universität innsbruck
Universitäts- und Landesbibliothek Tirol

Universitäts- und Landesbibliothek Tirol

Ein verbessertes Verfahren zur Feststellung der Isomorphie endlicher Graphen

Hinteregger, Josef

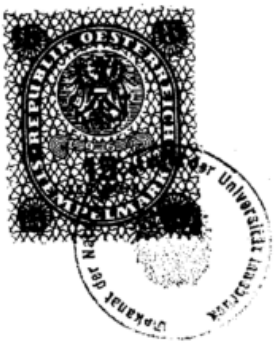
1976

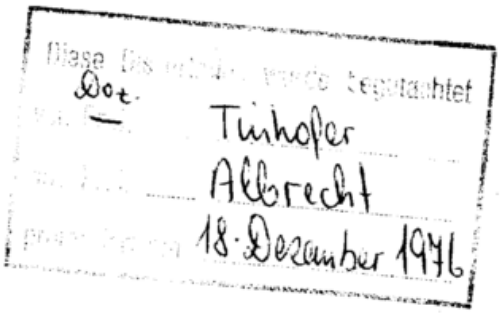
urn:nbn:at:at-ubi:2-12744

EIN VERBESSERTES VERFAHREN ZUR FESTSTELLUNG DER ISOMORPHIE ENDLICHER GRAPHEN.

Josef Hinteregger

Dissertation zur Erlangung des Doktorgrades der philosophischen Fakultät der Universität Innsbruck, eingereicht am Institut für Informatik und numerische Mathematik im April 1976





UB INNSBRUCK
+C177695200

(Dg 12.486)



Vorwort

Um einen Isomorphismus zwischen zwei endlichen Graphen zu finden, werden üblicherweise verschiedene isomorphismus-invariante Eigenschaften des Graphen berechnet, durch die die Anzahl der in Frage kommenden Abbildungen eingeschränkt wird.

Das in dieser Arbeit beschriebene Verfahren berechnet systematisch alle Eigenschaften aus einer bestimmten Klasse von isomorphismus-invarianten Eigenschaften, und zwar der Klasse der Grade der Knoten bezüglich aller Graphen, die aus den gegebenen Graphen mit Hilfe der Operationen Transposition(=Umkehrung), Vereinigung, Durchschnitt, Komplement und Produkt(Hintereinanderausführung) gebildet werden können. Dies stellt eine Verbesserung dar gegenüber der willkürlichen Auswahl von Invarianten aus der genannten Klasse, wie sie in den meisten Verfahren vorgenommen wird, soweit diese Verfahren nicht Eigenschaften verwenden, die nicht in dieser Klasse enthalten sind.

Weiters wird ein Computer-Programm zu diesem Verfahren angegeben. Damit konnte an den Beispielen 4.12 und 4.13(Siehe auch Seite 61) auch eine praktische Verbesserung beobachtet werden.

Inhalt

1. Einführung 1

2. Operationen mit Relationen 7

Basis eines Mengensystems 1o

3. Einiges über Partitionen 12

4. Konstruktion einer disjunkten Basis von $H(\Delta,G_1,..,G_k)$ 17

Beispiele 28

5. Programmierung des Verfahrens 36

Zeitabschätzung 42

Programm 44

weitere Beispiele 61

Literaturverzeichnis 66

1. Kapitel. Einführung

1.1 Definition

Ein endlicher Graph ist ein Paar (X,K) einer nichtleeren endlichen Menge X und einer endlichen Menge $K \subset X \times X \times \mathbb{N}$, wobei gilt: $\forall (x,y,i) \in K \; \forall j,\ 1 \leq j \leq i$: $(x,y,j) \in K$.
Die Elemente von X heißen Knoten, diejenigen von K Kanten.
Für eine Kante $(x,y,j) \in K$ heißt
x Ausgangsknoten,
y Endknoten und
j fortlaufende Zählung der Kanten von x nach y.
Eine Kante (x,x,j) heißt Schleife.
Ein Graph (X,K) heißt symmetrisch oder ungerichtet, falls gilt: $\forall (x,y,j) \in K$: $(y,x,j) \in K$.

In dieser Arbeit werden nur endliche Graphen behandelt.

1.2 Definition

Ein Graph (X,K) heißt einfach, falls gilt:
$\forall (x,y,j) \in K$: $j=1$ ("Von x nach y höchstens eine Kante").
Üblicherweise wird für einfache Graphen die Menge
$K' := \{(x,y) \in X \times X \mid (x,y,1) \in K\}$ als Kantenmenge definiert.
K' ist eine Relation auf X und ist durch K eindeutig festgelegt, ebenso K durch K'.
Ein einfacher Graph (X,K) heißt vollständig, falls
$\forall x,y \in X,\ x \neq y$: $(x,y,1) \in K$.

1.3 Definition

Ein benannter Graph ist ein Graph (X,K) zusammen mit einer Benennungsfunktion $l: K \cup X \longrightarrow L$, L Menge.

Die Angabe der Kantenmenge K erfolgt durch Aufzählung der Elemente oder durch die Adjazenzmatrix (auch Inzidenzm.)
(a_{ij}) mit $a_{ij} := |\{(x,y,j) \in K \mid x=x_i, y=y_j\}|$, falls $X=\{x_1,..,x_n\}$.
Für einen einfachen Graphen (X,K) sind das die Werte der charakteristischen Funktion von K': $a_{ij} = \chi_{K'}(x_i,x_j)$.

Die Frage, um die es in dieser Arbeit geht, ist folgende: Wie kann man anhand zweier gegebener Graphen feststellen, ob diese isomorph sind, d.h. sich nur durch die Bezeichnung der Knoten unterscheiden, oder ob sie wesentlich verschieden sind?

Diese Frage stellt sich bei den verschiedensten Struktur-Vergleichen:
In der Automatentheorie wird die Funktion eines Automaten durch den "Zustands-Überführungs-Graphen" beschrieben, welcher den jeweils nächsten Zustand des Automaten angibt (Siehe [6]).
Eine chemische Formel ist ein benannter Graph:
Die Atome(=Knoten) haben Namen, die Bindungen(=Kanten) Wertigkeiten. Bei einer gegebenen Formel stellt sich die Frage, ob diese bereits in einer Bibliothek vorhanden ist oder nicht. ([8])
Auch elektrische Schaltungen können als benannte Graphen betrachtet werden. Das Problem des Vergleichs einer "neuen" Schaltung mit den bereits bekannten, in einer Bibliothek gespeicherten Schaltungen stellt sich hier etwa im Patentamt.
Ebenso lassen sich Strukturen in formalen Sprachen durch Graphen beschreiben.

1.4 Definition

Seien (X,K_1), (Y,K_2) zwei Graphen, möglicherweise mit den Benennungen l_1 und l_2.

Eine bijektive Abbildung $s: X \longrightarrow Y$ $\quad(\Rightarrow |X|=|Y|)$

heißt Isomorphismus von (X,K_1) nach (Y,K_2), falls gilt:

$\forall (x,y,j) \in X\times X\times \mathbb{N}: (x,y,j) \in K_1 \iff (s(x),s(y),j) \in K_2$.

Falls die Graphen benannt sind, muß zusätzlich gelten:

$\forall x\in X: l_1(x) = l_2(s(x))$ und $\forall (x,v,j) \in K: l_1(x,y,j) = l_2(s(x),s(y),j)$

Ein Isomorphismus von (X,K) nach (X,K) heißt
Automorphismus von (X,K).
Die Partition von X, welche durch die Äquivalenzklassen der Äquivalenzrelation

$x \sim y \quad :\Longleftrightarrow$ es gibt einen Automorphismus s von (X,K) mit $y \doteq s(x)$.

erzeugt wird, heißt Automorphismenpartition. (bezüglich K)

Da wir es nur mit endlichen Graphen zu tun haben, kann man zu jedem Graphen (X,K) mit $|X| = n$ einen dazu isomorphen Graphen (X',K) finden mit $X' = \{1,..,n\} \subset \mathbb{N}$.
Für das Isomorphismenproblem genügt es also, Graphen mit einer Knotenmenge $X = \{1,..,n\}$ zu betrachten.

Das Isomorphismenproblem ist theoretisch immer lösbar:
da X endlich ist, ist auch die Menge aller Anordnungen (Permutationen) von X endlich, und es brauchen nur alle diese Anordnungen durchprobiert zu werden;
da die Mächtigkeit dieser Menge aber gleich n! ist, bedeutet "endlich" bereits für n=15 "einige Jahrzehnte".
Man wird also die Anordnungen nicht wahllos durchprobieren, sondern ihre Anzahl einschränken durch die Berechnung verschiedener Isomorphismus-invarianter Eigenschaften der Knoten, Kanten etc.: Durch diese Invarianten wird eine Partition der Knotenmenge X (der Menge X×X) erzeugt, sodaß dann nur noch Knoten (Kanten) der entsprechenden Partitionsmengen einander zugeordnet werden können. Falls eine Partitionsmenge des einen Graphen beim anderen keine entsprechende hat, so schließt dies einen Isomorphismus bereits aus.
Alle bisher bekannten Verfahren arbeiten nach diesem Prinzip, sie unterscheiden sich nur durch die gewählten Invarianten.
In günstigen Fällen erreicht man auf diese Art die jeweilige Automorphismenpartition, die aber selber erst eine relativ grobe Beschreibung der Automorphismengruppe ist.

Gesucht wäre ein vollständiges System von Invarianten, d.h. ein System von Invarianten, aus deren Gleichheit für zwei Graphen die Existenz eines Isomorphismus folgt. Es ist nicht einmal ein System von Invarianten bekannt, die sicher zur Automorphismenpartition führen.

Aus der Algebra ist die Menge der Invariantenteiler als vollständiges System für die Ähnlichkeit zweier Matrizen über Z bekannt: Falls zwei Matrizen $A,B \in \mathbb{Z}_{(n,n)}$ dieselben Invariantenteiler besitzen, so gibt es eine n,n-Matrix P mit $\det(P) = 1$, sodaß $B = PAP^{-1}$ ist.

Zwei einfache Graphen (X,G_1), (X,G_2) mit den Adjazenzmatrizen A und B sind genau dann isomorph, falls es eine Permutationsmatrix P (in jeder Zeile und Spalte genau eine 1) gibt, sodaß $B = PAP^{-1}$ ist.

"det(P) = 1" ist jedoch leider nicht hinreichend dafür, daß P eine Permutationsmatrix ist.
In [6] werden Zusammenhänge zwischen den Invariantenteilern der Adjazenzmatrix und den Eigenschaften des Graphen untersucht.

Die Invariantenteiler der Adjazenzmatrix liefern zwar eine notwendige Bedingung für die Existenz eines Isomorphismus, jedoch keine Information darüber, welche Knoten einander zugeordnet werden können im Fall, daß die Invariantenteiler einen Isomorphismus nicht ausschließen. Da ihre Berechnung einen beträchtlichen Rechenaufwand erfordert, wird diese invariante Eigenschaft nur selten verwendet.

Die am leichtesten zu bestimmende invariante Knoteneigenschaft ist die folgende:

1.5 Definition (X,G) Graph.

$N_G: X \longrightarrow \{0,..,|X|\}$

$x \longmapsto |\{z \in X | \exists j \in N: (x,z,j) \in G\}|$ bzw. für einfache Graphen:

$|\{z \in X | (x,z) \in G\}|$, die Anzahl der Nachfolger von x,

heißt Ausgangsgrad oder Grad von x bezüglich G.

1.6 Folgerung

Seien (X,A), (X,B) zwei einfache Graphen mit $A,B \subset X\times X$ und $A \cap B = \emptyset$.

Dann ist $\forall\, x\in X\colon N_{A\cup B}(x) = N_A(x) + N_B(x)$.

Beweis $\{z\in X \mid (x,z)\in A\cup B\} = \{z\in X \mid (x,z)\in A\} \dot\cup \{z\in X \mid (x,z)\in B\}$.

Der folgende Satz zeigt, daß man sich für das Isomorphismenproblem auf k-Tupel einfacher Graphen beschränken kann:

1.7 Satz

Seien (X,G_1) und (X,G_2) zwei Graphen mit den Benennungen

$l_1\colon X \cup G_1 \longrightarrow \mathbb{N}$ und $l_2\colon X \cup G_2 \longrightarrow \mathbb{N}$.

Eine Bijektion $s\colon X \longrightarrow X$ ist ein Isomorphismus von (X,G_1) nach (X,G_2) genau dann, wenn gilt:

$\forall\ (a,b,c) \in \{1,..,\max\{j \mid (x,y,j)\in G_1\}\} \times l_1(G_1) \times l_1(X)$:

s ist ein Isomorphismus der einfachen Graphen (X,H^1_{abc}) und (X,H^2_{abc}), wobei

$H^i_{abc} := \{(x,y)\in X\times X \mid (x,y,a) \in G_i \wedge l_i(x,y,a)=b \wedge l_i(x)=c\}$.

Beweis

$\Rightarrow$: $\underline{(x,y) \in H^1_{abc}} \iff (x,y,a)\in G_1 \wedge l_1(x,y,a)=b \wedge l_1(x)=c$

$\Big((x,y,a)\in G_1 \iff (s(x),s(y),a)\in G_2,\ l_1(x,y,a)=l_2(s(x),s(y),a)=b,\ l_1(x) = l_2(s(x)) = c \Big)$

$\iff \underline{(s(x),s(y)) \in H^2_{abc}}$

$\Leftarrow$: $(x,y,j) \in G_1,\ b := l_1(x,y,j),\ c:=l_1(x). \iff$

$(x,y) \in H^1_{jbc} \iff (s(x),s(y)) \in H^2_{jbc}$

$\iff (s(x),s(y),j) \in G_2,\ l_2(s(x),s(y),j)=l_1(x,y,j)=b,\ l_2(s(x)) = l_1(x) = c$.

Wir werden uns also jetzt auf k-Tupel einfacher Graphen $((X,G_1),\ldots,(X,G_k))$ beschränken, falls die zugehörige Knotenmenge X außer Zweifel steht, sogar auf die k-Tupel $(G_1,..,G_k)$ der jeweiligen Kantenmengen oder Relationen(1.2!).

Tinhofer hat in [9] gezeigt, daß die Grade bezüglich aller Graphen aus $E := \left\{ K \subseteq X \times X \;\middle|\; \begin{array}{l} s \text{ ist Automorphismus von } (G_1,..,G_k) \\ \Rightarrow s \text{ ist Automorphismus von } K \end{array} \right\}$ ausreichen, die Automorphismenpartition der Knotenmenge X zu bestimmen. Diese Menge ist gegenüber den Operationen von Definition 2.1 (Durchschnitt, Produkt etc.) nach Satz 2.5 abgeschlossen.

In meiner Arbeit habe ich die Menge $H(\Delta,G_1,..,G_k)$ (Def 2.4) der mit den Operationen von 2.1 von $G_1,..,G_k,\Delta$ ableitbaren Relationen untersucht. Durch die Grade bezüglich der Relationen aus $H(\Delta,G_1,..,G_k)$ kommt man jedoch nur in bestimmten Fällen (Satz 4.11) zur Automorphismenpartition. Daß dies nicht allgemein der Fall ist, zeigt der symmetrische Graph der Ordnung 26 aus [3], siehe 4.15.

In Kapitel 4 wird ein Verfahren zur Konstruktion einer disjunkten Basis von $H(\Delta,G_1,..,G_k)$ angegeben. Diese Basis enthält eine Partition der Diagonale Δ, durch die wiederum eine Partition der Knotenmenge $X = \{1,..,n\}$ erzeugt wird, die nach Satz 4.8 durch keinen Grad bezüglich irgendeiner Relation $K \in H(\Delta,G_1,..,G_k)$ mehr verfeinert werden kann. Die disjunkte Basis ermöglicht also die systematische Verwendung der Grade bezüglich aller Relationen $K \in H(\Delta,G_1..G_k)$, während die bisher bekannten Verfahren als Invarianten hauptsächlich die Grade bezüglich willkürlich aus $H(\Delta,G_1..G_k)$ ausgewählter Relationen verwenden und mit diesen die Knotenpartition verfeinern. Auch die Berechnung der Knoten-Quotienten-Graphen nach Corneil([1],[2]) führt nicht aus $H(\Delta,G_1..G_k)$ hinaus, wie in der Bemerkung nach 4.12 angedeutet wird. Zum Unterschied von dem in [5] beschriebenen Verfahren wird bei der Konstruktion der Basis ganz X×X aufgeteilt und nicht nur, wie dort, die Kantenmenge. Außerdem ist durch Satz 4.4 gesichert, daß keine Eigenschaften mehr separat berechnet werden müssen, dies geschieht schon durch die fortschreitende Partition von X×X .

2. Kapitel. Operationen mit Relationen, Basis eines Mengensystems.

2.1 Definition

Seien $A,B \subset X\times X$ Relationen auf der Menge $X = \{1,..,n\}$, $(a_{ij}),(b_{ij}) \in \{o,1\}_{(n,n)}$ die zugehörigen o-1-Adjazenzmatrizen.

Operation auf $P(X\times X)$	entsprechende Operation auf $\{o,1\}_{(n,n)}$ mit Ergebnis (c_{ij})
1. Komplement $\bar{A} = X\times X \setminus A$	$c_{ij} = \bar{a}_{ij}$ entsprechende
2. Vereinigung $A \cup B$	$c_{ij} = a_{ij} \vee b_{ij}$ boole'sche
3. Durchschnitt $A \cap B$	$c_{ij} = a_{ij} \wedge b_{ij}$ Operation
4. Umkehrung oder Transposition	
$A^T := \{(x,y) \mid (y,x) \in A\}$	$c_{ij} = a_{ji}$ (transponierte M.)
5. Produkt oder Hintereinanderausführung	
$A \circ B :=$ $\{(x,y) \mid \exists\, z\in X: (x,z)\in A \wedge (y,z)\in B\}$	$c_{ij} = \begin{cases} 1 & \exists\, k\in X: a_{ik}\wedge b_{ik}=1 \\ o & \text{sonst} \end{cases}$

Um das Ergebnis einer Multiplikation von (a_{ij}) und (b_{ij}) über den ganzen Zahlen verwenden zu können, definieren wir noch das folgende

6. (m)-Produkt (m=O..n)	
$A\ (m)\ B :=$ $\{(x,y) \mid m = \lvert\{z\in X \mid (x,z)\in A \wedge (z,y)\in B\}\rvert\}$	$c_{ij} = \begin{cases} 1 & m = \lvert\{k\in X \mid a_{ik}\wedge b_{kj}=1\}\rvert \\ o & \text{sonst} \end{cases}$
	$[m = \lvert\{k\in X\ a_{ik}\wedge b_{kj}=1\}\rvert$ $\leftrightarrow$ (jetzt $a_{ij}, b_{ij} \in \mathbb{Z}$) $m = \sum_{k\in X} a_{ik}\cdot b_{kj}]$

Bemerkung Die Einheit bezüglich des Produktes 2.2.5 heißt Diagonale: $\Delta := \{(x,x) \mid x\in X\} \subset X\times X$, $\forall\, K \subset X\times X$: $K\circ\Delta = \Delta\circ K = K$; ihre Adjazenzmatrix ist die Einheitsmatrix.

2.2 Satz

Zwischen den soeben definierten Operationen gelten folgende Beziehungen: $(A,B \subset X\times X)$

1. De-Morgan-Regeln, Distributivität von $\cap$ und $\cup$.
2. $[\overline{A}]^T = \overline{[A^T]}$, $(A \cup B)^T = A^T \cup B^T$ und $(A \cap B)^T = A^T \cap B^T$.
3. $[A \circ B]^T = B^T \circ A^T$ und $[A\ (m)\ B]^T = B^T\ (m)\ A^T$.
4. $A \circ B = \bigcup_{j=1}^{n} A(j)B$.
5. $A(i)B \cap A(j)B \neq \emptyset \Rightarrow i=j$.

Seien zusätzlich $B_1, B_2 \subset X\times X$ und $B_1 \cap B_2 = \emptyset$.

6. $A\ (m)\ \left[B_1 \cup B_2\right] = \bigcup_{\substack{0\leq m_1, m_2 \leq m \\ m_1+m_2=m}} \left[A(m_1)B_1\right] \cap \left[A(m_2)B_2\right]$.
7. $\left[B_1 \cup B_2\right]\ (m)\ A = \bigcup_{\substack{0\leq m_1, m_2 \leq m \\ m_1+m_2=m}} \left[B_1(m_1)A\right] \cap \left[B_2(m_2)A\right]$.

Beweis

1. bis 3. bekannt, bzw. Einsetzen der Definitionen.
4. $\exists\, z\in X: (x,z)\in A \wedge (z,y)\in B \iff$
   $1 \leq |\{z\in X \mid (x,z)\in A \wedge (z,y)\in B\}| \leq |X| = n$.
5. $(x,y) \in [A(i)B] \cap [A(j)B] \Rightarrow$
   $i = |\{z\in X \mid (x,z)\in A \wedge (z,y)\in B\}| = j$.
6. $C(x,y) := \{z\in X \mid (x,z)\in A \wedge (z,y) \in B_1 \cup B_2\}$
   $D(x,y) := \{z\in X \mid (x,z)\in A \wedge (z,y)\in B_1\}$
   $E(x,y) := \{z\in X \mid (x,z)\in A \wedge (z,y)\in B_2\}$

Es gilt dann: $C(x,y) = D(x,y) \cup E(x,y)$, $D(x,y) \cap E(x,y) \neq \emptyset$, also $|C(x,y)| = |D(x,y)| + |E(x,y)|$ (*).

$\subseteq$: $(x,y) \in A(m)[B_1 \cup B_2] \Rightarrow \quad |C(x,y)| = m \quad \Rightarrow$

$(m_1 := |D(x,y)|, m_2 := |E(x,y)| \quad m_1+m_2 \overset{(*)}{=} |C(x,y)| = m) \Rightarrow$

$\Rightarrow \exists\, m_1,m_2: 0 \leq m_1,m_2 \leq m,\ m_1+m_2=m,\ (x,y) \in A(m_1)B_1 \wedge$

$\wedge\ (x,y) \in A(m_2)B_2 \quad \Rightarrow$

$\Rightarrow$ (x,y) rechte Seite.

$\supseteq$: $(x,y) \in A(m_1)B_1 \wedge (x,y) \in A(m_2)B_2$, $m_1+m_2=m \quad \Rightarrow$

$m_1 = |D(x,y)| \wedge m_2 = |E(x,y)| \qquad \underline{|C(x,y)|} \overset{(*)}{=} m_1+m_2 = m$

$\Rightarrow$ $(x,y) \in A(m)[B_1 \cup B_2]$.

7.: 6. auf $A^T(m)[B_1^T \cup B_2^T]$ anwenden und die Gleichung transponieren (Regeln 2. und 3.).

2.3 Folgerung

Seien $A,B \subset X \times X$, $A = \bigcup_{i \in I_A} A_i$, $B = \bigcup_{i \in I_B} B_i$,
$(A_i)_{i \in I_A}$ disjunkt und $(B_i)_{i \in I_B}$ disjunkt.

Dann gilt:

$$A(m)B = \underbrace{\qquad}_{(m_{st}) \in I} \overbrace{\qquad}^{(s,t) \in I_A \times I_B} A_s(m_{st})B_t$$

$$\text{mit} \quad I := \{(m_{st}) \in \{0..m\}^{I_A \times I_B} \mid m = \sum_{(s,t) \in I_A \times I_B} m_{st}\}$$

Beweis:

Aus 2.2.6 erhält man durch Induktion nach $|I_B|$:

a) $$A(m)B = \underbrace{\qquad}_{(m_t) \in I_1} \overbrace{\qquad}^{t \in I_B} A(m_t)B_t,$$

$$I_1 := \{(m_t) \in \{0..m\}^{I_B} \mid m = \sum_{t \in I_B} m_t\}$$

und aus 2.2.7 durch Induktion nach $|I_A|$:

b) $$A(m_t)B_t = \underbrace{\qquad}_{(m_{st}) \in I_2} \overbrace{\qquad}^{s \in I_A} A_s(m_{st})B_t \ ,$$

$$I_2 := \{(m_{st}) \in \{0..m\}^{I_A} \mid m_t = \sum_{s \in I_A} m_{st}\}$$

Die Behauptung folgt nun durch Einsetzen von b) in a) und Anwendung des Distributivgesetzes für $\cap$ und $\cup$.

Beweis(2.5)

$\forall\ (x,y)\in X\times X$: $(x,y)\in A \iff (s(x),s(y))\in A'$
$(x,y)\in B \iff (s(x),s(y))\in B'$.

Daß dann $(x,y)\in \overline{A}$, A^T, $A\cup B$, $A\cap B$ $\iff (s(x),s(y))\in \overline{A'}$, A'^T, $A'\cup B'$, $A'\cap B'$

ist leicht zu sehen.

$$(x,y)\in A(m)B \iff m = |\{z\in X|\ (x,z)\in A \wedge (z,y)\in B\}| =$$
$$= |\{z\in X|(s(x),s(z))\in A' \wedge (s(z),s(y))\in B'\}| =$$
$$=(s \text{ bijektiv}) |s(\{z\in X|(s(x),s(z)\in A'\wedge (s(z),s(y))\in B'\})|=$$
$$= |\{s(z)\in X|(s(x),s(z)\in A'\wedge (s(z),s(y))\in B'\}| = m \iff$$
$$\iff (s(x),s(y))\in A'(m)B' .$$

$(x,y)\in A\circ B \iff (s(x),s(y))\in A'\circ B'$ gilt dann wegen 2.2.4 .

2.6 Definition

Seien M und $\mathcal{B}$ Mengensysteme.

$\mathcal{B}$ heißt Basis von M, falls gilt:

$$\forall M\in M\ \exists \mathcal{B}'\subset\mathcal{B}: M = \bigcup_{B\in\mathcal{B}'} B \text{ und } \emptyset\notin\mathcal{B}.$$

Eine Basis heißt disjunkt, falls $\forall B,C\in\mathcal{B}: B\cap C\neq\emptyset \Rightarrow B=C$.

2.7 Satz

Falls für eine Basis $\mathcal{B}$ von M zusätzlich gilt:

$\mathcal{B}\subset M$ und $\mathcal{B}$ disjunkt, so ist diese disjunkte Basis $\mathcal{B}$ eindeutig.

Beweis

Sei $C\subset M$ eine weitere disjunkte Basis von M.

Sei $B_o\in\mathcal{B}$: $\exists\ C'\subset C: B_o = \bigcup_{C\in C'} C = (\forall\ C\in C'\ \exists\ \mathcal{B}_C\subset\mathcal{B}: C = \bigcup_{B\in\mathcal{B}_C} B)$

$$B_o = \bigcup_{C\in C'}\ \bigcup_{B\in\mathcal{B}_C} B \Rightarrow \forall\ C\in C'\ \forall\ B\in\mathcal{B}_C: B_o\cap B\neq\emptyset \Rightarrow B=B_o$$

$\forall\ C\in C'$: $\mathcal{B}_C = \{B_o\}, C = B_o\in C$, also $\mathcal{B}\subset C$ und durch Vertauschen von $\mathcal{B}$ und C $C\subset\mathcal{B}$.

Für vorgegebene Relationen wird jetzt eine "Hülle" bezüglich der Operationen von 2.1 definiert:

2.4 Definition

Seien $G_1,..,G_k \subset X\times X$ Relationen auf der Menge $X = \{1,..,n\}$, Δ die Diagonale von X.

Dann bezeichne $H(\Delta,G_1,..,G_k)$ die Menge aller aus $\Delta, G_1,..,G_k$ mittels der Operationen von 2.1 ableitbaren Relationen auf X.

Formal: $(H_i)_{i\in\mathbb{N}}$ sei eine Folge von Teilmengen von X×X,

definiert durch: $H_o := \{\Delta,G_1,..,G_k\}$

$$H_{i+1} := H_i \cup \{G\subset X\times X \mid \left[\exists\, A\in H_i,\ G=\bar{A} \vee G=A^T\right] \vee \left[\exists\, A,B\in H_i,\ G=A\cup B \vee G=A\cap B \vee \vee(\exists\, m,\ o\leq m\leq n:\ G=A(m)B)\right]\}$$

$$(\forall\, i\in\mathbb{N}:\ H_i \subset P(X\times X) \Rightarrow \bigcup_{i\in\mathbb{N}} H_i \subset P(X\times X))$$

$$H(\Delta,G_1,..,G_k) := \bigcup_{i\in\mathbb{N}} H_i \quad \text{endlich.}$$

Bemerkung: Auf Grund von 2.2.4 ist $H(\Delta,G_1,..,G_k)$ auch gegenüber "o" abgeschlossen.

Die soeben definierte Menge ermöglicht es, aus einer gegebenen Isomorphismus-Invarianten weitere abzuleiten: Jede Invariante eines $K \in H(\Delta,G_1,..,G_k)$ ist auch eine Invariante der G_i, denn es gilt der Satz:

2.5 Satz

Seien $A,B,A',B' \subset X\times X$ Relationen auf $X = \{1,..,n\}$.

Ein Isomorphismus $s:X \longrightarrow X$ von A nach A' und von B nach B' ist dann auch ein Isomorphismus

von A nach A',
" A^T " A'^T,
" $A \cup B$ " $A' \cup B'$,
" $A \cap B$ " $A' \cap B'$,
" $A \circ B$ " $A' \circ B'$ und
von $A(m)B$ nach $A'(m)B'$ $(m=o..n)$.

3. Kapitel. Einiges über Partitionen

3.1 Definition

Ein System $P \subseteq \mathcal{P}(X)$ von Teilmengen einer Menge X heißt Partition von X, falls gilt:

1) $\bigcup_{P \in P} P = X$.

2) $\forall P_1, P_2 \in P : P_1 \cap P_2 \neq \emptyset \Rightarrow P_1 = P_2$.

3) $\emptyset \notin P$.

3.2 Definition

Sei $P = \{P_i\}_{i \in I} \subset \mathcal{P}(X)$ eine indizierte Partition.

Die Funktion $f: X \longrightarrow I$
$x \longmapsto i$, falls $x \in P_i$

heißt Indikatorfunktion von P (bezüglich dieser Indizierung).

Bemerkung Jede Funktion $f: X \longrightarrow I$ erzeugt eine Partition von X auf natürliche Weise:

$\{f^{-1}(i) \mid i \in \operatorname{Im} f\}$,die Äquivalenzklassen bezüglich der Äquivalenzrelation $\{(x,y) \mid f(x)=f(y)\}$.

3.3 Definition

Seien P_1 und P_2 zwei Partitionen von X.

P_1 heißt feiner als P_2 ($P_1 \leq P_2$), falls gilt:

$\forall P \in P_1 \ \exists Q \in P_2 : P \subseteq Q$.

Seien $f_1: X \longrightarrow I$, $f_2: X \longrightarrow J$ zwei Abbildungen.

f_1 heißt feiner als f_2 ($f_1 \leq f_2$), falls gilt:

$\forall x,y \in X: f_1(x)=f_1(y) \Rightarrow f_2(x)=f_2(y)$.

3.4 Satz

Diese beiden Begriffe (von 3.3) sind äquivalent, d.h. für zwei Partitionen P_1, P_2 der Menge X mit den Indizierungen $P_1 = \{P_i\}_{i\in I}$ und $P_2 = \{Q_j\}_{j\in J}$ und die dazugehörigen Indikatorfunktionen f_1, f_2 gilt:

$$P_1 \leq P_2 \iff f_1 \leq f_2 .$$

Beweis

$\Rightarrow$: $x,y \in X$ mit $f_1(x)=f_2(y) =: i \Rightarrow x,y \in P_i = f^{-1}(i) \Rightarrow$

$\Rightarrow \exists Q_j \in P_2 : P_i \subseteq Q_j,\ x,y \in Q_j \Rightarrow f_2(x)=j=f_2(y)$.

$\Leftarrow$: $P_i \in P_2$, $P_i \neq \emptyset$: $\exists\ a \in P_i$, $j:=f_2(a)=:$

Sei $x \in P_i \Rightarrow f_1(x)=i=f_1(a) \Rightarrow f_2(x)=f_2(a)=:j \Rightarrow$

$\Rightarrow x \in Q_j$, also $P_i \subseteq Q_j$.

3.5 Satz und Definition

Seien $f_1: X \longrightarrow I$, $f_2: X \longrightarrow J$ zwei Funktionen, P_1 und P_2 die von ihnen erzeugten Partitionen von X. Folgende Behauptungen sind äquivalent:

1) $f_1 \leq f_2$ und $f_2 \leq f_1$.
2) $\forall\ x,y \in X$: $f_1(x)=f_1(y) \iff f_2(x)=f_2(y)$.
3) $P_1 = P_2$.
4) Es existiert eine Bijektion g: Im $f_1 \longrightarrow$ Im f_2, sodaß $f_2 = g \circ f_1$ ist.

Falls 1) bis 4) erfüllt sind, heißen die beiden Funktionen f_1 und f_2 äquivalent: $f_1 \equiv f_2$.

Beweis

1) $\Leftrightarrow$ 2) nach Definition 3.3 .

2) $\Rightarrow$ 3) Da nach 2) f_1 und f_2 dieselbe Äquivalenzrelation (siehe Bemerkung nach 3.2) definieren, sind auch die Äquivalenzklassen gleich.

3) $\Rightarrow$ 4) Definiere g folgendermaßen: $P_i \in P_1$, $i \in \text{Im } f_1$. $\exists!\ Q_j \in P_2$: $P_i = Q_j$. $g(i) := j$.

4) $\Rightarrow$ 2) $gof_1(x) = gof_1(y) \Leftrightarrow f_1(x) = f_1(y)$, da g Bijektion.

3.6 Bemerkung Da wir es in dieser Arbeit mit endlichen Mengen zu tun haben, können wir zu jeder Funktion $f:X \longrightarrow I$, X endliche Menge, eine äquivalente Funktion f' finden, welche surjektiv ist und ein Intervall von natürlichen Zahlen als Bildmenge hat: $f':X \longrightarrow \{1,..,|\text{Im } f|\}$, indem wir Im f irgendwie anordnen("durchnumerieren"). Eine solche Anordnung ist dann eine Bijektion $g: \text{Im } f \longrightarrow \{1,..,|\text{Im } f|\}$ von 3.5.4 und $f' := gof \equiv f$ die gewünschte Funktion.

3.7 Satz

Seien $P_1 = \{P_i\}_{i \in I}$, $P_2 = \{Q_j\}_{j \in J}$ zwei Partitionen von X mit $P_1 \leq P_2$. Dann gilt: $|P_1| \geq |P_2|$.

Beweis

$g: P_1 \longrightarrow P_2$

$P_i \longmapsto Q_j$ falls $P_i \subset Q_j$ ist eine surjektive Abbildung:

$Q_j \in P_2$, $Q_j \neq \emptyset$: $x \in Q_j, \exists\ P_i \in P_1$, $x \in P_i \Rightarrow$

$\Rightarrow \exists\ j'$ mit $P_i \subset Q_{j'}$, $x \in Q_j \cap Q_{j'} \Rightarrow Q_j = Q_{j'}$ (Def 3.1),

$Q_j = g(P_i)$.

3.8 Folgerung

Für jede Partition P einer endlichen Menge $X = \{x_1,..,x_n\}$ gilt: $1 \leq |P| \leq n$.

Beweis Es gilt natürlich: $\{\{x_1\},..,\{x_n\}\} \leq P \leq \{X\}$, mit 3.7 folgt die Behauptung.

3.9 Folgerung

Seien P_1 und P_2 zwei Partitionen einer endlichen Menge X. $P_1 \leq P_2$ und $|P_1| = |P_2| \Rightarrow P_1 = P_2$.

Beweis g vom Beweis von 3.7 ist surjektiv und, da P_1 und P_2 endlich sind, wegen $|P_1| = |P_2|$ auch bijektiv.

3.1o Satz

Sei $(P_i)_{i\in\mathbb{N}}$ eine Folge von Partitionen der Menge $X = \{x_1,..,x_n\}$, für die gilt: $\forall i\in\mathbb{N}: P_{i+1} \leq P_i$.

Es gibt ein $n_o \in\mathbb{N}: \forall k \geq n_o \; P_{n_o} = P_k$. (Die Folge wird stationär)

Beweis indirekt

Annahme $\forall m \in N \; \exists k > m \; P_m \neq P_k$.

Wir definieren nun eine Folge $(m_i)_{i\in\mathbb{N}}$ folgendermaßen:

$m_1 := \min\{k\in\mathbb{N} \mid k > 1, P_1 \neq P_k\}$ ...

$m_{i+1} := \min\{k\in\mathbb{N} \mid k > m_i, P_{m_i} \neq P_k\}$.

Nach 3.7 gilt $\forall i\in\mathbb{N}: |P_{m_i}| \leq |P_{m_{i+1}}|$ und aus $P_{m_i} \neq P_{m_{i+1}}$ folgt nach 3.9 $|P_{m_i}| \neq |P_{m_{i+1}}|$, also: $|P_{m_{i+1}}| \geq |P_{m_i}| + 1$,

durch Induktion: $|P_{m_{i+1}}| \geq |P_{m_1}| + i \geq 1 + i$ (3.8),

insbesondere liefert: $|P_{m_{n+1}}| \geq n+1$ einen Widerspruch zu 3.8 .

3.11 Folgerung

Voraussetzungen wie 3.1o, dazu: $P_i = P_{i+1} \Rightarrow \forall k\in\mathbb{N}: P_i = P_{i+k}$.

Dann gilt für das n_o von 3.1o: $n_o \leq |X| \leq n$.

Beweis(3.11)

$n_o := \min\{m | \forall\ k > m: P_m = P_k\}$ = (nach Vorauss.)

$= \min\{m |\ P_m = P_{m+1}\ \}$.

$\Rightarrow$ $\forall\ m < m_o$ $P_m \neq P_{m+1}$, und wie im Beweis von 3.1o gilt:

$\forall\ m > m_o$: $|P_{m+1}| \geq |P_m| + 1 \geq |P_1| + m \geq 1 + m$,

also $(m=m_o-1)$: $n \geq |P_{m_o}| \geq m_o$.

3.12 Folgerung

Sei $F := \{f | f: X \longrightarrow \{1,..,m_f\}$ surjektiv $\}$,

$A: F \longrightarrow F$ eine Abbildung, für die gilt:

$\forall\ f \in F$: $A(f) \leq f$ und $\forall\ f,g \in F$: $f \equiv g \Rightarrow A(f) \equiv A(g)$

und $f_1 \in F$ eine weitere Abbildung. Die Funktionenfolge

$(f_i)_{i \in \mathbb{N}}$ sei definiert durch $f_{i+1} := A^i(f_1)$.

Dann gibt es ein $n_o \leq |X|$, sodaß $\forall\ k > n_o$: $f_k \equiv f_{n_o}$.

Die Folge wird stationär.

Beweis

P_i bezeichne die von f_i erzeugte Partition. Nach 3.4

ist $P_{i+1} \leq P_i$ $\forall\ i \in \mathbb{N}$.

$i+1 = i \Rightarrow f_{i+1} \equiv f_i \Rightarrow f_{i+2} = A(f_{i+1}) \equiv A(f_i) \equiv f_i$,

durch Induktion: $\forall\ k > i$: $f_k \equiv f_i \Rightarrow \forall\ k > i$: $P_k = P_i$.

Die Voraussetzungen für 3.1o, 3.11 sind also erfüllt:

$\exists\ n_o \leq |X|$: $\forall\ k > n_o$: $P_k = P_{n_o} \iff f_k \equiv f_{n_o}$ (3.5).

4. Kapitel. Konstruktion einer disjunkten Basis von $H(\Delta,G_1,..,G_k)$.

In diesem Kapitel wird ein iteratives Verfahren zur Konstruktion einer disjunkten Basis von $H(\Delta,G_1,..,G_k)$ (Def. 2.4) angegeben, ausgehend von den Relationen auf einer Paarmenge X×X (oBdA X={1,..,n}).
Zunächst wird ein Iterationsschritt definiert:

4.1 Definition

F sei die Menge der surjektiven Indikatorfunktionen auf X×X mit einem Intervall von natürlichen Zahlen als Bildmenge: $F := \{f \mid f:X^xX \longrightarrow \{1,..,m_f\}\text{surjektiv }\}$.
Für eine Indikatorfunktion $f \in F$ sei f_{st}, $1\leq s,t\leq m_f$ folgende Abbildung:

$$f_{st}: X_xX \longrightarrow \{0,1,..,n\}$$
$$(x,y) \longmapsto |\{z\in X|\ f(x,z)=s \wedge f(y,z)=t\ \}|,$$

das ist die Indikatorfunktion der Partition

$$\left\{[f^{-1}(s)\,]\ (j)\ [f^{-1}(t)\,]^T\right\}_{j=o..n} \setminus \{\emptyset\}$$

g_f bezeichne eine Anordnung von $Im(f,f_{m_fm_f},..,f_{12},f_{11}) \subset \subset \mathbb{N}^{(m_f)^2+1}$. (wie in 3.6)
Die Abbildung

$$A: F \longrightarrow F$$
$$f \mapsto g_f o(f,f_{m_fm_f},..,f_{12},f_{11})$$

heißt dann "Verfahrensschritt A".

A verfeinert die Funktion f bzw. die jeweils erzeugte Partition:

4.2.1 Satz $\forall f \in F: A(f) \leq f$.

4.2.2 Satz

$\forall f,g \in F$: $f \leq g \Rightarrow A(f) \leq A(g)$, d.h. A ist bezüglich "feiner" relationstreu.

Beweis

1) $A(f)(k_1) = A(f)(k_2) =$

$= g_f \circ (f,..,f_{st},..)(k_1) = g_f \circ (f,..,f_{st},..)(k_2)$, g_f bijektiv:

$\Rightarrow \quad f(k_1) = f(k_2)$.

2) $f \leq g$. $A(f)(x,y) = A(f)(a,b) =$

$= g_f \circ (f,..,f_{st},..)(x,y) = g_f \circ (f,..,f_{st},..)(a,b) \quad \Rightarrow$

$\forall\, s,t \quad 1 \leq s,t \leq m_f$: $f_{st}(x,y) = f_{st}(a,b)$.

Zeige zunächst: $L := \{z \in X \mid g(x,z)=p \wedge g(y,z)=q\} =$

$= \dot{\bigcup}_{(s,t)\in J_{p,q}} \{z \in X \mid f(x,z)=s \wedge f(y,z)=t\} =: R \quad (p,q \leq m_g)$

$J_{p,q} := \{(s,t) \mid \exists\, x_1,y_1,x_2,y_2 \in X : g(x_1,y_1)=p \wedge g(x_2,y_2)=q \wedge f(x_1,y_1)=s \wedge f(x_2,y_2)=t\}$

$\subset$: $z \in L$: $g(x,z)=p$, $g(y,z)=q \quad \Rightarrow (f(x,z),f(y,z)) \in J_{p,q} \quad \Rightarrow$

$\Rightarrow \quad z \in R$.

$\supset$: $z \in R$: $f(x,z)=s$, $f(y,z)=t$, $\exists\, x_1,y_1,x_2,y_2 \in X$

$f(x_1,y_1)=s=f(x,z) \Rightarrow (g \leq f) \quad g(x,z)=g(x_1,y_1)=p$

und $f(x_2,y_2)=t=f(y,z) \quad \Rightarrow \quad g(y,z)=g(x_2,y_2)=q$

also $z \in L$.

Es gilt also $\forall\, p,q \quad 1 \leq p,q \leq m_g$: $g_{p,q}(x,y) = \sum_{(s,t)\in J_{pq}} f_{st}(x,y) =$

$= \sum_{(s,t)\in J_{p,q}} f_{st}(a,b) = g_{pq}(a,b)$.

und $f(x,y) = f(a,b) \Rightarrow g(x,y) = g(a,b)$

$A(g)(x,y) = g_g \circ (g,..,g_{pq},..)(x,y) = g_g \circ (g,..,g_{pq},..)(a,b) =$

$A(g)(a,b)$ □

Der Iterationsschritt A braucht höchstens n^2-mal durchgeführt zu werden ($|X|=n$):

4.3 Folgerung

$$\forall f \in F \; \exists n_o \leq n^2 \; \forall k > n_o : A^k(f) \equiv A^{n_o}(f) .$$

Beweis

Mit $f \equiv g \iff f \leq g$ und $g \leq f$ (3.5)

$\Rightarrow$(4.2.2) $A(f) \leq A(g)$ und $A(g) \leq A(f) \iff A(f) \equiv A(g)$

und 4.2.1 sind die Voraussetzungen von 3.12 erfüllt ($n^2=|X\times X|$).

Im nächsten Satz wird gezeigt, daß dieses Iterationsverfahren, ausgehend von einer speziellen Indikatorfunktion $f^{(o)} \in F$, tatsächlich zu einer disjunkten Basis von $H(\Delta,G_1,..,G_k)$ führt.

4.4 Satz

Es seien $X = \{1,..,n\}$ eine Menge,

$G_1,..,G_k$ k Relationen auf $X\times X$,

$H(\Delta,G_1,..,G_k) \subset P(X\times X)$ die Menge der aus Δ(der Diagonale von $X\times X$), $G_1,...,G_k$ mittels der Operationen $\cup$, $\cap$, (j), $^{-}$, ^{T} abgeleiteten Relationen (Def 2.4) und $f^{(o)}$ die folgendermaßen definierte Indikatorfunktion:

$f^{(o)} := g_o \circ (\chi_\Delta, \chi_{G_1}, \chi_{G_1^T}, .., \chi_{G_k}, \chi_{G_k^T})$, wobei χ_A die charakteristische Funktion einer Menge A bezeichnet und g_o eine Anordnung von $\mathrm{Im}(\chi_\Delta,..,\chi_{G_i},\chi_{G_i^T},..)$ wie in 3.6 .

Behauptung:

Die durch die Indikatorfunktion $f := (A)^{n^2}(f^{(o)})$ (n^2-malige Anwendung des Verfahrensschrittes A auf $f^{(o)}$) erzeugte Partition P von $X\times X$: $P = \{f^{-1}(j) \mid j \in \mathrm{Im}\, f\}$ ist eine disjunkte Basis von $H(\Delta,G_1,..,G_k)$.

Beweis

Bezeichnungen: $P^+ := \{S \mid \exists\, I \subset \mathrm{Im}\, f : S = \bigcup_{i\in I} f^{-1}(i)\}$,

$f^{(i)} := (A^i)(f^{(o)})$ ... i-malige Anwendung von A auf $f^{(o)}$,

$m_i := |\mathrm{Im}\, f^{(i)}|$, d.h. $f^{(i)} : X\times X \longrightarrow \{1,..,m_i\}$,

$m := |\mathrm{Im}\, f|$, $M_j^{(i)} := \left[f^{(i)}\right]^{-1}(j)$ für $j \in \mathrm{Im}\, f^{(i)}$,

$M_j := f^{-1}(j)$ für $j \in \mathrm{Im}\, f$,

$M^{(i)} := \left\{M_j^{(i)}\right\}_{j=1,..,m_i}$ ... die von $f^{(i)}$ erzeugte Partition,

$M := \left\{M_j\right\}_{j=1,..,m}$ ... die von f erzeugte Partition.

Zu zeigen ist: $P^+ = H(\Delta, G_1, .., G_k)$

I) $P^+ \subset H(\Delta, G_1, .., G_k)$.

Induktionsbeginn: "$M_t^{(o)} \in H(\ , G_1, .., G_k)$" für $i=1,..,m_o$

Sei $(a,b) \in M_t^{(o)}$, $s_o := \chi_\Delta(a,b)$, $s_{2i-1} := \chi_{G_i}(a,b)$,

$s_{2i} := \chi_{G_i^T}(a,b)$. $(i=1,..,k)$

Dann gilt: +): $M_t^{(o)} = \left(\bigcap_{j=o,..,2k}\right) R_j$, wobei:

$R_j := \begin{cases} \overline{S_j} & \text{falls } s_j = o \\ S_j & s_j = 1 \end{cases}$, $S_o := \Delta$, $S_{2i-1} := G_i$ und

$S_{2i} := G_i^T$. Zeige +):

$(x,y) \in M_t^{(o)} \iff f^{(o)}(x,y) = f^{(o)}(a,b) \iff$

$(\chi_{S_o},..,\chi_{S_{2k}})(x,y) = (s_o,..,s_{2k}) \iff \forall\, i=1,..,2k\ \chi_{S_i}(x,y) = s_i$

$\iff \forall\, i=1,..,2k \quad (x,y) \in R_i$.

$S_i, R_j \in H(\Delta, G_1, .., G_k) \Rightarrow \underline{M_t^{(o)} \in H(\Delta, G_1, .., G_k)}$.

4.5 Hilfssatz

Sei $M_l^{(i+1)} \in M^{(i+1)}$, $(a,b) \in M_l^{(i+1)}$,

$(r, r_{m_i m_i}, .., r_{11}) := (f^{(i)}, f^{(i)}_{m_i m_i}, .., f^{(i)}_{11})(a,b)$.

Dann ist $M_l^{(i+1)} = M_r^{(i)} \cap \left(\bigcap_{s,t=1,..,m_i}\right) M_s^{(i)}(r_{st}) \left[M_t^{(i)}\right]^T$.

Beweis(4.5)

$(x,y) \in M_1^{(i+1)} \iff 1 = f^{(i+1)}(x,y) = f^{(i+1)}(a,b) \iff$

$\iff f^{(i)}(x,y) = r$ und $\forall\, s,t=1,..,m_i$: $f_{st}^{(i)}(x,y) = r_{st}$

$\iff (x,y) \in M_r^{(i)}$ und $\forall\, s,t=1,..,m_i$: $(x,y) \in M_s^{(i)}\ (r_{st})\ \left[M_t^{(i)}\right]^T$.

---

also: $M^{(i)} \subset H(\Delta,G_1,..,G_k) \Rightarrow M^{(i+1)} \subset H(\Delta,G_1,..,G_k)$,

die $M_j^{(i+1)}$ lassen sich als Durchschnitt, Transposition und (m)-Produkt von $M_j^{(i)}$ anschreiben.

Induktionsschluß: $\forall\, i \in \mathbb{N}$: $M^{(i)} \subset H(\Delta,G_1,..,G_k)$.

Insbesondere ist $M^{(n^2)} = P \subset H(\Delta,G_1,..,G_k)$ und somit auch

$P^+ \subset H(\Delta,G_1,..,G_k)$.

Hiemit ist I) gezeigt.

II) $H(\Delta,G_1,..,G_k) \subset P^+$

Dazu ist nach Definition 2.4 zu zeigen:

1. $\Delta,G_1,..,G_k \in P^+$
2. $A,B \in P^+ \Rightarrow$
   - 2.1 $A \cup B \in P^+$
   - 2.2 $A \cap B \in P^+$
   - 2.3 $\bar{A} \in P^+$
   - 2.4 $A^T \in P^+$
   - 2.5 $A(m)B \in P^+$ für $m=o,..,n = |X|$.

1. $G_o := \Delta$.

Es gilt: $\forall\, i=o,..,k \quad G_i = \bigcup_{j \in f(G_i)} M_j$.

$\underline{(x,y) \in G_i} \Rightarrow j_o := f(x,y) \in f(G_i) \Rightarrow \underline{(x,y) \in M_{j_o}}$

$j \in f(G_i),\ \underline{(x,y) \in M_j} \Rightarrow \exists\, (a,b) \in G_i,\ f(a,b)=j=f(x,y)$

(nach Satz 4.2.1 ist $f = f^{(n^2)} \leq f^{(o)}$) $\Rightarrow$

$\Rightarrow f^{(o)}(a,b) = f^{(o)}(x,y)$ (Indizierung bijektiv) $\Rightarrow$

$\Rightarrow \forall\, i=o,..,k \quad \chi_{G_i}(x,y) = \chi_{G_i}(a,b) = 1 \Rightarrow \underline{(x,y) \in G_i}$

Nach Definition von P^+ ist also $G_i \in P^+$.(i=o,..,k)

2. $A,B \in P^+ \Rightarrow \exists\ I_A, I_B \subset \{1,..,m\}\quad A = \bigcup_{j\in I_A} M_j$,

$B = \bigcup_{j\in I_B} M_j$.

2.1 $A \cup B = \bigcup_{j\in I_A \cup I_B} M_j \in P^+$.

2.2 $A \cap B = \bigcup_{j\in I_A \cap I_B} M_j \in P^+$.

2.3 $\bar{A} = X\times X \setminus A = \bigcup_{j \in \{1,..,m\}\setminus I_A} M_j$. ($X\times X = \bigcup_{j=1..m} M_j$, da P Partition)

4.6 Hilfssatz

$\forall\ j=1,..,m_i\quad \exists\ j' : \left[M_j^{(i)}\right]^T = M_{j'}^{(i)}$.

Beweis

g_i seien die Indizierungen zur Berechnung von $f^{(i)}$.

($i\neq o$: $f^{(i)} = A(f^{(i-1)})$, siehe 3.6, 4.1).

Induktionsbeginn: $i=0$, $M_j^{(o)} \in M^{(o)}$:

$(s_o,s_1,..,s_{2k}) := g_o^{-1}(j)$ wie in Teil I) dieses Beweises.

Zeige: $j' = g_o(s_o,s_2,s_1,s_4,s_3,..,s_{2k},s_{2k-1})$.

$(x,y) \in \left[M_j^{(o)}\right]^T \iff (y,x) \in M_j^{(o)} \iff (\chi_\Delta,\chi_{G_1},..,\chi_{G_k}^T)(y,x) =$

$= (s_o,s_1,..,s_{2k}) \iff (\chi_\Delta,\chi_{G_1},..,\chi_{G_k}^T)(x,y) = (s_o,s_2,s_1,..,s_{2k-1})$

$\iff (x,y) \in M_{j'}^{(o)}$.

Induktionsannahme: $\forall\ j=1,..,m_i\quad \exists\ j' : \left[M_j^{(i)}\right]^T = M_{j'}^{(i)}$.

$M_j^{(i+1)} \in M^{(i+1)}$: $(r,r_{m_im_i},..,r_{st},..,r_{1,2},r_{1,1}) := (g_{i+1})^{-1}(j)$

$M_j^{(i+1)} = M_r^{(i)} \cap \bigcap_{s,t=1,..,m_i} M_s^{(i)}(r_{st}) \left[M_t^{(i)}\right]^T \cdot \left[M_j^{(i+1)}\right]^T =$

(Satz 2.2.2 und 2.2.3)

$= \left[M_r^{(i)}\right]^T \cap \bigcap_{s,t=1,..,m_i} M_t^{(i)}(r_{st}) \left[M_t^{(i)}\right]^T =$ (s,t vertauschen)

$= M_{r'}^{(i)} \cap \bigcap_{t,s=1,..,m_i} M_s^{(i)}(r_{ts}) \left[M_t^{(i)}\right]^T$. Setze:

$j' = g_{i+1}(r',r_{m_im_i},..,r_{ts},..,r_{2,1},r_{1,1})$.

Induktionsschluß: $\forall i\in\mathbb{N}\ \forall\ j=1,..,m_i\ \exists\ j' : M_{j'}^{(i)} = \left[M_j^{(i)}\right]^T$.

2.4 Insbesondere gilt für $i=n^2$ $(P = M^{(n^2)})$:

$\forall\ j=1,..,m\ \exists\ j' : M_j^T = M_{j'}$.

$A^T = \bigcup_{j\in I_A} M_j^T = \bigcup_{j\in I_A} M_{j'} = (I'_A := \{j' \mid j\in I_A\}) =$

$= \bigcup_{j\in I'_A} M_j \in P^+$.

2.5

$A(j)B = \left[\bigcup_{i\in I_A} M_i\right](j)\left[\bigcup_{i\in I_B} M_i\right] =$ (Satz 2.3) $=$

$= \underbrace{\qquad\qquad}_{(j_{st}) \in \{o,..,j\}^{I_A\times I_B},\ \sum_{(s,t)\in I_A\times I_B} j_{st} = j} \underbrace{\qquad\qquad}_{(s,t)\in I_A\times I_B} M_s(j_{st})M_t$.

Es genügt daher zu zeigen:

$\forall\ s,t=1,..,m,\ \forall\ j=o,..,n : M_s(j)M_t \in P^+$.

Wegen Hilfssatz 4.6 ist dies äquivalent zu: $\therefore M_s(j)M_t^T \in P^+$.

Zeige: $M_s(j)\,M_t^T = \bigcup_{i\in I} M_i$, $I := f(M_s(j)M_t^T)$.

a)
$M_s(j)M_t^T = \emptyset = \bigcup_{i\in\emptyset} M_i$

b) $M_s(j)M_t^T \neq \emptyset$.

$M_s(j)M_t^T \subset \bigcup_{i\in I} M_i$: $(x,y) \in M_s(j)M_t^T$: $f(x,y)=:i \in I \Rightarrow (x,y)\in M_i$.

$M_s(j)M_t^T \supset \bigcup_{i\in I} M_i$: $i\in I$, $(x,y) \in M_i$, $f(x,y) = i \Rightarrow$

$\Rightarrow \exists\ (a,b) \in M_s(j)M_t^T$: $f(a,b)=i$, $f_{st}(a,b) = j$ (siehe 4.1).

$f^{(n^2+1)} \equiv f^{(n^2)} \equiv (f, f_{mm}, .., f_{st}, .., f_{1,1})$, $f(x,y)=i=f(a,b) \Rightarrow$

$\Rightarrow f_{st}(x,y) = f_{st}(a,b) = j \Rightarrow (x,y) \in M_s(j)M_t^T$.

Ende des Beweises 4.4 .

4.7 Folgerung

Voraussetzungen wie in 4.4, f definiert wie in 4.4.

$\forall\, s,t \in \{1,..,m\}: f \leq f_{st}$

Beweis

Nach Folgerung 4.3 ist $f \equiv \Lambda(f) = g \circ (f,..,f_{st},..) \equiv$
$\equiv (f,..,f_{st},..)$, also $f(x,y) = f(a,b) \Rightarrow f_{st}(x,y) = f_{st}(a,b)$.

4.8 Folgerung

Voraussetzungen wie 4.4, f definiert wie in 4.4 .

Abbildung $f^*: X \longrightarrow f(\Delta)$
$x \longmapsto f(x,x)$.

Dann gilt: $\forall\, K \in H(\Delta,G_1,..,G_k): f^* \leq N_K$.

Beweis Sei $f^*(x) = f^*(y)$.

$K = \bigcup_{i\in I_K} M_i$. $N_{M_i}(x) = \underset{(x,z)\in M_i}{z\in X} = f_{ii}(x,x) =$

$\left(\begin{array}{l} f(x,x) = f^*(x) = f^*(y) = f(v,v) \quad \Rightarrow (4.7) \\ \forall\, i \in 1..m : f_{ii}(x,x) = f_{ii}(v,y) \end{array} \right)$

$= f_{ii}(y,v) = N_{M_i}(y)$, also $N_K(x) = \sum_{i\in I_K} N_{M_i}(x) = \sum_{i\in I_K} N_{M_i}(y) =$

$= N_K(y)$.

4.9 Satz

Seien $G_1,..,G_k,G_1',..,G_k' \subset X\times X$ je k Relationen auf $X = \{1..n\}$.

f und $\{M_j\}$, f' und $\{M_j'\}$ seien die jeweiligen parallel berechneten Ergebnisse von 4.4, d.h. die Bedingungen:

$\forall\, i=1,..,k: |G_i| = |G_i'|$; $\forall\, i=o,..,n^2: |M_j^{(i)}| = |M_j'^{(i)}|$

$\forall\, i=o,..,n^2: g_i = g_i'$, d.h. insbesondere $Im(f^{(i)},..,f_{st}^{(i)},..) = Im(f'^{(i)},..,f_{st}'^{(i)},..)$.

seien erfüllt.

Dann ist für jeden Isomorphismus von $(G_1,..,G_k)$ nach $(G_1',..,G_k')$ (d.h. $\forall\, i=1,..,k$: s ist Isom. von G_i nach G_i')

und $\forall\, (x,y) \in X\times X: f(x,v) = f'(s(x),s(y))$.

Beweis (4.9)

1) $\forall$ i $\in\{1..m\}$ "M_i' kann mit Hilfe derselben Formel (∩, (m)) aus $\Delta, G_1', .., G_k'$ abgeleitet werden wie M_i aus $\Delta, G_1, .., G_k$ "

Induktionsbeginn:

Sei $M_t^{(o)} \in M^{(o)}$: die Formel im Beweis von 4.4, Teil I), ist durch $(s_o, .., s_{2k})$ festgelegt. Nach Voraussetzung ist $g_o^{-1}(t) = g_o'^{-1}(t) = (s_o, .., s_{2k})$.

Induktionsannahme:

Sei für $j=1,..,m_i$ $M_j'^{(i)}$ mit derselben Formel ableitbar wie $M_j^{(i)}$. Setze diese Formeln ein in die jeweilige Formel von Hilfssatz 4.5, die durch $(r, .., r_{st}, ..)$ festgelegt ist. Nach Voraussetzung ist auch $g_{i+1}^{-1}(1) = g_{i+1}'^{-1}(1) = (r, .., r_{st}, ..)$, also ist auch für $1=1,..,m_{i+1}$ $M_1'^{(i+1)}$ mit derselben Formel ableitbar wie $M_1^{(i+1)}$.

Induktionsschluß: Behauptung gilt für alle natürlichen Zahlen i, insbesondere für $i=n^2$.

2) Sei s ein gemeinsamer Isomorphismus von $(G_1, .., G_k)$ nach $(G_1', .., G_k')$. Nach Satz 2.5 ist s dann auch ein Isomorphismus von M_j nach M_j' für $j=1..m$: $\forall\, x,y \in X$: $(x,y) \in M_j \iff (s(x), s(y)) \in M_j'$

$\Rightarrow$ $\forall\, x,y \in X$: $f(x,y) = f'(s(x), s(y))$

4.1o Hilfssatz

Voraussetzungen wie in 4.9, dazu: $x_o, x_1, x_2, y_o \in X$ mit $f(x_o, x_o) = f'(y_o, y_o)$, $t \in \{1, .., m\}$.

(i) $|\{x | f(x_o, x) = t\}| = |\{y | f'(y_o, y) = t\}|$

(ii) Sei zusätzlich $a \in X$ mit $f(x_o, x) = t \Rightarrow x = a$. ($\Leftrightarrow f_{tt}(x_o, x_o) = 1$)
Dann gibt es genau ein $b \in X$ mit $f'(y_o, b) = t$.

(iii) Vorauss. wie (ii): $f(x_o, x_1) = f'(y_o, x_2) \Rightarrow f(x_1, a) = f'(x_2, b)$

Beweis (4.1o)

(i) $|\{x|f(x_o,x)=t\}| = f_{tt}(x_o,x_o)$

$|\{y|f'(y_o,y)=t\}| = f'_{tt}(y_o,v_o)$

$f(x_o,x_o)=f'(y_o,y_o) \Rightarrow$ (Folgerung 4.7) $\Rightarrow f_{tt}(x_o,x_o)=f'_{tt}(y_o,y_o)$.

(ii) $f_{tt}(x_o,x_o) = 1 = f'_{tt}(y_o,y_o) = |\{y|f'(y_o,y)=t\}|$,

d.h. die Menge $\{y|f'(y_o,v)=t\}$ hat genau ein Element, das wir mit b bezeichnen: $f'(y_o,v)=t \Rightarrow y=b$.

(iii) $i := f(x_1,a)$

$f_{ti}(x_o,x_1) = |\{z\in X|f(x_o,z)=t \wedge f(x_1,z)=i\}| = |\{a\}| = 1$
$(\Rightarrow z=a)$

$=$ (Folgerung 4.7) $= f'_{ti}(y_o,x_2) = |\{z|f'(y_o,z)=t \wedge f'(x_2,z)=i\}| =$
$(ii) \Rightarrow z=b$

$= |\{b\}|$, also $f'(x_2,b) = i = f(x_1,a)$.

4.11 Satz

Seien $G_1,..,G_k,G'_1,..,G'_k \subset X\times X$ je k Relationen auf $X=\{1..n\}$
f,f' wie in Satz 4.9 definiert , $x_o \in X$ mit:
$\forall\, x,y \in X$: $f(x_o,x) = f(x_o,y) \Rightarrow x = y$.
Dann ist

$$S = \left\{ p_{y_o} \;\middle|\; \begin{array}{l} p_{y_o}: X \longrightarrow X \quad \exists\, y_o \in X: f(x_o,x_o)=f'(y_o,y_o) \\ x \longrightarrow y, \text{falls } f(x_o,x)=f'(y_o,y) \end{array} \right\}$$

die Gruppe aller gemeinsamen Isomorphismen von $(G_1,..,G_k)$ nach $(G'_1,..,G'_k)$.

Beweis

Die Definition von S ist sinnvoll:

p_{y_o} wohldefiniert: Nach Hilfssatz 4.1o gibt es genau ein b (=:y) zu x, sodaß $f(x_o,x) = f'(y_o,b)$ ist.

p_{y_o} bijektiv: $p_{y_o}(x_1)=p_{y_o}(x_2)=y \Rightarrow f(x_o,x_1)=f'(y_o,y)=f(x_o,x_2)$

$\Rightarrow$ (Voraussetzung) $x_1 = x_2$.

$\supseteq$: Sei s ein Isomorphismus von $(G_1,..,G_k)$ nach $(G'_1,..,G'_k)$.

Nach Satz 4.9 gilt dann:

$\forall\, x,y \in X$: $f(x,y) = f'(s(x),s(y))$.

$y_o := s(x_o)$: $f(x_o,x_o) = f'(y_o,y_o)$ und

$\forall\, x \in X$: $f(x_o,x) = f'(y_o,y)$, also $s = p_{y_o} \in S$.

$\subseteq$: Sei $y_o \in X$, $p_{y_o} = s \in S$. Nach Hilfssatz 4.1o gilt

mit der Umbenennung $x_1, x_2, a, b \to a, s(x), y, s(y)$:

$\forall\, x,y \in X$: $f(x,y) = f'(s(x),s(y)) \Rightarrow$

$\forall\, j=1,..,m$: $(x,y) \in M_j \iff (s(x),s(y)) \in M'_j \Rightarrow$

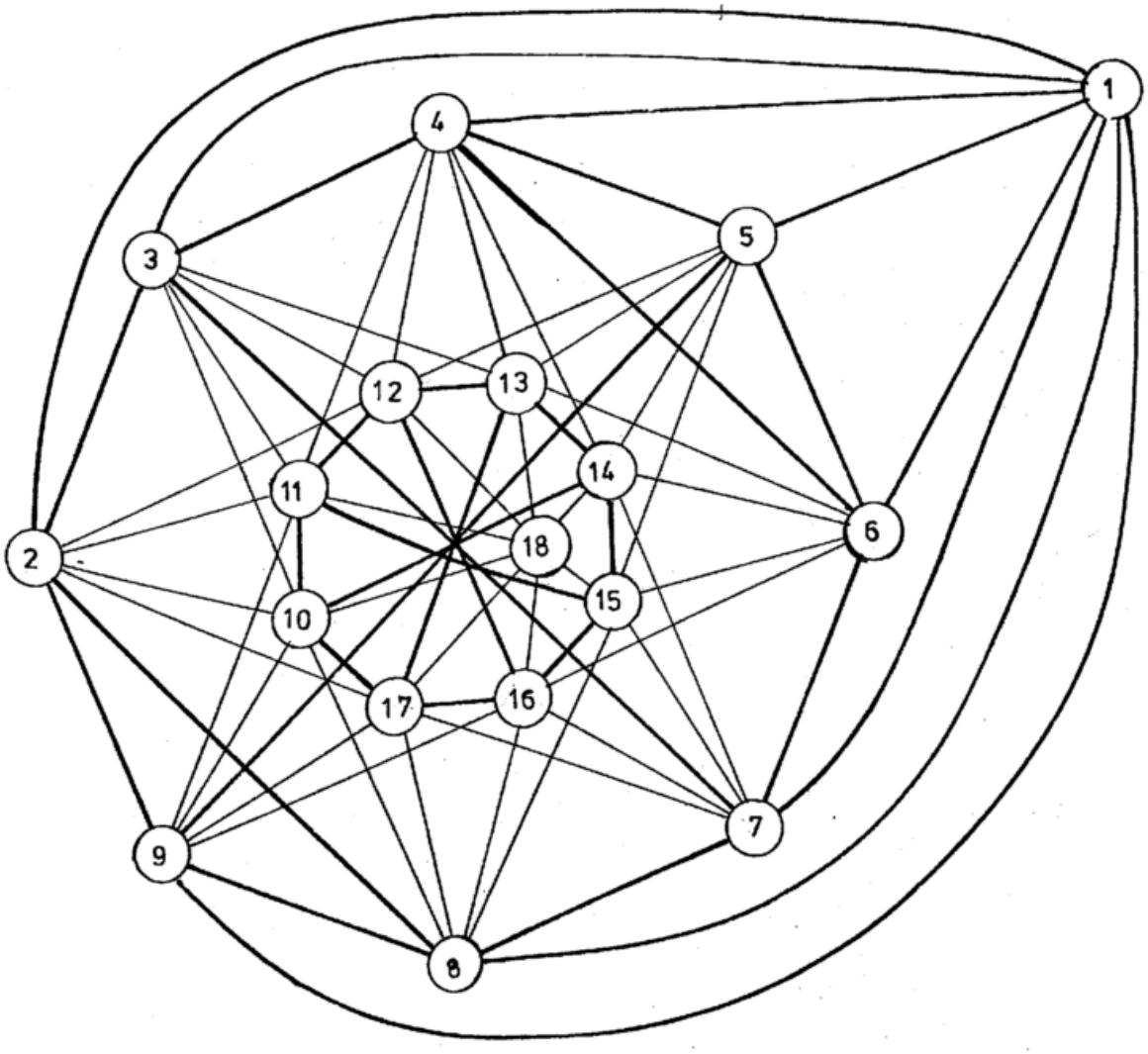
$\Rightarrow (\forall\, i=1,..,k \;\; \exists\, J_i \subset \{1,..,m\} \;\; G_i = \bigcup_{j \in J_i} M_j,\; G'_i = \bigcup_{j \in J_i} M'_j)$

$\forall\, i=1,..,k$: $(x,y) \in G_i \iff (s(x),s(y)) \in G'_i$,

s ist ein Isomorphismus von $(G_1,..,G_k)$ nach $(G'_1,..,G'_k)$.

4.12 Beispiel

einfacher, symmetrischer Graph der Ordnung 18 :
($k=1$, $G_1=G_1^T \subset X\times X$, $X = \{1,..,18\}$)



Berechnung von $f^{(o)}$: $(\chi_\Delta, \chi_{G_1}, \chi_{G_1^T})(x,y) = \begin{cases} (1,o,o) & \text{für } x=y \\ (o,1,1) & x\neq y, (x,y)\in G_1 \\ (o,o,o) & \text{sonst} \end{cases}$

Anordnung von $\mathrm{Im}(\chi_\Delta, \chi_{G_1}, \chi_{G_1^T})$:

$g_o \begin{pmatrix} (1,o,o) \\ (o,1,1) \\ (o,o,o) \end{pmatrix} = \begin{matrix} 3 \\ 2, \\ 1 \end{matrix}$ somit $f^{(o)}(x,y) = \begin{cases} 3 & \text{für } x=y \\ 2 & x\neq y,\ (x,y)\in G_1 \\ 1 & \text{sonst} \end{cases}$

$f_{st}^{(o)}$: z.B. $f_{2,1}^{(o)}((1,2)) = |\{z\in X \mid f(x,z)=2 \wedge f(y,z)=1\}| =$
$= |\{4,5,6,7\}| = 4$ etc.

Tabelle der Funktionswerte von f, nach 4.1 und 4.4 berechnet (lexikographische Anordnung von $Im(f^{(i)}_{...,f^{(i)}_{st},..})$ beginnt allerdings bei o anstatt bei 1) :

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1:	84	57	55	57	58	57	55	57	58	18	16	16	18	18	16	16	18	1
2:	62	89	72	33	14	3	37	77	81	80	75	51	13	4	10	40	65	42
3:	59	67	85	67	30	26	44	26	30	49	68	68	49	23	6	6	23	24
4:	62	33	72	89	81	77	37	3	14	13	51	75	80	65	40	10	4	42
5:	63	12	43	62	88	82	43	12	47	9	39	53	78	78	53	39	9	36
6:	62	3	37	77	81	89	72	33	14	4	10	40	65	80	75	51	13	42
7:	59	26	44	26	30	67	85	67	30	23	6	6	23	49	68	68	49	24
8:	62	77	37	3	14	33	72	89	81	65	40	10	4	13	51	75	80	42
9:	63	82	43	12	47	12	43	82	88	78	53	39	9	9	39	53	78	36
10:	41	79	52	11	8	2	34	64	76	87	74	38	32	46	31	35	73	61
11:	25	71	69	48	22	5	7	28	50	70	86	66	27	20	45	21	29	60
12:	25	48	69	71	50	28	7	5	22	27	66	86	70	29	21	45	20	60
13:	41	11	52	79	76	64	34	2	8	32	38	74	87	73	35	31	46	61
14:	41	2	34	64	76	79	52	11	8	46	31	35	73	87	74	38	32	61
15:	25	5	7	28	50	71	69	48	22	20	45	21	29	70	86	66	27	60
16:	25	28	7	5	22	48	69	71	50	29	21	45	20	27	66	86	70	60
17:	41	64	34	2	8	11	52	79	76	73	35	31	46	32	38	74	87	61
18:	0	19	15	19	17	19	15	19	17	56	54	54	56	56	54	54	56	83

in Zeile 2 stehen lauter verschiedene Werte:

$x_o = 2$ erfüllt die Voraussetzungen von 4.11 ($G_1 = G_1'$).

Für y_o mit $f(y_o,y_o) = 89 = f(x_o,x_o)$ kann jetzt 2, 4, 6 oder 8 genommen werden. Durch die Wahl von y_o ist ein Automorphismus p_{y_o} bestimmt: z,B. $y_o := 4$; $p_4(2) = 4$

$f(2,4) = f(4,2) = 33$: $p_4(4) = 2$

$f(2,1) = f(4,1) = 62$: $p_4(1) = 1$

$f(2,3) = f(4,3) = 72$: $p_4(3) = 3$

$f(2,5) = f(4,9) = 14$: $p_4(5) = 9$

$f(2,6) = f(4,8) = 3$: $p_4(6) = 8$

etc.

Die Automorphismengruppe dieses Graphen besteht also aus den Abbildungen p_2, p_4, p_6 und p_8 (Diese Bezeichnung ist abhängig von der Wahl von $x_o = 2$; dieselben Abbildungen ergeben sich auch bei der Wahl von $x_o \in f^{-1}(87)$ oder $x_o \in f^{-1}(86)$, wobei ebenfalls die Voraussetzungen von 4.11 erfüllt sind.

Bemerkung

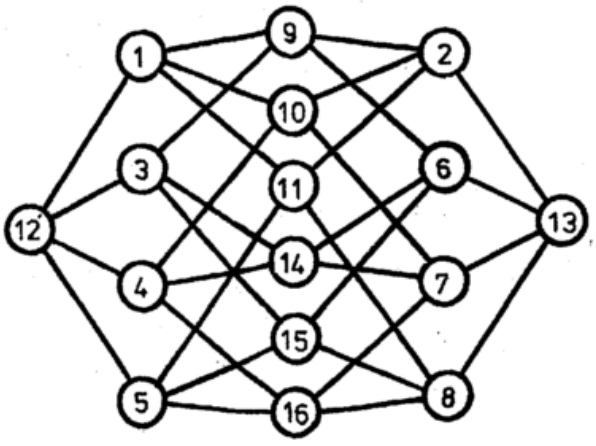
Sei f* wie in Folgerung 4.8 definiert. Bezeichne
$h: X \longrightarrow \{1,..,l\}$ die Indikatorfunktion der Partition von X, die durch die Knoten-Quotientengraphen nach Corneil ([1]) erzeugt wird. Dann gilt auch $f^* \leq h$:

In den Begriffen von 4.1 und 4.4 läßt sich das Verfahren("B") von Corneil zur Berechnung der Knotenquotientengraphen folgendermaßen beschreiben:
die Abbildungen f_{st} von 4.1 sind dort folgendermaßen definiert: $f'_{st}(x,y) := |\{z \in X | f(x,z)=s \wedge f^{(o)}(y,z)=t\}|$,

$f'_{st} = \prod_{r \in f(M_t^{(o)})} f_{sr}$. Die Anordnung g von $Im(f',..,f'_{st},..)$ erfolgt außerdem nicht für ganz X×X auf einmal, sondern zeilenweise. g ist dann nicht mehr notwendig bijektiv.
Es gilt $A(f) \leq B(f)$ und auch $A^{n^2}(f) \leq B^{n^2}(f)$.
(durch Induktion: $A^i(f) \leq B^i(f) \Rightarrow$(4.2.2)$\Rightarrow A^{i+1}(f) \Rightarrow A(B^i(f)) \leq B^{i+1}(f)$).
Die Berechnung der Knotenquotientengraphen führt also nicht aus $H(\Delta,G_1,..,G_k)$ hinaus. Es gilt jedoch nicht allgemein $h \leq B^n(f^{(o)})$, wie das Beispiel 4.12 zeigt: mit den Knoten-Quotientengraphen können Knoten 1 und 18 nicht unterschieden werden, während $f(1,1) = 84 \neq f(18,18) = 83$.
Auch beim folgenden Beispiel können mit den KnotenQuotienten-Graphen die Knoten 12 und 13 nicht von den Knoten 9,1o,11, 14,15 und 16 unterschieden werden.

4.13 Beispiel

einfacher symmetrischer Graph der Ordnung 16 aus [12].



Hier sind die Voraussetz. von 4.11 für kein x∈X erfüllt; durch Ausprobieren kann man jedoch zeigen, daß die von f erzeugte Kantenpartition die Automorphismenpartition ist.

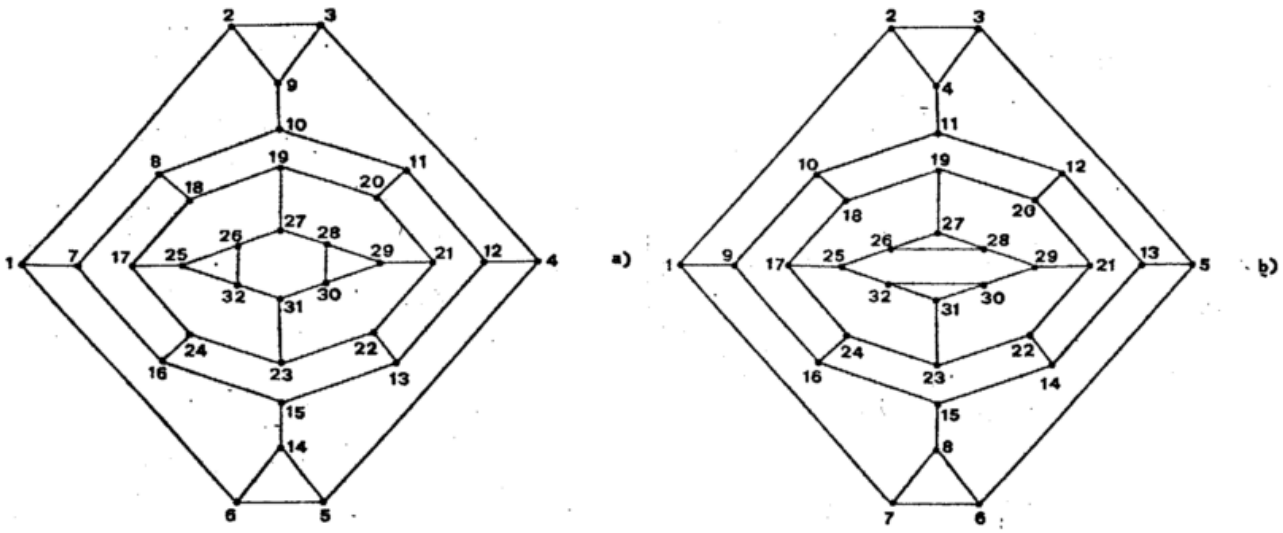
Tabelle von f, für den Graphen 4.13 nach 4.4 berechnet:
(Anordnung beginnt ebenfalls bei o anstatt bei 1)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1:	18	11	10	10	10	6	6	6	14	14	14	13	4	3	3	3
2:	11	18	6	6	6	10	10	10	14	14	14	4	13	3	3	3
3:	10	6	18	10	10	11	6	6	14	3	3	13	4	14	14	3
4:	10	6	10	18	10	6	11	6	3	14	3	13	4	14	3	14
5:	10	6	10	10	18	6	6	11	3	3	14	13	4	3	14	14
6:	6	10	11	6	6	18	10	10	14	3	3	4	13	14	14	3
7:	6	10	6	11	6	10	18	10	3	14	3	4	13	14	3	14
8:	6	10	6	6	11	10	10	18	3	3	14	4	13	3	14	14
9:	15	15	15	2	2	15	2	2	17	9	9	8	8	9	9	1
10:	15	15	2	15	2	2	15	2	9	17	9	8	8	9	1	9
11:	15	15	2	2	15	2	2	15	9	9	17	8	8	1	9	9
12:	12	5	12	12	12	5	5	5	7	7	7	16	0	7	7	7
13:	5	12	5	5	5	12	12	12	7	7	7	0	16	7	7	7
14:	2	2	15	15	2	15	15	2	9	9	1	8	8	17	9	9
15:	2	2	15	2	15	15	2	15	9	1	9	8	8	9	17	9
16:	2	2	2	15	15	2	15	15	1	9	9	8	8	9	9	17

4.14 Beispiel

Zwei nicht isomorphe Graphen der Ordnung 32, symmetrisch, nach Grace [11]. Hier kann nicht nur die Nicht-Isomorphie der beiden Graphen durch die Berechnung von f, f' (wie in 4.9 beschrieben) gezeigt werden, für jeden einzeln erhält man durch die Berechnung von f nach 4.4 die Automorphismengruppe nach Satz 4.11:

Für den Graphen a) wird sie z.B. durch $x_o = 2$ folgendermaßen angegeben: $\{p_2, p_3, p_5, p_6, p_{26}, p_{28}, p_{30}, p_{32}\}$;
für b) durch $x_o = 2$: $\{p_2, p_3, p_6, p_7, p_{26}, p_{28}, p_{30}, p_{32}\}$.
Die beiden Funktionen f_a und f_b unterscheiden sich auch noch, ohne daß sie parallel berechnet worden wären:
$f_a(2,2)=143=f_b(2,2)$, aber $f_a(2,27) = 67$, während dieser Wert in der zweiten Zeile von f_b nicht vorkommt.



```
**********
N=32  SYMETRISCH  EINDEUTIG
1|2,4,7*2|3,6*3|4,9*4|5,12*5|6,14*6|7|4*7|8,16*8|10,18*9|11*9|12*10|11|12,20*
2|1|3*3|1|5,22*4|1|5*5|1*6|1*6|2*6|4*7|1|8,24,25*8|1*8|10*8|9|20,27*9|2*9|2|1|22,29*
2|2|3*2|3|2|4*3|1*3|4*3|5|2*6|2|7,32*6|2*8|2*8|1,30*9|2*9|3*9|3*9|3|1*3|1|32*
2|7|2*8*
```

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1:	141	132	117	61	117	132	130	112	116	60	37	38	37	116	60	112	36	50	21	7	5	7	21	50	18	17	6	14	15	16	6	17
2:	134	143	137	123	102	120	110	101	136	121	97	98	88	96	90	95	73	91	74	87	70	71	68	86	63	81	67	80	82	78	65	79
3:	123	137	143	134	120	102	98	97	136	121	101	110	95	96	90	88	70	87	74	91	73	86	68	71	82	80	67	81	83	79	65	78
4:	61	117	132	141	132	117	30	37	116	60	112	130	112	116	60	37	5	7	21	50	36	50	21	7	15	16	6	17	18	17	6	16
5:	123	102	120	134	143	137	98	88	96	90	95	110	101	136	121	97	70	71	68	86	73	91	74	87	82	78	65	79	83	81	67	80
6:	134	120	102	123	137	143	110	95	96	90	88	98	97	136	121	101	73	86	68	71	70	87	74	91	83	79	65	78	82	80	67	81
7:	128	114	54	32	54	114	130	126	63	109	46	29	46	63	109	126	56	106	44	30	1	30	44	106	39	12	20	10	8	10	20	12
8:	108	62	53	31	24	52	125	130	113	127	107	47	28	41	45	105	103	124	104	57	27	2	26	55	51	40	43	22	11	9	0	23
9:	122	135	135	122	99	99	100	118	142	133	118	100	92	93	75	92	85	94	89	64	85	72	66	72	84	77	69	77	84	76	64	76
10:	59	115	115	59	42	42	111	129	131	140	129	111	49	19	34	49	48	110	58	110	48	35	4	35	25	14	33	14	25	13	3	13
11:	31	53	62	108	52	24	47	107	113	127	138	125	105	41	45	28	27	57	104	124	103	55	26	2	11	22	43	40	51	23	0	9
12:	32	54	114	128	114	54	29	46	63	109	126	130	126	63	109	46	1	30	44	106	56	106	44	30	8	10	20	12	39	12	20	10
13:	31	24	52	108	62	53	47	28	41	45	105	125	138	113	127	107	27	2	26	55	103	124	104	57	11	9	0	23	51	40	43	22
14:	122	99	99	122	135	135	100	92	93	75	92	100	118	142	133	118	85	72	66	72	85	94	89	94	84	76	64	76	84	77	69	77
15:	59	42	42	59	115	115	111	49	19	34	49	111	129	131	140	129	48	35	4	35	48	110	58	110	25	13	3	13	25	14	33	14
16:	108	52	24	31	53	62	125	105	41	45	28	47	107	113	127	138	103	55	26	2	27	57	104	124	51	23	0	9	11	22	43	40
17:	33	14	13	3	13	14	58	110	25	48	35	4	35	25	48	110	140	129	111	49	34	49	111	129	131	115	59	42	19	42	59	115
18:	43	40	23	0	9	22	104	124	51	103	55	26	2	11	27	57	127	138	125	105	45	28	47	107	113	62	108	62	41	24	31	53
19:	26	12	12	20	10	10	44	106	39	56	106	44	30	8	1	30	109	126	130	126	109	46	29	46	63	114	128	114	63	54	32	54
20:	0	23	40	43	22	9	26	55	51	103	124	104	57	11	27	2	45	105	125	138	127	107	47	28	41	52	108	62	113	53	31	24
21:	3	13	14	33	14	13	4	35	25	48	110	58	110	25	48	35	34	49	111	129	140	129	111	49	19	42	59	115	131	115	59	42
22:	0	9	22	43	40	23	26	2	11	27	57	104	124	51	103	55	45	28	47	107	127	138	125	105	41	24	31	53	113	62	108	52
23:	20	10	10	20	12	12	44	30	8	1	30	44	106	39	56	106	109	46	29	46	109	126	130	126	63	54	32	54	63	114	128	114
24:	43	22	9	0	23	40	104	57	11	27	2	26	55	51	103	124	127	107	47	28	45	105	125	138	113	53	31	24	41	52	108	62
25:	69	77	76	64	76	77	89	84	84	85	72	66	72	84	85	94	133	118	100	92	75	92	100	118	142	135	122	69	93	99	122	135
26:	67	81	79	65	78	80	74	91	83	73	86	68	71	82	70	87	121	101	119	95	90	88	68	97	136	143	134	120	96	102	123	137
27:	6	17	17	6	16	16	21	50	18	36	50	21	7	15	5	7	60	112	130	112	60	37	38	37	116	132	141	122	116	117	61	117
28:	65	79	81	67	80	78	68	86	83	73	91	74	87	82	70	71	90	95	119	101	121	97	98	88	96	120	134	143	136	137	123	102
29:	64	76	77	69	77	76	66	79	84	85	94	89	94	84	85	72	75	92	100	118	133	118	100	92	93	99	122	135	142	135	122	99
30:	65	78	80	67	81	79	68	71	82	70	87	74	91	83	73	86	90	88	98	97	121	101	119	95	96	102	123	137	136	143	134	120
31:	6	16	16	6	17	17	21	7	15	5	7	21	50	18	36	50	60	37	38	37	60	112	130	112	116	117	61	117	116	132	141	122
32:	67	80	78	65	79	81	74	87	82	70	71	68	86	83	73	91	121	97	98	88	90	95	119	101	136	137	123	102	96	120	134	143

4.14.a)
Tabelle der Funktionswerte von f_a.

```
**********
N=32  SYMMETRISCH  EINGEFUEGT:
1:2.7.8*2:3.4*3:4.5**1:*5:6.1*6*4:7.8*7:8*8:1*6*9:10.16*10:11.18*11:12*12:13.20*
13:14*14:15.22*15:16*16:24*17:18.24.25*19:20.27*20:21*21:22.29*22:23*
23:24.31*25:26.32*26:27.28*27:28*28:29*29:30*30:31.32*31:32*
```

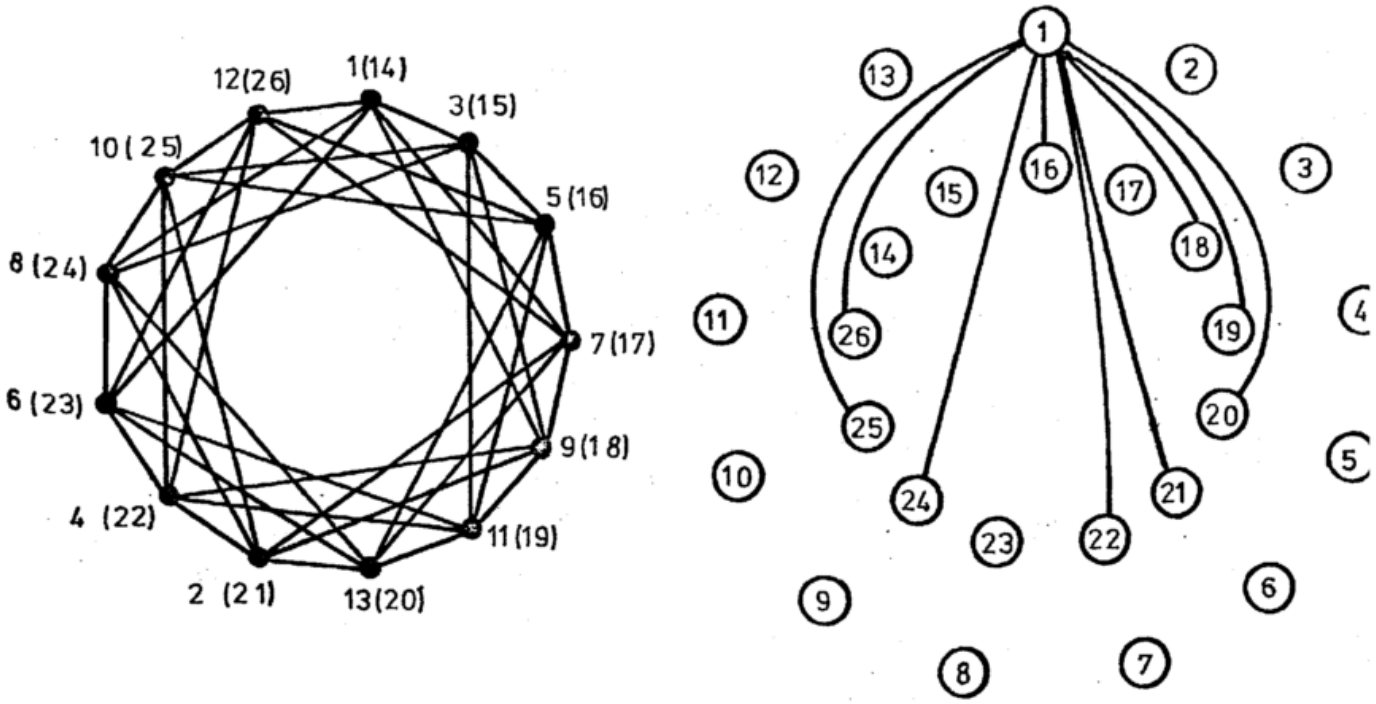
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1:	141	132	117	114	81	117	132	114	130	112	80	37	38	37	60	112	36	50	21	8	6	8	21	50	7	17	18	16	5	16	18	17
2:	134	143	137	136	123	102	120	96	119	101	121	97	98	88	90	95	72	91	73	87	69	70	68	86	65	81	83	80	64	78	82	79
3:	123	137	143	138	134	120	102	96	98	97	121	101	119	95	90	88	69	87	73	91	72	86	68	70	64	80	83	81	65	79	82	78
4:	122	135	135	142	122	94	90	92	100	118	133	119	100	92	74	92	85	94	89	94	85	71	66	71	67	77	84	77	67	76	75	76
5:	81	117	132	114	141	132	117	114	38	37	60	112	130	112	60	37	6	8	21	50	36	50	21	8	5	16	18	17	7	17	18	16
6:	123	102	120	96	134	143	137	136	88	88	90	95	119	101	121	97	69	70	68	86	72	91	73	87	64	78	82	79	65	81	83	80
7:	134	120	102	96	123	137	143	138	119	95	90	88	98	97	121	101	72	86	68	70	69	87	73	91	65	79	82	78	64	80	83	81
8:	122	96	98	93	122	135	135	142	100	92	74	92	100	118	133	119	85	71	66	71	85	94	89	94	67	76	75	76	67	77	84	77
9:	128	114	54	63	33	64	114	68	130	126	109	46	30	46	109	126	56	106	44	31	2	31	44	106	27	12	23	10	0.	10	23	12
10:	108	82	53	113	32	25	52	40	125	138	127	107	47	29	45	105	103	124	104	57	28	3	26	55	43	39	51	24	1	9	11	22
11:	69	115	115	131	50	42	42	19	111	129	140	129	111	49	34	49	48	110	58	110	48	35	4	35	20	15	41	15	20	14	13	14
12:	32	53	82	113	108	52	25	40	47	107	127	138	125	105	45	29	28	57	104	124	103	55	26	3	1	24	51	39	43	22	11	9
13:	33	54	114	63	128	114	54	63	30	46	109	126	139	126	109	46	2	31	44	106	56	106	44	31	0	10	23	12	27	12	23	10
14:	32	25	52	40	108	82	53	113	47	29	45	105	125	138	127	107	28	3	26	55	103	124	104	57	1	9	11	22	43	39	51	24
15:	69	42	42	19	69	115	115	131	111	49	34	49	111	129	140	129	48	35	4	35	48	110	58	110	20	14	13	14	20	15	41	15
16:	108	52	25	40	32	53	82	113	125	105	45	29	47	107	127	138	103	55	26	3	28	57	104	124	43	22	11	9	1	24	51	39
17:	27	12	10	23	0	10	12	23	56	106	44	31	2	31	44	106	126	124	109	46	30	46	109	126	128	114	63	54	33	54	63	114
18:	43	39	24	51	1	9	22	11	103	124	104	57	28	3	26	55	125	138	127	107	47	29	45	105	108	82	113	53	32	25	40	52
19:	20	15	15	41	20	14	14	13	48	110	58	110	48	35	4	35	111	129	140	129	111	49	34	49	59	115	131	115	59	42	19	42
20:	1	24	39	51	43	22	9	11	28	57	104	124	103	55	26	3	47	107	127	138	125	105	45	29	32	53	113	82	108	52	40	25
21:	8	10	12	23	27	12	10	23	2	31	44	106	56	106	44	31	30	46	109	126	139	126	109	46	33	54	63	114	128	114	63	54
22:	1	9	22	11	43	39	24	51	28	3	26	55	103	124	104	57	47	29	45	105	125	138	127	107	32	25	40	52	108	82	113	53
23:	20	14	14	13	20	15	15	41	48	35	4	35	48	110	58	110	111	49	34	49	111	129	140	129	59	42	19	42	59	115	131	115
24:	43	22	9	11	1	24	39	51	103	55	26	3	28	57	104	124	125	105	45	29	47	107	127	138	108	52	40	25	32	53	113	82
25:	7	17	16	18	5	16	17	18	36	50	21	8	6	8	21	50	130	112	60	37	38	37	60	112	141	132	116	117	61	117	116	132
26:	65	81	80	83	64	78	78	82	72	91	73	87	69	70	68	86	119	101	121	97	98	88	90	95	134	143	136	137	123	102	96	120
27:	67	77	77	84	67	76	76	75	85	94	89	94	85	71	66	71	100	118	133	118	100	92	74	92	122	135	142	135	122	99	93	99
28:	64	80	81	83	65	79	78	82	69	87	73	91	72	86	68	70	98	97	121	101	119	95	90	88	123	137	136	143	134	120	96	102
29:	5	16	17	18	7	17	16	18	6	8	21	50	36	50	21	8	38	37	60	112	130	112	60	37	61	117	116	132	141	132	116	117
30:	64	78	79	82	65	81	80	83	69	70	68	86	72	91	73	87	98	88	90	95	119	101	121	97	123	102	96	120	134	143	136	137
31:	67	76	76	75	67	77	77	84	85	71	66	71	85	94	89	94	100	92	74	92	100	118	133	118	122	99	93	99	122	135	142	135
32:	65	79	78	82	64	80	81	83	72	86	68	70	69	87	73	91	119	95	90	88	98	97	121	101	134	120	96	102	123	137	136	143

(ZI - Bibliothek der Universität Innsbruck)

4.14.b)
Tabelle der Funktionswerte von f_b.

4.15 Beispiel

Einfacher, symmetrischer Graph der Ordnung 26 aus [3]. Der Graph besteht aus zwei gleichen Untergraphen der Ordnung 13, die nach dem rechtsstehenden Schema miteinander verbunden sind(es sind nur die Nachfolger von 1 eingezeichnet, das Schema ist für 2,..,13 zyklisch fortzusetzen):



Dieser Graph ist "2-strongly regular"([13]), d.h. er erfüllt die Bedingung:
für $(x,y) \in G$ bzw. $\overline{G} \setminus \Delta$ sind die die $f_{st}^{(o)}(x,y)$ konstant, also muß $A(f^{(o)}) \equiv f^{(o)}$ sein, und die disjunkte Basis von $H(\Delta,G)$ besteht nur aus den Mengen Δ, G und $\overline{G} \setminus \Delta$.

Die Automorphismenpartition der Knoten dieses Graphen besteht jedoch aus den beiden Mengen $M_1 = \{1,..,13\}$ und $M_2 = \{14,..,26\}$, was durch Grade von Graphen aus $H(\Delta,G)$ nicht festgestellt werden kann.
Die Bestimmung der maximalen vollständigen Untergraphen ([4]) zeigt, daß die Knoten 1 bis 13 in je 6 max. vollst. Untergraphen der Ordnung 5 enthalten sind, die Knoten 14 bis 26 jedoch in je 9. Von Corneil ([1], [2]) werden die beiden Mengen durch die Berechnung der 2-Knoten-Quotientengraphen unterschieden.

$f \equiv f^{(o)}$ von 4.15:

```
**********
ANZAHL SYMMETRISCH EINGEFAERBTE
 1: 1, 7, 8, 9, 12, 16, 18, 19, 20, 21, 22, 24, 25, 26*
 2: 2, 7, 8, 9, 10, 13, 14, 17, 18, 20, 21, 23, 25, 26*
 3: 3, 6, 10, 11, 14, 15, 18, 20, 21, 22, 23, 24, 26*
 4: 4, 6, 10, 11, 12, 14, 15, 18, 19, 21, 22, 23, 24, 25*
 5: 5, 10, 11, 12, 13, 15, 16, 17, 20, 22, 23, 24, 25, 26*
 6: 8, 11, 12, 13, 14, 16, 17, 18, 21, 23, 24, 25, 26*
 7: 9, 12, 13, 14, 15, 17, 18, 19, 22, 24, 25, 26*
 8: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 14, 15, 16, 17, 19, 20, 21, 24, 26*
 9: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 15, 16, 17, 18, 19, 22, 23, 26*
10: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 15, 16, 17, 18, 19, 21, 22, 23, 26*
11: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 16, 17, 18, 20, 21, 23, 24, 26*
12: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 17, 18, 19, 20, 21, 22, 24, 25*
13: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 18, 19, 20, 21, 23, 25*
14: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 17, 19, 20, 22, 24, 26*
15: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 18, 21, 22, 23, 25*
16: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 20, 21, 22, 24*
17: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 20, 21, 26*
18: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 19, 20, 25, 26*
19: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 20, 21, 24*
20: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 23*
21: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 22*
22: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21*
23: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22*
24: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 23, 24*
25: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 24, 25*
26: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 26*
```

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
1:	2	0	1	0	0	1	1	1	1	0	0	1	0	0	0	1	0	1	1	1	1	1	0	1	1	1
2:	0	2	0	1	0	0	1	1	1	1	0	0	1	1	0	0	1	0	1	1	1	1	1	0	1	1
3:	1	0	2	0	1	0	0	1	1	1	1	0	0	1	1	0	0	1	0	1	1	1	1	1	0	1
4:	0	1	0	2	0	1	0	0	0	1	1	1	1	0	1	1	1	0	0	1	0	1	1	1	1	0
5:	0	0	1	0	2	0	1	0	0	0	1	1	1	1	0	1	1	1	0	0	1	0	1	1	1	1
6:	1	0	0	1	0	2	0	0	1	0	0	1	1	1	1	0	1	1	1	0	0	1	0	1	1	1
7:	1	1	0	0	1	0	2	0	0	1	0	0	1	1	1	1	0	1	1	1	0	0	1	0	1	1
8:	1	1	1	0	0	1	0	2	0	0	1	0	0	1	1	1	1	0	1	1	1	0	0	1	0	1
9:	1	1	1	1	0	0	1	0	2	0	1	0	1	0	1	1	1	1	0	1	1	1	0	0	1	0
10:	0	1	1	1	1	0	0	1	0	2	0	1	0	1	0	1	1	1	1	0	1	1	1	0	0	1
11:	0	0	1	1	1	1	0	0	1	0	2	0	1	0	1	0	1	1	1	1	0	1	1	1	0	0
12:	1	0	0	1	1	1	1	0	0	1	0	2	0	1	0	1	0	1	1	1	1	0	1	1	1	0
13:	0	1	0	0	1	1	1	1	1	0	1	0	2	1	0	1	1	0	0	0	0	1	1	0	1	1
14:	0	1	1	1	0	1	1	1	1	1	0	1	1	2	1	0	1	1	0	0	0	0	1	1	0	1
15:	0	0	1	1	1	0	1	1	1	1	1	0	1	0	2	1	0	1	1	0	0	0	0	0	1	0
16:	1	0	0	1	1	1	0	1	1	1	1	1	0	1	1	2	1	0	1	1	0	0	0	0	0	1
17:	0	1	0	0	1	1	1	1	0	1	1	1	1	0	1	1	2	1	0	1	1	0	0	0	0	0
18:	1	0	1	0	0	1	1	1	0	1	0	1	1	1	0	1	1	2	1	0	1	1	0	0	0	0
19:	1	1	0	1	0	0	1	1	1	0	1	1	1	0	0	1	1	1	2	1	0	1	1	0	0	0
20:	1	1	1	0	1	0	0	1	1	1	0	1	0	0	0	0	1	1	1	2	1	0	1	1	0	0
21:	1	1	1	1	0	1	0	0	1	1	1	0	1	0	0	0	0	1	1	1	2	1	0	1	1	0
22:	1	1	1	1	1	0	1	0	0	1	1	1	1	0	1	0	0	0	1	1	1	2	1	0	1	1
23:	0	1	1	1	1	1	0	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	2	1	0	1
24:	1	0	1	1	1	1	1	0	1	0	0	1	0	1	1	1	0	0	0	0	1	1	1	2	1	0
25:	1	1	0	1	1	1	1	1	0	1	0	0	1	0	1	1	0	0	0	0	0	1	1	1	2	1
26:	1	1	1	0	1	1	1	1	1	0	1	0	0	1	0	1	1	0	0	0	0	0	1	1	1	2

Berechnung von f, nachdem die Unterscheidung von M_1 und M_2 dadurch vorgegeben wurde, daß die Knoten 1 bis 13 mit Schleifen versehen wurden: ergibt die Automorphismenpartition der Kanten (überprüft durch Ausprobieren).

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
1:	19	4	17	4	5	17	17	16	16	4	5	16	5	7	6	14	6	15	14	14	13	13	6	15	13	15
2:	5	19	4	17	4	5	17	17	16	16	4	5	16	15	7	6	14	6	15	14	14	13	13	6	15	13
3:	16	5	19	4	17	4	5	17	17	16	16	4	5	13	15	7	6	14	6	15	14	14	13	13	6	15
4:	5	16	5	19	4	17	4	5	17	17	16	16	4	15	13	15	7	6	14	6	15	14	14	13	13	6
5:	4	5	16	5	19	4	17	4	5	17	17	16	16	6	15	13	15	7	6	14	6	15	14	14	13	13
6:	16	4	5	16	5	19	4	17	4	5	17	17	16	13	6	15	13	15	7	6	14	6	15	14	14	13
7:	16	16	4	5	16	5	19	4	17	4	5	17	17	13	13	6	15	13	15	7	6	14	6	15	14	14
8:	17	16	16	4	5	16	5	19	4	17	4	5	17	14	13	13	6	15	13	15	7	6	14	6	15	14
9:	17	17	16	16	4	5	16	5	19	4	17	4	5	14	14	13	13	6	15	13	15	7	6	14	6	15
10:	5	17	17	16	16	4	5	16	5	19	4	17	4	15	14	14	13	13	6	15	13	15	7	6	14	6
11:	4	5	17	17	16	16	4	5	16	5	19	4	17	6	15	14	14	13	13	6	15	13	15	7	6	14
12:	17	4	5	17	17	16	16	4	5	16	5	19	4	14	6	15	14	14	13	13	6	15	13	15	7	6
13:	4	17	4	5	17	17	16	16	4	5	16	5	19	6	14	6	15	14	14	13	13	6	15	13	15	7
14:	3	12	16	12	0	10	10	11	11	12	0	11	0	18	9	1	9	8	1	1	2	2	9	8	2	8
15:	0	3	12	10	12	0	10	10	11	11	12	0	11	8	18	9	1	9	8	1	1	2	2	9	8	2
16:	11	0	3	12	10	12	0	10	10	11	11	12	0	2	8	18	9	1	9	8	1	1	2	2	9	8
17:	0	11	0	3	12	10	12	0	10	10	11	11	12	8	2	8	18	9	1	9	8	1	1	2	2	9
18:	12	0	11	0	3	12	10	12	0	10	10	11	11	9	8	2	8	18	9	1	9	8	1	1	2	2
19:	11	12	0	11	0	3	12	10	12	0	10	10	11	2	9	8	2	8	18	9	1	9	8	1	1	2
20:	11	11	12	0	11	0	3	12	10	12	0	10	10	2	2	9	8	2	8	18	9	1	9	8	1	1
21:	10	11	11	12	0	11	0	3	12	10	12	0	10	1	2	2	9	8	2	8	18	9	1	9	8	1
22:	10	10	11	11	12	0	11	0	3	12	10	12	0	1	1	2	2	9	8	2	8	18	9	1	9	8
23:	0	10	10	11	11	12	0	11	0	3	12	10	12	8	1	1	2	2	9	8	2	8	18	9	1	9
24:	12	0	10	10	11	11	12	0	11	0	3	12	10	9	8	1	1	2	2	9	8	2	8	18	9	1
25:	10	12	0	10	10	11	11	12	0	11	0	3	12	1	9	8	1	1	2	2	9	8	2	8	18	9
26:	12	10	12	0	10	10	11	11	12	0	11	0	3	9	1	9	8	1	1	2	2	9	8	2	8	18

5. Kapitel. Programmierung des Verfahrens zur Konstruktion der disjunkten Basis von $H(\Delta,G_1,..,G_k)$. Zeitabschätzung.

Zunächst werden einige Abänderungen des Verfahrensschrittes A (Definition 4.1) durchgeführt, die zwar das Ergebnis (die Konstruktion der disjunkten Basis) nicht beeinträchtigen, in manchen Fällen das Verfahren jedoch beschleunigen.

5.1 Definition

Verfahrensschritt A' bezeichne die Abbildungsvorschrift, die aus dem Verfahrensschritt A von 4.1 dadurch entsteht, daß dort die Abbildungen f_{st} durch die Abbildungen

$$f_{rst} \quad \left(f_{rst}(x,y) := \begin{array}{l}|\{z\in X \mid f(z,z)=r \wedge f(x,z)=s \wedge f(y,z)=t\}| \\ \text{für } r\in f(\Delta),\ s,t\in \operatorname{Im} f\end{array}\right)$$

ersetzt werden:

$A'(f) := g \circ (f,..,f_{rst},..)$, g Anordnung von $\operatorname{Im}(f..f_{rst}..)$.

5.2 Hilfssatz

Sei $f \in F$ (von 4.1), $s,t \in \operatorname{Im} f$.

Dann gilt: $$f_{st} = \sum_{r\in f(\Delta)} f_{rst} .$$

Beweis $M_{st}(x,y) := \{z\in X \mid f(x,z)=s \wedge f(y,z)=t\}$

$M_{rst}(x,y) := \{z\in X \mid f(x,z)=s \wedge f(y,z)=t \wedge f(z,z)=r\}$.

$M_{st}(x,y) = \bigcup_{r\in f(\Delta)} M_{rst}(x,y)$ für $x,y \in X$:

$r\in f(\Delta) \Rightarrow M_{rst}(x,y) \subset M_{st}(x,y)$ klar.

$z \in M_{st}(x,y)$, $r_o := f(z,z) \Rightarrow z \in M_{r_o st}(x,y)$.

Für $r_1,r_2 \in f(\Delta)$ gilt: $M_{r_1 st}(x,y) \cap M_{r_2 st}(x,y) \neq \emptyset \Rightarrow$

$\Rightarrow r_1 = f(z,z) = r_2$, also $M_{rst}(x,y)$ disjunkt für festes s,t.

$$\Rightarrow f_{st}(x,y) = |M_{st}(x,y)| = \sum_{r\in f(\Delta)} |M_{rst}(x,y)| =$$

$$= \sum_{r\in f(\Delta)} f_{rst}(x,y)$$

□

5.3 Folgerung

$\forall f \in F$ (siehe 4.1): $A'(f) \leq A(f) \leq f$.

Beweis

$A'(f)(x,y) = A'(f)(a,b) \Rightarrow$

$\forall s,t \in \mathrm{Im}\, f, \forall r \in f(\Delta)$: $f_{rst}(x,y) = f_{rst}(a,b)$, $f(x,y)=f(a,b)$

$\Rightarrow \forall s,t \in \mathrm{Im}\, f$: $\underline{f_{st}(x,y)} = \sum_{r\in f(\Delta)} f_{rst}(x,y) =$

$= \sum_{r\in f(\Delta)} f_{rst}(a,b) = \underline{f_{st}(a,b)}$,

$f(x,y) = f(a,b)$,

also $A(f)(x,y) = A(f)(a,b)$.

$A(f) \leq f$ nach 4.2.1 .

5.4 Hilfssatz

Sei $f \in F$ mit $f \leq \chi_\Delta$, d.h. $f(x,y)=f(a,b) \Rightarrow \left[x=y \Leftrightarrow a=b\right]$,

$r \in f(\Delta)$, $s,t \in \mathrm{Im}\, f$, $j \in \mathbb{N}$, $0 \leq j \leq n = |X|$.

Dann gilt: $f^{-1}_{rst}(j) = \left[f^{-1}(s)\ (1)\ f^{-1}(r)\right]\ (j)\ \left[f^{-1}(t)\right]^T$.

Beweis

$B(x,z) := \{z' \in X \mid f(x,z)=s \wedge f(z,z')=r\}$.

Es gilt: $z' \in B(x,z) \Rightarrow f(z,z') = r \Rightarrow (r\in f(\Delta), f\leq\chi_\Delta) \Rightarrow z=z'$,

also $B(x,z) = \emptyset$ oder $B(x,z) = \{z\}$.

$f_{rst}(x,y) = j \Leftrightarrow j = |\{z \in X \mid f(z,z)=r \wedge f(x,z)=s \wedge f(y,z)=t\}| =$

$(\ f(z,z)=r \wedge f(x,z)=s \Leftrightarrow B(x,z)=\{z\} \Leftrightarrow |B(x,z)|=1\ )$

$= |\{z\in X \mid |B(x,z)| = 1 \wedge f(y,z) = t\}| \Leftrightarrow$

$(\ |B(x,z)| = 1 \Leftrightarrow (x,z) \in f^{-1}(s)\ (1)\ f^{-1}(r)\ )$

$\Leftrightarrow$ $(x,y) \in$ rechte Seite.

Der Verfahrensschritt A' wird jetzt in zwei Schritte (A_1, A_2) unterteilt $(A' = A_2 \circ A_1)$, nämlich in die Berechnung von $(f,..,f_{rst},..)$ für die Elemente (x,x) der Diagonale, A_1, und für die übrigen Elemente (x,y) mit $x \neq y$, A_2:

$$h_1(x,y) := \begin{cases} (f,..,f_{rst},..)(x,v) & \text{für } x=y \\ f(x,y) & x\neq y \end{cases}$$

$A_1(f) := g_1 \circ h_1$, wobei g_1 Anordnung von Im h_1.

$$h_2(x,y) := \begin{cases} f(x,y) & \text{für } x=y \\ (f,..,f_{rst},..)(x,y) & x\neq y \end{cases}$$

$A_2(f) := g_2 \circ h_2$, g_2 Anordnung von Im h_2.

A_1 entspricht der Anwendung von "extended features" bei Unger [1o] und Levi [5] , dem "Algoritmo di affinamento" bei Sirovich [7].
Bei vielen "unregelmäßigen" Graphen genügt bereits die iterierte Anwendung von A_1 auf die durch den Grad des gegebenen Graphen erzeugte Partition der Diagonale zur Bestimmung der Automorphismenpartition der Knoten.
Für spezielle Graphen, nämlich für Arboreszenzen und zusammenhängende funktionale Graphen hat Tinhofer in [9] bewiesen, daß dadurch die Automorphismenpartition erhalten wird.
Da A_1 außerdem nur n Berechnungen von Funktionswerten $(f,..,f_{rst},..)(x,x)$ erfordert gegenüber $n.(n-1)$ bei A_2, definieren wir den folgenden Verfahrensschritt A":

5.5 Definition
"Verfahrensschritt A"" bezeichne die Abbildung: $A" := A_2 \circ A_1^n$, d.h. zuerst wird A_1 sooft durchgeführt, bis die Diagonale nicht mehr weiter verfeinert wird(nach Satz 3.11 ist dies nach spätestens n-mal der Fall), und dann A_2.

Der "Verfahrensschritt A" führt ebenfalls zur disjunkten Basis von $H(\Delta,G_1,..,G_k)$:

5.6 Satz
Voraussetzungen wie 4.4 . $f' := (A")^{n^2}(f^{(o)})$.
Dann gilt: $f' \equiv f$.

Beweis

$f' \leq f$: durch Induktion.

$A''(f^{(o)}) = A_2 o A_1 o A_1^{n-1}(f^{(o)}) = A' \circ A_1^{n-1}(f^{(o)}) \overset{(5.3)}{\leq}$

$A \circ A_1^{n-1}(f^{(o)})$. 5.3 gilt auch für A_1: $A_1(f) \leq f$:

$A \circ A_1^{n-1}(f^{(o)}) \overset{(4.2.2)}{\leq} A(f^{(o)})$.

Sei jetzt $(A'')^i(f^{(o)}) \leq A^i(f^{(o)})$.

$(A'')^{i+1}(f^{(o)}) = A' \circ A_1^{n-1} \circ (A'')^i(f^{(o)}) \overset{(5.3)}{\leq}$

$\leq A \circ A_1^{n-1} \circ (A'')^i(f^{(o)}) \leq$

$\left[A_1^{n-1} \circ (A'')^i(f^{(o)}) \overset{(5.3)}{\leq} (A'')^i(f^{(o)}) \overset{(I.A.)}{\leq} A^i(f^{(o)})\right]$

$\leq A^{i+1}(f^{(o)})$.

$f \leq f'$:

Sei $M' := \left\{M'_j\right\}_{j \in \mathrm{Im}\, f'}$, $M'_j := f'^{-1}(j)$.

Zeige: $M' \subset H(\Delta, G_1, .., G_k)$.

genau wie in Teil I des Beweises von 4.4: $M^{(o)}$ bleibt gleich;
in Hilfssatz 4.5 ist $(r, r_{m_i m_i}, .., r_{11}) := (f^{(i)}, .., f^{(i)}, .)(a,b)$
zu ersetzen durch $(m_o, .., m_{rst}, ..) := (f'^{(i)}, .., f'^{(i)}_{rst}, ..)(a,b)$,
weiters $M_s^{(i)}(r_{st}) M_t^{(i)T}$ durch:
$\left[M_s^{(i)}\ (1)\ M_r^{(i)}\right](m_{rst})\left[M_t^{(i)}\right]^T$, wobei jetzt der Durchschnitt
über die $r \in f(\Delta)$, $s,t \in \mathrm{Im}\, f'^{(i)}$ zu nehmen ist, und
$M_r^{(i)}$ durch $M_{m_o}^{(i)}$ zu ersetzen. (Siehe auch Hilfssatz 5.4).
Die Aufteilung von A' in A_1 und A_2 ändert hier nichts, die
Formel gilt dann eben nur für die $l \in f'^{(i+1)}(\Delta)$
(bzw. $f'^{(i+1)}(X \times X \setminus \Delta)$), für die übrigen l haben sich die
$M_l^{(i+1)}$ nicht geändert, d.h. es gibt ein l_1 mit $M_{l_1}^{(i)} = M_l^{(i+1)}$.
Induktionsschluß wie in Teil I) des Beweises 4.4:
$M' \subset H(\Delta, G_1, .., G_k)$.

also $M' \subset H(\Delta, G_1, .., G_k)$:

$\forall\, j \in \mathrm{Im}\, f'\ \exists\, I_j \in \mathrm{Im}\, f : \quad M'_j = \bigcup_{s \in I} M_s$.

Seien $x,y,a,b \in X$, $j := f'(x,y)$:

$f(x,y) = f(a,b) =: s \Rightarrow s \in I_j$, $\underline{(a,b) \in M_s \subset M'_j} \Rightarrow$

$\Rightarrow f'(a,b) = j = f'(x,v)$,

also $f \leq f'$ und somit $f \equiv f'$.

5.7 Hilfssatz

Hilfssatz 4.6 gilt auch für A_1 und A_2, d.h.
falls $f'^{(i+1)} = A_1(f'^{(i)})$ und falls $f'^{(i+1)} = A_2(f'^{(i)})$.

Beweis für A_2 wie bei 4.6 .

$\left[M_s^{(i)}\ (1)\ M_r^{(i)}\right]\ (m)\ \left[M_t^{(i)}\right]^T = \left[M_t^{(i)}\ (1)\ M_r^{(i)}\right]\ (m)\ \left[M_s^{(i)}\right]^T$,

leicht zu sehen mit Hilfssatz 5.4 .

für A_1 durch Induktion:

für $f^{(o)}$ wie in 4.6. Behauptung gelte für i:

$M_1^{(i+1)}$, $1 \in f^{(i+1)}(\Delta)$: $M_1^{(i+1)} \subset \Delta$, $M_1^{(i+1)T} = M_1^{(i+1)}$.

$1 \notin f^{(i+1)}(\Delta)$: $\exists\, 1_1 \in \mathrm{Im}\, f^{(i)}$: $M_{1_1}^{(i)} = M_1^{(i+1)}$; nach I.A.

oder weil $f^{(i)} = A_2(f^{(i-1)})$: $\exists\, 1'_1 \in \mathrm{Im}\, f^{(i)}$: $M_{1_1}^{(i)\,T} = M_{1'_1}^{(i)}$

$1' := g_i(1'_1) \notin f^{(i+1)}(\Delta)$: $M_{1'_1}^{(i)} = M_{1'}^{(i+1)} =$

$= M_1^{(i+1)T}$ □

Als Anordnung g_i in 4.1 und g_o in 4.4 wird folgende Vorschrift verwendet:

Zuerst wird $\mathrm{Im}(f^{(i)}, .., f_{rst}^{(i)}, ..)$ lexikographisch angeordnet. (im Programm Zeile 642-658 für A_2, 5o7-526 für A_1),
dann werden diejenigen Tupel $(f^{(i)}, .., f_{rst}^{(i)}, ..)(x,y)$, welche lexikographisch größer sind als $(f^{(i)}, .., f_{rst}^{(i)}, ..)(y,x)$, unmittelbar hinter $(f^{(i)}, .., f_{rst}^{(i)}, ..)(y,x)$ eingefügt. (Zeile 746-751). Nach Hilfssatz 5.6 ist diese zweite Vor-

schrift unabhängig von der Wahl von (x,y). Falls $M_j^{(i)T} \neq M_j^{(i)}$ ist, gilt dann aufgrund dieser Anordnung: $M_j^{(i)T} = M_{j+1}^{(i)}$. Hilfssatz 5.6(4.6) erspart dann einige Berechnungen von $(f^{(i)},..,f_{rst}^{(i)},..)(y,x)$:

Falls $M_j^{(i)}$ aufgeteilt wurde auf $M_{j_1}^{(i+1)},..,M_{j_r}^{(i+1)}$, so wird auch $M_j^{(i)}$ aufgeteilt auf $M_{j'_1}^{(i+1)},..,M_{j'_r}^{(i+1)}$;

für die $(x,y) \in M_j^{(i)T}$ braucht also $(f^{(i)},..,f_{rst}^{(i)},..)(x,y)$ nicht mehr berechnet zu werden. (Zeile 708-723, bzw. 680-692 für r=1).

Eine Einsparung von Berechnungen von $f_{rst}^{(i)}(x,y)$ bringt auch folgende Tatsache:

Falls für ein $r \in f^{(i+1)}(\Delta)$ und für $s,t \in \text{Im } f^{(i+1)}$ gilt:

$\exists\, r' \in f^{(i)}(\Delta)$, $s',t' \in \text{Im } f^{(i)}$ mit $M_r^{(i+1)}=M_{r'}^{(i)}$, $M_s^{(i+1)}=M_{s'}^{(i)}$ und $M_t^{(i+1)}=M_{t'}^{(i)}$ ($M_{r'}^{(i)}$, $M_{s'}^{(i)}$ und $M_{t'}^{(i)}$ wurden durch Schritt i+1 nicht aufgeteilt),

so ist für $x,y \in X$: $f_{rst}^{(i+1)}(x,y) = f_{r's't'}^{(i)}(x,y)$

und wegen $f^{(i+1)} \leq f_{r's't'}^{(i)}$ braucht $f_{rst}^{(i+1)}$ nicht berechnet zu werden.

Für $M_j^{(i+1)}$ wird die Eigenschaft: $\exists\, j' \in \text{Im } f^{(i)}$: $M_j^{(i+1)}=M_{j'}^{(i)}$ im Attribut "ALT" der Partitionsmenge festgehalten. (Zeile 167) ("GANZ ALT" gibt an, ob die Partitions-menge $M \subset \Delta$ seit der letzten Anwendung von A_2 einmal aufgeteilt wurde)

5.8 Abschätzung der vom Verfahren benötigten Rechenzeit

1) Zeitaufwand zur Berechnung von

$(f,..,f_{rst},..)(x,y)$ für ein $(x,y) \in X \times X$:

Für höchstens n Tripel (r_j,s_j,t_j) $(r_j \in f(\Delta), s,t \in \mathrm{Im}\, f)$

ist $m_{r_j s_j t_j} := f_{r_j s_j t_j}(x,y)$ von Null verschieden, denn

$$\sum_{r \in f(\Delta), s,t \in \mathrm{Im}\, f} f_{rst}(x,y) = |X| = n.$$

$(f,..,f_{rst},..)(x,y)$ wird daher als Liste von lexikografisch absteigend angeordneten Quadrupeln $(r_j,s_j,t_j,m_{r_j s_j t_j})$, wobei $m_{r_j s_j t_j} \neq o$ ist, abgespeichert.

Die Berechnung der $m_{r_j s_j t_j} \neq o$ geschieht nach folgender Methode:

```
integer x,y,z,r,s,t;
for z := 1 step 1 until n do
begin      r := f(z,z);    s := f(x,z);    t := f(y,z);
     [*]  if ("Zähler (r,s,t,m_rst) in der Quadrupelliste
              vorhanden") then m_rst := m_rst + 1
          else ("Füge neuen Zähler (r,s,t,1) an der
                richtigen Stelle in die Quadrupel-
                liste ein");
end;
```

Diese Berechnung der m_{rst} für ein (x,y) erfolgt innerhalb der procedure G(K), die zur class GRAPH gehört. (Zeile 22o)

Bei $z = j$ muß r,s,t mit den höchstens j r',s',t' der bereits vorhandenen Quadrupel verglichen werden:
die Berechnung von $(f,..,f_{rst},..)(x,y)$ erfordert höchstens
$\sum_{j=1}^{n} j \simeq \underline{\underline{n^2 \text{ Schritte}}}$.

2) Zeitaufwand zur Anordnung der m Listen $(f,..,f_{rst},..)(x,y)$
$\mathrm{Im}\, f$

Für das Einordnen der j-ten Liste sind höchstens j Vergleiche

mit bereits errechneten Listen nötig, ein solcher Vergleich erfordert den Vergleich von höchstens n Quadrupeln:

Die Anordnung von $Im(f,..,f_{rst},..)$ erfordert höchstens $\sum_{j=1}^{m} j \cdot n$ Schritte.

Daraus ergibt sich der

3) Zeitaufwand für einen Schritt A_1 (Partition von X):

n Berechnungen von $(f,..,f_{rst},..)(x,y)$ $\quad \simeq n^3$

Anordnung von höchstens n Listen: $\sum_{j=1}^{n} j \cdot n \quad \simeq n^3$

mit t_1 proportional n^3

und der

4) Zeitaufwand für einen Schritt A_2 (Partition von X×X):

$\sim n^2$ Berechnungen von $(f,..,f_{rst},..)(x,y)$: $\quad \simeq n^4$

Anordnung von höchstens n^2 Listen $\quad \sum_{j=1}^{n^2} j \cdot n \quad \simeq n^5$

mit t_2 proportional n^5.

Die Anzahl der Durchführungen von A_1 ist höchstens gleich n (3.11). Für A_2 ist dieselbe Abschätzung nach (3.11) n^2; in den praktisch berechneten Beispielen mit n zwischen 1o und 2o brauchte A_2 jedoch höchstens fünfmal durchgeführt zu werden.

Die Rechenzeit des Verfahrens ist somit theoretisch maximal der siebten Potenz von n proportional; die praktisch erprobten Werte liegen jedoch deutlich darunter.

Die am häufigsten durchlaufenen Unterprogramme wurden in Assembler-Sprache programmiert:
INCM1 (Zeile 142) führt die Anweisung [*] durch,
COMPARE (139) vergleicht zwei Listen von ganzen Zahlen,
COMPR(qliste,L) (145) speichert die Quadrupelliste qliste als Liste von ganzen Zahlen in einen externen Speicherbereich(außerhalb des SIMULA-Programms) von L.I1 bis L.I2 ab, und SETLU(I) (148) ermöglicht es, derartigen Speicherbereich wieder freizugeben.

5.9 Programm

Da ein Programm für eine spezielle Anwendung natürlich auf die dort vorkommenden Graphen ausgerichtet sein muß, wurde hier versucht, mehr Wert auf Übersichtlichkeit zu legen. Daher wurde auch die Sprache SIMULA verwendet, die durch das Klassenkonzept die Möglichkeit bietet, Objekte mit Attributen und spezifischen Anweisungen zu definieren:

z.B. class partitionsmenge; ...eine Partitionsmenge $M_j^{(i)}$ (Zeile 166).

```
begin boolean alt, ganz alt;..bereits erwähnte logische
                              Variable.
      ref(iliste) liste; .....Verweis auf die Liste
                              g_i^-1(j).
      ref(head) kliste1,kliste2;...Listenköpfe der Liste
                              der (x,y),die Elemente dieser
                              P.menge sind, bezüglich Graph1
                              bzw. Graph2.
      integer INDEX;..... INDEX = j.
      boolean procedure complete; dieses Unterprogramm prüft,
      (für eine Partitionsmenge ⊂ Δ ), ob der durch die
      Knoten z, für die (z,z) Element dieser Partitions-
      menge ist, bestimmte Untergraph"vollständig"ist,
      d.h. alle Kanten (x,y) mit x≠y aus derselben
      Partitionsmenge sind.
    die Anweisungen dieser Klasse führen eine Vorbesetzung
    der Attribute durch.
end PARTITIONSMENGE;
```

Weiters wird z.B. der Verfahrensschritt A_1 von den Anweisungen der Class KNOTENPARTITION durchgeführt (Zeile 5oo - 582), der Verfahrensschritt A_2 jedoch von den Anweisungen der Class KANTENPARTITION (Zeile 632 - 752).

Das Programm führt für zwei gegebene Graphen die parallele Berechnung von f und f' durch, wobei die Bedingungen von 4.9 bei jedem Schritt überprüft werden.
Es sind folgende Ausgänge möglich:

- bei der Überprüfung der Bedingungen von 4.9 werden Unterschiede festgestellt, die eine Isomorphie ausschließen.
- die Knoten beider Graphen werden eindeutig angeordnet: ein einziger Isomorphismus existiert.
- die Voraussetzungen von 4.11 sind für ein x_o mit $f(x_o,x_o) =: j$. Es sind dann $|M_j|$ Isomorphismen möglich.
- falls keine der obigen Bedingungen erfüllt ist, wird versucht, die Graphen parallel in unabhängige Untergraphen zu zerlegen, für die dann einzeln die Voraussetzungen von 4.11 geprüft werden.

Die beiden Graphen können auf folgende Arten gegeben sein:

- durch Einlesen der Nachfolgerliste für jeden Knoten (procedure EINLESENG, Zeile 1o7).
- durch Besetzung der Adjazenzmatrix nach einer Zufallsfunktion (procedure ZUFALLSGRAPH, Zeile 29).
- durch Erzeugung eines "starred polygon" ([14]) (procedure STARRED POLYGON, Zeile 49):
  das ist ein Graph, dessen Knoten derart angeordnet werden können($X = \{x_1,..,x_n\}$, daß gilt:
  $(x_i,x_j) \in G \iff \forall\, k=1,..,n-1: (x_{i+k},x_{j+k}) \in G.$
  (i+k,j+k modulo n gerechnet)
  Ein solcher Graph ist gegeben durch sein "Symbol"
  $S := \{s \mid (x_1,x_{1+s}) \in G\}$. (Der bei 4.15 links abgebildete Untergraph ist ein "starred polygon" und hat das Symbol {1,3,4,9,1o,12})

Der zweite Graph kann außerdem durch eine Zufallspermutation (procedureZUFALLSPERMUTATION, Zeile 81) aus dem ersten erzeugt werden (procedure PERMUTIERE , Zeile 99).

```
 0 ≠BEGIN≠  ≠REAL≠ ZEITLIMIT;
 1
 2          ≠INTEGER≠ N,U;
 3   ≠REAL≠ X;   ≠BOOLEAN≠ SYMMETRISCH, EINLESEN, SCHLEIFEN, AUS;
 4          ZEITLIMIT := 1000*ININT;
 5 IMAGE :- BLANKS(70);
 6 AIF..
 7 OUTIMAGE;  EJECT(LINE+3);
 8 OUTIMAGE; OUTTEXT(≠≠-*-*-*
```

```
                    ≠FOR≠ I := 1 ≠STEP≠ 1 ≠UNTIL≠ N ≠DO≠
                      F[I,(I+J)≠MOD≠N + 1] := 1;
                    J := N - J - 2;
                    ≠IF≠ ¬ SYMMETRISCH ≠THEN≠ ≠ELSE≠
                    ≠FOR≠ I := 1 ≠STEP≠ 1 ≠UNTIL≠ N ≠DO≠
                      F[I,(I+J)≠MOD≠N + 1] := 1;
          ≠END≠ WHILE ¬ENDIMAGE;
          SETPOS(POS-1);   OUTCHAR(§)§); OUTIMAGE;
≠END≠ STARRED POLYGON;

≠PROCEDURE≠ AUSDR(F,N);
≠INTEGER≠ ≠ARRAY≠ F; ≠INTEGER≠ N;
≠BEGIN≠ ≠INTEGER≠ I,J;
          OUTIMAGE;
          ≠FOR≠ I := 1 ≠STEP≠ 1 ≠UNTIL≠ N ≠DO≠
          ≠BEGIN≠ OUTINT(I,2);  OUTCHAR($:$);
                    ≠FOR≠ J := 1 ≠STEP≠ 1 ≠UNTIL≠ N ≠DO≠
                    ≠IF≠ F[I,J]  = 1 ≠THEN≠
                      ≠BEGIN≠ OUTINT(J,2); OUTCHAR($,$); ≠END≠;
                    SETPOS(POS-1); OUTCHAR($*$);  SETPOS(POS+2);
          ≠END≠ FOR I;
≠END≠ AUSDR;


≠PROCEDURE≠       ZUFALLSPERMUTATION          (PERMUT,N);
          ≠INTEGER≠ ≠ARRAY≠ PERMUT; ≠INTEGER≠ N;

≠BEGIN≠ ≠INTEGER≠ I,J,K,M,Z;
          ≠FOR≠ K := 1 ≠STEP≠ 1 ≠UNTIL≠ N ≠DO≠ PERMUT[K] := K;
OUTIMAGE; OUTTEXT(≠≠ZUFALLSPERMUTATION, U=≠≠);
OUTINT(U,TRUNK(LN(U)/LN(10.0)) + 1 );  OUTIMAGE;
          M := RANDINT(1,N,U);
          ≠FOR≠ K := 1 ≠STEP≠ 1 ≠UNTIL≠ M ≠DO≠
        ≠BEGIN≠  I := RANDINT(1,N,U);        J := RANDINT(1,N,U);
                    Z := PERMUT[I];  PERMUT[I] := PERMUT[J];
                    PERMUT[J] := Z;
          ≠END≠ FOR K;
≠FOR≠ K := 1 ≠STEP≠ 1 ≠UNTIL≠ N ≠DO≠ OUTINT(K,3); OUTIMAGE;
≠FOR≠ K := 1 ≠STEP≠ 1 ≠UNTIL≠ N ≠DO≠ OUTINT(PERMUT[K],3); OUTI
                                                    MAGE;
≠END≠ ZUFALLSPERMUTATION;


≠PROCEDURE≠       PERMUTIERE          (F1,P,F2,N);
          ≠INTEGER≠ ≠ARRAY≠ F1,F2,P; ≠INTEGER≠ N;
≠BEGIN≠ ≠INTEGER≠ I,J;
        ≠FOR≠I:=1≠STEP≠1≠UNTIL≠N≠DO≠≠FOR≠J:=1≠STEP≠1≠UNTIL≠N≠DO≠
          F2[P[I],P[J]] := F1[I,J];
≠END≠PERMUTIERE;


≠PROCEDURE≠ EINLESENG (F,N,SYMMETRISCH);
      ≠INTEGER≠ ≠ARRAY≠ F;  ≠INTEGER≠ N; ≠BOOLEAN≠ SYMMETRISCH;

≠BEGIN≠ ≠INTEGER≠ I,J,K;
          INIMAGE;    OUTTEXT(≠≠>>>>>>>≠≠);
          ≠WHILE≠ SYSIN.IMAGE.SUB(1,4) ≠NE≠ ≠≠ENDE≠≠ ≠DO≠
          ≠BEGIN≠
OUTTEXT(SYSIN.IMAGE.SUB(1,70));OUTIMAGE; SYSIN.SETPOS(71);
SYSIN.IMAGE.SKIPBLANKS; ≠IF≠ SYSIN.POS < 81 ≠THEN≠ ≠BEGIN≠
```

```
116 OUTTEXT(SYSIN.IMAGE.SUB(7,10)); OUTIMAGE; ≠END≠; SYSIN.SETPOS
                                                    (1);
117                     ≠WHILE≠ ¬ ENDIMAGE ≠DO≠
118                     ≠BEGIN≠ I := ININT;
119                               ≠WHILE≠ INCHAR ≠NE≠ $*$ ≠DO≠
120                             ≠BEGIN≠  J := ININT;        F[I,J] :=1;
121                                 ≠IF≠ SYMMETRISCH ≠THEN≠ F[J,I]:=1;
122                               ≠END≠ WHILE NE * ;
123                     ≠END≠  WHILE < K;
124                     ININIMAGE;
125         ≠END≠ WHILE NE ENDE;        SYSIN.SETPOS(79);    OUTIMAGE;
126 ≠END≠ EINLESEN;
127
128 LINK     ≠CLASS≠  QUADRUPEL;
129 ≠BEGIN≠ ≠INTEGER≠K,S,T,M ;≠END≠;
130
131 HEAD     ≠CLASS≠  QUADRUPELLISTE;
132 ≠BEGIN≠ ≠REF≠(HEAD)RESERVOIR;
133          ≠INTEGER≠ I;
134          RESERVOIR :- ≠NEW≠ HEAD;
135          ≠FOR≠ I := 1 ≠STEP≠ 1 ≠UNTIL≠ N ≠DO≠
136          ≠NEW≠ QUADRUPEL.INTO(RESERVOIR);
137 ≠END≠    QUADRUPELLISTE;
138
139 ≠EXTERNAL≠ ≠COMPASS≠ ≠INTEGER≠ ≠PROCEDURE≠          COMPARE(L1,L
                                                    2);
140 ≠REF≠(ILISTE) L1,L2;;
141
142 ≠EXTERNAL≠ ≠COMPASS≠     ≠PROCEDURE≠          INCM1(QLISTE,PR,PS,P
                                                    T);
143 ≠INTEGER≠ PR,PS,PT;  ≠REF≠(HEAD) QLISTE;;
144
145 ≠EXTERNAL≠ ≠COMPASS≠     ≠PROCEDURE≠          COMPR(QLISTE,L);
146 ≠REF≠(HEAD) QLISTE; ≠REF≠(ILISTE) L;;
147
148 ≠EXTERNAL≠ ≠COMPASS≠     ≠PROCEDURE≠          SETLU(I);
149 ≠INTEGER≠ I;;
150
151 ≠CLASS≠ ILISTE;
152 ≠BEGIN≠ ≠INTEGER≠ I1,I2;
153          ≠INTEGER≠ ≠PROCEDURE≠ LAENGE;  LAENGE := I2 - I1 + 1;
154 ≠END≠ ILISTE;
155
156
157
158 LINK     ≠CLASS≠  KK;
159 ≠BEGIN≠ ≠REF≠(PARTITIONSMENGE)PM; ≠END≠;
160 KK       ≠CLASS≠  KNOTEN(X);          ≠INTEGER≠ X;
161          ≠BEGIN≠  ≠END≠     KNOTEN;
162 KK       ≠CLASS≠  KANTE(X,Y);         ≠INTEGER≠ X,Y;
163          ≠BEGIN≠ ≠REF≠(KANTE) T ;;  ≠END≠     ;
164
165
166 LINK     ≠CLASS≠  PARTITIONSMENGE;
167          ≠BEGIN≠  ≠BOOLEAN≠ ALT, GANZ ALT;
168                   ≠REF≠(ILISTE) LISTE;
169                   ≠REF≠(HEAD) KLISTE1,KLISTE2;
170                   ≠INTEGER≠ INDEX;
171
172 ≠BOOLEAN≠ ≠PROCEDURE≠ COMPLETE;
```

```
173 ≠IF≠ KLISTE1.FIRST.SUC = = KLISTE1.LAST ≠THEN≠ COMPLETE := ≠TR
                                                    UE≠≠ELSE≠
174 ≠BEGIN≠
175           ≠REF≠(KNOTEN)K;    ≠INTEGER≠X;
176           ≠REF≠(PARTITIONSMENGE)PV;
177           K:-KLISTE1.FIRST;       X:=K.X;
178           K:-K.SUC;       PV :- GRAPH1.KF[X,K.X].PM;
179           ≠FOR≠ K:- K.SUC    ≠WHILE≠ K =/= ≠NONE≠ ≠DO≠
180        ≠IF≠ PV =/= GRAPH1.KF[X,K.X].PM ≠THEN≠ ≠GOTO≠ E COMPLETE;
181           COMPLETE := ≠TRUE≠;
182 E COMPLETE.. ;  ≠END≠ COMPLETE ELSE;
183                      LISTE :- ≠NEW≠ ILISTE;
184                      KLISTE1 :- ≠NEW≠ HEAD;
185                      KLISTE2 :- ≠NEW≠ HEAD;
186           ≠END≠ PARTITIONSMENGE;
187
188 ≠PROCEDURE≠       AUSGABE(H);                  ≠REF≠ (HEAD) H;
189
190 ≠BEGIN≠ ≠REF≠(PARTITIONSMENGE) PM; ≠REF≠(KNOTEN)K;
191           PM :- H.FIRST;          OUTIMAGE;
192           ≠WHILE≠ PM =/= ≠NONE≠ ≠DO≠  ≠INSPECT≠ PM ≠DO≠
193           ≠BEGIN≠  ≠FOR≠ K :- KLISTE1.FIRST, KLISTE2.FIRST ≠DO≠
194                      ≠BEGIN≠  OUTCHAR($[$);
195                               ≠WHILE≠ K =/= ≠NONE≠ ≠DO≠
196                              ≠BEGIN≠  OUTINT(K.X,2); OUTCHAR($,$);
197                                       K :- K.SUC;
198                               ≠END≠ WHILE K;
199                               SETPOS(POS-1);     OUTCHAR($]$);
200                             OUTCHAR($:$);
201                      ≠END≠ FOR K:- KLISTE1 ;
202                 SETPOS(POS-1); OUTCHAR($;$);          SETPOS(POS+2);
203                      PM :- PM.SUC;
204           ≠END≠ WHILE PM;
205 ≠END≠ AUSGABE;
206
207
208
209
210
211
212
213 ≠REF≠(GRAPH)      GRAPH1, GRAPH2;
214
215 ≠CLASS≠ GRAPH (F,N);
216           ≠INTEGER≠ ≠ARRAY≠ F;        ≠INTEGER≠ N;
217
218 ≠BEGIN≠ ≠REF≠ (KK)          ≠ARRAY≠  KF [1..N,1..N] ;
219
220       ≠REF≠ (ILISTE) ≠PROCEDURE≠           G(K);    ≠REF≠(KK) K;
221   ≠BEGIN≠   ≠REF≠(PARTITIONSMENGE) PR,PS,PT;     ≠REF≠(KNOTEN)KZ;
222                      ≠INTEGER≠ X,Y,Z,R;          ≠BOOLEAN≠ ALT;
223                      ≠REF≠(HEAD)KLISTE;
224
225                      ≠IF≠ K ≠IS≠ KNOTEN ≠THEN≠ ≠GOTO≠ KN;
226                    X := K≠QUA≠KANTE.X;          Y := K≠QUA≠KANTE.Y;
227                      PR :- KNOTP.LAST;
228                      ≠WHILE≠ PR =/= ≠NONE≠ ≠DO≠
229                      ≠BEGIN≠
230                               ALT := PR.GANZ ALT;
231                  R := PR.INDEX;   KLISTE:-≠IF≠≠THIS≠GRAPH==GRAPH1
```

```
232                     ≠THEN≠ PR.KLISTE1 ≠ELSE≠ PR.KLISTE2;
233                             KZ :- KLISTE.FIRST;
234                             ≠WHILE≠ KZ =/= ≠NONE≠ ≠DO≠
235                             ≠BEGIN≠  Z := KZ.X;
236                             PS :- KF [X,Z].PM;
237                             PT :- KF [Y,Z].PM;
238                                     ≠IF≠ ALT ≠THEN≠
239                                   ≠BEGIN≠  ≠IF≠ PS.ALT ∧ PT.ALT
240                                                 ≠THEN≠ ≠GOTO≠ E1;
241                                     ≠END≠ THEN;
242
243                               INCM1(GLISTE,R,PS.INDEX,PT.INDEX);
244
245                             E1..    KZ :- KZ.SUC;
246                             ≠END≠   WHILE KZ =/= NONE;
247                             PR :- PR.PRED;
248                     ≠END≠   WHILE PR =/= NONE;
249                     ≠GOTO≠ E;
250
251
252             KN..    X := K≠QUA≠KNOTEN.X;
253                     PR :- KNOTP.LETZTE ALT;
254 B..                 ALT := PR.ALT;  ≠IF≠ ALT ∧ ¬KN1 ≠THEN≠ ≠GOTO≠
                                                        E3;
                  R := PR.INDEX;   KLISTE:-≠IF≠≠THIS≠GRAPH==GRAPH1
255                   ≠THEN≠ PR.KLISTE1 ≠ELSE≠ PR.KLISTE2;
256                 KZ :- KLISTE.FIRST;
257                 ≠WHILE≠ KZ =/= ≠NONE≠ ≠DO≠
258                 ≠BEGIN≠  Z := KZ.X;
259                             PS :- KF [X,Z].PM;
260                  ≠IF≠ (≠IF≠ALT∧KN1≠THEN≠PS.ALT≠ELSE≠ ≠FALSE≠)
261                             ≠THEN≠ ≠GOTO≠ E2;
262
263                               INCM1(GLISTE,R,PS.INDEX,PS.INDEX);
264
265                             KZ :- KZ.SUC;
266 E2..                ≠END≠   WHILE KZ =/= NONE;
267                     PR :- PR.PRED;
268 E3..                ≠IF≠ PR =/= ≠NONE≠ ≠THEN≠ ≠GOTO≠ B;
269                     ;
270 E..                COMPR(QLISTE,L);     G:- L;
271             ≠END≠ G(K);
272
273
274             ≠REF≠(HEAD) P;              ≠INTEGER≠ I,J,F1,F2,MAX;
275             ≠REF≠(PARTITIONSMENGE) ≠PROCEDURE≠          WO(I);
276                     ≠INTEGER≠ I;
277             ≠BEGIN≠ ≠ REF≠ (PARTITIONSMENGE) PM;
278                     PM :- P.FIRST;
279                     ≠WHILE≠ PM =/= ≠NONE≠ ≠DO≠
280                 ≠BEGIN≠  ≠IF≠ I < PM.INDEX ≠THEN≠ ≠GOTO≠ KLEINER;
281                             ≠IF≠ I = PM.INDEX ≠THEN≠ ≠GOTO≠ GLEICH;
282                               PM :- PM.SUC;
283                     ≠END≠ WHILE PM;
284                     PM :- ≠NEW≠ PARTITIONSMENGE;  PM.INDEX := I;
285                     PM.INTO(P);
286                     ≠GOTO≠ GLEICH;
287                     ≠NEW≠ PARTITIONSMENGE.PRECEDE(PM);
288 KLEINER..           PM :- PM.PRED;    PM.INDEX := I;
289                     WO :- PM;
290 GLEICH..
```

```
291           ≠END≠ W0;
292
293
294           ≠REF≠ (KK) K,K1;
295           ≠REF≠(ILISTE) L;              L:- ≠NEW≠ ILISTE;
296
297         ≠FOR≠I:=1≠STEP≠1≠UNTIL≠N≠DO≠≠FOR≠J:=1≠STEP≠1≠UNTIL≠N≠DO≠
298             ≠IF≠ MAX < F[I,J] ≠THEN≠ MAX := F[I,J]
299          ≠ELSE≠ ≠IF≠ F[I,J] < 0 ≠THEN≠ ABORT(≠≠LABEL NEGATIV≠≠);
300           MAX := MAX + 10;
301
302           P :- KNOTP;
303           ≠FOR≠ I := 1 ≠STEP≠ 1 ≠UNTIL≠ N ≠DO≠
304           ≠BEGIN≠  K :- ≠NEW≠ KNOTEN(I);
305                    K.PM :- W0(F[I,I]);
306                    KF[I,1] :- K;
307           ≠END≠ FOR I;
308
309           P :- KANTP;
310       ≠FOR≠I:=1≠STEP≠1≠UNTIL≠N≠DO≠≠FOR≠J:=I+1≠STEP≠1≠UNTIL≠N≠DO≠
311      ≠BEGIN≠ K:- ≠NEW≠ KANTE(I,J);          K1 :- ≠NEW≠ KANTE(J,I);
312                K≠QUA≠KANTE.1 :- KT;            KT≠QUA≠KANTE.T :- K;
313                    F1 := F[I,J];                 F2 := F[J,I];
314               K.PM :- W0(F1*MAX + F2);     K1.PM :- W0(F2*MAX+F1);
315                    KF[I,J] :- K;                 KF[J,I] :- K1;
316           ≠END≠ FOR I,J;
317
318           ≠IF≠ GRAPH1 =/= ≠NONE≠ ≠THEN≠ ≠GOTO≠ M2;
319         ≠FOR≠I:=1≠STEP≠1≠UNTIL≠N≠DO≠≠FOR≠J:=1≠STEP≠1≠UNTIL≠N≠DO≠
320             ≠INSPECT≠ KF[I,J] ≠DO≠ INTO(PM.KLISTE1);
321           ≠GOTO≠ E;
322 M2..      ≠FOR≠I:=1≠STEP≠1≠UNTIL≠N≠DO≠≠FOR≠J:=1≠STEP≠1≠UNTIL≠N≠D
                                                          O≠
323             ≠INSPECT≠ KF[I,J] ≠DO≠ INTO(PM.KLISTE2);
324 E..       ;
325 ≠END≠ GRAPH;
326
327
328 ≠REF≠(HEAD) SINGLES;
329
330
331
332 ≠REF≠(HEAD) UGL;     ≠REF≠(SUBGRAPH) SG;
333 LINK      ≠CLASS≠  SUBGRAPH;
334 ≠BEGIN≠  ≠REF≠(PARTITION) P;
335           ≠PROCEDURE≠ ISOTEST2;
336            ≠BEGIN≠
337                      ≠REF≠(PARTITIONSMENGE)PM,PN,PV;
338                     ≠REF≠(PARTITION) SAVP;      ≠INTEGER≠ N1;
339                       ≠REF≠(KNOTEN)K1,K2,K3;
340                     SAVP:-KNOTP;
341                     P.LETZTE ALT :- P.LAST;
342                     KNOTP :- P;
343           N1 := 4*P.CARD;
344           SETLU(0);
345
346 PM :- KNOTP . FIRST;
347
348 ≠WHILE≠ PM =/= ≠NONE≠ ≠DO≠
349 ≠BEGIN≠
```

```
350         ≠IF≠ GRAPH1.G(PM.KLISTE1.FIRST≠QUA≠KNOTEN).LAENGE  =  N1
351             ≠THEN≠ ≠GOTO≠ ISO2;
352           PM :- PM.SUC;
353 ≠END≠ WHILE PM;
354
355 WEISS NICHT..
356 OUTIMAGE; OUTTEXT(≠≠KEINE AUSSAGE UEBER ISOMORPHISMUS≠≠);
357 OUTIMAGE;  OUTTEXT(≠≠RESULTIERENDE KNOTENPARTITION: ≠≠);
358           AUSGABE(KNOTP);
359           ≠GOTO≠ E ISOTEST2;
360
361 ISO2..
362  OUTIMAGE; OUTINT(PM.KLISTE1.CARDINAL,2);
363 OUTTEXT(≠≠  ISOMORPHISMEN MOEGLICH; EINER DAVON:    ≠≠);
364
365 K1 :- PM.KLISTE1.FIRST; K2 :- pM.KLISTE2.FIRST;
366 X1 :- K1.X;              X2 := K2.X;
367 PN :- KNOTP.FIRST;                ≠WHILE≠ PN =/= ≠NONE≠ ≠DO≠
368 ≠BEGIN≠ K1 :- PN.KLISTE1.FIRST;    K2 :- PN.KLISTE2.FIRST;
369           ≠WHILE≠ K1 =/= ≠NONE≠ ≠DO≠
370           ≠BEGIN≠  PV :- GRAPH1.KF[X1,K1.X].PM;          K3 :- K2;
371            ≠WHILE≠ GRAPH2.KF[X2,K3.X].PM=/=PV ≠DO≠ K3 :- K3.SUC;
372                    ≠IF≠ K3 == K2 ≠THEN≠ K2 :- K2.SUC
373                    ≠ELSE≠ K3.PRECEDE(K2);
374                    K1 :- K1.SUC;
375           ≠END≠ WHILE K1;
376           PN :- PN.SUC;
377 ≠END≠ WHILE PN;
378
379 AUSGABE(KNOTP);
380 E ISOTEST2..      KNOTP :- SAVP;
381           ≠END≠ ISOTEST2;
382           P :- ≠NEW≠ PARTITION;
383 ≠END≠ SUBGRAPH;
384
385 ≠PROCEDURE≠ DECOMPOSE(KNOTP)INTO LIST OF COMPONENTS:(L);
386 ≠REF≠(PARTITION)KNOTP;     ≠REF≠(HEAD) L;
387 ≠BEGIN≠
388         ≠REF≠(KNOTEN) K;          ≠REF≠(PARTITIONSMENGE) P1,P2,P3;
389           ≠REF≠(SUBGRAPH)SG;          ≠INTEGER≠ X;
390           ≠FOR≠ P1 :- KNOTP.FIRST ≠WHILE≠ P1 =/= ≠NONE≠ ≠DO≠
391           ≠BEGIN≠
392           SG :- ≠NEW≠ SUBGRAPH; SG.INTO(L);   P1.INTO(SG.P);
393           ≠WHILE≠ P1 =/= ≠NONE≠ ≠DO≠
394           ≠BEGIN≠  K :- P1.KLISTE1.FIRST;      X:=K.X;
395                    P2 :- KNOTP.FIRST; ≠WHILE≠ P2 =/= ≠NONE≠ ≠DO≠
396                    ≠BEGIN≠  K :- P2.KLISTE1.FIRST;
397                             P3 :- GRAPH1.KF[X,K.X].PM;
398                             ≠FOR≠K:-K.SUC≠WHILE≠K=/=≠NONE≠ ≠DO≠
399                             ≠IF≠ P3 =/= GRAPH1.KF[X,K.X].PM
400                             ≠THEN≠ ≠BEGIN≠ P3 :- P2.SUC;
401                                      P2.INTO(SG.P);   P2 :- P3;
402                                    ≠GOTO≠ E1;
403                             ≠END≠THEN;
404                             P2 :- P2.SUC;
405                    E1..   ;  ≠END≠ WHILE P2;
406                    P1 :- P1.SUC;
407           ≠END≠ WHILE P1;
408           ≠END≠ FOR P1;
409 ≠END≠ DECOMPOSE;
```

```
410
411 ≠REF≠( PARTITION) KNOTP,KANTP;
412 HEAD≠CLASS≠ PARTITION;
413 ≠BEGIN≠ ≠REF≠(PARTITIONSMENGE) LETZTE ALT;
414          ≠BOOLEAN≠ NEU, NON ISOMORPH;
415          ≠INTEGER≠ ≠PROCEDURE≠ CARD;
416          ≠BEGIN≠  ≠REF≠(PARTITIONSMENGE)PM; ≠INTEGER≠ C;
417                   PM :- FIRST;
418              ≠WHILE≠PM=/=≠NONE≠ ≠DO≠ ≠BEGIN≠ C:= C + PM.KLISTE1
419                    .CARDINAL ;  PM :- PM.SUC; ≠END≠; CARD := C;
420          ≠END≠ CARDINAL;
421 ≠END≠PARTITION;
422
423 PARTITION ≠CLASS≠ KNOTENPARTITION;
424 ≠BEGIN≠ ≠REF≠(HEAD) RESERVE;
425          ≠REF≠(PARTITIONSMENGE) ≠PROCEDURE≠ NEUKNOTENPM(L);
426                   ≠REF≠ (ILISTE) L;
427
428          ≠BEGIN≠  ≠REF≠(PARTITIONSMENGE) PM;
429           PM :- ≠IF≠ RESERVE.EMPTY ≠THEN≠ ≠NEW≠ PARTITIONSMENGE
430                   ≠ELSE≠ RESERVE.FIRST;
431                   ≠INSPECT≠ PM ≠DO≠
432                   ≠BEGIN≠ ≠IF≠ L == ≠NONE≠ ≠THEN≠ ≠GOTO≠ NO;
433                   LISTE.I1 := L.I1;       LISTE.I2 := L.I2;
434        NO..      ALT := GANZ ALT := ≠FALSE≠;   ≠END≠ INSPECT PM;
435                   NEUKNOTENPM :- PM;
436          ≠END≠ NEUKNOTENPM;
437
438          ≠PROCEDURE≠ ENTFERNE(PN);
439                   ≠REF≠(PARTITIONSMENGE) PN;
440
441        ≠BEGIN≠  ≠REF≠(PARTITIONSMENGE) PV; ≠REF≠(KNOTEN) K1,K2;
442                   ≠INTEGER≠ X1,X2;
443                            ≠INSPECT≠ PN ≠DO≠
444                            ≠BEGIN≠  INTO(SINGLES);
445                                     K1 :- KLISTE1.FIRST;
446                                     K2 :- KLISTE2.FIRST;
447                                  ≠IF≠ PRED == ≠NONE≠ ≠THEN≠ INDEX := 1
448                                   ≠ELSE≠  INDEX := PRED≠QUA≠PARTITIONS
449                                              MENGE.INDEX + 1 ;
450                                   ≠END≠     INSPECT PN;
451                                 X1 := K1.X;  X2 := K2.X;  PV :- FIRST;
452 E2..                              ≠IF≠ PV == ≠NONE≠ ≠THEN≠ ≠GOTO≠ E2;
453                                   K1 :- PV.KLISTE1.FIRST;
454                                   K2 :- PV.KLISTE2.FIRST;
455                                   ≠WHILE≠ K1 =/= ≠NONE≠ ≠DO≠
456                                   ≠BEGIN≠  ≠INSPECT≠ GRAPH1 ≠DO≠
457                            ≠BEGIN≠  ≠INSPECT≠ KF [X1,K1.X]≠WHEN≠KANTE≠DO≠
458                                        ≠BEGIN≠  OUT; PM :- ≠NONE≠ ≠END≠;
459                                     ≠INSPECT≠ KF [K1.X,X1]≠WHEN≠KANTE≠DO≠
460                                        ≠BEGIN≠  OUT; PM :- ≠NONE≠ ≠END≠;
461                                                 KF [X1,K1.X] :- ≠NONE≠;
462                                                 KF [K1.X,X1] :- ≠NONE≠;
463                                   ≠END≠;
464                                   ≠INSPECT≠ GRAPH2 ≠DO≠
465                            ≠BEGIN≠  ≠INSPECT≠ KF [X2,K2.X]≠WHEN≠KANTE≠DO≠
466                                        ≠BEGIN≠  OUT; PM :- ≠NONE≠ ≠END≠;
467                                     ≠INSPECT≠ KF [K2.X,X2]≠WHEN≠KANTE≠DO≠
468                                        ≠BEGIN≠  OUT; PM :- ≠NONE≠ ≠END≠;
469                                                 KF [K2.X,X2] :- ≠NONE≠;
```

```
                                        FF [X2,K2.X] :- ≠NONE≠;
                              ≠END≠;
                                   K1 :- K1.SUC;      K2 :- K2.SUC;
                              ≠END≠     WHILE K1 =/= NONE;
                              PV :- PV.SUC;      ≠GOTO≠ B2;
E2..          ;
        ≠END≠ ENTFERNE;

        ≠REF≠(PARTITIONSMENGE) PA,PN,PV,LETZTE NEU;
        ≠REF≠ (ILISTE) L;
        ≠INTEGER≠ I,IND;              ≠REF≠ (KNOTEN) K1,K2;
        ≠REAL≠ ZEIT;

        SINGLES  :- ≠NEW≠ HEAD;
        RESERVE :- ≠NEW≠ HEAD;
        DETACH;
WEITER1..
        KNI := ≠TRUE≠;
        ZEIT := ELAPSED;
        LETZTE ALT :- LAST;
        PA :- FIRST;
        ≠WHILE≠ PA =/= ≠NONE≠ ≠DO≠  ≠INSPECT≠ PA ≠DO≠
        ≠BEGIN≠  ALT := GANZ ALT;
                 GANZ ALT := ≠TRUE≠;
                 PA :- SUC;
        ≠END≠ WHILE PA =/= NONE;

WEITER..
        OUTCHAR(5*6);
        I := KANTP.LAST≠QUA≠PARTITIONSMENGE.INDEX;
        LETZTE ALT :- LAST;
        PA :- FIRST;

B1..    K1 :- PA.KLISTE1.FIRST;
        K2 :- PA.KLISTE2.FIRST;
        LETZTE NEU :- LAST;
        SETLU(0);
                              ≠WHILE≠ K1 =/= ≠NONE≠ ≠DO≠
                              ≠BEGIN≠ L :- GRAPH1.G(K1);
                                       PN :- LETZTE NEU  ;
                                       ≠FOR≠ PN :- PN.SUC ≠WHILE≠
                                          PN =/= ≠NONE≠ ≠DO≠
                              ≠BEGIN≠   IND := COMPARE(L,PN.LISTE);

                                                ≠IF≠ IND > 0 ≠THEN≠
                                                   ≠GOTO≠ MAGGICRE;
                                                ≠IF≠ IND = 0 ≠THEN≠
                       ≠BEGIN≠ SETLU(L.I1-1); ≠GOTO≠ UGUALE; ≠END≠;
                                       ≠END≠     FOR PN :- PN.SUC;
                                       PN :- NEUKNOTENPM(L);
                                    PN.INTO(≠THIS≠KNOTENPARTITION);
                                       ≠GOTO≠ UGUALE;
                       MAGGICRE..      NEUKNOTENPM(L).PRECEDE(PN);
                                       PN :- PN.PRED;
                       UGUALE..        K1.PM :- PN;
                                       K1 :- K1.SUC;
                              ≠END≠     WHILE K1 =/= NONE;

                              ≠WHILE≠ K2 =/= ≠NONE≠ ≠DO≠
                              ≠BEGIN≠ L :- GRAPH2.G(K2);
```

```
                                        PN :- LETZTE NEU;
                                    ≠FOR≠ PN :- PN.SUC ≠WHILE≠ PN =/=
                                        ≠NONE≠ ≠DO≠
                                ≠BEGIN≠  IND := COMPARE(L,PN.LISTE);

                                              ≠IF≠ IND > 0 ≠THEN≠
                                            ≠GOTO≠ NICHT ISOMORPH;
                                              ≠IF≠ IND = 0 ≠THEN≠
                        ≠BEGIN≠ SETLU(L,I1-1); ≠GOTO≠ IGUAL ; ≠END≠;
                       ≠END≠    FOR PN:-PN.SUC; ≠GOTO≠NICHTISOMORPH;
                       IGUAL..          K2.PM :- PN;
                                        K2 :- K2.SUC;
                                ≠END≠   WHILE K2 =/= NONE;

                                ≠IF≠ LETZTE NEU.SUC == LAST ≠THEN≠
                            ≠BEGIN≠ ≠INSPECT≠ LAST≠QUA≠PARTITIONSMENGE
                                ≠DO≠ ≠BEGIN≠    ALT := ≠TRUE≠;
                                        GANZ ALT := PA.GANZ ALT;

                                ≠END≠;   ≠END≠    THEN ;


                                PN :- LETZTE NEU;
                        ≠FOR≠ PN :- PN.SUC ≠WHILE≠ PN =/= ≠NONE≠ ≠DO≠
                                ≠INSPECT≠ PN ≠DO≠
                                ≠BEGIN≠  INDEX := I := I+1;

                                ≠END≠    INSPECT PN;
    ≠IF≠PA=/=LETZTE ALT≠THEN≠ ≠BEGIN≠PA:-PA.SUC; ≠GOTO≠B1≠END≠;
B3..    PA :- FIRST;
        K1 :- PA.KLISTE1.FIRST;
        K2 :- PA.KLISTE2.FIRST;
        ≠WHILE≠ K1 =/= ≠NONE≠ ≠DO≠
        ≠BEGIN≠  ≠INSPECT≠ K1 ≠DO≠ INTO(PM.KLISTE1);
                 ≠INSPECT≠ K2 ≠DO≠ INTO(PM.KLISTE2);
                 K1 :- PA.KLISTE1.FIRST;
                 K2 :- PA.KLISTE2.FIRST;
        ≠END≠ WHILE K1 =/= NONE;
        PA.INTO(RESERVE);
        ≠IF≠ PA =/= LETZTE ALT ≠THEN≠ ≠GOTO≠ B3;

        NEU := ≠FALSE≠;
        PN :- FIRST;
        ≠WHILE≠ PN =/= ≠NONE≠ ≠DO≠
        ≠INSPECT≠ PN ≠DO≠
        ≠IF≠ KLISTE1.CARDINAL ≠NE≠ KLISTE2.CARDINAL
        ≠THEN≠ ≠GOTO≠ NICHT ISOMORPH
        ≠ELSE≠   ≠BEGIN≠  PN :- SUC;
                          ≠IF≠ ¬ALT ≠THEN≠ NEU := ≠TRUE≠
                     ≠ELSE≠ ≠IF≠ KLISTE1.FIRST== KLISTE1.LAST
                       ≠THEN≠ENTFERNE(≠THIS≠PARTITIONSMENGE);
                 ≠END≠    ELSE;

≠IF≠ ZEITLIMIT < E LAPSED ≠THEN≠ ABORT(≠≠ZEITLIMIT≠≠);
        KN1 := ≠FALSE≠;
        ≠IF≠ NEU ∧ ¬EMPTY ≠THEN≠ ≠GOTO≠ WEITER;
        ZEIT :=(ELAPSED-ZEIT)/100.0;
        OUTFRAC(ZEIT,1,≠IF≠ZEIT<10.0≠THEN≠3≠ELSE≠TRUNK(LN(ZEIT
                                                )/LN(10.0)
```

```
589             )+2);
590          LETZTE ALT :- LAST;
591          DETACH;
592
593
594          ≠GOTO≠ WEITER1;
595
596 NICHT ISOMORPH..
597          NON ISOMORPH := ≠TRUE≠;
598 ≠END≠    KNOTENPARTITION;
599
600
601 PARTITION ≠CLASS≠ KANTENPARTITION;
602 ≠BEGIN≠
603
604          ≠REF≠(HEAD)RESERVE;
605          ≠REF≠ (PARTITIONSMENGE) ≠PROCEDURE≠ NEUKANTENPM(L);
606          ≠REF≠(ILISTE) L;
607
608          ≠BEGIN≠  ≠REF≠(PARTITIONSMENGE) PM;
609           PM :- ≠IF≠ RESERVE.EMPTY ≠THEN≠ ≠NEW≠ PARTITIONSMENGE
610                    ≠ELSE≠ RESERVE.FIRST;
611                    ≠INSPECT≠ PM ≠DO≠
612                    ≠BEGIN≠ ≠IF≠ L == ≠NONE≠ ≠THEN≠ ≠GOTO≠ NO;
613                    LISTE.I1 := L.I1;  LISTE.I2 := L.I2;
614          NO..      ALT := ≠FALSE≠;      ≠IF≠ KLIST E1 == ≠NONE≠
615                           ≠THEN≠    ≠BEGIN≠ KLISTE1 :-≠NEW≠HEAD;
616                                             KLISTE2 :-≠NEW≠HEAD;
617                                               ≠END≠  THEN;
618                    ≠END≠ INSPECT PM;
619                    NEUKANTENPM :-  PM;
620          ≠END≠ NEU KANTENPM;
621          ≠BOOLEAN≠ SYM;
622          ≠REF≠(PARTITIONSMENGE) PA,PN,PV,LETZTE NEU;
623          ≠REF≠(KANTE) K1,K2;
624          ≠REF≠(ILISTE) L;            ≠INTEGER≠ I, IND;
625          ≠REAL≠ ZEIT;
626
627
628          RESERVE :- ≠NEW≠ HEAD;
629          DETACH;
630 WEITER ..
631          ZEIT := ELAPSED;
632          I := FIRST≠QUA≠PARTITIONSMENGE.INDEX - 1;
633          NEU := ≠FALSE≠;
634          LETZTE ALT :- LAST;
635          PA :- FIRST;
636
637 B1..     LETZTE NEU :- LAST;
638          SETLU(0);
639          ≠IF≠ PA.KLISTE1.EMPTY ≠THEN≠ ≠GOTO≠ E1;
640          K1 :- PA.KLISTE1.FIRST;
641          SYM :=              K1.T.PM == PA    ;
642         ≠FOR≠ K1 :- PA.KLISTE1.FIRST ≠WHILE≠ K1 =/= ≠NONE≠ ≠DO≠
643          ≠BEGIN≠  L :- GRAPH1.G(K1);
644                   PN :- LETZTE NEU;
645                   ≠FOR≠ PN :- PN.SUC ≠WHILE≠ PN =/= ≠NONE≠ ≠DO≠
646                            ≠BEGIN≠  IND := COMPARE(L,PN.LISTE);
647                                              ≠IF≠ IND > 0 ≠THEN≠
648                            ≠GOTO≠ MAGGIORE;
```

```
649                                                  ≠IF≠ IND = 0 ≠THEN≠
650                    ≠BEGIN≠ SETLU(L.I1-1); ≠GOTO≠ UGUALE; ≠END≠;
651                  ≠END≠ FOR PN .- PN.SUC;
652                  PN :- NEUKANTENPM(L);
653                  PN.INTO(≠THIS≠ KANTENPARTITION);
654                  ≠GOTO≠ UGUALE;
655 MAGGIORE..       NEUKANTENPM(L).PRECEDE(PN);
656                  PN :- PN.PRED;
657 UGUALE..         K1.INTO(PN.KLISTE1);
658 ≠END≠ FOR K1;
659
660        ≠FOR≠ K2 :- PA.KLISTE2.FIRST ≠WHILE≠ K2 =/= ≠NONE≠ ≠DO≠
661         ≠BEGIN≠  L :- GRAPH2.G(K2);
662                  PN :- LETZTE NEU;
663                  ≠FOR≠ PN :- PN.SUC ≠WHILE≠ PN =/= ≠NONE≠ ≠DO≠
664                           ≠BEGIN≠  IND := COMPARE(L,PN.LISTE);
665                                                  ≠IF≠ IND > 0 ≠THEN≠
666                           ≠GOTO≠ NICHT ISOMORPH;
667                                                  ≠IF≠ IND = 0 ≠THEN≠
668                    ≠BEGIN≠ SETLU(L.I1-1); ≠GOTO≠ IGUAL ; ≠END≠;
669                  ≠END≠    FOR PN :- PN.SUC;
670                  ≠GOTO≠ NICHT ISOMORPH;
671 IGUAL..          K2.INTO(PN.KLISTE2);
672 ≠END≠ FOR K2;
673
674         PN :- LAST;
675         ≠IF≠ PN == LETZTE NEU.SUC
676         ≠THEN≠   ≠BEGIN≠  ≠INSPECT≠ PN ≠DO≠
677                           ≠BEGIN≠  ALT := ≠TRUE≠;
678
679                           ≠END≠    INSPECT PN;
680                           ≠IF≠ SYM ≠THEN≠ ≠GOTO≠ SYM1;
681
682                           PN :- NEUKANTENPM(≠NONE≠);
683                           PN.INTO(≠THIS≠  KANTENPARTITION);
684                           PA :- PA.SUC;
685                    ≠COMMENT≠ D.H. DIE ZUGEHOERIGE,TRANSPONIERTE
686                                    PARTITIONSMENGE;
687                           ≠INSPECT≠ PN ≠DO≠
688                           ≠BEGIN≠  KLISTE1 :- PA.KLISTE1;
689                                    KLISTE2 :- PA.KLISTE2;
690
691                                    ALT := ≠TRUE≠;
692                           ≠END≠    INSPECT PN;
693                     ≠INSPECT≠ PA ≠DO≠ KLISTE1:-KLISTE2:-≠NONE≠;
694 SYM1..   ;        ≠END≠ THEN
695
696         ≠ELSE≠   ≠BEGIN≠  NEU := ≠TRUE≠;
697 ≠IF≠ ZEITLIMIT < E LAPSED ≠THEN≠ ABORT(≠≠ZEITLIMIT≠≠);
698
699
700                           PV :- LETZTE NEU;
701                  ≠FOR≠ PV :- PV.SUC ≠WHILE≠ PV =/= ≠NONE≠ ≠DO≠
702                           ≠INSPECT≠ PV ≠DO≠
703                       ≠IF≠ KLISTE1.CARDINAL = KLISTE2.CARDINAL
704                           ≠THEN≠
705                           ≠ELSE≠   ≠GOTO≠ NICHT ISOMORPH;
706                           ≠IF≠ SYM ≠THEN≠ ≠GOTO≠ SYM2;
707
708                           PA :- PA.SUC;
```

```
709
710                                 PV :- LETZTE NEU;
711                         ≠FOR≠ PV :- PV.SUC ≠WHILE≠ PV =/= ≠NONE≠ ≠DO≠
712                                 ≠INSPECT≠ NEUKANTENPM(≠NONE≠) ≠DO≠
713                                 ≠BEGIN≠
714                                           K1 :- PV.KLISTE1.FIRST;
715                                           K2 :- PV.KLISTE2.FIRST;
716                                           ≠WHILE≠ K1 =/= ≠NONE≠ ≠DO≠
717 ≠BEGIN≠ K1.T.INTO(KLISTE1);
718         K2.T.INTO(KLISTE2);
719         K1  :- K1.SUC;                    K2 :- K2.SUC;
720 ≠END≠   WHILE K1 =/= NONE;
721                                           FOLLOW(PV);
722                                         PV :- ≠THIS≠ PARTITIONSMENGE;
723                                 ≠END≠     INSPECT NEUKANTENPM;
724 SYM2..   ;          ≠END≠     ELSE;
725
726 E1..    ≠IF≠ PA =/= LETZTE ALT
727         ≠THEN≠ ≠BEGIN≠ PA :- PA.SUC; ≠GOTO≠ B1    ≠END≠;
728
729 B2..    PA :- FIRST;
730         PA.INTO(RESERVE);
731         ≠IF≠ PA =/= LETZTE ALT ≠THEN≠ ≠GOTO≠ B2;
732
733         PN :- FIRST;
734         ≠WHILE≠ PN =/= ≠NONE≠ ≠DO≠
735         ≠BEGIN≠  K1 :- PN.KLISTE1.FIRST;
736                  K2 :- PN.KLISTE2.FIRST;
737                  ≠WHILE≠ K1 =/= ≠NONE≠ ≠DO≠
738                  ≠BEGIN≠  K1.PM :- PN;    K1 :- K1.SUC;
739                           K2.PM :- PN;    K2 :- K2.SUC;
740                  ≠END≠ WHILE K1 =/= NONE;
741                  PN :- PN.SUC;
742         ≠END≠    WHILE PN =/= NONE;
743
744         PN :- FIRST;       ≠WHILE≠ PN =/= ≠NONE≠ ≠DO≠
745         ≠BEGIN≠  PN.INDEX := I := I+1;
746                  PV :- PN.KLISTE1.FIRST≠QUA≠KANTE.T.PM;
747                  ≠IF≠ PV =/= PN ≠THEN≠
748              ≠BEGIN≠  ≠IF≠ PV =/= PN.SUC ≠THEN≠ PV.FOLLOW(PN);
749                           PV.INDEX := I := I+1;
750                           PN :- PV;
751                  ≠END≠ THEN;
752                  PN :- PN.S UC;
753         ≠END≠ WHILE PN;
754         ZEIT :=(ELAPSED-ZEIT)/100.0;      OUTCHAR($/$);
755         OUTFRAC(ZEIT,1,≠IF≠ZEIT<10.0≠THEN≠3≠ELSE≠TRUNK(LN(ZEIT
                                                    )/LN(10.0)
756            )+2);
757         DETACH;
758         ≠GOTO≠ WEITER;
759
760 NICHT ISOMORPH..
761         NON ISOMORPH := ≠TRUE≠;
762 ≠END≠   KANTENPARTITION;
763
764
765
766 ≠REF≠ (PARTITIONSMENGE) PM,PN,PV;          ≠REF≠ (KNOTEN) K1,
                                                K2,K3;
```

```
767 ≠INTEGER≠ X1,X2,I,N1                                                    ;
768 ≠BOOLEAN≠ KN1;
769 ≠INTEGER≠ ≠ARRAY≠  FELD1,FELD2[1..N,1..N];
770 ≠INTEGER≠ ≠ARRAY≠ PERM[1..N];
771 ≠REF≠(HEAD) QLISTE;
772 QLISTE :- ≠NEW≠ QUADRUPELLISTE;
773
774 ≠IF≠ ¬ ENDIMAGE ≠THEN≠ ≠BEGIN≠ ≠INSPECT≠ SYSIN ≠DO≠
775 ≠IF≠  IMAGE.SUB(POS,1).GETCHAR = $*$ ≠THEN≠
776   STARRED POLYGON(FELD1,N,SYMMETRISCH)
777 ≠ELSE≠ ZUFALLSGRAPH(FELD1,N,  X,SYMMETRISCH,SCHLEIFEN); ≠END≠
                                                            ≠ELSE≠
778 ≠IF≠ EINLESEN ≠THEN≠ EINLESENG(FELD1,N,SYMMETRISCH)                    ;
779 !!!IMAGE;  ≠IF≠ SYSIN.IMAGE.SUB(1,2) = ≠≠ID≠≠
780 ≠THEN≠ ≠BEGIN≠ ZUFALLSPERMUTATION(PERM,N);    PERMUTIERE(FELD1,
                                                            PERM,
781                       FELD2,N)  ≠END≠
782 ≠ELSE≠
783 ≠IF≠ ¬ ENDIMAGE ≠THEN≠ ≠BEGIN≠ ≠INSPECT≠ SYSIN ≠DO≠
784 ≠IF≠ IMAGE.SUB(POS,1).GETCHAR = $*$ ≠THEN≠
785   STARRED POLYGON(FELD2,N,SYMMETRISCH)
786 ≠ELSE≠ ZUFALLSGRAPH(FELD2,N,  X,SYMMETRISCH,SCHLEIFEN); ≠END≠
                                                            ≠ELSE≠
787 ≠IF≠ EINLESEN ≠THEN≠ EINLESENG(FELD2,N,SYMMETRISCH)                    ;
788 SYSIN.SETPOS(80);
789
790 KNOTP :- ≠NEW≠ KNOTENPARTITION;    KANTP :- ≠NEW≠ KANTENPARTIT
                                                            ION;
791
792 GRAPH1 :- ≠NEW≠ GRAPH(FELD1,N);    GRAPH2 :- ≠NEW≠ GRAPH(FELD2
                                                            ,N);
793
794 ≠FOR≠ PM :- KNOTP.FIRST, KANTP.FIRST ≠DO≠≠WHILE≠PM=/=≠NONE≠≠DO
                                                            ≠
795  ≠IF≠ PM.KLISTE1.CARDINAL ≠NE≠ PM.KLISTE2.CARDINAL
796  ≠THEN≠ ≠GOTO≠ NICHT ISO
797  ≠ELSE≠ PM :- PM.SUC;
798
799 PM :- KANTP.FIRST;
800 ≠WHILE≠ PM =/= ≠NONE≠ ≠DO≠
801 ≠BEGIN≠ PN :- PM.KLISTE1.FIRST≠QUA≠KANTE.T.PM;
802          ≠IF≠ PN =/= PM ≠THEN≠ PN.FOLLOW(PM);
803          PM :- PN.SUC;
804 ≠END≠ WHILE PM;
805
806
807 I := 0;
808 ≠FOR≠ PM :- KANTP.FIRST, KNOTP.FIRST≠DO≠≠WHILE≠PM=/=≠NONE≠ ≠DO
                                                            ≠
809  ≠INSPECT≠ PM ≠DO≠
810 ≠BEGIN≠
811          INDEX :- I := I+1;
812          PM :- SUC;
813 ≠END≠ INSPECT PM;
814
815
816                                                         ≠COMMENT≠
817
818      *****  START KNOTENPARTITION  *****                          ;
819
```

```
820 W.. CALL(KNOTP);
821 ≠IF≠ KNOTP.NON ISOMORPH ≠THEN≠ ≠GOTO≠ NICHT ISO;
822 ≠IF≠ KNOTP.CARDINAL + SINGLES.CARDINAL = N ≠THEN≠ ≠GOTO≠ ISO1;
823
824                                                          ≠COMMENT≠
825
826     *****  START DER KANTENPARTITION  *****                        ;
827
828 CALL(KANTP);
829 ≠IF≠ KANTP.NON ISOMORPH ≠THEN≠ ≠GOTO≠ NICHT ISO;
830 ≠IF≠ KANTP .NEU ≠THEN≠ ≠GOTO≠ W;
831
832 ≠FOR≠ PM :- KNOTP.FIRST, KANTP.FIRST ≠DO≠≠WHILE≠ PM =/= ≠NONE≠
                                                          ≠DO≠
833  ≠INSPECT≠ PM ≠DO≠
834 ≠BEGIN≠ GANZ ALT := ALT := ≠FALSE≠; PM :- SUC          ≠END≠;
835
836
837 UGL :- ≠NEW≠ HEAD;
838 DECOMPOSE(KNOTP,UGL);
839 ≠IF≠ UGL.CARDINAL = 1 ≠THEN≠ ≠GOTO≠ NZERL;
840 OUTIMAGE; OUTTEXT(≠≠ZERLEGUNG IN UNABHENGIGE UNTERGRAPHEN: ≠≠)
                                                          ;
841 NZERL..
842 SG :- UGL.FIRST;    ≠WHILE≠ SG =/= ≠NONE≠ ≠DO≠
843      ≠IF≠ (≠IF≠SG.P.FIRST.SUC == ≠NONE≠ ≠THEN≠ SG.P.FIRST≠QUA≠
844                PARTITIONSMENGE.COMPLETE ≠ELSE≠ ≠FALSE≠ ) ≠THEN≠
845                   ≠BEGIN≠  OUTIMAGE; OUTIMAGE;
846                      OUTTEXT(≠≠VOLLSTAENDIG:≠≠); AUSGABE(SG.P);
847                   ≠END≠THEN    ≠ELSE≠
848 ≠BEGIN≠
849          OUTIMAGE; OUTIMAGE;
850          SG.ISOTEST2;          SG :- SG.SUC;
851 ≠END≠ WHILE SG;
852 ≠IF≠ SINGLES.EMPTY ≠THEN≠ ≠GOTO≠ ENDE;
853 OUTIMAGE; OUTTEXT(≠≠FIXPUNKTE: ≠≠);          AUSGABE(SINGLES);
854 ≠GOTO≠ ENDE;
855
856 ISO1..
857 OUTIMAGE; OUTTEXT(≠≠EINZIGER ISOMORPHISMUS: ≠≠);
858 AUSGABE(SINGLES);  AUSGABE(KNOTP);
859 ≠GOTO≠ ENDE;
860 NICHT ISO..OUTIMAGE;OUTTEXT(≠≠DIE BEIDEN GRAPHEN SIND NICHT IS
                                                          OMORPH≠≠);
861 OUTIMAGE;
862 ENDE..    ;
863 ≠END≠;
864 ≠IF≠ ¬ LASTITEM ≠THEN≠ ≠GOTO≠ ANF;
865 ≠END≠;
866           FINIS
```

Beispiele, die mit diesem Programm gerechnet wurden.

In der jeweils mittleren Zeile steht ein Stern für eine Durchführung von A_1, ein Schrägstrich für eine Durchführung von A_2. Dazwischen stehen die benötigten Rechenzeiten in Sekunden.
Man sieht, daß bei "unregelmäßigen" Graphen gewöhnlich die Anwendung von A_1 ausreicht.

Die Knotenpartition, die durch f erzeugt wird, ist in der Form $M_j:M'_j$; ausgedruckt.

Die Schreibweise [11,12,15,16] : [12,18,16,15] für einen Isomorphismus p ist folgendermaßen zu lesen:
p(11)=12, p(12)=18 und so weiter der Reihe nach.

Graph von Beispiel 4.12:

```
 N= 18  SYMMETRISCHER GRAPH, EINGELESEN: >>>>>>
1: 2,3,4,5,6,7,8,9*    2: 3,8,9,10,11,12,17*    3:  4,7,10,11,12,13*
4:  5,6,11,12,13,14*    5:  6,9,12,13,14,15*    6:  7,13,14,15,16*
7: 8,14,15,16,17*       8:  9,10,15,16,17*      9:  10,11,16,17*
10:  11,14,17,18*       11:   12,15,18*   12:  13,16,18*   13:  14,17,
18*
14:  15,18*   15:  16,18*   16:  17,18*      17:  18*


ZUFALLSPERMUTATION, U=105
  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
 17  9 14  4 11  6  3 13  2  5 18 12  8  1 15 16 10  7
*0.4/8.5***1.5/8.1*0.5/5.5


 4  ISOMORPHISMEN MOEGLICH; EINER DAVON:
[11,12,15,16]:[12,18,16,15]; [ 3, 7]:[14, 3]; [ 5, 9]:[ 2,11]; [10,
13,14,17]:[ 8, 5,10, 1]; [ 2, 4, 6, 8]:[ 4, 9,13, 6];
FIXPUNKTE:
[ 1]:[17]; [18]:[ 7];
```

Graph von Beispiel 4.13:

Hier ist ein Graph $K \in H(\Delta, G_1, .., G_k)$, dessen Grad die Knoten 12 und 13 auszeichnen soll, nicht leicht zu finden. Auch die Knoten-Quotientengraphen von $[1]_K[2]$ können die Knoten 12, 13 nicht von den Knoten 9,1o,11,14,15,16 unterscheiden.

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
 N= 16  SYMMETRISCHER GRAPH, EINGELESEN: >>>>>>
1:9,10,11,12*2:9,10,11,13*3:9,12,14,15*4:10,12,14,16*5:11,12,15,16*
6:9,13,14,15*7:10,13,14,16*8:11,13,15,16*

ZUFALLSPERMUTATION, U=83
  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
  5  2  3  1 14  6  7  8 16  4 11 10 12 13 15  9
*0,3/6,6**0,6/6,2**0,6/4,6*0,3/3,1

KEINE AUSSAGE UEBER ISOMORPHISMUS
RESULTIERENDE KNOTENPARTITION:
[12,13]:[10,12]:  [ 1, 2, 3, 4, 5, 6, 7, 8]:[ 1, 2, 3, 5, 6, 7, 8,14]
[ 9,10,11,14,15,16]:[ 4, 9,11,13,15,16]:
```

Der folgende Graph besteht aus den beiden getrennten Komponenten 1 bis 8 und 9 bis 16.

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
 N= 16  SYMMETRISCHER GRAPH, EINGELESEN: >>>>>>
1:3,4,6,7*2:4,5,6,7*3:5,6,8*4:7,8*5:7,8*6:8*9:11,12,14,15*10:12,13,15,
16*
11:13,14,16*12:14,15,15*13:15,16*14:16*

ZUFALLSPERMUTATION, U=71
  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
  5 10  7  6  1 11  3  8  9  2 14 12 13  4 15 16
*0,3/6,3***1,1/7,0*0,4/4,6
ZERLEGUNG IN UNABHENGIGE UNTERGRAPHEN:

 4  ISOMORPHISMEN MOEGLICH; EINER DAVON:
[ 2, 4, 6, 8]:[ 6,10, 8,11]:  [ 1, 5]:[ 1, 5]:  [ 3, 7]:[ 7, 3]:

KEINE AUSSAGE UEBER ISOMORPHISMUS
RESULTIERENDE KNOTENPARTITION:
[ 9,10,11,12,13,14,15,16]:[ 2, 4, 9,12,13,14,15,16]:
```

```
-*-*-*
```

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
 N= 17 GRAPH, EINGELESEN:  STARRED POLYGON, SYMBOL=( 1, 3)
 STARRED POLYGON, SYMBOL=( 2, 6)
*0.3/6.8*0.3/6.1*0.4/4.2


17  ISOMORPHISMEN MOEGLICH; EINER DAVON:
[ 1, 2, 3, 4, 5, 6, 7, 8, 9,10,11,12,13,14,15,16,17]:[ 1, 3, 5, 7, 9,
11,13,15,17, 2, 4, 6, 8,10,12,14,16];
```

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
 N= 13 ZUFALLSGRAPH, KANTENHAEUFIGKEIT=0.5, U=          1111
ZUFALLSPERMUTATION, U=4605595
  1  2  3  4  5  6  7  8  9 10 11 12 13
 12  2  9  1  5  6 11  7 10  3 13  4  8
***).6
EINZIGER ISOMORPHISMUS:
[ 6]:[ 6];  [ 8]:[ 7];  [10]:[ 3];  [13]:[ 8];  [ 9]:[10];  [ 3]:[ 9];
[ 7]:[11];  [ 5]:[ 5];  [ 4]:[ 11];  [ 1]:[12];  [11]:[13];  [12]:[ 4];
[ 2]:[ 2];
```

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
 N= 13 ZUFALLSGRAPH, KANTENHAEUFIGKEIT=0.5, U=          3333
ZUFALLSPERMUTATION, U=1778361
  1  2  3  4  5  6  7  8  9 10 11 12 13
  1  2  3  4  5  6 10  8  9 12 11  7 13
**1.1
EINZIGER ISOMORPHISMUS:
[ 3]:[ 3];  [ 5]:[ 5];  [ 9]:[ 9];  [11]:[11];  [12]:[ 7];  [ 2]:[ 2];
[ 4]:[ 4];  [ 6]:[ 6];  [ 7]:[10];  [13]:[13];  [10]:[12];  [ 8]:[ 8];
[ 1]:[ 1];
```

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
 N= 13 ZUFALLSGRAPH, KANTENHAEUFIGKEIT=0.5, U=          5555
ZUFALLSPERMUTATION, U=7339335
  1  2  3  4  5  6  7  8  9 10 11 12 13
  1  2  8 11  5  6 13  3  9 10  4 12  7
***1.1
EINZIGER ISOMORPHISMUS:
[11]:[ 4];  [ 5]:[ 5];  [ 7]:[13];  [ 1]:[ 1];  [10]:[10];  [ 6]:[ 6];
[ 3]:[ 8];  [ 9]:[ 9];  [12]:[12];  [ 4]:[11];  [13]:[ 7];  [ 2]:[ 2];
[ 8]:[ 3];
```

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
 N= 18 ZUFALLSGRAPH, KANTENHAEUFIGKEIT=0.3, U=          4565
 1: 2, 5, 6,10,18*   2: 2, 3, 5, 9,13*   3: 2, 5, 8,14,16*   4:15,16,
18*   5: 1, 5, 6*   6: 5, 7, 8, 9,10,11,14,18*   7:10,11,15*   8: 2, 6
, 8,16*   9: 7, 9,12,15*  10:13*  11: 2, 3,10,11,12,13,17,18*  12: 2,
 3, 7, 8,14*  13: 5, 6,11,12,13,15,17*  14: 5,16*  15: 1, 2, 6*  16: 4
, 8,17*  17: 2,10,13,17,18*  18: 2, 6, 7, 8,17*
ZUFALLSPERMUTATION, U=1123069
  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
  5  2  3  4  1  6 18  8  9 10 11 12 13 14 15 16 17  7
***2.4
EINZIGER ISOMORPHISMUS:
[ 6]:[ 6]:  [13]:[13]:  [18]:[ 7]:  [17]:[17]:  [ 8]:[ 8]:  [16]:[16]:
[ 5]:[ 1]:  [11]:[11]:  [ 3]:[ 3]:  [ 1]:[ 5]:  [ 2]:[ 2]:  [ 4]:[ 4]:
[12]:[12]:  [ 9]:[ 9]:  [14]:[14]:  [10]:[10]:  [ 7]:[18]:  [15]:[15]:
```

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
 N= 18  SYMMETRISCHER ZUFALLSGRAPH, KANTENHAEUFIGKEIT=0.5, U=
          4565
 1: 2, 5,10,15,18*   2: 1, 3, 5, 8, 9,11,12,15,17*   3: 2, 5, 8,11,14,
16*   4:15,16,18*   5: 1, 2, 3, 5, 6,13,14*   6: 5, 7, 8, 9,10,11,13,
15,18*   7: 6,10,12,15,18*   8: 2, 3, 6, 8,16,18*   9: 2, 6, 9,12,15*
10: 1, 6, 7,11,13,17*  11: 2, 3, 6,10,11,12,13,17,18*  12: 2, 7, 9,11,
13,14*  13: 5, 6,10,11,12,13,15,17*  14: 3, 5,12*  15: 1, 2, 4, 6, 7,
 9,13*  16: 3, 4, 8,17*  17: 2,10,11,13,16,17,18*  18: 1, 4, 6, 7, 8,
11,17*
ZUFALLSPERMUTATION, U=7463069
  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
 11  6  3  4 12  7  2  8 16 10  1 18 13 14 15  9 17  5
****2.1
EINZIGER ISOMORPHISMUS:
[11]:[ 1]:  [ 2]:[ 6]:  [ 6]:[ 7]:  [15]:[15]:  [18]:[ 5]:  [13]:[13]:
[ 3]:[ 3]:  [12]:[18]:  [ 5]:[12]:  [17]:[17]:  [10]:[10]:  [ 8]:[ 8]:
[16]:[ 9]:  [ 9]:[16]:  [ 4]:[ 4]:  [14]:[14]:  [ 1]:[11]:  [ 7]:[ 2]:
```

```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
 N= 18  SYMMETRISCHER ZUFALLSGRAPH, KANTENHAEUFIGKEIT=0.3, U=
          4565
 1: 2, 5,10,15*   2: 1, 3, 5, 8, 9,15,17*   3: 2, 8,11,16*   4:16,18*
 5: 1, 2, 5, 6,13*   6: 5, 8,10,11,18*   7:10,15,18*   8: 2, 3, 6, 8,
16,18*   9: 2,15*  10: 1, 6, 7,11,13,17*  11: 3, 6,10,11,12,13*  12:11
,13*  13: 5,10,11,12*  14*  15: 1, 2, 7, 9*  16: 3, 4, 8,17*  17: 2,10
,16,18*  18: 4, 6, 7, 8,17*
ZUFALLSPERMUTATION, U=3511241
  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
  1 15  3  6 12  9 18  8  2 13  5  7 10  4 17 16 14 11
****1.9
EINZIGER ISOMORPHISMUS:
[ 2]:[15]:  [10]:[13]:  [ 7]:[18]:  [14]:[ 4]:  [18]:[11]:  [11]:[ 5]:
[ 8]:[ 8]:  [ 6]:[ 9]:  [15]:[17]:  [16]:[16]:  [13]:[10]:  [ 5]:[12]:
[ 1]:[ 1]:  [ 3]:[ 3]:  [17]:[14]:  [ 9]:[ 2]:  [12]:[ 7]:  [ 4]:[ 6]:
```

Literatur

[1] Corneil, D.G. and Gotlieb C.C., An efficient Algorithm for graph isomorphism, J. ACM 17(197o), 51-64.

[2] Corneil, D.G., Graph Isomorphism, Ph.D. thesis, University of Toronto 1968.

[3] Goethals, J.M. and Seidel, J.J., Orthogonal matrices with zero diagonal. Canad. J. Math. 19 (1967), 1oo1-1o1o.

[4] Knödel, W., Ein Verfahren zur Feststelllung der Isomorphie von endlichen zusammenhängenden Graphen, Computing 8 (1971), 329-334.

[5] Levi, G., Graph Isomorphism: A heuristic edge-partitioning-oriented algorithm, Computing 12 (1974), 291-313.

[6] Meyer, J.F., Algebraic isomorphism invariants for Graphs of automata, Graph Theory and Computing, ed. by R.C.Read, Academic Press, New York and London(1972).

[7] Sirovich, F., Isomorfismi fra grafi: un algoritmo efficiente per trovare tutti gli isomorfismi. Calcolo 8 (1971), 3o1-337.

[8] Sussenguth, E.H., Jr., A graph-theoretical algorithm for matching chemical structures, J. Chem. Doc. 5 (1965), 36-43.

[9] Tinhofer,G., Zur Bestimmung der Automorphismen eines endlichen Graphen, Bericht Nr. 1974-8 des Instituts für Informatik und numerische Mathematik der Universität Innsbruck.

[1o] Unger, S.H., G I T - a heuristic program for testing pairs of directed line graphs for isomorphism, Comm. ACM 7 (1964), 26-34.

[11] Grace, D.W., Computer search for non-isomorphic convex polyhedra. Ph.D. thesis, Stanford University 1965.

[12] Hoffman, A.J., On the polynomial of a graph. Am. Math. Monthly 7o (1963), 3o-36.

[13] Bose, R.C, Strongly regular graphs, partial geometries, and partially balanced designs. Pacific J. Math. 13 (1963), 389-42o.

[14] Turner, J., Point-symmetric graphs with a prime number of points. J. Comb. Theory 3 (967) 136-145.