$K_{1,16}$	$4, 0^{15}$	integral
17	4	$K_{1,7} \sim SK_{1,4}$	$3, 2, 1^3, 0^7$	integral
19	4	$K_{1,5} \sim SK_{1,6}$	$3, 2, 1^5, 0^5$	integral
20	2	$K_{1,19}$	$\sqrt{19}, 0^{18}$	Stern
21	2	$K_{1,20}$	$\sqrt{20}, 0^{19}$	Stern
...	...	...	...	...
27	4	$SK_{1,13}$	$\sqrt{14}, 1^1 2, 0$	unterteilter Stern

Tabelle 2: Komponenten integraler Bäume bis 62 Knoten. Spektren sind nur im positiven Bereich angegeben.

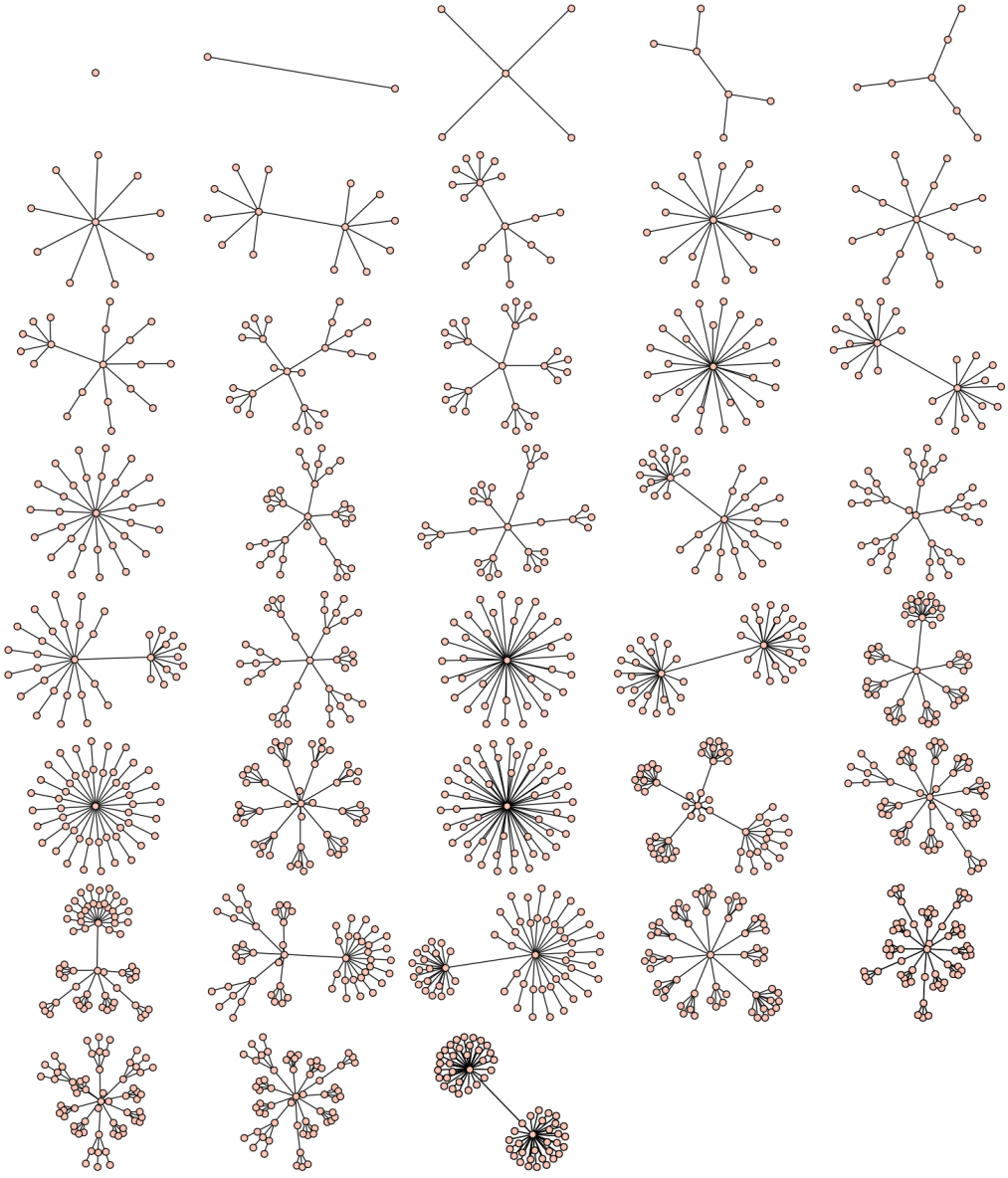


Abbildung A.8: Grafische Darstellung aller integralen Bäume bis 62 Knoten.