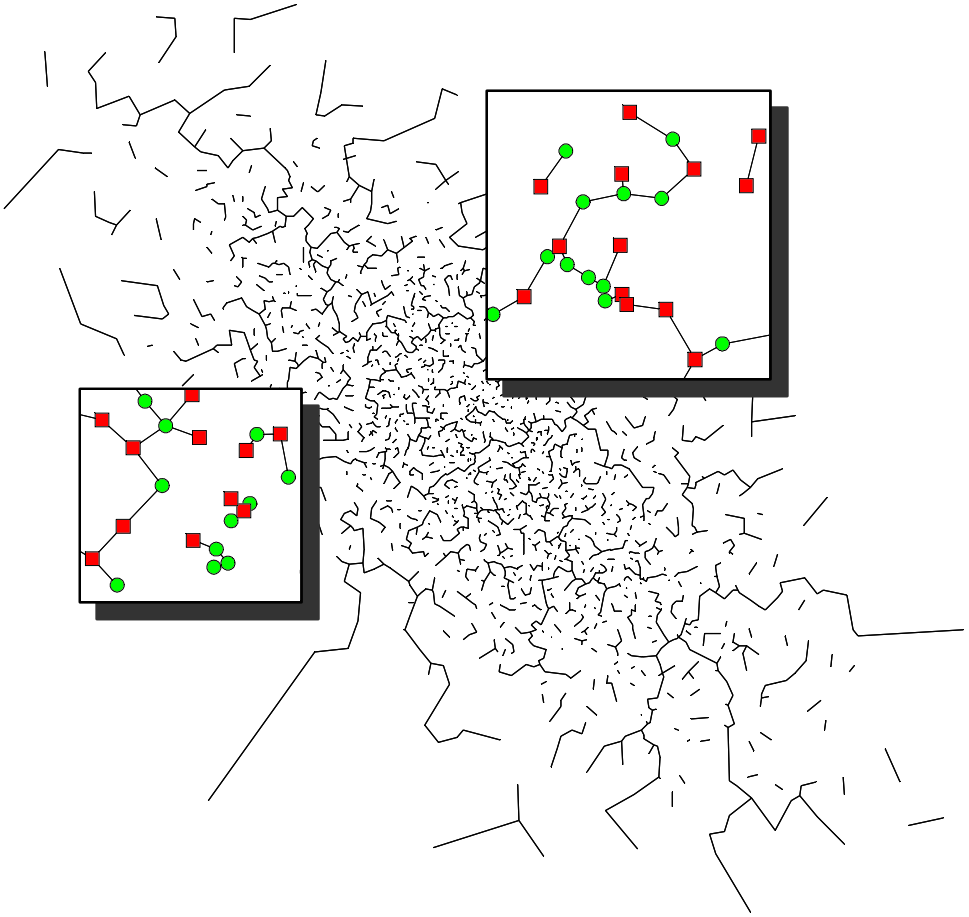
Approximative Matchings in Cross-Match-Tests

Studienarbeit von Tanja Hartmann
5. September 2007



Betreuer: Prof. Dorothea Wagner und Martin Nöllenburg

Institut für Theoretische Informatik
Universität Karlsruhe (TH)

Die Titelabbildung zeigt das Ergebnis der Waldberechnung des Matchingalgorithmus nach Wattenhofer und Wattenhofer für einen verteilungsgenerierten Graphen mit 3000 Knoten. Der zugrundeliegende Graph besteht aus Sicht des Cross-Match-Tests aus zwei standardnormalverteilten, zweidimensionalen Stichproben, wobei die erste Stichprobe 1200, die zweite 1800 Werte umfasst. Der Korrelationskoeffizient beträgt $-0{,}6$ für die erste, $-0{,}8$ für die zweite Stichprobe.

Danksagung

Mein Dank gebührt an dieser Stelle Frau Prof. Dorothea Wagner, die bei der Themenanfrage durch Herrn Prof. Henze an mich gedacht und so die Themenfindung angeregt hat. Außerdem möchte ich meinem Betreuer Martin Nöllenburg danken, der mir mit Rat und Tat zur Seite stand und mit großer Akribie und Sorgfalt meine Ausarbeitung kommentiert und korrigiert hat. Des Weiteren stand Bruno Ebner als Ansprechpartner für jegliche Fragen stochastischer Natur zur Verfügung und Devin Gharibian-Saki hat mir Einblick in seine Diplomarbeit gewährt. Hierfür den beiden ebenfalls herzlichen Dank. Mir hat diese Arbeit sehr viel Spaß und eine Fülle neuer Erfahrungen bereitet.

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig verfasst und die vorgestellten Ergebnisse ohne die Hilfe Dritter erarbeitet habe. Ich habe auf keine anderen als die angegebenen Quellen und Hilfsmittel zurückgegriffen.

Tanja Hartmann
Karlsruhe, den 5. September 2007

Zusammenfassung

Ziel dieser Arbeit war es zu untersuchen, ob neben dem exakten Matchingalgorithmus von Edmonds auch nicht exakte, dafür aber schnellere und einfacher zu implementierende Algorithmen für die Anwendung innerhalb des Cross-Match-Tests geeignet sind. Die experimentelle Untersuchung beschränkte sich dabei auf den Algorithmus nach Wattenhofer und Wattenhofer als Beispiel approximativer Matchingalgorithmen.

Der Cross-Match-Test ist ein nichtparametrischer Zweistichprobentest, der sich eines minimalen, perfekten Matchings bedient um eine geeignete Teststatistik zu definieren. Wie sich bei genauerer Betrachtung herausstellte, spielt dabei zwar das Streben nach Minimierung des Matchinggewichts, nicht jedoch die absolute Exaktheit eine Rolle. Statt dessen ist die Eindeutigkeit des Matchings für eine beliebige Punktmenge maßgebend.

Es konnte experimentell gezeigt werden, dass der untersuchte Approximationsalgorithmus die Konsistenz des Testverfahrens gewährleistet und mit geringem Aufwand so modifiziert werden kann, dass er auch die Bedingung der Eindeutigkeit des berechneten Matchings für eine beliebige Punktmenge erfüllt. Außerdem berechnete der untersuchte Algorithmus für alle getesteten Graphen eine sehr gute Approximation des exakten Matchings bei beachtlicher Laufzeitersparnis für große Knotenanzahlen. Damit bietet sich der Algorithmus nach Wattenhofer und Wattenhofer als schnelle Alternative zu Edmonds' Matchingalgorithmus für die Anwendung innerhalb des Cross-Match-Tests an.

Inhaltsverzeichnis

1 Einleitung **1**

1.1 Idee und Ziel des Cross-Match-Tests 2

1.1.1 Nichtparametrische Tests 2

1.1.2 Ein Anwendungsbeispiel 2

1.2 Problemstellung 4

2 Grundlagen **5**

2.1 Graphentheoretische Grundlagen 5

2.1.1 Allgemeine Definitionen 5

2.1.2 Matchings ungerichteter Graphen 6

2.2 Stochastische Grundlagen 7

3 Der Cross-Match-Test **10**

3.1 Null- und Alternativhypothese 10

3.2 Die Teststatistik 11

3.2.1 Die diskrete Verteilungsfunktion unter H_0 12

3.2.2 Analyse des Testverfahrens unter Voraussetzung nicht exakter Matchingalgorithmen 14

3.3 Der Cross-Match-Test als Graphenproblem 15

3.3.1 Edmonds' exakter Matchingalgorithmus 15

3.3.2 Erste Ideen zur Problemreduktion 17

4 Der Matchingalgorithmus nach Wattenhofer und Wattenhofer **19**

4.1 Algorithmenbeschreibung 19

4.1.1 Die Waldberechnung 19

4.1.2 Die Matchingberechnung 22

4.2 Analyse des Approximationsfaktors 23

4.3 Implementierung und Datenstrukturen 25

5 Experimenteller Algorithmenvergleich **27**

5.1 Versuchsdesign und automatisierte Testreihen 27

5.1.1 Getestete Verteilungsarten 27

5.1.2 Testreihenparameter 28

5.2 Verhaltensprüfung unter der Nullhypothese 29

5.3 Untersuchung des Konsistenzverhaltens 32

5.4 Auswertung des Eingangsbeispiels 34

5.5 Sonstige Beobachtungen 35

5.6 Laufzeitvergleich 37

6 Abschließende Bewertung **39**

6.1 Fazit 39

6.2 Ausblick 40

7 Anhang **41**

Literatur **46**

Kapitel 1

Einleitung

Als *Matching* oder auch *Paarung* eines beliebigen Graphen $G = (V, E)$ bezeichnet man eine Kantenmenge $M \subseteq E$, innerhalb derer keine zwei Kanten einen gemeinsamen Knoten besitzen. Ein *Matchingproblem* beschreibt allgemein die Suche nach einem Matching unter bestimmten Nebenbedingungen. Eine häufig geforderte Nebenbedingung ist die *Kardinalitätsmaximalität* des Matchings. Gesucht ist hierbei ein Matching, das unter allen Matchings die größte Kantenanzahl enthält. Unter einem *gewichtsmaximalen* Matching eines gewichteten Graphen versteht man ein Matching, dessen Summe aller Kantengewichte maximal ist unter allen Matchings.

Die Familie der Matchingprobleme nimmt in der Reihe der Graphenprobleme eine Sonderstellung ein. Anders als viele andere Problemklassen innerhalb der Graphentheorie sind Matchingprobleme auch für allgemeine Graphen mit zwar hohem asymptotischem, dennoch aber polynomiellem Aufwand lösbar. Außerdem findet sich eine Vielzahl von Anwendungen, die sich auf eine solche Problemstellung reduzieren lassen oder die Berechnung eines Matchings als Teilschritt zur Lösung eines übergeordneten Problems nutzen. Es verwundert daher nicht, dass nach und nach sowohl exakte als auch asymptotisch schnellere, approximative Algorithmen für nahezu jede Spielart der Matchingberechnung entwickelt wurden.

Grundlage all jener Algorithmen, die sich dem allgemeinen, gewichteten Matchingproblem widmen, sind das Edmonds-Theorem (siehe Abschnitt 3.3.1) und der aus dessen Beweis konstruktiv hervorgehende, polynomielle Algorithmus von Edmonds mit einer Laufzeit in $O(n^4)$, wobei n für die Anzahl der Knoten im Graphen steht (vergleiche [PS82]).

In dieser Arbeit wird auf eine spezielle Anwendung eines *gewichtsminimalen* Matchings in der Statistik eingegangen. Rosenbaum [Ros05] nutzt ein gewichtsminimales, *perfektes* Matching als Instrument zur Berechnung der Teststatistik eines Zweistichprobentests und entwickelt daraus den sogenannten *Cross-Match-Test*.

Die vorliegende Arbeit führt zunächst mit einem kurzen Anwendungsbeispiel des Cross-Match-Tests in die Thematik und Problemstellung ein, ohne näher auf die Funktion der Matchingberechung innerhalb des Cross-Match-Tests einzugehen. Anschließend werden die notwendigen theoretischen Grundlagen der beiden Bereiche Graphentheorie und Stochastik wiederholt. Kapitel 3 erläutert den Hintergrund des Cross-Match-Tests und schlägt den Bogen zur Betrachtung des Cross-Match-Tests als Graphenproblem. An dieser Stelle wird erstmals näher auf den Matchingalgorithmus von Edmonds eingegangen und es werden erste Lösungsideen bezüglich der Problemstellung diskutiert. Im darauf folgenden Kapitel 4 liegt der Schwerpunkt

auf dem in dieser Arbeit speziell untersuchten approximativen Matchingalgorithmus nach Wattenhofer und Wattenhofer [WW04]. Die Darstellung der Ergebnisse des experimentellen Algorithmenvergleichs in Kapitel 5 bildet schließlich den Abschluss der Arbeit.

1.1 Idee und Ziel des Cross-Match-Tests

Der Cross-Match-Test ist ein multivariater, nichtparametrischer Zweistichprobentest. Allen statistischen Tests ist das Ziel gemein, über die Verteilung einer oder mehrerer gegebener Stichproben gewisse Aussagen zu treffen. Dies können, unter der Annahme einer bestimmten Verteilungsart, Aussagen über Verteilungsparameter sein oder aber vergleichende Aussagen über mehrere, nicht näher spezifizierte Verteilungen.

1.1.1 Nichtparametrische Tests

Nichtparametrische Tests setzen, anders als parametrische Tests, außer der Existenz von Dichten keine weiteren Verteilungsannahmen der Stichproben voraus. Da der nichtparametrische Test nur jene Informationen extrahiert und auswertet, die aus gegenseitigen, relativen Größenbeziehungen der beiden Stichproben resultieren, ist er robuster als sein parametrisches Pendant gegenüber Störungen innerhalb der Daten [Hen95]. Statt nach Verteilungsparametern fragt ein nichtparametrischer Zweistichprobentest nach der Gleichheit der Verteilungen zweier Stichproben. Ziel eines solchen Tests ist es also, für die Verteilungen zweier mehrdimensionaler Stichproben zu entscheiden, ob die *Nullhypothese* oder die *Alternativhypothese* zutrifft. Dabei geht die Nullhypothese von der Gleichheit der Verteilungen aus, die Alternativhypothese dagegen betrachtet die Verteilungen als verschieden. Nichtparametrische Tests finden unter anderem in der Medizin und Biologie Anwendung.

Die zugrundeliegende Idee der Auswertung relativer Größenbeziehungen ist die Vorstellung, dass beliebigdimensionale, reellwertige Stichproben gleicher Verteilung, im Gegensatz zu verschieden verteilten Stichproben, mit großer Wahrscheinlichkeit stark durchmischt sind. Es bedarf also einer berechenbaren Kenngröße für das Maß der „Durchmischung“ der Stichproben, um eine Aussage über die Wahrscheinlichkeit der Gleichheit der beiden Stichprobenverteilungen treffen zu können. Die jeweilige Kenngröße wird auch *Teststatistik* genannt. Der in Kapitel 3 detailliert beschriebene Cross-Match-Test bedient sich des Matchingproblems der Graphentheorie um eine geeignete Teststatistik zu definieren und zu berechnen.

1.1.2 Ein Anwendungsbeispiel

Die Messwerte des folgenden Beispiels sind dem Artikel von Rosenbaum [Ros05] entnommen. Sie repräsentieren die relative Aktivität von linker und rechter Hirnhälfte, den sogenannten Hemisphären, gemessen an 18 Probanden mit Hilfe der Kernspintomographie. Untersucht wurden neun Patienten mit arteriovenösen Fehlbildungen in der linken Hemisphäre sowie eine Kontrollgruppe von neun gesunden Personen. Während der Messungen wurden den Probanden einmal Geschichten vorgelesen, ein anderes Mal wurden sie aufgefordert, einen gehörten Satz mental zu wiederholen. Das bildgebende Verfahren der Kernspintomographie kann die dadurch aktivierten Bereiche in beiden Hirnhälften sichtbar machen. So lässt sich aus dem Verhältnis $(L-R)/(L+R)$ der sogenannte *Lateralitätsindex* berechnen, wobei L für die Anzahl

der aktivierten Pixel der linken, R für die der rechten Hemisphäre steht. Die beiden so entstandenen Indexreihen können als Punktwolken in der Ebene aufgefasst werden. Jeder Punkt entspricht dann einem Probanden mit je zwei zugehörigen Messwerten als Koordinaten. In Abbildung 1.1 ist die Kontrollgruppe durch Kreise, die Patientengruppe durch Quadrate dargestellt.

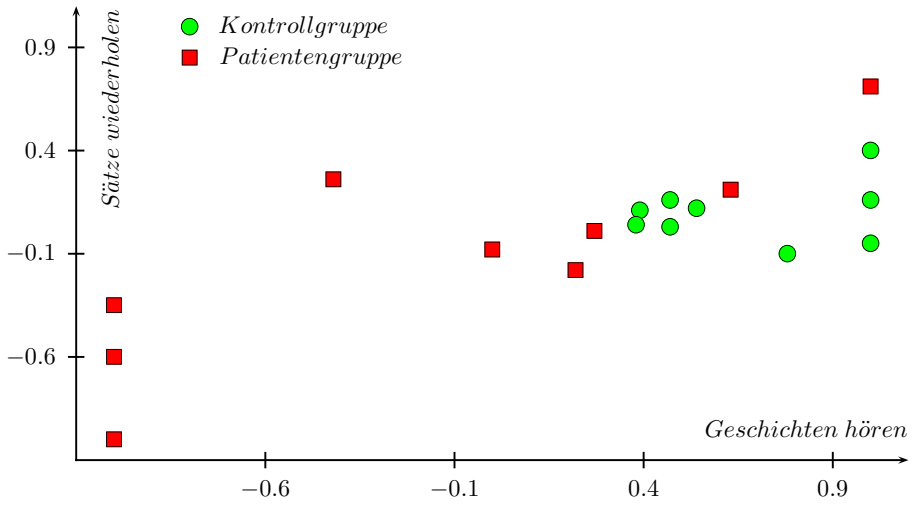


Abb. 1.1: Punktwolken aus Kernspinmesswerten.

Ziel dieser Untersuchung war es, eine Aussage zu treffen, inwieweit die Fehlbildungen die Hirnaktivität der Patienten beim Verarbeiten von Gehörtem beeinflussen. Falls die Fehlbildungen keinen Einfluss auf die Hirnaktivität haben, sind die Patienten in ihrem Verhalten nicht von der Kontrollgruppe zu unterscheiden und es ist somit zu erwarten, dass die Messwerte der Patientengruppe der gleichen Verteilung folgen wie die der Kontrollgruppe (Nullhypothese). Weichen die Patientenmesswerte dagegen von der Kontrollgruppenverteilung ab, so ist dies ein Indikator für die Beeinflussung der Hirnaktivität durch die arteriovenösen Fehlbildungen (Alternativhypothese).

Die Punktwolken in Abbildung 1.1 scheinen auf den ersten Blick nicht besonders stark durchmischt. Man kann daher zunächst von zwei verschiedenen Verteilungen der Kontroll- und Patientenmesswerte ausgehen. Dies ist jedoch eine subjektive, nicht quantifizierbare Einschätzung des Betrachters und funktioniert bei weniger eindeutigen Punktwolkenanordnungen nicht mehr so zuverlässig, wie es in diesem Beispiel den Anschein hat.

Der Cross-Match-Test ist ein Instrument, um mit einer quantifizierbaren Wahrscheinlichkeit, dem sogenannten *Niveau*, korrekt zu entscheiden, ob sich die beiden Verteilungen der gegebenen Stichproben (Kontroll- und Patientengruppe) gleichen. Die Entscheidung des Tests beruht dabei auf der Anzahl der gemischten Paare, sogenannter *Cross-Matches*, die durch die Berechnung eines distanzminimalen Matchings der als Punktwolken angeordneten Stichprobenwerte entstehen. Die Stichprobenzugehörigkeit wird bei der Matchingberechnung vernachlässigt. Als gemischt gelten jene Matchingpaare, deren Knoten verschiedenen Stichproben angehören. Desto stärker die Stichprobenpunktwolken durchmischt sind, desto mehr Cross-Matches sind zu erwarten. Die Entscheidung des Cross-Match-Tests für das hier vorgestellte Anwendungsbeispiel wird in Kapitel 5 erläutert.

1.2 Problemstellung

Um eine Aussage über die Gleichheit der Verteilungen zweier Stichproben zu treffen bedient sich der Cross-Match-Test der Matchingberechnung aus der Graphentheorie. Rosenbaum verweist in seinem Artikel [Ros05] auf eine Implementierung des Matchingalgorithmus von Edmonds in der Programmiersprache C [Rot]. Diese in übergeordnete Anwendungen einzubinden gestaltet sich jedoch mitunter sehr zeitaufwändig, da sie auf einer Dissertation von Gabow [Gab73, Gab76] basiert und speziell zugeschnittene Datenstrukturen nutzt, um die ohnehin recht komplexe Algorithmusidee von Edmonds (siehe Kapitel 3.3.1) statt mit einer Laufzeit in $O(n^4)$ in $O(n^3)$ zu realisieren. Die Konvertierung der Implementierung nach Gabow in eine andere Programmiersprache ist somit ebenfalls nicht trivial.

Außerdem ist für große Stichprobenumfänge eine schnellere als die hier verfügbare, kubische Laufzeit wünschenswert. Gabow gelingt es zwar konstruktiv nachzuweisen, dass Edmonds' Algorithmus sogar mit einer Laufzeit in $O(nm + n^2 \log n)$ implementiert werden kann [Gab90], die im Rahmen des Cross-Match-Tests betrachteten Graphen sind jedoch vollständig und es gilt somit $m \in O(n^2)$, wodurch sich nach wie vor eine kubische Laufzeit für die Matchingberechnung ergibt.

Vor diesem Hintergrund entstand die Frage nach der Eignung anderer, weniger komplexer Matchingalgorithmen für die Anwendung innerhalb des Cross-Match-Tests. Diese Arbeit untersucht im Speziellen das Verhalten des schnellen und einfachen Approximationsalgorithmus nach Wattenhofer und Wattenhofer [WW04] in Bezug auf den Cross-Match-Test und vergleicht dieses mit den Ergebnissen des exakten Algorithmus von Edmonds. Die Entscheidung für die nähere Untersuchung des Algorithmus nach Wattenhofer und Wattenhofer basiert unter anderem auf der Tatsache, dass dieser bereits in einer Diplomarbeit zur Untersuchung des nichtparametrischen Zweistichprobenproblems verwendet wurde, um den Cross-Match-Test zu realisieren [GS07].

Kapitel 2

Grundlagen

Das folgende Kapitel wiederholt die wichtigsten, für das Verständnis dieser Arbeit notwendigen Begriffe sowohl aus dem Bereich der Graphentheorie als auch aus dem Bereich der Stochastik.

2.1 Graphentheoretische Grundlagen

Die graphentheoretischen Grundlagen gliedern sich in allgemeine Definitionen sowie in Begriffe, die speziell der Formulierung von Matchingproblemen dienen. Dabei wurden die allgemeinen Definitionen unter anderem deshalb in die Grundlagenwiederholung aufgenommen, weil viele Objekte der Graphentheorie in der Literatur keine einheitliche Bezeichnung besitzen. Umgekehrt kann ein und derselbe Begriff je nach Autor verschieden definiert sein.

2.1.1 Allgemeine Definitionen

Ein Graph bezeichnet zunächst eine endliche Menge von Objekten, auch als *Knoten* bezeichnet, zusammen mit einer auf dieser Knotenmenge definierten Menge von Knotenpaaren, von denen jedes eine *Kante* zwischen den entsprechenden Knoten definiert. Diese Arbeit betrachtet ausschließlich *ungerichtete* Graphen. Die Kanten ungerichteter Graphen werden durch zweielementige Knotenmengen repräsentiert.

Definition 2.1 *Sei V eine gegebene Knotenmenge, E bezeichne die zugehörige, auf V definierte Kantenmenge.*
Für $E \subseteq \binom{V}{2}$ definieren V und E einen ungerichteten Graphen $G = (V, E)$*, wobei $\binom{V}{2}$ für die Menge aller zweielementigen Teilmengen von V steht. Die Menge $\{v, u\} \in E$ repräsentiert eine* ungerichtete *Kante.*

Die Kardinalitäten der Knoten- und Kantenmengen werden mit $n := |V|$ und $m := |E|$ abgekürzt. In Zusammenhängen, in denen die Unterscheidung in gerichtet und ungerichtet unerheblich ist oder aus dem Kontext hervorgeht, spricht man allgemein von einem Graphen und nutzt für Kanten die vereinfachte Schreibweise „$vu \in E$“ oder „$e \in E$“. Zwei Knoten v und u, die eine Kante $vu \in E$ definieren, heißen *adjazent*, die Kante vu heißt *inzident* zu v und u.

Neben der Richtung können weitere Eigenschaften der Kanten definiert werden. Eine oft gebrauchte Kanteneigenschaft ist das *Gewicht* einer Kante. Formal spricht

man von einer auf den Kanten definierten *Gewichtungsfunktion* und bezeichnet den Graphen als *gewichtet*.

Definition 2.2 *Sei $G = (V, E)$ ein Graph mit* Gewichtungsfunktion $w : E \to X \subseteq \mathbb{R}$. *Der Graph G heißt* gewichtet*, das* Gewicht *einer Kante $e \in E$ ist definiert durch $w(e) \in X$.*

Viele Algorithmen auf gewichteten Graphen setzten einen bestimmten, eingeschränkten Wertebereich der Gewichte voraus. Diese Arbeit betrachtet ausschließlich nichtnegativ gewichtete Graphen, das heißt es gilt $X = \mathbb{R}_{\geq 0}$.

Als eine Eigenschaft des gesamten Graphen kann man die Kantenanzahl betrachten. Die *vollständigen Graphen* nehmen diesbezüglich eine Sonderstellung ein.

Definition 2.3 *Ein ungerichteter Graph $G = (V, E)$ heißt* vollständig*, wenn gilt:*

$$E = \binom{V}{2} .$$

Da diese Arbeit nur ungerichtete Graphen betrachtet, wird hier die Vollständigkeit nur für solche Graphen definiert. Vollständige Graphen haben maximale Kantenanzahl, das heißt, jeder Knoten ist adjazent zu allen anderen Knoten. Ein solcher Graph hat also $m = |E| = \binom{n}{2} = n(n-1)/2 \in O(n^2)$ Kanten. Die Kantenanzahl verhält sich also quadratisch zur Anzahl der Knoten.

2.1.2 Matchings ungerichteter Graphen

Unter einem Matching eines ungerichteten Graphen versteht man eine symmetrische, disjunkte Paarung von Knoten, wobei nur adjazente Knoten gepaart werden dürfen. Ein Matching heißt perfekt, wenn es alle Knoten des zugrundeliegenden Graphen überdeckt. Die formale Definition versteht unter einem Matching eine Kantenteilmenge.

Definition 2.4 *Sei $G = (V, E)$ ein ungerichteter Graph. Ein* Matching $M \subseteq E$ *ist eine Kantenteilmenge, für die gilt:*

$$\forall \{v_1, u_1\}, \{v_2, u_2\} \in M : \quad \{v_1, u_1\} \cap \{v_2, u_2\} = \emptyset .$$

Das Matching M heißt perfekt*, wenn gilt:*

$$\forall v \in V \; \exists u \in V : \quad \{v, u\} \in M .$$

Die Berechnung von Matchings ist ein Optimierungsproblem. Für ungerichtete Graphen sucht man kardinalitätsmaximale Matchings. Betrachtet man dagegen gewichtete Graphen, so lässt sich die Summe aller Kantengewichte in einem Matching auffassen als das *Gewicht des Matchings*. So ergibt sich die Frage nach einem gewichtsminimalen oder gewichtsmaximalen Matching unter eventuell weiteren Nebenbedingungen.

Definition 2.5 *Sei $G = (V, E)$ ein ungerichteter, gewichteter Graph mit Gewichtungsfunktion $w : E \to X \subseteq \mathbb{R}$. Das* Gewicht $w(M)$ *eines Matchings $M \subseteq E$ ist definiert als:*

$$w(M) := \sum_{e \in M} w(e) .$$

Ein Matching M heißt gewichtsminimal *bzw.* gewichtsmaximal*, wenn gilt:*

$$w(M) = min\{w(M)|M \text{ ist Matching von } G\}$$

$$w(M) = max\{w(M)|M \text{ ist Matching von } G\} \ .$$

Für nichtnegative Gewichtungsfunktionen ergänzt man oftmals die Definition des gewichtsminimalen Matchings durch die Bedingung der Nichterweiterbarkeit des Matchings, da sonst bereits die leere Menge ein gewichtsminimales Matching ergäbe. Ein gewichtsmaximales Matching ist für nichtnegative Gewichtungsfunktionen ohnehin nur durch nullgewichtete Kanten erweiterbar. Die Nichterweiterbarkeit erzwingt für vollständige Graphen mit gerader Knotenanzahl die Perfektion des Matchings.

Die Suche nach einem gewichtsminimalen Matching $M_{\min}$ eines Graphen $G = (V, E)$ mit nichtnegativer Gewichtungsfunktion w lässt sich mit Hilfe der Berechnung eines gewichtsmaximalen Matchings $M_{\max}$ bezüglich der Gewichtungsfunktion $\hat{w}(e) = W - w(e)$, mit $W := \max\{w(e)|e \in E\}$, lösen. Das *allgemeine, nichtnegativ gewichtete Matchingproblem* ist demnach ein duales Problem, das sich sowohl als Maximierungsproblem als auch als Minimierungsproblem formulieren lässt. Gegenstand dieser Arbeit sind im Folgenden gewichtsminimale Matchings von vollständigen, ungerichteten, nichtnegativ gewichteten Graphen mit gerader Knotenanzahl.

2.2 Stochastische Grundlagen

Ziel dieser Grundlagenwiederholung ist es, einige der oftmals intuitiv gebrauchten Begriffe, wie *Verteilung* oder *Zufallsvariable*, durch formale Definitionen zu konkretisieren, ohne den Anspruch auf Vollständigkeit zu erheben. So fließt in Kapitel 3 auch der ein oder andere Begriff ein, der hier nicht definiert wurde, da er für das Verständnis des Cross-Match-Tests nicht zwingend ist. Spezielle Annahmen, die unter anderem für die Konsistenz des Cross-Match-Tests notwendig sind, werden bei der theoretischen Betrachtung des Tests in Kapitel 3 erwähnt, eine detaillierte Erläuterung der in diesem Zusammenhang vorkommenden Begriffe würde jedoch zu weit führen. Eine umfangreiche Einführung in die Grundlagen der Stochastik bietet zum Beispiel das Buch von Krengel [Kre91]. Auf die Wiederholung von Grundlagen aus dem Bereich der euklidischen Geometrie wurde ebenfalls verzichtet.

Die erste Definition widmet sich dem *kontinuierlichen Wahrscheinlichkeitsraum* bestehend aus einem *Grundraum* Ω, einer *σ-Algebra* $\mathcal{A}$ und dem *Wahrscheinlichkeitsmaß* P.

Definition 2.6 *Ein* kontinuierlicher Wahrscheinlichkeitsraum *ist ein 3-Tupel $(\Omega, \mathcal{A}, P)$, wobei Ω eine überabzählbare Menge ist, die den* Grundraum *beschreibt, $\mathcal{A} \subset \mathcal{P}(\Omega)$ eine* σ-Algebra *und $P : \mathcal{A} \longrightarrow [0, 1]$ das* Wahrscheinlichkeitsmaß *auf $\mathcal{A}$ in Form einer reellwertigen Funktion mit folgenden Eigenschaften:*

(a) $P(A) \geq 0 \quad \forall A \in \mathcal{A}$ (Nichtnegativität)

(b) $P(\Omega) = 1$ (Normiertheit)

(c) $P(\sum_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i)$ (σ-Additivität)

für jede Folge $(A_i)_{i\in\mathbb{N}}$ paarweise disjunkter Ereignisse aus $\mathcal{A}$, wobei $\sum A_i$ für die Vereinigung paarweiser disjunkter Ereignisse steht.

Die Teilmengen des Grundraums Ω stellen *Ereignisse* dar, wobei sich mehrelementige Teilmengen aus einelementigen Teilmengen, den *Elementarereignissen*, zusammensetzen. Die σ-Algebra $\mathcal{A}$ ist eine Teilmenge der Potenzmenge von Ω und somit eine Menge von Ereignissen. Sie enthält nach Definition den Grundraum sowie die leere Menge und ist abgeschlossen bezüglich endlichem Schnitt, endlicher Vereinigung und Komplementbildung. Das Wahrscheinlichkeitsmaß P definiert die Eintrittswahrscheinlichkeiten der Ereignisse in $\mathcal{A}$.

In kontinuierlichen Wahrscheinlichkeitsräumen folgt aus der Normiertheit eine Eintrittswahrscheinlichkeit von Null für Elementarereignisse. Die gleichzeitige Erfüllung der σ-Additivität erzwingt laut Maßtheorie die Definition des Wahrscheinlichkeitsmaßes P auf einer σ-Algebra $\mathcal{A}$ als echter Teilmenge der Potenzmenge des Grundraums Ω [Hen95]. *Diskreten Wahrscheinlichkeitsräumen* (Ω, P) liegt eine abzählbare Menge Ω zugrunde. Sie bedürfen keiner σ-Algebra. Ihr Wahrscheinlichkeitsmaß P ist auf der gesamten Potenzmenge $\mathcal{P}(\Omega)$ definiert und es gibt mindestens ein Elementarereignis, dessen Eintrittswahrscheinlichkeit echt größer als Null ist.

Das Wahrscheinlichkeitsmaß jedes Wahrscheinlichkeitsraums lässt sich auffassen als die *Verteilung* der *Zufallsvariablen*, die der Identität auf dem Grundraum Ω entspricht.

Definition 2.7 *Sei $(\Omega, \mathcal{A}, P)$ ein kontinuierlicher Wahrscheinlichkeitsraum, Ω' eine nichtleere Menge mit einer σ-Algebra $\mathcal{A}' \subset \mathcal{P}(\Omega')$ und $X : \Omega \longrightarrow \Omega'$ eine Abbildung. Die Abbildung X heißt* Ω'-wertige Zufallsvariable *und ist* $(\mathcal{A}, \mathcal{A}')$-messbar*, wenn gilt:*

$$X^{-1}(A') \in \mathcal{A} \quad \forall A' \in \mathcal{A}' \; .$$

Diskrete Zufallsvariablen und ihre Verteilungen sind analog definiert, wobei die Existenz der σ-Algebren $\mathcal{A}$ und $\mathcal{A}'$ sowie die Messbarkeit als Bedingung entfallen.

Definition 2.8 *Sei $X : \Omega \longrightarrow \Omega'$ eine $(\mathcal{A}, \mathcal{A}')$-messbare Zufallsvariable. Die* Verteilung P^X *von X ist ein Wahrscheinlichkeitsmaß auf der σ-Algebra $\mathcal{A}'$ und wird definiert durch:*

$$P^X : \mathcal{A}' \longrightarrow \mathbb{R}, \quad A' \longmapsto P^X(A') := P(X^{-1}(A')) \; .$$

Für die Identität $X : \Omega \longrightarrow \Omega$ gilt $X^{-1}(A) = A$, für alle $A \in \mathcal{A}$. Die Zufallsvariable X ist damit $(\mathcal{A}, \mathcal{A}')$-messbar und für ihre Verteilung gilt $P^X = P$. Die Elemente des Grundraums Ω werden als *Stichproben* bezeichnet, die Bilder der Stichproben unter einer Zufallsvariablen X als *Realisierungen* der Zufallsvariablen.

Diese Arbeit betrachtet Wahrscheinlichkeitsräume über dem Grundraum $\mathbb{R}^k$ $(k \geq 2)$ mit der σ-Algebra $\mathcal{B}^k$ der *Borelmengen*, welche von den offenen Mengen des $\mathbb{R}^k$ erzeugt wird. Das Wahrscheinlichkeitsmaß P eines solchen Raums entspricht also der Verteilung der Zufallsvariablen $X : \mathbb{R}^k \longrightarrow \mathbb{R}^k$ mit $X(v) = v$. Oftmals schreibt man auch $X \in \mathbb{R}^k$ und meint damit die Realisierungen von X.

Definition 2.9 *Die* Dichtefunktion f *des Wahrscheinlichkeitsmaßes P eines kontinuierlichen Wahrscheinlichkeitsraums $(\mathbb{R}^k, \mathcal{B}^k, P)$ ist definiert als Riemann-integrierbare Funktion $f : \mathbb{R}^k \longrightarrow \mathbb{R}$ mit*

$$\int_{-\infty}^{\infty} f(t)\,\mathrm{d}t = 1 \quad \textit{und}$$

$$\int_{-\infty}^{x} f(t)\,\mathrm{d}t = P(A), \quad x = \begin{pmatrix} x_1 \\ \vdots \\ x_k \end{pmatrix} \in \mathbb{R}^k,$$

$$\textit{mit offenem Würfel} \quad A := \{v \in \mathbb{R}^k \mid v_i \leq x_i,\ i = 1, \ldots, k\}.$$

Für die Zufallsvariable $X : \mathbb{R}^k \longrightarrow \mathbb{R}^k$ mit $X(v) = v$ ist dann $P(X \leq x) = \int_{-\infty}^{x} f(t)\,\mathrm{d}t$. Damit induziert X die *Verteilungsfunktion* $F : \mathbb{R}^k \longrightarrow [0,1]$ mit

$$F(x) := P(X \leq x) = \int_{-\infty}^{x} f(t)\,\mathrm{d}t\ .$$

Für diskrete Zufallsvariablen $X : \Omega \longrightarrow \mathbb{R}$ gilt analog $P(X = x) = \sum_{\omega \in X^{-1}(x)} f(\omega)$ mit *Wahrscheinlichkeitsfunktion* f.

Definition 2.10 *Die* Wahrscheinlichkeitsfunktion f *des Wahrscheinlichkeitsmaßes* P *eines diskreten Wahrscheinlichkeitsraums* (Ω, P) *ist definiert als* $f : \Omega \longrightarrow \mathbb{R}$ *mit*

$$\sum_{\omega \in \Omega} f(\omega) = 1 \quad \textit{und}$$
$$\sum_{\omega \in A} f(\omega) = P(A), \quad A \subseteq \Omega\ .$$

Die *diskrete Verteilungsfunktion* $F : \mathbb{R} \longrightarrow [0,1]$ ist definiert als

$$F(x) := P(X = x) = \sum_{\omega \in X^{-1}(x)} f(\omega)\ .$$

Kapitel 3

Der Cross-Match-Test

Um die Eignung verschiedener Matchingalgorithmen für den Cross-Match-Test beurteilen zu können, ist es notwendig, den theoretischen Hintergrund des Tests und seiner Teststatistik etwas genauer zu beleuchten. Dieses Kapitel konkretisiert die in Abschnitt 1.1 dargestellte Idee des Cross-Match-Tests und zeigt, wie Rosenbaum [Ros05] mit Hilfe der Matchingberechnung eine geeignete Teststatistik definiert.

Der Cross-Match-Test ist ein nichtparametrischer, multivariater Zweistichprobentest. Da nichtparametrische Tests keine Verteilungsannahmen machen eignen sie sich insbesondere für Stichproben, für die die Voraussetzung einer gewissen Verteilungsart nicht sinnvoll erscheint. Bewährte nichtparametrische Zweistichprobentests für eindimensionale Stichproben sind zum Beispiel der *Wald-Wolfowitz-Runs-Test* oder der *Wilcoxon-Rangsummentest* [GC03]. Für mehrdimensionale Stichproben verallgemeinerten Friedman und Rafsky [FR79] den Wald-Wolfowitz-Runs-Test unter Verwendung eines *minimalen Spannbaums* (MST), womit bereits 1979 eine graphentheoretische Problemstellung innerhalb eines stochastischen Tests verwendet wurde. Später kamen Testverfahren basierend auf einer *k-nächste-Nachbarn*-Berechnung (k-NN) hinzu, [Sch86, Hen88], und schließlich fand die Verwendung von gewichtsminimalen Matchings Eingang in diesen Themenbereich.

3.1 Null- und Alternativhypothese

Für den Cross-Match-Test sind zwei multivariate, reellwertige Stichproben gegeben. Dabei bezeichne s die Anzahl der Stichprobenwerte der ersten Stichprobe, t die der zweiten Stichprobe. Insgesamt betrachtet man also $g := s + t$ Werte, wobei g als gerade vorausgesetzt wird. Die Stichprobenwerte können sowohl unendlichdimensional sein, wie zum Beispiel kontinuierliche Kurven, als auch endlichdimensional, wie zum Beispiel diskrete Messreihen über eine endliche Anzahl von Merkmalen. In dieser Arbeit werden ausschließlich endlichdimensionale Stichproben untersucht.

Formal bewegt sich der Cross-Match-Test in den kontinuierlichen Wahrscheinlichkeitsräumen $(\mathbb{R}^k, \mathcal{B}^k, P_1)$ der ersten und $(\mathbb{R}^k, \mathcal{B}^k, P_2)$ der zweiten Stichprobe. Die s gemessenen Stichprobenwerte der ersten Stichprobe werden interpretiert als Realisierungen x_i von Zufallsvariablen $X_i,\ i = 1, \ldots, s$, die jeweils der Identität auf dem Grundraum $\mathbb{R}^k$ entsprechen und alle die Verteilung $F(\cdot) = P_1$ haben. Analoges gilt für die Stichprobenwerte der zweiten Stichprobe, die den Zufallsvariablen $Y_i,\ i = 1, \ldots, t$, mit Verteilung $G(\cdot) = P_2$ und Realisierungen y_i zugeordnet sind. Statt von Zufallsvariablen auf dem Grundraum $\mathbb{R}^k$ spricht man auch anschaulicher

von *Zufallsvektoren* mit Vektoren aus $\mathbb{R}^k$ als Realisierungen. Um von der Gesamtheit aller X_i und Y_i sprechen zu können, fasst man diese zu Hilfsgrößen $Z_i,\ i = 1, \ldots, g$, zusammen.

Als Voraussetzung wird unterstellt, dass die beiden unbekannten Wahrscheinlichkeitsmaße P_1 und P_2 auf der σ-Algebra $\mathcal{B}^k$ der Borelmengen definiert sind und alle Zufallsvariablen Z_i stochastisch unabhängig sind. Formal gilt also:

$$\begin{array}{lll} X_i \overset{u.i.v.}{\sim} F(\cdot) & i = 1, \ldots, s & \mathcal{X} := \{X_1, \ldots, X_s\} \\ Y_i \overset{u.i.v.}{\sim} G(\cdot) & i = 1, \ldots, t & \mathcal{Y} := \{Y_1, \ldots, Y_t\} \\ Z_i := \begin{cases} X_i, & i = 1, \ldots, s \\ Y_{i-s}, & i = (s+1), \ldots, g \end{cases} & & \mathcal{Z} := \{Z_1, \ldots, Z_g\}\ . \end{array}$$

Um ausschließen zu können, dass innerhalb einer Stichprobe identische Vektoren $z_i = z_j,\ i \neq j$, auftreten, wird außerdem die *Stetigkeit* der Verteilungen $F(\cdot)$ und $G(\cdot)$ vorausgesetzt.

Die Nullhypothese H_0 geht nun von der Gleichheit der beiden Stichprobenverteilungen $F(\cdot)$ und $G(\cdot)$ aus, die Alternativhypothese H_1 entsprechend von der Ungleichheit. Unter der Nullhypothese H_0 können also identische Vektoren nicht nur innerhalb einer Stichprobe, sondern für die Gesamtheit aller betrachteten Realisierungen $z_i, \ldots, z_g$ ausgeschlossen werden. Der Cross-Match-Test testet die Hypothese $H_0 : F(\cdot) = G(\cdot)$ gegen die Alternative $H_1 : F(\cdot) \neq G(\cdot)$.

3.2 Die Teststatistik

Der zentrale Gedanke des Cross-Match-Tests ist die Definition einer Teststatistik mit Hilfe der Matchingberechnung. Dabei werden die Vektoren z_i unabhängig von ihrer Stichprobenzugehörigkeit als Datenpunkte in $\mathbb{R}^k$ betrachtet. Da jede *Norm* $\| \cdot \|$ auf $\mathbb{R}^k$ eine *Metrik* induziert, kann mit deren Hilfe ein Distanzmaß $d(y_i, y_j) := \| y_i - y_j \|$ auf diesen Datenpunkten definiert werden. Der experimentelle Algorithmenvergleich in Kapitel 5 beschränkt sich hierbei auf die Verwendung der *euklidischen Norm*. Da nach Voraussetzung identische Vektoren $z_i = z_j,\ i \neq j$, ausgeschlossen sind, gilt $d(y_i, y_j) > 0$ für $i \neq j$.

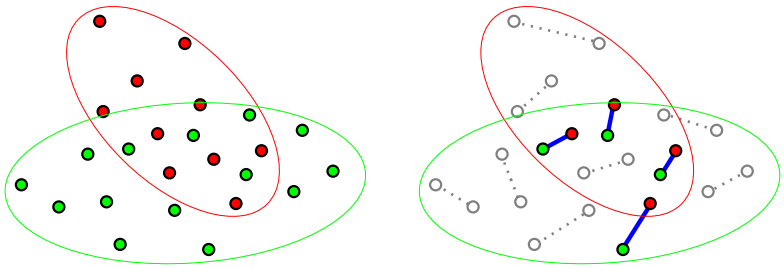


Abb. 3.1: Cross-Matches eines minimalen Matchings von Stichprobenpunktwolken.

Rosenbaum berechnet nun disjunkte Paarungen der Datenpunkte so, dass die Summe aller Paarungsdistanzen bezüglich des Distanzmaßes d minimal wird (siehe Abbildung 3.1). Dies entspricht der Berechnung eines gewichtsminimalen, perfekten Matchings des bezüglich d gewichteten, vollständigen Graphen über allen Datenpunkten. Dabei beschreibt $h := g/2$ die Kardinalität dieses Matchings.

Unter Berücksichtigung der Stichprobenzugehörigkeit erwartet man intuitiv, dass bei einer guten „Durchmischung“ beider Stichproben ein distanzminimales Matching häufig zwei Punkte aus verschiedenen Stichproben paart. Umgekehrt erwartet man bei deutlich getrennt liegenden Punktwolken ein geringes Auftreten von gemischten Paaren, sogenannten *Cross-Matches*. Die Anzahl A_1 der Cross-Matches in einem distanzminimalen Matching erscheint somit intuitiv als geeignete Kenngröße für das Maß der „Stichprobendurchmischung“. Dass die so definierte Teststatistik A_1 tatsächlich eine aussagekräftige Größe ist, zeigt im Folgenden die Untersuchung ihrer Verteilungsfunktion unter der Nullhypothese H_0.

3.2.1 Die diskrete Verteilungsfunktion unter H_0

Die Berechnung der diskreten Verteilungsfunktion Q von A_1 unter der Nullhypothese setzt die Einführung weiterer Zufallsvariablen U_i als Indikatoren für die Stichprobenzugehörigkeit der Z_i voraus. Es gilt

$$U_i = \begin{cases} 1, & Z_i \in \mathcal{X} \\ 0, & Z_i \in \mathcal{Y} \end{cases}.$$

Für die Menge $z := \{z_1, \ldots, z_g\}$ aller durch Messung entstandenen Datenpunkte existiert ein eindeutiges gewichtsminimales Matching, da die Voraussetzung stetiger Verteilungsfunktionen $F(\cdot)$ und $G(\cdot)$ (siehe Abschnitt 3.1) neben der Existenz identischer Datenpunkte auch verhindert, dass zwei Datenpunkte bezüglich d den gleichen Abstand zu einem dritten Punkt haben. Man kann also die Indizes der Datenpunkte so umsortieren und die neu indizierten Vektoren in ein geordnetes Tupel $(z_1, \ldots, z_g)$ schreiben, dass die Mengen $\{z_{2k-1}, z_{2k}\},\ k \in \{1, \ldots, h\}$, die Kanten des zugehörigen eindeutigen Matchings repräsentieren. Dies wiederum induziert eine entsprechende Umsortierung der Indizes der Z_i und U_i und deren Auflistung zu geordneten Tupeln $Z := (Z_1, \ldots, Z_g)$ und $U := (U_1, \ldots, U_g) \in \{0,1\}^g$. Dabei beschreibt U die Stichprobenzuordnung aller Z_i. Der Indikator $G_k \in \{0,1\}$ für eine Matchingkante $\{Z_{2k-1}, Z_{2k}\}$ an der Stelle k ist dann gegeben durch $G_k := U_{2k-1} + U_{2k} - 2U_{2k-1}U_{2k}$ und es gilt:

$$G_k = \begin{cases} 1, & \{Z_{2k-1}, Z_{2k}\} \text{ ist Cross-Match-Kante} \\ 0, & \text{sonst} \end{cases}.$$

Die Teststatistik A_1 lautet formal

$$A_1 := \sum_{k=1}^{h} G_k \ .$$

Das Messen von Stichproben aus einem Raum mit Wahrscheinlichkeitsmaß P entspricht stochastisch dem Generieren von Werten durch einen Zufallsgenerator mit Verteilung $F(\cdot) = P$. Unter der Nullhypothese H_0 entstammen alle Datenpunkte dem selben Zufallsgenerator. Betrachtet man also die Menge D aller generierten Datenpunkte, so ist es für die Wahrscheinlichkeit einer bestimmten Stichprobenzuordnung der Vektoren aus D unerheblich, ob die Vektoren nach Stichprobenzugehörigkeit geordnet generiert wurden, oder ob auf D nachträglich s zufällig ausgewählte Elemente als der ersten Stichprobe zugehörig definiert werden. Unter der Nullhypothese kann also die Stichprobenzuordnung U in $Z = (Z_1, \ldots, Z_g)$ als während der Stichprobenmessung nicht bekannt angenommen werden.

Für die Bestimmung der Verteilungsfunktion Q ist nun für festes $a_1 \in \mathbb{N}_0$ die bedingte Wahrscheinlichkeit $P(A_1 = a_1 \mid Z)$, genau a_1 Cross-Matches für gegebene

Realisierungen $(z_1, \ldots, z_g)$ von Z zu erhalten, gesucht. Für eine beliebige Stichprobenzuordnung U sei a_0 die Anzahl der Kanten zwischen Punkten aus $\mathcal{X}$, a_2 die Anzahl der Kanten zwischen Punkten aus $\mathcal{Y}$. Dann gilt

$$a_0 + a_1 + a_2 = h, \quad a_1 + 2a_0 = s, \quad a_1 + 2a_2 = t \; .$$

Man beachte, dass durch die Festlegung der Stichprobenumfänge s und t nur diejenigen Tupel $U \in \{0,1\}^g$ eine gültige Stichprobenzuordnung darstellen, für die $\sum_{i=1}^{g} U_i = s$ gilt. Es gibt also genau $\binom{g}{s}$ mögliche Zuordnungen. Die Teststatistik $A_1 : \Omega \longrightarrow \mathbb{N}_0$ der Anzahl der Cross-Matches ist damit eine diskrete Zufallsvariable auf dem Wahrscheinlichkeitsraum Ω aller gültigen Zuordnungen.

Für festes a_1 entsteht in Z eine zufällige Stichprobenzuordnung U durch die Definition von a_1 zufällig gewählten Kanten als Cross-Matches und die zufällige Wahl weiterer a_0 Kanten als Kanten zwischen Punkten aus $\mathcal{X}$. Die Anzahl der hierfür bestehenden Möglichkeiten wird durch den *Multinomialkoeffizienten* $\binom{h}{a_0, a_1, a_2}$ beschrieben. Zusätzlich können die Zuordnungen der beiden Punkte einer Cross-Match-Kante vertauscht werden, ohne die ausgewählten Kantenmengen zu beeinflussen. So kommen pro möglicher Wahl der Kantenmengen 2^{a_1} Zuordnungsmöglichkeiten hinzu. Insgesamt gibt es also für gegebene Realisierungen $(z_1, \ldots, z_g)$ von Z und gegebene Cross-Match-Anzahl a_1 unter Vernachlässigung der Stichprobenzugehörigkeit genau

$$2^{a_1} \frac{h!}{a_0! a_1! a_2!}$$

mögliche Stichprobenzuordnungen U. Damit ergibt sich die bedingte Wahrscheinlichkeit $P(A_1 = a_1 \mid Z)$ zu

$$P(A_1 = a_1 \mid Z) = \frac{2^{a_1} h!}{\binom{g}{s} a_0! a_1! a_2!} \; .$$

Da die rechte Seite dieser Gleichung nicht von Z abhängt, gilt $P(A_1 = a_1 \mid Z) = P(A_1 = a_1)$. Die Teststatistik A_1 hat damit eine exakte, von den Realisierungen $(z_1, \ldots, z_g)$ und damit von der Verteilung $F(\cdot)$ unabhängige Verteilungsfunktion Q mit $Q(x) := P(A_1 = x)$.

Abbildung 3.2 zeigt, dass unter der Nullhypothese die Wahrscheinlichkeit für eine geringe Anzahl an Cross-Matches sehr klein ist. Daher lehnt der Cross-Match-Test die Nullhypothese H_0 für kleine Werte von x ab. Für große x-Werte ist die Wahrscheinlichkeit $Q(x)$ ebenfalls sehr klein. Durch die Minimierung des Matchinggewichts erwartet man jedoch unter der Alternative H_1 eine noch geringere Wahrscheinlichkeit vieler Cross-Matches. Daher wird für große Werte von x die Hypothese

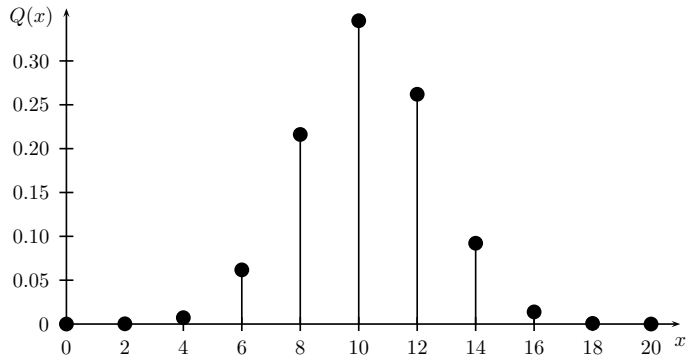


Abb. 3.2: Diskrete Verteilungsfunktion $Q(x) := P(A_1 = x)$ für $s = 20, t = 20$.

H_0 angenommen. Das *Niveau* $\alpha \in [0, 1]$ legt die Grenze für den Übergang von der Entscheidung gegen H_0 zur Entscheidung für H_0 fest. Für festes α wird H_0 für alle x abgelehnt, für die gilt

$$P(A_1 = x) \leq \alpha \ .$$

Für die hier abgebildeten, geraden Stichprobenumfänge $s = t = 20$ sind insbesondere nur gerade Cross-Match-Anzahlen x möglich, da die Perfektion des Matchings für jedes gemischte Punktepaar, das die noch ungepaarten Punkte jeder Stichprobe auf eine ungerade Anzahl verringert, ein weiteres Cross-Match-Paar erzwingt. Analoges gilt für ungerade Stichprobenumfänge s und t.

3.2.2 Analyse des Testverfahrens unter Voraussetzung nicht exakter Matchingalgorithmen

Bei genauerer Betrachtung fällt auf, dass für die Berechnung der Verteilungsfunktion Q unter der Nullhypothese die Minimalität des berechneten Matchings keine Rolle spielt. Falls also tatsächlich gleiche Verteilungen $F(\cdot) = G(\cdot)$ vorliegen, ist zu erwarten, dass die Abweichung des Matchinggewichts eines approximativen Matchings, das lediglich eine Näherungslösung darstellt, vom exakten minimalen Matchinggewicht keinen Einfluss auf die Häufigkeit des Fehlers erster Art hat (siehe Abbildung 3.3). Allerdings ist hierfür die Eindeutigkeit des approximativen Matchings für eine feste Menge von Stichprobenwerten zentral, wie in Kapitel 5 deutlich werden wird.

Entscheidung \ *Wirklichkeit*	H_0	H_1
H_0	**Richtige** *Entscheidung*	*Fehler* 2. *Art*
H_1	*Fehler* 1. *Art*	**Richtige** *Entscheidung*

Abb. 3.3: Wirkungstabelle eines Tests.

Umso bedeutender ist die Minimalität des Matchings dagegen für den Nachweis der *Konsistenz* des Testverfahrens, auf den hier jedoch nicht eingegangen werden soll. Rosenbaum diskutiert die Schwierigkeiten des Konsistenznachweises in seinem Artikel [Ros05]. Die Konsistenz ist eine Grundvoraussetzung für den praktischen Wert eines Tests und beschreibt das asymptotische Entscheidungsverhalten des Tests unter der Alternative H_1. Für einen konsistenten Test konvergiert die Wahrscheinlichkeit eines Fehlers zweiter Art gegen Null, für $g \to \infty$. Für den Cross-Match-Test ist der Rückgang der Fehlerhäufigkeit zweiter Art bei Verwendung eines exakten Matchingalgorithmus im Rahmen der experimentellen Untersuchung in Abschnitt 5.3 deutlich beobachtbar. Aus der Konsistenz des Cross-Match-Test folgt, dass die Wahrscheinlichkeit großer Cross-Match-Anzahlen bei tatsächlich verschiedenen Verteilungen $F(\cdot)$ und $G(\cdot)$ für $g \to \infty$ gegen Null konvergiert. Dies rechtfertigt im Nachhinein die Annahme der Hypothese H_0 für wachsende x-Werte trotz fallender diskreter Verteilungsfunktion Q.

Für die Beurteilung der Eignung eines nicht exakten Matchingalgorithmus in Zusammenhang mit dem Cross-Match-Test ist also experimentell zu prüfen, ob das Konsistenzverhalten des Testverfahrens erhalten bleibt und ob sich das erwartete, unveränderte Entscheidungsverhalten des Tests unter der Nullhypothese bestätigt.

3.3 Der Cross-Match-Test als Graphenproblem

Betrachtet man den Cross-Match-Test aus Sicht der Graphentheorie, so gibt er mit den Datenpunkten der beiden Stichproben eine gerade Anzahl an Knoten eines vollständigen, gewichteten Graphen $G = (V, E)$ vor. Die Gewichtungsfunktion $w : E \longrightarrow \mathbb{R}_{\geq 0}$ ist gegeben durch das auf den Datenpunkten definierte Distanzmaß d, das von der in $\mathbb{R}^k$ gewählten Norm abhängt. Zur Berechnung des eindeutigen gewichtsminimalen Matchings von G schlägt Rosenbaum Edmonds' Algorithmus vor und verweist in diesem Zusammenhang auf eine optimale Implementierung [Rot] nach Gabow [Gab73, Gab76] in $O(n^3)$.

3.3.1 Edmonds' exakter Matchingalgorithmus

Edmonds' Algorithmus ist ein exakter Algorithmus zur Berechnung eines nach Definition 2.5 gewichtsmaximalen Matchings eines vollständigen, ungerichteten, nichtnegativ gewichteten Graphen mit gerader Knotenanzahl [PS82]. Aufgrund der echt positiven Gewichtungsfunktion des aus dem Cross-Match-Test resultierenden Graphen (vergleiche Abschnitt 3.2) berechnet Edmonds' Algorithmus immer ein perfektes Matching der Stichprobenpunkte (vergleiche Abschnitt 2.1.2).

Bei genauerer Betrachtung sind die Voraussetzungen für die Anwendung dieses Algorithmus jedoch weit weniger streng, denn für einen ungerichteten, nichtnegativ gewichteten Graphen $G = (V, E)$ gilt:

- Falls $|V|$ ungerade ist, kann G zu $G' = (V \cup \{\epsilon\}, E')$ um einen Dummyknoten ϵ erweitert werden, indem man ϵ mit nullgewichteten Dummykanten an mindestens einen Knoten aus V anbindet. Jedes gewichtsmaximale Matching M' von G' induziert dann ein gewichtsmaximales Matching M von G mit $w(M) = w(M')$.
- Falls G nicht vollständig ist, kann G zu $G' = (V, E')$ mit $E' = \binom{V}{2}$ erweitert werden, indem man die noch fehlenden Kanten als nullgewichtete Dummykanten hinzufügt. Jedes gewichtsmaximale Matching M' von G' induziert dann ein gewichtsmaximales Matching M von G mit $w(M) = w(M')$.

Diese beiden Aussagen sind leicht einzusehen, denn jedes Matching M von G stellt auch für G' ein gültiges Matching dar, und es gilt somit $w(M') \geq w(M)$. Um aber echt größer als $w(M)$ werden zu können, muss ein maximales Matching M' von G' eine in G nicht vorhandene Kante beinhalten. Diese haben aber alle Gewicht Null und können somit nicht zu einer Gewichtssteigerung beitragen. Die Anwendung des Algorithmus setzt also lediglich einen ungerichteten, nichtnegativ gewichteten Graphen $G = (V, E)$ voraus. Aufgrund der Dualität des allgemeinen, nichtnegativ gewichteten Matchingproblems (siehe Abschnitt 2.1.2) genügt es außerdem nur das Maximierungsproblem zu betrachten. Alle oben getroffenen Aussagen gelten für das Minimierungsproblem entsprechend.

Die Idee des Algorithmus von Edmonds basiert auf der Lösungstheorie linearer Programme in Verbindung mit der Dualität des allgemeinen, nichtnegativ gewichteten Matchingproblems [PS82]. Die Forderung nach einem gewichtsminimalen, perfekten Matching eines vollständigen Graphen $G' = (V', E')$ mit gerader Knotenanzahl lässt sich schreiben als *lineares Programm* (LP) in Standardform:

$$\min \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} x_{ij} \quad \text{mit} \quad \sum_{j=1}^{i-1} x_{ji} + \sum_{j=i+1}^{n} x_{ij} = 1, \;\; i = 1, \ldots, n \tag{3.1}$$

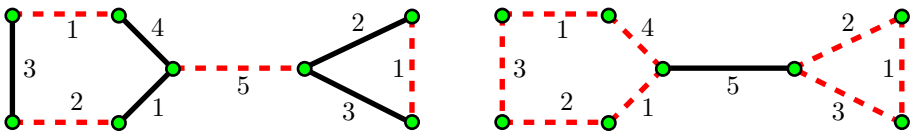
$$x_{ij} \geq 0, \;\; 1 \leq i < j \leq n \; . \tag{3.2}$$

Dabei bezeichnen die Indizes $i, j \in V'$ die Knoten, der jeweilige Wert w_{ij} bezeichnet das nichtnegative Gewicht der Kante $\{i, j\} \in E'$. Die Variablen x_{ij} sollen berechnet werden. Sie geben an, zu welchem Anteil die jeweilige Kante $\{i, j\}$ im Matching M' enthalten ist. Damit diese Interpretation eine sinnvolle Kantenteilmenge ergibt, muss $x_{ij} \in \{0, 1\}$ gelten. Die Interpretation lautet dann

$$x_{ij} = \begin{cases} 1, & \{i, j\} \in M' \\ 0, & \text{sonst} . \end{cases}$$

Die Nebenbedingungen (3.1) und (3.2) beschränken die Werte x_{ij} bereits auf $[0, 1]$. Innerhalb dieses Intervalls lässt die Lösungstheorie linearer Programme in Standardform jedoch zunächst alle Werte zu. Wendet man also auf dieses LP ein gängiges Lösungsverfahren wie zum Beispiel das *Simplexverfahren* an, so kann es vorkommen, dass die errechnete optimale Lösung nicht ganzzahlig ist und somit keine sinnvolle Kantenteilmenge darstellt! Falls jedoch eine sinnvolle Kantenmenge M' errechnet wird, stellt Nebenbedingung (3.1) sicher, dass zu jedem Knoten j genau eine inzidente Kante in M' liegt und M' damit perfekt ist. Würde man die Ganzzahligkeit der x_{ij} als Bedingung in das LP mit aufnehmen, so ergäbe sich daraus ein ganzzahliges, lineares Programm (ILP). Für ILPs gibt es jedoch keinen allgemein anwendbaren, polynomiellen Algorithmus. ILPs sind im Allgemeinen NP-schwer.

Edmonds findet jedoch eine Erweiterung des linearen Programms, die die polynomielle Lösbarkeit erhält und gleichzeitig eine ganzzahlige Lösung garantiert. Hintergrund seiner Überlegung ist die Tatsache, dass Kreise mit ungerader Knotenanzahl nicht perfekt gepaart werden können. Abbildung 3.4a zeigt einen Beispielgraphen $G = (V, E)$ mit gerader Knotenanzahl aber zwei ungeraden Kreisen. Die Dummykanten, die aus G einen vollständigen Graphen $G' = (V', E')$ machen, sind hier nicht eingezeichnet, da sie in diesem Beispiel nicht zur optimalen Lösung beitragen. Die gestrichelt eingezeichneten Kanten beschreiben ein minimales



(a) optimale Lösung des ILPs **(b)** Lösungskandidat des LPs

Abb. 3.4: Kreise mit ungerader Knotenanzahl.

ganzzahliges Matching $M_{\mathbb{Z}}$ sowohl von G' als auch von G, das den Nebenbedingungen des LPs genügt. Dieses Matching ist also eine optimale Lösung des ILPs mit $w(M_{\mathbb{Z}}) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} x_{ij} = 9$. Es ist jedoch keine optimale Lösung des LPs. Dies zeigt Abbildung 3.4b. Hier erzwingt der Verzicht auf die relativ schwer gewichtete „Brückenkante“ aufgrund von Nebenbedingung (3.2) eine Anpassung der Kantenanteile innerhalb der Kreise auf $x_{ij} = 1/2$. Dies führt zu einem geringeren Matchinggewicht $w(M) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} x_{ij} = (1+2+3+1+4+3+1+2)/2 = 8{,}5 < 9 = w(M_{\mathbb{Z}})$.

Durch die nicht ganzzahligen x_{ij} enthalten beide ungeraden Kreise jedoch mehr Matchingkantenanteile, als im ganzzahligen Fall zugelassen. Der linke Kreis K_L kann maximal zwei ganze Matchingkanten aufnehmen, hat aber hier einen Gesamtkantenanteil von 2,5, der rechte Kreis K_R enthält 1,5 Kanten, statt nur einer Kante bei ganzzahligem Anteil. Die Kreise sind also überfüllt. In einem perfekten Matching $M_{\mathbb{Z}}$ eines analog aufgebauten Graphen mit geraden Kreisen wäre die „Brückenkante“ unabhängig von ihrer Gewichtung nicht enthalten, da ein perfektes Matching

eines geraden Kreises genau die Hälfte aller Kreiskanten enthalten muss. Insbesondere gibt es genau zwei perfekte Matchings eines geraden Kreises. Das ILP würde als Ergebnis $M_{\mathbb{Z}}$ das jeweils leichtere Matching M_L und M_R des linken und rechten Kreises mit $w(M_L) \leq w(K_L)/2$ und $w(M_R) \leq w(K_R)/2$ wählen. Damit brächte die Änderung der Kreiskantenanteile auf $x_{ij} = 1/2$ keine Gewichtsersparnis mehr, denn es gilt $w(M) = w(K_L)/2 + w(K_R)/2 \geq w(M_L) + w(M_R) = w(M_{\mathbb{Z}})$.

Der vollständige Beweis, dass ausschließlich die Überfüllung der ungeraden Kreise für das Phänomen der nichtganzzahligen Lösungen verantwortlich ist, geht aus der detaillierten Herleitung von Edmonds' Algorithmus [PS82] hervor, auf die hier verzichtet wird. Edmonds' Erweiterung des LPs zur Lösung des eigentlichen ILPs ist hinreichend [PS82] und wird in Satz 3.1 formuliert.

Satz 3.1 *Das allgemeine, nichtnegativ gewichtete Matchingproblem ist äqivalent zu folgendem LP:*

$$\min \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} x_{ij} \quad \text{mit} \quad \begin{aligned} \sum_{j=1}^{i-1} x_{ji} + \sum_{j=i+1}^{n} x_{ij} &= 1, && i = 1, \ldots, n \\ x_{ij} &\geq 0, && 1 \leq i < j \leq n \\ \sum_{\substack{i,j \in S_k \\ i<j}} x_{ij} + y_k &= s_k, && k = 1, \ldots, N \\ y_k &\geq 0, && k = 1, \ldots, N\ . \end{aligned}$$

Dabei bezeichnen $S_1, \ldots, S_N$ alle ungeraden Knotenteilmengen und somit, da dem LP ein vollständiger Graph zugrundeliegt, alle ungeraden Kreise. Der Wert $s_k = (|S_k| - 1)/2$ gibt für jeden ungeraden Kreis S_k die maximal mögliche Anzahl an Matchingkanten im Falle einer ganzzahligen Lösung an. Über die Pufferwerte $y_k \geq 0$ wird so garantiert, dass keiner der ungeraden Kreise überfüllt wird.

Das so erweiterte LP ist nun mit dem *Primal-Dual-Algorithmus* lösbar, der eine Iteration über das zugehörige *duale Programm* durchführt. Der Schlüssel zur polynomiellen Berechnung des Matchingproblems liegt also in der Ausnutzung der Dualität. Diese erklärt auch, weshalb der eigentliche Algorithmus von Edmonds das Maximierungsproblem löst, während die theoretische Herleitung mit der Formulierung des Minimierungsproblems beginnt.

3.3.2 Erste Ideen zur Problemreduktion

Das detaillierte Verständnis des Algorithmus von Edmonds setzt fundierte Kenntnisse der Lösungstheorie linearer Programme voraus, welche bereits selbst ein komplexes Gebiet der Algorithmik darstellt. Dies ist einer der Aspekte, die die Implementierung dieses Algorithmus sehr erschweren. Will man zusätzlich die Laufzeit von $O(n^4)$ verbessern, sind außerdem spezielle, ebenfalls aufwändig zu implementierende Datenstrukturen unerlässlich [Gab76, Gab90]. Hieraus ergibt sich die Frage nach der Eignung einfacherer und schnellerer Algorithmen für den Cross-Match-Test, die nicht auf dem Dualitätsansatz von Edmonds basieren.

Es gibt exakte Algorithmen, die durch spezielle Anforderungen an den Eingabegraphen eine Laufzeitverbesserung erzielen. Vaidya [Vai88] nutzt geometrische Aspekte um ein exaktes, gewichtsminimales Matching eines vollständigen Graphen mit Knoten in $\mathbb{R}^2$ zu berechnen. Dies gelingt mit einer Laufzeit in $O(n^{5/2} \log^4 n)$. Allerdings sind für die Gewichtungsfunktion nur Distanzmaße zugelassen, die auf der *Manhattannorm* L_1, der *euklidischen Norm* L_2 oder der *Tschebyschewnorm*

L_∞ basieren. Die Verwendung dieses Algorithmus innerhalb des Cross-Match-Test macht somit für jedes andere Distanzmaß, wie zum Beispiel die *Mahalanobisdistanz*, eine aufwändige Umrechnung notwendig. Außerdem stecken auch in diesem Algorithmus Ansätze der Lösungstheorie linearer Programme und seine Anwendbarkeit ist auf zweidimensionale Daten beschränkt.

Für einen *bipartiten* Graphen ist das Matchingproblem generell einfacher zu lösen. Grundlage der meisten exakten Algorithmen für bipartites Matching ist die *Ungarische Methode* [NGH72] als Spezialfall linearer Optimierung. So stellt Vaidya [Vai88] auch eine bipartite Version seines Algorithmus in $O(n^{5/2} \log n)$ vor. Die Reduktion des aus dem Cross-Match-Test resultierenden Matchingproblems auf einen bipartiten Fall ist jedoch nicht möglich. Zwar gibt die Ausgangssituation des Tests mit den beiden Stichproben eine Partitionierung der Knotenmenge vor, würde diese jedoch bei der Matchingberechnung berücksichtigt, so ginge die zentrale Eigenschaft des Tests, nämlich die Vernachlässigbarkeit der Stichprobenzugehörigkeit unter der Nullhypothese, verloren.

Zwei weitere spezielle Grapheneigenschaften, die zu einer verbesserten Laufzeit führen, jedoch im Zusammenhang mit dem Cross-Match-Test nicht genutzt werden können, sind die *Planarität* sowie die konvexe Lage der Knoten in der Ebene. Für planare Graphen ist das allgemeine Matchingproblem mit Hilfe eines Divide-And-Conquer-Algorithmus rekursiv in $O(n^{3/2} \log^{3/2} n)$ lösbar, [Edm65a, Edm65b], Knoten in konvexer Lage ermöglichen eine Laufzeit in $O(n \log n)$ [MS91].

Approximative Matchingalgorithmen erreichen eine Laufzeitersparnis auf Kosten der Lösungsexaktheit. Sie berechnen lediglich eine Näherungslösung des Matchingproblems, die in ihrer Gewichtsabweichung durch den Approximationsfaktor begrenzt ist. Dieser beschreibt eine garantierte obere Schranke für das Gewichtsverhältnis $w(M'_{\min})/w(M_{\min}) \geq 1$ von approximativer Lösung $M'_{\min}$ und exakten Lösung $M_{\min}$ des selben Graphen. Weitere Details zu Approximationsalgorithmen erläutern zum Beispiel Wanka [Wan06], Cormen et al. [CLRS07] und Vazirani [Vaz01]. Allerdings kann man den Aspekt der Dualität von minimalem und maximalem Matching nicht auf alle Approximationsalgorithmen übertragen. So gibt es zum Beispiel einen sehr einfachen Greedy-Algorithmus für die Berechnung eines gewichtsmaximalen Matchings mit einer Laufzeit in $O(m \log n)$ für nach Gewicht vorsortierte Kanten und einem Approximationsfaktor von $1/2$ [DH03], der, wendet man ihn auf eine für das duale Minimierungsproblem modifizierte Gewichtungsfunktion an, keine sinnvolle Lösung liefert. Die Idee des Algorithmus basiert auf einer Abschätzung des durch das Greedy-Vorgehen entstehenden Fehlers nach oben. Im Falle einer dualen Gewichtungsfunktion kehren sich die Vorzeichen der Abschätzung um, so dass für die tatsächliche Gewichtungsfunktion des Minimierungsproblems nur eine untere, jedoch keine obere Schranke für den entstehenden Fehler und somit kein Approximationsfaktor mehr angegeben werden kann.

Der in dieser Arbeit näher untersuchte Algorithmus nach Wattenhofer und Wattenhofer [WW04] ist nicht auf die Dualität von minimalem und maximalem Matching angewiesen. Statt dessen berechnet dieser direkt ein approximatives, minimales Matching mit einem Approximationsfaktor von $2 \log n$. Dabei wird lediglich ein vollständiger Graph mit einer Gewichtungsfunktion, die die Dreiecksungleichung erfüllt, vorausgesetzt. Da jede Metrik auf $\mathbb{R}^k$, und damit das Distanzmaß des Cross-Match-Tests, dieser Anforderung genügt und die Stichproben des Tests einen vollständigen Graphen induzieren, ist dieser Algorithmus also ein geeigneter Kandidat um den Algorithmus von Edmonds zu ersetzen. Die Idee des Algorithmus nach Wattenhofer und Wattenhofer ist leicht verständlich und er hat eine schnelle Laufzeit in $O(n^2 \log n)$. In Kapitel 5 wird untersucht, ob und wie die mangelnde Lösungsexaktheit das Entscheidungsverhalten des Cross-Match-Tests beeinflusst.

Kapitel 4

Der Matchingalgorithmus nach Wattenhofer und Wattenhofer

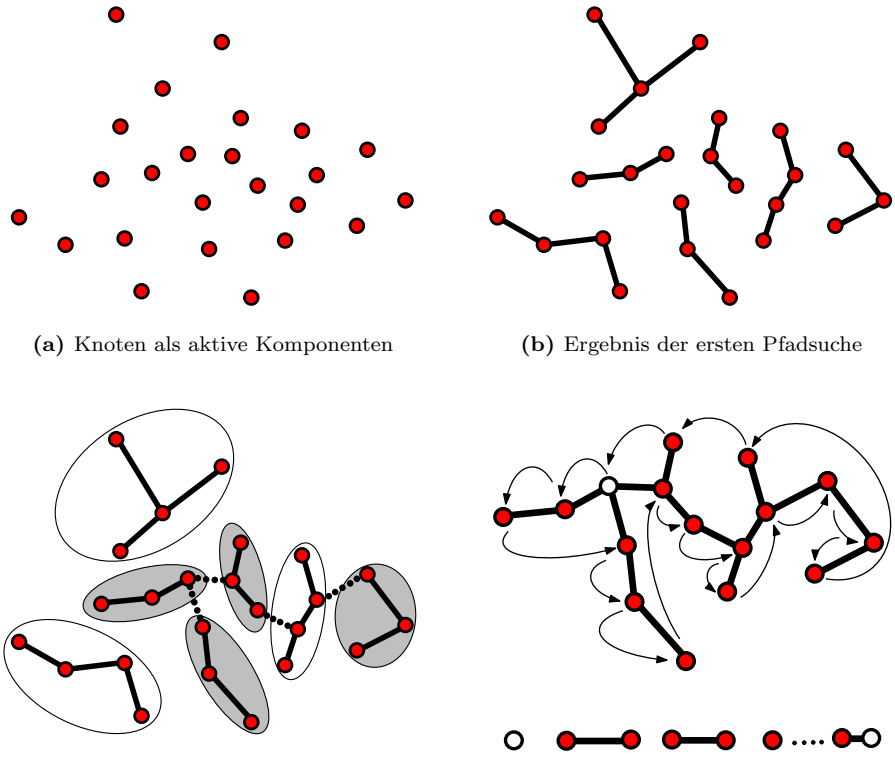
Der Matchingalgorithmus nach Wattenhofer und Wattenhofer ist ein approximativer Algorithmus zur Berechnung eines nicht erweiterbaren, minimalen Matchings auf einem vollständigen, ungerichteten, nichtnegativ gewichteten Graphen mit gerader Knotenanzahl. Er berechnet somit immer ein perfektes Matching (siehe Bemerkung zu Definition 2.5). Im Vergleich zu Edmonds' Algorithmus ist der Algorithmus nach Wattenhofer und Wattenhofer schneller mit einer Laufzeit in $O(n^2 \log n)$ und leicht zu implementieren. Gleichzeitig setzt dieser jedoch voraus, dass die verwendete Gewichtungsfunktion der Dreiecksungleichung genügt, was im Falle des Cross-Match-Tests durch die Verwendung eines metrischen Distanzmaßes garantiert ist.

4.1 Algorithmenbeschreibung

Der Algorithmus nach Wattenhofer und Wattenhofer geht in zwei Schritten vor. Schritt 1 berechnet auf dem gegebenen Graphen einen Wald aus minimalen *Spannbäumen* (siehe Algorithmus 1). Unter einem minimalen Spannbaum, oder auch *Minimum Spanning Tree (MST)*, eines zusammenhängenden, gewichteten Graphen G versteht man einen Teilbaum, also einen kreisfreien Teilgraphen, der alle Knoten von G abdeckt und dabei gewichtsminimal ist. Schritt 2 berechnet auf jedem Spannbaum eine *Eulertour mit Abkürzungen* (siehe Algorithmus 3) und daraus wiederum ein Matching, das maximal um den Faktor $2 \log n$ größer ist als ein minimales Matching auf dem Eingangsgraphen. Eine Eulertour mit Abkürzungen entspricht dabei der Reihenfolge der erstmals besuchten Knoten während einer Tiefensuche (DFS) mit beliebigem Startknoten.

4.1.1 Die Waldberechnung

Die Waldberechnung beginnt in Algorithmus 1, Zeile 3, mit der Initialisierung aller Knoten als Bäume für den ersten Schleifendurchlauf (siehe Abbildung 4.1a). Jeder Schleifendurchlauf fasst Knoten zu Bäumen, hier als Komponenten bezeichnet, zusammen, die im darauf folgenden Durchlauf als Superknoten betrachtet und erneut



(a) Knoten als aktive Komponenten

(b) Ergebnis der ersten Pfadsuche

(c) inaktive Komponenten (weiß) als Brücken

(d) Eulertour mit Abkürzungen

Abb. 4.1: Graphische Darstellung des Algorithmus nach Wattenhofer und Wattenhofer.

zusammengefasst werden. Dabei wird nach *aktiven* und *inaktiven* Komponenten unterschieden. Aktive Komponenten sind diejenigen, die sich aus einer ungeraden Anzahl von Knoten des Ursprungsgraphen zusammensetzen, Komponenten mit gerader Knotenanzahl sind inaktiv. Die aktiven Komponenten werden solange weiter zusammengefasst, bis am Ende ein Wald aus Bäumen mit jeweils gerader Knotenanzahl entsteht. Die gerade Knotenanzahl der Bäume ist Voraussetzung für das im zweiten Schritt für jeden Baum berechnete, perfekte Matching. Da immer eine gerade Anzahl aktiven Komponenten zusammengefasst werden muss, um eine inaktive Komponente zu erhalten, und die Anzahl der als aktive Komponenten initialisierten Knoten zu Beginn gerade ist, existiert auch zu Beginn jedes weiteren Durchlaufs eine gerade Anzahl aktiver Komponenten, was eine vollständige Zusammenfassung der aktiven Komponenten ermöglicht.

Innerhalb der Schleife in Algorithmus 1, Zeile 4, entscheidet Algorithmus 2 welche Komponenten zusammengefasst werden. Für jede aktive Komponente C wird die nächstliegende, aktive Komponente C' gesucht. Dabei entspricht die Distanz zweier aktiver Komponenten C und C' dem kürzesten Pfad P, der einen beliebigen Knoten aus C mit einem beliebigen Knoten aus C' verbindet, wobei bereits in inaktive Komponenten eingebundene Kanten als nullgewichtete Kanten genutzt werden dürfen. Inaktive Komponenten können also als Brücken oder Abkürzungen dienen. In Abbildung 4.1b ist der erste Durchlauf der Pfadsuche angedeutet. Da zu Beginn die Komponenten gerade den Knoten entsprechen und diese alle aktiv sind, wird für jeden Knoten der direkte nächste Nachbar gesucht.

Algorithmus 1 : PERFECTMATCHING(G)

Eingabe : vollständiger, ungerichteter, gewichteter Graph $G = (V, E)$
mit $|V| = n$ gerade und Gewichtungsfunktion w mit $w(e) \geq 0$ für alle $e \in E$
Ausgabe : Wald $F = (V, E')$ aus minimalen Spannbäumen

1 C_A = Menge aller aktiven Komponenten
2 z = Schleifenzähler
`%Initialisierung`
3 $F \leftarrow \emptyset$, $C_\mathrm{A} \leftarrow V$, $z = 0$
`%Schleife fasst Komponenten zusammen`
4 **solange** $C_\mathrm{A} \neq \emptyset$ **tue**
5 $findPath(G)$
6 $z = z + 1$
7 **Ende**
8 **zurück** F

Algorithmus 2 : FINDPATH(G)

Eingabe : vollständiger, ungerichteter, gewichteter Graph $G = (V, E)$
mit $|V| = n$ gerade und Gewichtungsfunktion w mit $w(e) \geq 0$ für alle $e \in E$
`%Diese Funktion hat Zugriff auf alle Variablen aus Algorithmus 1`
1 E_z = Kantenmenge, die in Durchlauf z zum Wald F hinzugefügt wird
2 P = Kantenmenge eines kürzesten Pfades zwischen zwei aktiven Komponenten
`%Initialisierung`
3 $E_z \leftarrow \emptyset$
`%Schleife berechnet kürzeste Pfade und speichert hinzukommende Waldkanten`
4 **für** jede Komponente $C \in C_\mathrm{A}$ **tue**
5 berechne kürzesten Pfad P zu einer aktiven Komponente $C' \neq C$
6 $E_z \leftarrow E_z \cup (P \setminus F)$
7 **Ende**
8 $F \leftarrow F \cup E_z$
9 update C_A

Sind für alle aktiven Komponenten die nächsten Nachbarn und die zugehörigen kürzesten Pfade gefunden, werden alle Pfadkanten, die noch zu keiner bereits bestehenden Komponente gehören, also noch nicht als Waldkanten ausgezeichnet sind (siehe Algorithmus 2, Zeile 6), in den Wald F aufgenommen (siehe Zeile 8). Dadurch wird jede aktive Komponente mit ihrem nächsten Nachbarn und allen auf dem Pfad liegenden inaktiven Komponenten zusammengefasst. Abbildung 4.1c zeigt die kürzesten Pfade (gepunktete Kanten) zwischen aktiven Komponenten (hellgrau) über eine inaktive Brückenkomponente (weiß) hinweg. Durch das Zusammenfassen entstehen neue aktive und inaktive Komponenten. Die Menge der aktuell aktiven Komponenten muss also für den nächsten Durchlauf der Schleife in Algorithmus 1 aktualisiert werden (siehe Algorithmus 2, Zeile 9).

Um durch diese Art des Zusammenfassens tatsächlich Bäume, also kreisfreie Graphen, zu erhalten, muss bei der Implementierung der Minimumsberechnung aller Pfade darauf geachtet werden, dass die Knoten in gleichbleibender Reihenfolge abgearbeitet werden und die Ersetzung des aktuellen Minimums nach gleichbleibender Vorschrift geschieht. Abbildung 4.2 zeigt als Beispiel drei Knoten, jeweils über Kanten vom Gewicht a verbunden.

(a) Initialisierung in Indexreihenfolge **(b)** Verletzung der Indexreihenfolge

Abb. 4.2: Zyklische Komponenten durch Verletzung der Abarbeitungsreihenfolge.

In Abbildung 4.2a wird der kürzeste, von Knoten 1 ausgehende Pfad mit dem Kantengewicht $w(\{1,2\})$ initialisiert. Der Vergleich dieses Initialgewichts mit $w(\{1,3\})$ ergibt keine Reduzierungsmöglichkeit. Knoten 1 wird also zu seinem nächsten Nachbarn, Knoten 2, verbunden. Von Knoten 2 ausgehend werden die übrigen Knoten wiederum in der durch die Indizes gegebenen Reihenfolge abgearbeitet und es wird so Knoten 1 als nächster Nachbar von Knoten 2 erkannt. Analog ergibt sich auch für Knoten 3 als nächster Nachbar Knoten 1. In Abbildung 4.2b wird für Knoten 2 jedoch die Indexreihenfolge bei der Initialisierung verletzt und so der kürzeste, von Knoten 2 ausgehende Pfad mit dem Gewicht $w(\{2,3\})$ statt mit $w(\{2,1\})$ initialisiert. Damit ist Knoten 3 nächster Nachbar zu Knoten 2 und es entsteht ein Kreis. Falls der Pfad für Knoten 2 zwar mit $w(\{2,1\})$ initialisiert wird, aber das aktuelle Minimum bereits bei Gleichheit der Gewichte ersetzt wird, entsteht ebenfalls ein Kreis. Im Falle des aus dem Cross-Match-Test resultierenden Graphenproblems sind identische Kantengewichte jedoch ausgeschlossen.

Unter Voraussetzung einer in diesem Sinne korrekten Implementierung ist jedoch klar, dass das Zusammenfassen von minimalen Spannbäumen über kürzeste Pfade wieder minimale Spannbäume ergibt. Dabei sind die zu Beginn initialisierten Einknotenbäume offensichtlich minimal auf dem durch ihre Knotenmenge definierten, vollständigen Teilgraphen.

Die Berechnung der kürzesten Pfade kann mit einem Zeitaufwand in $O(n^2)$ mit Hilfe eines *verallgemeinerten Voronoidiagramms* (siehe Abschnitt 4.3, Algorithmus 4) realisiert werden. Damit hat Algorithmus 2 einen Aufwand von $O(n^2)$. Die Schleife in Algorithmus 1 terminiert nach weniger als $\log n$ Durchläufen, da mindestens drei aktive Komponenten zusammengefasst werden müssen, um für den nächsten Durchlauf wieder eine aktive Komponente zu erhalten. Damit schrumpft die Zahl der aktiven Komponenten pro Durchlauf um mehr als die Hälfte. Algorithmus 1 hat also einen Gesamtaufwand von $O(n^2 \log n)$.

4.1.2 Die Matchingberechnung

Der in Algorithmus 1 berechnete Wald aus minimalen Spannbäumen gerader Knotenanzahl umfasst alle Knoten des Ausgangsgraphen G und kann nun verwendet werden, um ein approximatives, minimales Matching auf G zu berechnen. Diese Aufgabe übernimmt Algorithmus 3. Die Schleife in Zeile 4 berechnet für jeden Spannbaum S eine Eulertour T_S mit Abkürzungen. Dazu werden die Baumkanten formal als Zweifachkanten aufgefasst, wodurch jeder Baumknoten einen geraden Knotengrad erhält. Eine Abkürzung entsteht, wenn zwei Baumknoten statt über einen Pfad aus Baumkanten über eine direkte Kante außerhalb des Baumes verbunden werden. Dies ist möglich, da mit G ein vollständiger Graph zugrunde liegt. Die Abkürzungskante repräsentiert dann den umgangenen Abschnitt der Eulertour.

Algorithmus 3 : IDLE MATCH(F)

Eingabe : Wald F aus minimalen Spannbäumen gerader Knotenanzahl
Ausgabe : approximatives minimales Matching M auf Graph G

```
1 T_S = Eulertour mit Abkürzungen für Spannbaum S
2 M^S_min = kürzestes Matching aus Eulertour T_S
  %Initialisierung
3 M ← ∅
  %Schleife berechnet Eulertour mit Abkürzungen
4 für jeden Spannbaum S ⊆ F tue
5     berechne Eulertour T_S mit Abkürzungen
6     berechne kürzeres Matching M^S_min
7     M ← M ∪ M^S_min
8 Ende
9 zurück M
```

Wird die Eulertour mit Abkürzungen wie hier mit Hilfe einer Tiefensuche (DFS) berechnet, so kürzt diese, sobald ein Blatt erreicht ist, zum nächsten, noch nicht besuchten Knoten ab (siehe Abbildung 4.1d). Die so entstandene Eulertour wird durch eine geschlossene Knotenfolge, beginnend und endend mit einem beliebig gewählten Startknoten (weißer Knoten) repräsentiert. Der Startknoten ist der einzige mehrfach vorkommende Knoten. Zwischen je zwei aufeinanderfolgenden Knoten existiert immer eine Baumkante oder eine Abkürzungskante. Die Knotenfolge zeichnet somit einen Kreis in G aus. In diesem Kreis können genau zwei komplementäre, perfekte Matchings definiert werden. Der Algorithmus wählt pro Spannbaum S das jeweils kürzere Matching $M^S_{\min}$ und generiert daraus ein Matching M für den gesamten Graphen G, dessen Approximationsfaktor im folgenden Abschnitt 4.2 analysiert wird.

Der Zeitaufwand einer Eulertourberechnung ist, wie der einer Tiefensuche, in $O(|V| + |E|)$ bezüglich des zugrundeliegenden, zusammenhängenden Graphen. Da es sich hier um eine Tiefensuche auf Bäumen handelt, ergibt sich für die Berechnung aller Eulertouren ein Aufwand in $O(n)$. Die Gewichte der beiden komplementären Matchings pro Eulertour T_S können bei der Berechung der Tour bereits mitprotokolliert werden. Die Wahl des kürzeren Matchings $M^S_{\min}$ hat somit konstanten Aufwand. Die Hintereinanderausführung der Algorithmen 1 und 3 belässt den Gesamtaufwand in $O(n^2 \log n)$.

4.2 Analyse des Approximationsfaktors

Zu Beginn des Kapitels 4 wird behauptet, der Algorithmus nach Wattenhofer und Wattenhofer habe einen Approximationsfaktor von maximal $2 \log n$ und eine Laufzeit in $O(n^2 \log n)$. Der Nachweis des Zeitaufwands ist bereits erbracht. Dieser Abschnitt leitet nun die obere Schranke des Approximationsfaktors her.

Proposition 4.1 *Das Gewicht aller Kanten $e \in E_z$, die im z-ten Aufruf von* findPath *in Algorithmus 1 dem Wald F hinzugefügt werden, ist maximal so groß wie das zweifache Gewicht eines optimalen, minimalen Matchings M_{opt} über G. Es gilt also $w(E_z) \leq 2w(M_{\text{opt}})$ für alle z.*

Beweis. Sei C_{G} die Menge der aktuellen Komponenten eines beliebigen *findPath*-Aufrufs. Als rote Kante sei jede Kante $e \in M_{\text{opt}}$ bezeichnet, die zwei verschiedene

Komponenten aus C_G verbindet. Kanten in E_z werden in Anlehnung an Abbildung 4.1c als gepunktet bezeichnet. Es kann also Kanten geben, die sowohl gepunktet als auch rot sind. Kanten, die innerhalb von Komponenten verlaufen, sind nicht von Bedeutung. Da nun jede aktive Komponente C eine ungerade Anzahl von Knoten vereint, welche nicht perfekt gepaart werden kann, gibt es mindestens eine rote Kante, die C entweder mit einer inaktiven Komponente oder mit einer anderen aktiven Komponente verbindet. Mit dem gleichen Argument ist jede inaktive Komponente zu einer geraden Anzahl roter Kanten inzident. Für eine beliebige aktive Komponente C gibt es also mindestens einen roten Pfad $P_{C,C''}$ zu einer anderen aktiven Komponente C''. Außerdem gibt es einen gepunkteten Pfad $P_{C,C'}$ von C zur nächstgelegenen aktiven Komponente C'. Aus der Minimalitätsbedingung für $P_{C,C'}$ folgt $w(P_{C,C'}) \leq w(P_{C,C''})$. Sei C_A die Menge aller aktiven Komponenten. Dann ist auch das Gewicht aller gepunkteten Pfade ausgehend von allen aktiven Komponenten in C_A kleiner oder gleich dem Gewicht aller roten Pfade ausgehend von allen aktiven Komponenten in C_A. Es gilt also $\sum_{C \in C_\mathrm{A}} w(P_{C,C'}) \leq \sum_{C \in C_\mathrm{A}} w(P_{C,C''})$. Hierbei wird jeder Pfad höchstens zweimal gezählt. Die Summe der roten Pfadgewichte beträgt somit maximal das zweifache Gewicht eines optimalen, minimalen Matchings M_opt. Damit gilt $w(E_z) \leq \sum_{C \in C_\mathrm{A}} w(P_{C,C'}) \leq \sum_{C \in C_\mathrm{A}} w(P_{C,C''}) \leq 2w(M_\mathrm{opt})$. □

Proposition 4.2 *Das in Algorithmus 3, Zeile 6, für jeden Spannbaum S berechnete kürzere Matching M^S_min hat maximal das Gewicht aller Kanten des Spannbaums S. Es gilt also $w(M^S_\mathrm{min}) \leq w(S)$ für alle S.*

Beweis. Sei $\hat{T}_S$ eine Eulertour des Spannbaums S ohne Abkürzungen. Diese Tour hat das zweifache Baumgewicht, da sie jede Baumkante zweimal enthält. Da die Kostenfunktion des zugrundeliegenden Ursprungsgraphen G der Dreiecksungleichung genügt, bewirkt jede Abkürzung in T_S gegenüber $w(\hat{T}_S)$ eine Gewichtseinsparung. Es gilt also $w(T_S) \leq w(\hat{T}_S) = 2w(S)$. Außerdem hat jedes perfekte Matching M_S über T_S mit $w(M_S) > w(T_S)/2$ ein komplementäres Matching M^S_min mit $w(M^S_\mathrm{min}) < w(T_S)/2$, da $w(M_S) + w(M^S_\mathrm{min}) = w(T_S)$ gilt. Damit gilt $w(M^S_\mathrm{min}) \leq w(T_S)/2 \leq w(\hat{T}_S)/2 = w(S)$. □

Der Wald F, der an Algorithmus 3 übergeben wird, entsteht durch weniger als $\log n$ Aufrufe von *findPath* in Algorithmus 1. Aus Proposition 4.1 folgt so $w(F) < 2 \log n \, w(M_\mathrm{opt})$. Der Wald F entspricht der Vereinigung der einzelnen Spannbäume S und das gesamte approximative Matching M_app setzt sich aus einzelnen Matchings M^S_min über diesen Spannbäumen zusammen. Nach Proposition 4.2 gilt somit $w(M_\mathrm{app}) = \sum_{S \subset F} w(M^S_\mathrm{min}) \leq \sum_{S \subset F} w(S) = w(F) < 2 \log n \, w(M_\mathrm{opt})$. Der Approximationsfaktor des Algorithmus ist also kleiner als $2 \log n$ und die hier bewiesene obere Schranke nach Wattenhofer und Wattenhofer für $n \geq 3$ damit unscharf.

Wollte man für $n = 2^a$ Knoten tatsächlich ein $2 \log n$-faches, approximatives Matching erzeugen, so dürfte die Schleife in Algorithmus 1 erst nach genau $\log n$ Durchläufen terminieren. Dies setzt jedoch genau eine Halbierung der aktiven Komponenten pro Durchlauf voraus, was wiederum die Terminierung nach bereits einem Durchlauf zur Folge hätte, da bei gerader Knotenanzahl eine Halbierung der aktiven Komponenten, denen zu Beginn gerade die Knoten entsprechen, bedeutet, dass jeweils zwei Knoten zu einer neuen Komponente verschmelzen. Damit wären bereits nach einem Durchlauf alle Komponenten inaktiv. Das heißt, die Abschätzung der Schleifendurchläufe in Algorithmus 1 ist bereits für $n \geq 3$ nicht scharf. Eine maximale Anzahl an Schleifendurchläufen in Algorithmus 1 erhält man, wenn pro Durchlauf möglichst viele aktive Komponenten aus jeweils genau drei zuvor aktiven Komponenten hervorgehen. Die Schranke kann also zu $2 \log_3 n$ verschärft werden.

Für die in dieser Arbeit betrachteten Graphen wird außerdem auch die obere Schranke des pro Durchlauf zum Wald hinzugefügten Kantengewichts nicht erreicht (siehe Algorithmus 2, Zeile 8). Um pro Durchlauf tatsächlich das zweifache Gewicht eines optimalen Matchings hinzuzufügen (Proposition 4.1 mit Beweis), muss der jeweils kürzeste Pfad einer aktiven Komponente C zu ihrem nächsten aktiven Nachbarn C' genau so lang sein, wie einer der roten Pfade dieser Komponente zu einer beliebigen anderen aktiven Komponente C''. Gleichzeitig dürfen die roten Kanten in keinem Durchlauf in eine Komponente aufgenommen werden. Das heißt, in jedem Durchlauf muss es pro aktiver Komponente C zwei weitere aktive Komponenten C' und C'' geben, die gleich weit und gleichzeitig minimal von C entfernt liegen. Außerdem muss eine der beiden aktiven Komponenten C' und C'' von C aus über gepunktete aber nicht gleichzeitig rote Kanten, die andere über rote Kanten erreichbar sein. Da es aber in den mit Hilfe von Verteilungsfunktionen generierten Graphen dieser Arbeit keine gleichabständigen Knoten gibt, ist diese Voraussetzung bereits für den ersten Durchlauf nicht erfüllt! Damit bleibt auch die verschärfte Schranke von $2\log_3 n$ für die in dieser Arbeit vorkommenden Graphen unscharf, was auch in der experimentellen Untersuchung in Kapitel 5 deutlich wird. Die Approximationsfaktoren der getesteten Matchings bewegen sich weit unterhalb der theoretischen Schranke von $2\log_3 n$ (siehe Abschnitt 5.5).

4.3 Implementierung und Datenstrukturen

Algorithmus 1 arbeitet auf einem vollständigen, gewichteten Eingangsgraphen. Sowohl der Speicheraufwand als auch der Zeitaufwand pro Iteration über alle Kanten liegt für vollständige Graphen in $\Omega(n^2)$. Es bietet sich also an, den Eingangsgraphen G für Algorithmus 1 in Form einer gewichteten Adjazenzmatrix bereitzustellen. Die Mengenoperationen für das Zusammenfassen der Komponenten während der Waldberechnung (Algorithmus 2, Zeile 9) können mit Hilfe einer *balancierten Union-Find-Struktur mit Pfadkompression* auf den Knoten realisiert werden. Damit ergibt sich ein amortisierter, fast konstanter Zeitaufwand in $O(\alpha(n, n))$ für die Vereinigung zweier Mengen [Tar75], mit α als vereinfachter, inverser Ackermannfunktion. Der Wald selbst entsteht in Algorithmus 2 in den Zeilen 6 und 8 als Menge von Adjazenzlisten. So kann die Tiefensuche in Algorithmus 3 mit konstantem Aufwand auf die jeweiligen Nachfolger zugreifen und erreicht so eine Laufzeit in $O(n)$.

Als letzter kritischer Aspekt bleibt die Berechnung des verallgemeinerten Voronoidiagramms für die Pfadsuche in Algorithmus 2, Zeile 5. Unter einem *verallgemeinerten Voronoidiagramm* versteht man in diesem Zusammenhang ein Voronoidiagramm, dessen Regionen nicht auf direkten Abständen basieren, sondern auf kürzesten Pfaden. Algorithmus 4 ist eine Abwandlung des *Algorithmus von Dijkstra* und berechnet ein solches Voronoidiagramm mit den aktiven Komponenten in C_A als Regionenzentren. Jede inaktive Komponente befindet sich in der Region ihres nächstliegenden aktiven Nachbars. Für eine inaktive Komponente bedeutet dies, dass mit konstantem Aufwand die Länge des kürzesten Pfades zu einer aktive Komponente und diese aktive Komponente selbst bestimmt werden kann. Außerdem speichert jede Komponente ihre Nachfolgerkomponente auf dem kürzesten Pfad zum nächstliegenden Regionenzentrum.

Die benötigten Kantengewichte der Gewichtungsfunktion w_C des Eingangsgraphen G_C, dessen Knoten gerade den aktuellen Komponenten entsprechen, können bereits bei der Aktualisierung der Komponentenmenge C_A in Algorithmus 2, Zeile 9, für den jeweils nächsten Schleifendurchlauf der Pfadsuche berechnet werden. Das Kantengewicht zwischen zwei Komponenten entspricht dabei dem minimalen Gewicht unter allen Kanten, die beide Komponenten verbinden.

Algorithmus 4 : VORONOI(G)

Eingabe : vollständiger, ungerichteter, gewichteter Graph $G_{\mathrm{C}} = (C_{\mathrm{G}}, E_{\mathrm{C}})$ aus aktiven Komponenten in C_{A} und inaktiven Komponenten in $C_{\mathrm{G}} \setminus C_{\mathrm{A}}$ mit Gewichtungsfunktion w_{C} und Gewichtungsfunktion w des Ursprungsgraphen G

Ausgabe : verallgemeinertes Voronoidiagramm $(n(C), r(C), l(C))$

1 $n(C)$ = Array über alle Komponenten $C \in C_{\mathrm{G}}$, speichert Nachfolger
2 $r(C)$ = Array über alle Komponenten $C \in C_{\mathrm{G}}$, speichert Region
3 $l(C)$ = Array über alle Komponenten $C \in C_{\mathrm{G}}$, speichert Pfadlänge, wird berechnet mit Hilfe von w
`%Initialisierung mit direktem, minimalem Abstand zu aktiver Komponente`
4 **für** jede Komponente $C \in C_{\mathrm{G}}$ **tue**
5 $\quad l(C) \leftarrow \min\{w(C, C') : C' \in C_{\mathrm{A}}\}$
6 $\quad r(C) \leftarrow C'$
7 $\quad n(C) \leftarrow C'$
8 **Ende**
`%Schleife berechnet minimalen Abstand zu einer aktiven Komponente über andere inaktive Komponente`
9 **solange** $C_{\mathrm{G}} \setminus C_{\mathrm{A}} \neq \emptyset$ **tue**
10 $\quad$ finde $C_{\min}$ mit $l(C_{\min}) = \min\{l(C) : C \in C_{\mathrm{G}} \setminus C_{\mathrm{A}}\}$
11 $\quad C_{\mathrm{G}} \leftarrow C_{\mathrm{G}} \setminus \{C_{\min}\}$
12 $\quad$ **für** jede inaktive Komponente $C \in C_{\mathrm{G}} \setminus C_{\mathrm{A}}$ **tue**
13 $\quad\quad l(C) \leftarrow \min\{l(C), w(C, C_{\min}) + l(C_{\min})\}$
14 $\quad\quad$ aktualisiere gegebenenfalls $r(C)$ und $n(C)$
15 $\quad$ **Ende**
16 **Ende**

Indem über alle Kanten des vollständigen Graphen aus aktiven und inaktiven Komponenten iteriert wird, kann dann mit Hilfe des verallgemeinerten Voronoidiagramms der jeweils nächstliegende, aktive Nachbar für alle aktiven Komponenten in einem Schritt in $O(n^2)$ berechnet werden (Algorithmus 2, Schleife in Zeile 4). Falls eine Kante e zwei Komponenten aus verschiedenen Voronoiregionen verbindet, und die Summe aus dem Kantengewicht $w(e)$ und den beiden Pfadlängen der durch e verbundenen Komponenten zum jeweiligen Regionenzentrum kleiner ist als die aktuelle Pfadlänge zwischen den Regionenzentren, aktualisiert man die minimale Pfadlänge zwischen denselben.

Kapitel 5

Experimenteller Algorithmenvergleich

Der experimentelle Algorithmenvergleich versucht die in Abschnitt 3.2.2 formulierten Erwartungen an das Entscheidungsverhalten des Cross-Match-Tests unter Verwendung eines approximativen Matchingalgorithmus am Beispiel des Algorithmus nach Wattenhofer und Wattenhofer zu bestätigen. Die beiden Hauptaspekte hierbei sind die Verhaltensprüfung unter der Nullhypothese, wobei man eine durch die fehlende Exaktheit unbeeinflusste Wahrscheinlichkeit des Fehlers erster Art erwartet, sowie die Untersuchung des Konsistenzverhaltens.

5.1 Versuchsdesign und automatisierte Testreihen

Grundlage des Versuchsdesigns stellt ein Stichprobengenerator dar, der Stichproben verschiedener Umfänge und Verteilungen generiert. Die Generierung großer Umfänge ist lediglich durch die Verfügbarkeit von Zeit und Speicher beschränkt. Als mögliche Verteilungen stehen neben der *Gauß-* oder auch *Normalverteilung*, die *logarithmische Normalverteilung*, die *zentrale Chi-Quadrat-Verteilung* sowie die *zentrale Student'sche t-Verteilung* mit verschiedenen Parametern zur Auswahl. Bei der Generierung multivariater Stichproben in $\mathbb{R}^d$ besteht für kleine Dimensionszahlen d die Möglichkeit, dem Stichprobengenerator für jedes Standardbasispaar, also jeweils zwei Dimensionen im kartesischen Koordinatensystem, einen beliebigen Korrelationsfaktor $c \in [0, 1]$ vorzugeben. Die einzelnen Koordinaten multivariater Stichprobenwerte sind Realisierungen jeweils einer Zufallsvariablen mit entsprechender Verteilung. Ein positiver Korrelationsfaktor zweier Dimensionen erzeugt eine Abhängigkeit zwischen den Zufallsvariablen der zugehörigen Koordinaten. Für hohe Dimensionen wird ein einheitlicher Korrelationsfaktor festgelegt, der dann für alle Standardbasispaare zum Tragen kommt.

5.1.1 Getestete Verteilungsarten

Die Dichtefunktion f einer normalverteilten Zufallsvariablen $X \sim \mathcal{N}(\mu, \sigma^2)$ hängt von den Parametern μ und σ^2 ab und hat die Form

$$f_{\mathcal{N}}(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right).$$

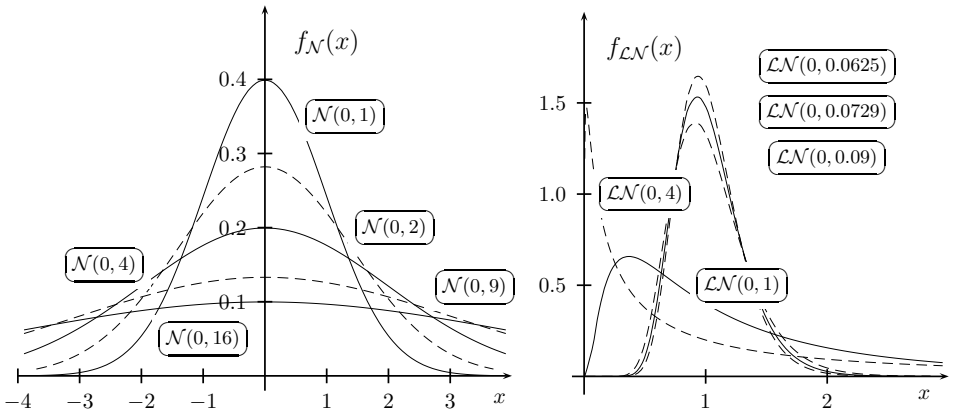


Abb. 5.1: Funktionsgraphen der Dichtefunktionen $f_{\mathcal{N}}$ und $f_{\mathcal{LN}}$.

Eine Zufallsvariable $X \sim \mathcal{N}(0,1)$ heißt *standardnormalverteilt*. Alle weiteren in dieser Arbeit verwendeten Verteilungen basieren auf der Normalverteilung. So geht die logarithmische Normalverteilung aus der Normalverteilung durch Logarithmieren hervor. Das heißt, aus einer normalverteilten Zufallsvariablen $X \sim \mathcal{N}(\mu, \sigma^2)$ entsteht mit $\ln(X) = Y$ eine logarithmisch normalverteilte Zufallsvariable $Y \sim \mathcal{LN}(\mu, \sigma^2)$ mit der Dichtefunktion

$$f_{\mathcal{LN}}(x) = \begin{cases} \frac{1}{x\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{ln(x)-\mu}{\sigma}\right)^2\right) & , x > 0 \\ 0 & , x \leq 0\,. \end{cases}$$

Abbildung 5.1 zeigt die Funktionsgraphen einer Parameterauswahl für die Dichtefunktionen $f_{\mathcal{N}}$ und $f_{\mathcal{LN}}$. Die Summe $Y = \sum_{k=1}^{n} X_k^2$ von n Quadraten über standardnormalverteilten Zufallsvariablen $X_1, \ldots, X_n$ ergibt eine zentral chi-quadratverteilte Zufallsvariable $Y \sim \mathcal{X}_n^2$ mit n Freiheitsgraden und der Dichtefunktion

$$f_{\mathcal{X}^2}^{(n)}(x) = \begin{cases} \frac{x^{\frac{n}{2}-1}\exp(-\frac{x}{2})}{2^{\frac{n}{2}}\Gamma(\frac{n}{2})} & , x > 0 \\ 0 & , x \leq 0\,. \end{cases}$$

Dabei steht Γ für die *Gammafunktion*. Die zentrale Student'sche t-Verteilung ist schließlich eine Kombination aus Normalverteilung und zentraler Chi-Quadrat-Verteilung. Ist $X \sim \mathcal{N}(0,1)$ standardnormalverteilt und $Y \sim \mathcal{X}_n^2$ zentral chi-quadratverteilt mit n Freiheitsgraden, so genügt $Z = X/Y$ der zentralen Student'schen t-Verteilung mit n Freiheitsgraden und der Dichtefunktion

$$f_{\mathrm{t}}^{(n)}(x) = \frac{\Gamma\left(\frac{n+1}{2}\right)}{\sqrt{n\pi}\,\Gamma\left(\frac{n}{2}\right)} \left(1 + \frac{x^2}{n}\right)^{-\frac{n+1}{2}}\,.$$

Eine Übersicht von Funktionsgraphen der Dichten $f_{\mathcal{X}^2}^{(n)}$ und $f_{\mathrm{t}}^{(n)}$ mit verschiedenen Freiheitsgraden zeigt Abbildung 5.2.

5.1.2 Testreihenparameter

Ziel des Testreihenentwurfs war es, das Verhalten der beiden Algorithmen bei verschieden stark voneinander abweichenden Stichprobenverteilungen abzubilden. Um die Stärke der Abweichung zweier generierter Stichproben quantifizieren zu können,

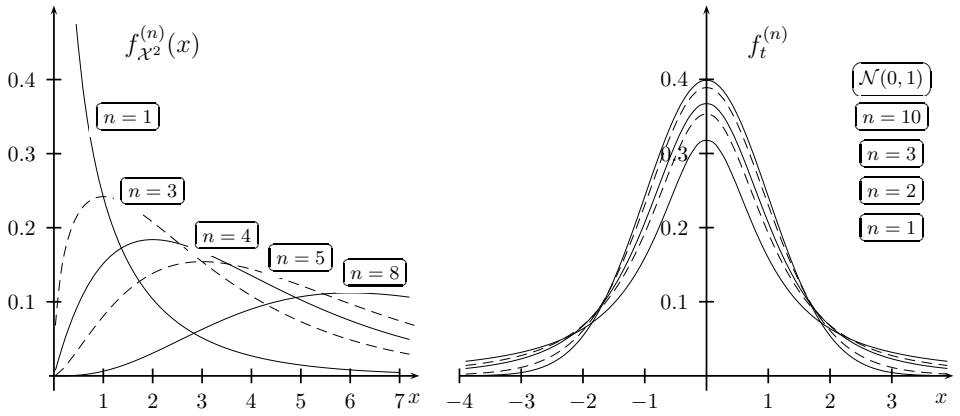


Abb. 5.2: Funktionsgraphen der Dichtefunktionen $f^{(n)}_{\mathcal{X}^2}$ und $f^{(n)}_{\mathrm{t}}$.

bietet die Testreihenautomatisierung für jede Verteilungsart einzeln variierbare Parameter. Dabei wird die Verteilung der ersten Stichprobe mit allen zugehörigen Parametern fest und unabhängig von der Verteilung der zweiten Stichprobe gewählt. Für die Verteilung der zweiten Stichprobe können bestimmte Parameter ausgezeichnet werden, deren Werte während einer beliebigen Anzahl von Schritten linear um einen festen Wert steigen, während die Verteilung der ersten Stichprobe in jedem Schritt unverändert bleibt. Pro Schritt kann außerdem die Anzahl der zu generierenden Fälle festgelegt werden.

5.2 Verhaltensprüfung unter der Nullhypothese

Da nach den Überlegungen in Abschnitt 3.2.1 die Minimalität des vorausgesetzten Matchings in die Berechnung der Verteilungsfunktion der Teststatistik A_1 nicht einfließt, erwartet man unter der Nullhypothese, wie in Abschnitt 3.2.2 erläutert, ein im Mittel gleiches Verhalten der beiden Algorithmen in Bezug auf die berechnete Cross-Match-Anzahl. Um diese Erwartung zu verifizieren, wurden Testreihen der Nullhypothese entsprechend mit jeweils gleicher Verteilung der beiden Stichproben berechnet, wobei jede Testreihe 5000 Fälle umfasst.

Abbildung 5.3 zeigt einen Auszug der Testreihen mit Stichprobenumfängen von jeweils 20 Werten, ausgewertet bezüglich zweier verschiedener Niveaus. Das Niveau von 0,007238 entspricht der Wahrscheinlichkeit genau vier Cross-Matches zu erhalten, das Niveau von 0,061761 der Wahrscheinlichkeit genau sechs Cross-Matches zu erhalten. Als Fehlerfall zum Niveau von 0,007238 gelten alle Fälle der jeweiligen Testreihe, für die vier oder weniger Cross-Matches berechnet wurden. Analoges gilt für die Fehlerfälle zum Niveau von 0,061761. Pro Testreihe wurde der prozentuale Anstieg der Fehlentscheidungen (Fehler erster Art) des approximativen Algorithmus bezüglich den Fehlerfällen des exakten Algorithmus von Edmonds in Form eines Balkens aufgetragen. Jede Testreihe entspricht einer getesteten Verteilung mit bestimmten Parametern. Die detaillierten Parameterwerte sind in Tabelle 7.1 im Anhang aufgelistet. Die Tabellenzeilen sind im Diagramm jeweils von links nach rechts angeordnet.

Entgegen der ursprünglichen Erwartung scheint jedoch laut Abbildung 5.3 die Wahrscheinlichkeit für einen Fehler erster Art unter Verwendung des approximativen Algorithmus für beide Niveaus nicht gegen jene unter Verwendung des exakten

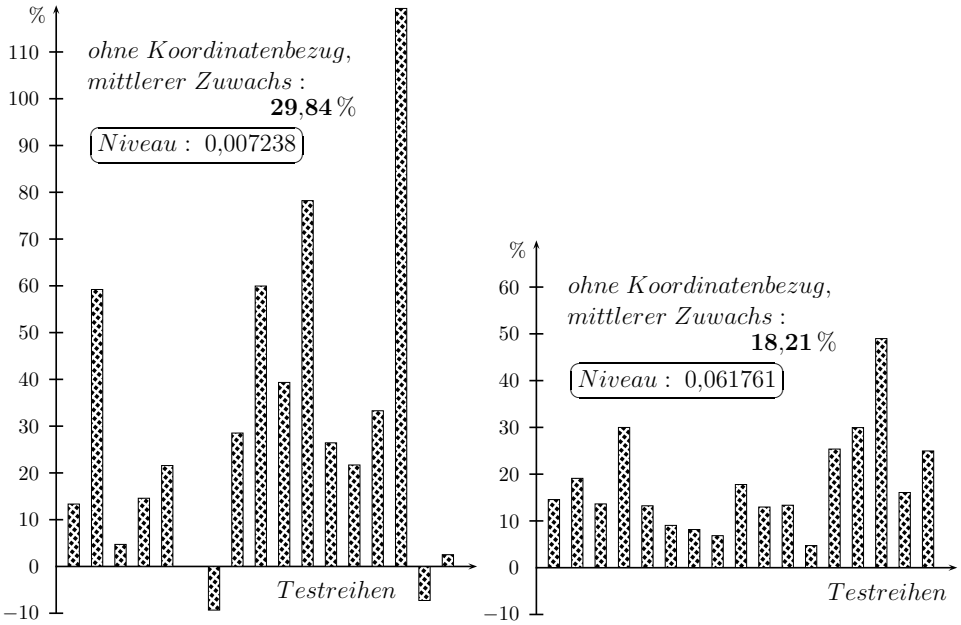


Abb. 5.3: Prozentualer Zuwachs der Fehlerfälle zu zwei verschiedenen Niveaus.

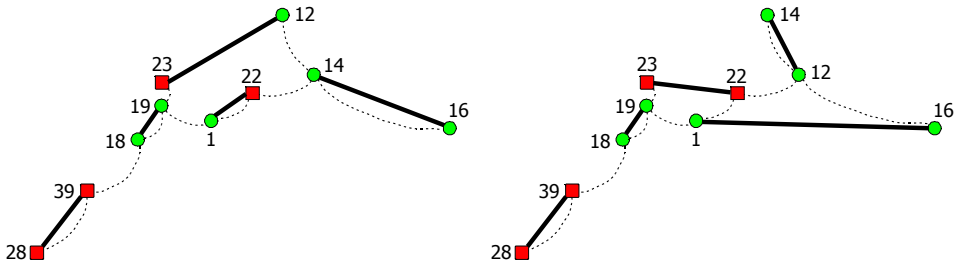
Algorithmus von Edmonds zu konvergieren. Statt dessen liegt die Anzahl der Fehlentscheidungen bis auf insgesamt vier Ausnahmen immer über jener des exakten Matchingalgorithmus. Der Grund für dieses unerwartete Verhalten liegt nur indirekt in der mangelnden Exaktheit des approximativen Matchinggewichts. Vielmehr ist dieses Verhalten eine direkte Folge der nicht gegebenen Eindeutigkeit des approximativen Matchings.

Die Eindeutigkeit des Matchings von Edmonds' Algorithmus folgt aus dessen Exaktheit und der vorausgesetzten Stetigkeit der Stichprobenverteilungen (vergleiche Abschnitt 3.2.1). Das Vorgehen des Approximationsalgorithmus gewährt dagegen bei der Berechnung der Eulertouren in Algorithmus 3 Wahlfreiheit bezüglich des jeweiligen Startknotens und der Abarbeitungsreihenfolge der Nachfolgerknoten während der Tiefensuche. Dies führt dazu, dass für identische Stichprobenpunktwolken abhängig von der Implementierung des Algorithmus unterschiedliche Cross-Match-Anzahlen berechnet werden können.

Die in dieser Arbeit ursprünglich verwendete Implementierung trifft die Wahl des Startknotens in Abhängigkeit von der Anordnung der Stichprobenpunkte in der Datenstruktur und damit in Abhängigkeit von deren Generierungsreihenfolge. Abbildung 5.4 zeigt einen Ausschnitt zweier unterschiedlicher, approximativer Matchings berechnet auf ein und derselben Punktmenge. Die Nummerierung der Punkte gibt deren Generierungsreihenfolge wieder. Die Matchingkanten sind durchgezogen dargestellt, die gepunkteten Kanten repräsentieren den Baum, aus dessen jeweiliger Eulertour das entsprechende Matching hervorgeht. Die Reihenfolge der generierten Punkte für Abbildung 5.4b entsteht aus der Punktreihenfolge für Abbildung 5.4a durch Vertauschung der Positionen 12 und 14 innerhalb der ersten Stichprobe. Dadurch entsteht bei der Waldberechnung eine andere balancierte Union-Find-Struktur (vergleiche Abschnitt 4.3), wodurch in Abbildung 5.4b Knoten 1 zum Startknoten gewählt wird. Die Eulertour in Abbildung 5.4a beginnt dagegen bei Knoten 12. So ergeben sich zwei verschiedene Eulertouren, die auf derselben Punktmenge zwei verschiedene Matchings induzieren. Im Vergleich zum Matching in Abbildung 5.4b beinhaltet das Matching in Abbildung 5.4a zwei zusätzliche Cross-

Matches. Das auf einer gegebenen Punktmenge approximativ berechnete Matching ist also für die hier ursprünglich verwendete Implementierung des Approximationsalgorithmus nicht eindeutig.

Dies hat zur Folge, dass die Umsortierung der Indizes in der Herleitung der Verteilungsfunktion Q für die Teststatistik A_1 in Abschnitt 3.2.1 unter Verwendung des Approximationsalgorithmus nicht zulässig ist und die berechnete Verteilungsfunktion somit ungültig. Die tatsächliche Verteilungsfunktion der Teststatistik unter Verwendung des approximativen Algorithmus ist nicht mehr unabhängig von den konkreten Stichprobenwerten und der Verteilung $F(\cdot)$. Für die hier getesteten Verteilungen scheint die Wahrscheinlichkeit kleiner Cross-Match-Anzahlen unter Verwendung des Approximationsalgorithmus angesichts der deutlich steigenden Fehlerfälle in Abbildung 5.3 größer zu sein als unter Verwendung des exakten Algorithmus. Die Tatsache, dass der prozentuale Fehlerzuwachs zum Niveau von 0,061761 in Abbildung 5.3 allgemein geringer ausfällt als zum strengeren Niveau von 0,007238 begründet sich darin, dass sich die absolute Zahl der Fehlerfälle im ersten Fall zwischen 300 und 400 bewegt, im Gegensatz zu Werten zwischen 20 und 40 im zweiten Fall. Damit wiegt ein Fehlerfall zum Niveau von 0,007238 prozentual schwerer.



(a) Eulertour: 12-14-16-22-1-19-18-39-28-23-12
sichtbare Cross-Matches: 2

(b) Eulertour: 1-19-18-39-28-23-22-12-14-16-1
sichtbare Cross-Matches: 0

Abb. 5.4: Mehrdeutigkeit des approximativen Matchings auf einer festen Punktmenge.

Die Mehrdeutigkeit des approximativen Matchings einer festen Punktmenge lässt sich beheben, indem man die Startknoten und Nachfolger während der Tiefensuche in Algorithmus 3 statt in Abhängigkeit von ihrer Position in der Datenstruktur, von ihren tatsächlichen Koordinaten abhängig wählt. Für die in Abbildung 5.5 und 5.6 dargestellten Testreihen wurde zunächst als jeweiliger Startknoten der Baumknoten mit der kleinsten ersten Koordinate gewählt (schraffierte Balken). In einem zweiten Schritt wurden anschließend auch die während der Tiefensuche betrachteten Nachfolger nach Abständen aufsteigend sortiert und somit ein Bezug zu deren tatsächlichen Koordinaten hergestellt (schwarze Balken). Durch diese Modifikation des Algorithmus ist nun die Eindeutigkeit des approximativen Matchings für eine feste Punktmenge ohne Beeinträchtigung der asymptotischen Laufzeit gewährleistet. Die gepunkteten Diagrammbalken stammen aus Abbildung 5.3. Sie wurden zum besseren Vergleich nochmals dargestellt. Die modifizierten Testreihen wurden mit den jeweils gleichen Parametern erstellt wie die zugehörigen Testreihen aus Abbildung 5.3. Die Diagramme 5.5 und 5.6 machen deutlich, dass bereits mit der Festlegung der Startknoten die Tendenz des Approximationsalgorithmus, öfter als der exakte Algorithmus die Nullhypothese zu verwerfen, zurückgeht. Im Falle des eindeutigen, approximativen Matchings ist keine Tendenz der Fehlerzuwächse mehr

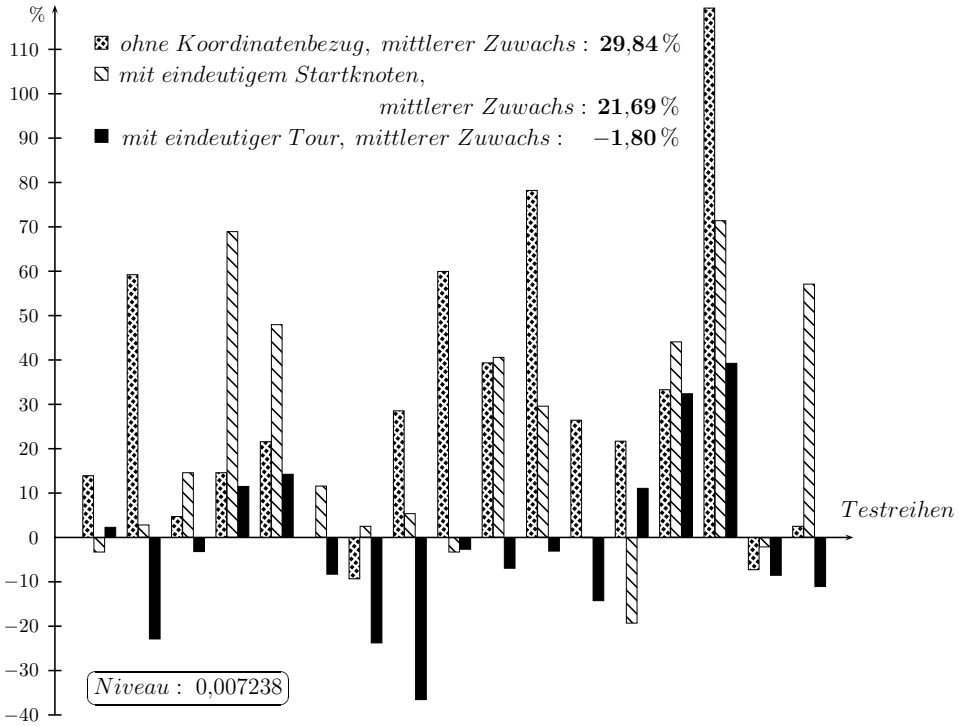


Abb. 5.5: Entwicklung des prozentualen Fehlerzuwachses mit zunehmender Eindeutigkeit, zum Niveau von 0,007238 (4 Cross-Matches).

zu erkennen. Das heißt, der modifizierte Approximationsalgorithmus entscheidet im Mittel genauso oft (oder selten) falsch wie Edmonds' Algorithmus. Er verhält sich somit tatsächlich der in Abschnitt 3.2.2 formulierten Erwartung entsprechend.

5.3 Untersuchung des Konsistenzverhaltens

Der Cross-Match-Test gilt als konsistent unter Verwendung des approximativen Matchingalgorithmus nach Wattenhofer und Wattenhofer, falls die Fehlerfälle zweiter Art für große Stichprobenumfänge zurückgehen. Da die Testreihenberechnung für große Stichprobenumfänge sehr zeitintensiv ist, wurden für die Konsistenzprüfung Testreihen mit nur jeweils 5 Fällen berechnet. Dennoch ist bereits an dieser

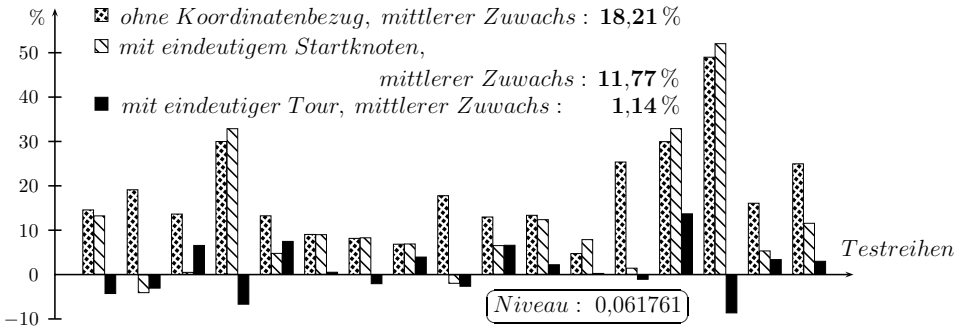
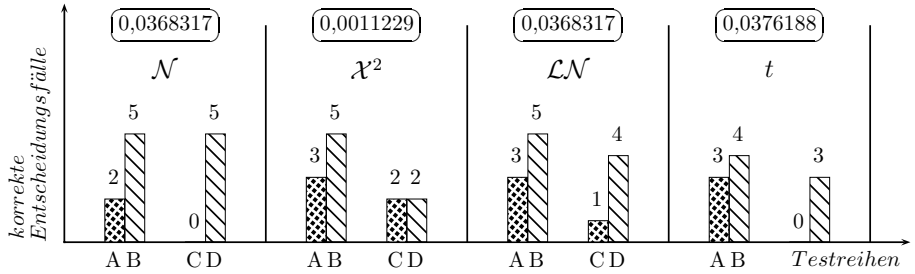
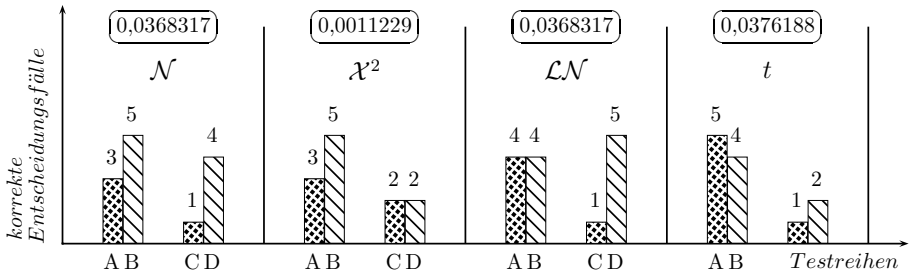


Abb. 5.6: Entwicklung des prozentualen Fehlerzuwachses mit zunehmender Eindeutigkeit, zum Niveau von 0,061761 (6 Cross-Matches).

kleinen Fallanzahl das Konsistenzverhalten sowohl unter Verwendung des exakten Algorithmus (siehe Abbildung 5.7b) als auch unter Verwendung des Approximationsalgorithmus (siehe Abbildung 5.7a) erkennbar. Abbildung 5.7 zeigt pro Verteilungsart vier Testreihen. Die gepunkteten Diagrammbalken der Testreihen A und C stellen jeweils die Anzahl der korrekten Entscheidungen bei einem Gesamtstichprobenumfang von 2000 Werten dar, den Testreihen B und D, hier schraffiert dargestellt, liegen dagegen insgesamt 3000 generierte Stichprobenwerte zugrunde. Als korrekt entschieden gilt ein Testfall, für den die Wahrscheinlichkeit der berechneten Cross-Match-Anzahl unter oder genau auf dem festgelegten Entscheidungsniveau, in abgerundeten Rechtecken angegeben, liegt. Das heißt, der Cross-Match-Test verwirft für diesen Fall die Nullhypothese, erkennt also, dass verschiedene Stichprobenverteilungen vorliegen.



(a) Konsistenz unter Verwendung des Approximationsalgorithmus



(b) Konsistenz unter Verwendung des exakten Algorithmus

Abb. 5.7: Konsistenzverhalten unter Verwendung verschiedener Algorithmen.

Die Testreihenauswahl basiert auf der Erwartung, dass bei gegebener Konsistenz des Cross-Match-Tests große Stichprobenumfänge geringe Verteilungsabweichungen besser erkennen lassen als kleine Stichprobenumfänge. Eine im Vergleich noch geringere Verteilungsabweichung der Stichproben sollte bei großen Stichprobenumfängen immerhin noch in manchen Fällen erkannt werden. Daher beschreiben die Parameter der ersten beiden Testreihen A und B jeweils gering differierende Stichprobenverteilungen. Die Stichprobenverteilungen der Testreihen C und D ähneln sich im Vergleich noch stärker. Die detaillierten Verteilungsparameter sind in Tabelle 7.2 im Anhang nachzulesen. Dabei enthält Zeile 1 jeweils die Parameter der ersten Stichprobe, Zeile 2 die der zweiten Stichprobe einer Testreihe.

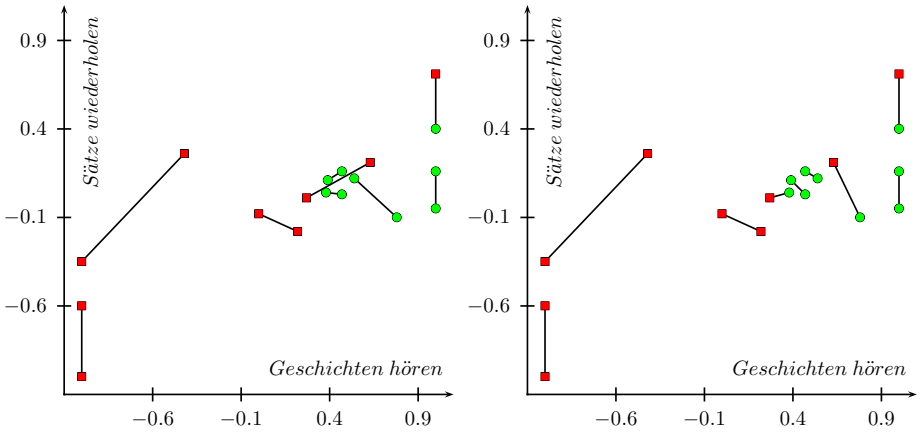
Bis auf die Testreihen A und B der zentralen Student'schen t-Verteilung bestätigt Diagramm 5.7 das erwartete Konsistenzverhalten sowohl für Edmonds' Algorithmus als auch für den Algorithmus nach Wattenhofer und Wattenhofer. Man erkennt deutlich die Überlegenheit der großen Stichprobenumfänge von 3000 Werten (schraffierte Balken) gegenüber den Stichprobengrößen von 2000 Werten (gepunktete Balken). Dass bei der Student'schen t-Verteilung in Abbildung 5.7b die Testreihe

mit 2000 Stichprobenwerten im Vergleich zur Testreihe mit 3000 Stichprobenwerten die abweichenden Stichprobenverteilungen besser erkennt, also öfter die Alternativhypothese bestätigt, liegt daran, dass die Punktwolken der beiden Stichproben so große Überschneidungen aufweisen, dass es vorkommt, dass Cross-Match-Anzahlen berechnet werden, für die die Verteilungsfunktion Q bereits wieder fällt. Diese entziehen sich der Auswertung bezüglich eines sinnvollen Niveaus. Für Cross-Match-Anzahlen aus diesem Bereich wird die Nullhypothese immer beibehalten (siehe Abschnitt 3.2.2). Im hier vorliegenden Fall wurde bei der umfangsstärkeren Testreihe einmal eine solch große Cross-Match-Anzahl durch Edmonds' Algorithmus berechnet, weshalb für die Auswertung zum gegebenen Niveau nur noch vier Testfälle übrig blieben.

Außerdem fällt auf, dass das Niveau der Chi-Quadrat-Verteilung sehr viel strenger angegeben ist als die übrigen Niveaus. Dies begründet sich darin, dass die hier getroffene Wahl der Testreihenparameter zu so geringen Verteilungsdifferenzen führt, dass ein besseres Entscheidungsverhalten großer Stichprobenumfänge gegenüber kleineren Umfängen erst bei einem sehr strengen Niveau sichtbar wird.

5.4 Auswertung des Eingangsbeispiels

Im Falle des eingangs in Abschnitt 1.1.2 erwähnten Anwendungsbeispiels der Kernspinmesswerte erkennt der Matchingalgorithmus nach Wattenhofer und Wattenhofer ebenso wie der exakte Algorithmus unterschiedliche Verteilungen der gegebenen Stichproben (siehe Abbildung 5.8). Während das exakte Matching drei Cross-Matches beinhaltet und damit erst zu einem relativ hohen Niveau von $0,2764$ die Nullhypothese verwirft, berechnet der Approximationsalgorithmus nur ein Cross-Match, was der minimalen Anzahl an Cross-Matches bei ungeraden Stichprobenumfängen entspricht, und führt somit eindeutig zur Ablehnung der Nullhypothese durch den Cross-Match-Test.



(a) approximatives Matching **(b)** exaktes Matching

Abb. 5.8: Matching von Punktwolken aus Kernspinmesswerten.

Die Tatsache, dass der approximative Algorithmus den Cross-Match-Test für dieses Beispiel zu einer sichereren Entscheidung führt, lässt jedoch keinen allgemeinen Schluss zu. Dieses Beispiel zeigt lediglich, dass je nach gegebener Punkteanordnung das approximative Matching dem exakten Matching in Bezug auf die Entscheidung des Cross-Match-Tests in manchen Fällen überlegen sein kann.

5.5 Sonstige Beobachtungen

Unabhängig von der Stichprobenzugehörigkeit der einzelnen Datenpunkte lassen sich zwei weitere Beobachtungen machen. Zum einen fällt auf, dass bei den für diese Arbeit erzeugten Graphen, deren Knotenanordnungen bestimmten Verteilungen unterliegen, selbst die in Abschnitt 4.2 theoretisch verschärfte obere Schranke des Approximationsfaktors von $2 \log_3 n$ bei Weitem nicht erreicht wird, zum anderen kann man mit wachsender Anzahl der Knoten eine abnehmende Streuung der Approximationsfaktoren der Fälle innerhalb einer Testreihe erkennen.

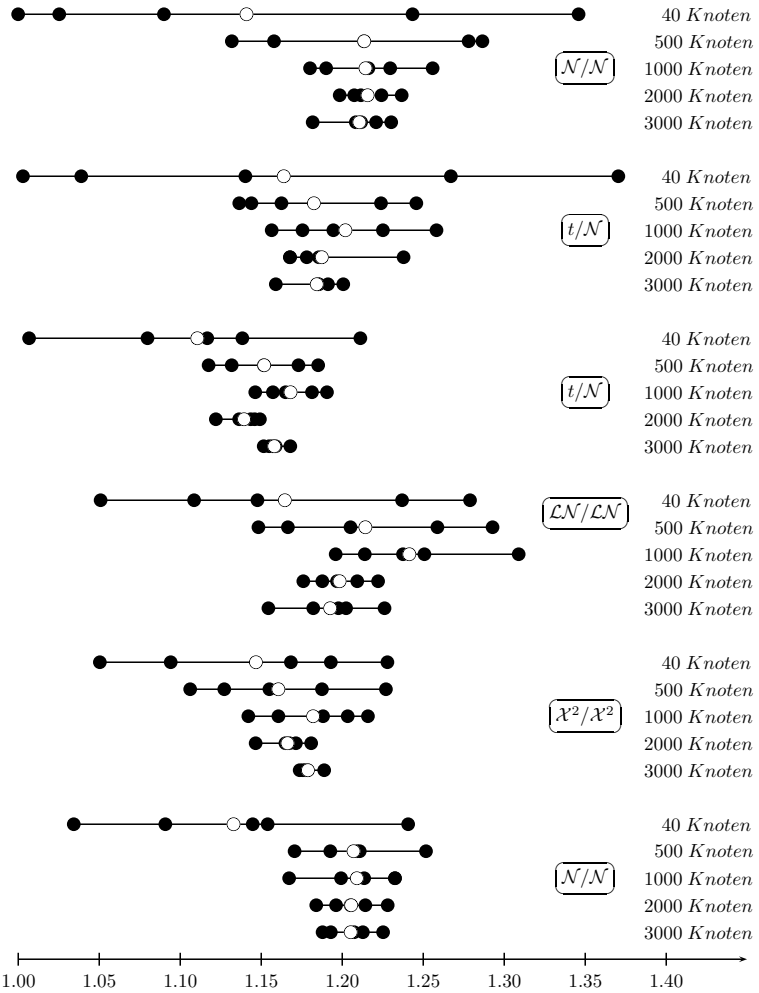


Abb. 5.9: Streuung der Einzelfaktoren bei verschiedenen Knotenanzahlen.

Abbildung 5.9 zeigt die Streuung der als schwarze Kreisscheiben dargestellten Faktoren in Testreihentupeln mit jeweils zunehmender Knotenanzahl von 40 bis 3000 Knoten. Die weißen Kreisscheiben markieren das jeweilige arithmetische Mittel der einzelnen Faktoren pro Testreihe. Die Testreihen mit 2000 und 3000 Stichprobenwerten wurden, mit Ausnahme jener der drei oberen Tupel, aus Abschnitt 5.3 übernommen. Die Testreihen mit 40 bis 1000 Stichprobenwerten wurden mit dazu passenden Parametern neu erstellt. Tabelle 7.3 zeigt eine Übersicht über alle in Abbildung 5.9 eingeflossenen Testreihenparameter. Die Tabellenzeilen sind in Abbildung 5.9 von oben nach unten angeordnet. Pro Testreihe standen, wie bereits in

Abschnitt 5.3, nur jeweils 5 Fälle zur Auswertung. Dennoch ist das Streuungsverhalten der Approximationsfaktoren pro Testreihe in Abbildung 5.9 deutlich erkennbar. Der einzelne Faktor eines Falls beschreibt dabei das tatsächliche Gewichtsverhältnis des für diesen Fall berechneten eindeutigen approximativen Matchings zum zugehörigen exakten Matching.

Die für den approximativen Algorithmus angegebene, theoretische obere Schranke des Approximationsfaktors von $2\log_3 n$ beträgt für 40 Knoten 6,7155256, für 500 Knoten 11,31356, für 1000 Knoten 12,57542, für 2000 Knoten 13,837279 und für 3000 Knoten 14,57542. Die in der Praxis gemessenen Approximationsfaktoren der Testreihen in Abbildung 5.9 liegen jedoch alle unterhalb von 1,4! Dass die theoretische obere Schranke in keiner der Testreihen erreicht wird, liegt unter anderem an der bereits in Abschnitt 4.2 hergeleiteten Unschärfe dieser Schranke für verteilungsgenerierte Graphen. Auf eine detaillierte Analyse der Approximationsfaktoren für verteilungsgenerierte Graphen wird hier verzichtet, da deren genaue Wertigkeit keinen maßgeblichen Einfluss auf die Anzahl der berechneten Cross-Matches der approximativen Matchings hat und damit aus Sicht des Cross-Match-Tests uninteressant ist.

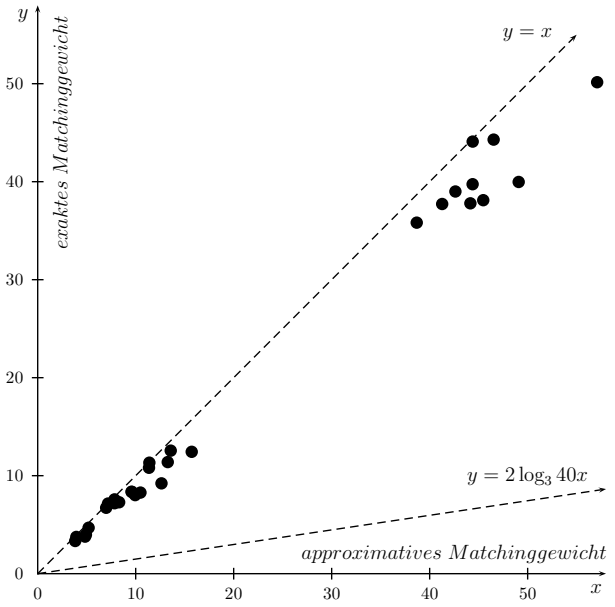


Abb. 5.10: Zusammenhang von exaktem und approximativem Matchinggewicht bei insgesamt 40 Stichprobenwerten.

Ebenso wie die Unschärfe der oberen Schranke des Approximationsfaktors von $2\log_3 n$ lässt sich auch die mit zunehmender Knotenanzahl abnehmende Streuung der Approximationsfaktoren auf die Tatsache zurückführen, dass die hier betrachteten Graphen verteilungsgeneriert sind. Jeder neu hinzukommende Datenpunkt folgt derselben Verteilung. Alle Datenpunkte sind also Ausprägungen identisch verteilter Zufallsvariablen. Je mehr Datenpunkte hinzukommen, desto klarer zeichnet sich nach dem *Gesetz der großen Zahlen* deren gemeinsame Verteilung ab. Das heißt, zwei mit derselben Verteilungsfunktion generierte Punktwolken ähneln sich zunehmend mit wachsender Punktezahl. Zunehmende Ähnlichkeit von Punktwolken gleicher Punktezahl kann hier verstanden werden als abnehmendes Gewicht eines perfekten, minimalen, bipartiten Matchings beider Punktwolken bezüglich eines festen Distanzmaßes d. Damit werden sich einerseits die exakten sowie auch andererseits

die approximativen Matchings mit zunehmender Knotenanzahl ähnlicher im Sinne einer abnehmenden Gewichtsdifferenz sich entsprechender Matchingkanten. Zwei Matchingkanten entsprechen sich, falls ihre Endpunkte beim Vergleich der zugrundeliegenden Punktwolken bipartit gepaart werden. Für Kanten eines Matchings, die auf diese Weise keine Entsprechung im zu vergleichenden Matching finden, existiert eine Kante, deren Endpunkte in jeweils lokalen Umgebungen der bipartiten Partner der eigenen Endpunkte liegen und die so als Entsprechung herangezogen werden kann. Die Streuung der Matchinggewichte und somit auch die Streuung der einzelnen Approximationsfaktoren geht damit zurück (siehe Tabelle 7.4).

Bei geringer Knotenanzahl können dagegen trotz der Generierung mit ein und derselben Verteilungsfunktion so stark differierende Graphen entstehen, dass die Gewichte der Matchings sehr viel breiter streuen, wodurch auch die Chance auf eine breitere Streuung der einzelnen Approximationsfaktoren steigt. Allerdings scheint für die hier untersuchten Graphen auch bei geringen Knotenanzahlen ein gewisser Zusammenhang zwischen exaktem und approximativem Matchinggewicht zu bestehen, der trotz breiterer Streuung eine Annäherung an die Schranke des Approximationsfaktors von $2\log_3 n$ verhindert. In Abbildung 5.10 sind die Matchinggewichte der Testreihen mit 40 Knoten aus Abbildung 5.9 als zweidimensionale Punktkoordinaten dargestellt. Jeder Punkt steht für einen Fall einer Testreihe. Die erste Winkelhalbierende markiert jene Fälle, bei denen approximatives und exaktes Matchinggewicht übereinstimmen, deren Matchinggewichte also einen Approximationsfaktor von eins ergeben. Die Gerade $y = 2\log_3 40$ repräsentiert all jene Fälle, deren approximatives Matchinggewicht das $2\log_3 40$-fache des exakten Matchinggewichts beträgt. Für die hier eingetragenen Testreihen geringer Knotenanzahl ist ein deutlicher Zusammenhang von exaktem und approximativem Matchinggewicht zu erkennen, der die einzelnen Approximationsfaktoren in ein sehr viel engeres Segment als das theoretisch zugelassene zwischen erster Winkelhalbierenden und oberer Schranke von $2\log_3 n$ zwingt.

Die Ballung der Punkte um ein oberes und unteres Zentrum entsteht durch die Wahl der Testreihenparameter, die dazu führt, dass die Matchings zweier Testreihen größere absolute Gewichte besitzen als die der übrigen (siehe Tabelle 7.4).

5.6 Laufzeitvergleich

Schließlich kann man auch die theoretische Laufzeitverbesserung des Algorithmus nach Wattenhofer und Wattenhofer mit einer Laufzeit in $O(n^2 \log n)$ im Vergleich zu Edmonds' exaktem Algorithmus mit einer Laufzeit in $O(nm + n^2 \log n)$ experimentell verdeutlichen. Dabei liegt die Laufzeit des Algorithmus von Edmonds für die in dieser Arbeit betrachteten Graphen in $O(n^3)$, da für vollständige Graphen $m \in O(n^2)$ gilt. Für den Laufzeitvergleich wurden 15 Testfälle aus zweidimensionalen, standardnormalverteilten Stichproben mit Umfängen von je 100 bis 1400 Werten berechnet und die jeweilige Laufzeit der Algorithmen inklusive des Einlesens des Eingangsgraphen und des Schreibens des Matchingergebnisses in Sekunden gemessen.

Abbildung 5.11 stellt die Laufzeitentwicklung beider Algorithmen für die sich aus den Stichprobenumfängen ergebenden Gesamtknotenanzahlen von 200 bis 2800 auf einer logarithmischen Skala dar. Den Messwerten liegt ein System mit AMD Athlon 64 Prozessor 3200+ und 1 GB RAM zugrunde. Als Betriebssystem wurde für dieses Experiment Knoppix 5.1 (Linux Kernel 2.16.9) verwendet. Trotz der nur groben Laufzeitmessung des exakten Algorithmus in Sekunden und des Vergleichs der Implementierung nach Gabow [Gab73] in C [Rot] mit einer Java-Implementierung

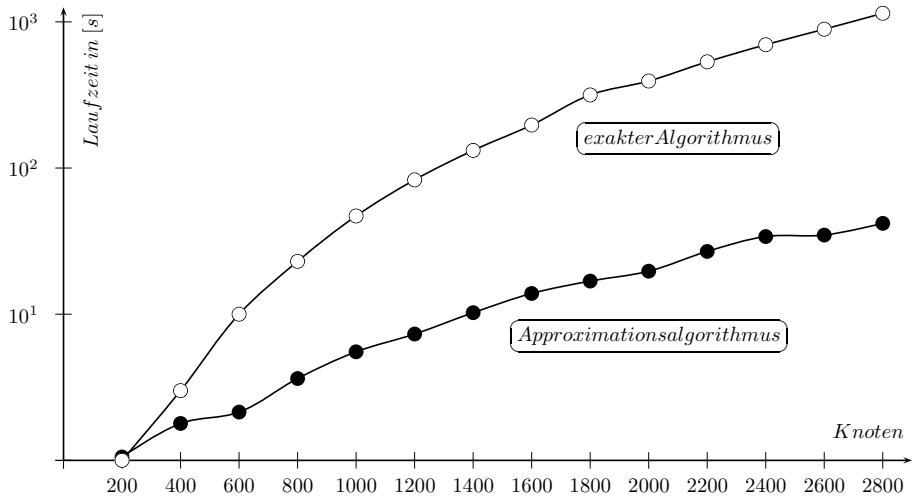


Abb. 5.11: Laufzeitvergleich bei großen Knotenanzahlen.

des Approximationsalgorithmus wird die Zeitersparnis durch die Anwendung des Algorithmus nach Wattenhofer und Wattenhofer in Abbildung 5.11 sehr deutlich! Während sich die Laufzeit des Approximationsalgorithmus im ein- und zweistelligen Sekundenbereich bewegt, beträgt die Laufzeit des exakten Algorithmus für große Stichprobenumfänge bis zum 27-fachen der Laufzeit des Approximationsalgorithmus (siehe Tabelle 7.5). Die im Vergleich zu Edmonds' Algorithmus etwas höhere Laufzeit des Approximationsalgorithmus für 200 Knoten resultiert aus der Tatsache, dass die gemessene Gesamtlaufzeit für diese Knotenanzahl noch vom Eingabe- und Ausgabeaufwand des Graphen bzw. des Matchings dominiert wird, diese Routinen in der Java-Implementierung jedoch nicht laufzeitoptimiert wurden.

Kapitel 6

Abschließende Bewertung

Der Cross-Match-Test bedient sich, als nichtparametrischer Zweistichprobentest, des exakten Matchingalgorithmus von Edmonds um eine geeignete Teststatistik zu definieren, mit deren Hilfe für zwei gegebene Stichproben mit gewisser Wahrscheinlichkeit korrekt entschieden werden kann, ob deren Verteilungen gleich sind. Ziel dieser Arbeit war es, zu untersuchen, ob an Stelle von Edmonds' exaktem Matchingalgorithmus auch andere, schnellere und einfacher zu implementierende, nicht exakte Algorithmen für die Anwendung innerhalb des Cross-Match-Tests geeignet sind. In Anlehnung an die Diplomarbeit von Gharibian-Saki [GS07] wurde in dieser Arbeit speziell das Verhalten des Approximationsalgorithmus nach Wattenhofer und Wattenhofer mit jenem des exakten Algorithmus von Edmonds experimentell verglichen.

6.1 Fazit

Die Analyse des Cross-Match-Tests ergab zwei Aspekte, die für die Aussagekraft des Algorithmenvergleichs maßgebend sind. Dies ist zum einen das Verhalten beider Algorithmen unter der Annahme, dass die Stichprobenverteilungen gleich sind, also unter Annahme der Nullhypothese, und zum anderen das asymptotische Verhalten der Algorithmen im Falle verschiedener Stichprobenverteilungen, also im Falle der sogenannten Alternativhypothese.

Falls beide Verteilungen sich gleichen, verlangt die theoretische Untersuchung der Teststatistik von einem geeigneten approximativen Algorithmus, dass er im Mittel genauso oft falsch bzw. richtig entscheidet wie Edmonds' exakter Algorithmus. Die experimentelle Untersuchung des Algorithmus nach Wattenhofer und Wattenhofer ergab, dass dieser unter der Nullhypothese tatsächlich das gewünschte Verhalten zeigt – allerdings erst nach einer leichten Modifikation, die für eine feste Punktmenge die Berechnung eines eindeutigen Matchings garantiert.

Unter der Alternativhypothese sollte ein geeigneter Algorithmus die Konsistenz des Testverfahrens gewährleisten, das heißt, die Wahrscheinlichkeit einer Fehlentscheidung sollte in diesem Falle gegen Null konvergieren für wachsende Stichprobenumfänge. Die experimentelle Auswertung verschiedener Testreihen hat auch hier gezeigt, dass der Rückgang der Fehlentscheidungen unter Verwendung des Approximationsalgorithmus durchaus dem Rückgang unter Verwendung des exakten Algorithmus vergleichbar ist.

Des Weiteren war zu beobachten, dass der tatsächliche Approximationsfaktor für alle in dieser Arbeit berechneten, approximativen Matchings nur gering vom optimalen Gewichtsverhältnis von eins abweicht. Der Approximationsalgorithmus nach Wattenhofer und Wattenhofer liefert also für die hier untersuchten Graphen sehr gute Approximationen der exakten Matchings. Für die einzelnen, experimentellen Approximationsfaktoren der Fälle gleicher Knotenanzahl innerhalb einer Testreihe war eine abnehmende Streuung bei zunehmender Anzahl der Knoten zu erkennen. Dabei ist jedoch zu beachten, dass die in dieser Arbeit untersuchten Knotenmengen allesamt verteilungsgeneriert sind und die zugehörigen Graphen dadurch bestimmte Eigenschaften aufweisen, welche für allgemeine Graphen nicht vorausgesetzt werden können. Das beobachtete Verhalten der Approximationsfaktoren lässt sich jedoch unter anderem auf eben diese Eigenschaften zurückführen.

Abschließend konnte auch die theoretische Laufzeitverbesserung des Approximationsalgorithmus gegenüber dem exakten Algorithmus von Edmonds experimentell belegt werden. Für die in dieser Arbeit getesteten Punktwolken betrug die Laufzeit des exakten Algorithmus für große Stichprobenumfänge bis zum 27-fachen der Laufzeit des Algorithmus nach Wattenhofer und Wattenhofer, während für kleine Stichprobenumfänge die Laufzeitersparnis kaum messbar war. Daher empfiehlt es sich, bei kleinen Stichprobenumfängen des Cross-Match-Tests auf den exakten Algorithmus zurückzugreifen, während die Verwendung des Approximationsalgorithmus für große Stichprobenumfänge eine klare Zeitersparnis bei nach wie vor guten Ergebnissen verspricht.

Die Untersuchungsergebnisse dieser Arbeit zeigen also, dass die Eignung nicht exakter Matchingalgorithmen für die Anwendung innerhalb des Cross-Match-Tests nicht generell ausgeschlossen ist. Jedoch muss der verwendete Algorithmus gewisse Voraussetzungen erfüllen. Der in dieser Arbeit untersuchte Approximationsalgorithmus kann mit geringem Aufwand so modifiziert werden, dass er diesen Voraussetzungen genügt. Die Verwendung des Algorithmus nach Wattenhofer und Wattenhofer und damit der Verzicht auf Exaktheit zu Gunsten einer schnellen Laufzeit und einer einfachen Implementierung ist daher aus Sicht des Cross-Match-Tests zulässig und sinnvoll.

6.2 Ausblick

Eine weiterführende, von der Anwendung der Matchingalgorithmen innerhalb des Cross-Match-Tests unabhängige Fragestellung könnte sich mit dem Algorithmus nach Wattenhofer und Wattenhofer in Bezug auf allgemeine Graphen beschäftigen. Die Untersuchungen dieser Arbeit haben gezeigt, dass die Wahlfreiheit, die der Algorithmus bei der Berechnung der Eulertouren gewährt, großen Einfluss auf das resultierende Matching hat. Eine genauere Untersuchung der Waldentstehung für allgemeine Graphen führt eventuell zu neuen Ideen, die eine weitere Verbesserung des Approximationsfaktors und der Laufzeit bewirken. Ein erster Schritt könnte darin bestehen, die verschiedenen Wahlmöglichkeiten des Startknotens bei der Berechnung der Eulertour hinsichtlich ihrer Auswirkungen auf den Approximationsfaktor zu bewerten. Des Weiteren scheint auch die Abarbeitungsreihenfolge der Knoten während der Tiefensuche ein gewisses Optimierungspotential bezüglich der Approximationsgüte zu bieten.

Kapitel 7

Anhang

	Verteilung	**Dimension**	**Korrelation**	**Korrelationsrichtungen**	**μ-Vektor**	**σ^2-Vektor**	**Freiheitsgrade**
1	$\mathcal{N}$	2	0	(1,2)	(0,0)	(1,1)	-
2	$\mathcal{N}$	2	0,5	(1,2)	(0,0)	(1,1)	-
3	$\mathcal{N}$	2	0,9	(1,2)	(0,0)	(1,1)	-
4	$\mathcal{N}$	10	0,9	alle	$(0)^{10}$	$(2)^{10}$	-
5	$\mathcal{N}$	2	0,5	(1,2)	(0,4)	(1,16)	-
6	$\mathcal{N}$	5	0,5	(1,2)	(0,4,2,0,0)	(1,16,4,1,1)	-
7	$\mathcal{N}$	2	0	(1,2)	(0,0)	(1,9)	-
8	$\mathcal{LN}$	2	0	(1,2)	(0,0)	(1,1)	-
9	$\mathcal{LN}$	2	0	(1,2)	(0,0)	(4,4)	-
10	$\mathcal{LN}$	2	0,5	(1,2)	(0,0)	(1/16,1)	-
11	$\mathcal{X}^2$	2	0	(1,2)	(0,0)	(1,1)	3
12	$\mathcal{X}^2$	2	0,5	(1,2)	(0,0)	(1,1)	8
13	$\mathcal{X}^2$	2	0	(1,2)	(0,0)	(1,1)	1
14	$\mathcal{X}^2$	7	0,4	alle	$(0)^7$	$(1)^7$	5
15	$\mathcal{X}^2$	10	0,4	alle	$(0)^{10}$	$(1)^{10}$	5
16	t	2	0	(1,2)	(0,0)	(1,1)	2
17	t	2	0,7	(1,2)	(0,0)	(1,1)	1

Tabelle 7.1: Testreihenparameter zur Verhaltensprüfung unter H_0.

	Verteilung	**Dimension**	**Korrelation**	**Korrelationsrichtungen**	**μ-Vektor**	**σ^2-Vektor**	**Freiheitsgrade**	**Werteanzahl**
A1	$\mathcal{N}$	2	-0,8	(1,2)	(0,0)	(1,1)	-	800
A2			-0,6					1200
B1	$\mathcal{N}$	2	-0,8	(1,2)	(0,0)	(1,1)	-	1200
B2			-0,6					1800
C1	$\mathcal{N}$	2	-0,8	(1,2)	(0,0)	(1,1)	-	800
C2			-0,7					1200
D1	$\mathcal{N}$	2	-0,8	(1,2)	(0,0)	(1,1)	-	1200
D2			-0,7					1800
A1	$\mathcal{X}^2$	3	0,5	alle	(0,0,0)	(1,1,1)	4	800
A2							3	1200
B1	$\mathcal{X}^2$	3	0,5	alle	(0,0,0)	(1,1,1)	4	1200
B2							3	1800
C1	$\mathcal{X}^2$	3	0,5	alle	(0,0,0)	(1,1,1)	4	800
C2							5	1200
D1	$\mathcal{X}^2$	3	0,5	alle	(0,0,0)	(1,1,1)	4	1200
D2							5	1800
A1	$\mathcal{LN}$	2	0,2	(1,2)	(0,0)	$(1/16)^2$	-	800
A2						$(9/100)^2$		1200
B1	$\mathcal{LN}$	2	0,2	(1,2)	(0,0)	$(1/16)^2$	-	1200
B2						$(9/100)^2$		1800
C1	$\mathcal{LN}$	2	0,2	(1,2)	(0,0)	$(1/16)^2$	-	800
C2						$(729/10000)^2$		1200
D1	$\mathcal{LN}$	2	0,2	(1,2)	(0,0)	$(1/16)^2$	-	1200
D2						$(729/10000)^2$		1800
A1	t	5	0	alle	$(0)^5$	$(1)^5$	10	1000
A2	$\mathcal{N}$						-	1000
B1	t	5	0	alle	$(0)^5$	$(1)^5$	10	1500
B2	$\mathcal{N}$						-	1500
C1	t	5	0	alle	$(0)^5$	$(1)^5$	25	1000
C2	$\mathcal{N}$						-	1000
D1	t	5	0	alle	$(0)^5$	$(1)^5$	25	1500
D2	$\mathcal{N}$						-	1500

Tabelle 7.2: Testreihenparameter zur Konsistenzprüfung unter H_1.

Testreihentupel	**Verteilung**	**Dimension**	**Korrelation**	**Korrelationsrichtungen**	**μ-Vektor**	**σ^2-Vektor**	**Freiheitsgrade**	**Werteverhältnis**
1	$\mathcal{N}$	2	0	(1,2)	(0,0) (0,2)	(1,1)	-	1:1
2	t $\mathcal{N}$	2	0,9 0	(1,2)	(0,0)	(1,1)	3 -	1:1
3	t $\mathcal{N}$	5	0	alle	$(0)^5$	$(1)^5$	3 -	1:1
4	$\mathcal{LN}$	2	0,2	(1,2)	(0,0)	$(1/16)^2$ $(9/100)^2$	-	2:3
5	$\mathcal{X}^2$	3	0,5	alle	(0,0,0)	(1,1,1)	4 3	2:3
6	$\mathcal{N}$	2	-0,8 -0,6	(1,2)	(0,0)	(1,1)	-	2:3

Tabelle 7.3: Testreihenparameter für sonstige Beobachtungen.

	Werte	**approximativ**					**exakt**				
		1. Wert	2. Wert	3. Wert	4. Wert	5. Wert	1. Wert	2. Wert	3. Wert	4. Wert	5. Wert
A	40	15,717	13,576	13,272	11,395	11,357	12,436	12,556	11,388	11,327	10,807
B	2000	98,325	102,07	102,64	102,98	100,23	80,31	84,250	85,643	83,264	83,013
C	3000	124,61	131,17	129,75	123,58	123,38	105,44	106,61	107,37	101,21	101,82
A	40	6,987	10,486	12,632	7,177	8,310	6,725	8,276	9,217	7,156	7,288
B	2000	65,319	70,909	68,551	66,308	73,465	55,441	60,714	58,705	55,925	59,344
C	3000	80,909	85,366	86,128	82,552	85,418	69,806	72,002	72,675	69,298	71,140
A	40	44,391	42,628	57,093	38,681	44,383	44,093	39,000	50,151	35,820	39,743
B	2000	1050,0	1067,3	1064,4	1129,1	1021,4	935,87	928,71	930,78	985,26	898,84
C	3000	1296,6	1273,4	1307,8	1259,3	1265,1	1110,1	1102,6	1128,4	1088,4	1098,6
A	40	3,941	4,921	4,834	3,828	5,196	3,750	3,978	3,779	3,335	4,687
B	2000	31,062	31,954	31,757	31,963	31,035	26,412	26,902	25,983	26,706	25,663
C	3000	40,568	38,659	40,339	38,446	38,891	33,084	33,486	33,680	32,521	32,343
A	40	44,155	41,277	46,529	49,084	45,469	37,792	37,724	44,295	39,972	38,112
B	2000	619,74	587,45	630,17	611,04	603,34	530,84	512,34	540,91	517,44	515,03
C	3000	889,62	884,26	897,38	882,68	902,96	767,55	759,75	750,97	748,36	760,06
A	40	7,842	7,840	9,624	9,570	9,946	7,189	7,580	8,340	8,359	8,016
B	2000	72,361	69,511	73,068	71,796	74,008	61,114	58,106	59,495	59,571	60,945
C	3000	86,110	88,438	88,143	88,611	86,355	72,487	72,921	71,936	73,388	72,383

Tabelle 7.4: Matchinggewichte zu Tabelle 7.3

Stichprobenumfang	**Laufzeiten**		**Verhältnis**
	exakt	**approximativ**	**exakt : approx.**
200	0	1,055	0,00
400	3	1,789	1,68
600	10	2,140	4,67
800	23	3,631	6,33
1000	47	5,531	8,5
1200	83	7,318	11,34
1400	132	10 ,241	12,89
1600	197	13,867	14,21
1800	316	16,851	18,75
2000	394	19,700	20,00
2200	533	26,886	19,82
2400	698	33,988	20,54
2600	891	34,789	25,61
2800	1145	41,841	27,37

Tabelle 7.5: Laufzeiten zu Abbildung 5.11 in Sekunden.

Literaturverzeichnis

[CLRS07] Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest und Clifford Stein: *Algorithmen- Eine Einführung*, Kapitel 35, Seiten 1025–1056. Oldenbourg Wissenschaftsverlag, 2007. 18

[DH03] Drake, Dorothea E. und Stefan Hougardy: *A Simple Approximation Algorithm for the Weighted Matching Problem.* Information Processing Letters 85, 211-213, 2003. 18

[Edm65a] Edmonds, Jack: *Maximum Matching and a Polyhedron with 0,1-Vertices.* Journal of Research of the National Bureau of Standards, 69 B:125–130, 1965. 18

[Edm65b] Edmonds, Jack: *Paths, Trees and Flowers.* Canadian Journal of Mathematics, 17:449–467, 1965. 18

[FR79] Friedman, Jerome H. und Lawrence C. Rafsky: *Multivariate Generalizations of the Wald-Wolfowitz and Smirnov Two-Sample Tests.* The Annals of Statistics, 7:697–717, 1979. 10

[Gab73] Gabow, Harold N.: *Implementation of Algorithms for Maximum Matching on Nonbipartite Graphs.* Doktorarbeit, Stanford University, 1973. 4, 15, 37

[Gab76] Gabow, Harold N.: *An Efficient Implementation of Edmonds' Algorithm for Maximum Matching on Graphs.* Journal of the ACM, 23(2):221–234, 1976. 4, 15, 17

[Gab90] Gabow, Harold N.: *Data Structures for Weighted Matching and Nearest Common Ancestors with Linking.* In: *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, Seiten 434–443. Society for Industrial and Applied Mathematics, 1990. 4, 17

[GC03] Gibbons, Jean Dickinson und Subhabrata Chakraborti: *Nonparametric Statistical Inference.* CRC Press, 2003. 10

[GS07] Gharibian-Saki, Devin: *Abstandsgraphen und das multivariate nichtparametrische Zwei-Stichproben-Problem.* Diplomarbeit, Universität Karlsruhe, 2007. 4, 39

[Hen88] Henze, Norbert: *A multivariate Two-Sample Test Based on the Number of Nearest Neighbor Type Coincidences.* The Annals of Statistics, 16:772–783, 1988. 10

[Hen95] Henze, Norbert: *Skript zur Vorlesung Stochastik I.* Universität Karlsruhe, 1995. 2, 8

[Kre91] Krengel, U.: *Einführung in die Wahrscheinlichkeitstheorie und Statistik*. Verlag F. Vieweg & Sohn, 1991. 7

[MS91] Marcotte, Odile und Subhash Suri: *Fast Matching Algorithms for Points on a Polygon*. Siam Journal on Computing, 20(3):405–422, 1991. 18

[NGH72] Nozicka, Frantisek, Jürgen Guddat und Horst Hollatz: *Theorie der linearen Optimierung*. Akademie-Verlag, 1972. 18

[PS82] Papadimitriou, Christos H. und Kenneth Steiglitz: *Combinatorial Optimization - Algorithms and Complexity*, Kapitel 11: Weighted Matching, Seiten 247–270. Prentice-Hall, 1982. 1, 15, 17

[Ros05] Rosenbaum, Paul R.: *An Exact Distribution-Free Test Comparing Two Multivariate Distributions Based on Adjacency*. Journal of the Royal Statistical Society, 67:515–530, 2005. 1, 2, 4, 10, 14

[Rot] Rothberg, Ed: *Solver for the Maximum Weight Matching Problem*. `http://elib.zib.de/pub/Packages/mathprog/matching/weighted/index.html`. 4, 15, 37

[Sch86] Schilling, M. F.: *Multivariate Two-Sample Tests Based on Nearest Neighbours*. Journal of the American Statistical Association, 81:799–806, 1986. 10

[Tar75] Tarjan, Robert Endre: *Efficiency of a Good But Not Linear Set Union Algorithm*. Journal of the Associatmn for Computing Machinery, 22(2):215–225, 1975. 25

[Vai88] Vaidya, Pravin M.: *Geometry Helps in Matching*. In: *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing (STOC)*, Seiten 422–425. ACM Press, 1988. 17, 18

[Vaz01] Vazirani, Vijay V.: *Approximation Algorithms*. Springer Verlag, 2001. 18

[Wan06] Wanka, Rolf: *Approximationsalgorithmen - Eine Einführung*. B. G. Teubner Verlag, 2006. 18

[WW04] Wattenhofer, Mirjam und Roger Wattenhofer: *Fast and Simple Algorithms for Weighted Perfect Matching*, 2004. 2, 4, 18