Cliquen in Graphen
Mathematische Grundlagen und der Bron-Kerbosch-Algorithmus

Karin Haenelt

24.11.2012

Fraunhofer

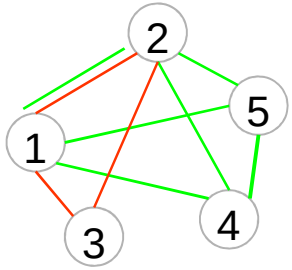
Themen

- Einführung
  - einige Clustering-Algorithmen
  - Clique-Algorithmus
- Graphentheoretische Definition: Clique
- Bron/Kerbosch-Algorithmus
  - Prinzipien
  - Tracing
  - Algorithmus
  - Optimierungen
  - Implementierungen in Bibliotheken
- Anhang 1: Originalformulierung des Algorithmus
- Anhang 2: Tracing mit Variablenlisten

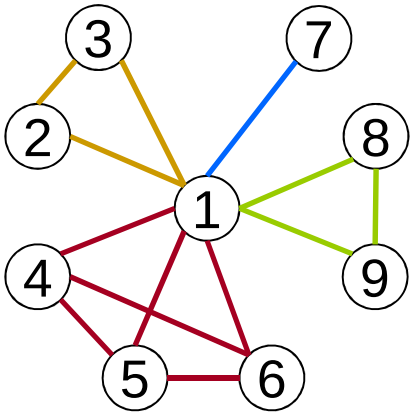
Einige Clustering-Algorithmen

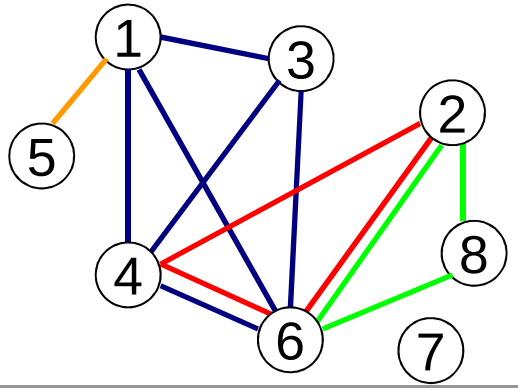
	Single Link	Clique	Star	String	k-Means	One-Pass-Assignment
Variablenauswahl						
Objekte	+	+	+	+	+	+
Attribute	+	+	+	+	+	+
Objekt-Attribut-Matrix	+	+	+	+	+	+
1. Klassifikation: Ähnlichkeitsberechnung						
Ähnlichkeitsmaß	+	+	+	+	+	+
Ähnlichkeitsmatrix	Objekt/ Objekt	Objekt/ Objekt	Objekt/ Objekt	Objekt/ Objekt	Objekt/ Zentroid	Objekt/ Zentroid
Schwellenwert	+	+	+	+	+	+
Relationsmatrix	+	+	+	+		
2. Klassifikation: Clustering						
Ableitung besser interpretierbarer Klassen	Prüfung teils komplexer Ähnlichkeitsrelationen				direkte Partition gemäß Ähnlichkeit zu Zentroid	
					mit Reallokation	one-Pass-Assignment

Cliquen
Beispiele



- Parties, auf denen sich alle gegenseitig kennen
- Terme, die alle einander ähnlich sind (auf bestimmte Weise)
  - C_1: *Note, Takt, Tempo*
  - C_2: *Note, Arbeit, Zeugnis, Schule*
  - C_3: *Note, Münze*
  - C_4: *Note, Diplomat, Regierung*
- Dokumente, die alle einander ähnlich sind (auf bestimmte W.)





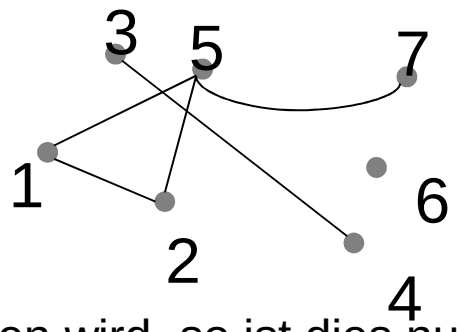
Themen

- Einführung
  - einige Clustering-Algorithmen
  - Clique-Algorithmus
- Graphentheoretische Definition: Clique
- Bron/Kerbosch-Algorithmus
  - Prinzipien
  - Tracing
  - Algorithmus
  - Optimierungen
  - Implementierungen in Bibliotheken
- Anhang 1: Originalformulierung des Algorithmus
- Anhang 2: Tracing mit Variablenlisten

Graph
Definition

Definition (Graph)

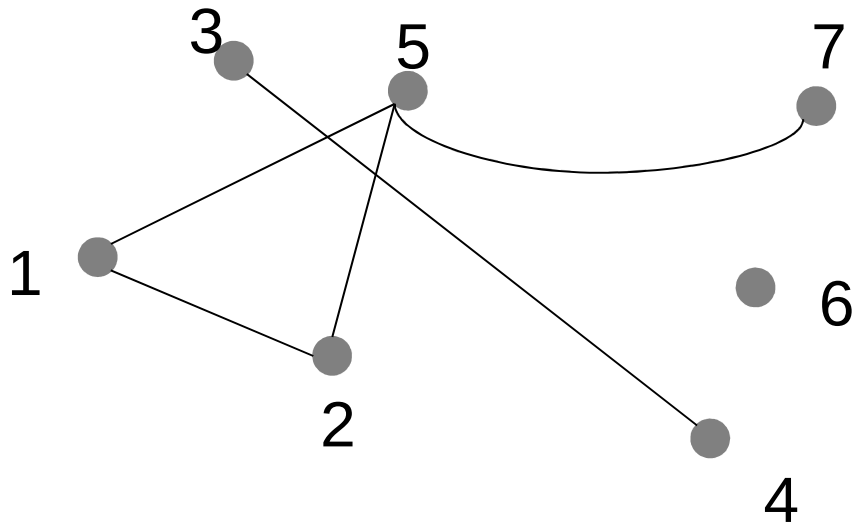
- Ein Graph G ist ein Zwei-Tupel $G = (V,E)$, wobei V eine Menge und E eine Teilmenge von $\{\{v_1,v_2\} \subset V \mid v_1 \neq v_2\}$ ist.
- Die Elemente von V heißen Knoten (vertices).
- Die Elemente von E sind 2-elementige Teilmengen von V, (also Relationen zwischen den Objekten aus V) und heißen Kanten (edges). ■



Einen Graphen kann man bildlich darstellen

- Konten als Punkte
- Kanten als Linien zwischen diesen Punkten
- „Wenn hier von Knoten und Kanten gesprochen wird, so ist dies nur eine Veranschaulichung, die sich an einen gezeichneten Graphen anlehnt. Die Definitionen sind davon unabhängig.“ (Kunze, 2001, 32)

Graph
Beispiel



Graph mit Knotenmenge $V = \{1,2,3,4,5,6,7\}$

Kantenmenge $E = \{\{1,2\}, \{1,5\}, \{2,5\}, \{3,4\}, \{5,7\}\}$

Diestel, 2006: 2

Graphen
Definitionen

Definition (gerichteter Graph / directed graph)

- ein Graph $G = (V,E,init,fin)$ mit zwei Funktionen
  - init: $E \rightarrow V$, ordnet jeder Kante *e* einen Anfangsknoten init(*e*) zu
  - fin: $E \rightarrow V$, ordnet jeder Kante *e* einen Endknoten fin(*e*) zu
- d.h. jede Kante $e = (v_1,v_2)$ ist ein geordnetes Paar. ■

Definition (ungerichteter Graph / undirected graph)

- jede Kante $e = \{v_1,v_2\}$ ist ein ungeordnetes Paar. ■

Diestel, 2006: 30

Graphen
Definitionen

Definition (benachbarte Knoten / adjacent)

- zwei Knoten v_1,v_2 eines Graphen G heißen benachbart, wenn sie durch eine Kante $e = \{v_1,v_2\}$ verbunden sind. ■

Definition (Grad eines Knoten / degree)

- der Grad eines Knoten ist die Anzahl der Kanten, die von einem Knoten ausgehen. ■

Definition (regulärer Graph / regular graph)

- Ein Graph G heißt regulär, wenn alle Knoten denselben Grad haben. ■

Graphen
Definitionen

Definition (vollständiger Graph / complete graph)

- ein Graph G heißt vollständig, wenn alle Knoten in G paarweise benachbart sind. ■

Definition (Teilgraph / subgraph)

- ein Graph $G_2 = (V_2, E_2)$ heißt Teilgraph eines Graphen $G_1 = (V_1, E_1)$ wenn gilt $V_2 \subseteq V_1$ und $E_2 \subseteq E_1$. ■

Definition (Untergraph = induzierter Teilgraph / induced subgraph)

- ein Teilgraph G_2 heißt induziert oder aufgespannt (V_2 induziert oder spannt G_2 in G_1 auf), wenn er alle Kanten $\{v,w\} \in E_1$ mit $v,w \in V_2$ enthält. ■

Abb. 0.1.2. Ein Graph G mit Teilgraphen G' und G'': G' ist Untergraph von G, G'' ist es nicht.

Diestel, 2006: 3

Clique
Definition

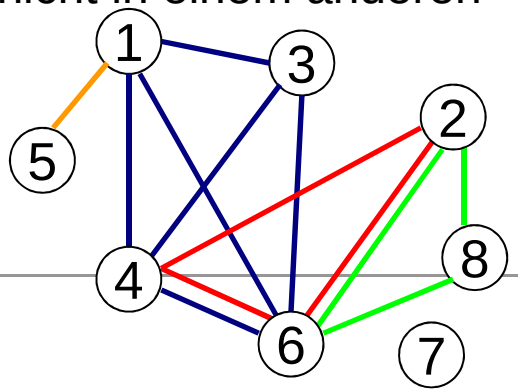
Definition (Clique)

- eine Teilmenge *C* der Knotenmenge eines ungerichteten Graphen *G* heißt Clique, falls der von *C* induzierte Untergraph von *G* vollständig ist. ■ (Turau, 1996: 131)

Definition (maximale Clique)

- eine maximale Clique *C* eines Graphen *G* ist eine Clique, die nicht echt enthalten ist in einer anderen Clique, d.h. es gibt keine Clique *D* von *G*, für die gilt $C \subseteq D$ und $C \neq D$. ■ (Valiente, 2002: 3001)
- m.a.W: ein vollständiger Untergraph, der nicht in einem anderen vollständigen Untergraphen enthalten ist

Beispiel: Graph G,
Darstellung der Cliquen durch farbige Kanten



Komplexität der Suche aller maximalen Cliquen

- theoretisch: exponentiell
  - zu einer Menge von *n* Knoten gibt es 2^n mögliche Teilmengen
  - diese können zwar nicht alle maximale Cliquen sein
  - aber auch die maximale Zahl maximaler Cliquen eines ungerichteten Graphen kann exponentiell zur Anzahl der Knoten sein
    Moon/Moser (1965): für $n \geq 2$: $f(n) = 3^{n/3}$
- für den allgemeinen Fall ist das Problem der Suche maximaler Cliquen exponentiell und damit NP-hart
- schlechtester Fall: $O(3^{n/3})$

Komplexität der Suche aller maximalen Cliquen

- praktisch
  - in vielen Fällen liegen in praktischen Anwendungen Graphen vor, deren Durchsuchung nicht die Komplexität des schlechtesten Falles erreicht

Themen

- Einführung
  - einige Clustering-Algorithmen
  - Clique-Algorithmus
- Graphentheoretische Definition: Clique
- Bron/Kerbosch-Algorithmus
  - Prinzipien
  - Tracing
  - Algorithmus
  - Optimierungen
  - Implementierungen in Bibliotheken
- Anhang 1: Originalformulierung des Algorithmus
- Anhang 2: Tracing mit Variablenlisten

Bron/Kerbosch-Algorithmus

Bron, Coen und Joep Kerbosch (1973). Finding all Cliques of an Undirected Graph. In: *Communications of the ACM*, 16, 9 S. 575-579, Algorithm 457.

- theoretisch exponentielle Laufzeit
- praktisch meist gute Laufzeit
- gilt als schnellster Algorithmus zur Aufzählung sämtlicher Cliquen in Graphen

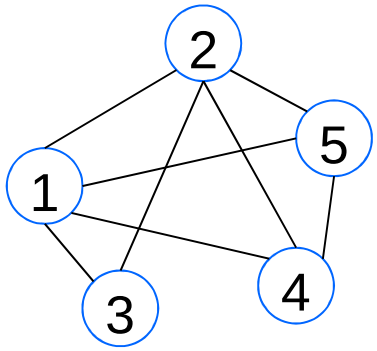
- Spezifikation des Algorithmus in Algol 60
- einige neuere Beschreibungen (mit unterschiedlicher Effizienz)
- Basis der folgenden Beschreibungen
  - Bron/Kerbosch (1973)
  - Samudrala/Moult (1998)

Bron/Kerbosch-Algorithmus
Prinzip

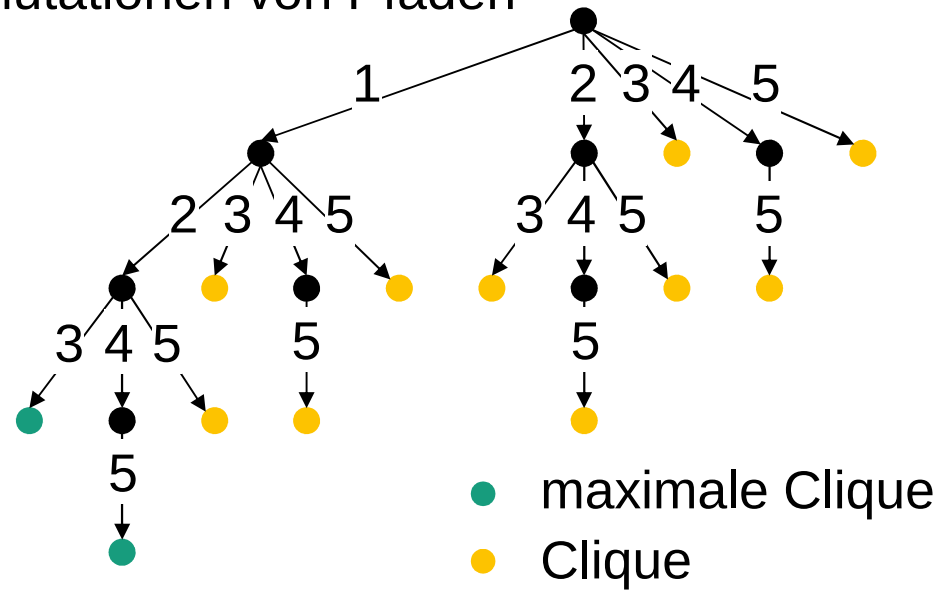
- kombiniert
  - rekursive backtracking-Prozedur
    - effiziente Durchsuchung eines Graphen:
    - keine Permutationen bereits gefundener Cliquen suchen:
      - zu einem Knoten nur die Knoten mit „höheren Adressen“ suchen, keine Rückkehr zu „niedrigeren Adressen“ (Hälfte der Relationsmatrix oberhalb der Diagonalen)
  - „branch and bound“-Technik
    - Begrenzung einer Suche, die nicht zu einer maximalen Clique führen kann:
    - keine identischen Teilbäume bei der Suche aufbauen:
      - nur neue Knoten testen, d.h. Knoten, die nicht Nachbarn eines bereits getesteten Knotens sind

Bron-Kerbosch-Algorithmus
1. Prinzip: Rekursive Backtracking-Prozedur

Graph G



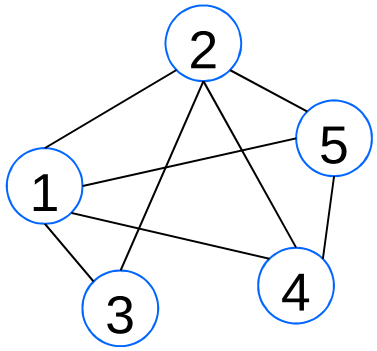
Rekursionsbaum der Traversion des Graphen G ohne Permutationen von Pfaden



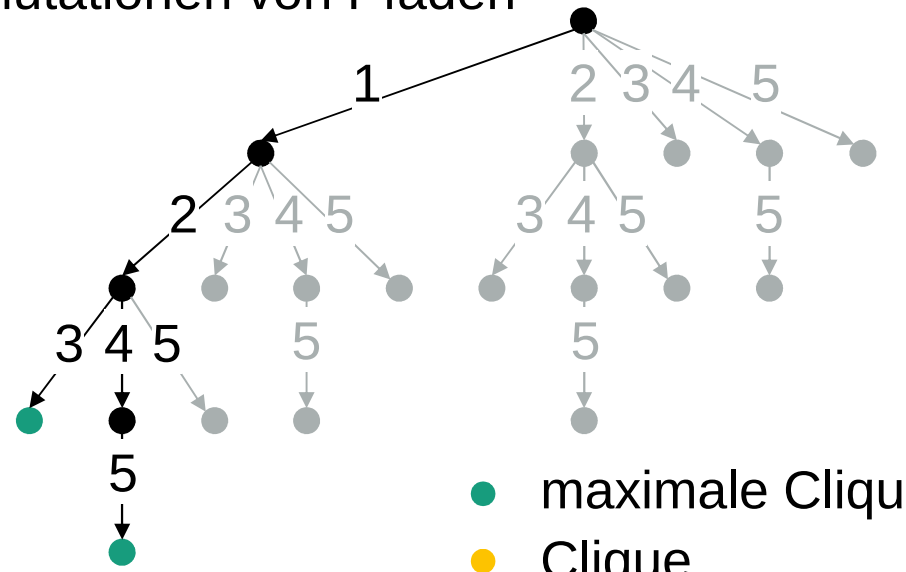
Beispiel aus Koch, 2001:10

Bron-Kerbosch-Algorithmus
2. Prinzip: branch-and-bound-Mechanismus

Graph G



Rekursionsbaum der Traversion des Graphen G ohne Permutationen von Pfaden



Begrenzung der Suche: ein bereits besuchter Knoten ist Nachbar aller unbesuchten Geschwisterknoten

Beispiel aus Koch, 2001:10

Bron/Kerbosch-Algorithmus

Umsetzung: schrittweise Aktualisierung der folgenden Knotenmengen

(1) Clique (C; wie „Clique“)

- jeweils aktueller Stand des schrittweisen Aufbaus
- Menge von Knoten, die alle mit allen verbunden sind
- jeder rekursive Aufruf
  - fügt entweder einen Knoten hinzu (weitere Tiefe im Suchbaum)
  - oder entfernt einen Knoten (backtracking im Suchbaum)

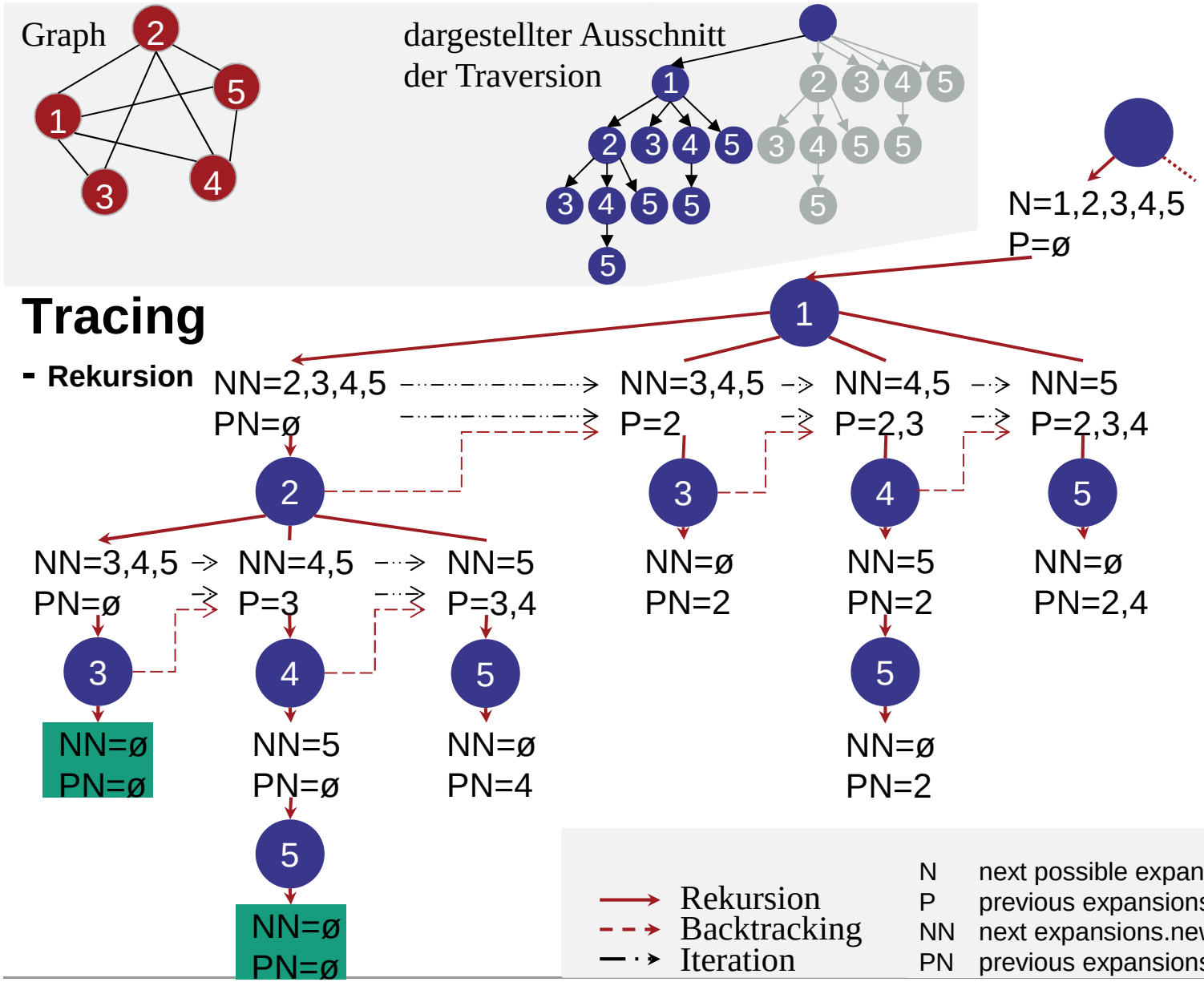
(2) mögliche Erweiterungen (N; wie „next potential expansions“)

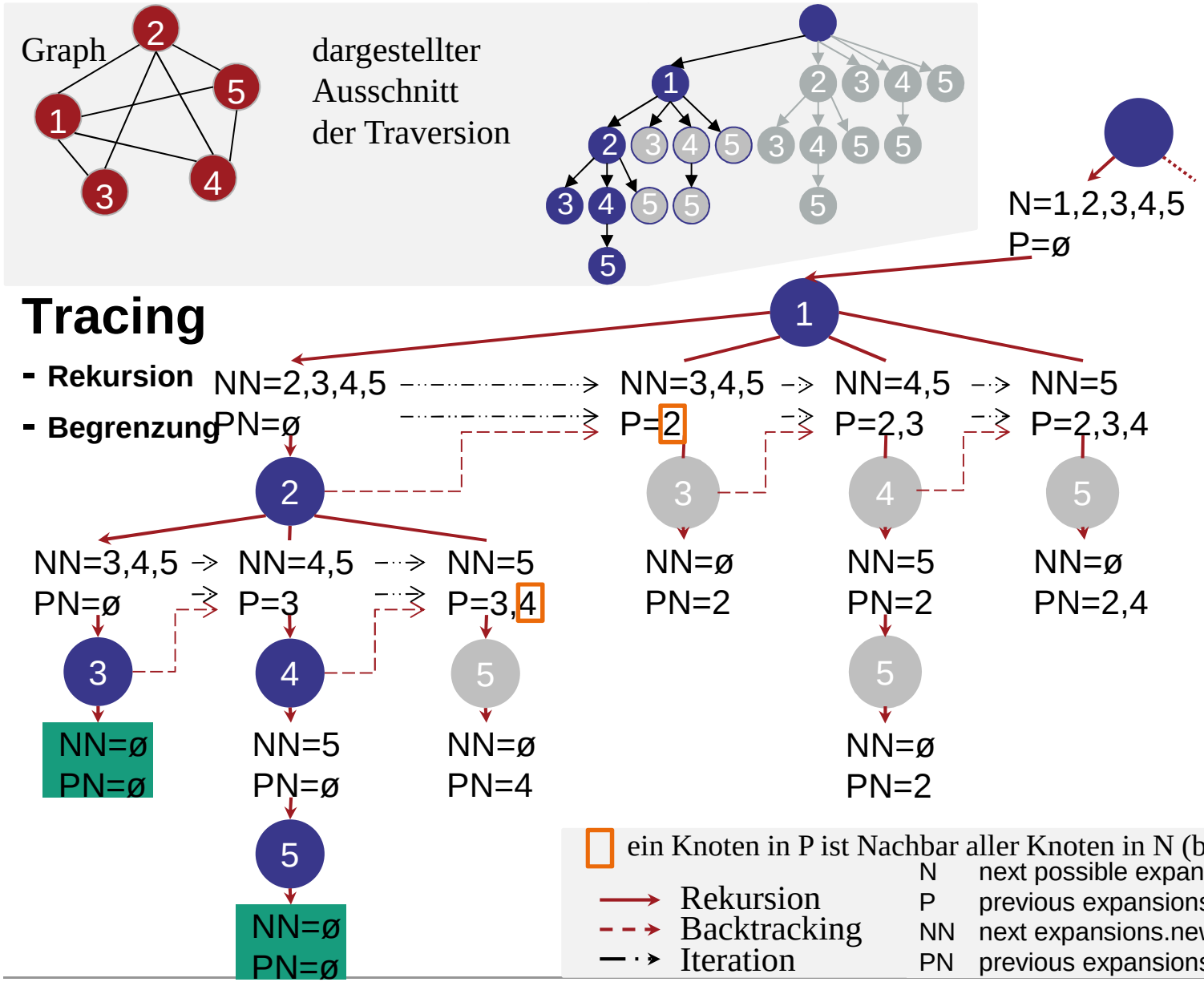
- Menge von Knoten, die noch als Erweiterung von (1) wählbar sind

(3) bekannte Erweiterungen (P; wie „previous expansions“)

- Menge von Knoten, die bereits als Erweiterungen der momentanen Konfiguration von Clique dienten und nicht noch einmal geprüft werden sollen

nach Samudrala/Moult, 1998:289


Graph
dargestellter Ausschnitt
der Traversion
N=1,2,3,4,5
P=ø
Tracing
- Rekursion
NN=2,3,4,5
PN=ø
NN=3,4,5
P=2
NN=4,5
P=2,3
NN=5
P=2,3,4
NN=3,4,5
PN=ø
NN=4,5
P=3
NN=5
P=3,4
NN=ø
PN=2
NN=5
PN=2
NN=ø
PN=2,4
NN=ø
PN=ø
NN=5
PN=ø
NN=ø
PN=4
NN=ø
PN=2
NN=ø
PN=ø
Rekursion
Backtracking
Iteration
N next possible expansions
P previous expansions
NN next expansions.new
PN previous expansions.new


Graph
dargestellter Ausschnitt der Traversion
N=1,2,3,4,5
P=ø
Tracing
- Rekursion
- Begrenzung
NN=2,3,4,5
PN=ø
NN=3,4,5
P=2
NN=4,5
P=2,3
NN=5
P=2,3,4
NN=3,4,5
PN=ø
NN=4,5
P=3
NN=5
P=3,4
NN=ø
PN=2
NN=5
PN=2
NN=ø
PN=2,4
NN=ø
PN=ø
NN=5
PN=ø
NN=ø
PN=4
NN=ø
PN=2
NN=ø
PN=ø
ein Knoten in P ist Nachbar aller Knoten in N (bound!)
Rekursion
Backtracking
Iteration
N next possible expansions
P previous expansions
NN next expansions.new
PN previous expansions.new

Clique-Algorithmus
Tracing

c	v	N	P	NN	PN
		1,2,3,4,5	-		
1	1	2,3,4,5	-	2,3,4,5	-
1,2	2	3,4,5	-	3,4,5	-
1,2,3	3	4,5	-	-	-
1,2		4,5	3		
1,2,4	4	5		5	-
1,2,4,5	5	-	-		
1,2,4					
1,2	5		3,4		

Bron-Kerbosch-Algorithmus

```
01  enumerateAllMaximalCliques()
02  {   graph G = (V,E); node u,v; set<node> C,N,P,NN,PN;
03      forall_nodes(v,G) N.insert(v);
04      nextMaxCliques(C,N,P)
05      {   if      {ein Knoten in P ist Nachbar aller Knoten in N
                    then keine max.Clique mehr auffindbar}
06          else
07                  {   forall_nodes(v,N)
08                      {   N ← N \ {v}
09                          C ← C ∪ {v}
10                          NN ← {w ∈ N | {v,w} ∈ E}
11                          PN ← {w ∈ P | {v,w} ∈ E}
12                          if   (NN == ∅ und
                                  PN == ∅)
                                  then reportClique();
                                  else nextMaxCliques(C, NN, PN)
13                          C ← C \ {v}
14                          P ← P ∪ {v}
15
16  }   }           }   }
```

// keine weiteren Erweiterungskandidaten

// Clique nicht enthalten in einer anderen

C	clique
N	next possible expansions
P	previous expansions
v	actual node
NN	next expansions.new
PN	previous expansions.new

Begrenzung (bound)

Iteration
schrittweise Erweiterung
von C durch Knoten aus N

Aktualisierung von N und P:
NNeu und PNeu:
Einschränkung auf Knoten,
die zu v benachbart sind

Rekursion

Backtracking
- Zurücksetzen von C
- Merken bekannter Erweiterungen von C in P

formalisiert auf der Basis von Samudrala/Moult(1988:290)

Bron-Kerbosch-Algorithmus

Tracing-Beispiel

C clique
N next possible expansions
P previous expansions
NN next expansions.new
PN previous expansions.new

	C	N	P	NN	PN
04	-	1,2,3,4,5	-		
08-11	1	2,3,4,5	-	2,3,4,5	-
08-11	2	3,4,5		3,4,5	1
08-11	3	4,5		-	1,2
08-11	4	5		5	1,2
08-11	5	-		-	1,2,4

	C	N	P	NN	PN
08-11	1,2	3,4,5	-	3,4,5	-
08-11	1,3	4,5		-	2
08-11	1,4	5		5	2
08-11	1,5	-		-	2,4
08-11	2,3	4,5		-	1
08-11	2,4	5		5	1
08-11	2,5	-		-	1,4
08-11	4,5	-		-	1,2

	C	N	P	NN	PN
08-11	1,2,3	4,5	-	-	-
12	Clique				
08-11	1,2,4	5		5	-
08-11	1,2,5	-		-	4
08-11	1,4,5	-		-	2
08-11	2,4,5	-	1		

	C	N	P	NN	PN
08-11	1,2,4,5	-	-	-	-
12	Clique				

Zahlen in den grauen Kästen: Zeilennummern des Algorithmus

Bron/Kerbosch-Algorithmus
Formulierungsvariante 1

enumerateNextMaximalClique(C,N,P)

- schrittweise Erweiterung von C durch Knoten aus N
  - Auswahl eines Knoten v aus next potential expansions (N)
  - Hinzufügen von v zu Clique (C)
- Aktualisierung von N und P: Erzeugung der Mengen
  - next potential expansions.new (NN) aus N
  - previous expansions.new (PN) aus P

  durch Reduktion auf Knoten, die zu v benachbart sind
- rekursive Verarbeitung für aktualisierte Mengen: enumerateNextMaximalClique(C,NN,PN)
- Vorbereitung für backtracking:
  - Entfernung von v aus C
  - Merken der bekannten Erweiterungen von C in P: Hinzufügung von v zu P

formuliert auf der Basis von
Bron/Kerbosch, 1973: 575

Bron/Kerbosch-Algorithmus

Formulierungsvariante 2

```
begin procedure find-cliques (C, N, P)
   if       ein Knoten in P ist Nachbar aller Knoten in N
            then es kann keine max.Clique mehr gefunden werden
                    (branch and bound step)
   else
            for jeden Knoten v in N do
               verschiebe Knoten v in C
               erzeuge NN durch Entfernung der Knoten aus N,
                              die keine Nachbarn von v sind
               erzeuge PN durch Entfernung der Knoten aus P,
                               die keine Nachbarn von v sind
               if           NN und PN sind leer
                            then C ist eine maximale Clique
               else         find-cliques (C, NN, PN)
               endif
               verschiebe v von C nach P
            endfor
   endif
end procedure find-cliques
```

C	clique
N	next possible expansions
P	previous expansions
v	actual node
NN	next expansions.new
PN	previous expansions.new

nach Samudrala/Moult (1998:290)

Bron/Kerbosch-Algorithmus
Versionen

- Bron/Kerbosch, Version 1:
  - Rekursion mit Begrenzungsbedingung
  - Betrachtung der Knoten in Speicherreihenfolge
- Bron/Kerbosch, Version 2:
  - Rekursion mit Begrenzungsbedingung
  - Betrachtung der Knoten in berechneter Reihenfolge
    (hat im gezeigten Beispiel keine Auswirkung auf Anzahl der besuchten Knoten)

Bron-Kerbosch-Algorithmus
Version 1

- Begrenzungsbedingung
  - ein Knoten in P ist Nachbar aller Knoten in N
- Prinzip 1
  - Knoten v aus N werden nach der Speicherabfolge gewählt

Bron-Kerbosch-Algorithmus

Version 2

zur Erinnerung: Begrenzungsbedingung:
{ein Knoten in P ist Nachbar aller Knoten in N}

- Grundgedanke
  Knoten *v* aus *N* werden nicht der Reihe nach gewählt, sondern so, dass die Begrenzungsbedingungen möglichst früh erkannt werden
- Prinzip 2: Reihenfolge der Abarbeitung der Knoten v aus N berechnet
  - jedem Knoten in P ist ein Zähler nd (number of disconnections) zugeordnet:
    - zählt Anzahl der Knoten in N, zu denen dieser Knoten nicht benachbart ist
  - Verschieben eines Knoten v von N in P (beim backtracking):
    - Zähler der Knoten in P, zu denen v nicht benachbart ist, um 1 vermindern
    - Zähler für v erzeugen
  - Wenn ein Zähler 0 erreicht, ist die Begrenzungsbedingung erfüllt
  - jeweils Auswahl des Knotens w in P mit dem niedrigsten Zähler
  - jeweils Auswahl eines Knotens v aus N, der zu w aus P nicht benachbart ist (d.h. Zähler von w erreicht am schnellsten 0)

Bron/Kerbosch (1973:575/576)

Bron-Kerbosch-Algorithmus
Eigenschaften

- findet jede maximale Clique genau einmal
- Optimierungen: möglichst frühzeitige Begrenzung einer Suchalternative
  - Rekursionsmechanismus:
    - keine Permutationen bereits gefundener Cliquen suchen: zu einem Knoten nur die Knoten mit „höheren Adressen“ suchen, keine Rückkehr zu „niedrigeren Adressen“: $N \leftarrow N \setminus \{v\}$
  - Bound-Bedingung
    - keine identischen Teilbäume bei der Suche aufbauen: nur neue Knoten testen, d.h. Knoten, die nicht Nachbarn eines bereits getesteten Knotens sind

Bron-Kerbosch-Algorithmus
Laufzeittests

- zufällige Graphen
  - Graphen mit
    - 10 bis 50 Knoten
    - 10%, 30%, 50%, 70%, 90%, 95% Kanten
  - Laufzeit linear
- Graphen mit der größten Anzahl von Cliquen pro Knoten
  - reguläre Graphen der Dimensionen $3 \times k$, konstruiert als Komplement von k disjunkten 3-Cliquen
  - Laufzeit exponentiell

Bron/Kerbosch (1973:576)

Bron-Kerbosch-Algorithmus
Implementierung in Bibliotheken

- JGraphT – Java Graph Library http://www.jgrapht.org/

- Valiente, Gabriel (2002). *Algorithms on Trees and Graphs*. Berlin / Heidelberg / New York: Springer-Verlag.
  Quellcode auf der Basis von LEDA (library of efficient data structures and algorithms 4.4.1)
  http://www.lsi.upc.edu/~valiente/algorithm/combin.cpp

Themen

- Einführung
  - einige Clustering-Algorithmen
  - Clique-Algorithmus
- Graphentheoretische Definition: Clique
- Bron/Kerbosch-Algorithmus
  - Prinzipien
  - Tracing
  - Algorithmus
  - Optimierungen
  - Implementierungen in Bibliotheken
- Anhang 1: Originalformulierung des Algorithmus
- Anhang 2: Tracing mit Variablenlisten

Bron-Kerbosch-Algorithmus
Originalformulierung

sets compsub (hier: Clique), candidates (hier: Next), not (hier: Previous)

Step 1. Selection of a candidate

Step 2. Adding the selected candidate to compsub

Step 3. Creating new sets candidates and not from the old sets by removing all points not connected to the selected candidate (to remain consistent with the definition), keeping the old sets in tact

Step 4. Calling the extension operator to operate on the sets just formed

Step 5. Upon return, removal of the selected candidate from compsub and its addition to the old set not.

If at some stage *not* contains a point connected to all points in *candidates*, we can predict that further extensions (further selection of candidates) will never lead to the removal (in Step 3) of that particular point from subsequent configurations of *not* and, therefore, not to a clique. This is the branch and bound method which enables us to detect in an early stage branches of the backtracking tree that do not lead to successful endpoints.

Bron/Kerbosch (1973:575)

Themen

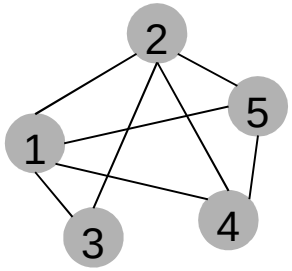
- Einführung
  - einige Clustering-Algorithmen
  - Clique-Algorithmus
- Graphentheoretische Definition: Clique
- Bron/Kerbosch-Algorithmus
  - Prinzipien
  - Tracing
  - Algorithmus
  - Optimierungen
  - Implementierungen in Bibliotheken
- Anhang 1: Originalformulierung des Algorithmus
- Anhang 2: Tracing mit Variablenlisten

Bron-Kerbosch-Algorithmus
Anhang 2

- Tracing mit Variablenlisten

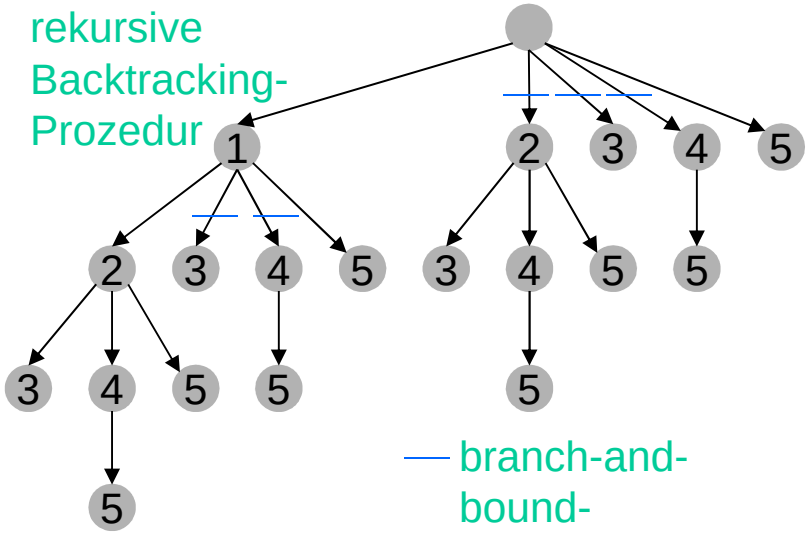
Bron-Kerbosch-Algorithmus

Graph G



Tiefensuch-Traversionsbaum von G
(ohne Permutationen von Pfaden)

rekursive Backtracking-Prozedur

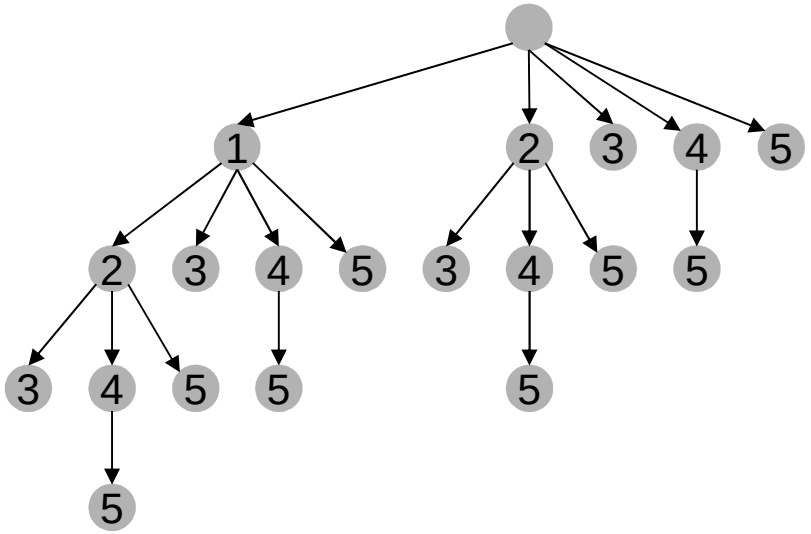


— branch-and-bound-Mechanismus

	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N | {v,w} ∈ E}	{w∈P |	
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○○		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○○	1,2,	3	4,5	-	-	-	c
○○○○	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

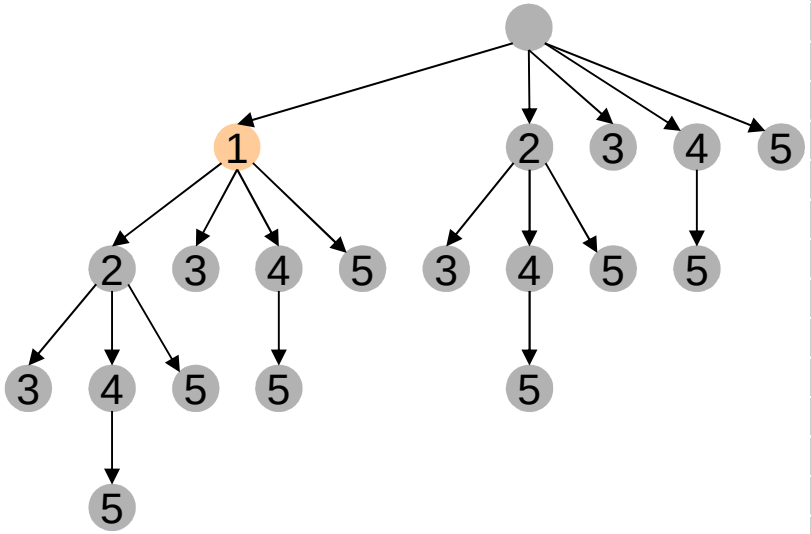
1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf
5. Entfernen von v aus C und
   Hinzufügen von v zu P
6. iterativer Aufruf



	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N | {v,w} ∈ E}	{w∈P |	
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○○		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○○	1,2,	3	4,5	-	-	-	c
○○○○	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

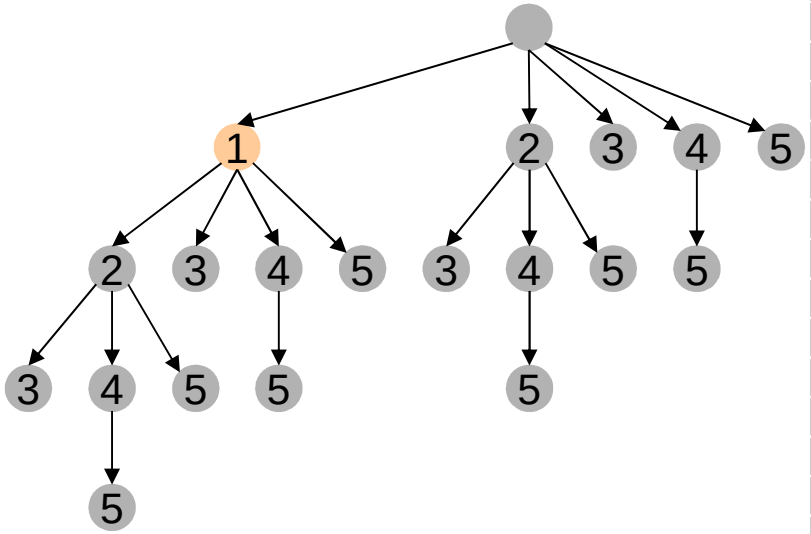
1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf
5. Entfernen von v aus C und
   Hinzufügen von v zu P
6. iterativer Aufruf



	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N | {v,w} ∈ E}	{w∈P |	
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○●		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○○	1,2,	3	4,5	-	-	-	c
○○○○	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

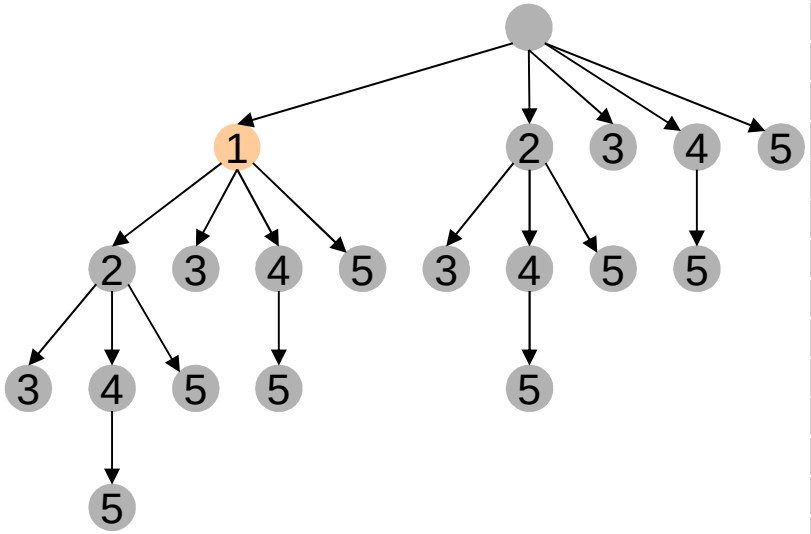
1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf
5. Entfernen von v aus C und
   Hinzufügen von v zu P
6. iterativer Aufruf



	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N |	{w∈P |	
					{v,w} ∈ E}		
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○●		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○○	1,2,	3	4,5	-	-	-	c
○○○○	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

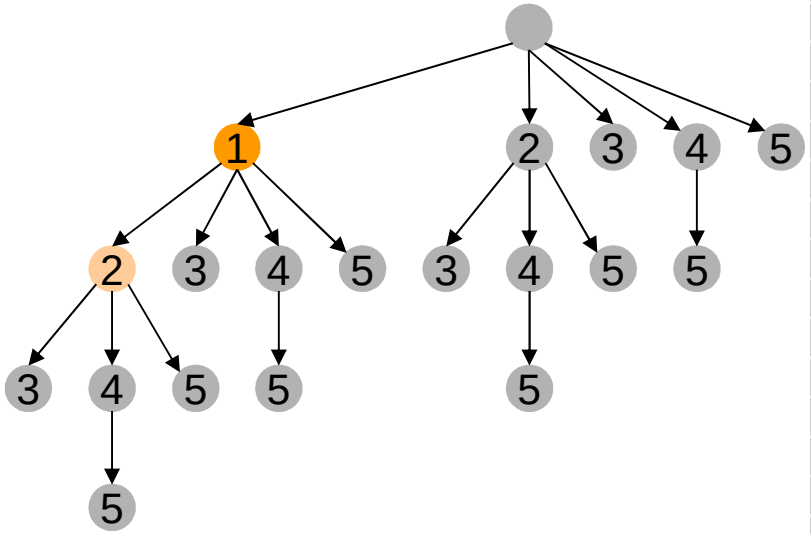
1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf
5. Entfernen von v aus C und
   Hinzufügen von v zu P
6. iterativer Aufruf



	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N |	{w∈P |	
					{v,w} ∈ E}		
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○●		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○○	1,2,	3	4,5	-	-	-	c
○○○○	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

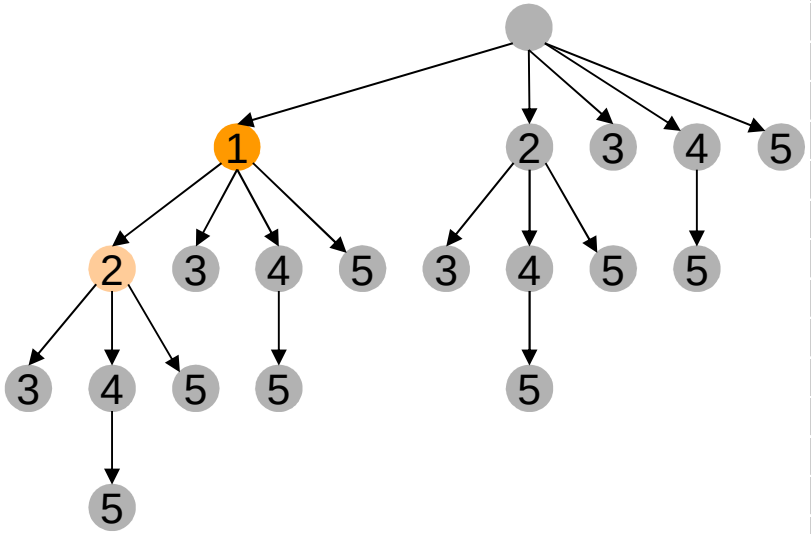
1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf
5. Entfernen von v aus C und
   Hinzufügen von v zu P
6. iterativer Aufruf



	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N | {v,w}	{w∈P |	
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○○		1	2,3,4,5	-	2,3,4,5	-	
○○●	1,	2	3,4,5	-	3,4,5	-	
○○○○	1,2,	3	4,5	-	-	-	c
○○○○	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

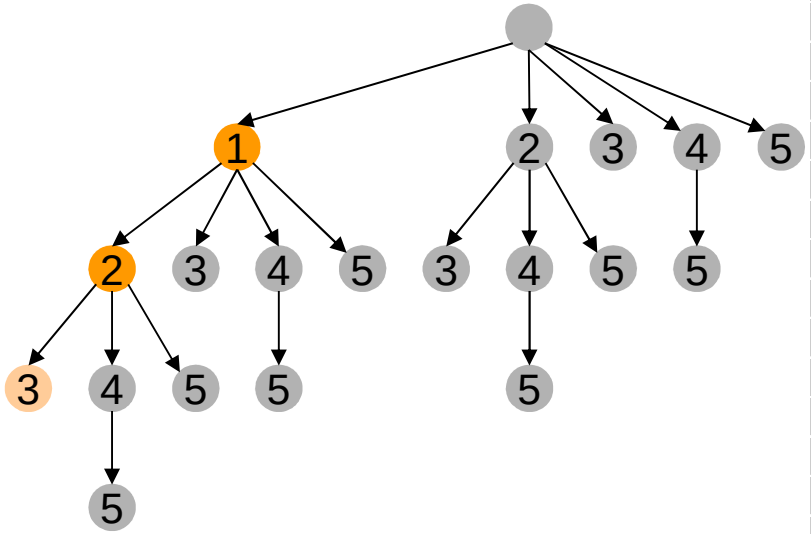
1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf
5. Entfernen von v aus C und
   Hinzufügen von v zu P
6. iterativer Aufruf



	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N | {v,w} ∈ E}	{w∈P |	
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○○		1	2,3,4,5	-	2,3,4,5	-	
○○●	1,	2	3,4,5	-	3,4,5	-	
○○○○	1,2,	3	4,5	-	-	-	c
○○○○	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

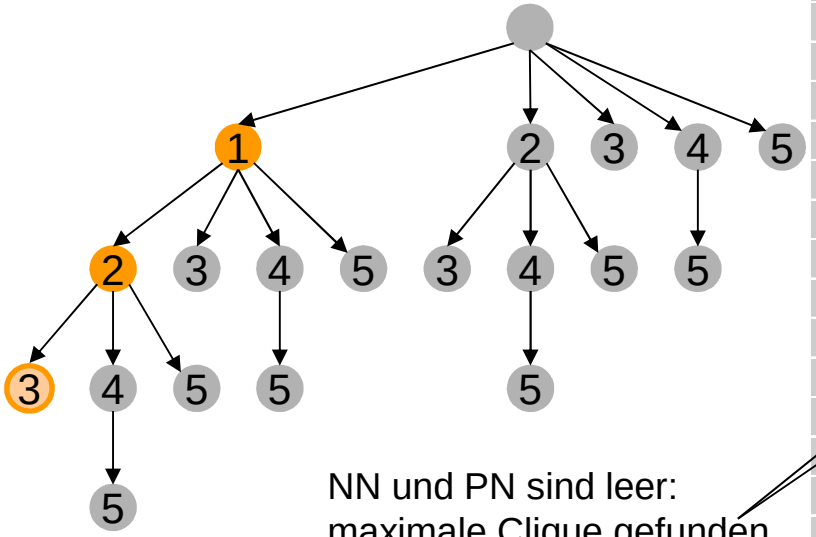
1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf
5. Entfernen von v aus C und
   Hinzufügen von v zu P
6. iterativer Aufruf



	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N | {v,w} ∈ E}	{w∈P |	
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○○		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○●	1,2,	3	4,5	-	-	-	c
○○○○	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf
5. Entfernen von v aus C und
   Hinzufügen von v zu P
6. iterativer Aufruf



NN und PN sind leer:
maximale Clique gefunden

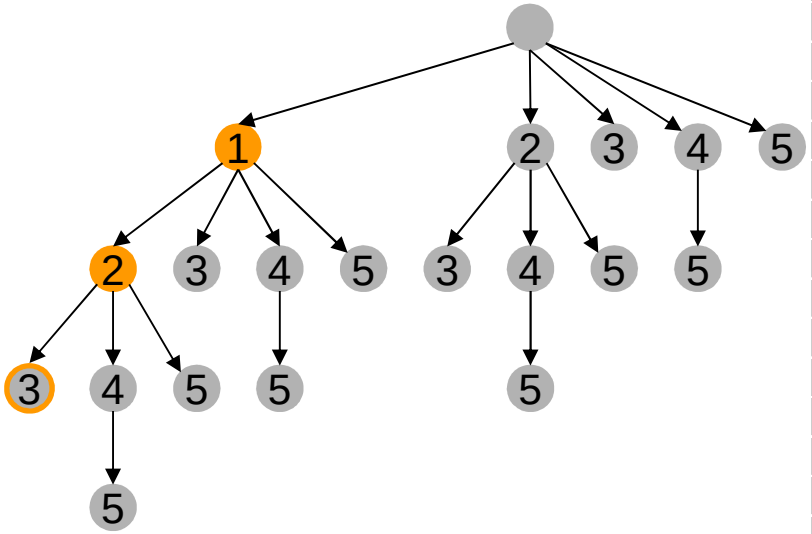
	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N | {v,w} ∈ E}	{w∈P |	
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○○		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○●	1,2,	3	4,5	-	-	-	c
○○○○	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf

○ Entfernen von v aus C und Hinzufügen von v zu P

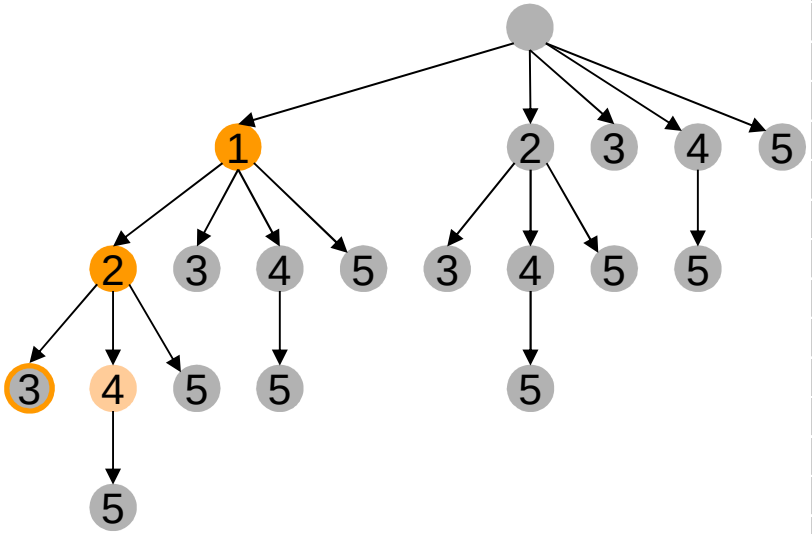
○ iterativer Aufruf



	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N | {v,w} ∈ E}	{w∈P |	
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○○		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○●	1,2,	3	4,5	-	-	-	c
○○○○	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

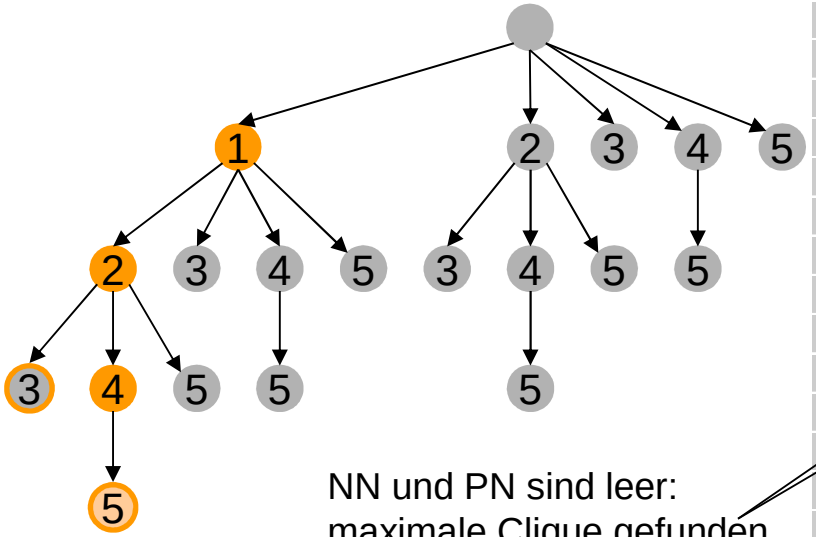
1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf
5. Entfernen von v aus C und
   Hinzufügen von v zu P
6. iterativer Aufruf



	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N | {v,w} ∈ E}	{w∈P |	
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○○		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○○	1,2,	3	4,5	-	-	-	c
○○○●	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf
5. Entfernen von v aus C und
   Hinzufügen von v zu P
6. iterativer Aufruf



NN und PN sind leer:
maximale Clique gefunden

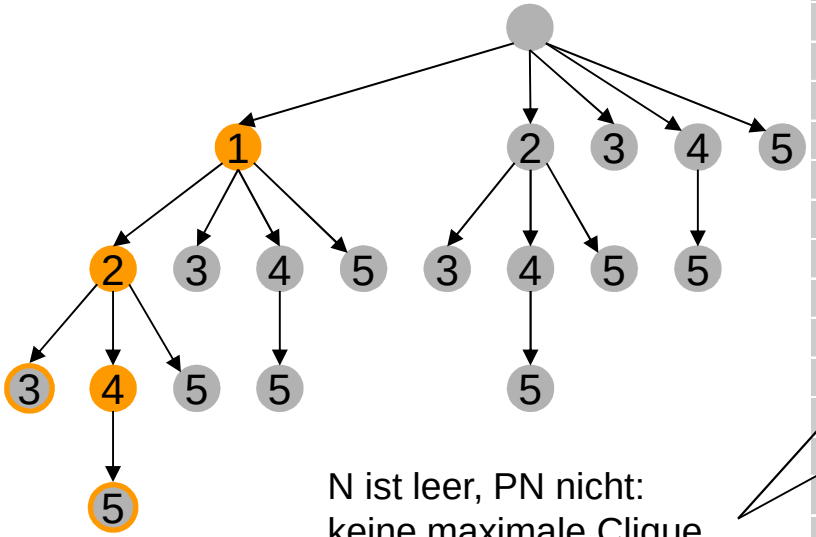
	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N | {v,w} ∈ E}	{w∈P |	
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○○		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○○	1,2,	3	4,5	-	-	-	c
○○○○	1,2,	4	5	3	5	-	
○○○○●	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf

○ Entfernen von v aus C und Hinzufügen von v zu P

○ iterativer Aufruf



N ist leer, PN nicht:
keine maximale Clique

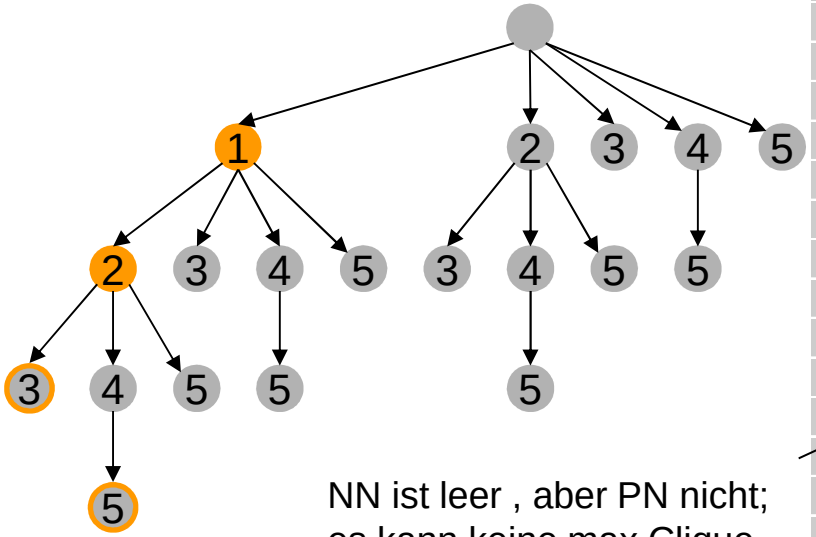
	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N |	{w∈P |	
					{v,w} ∈ E}		
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○○		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○○	1,2,	3	4,5	-	-	-	c
○○○●	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

Bron-Kerbosch-Algorithmus

1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf

○ Entfernen von v aus C und Hinzufügen von v zu P

○ iterativer Aufruf

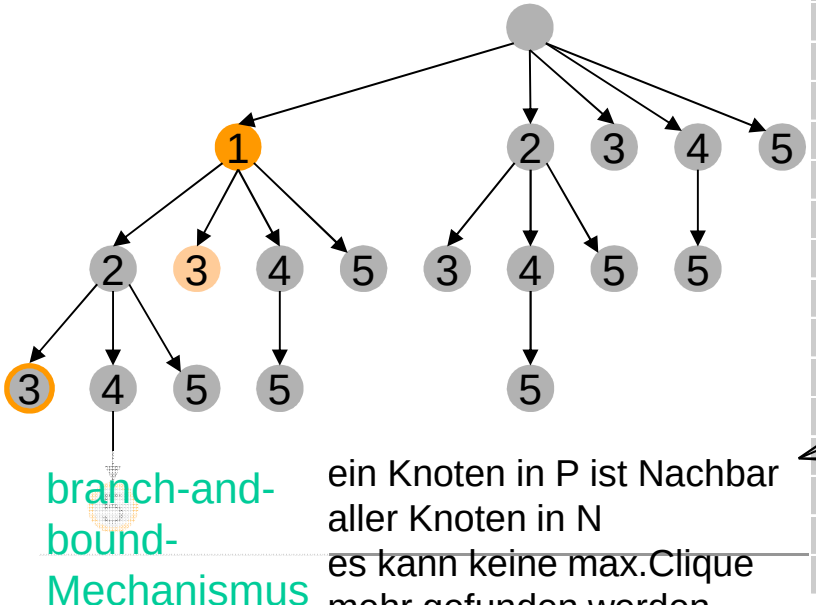


	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N |	{w∈P |	
					{v,w} ∈ E}		
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○○		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○○	1,2,	3	4,5	-	-	-	c
○○○●	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

NN ist leer , aber PN nicht;
es kann keine max.Clique
mehr gefunden werden

Bron-Kerbosch-Algorithmus

1. Auswahl eines Knotens v aus N
2. Verschieben von v aus N in C
3. NNeu und PNeu:
   nur Nachbarn von v
4. maxClique oder rekursiver Aufruf
5. Entfernen von v aus C und
   Hinzufügen von v zu P
6. iterativer Aufruf



	Clique		Next	Prev	Next New	Prev New	
	C←C∪{v}		N←N\{v}	P←P∪{v}	{w∈N |	{w∈P |	
					{v,w} ∈ E}		
	C	v	N	P	NN	PN	
○	-		1,2,3,4,5	-			
○○		1	2,3,4,5	-	2,3,4,5	-	
○○○	1,	2	3,4,5	-	3,4,5	-	
○○○○	1,2,	3	4,5	-	-	-	c
○○○○	1,2,	4	5	3	5	-	
○○○○○	1,2,4,	5	-	-	-	-	c
○○○○	1,2,	5	-	3,4	-	4	
○○○	1,	3	4,5	2	-	2	
○○○	1,	4	5	2,3	5	2	
○○○○	1,4,	5	-	2,3	-	2	
○○○	1,	5	-	2,3,4	-	2,4	
○○		2	3,4,5	1	3,4,5	1	
○○○	2,	3	4,5	1	-	1	
○○○	2,	4	5	1,3	5	1	
○○○○	2,4,	5	-	1			
○○○	2,	5	-	1,3,4	-	1,4	
○○		3	4,5	1,2	-	1,2	
○○		4	5	1,2,3	5	1,2	
○○○	4,	5	-	1,2,3	-	1,2	
○○		5	-	1,2,3,4	-	1,2,4	

branch-and-bound-Mechanismus

ein Knoten in P ist Nachbar aller Knoten in N
es kann keine max.Clique mehr gefunden werden

Vielen Dank

Für das Aufspüren von Fehlern in früheren Versionen und für Verbesserungsvorschläge danke ich

- Johannes Knopp. Tilman Wittl

Literatur

Originalartikel

Bron, Coen und Joep Kerbosch (1973). Finding all Cliques of an Undirected Graph. In: *Communications of the ACM*, 16, 9 S. 575-579, Algorithm 457.

Literatur

Beschreibungen und Modifikationen

Cazals, Frédérick und Chinmay Karande (2005). Reporting maximal cliques: new insights into an old problem. INRIA Rapport de recherche No. 5615. http://www.inria.fr/rrrt/rr-5615.html

Koch, Ina (2001). Enumerating all connected maximal common subgraphs in two graphs. In: *Theoretical Computer Science* 250, S. 1-30.

Samudrala, Ram und John Moult (1998). A Graph-theoretic Algorithm for Comparative Modeling of Protein Structure. In: Journal of Molecular Biology 279: 287-302.

Valiente, Gabriel (2002). *Algorithms on Trees and Graphs*. Berlin / Heidelberg / New York: Springer-Verlag.

Artymiuk, P., Poirrette, A., Rice, D. & Willett, P. (1995). Comparison of protein folds and sidechain clusters using algorithms from graph theory. In Proceedings of the CCP4 Study Weekend, SERC, Daresbury Laboratory, Daresbury

Brint, A.T., Willet, P. (1987). Algorithms for the Identification of three-dimensional maximal common substructures, J. Chem. Inform. Comput. Sci. 2 (1987) 311-320.

Gerhards, L., Lindenberg, W. (1979). Clique detection for nondirected graphs: two new algorithms, Computing 21 (1979) 295-322.

Literatur

Mathematische Grundlagen und Graphalgorithmen

Diestel, Reinhard (2006). Graphentheorie. 3. Auflage. Heidelberg: Springer-Verlag.

Kunze, Jürgen (2001). *Computerlinguistik. Voraussetzungen, Grundlagen, Werkzeuge*. Vorlesungsskript. Humboldt Universität zu Berlin. http://www2.rz.hu-berlin.de/compling/Lehrstuhl/Skripte/Computerlinguistik_1/index.html.

Moon, John W. und Leo Moser (1965). On Cliques in Graphs. Israel *Journal of Mathematics* 3 (1): 23-28.

Turau, Volker (1996). *Algorithmische Graphentheorie*. Bonn u.a.: Addison-Wesley Publishing Company.

Tracing der vorgestellten Versionen

Haenelt, Karin (2006). Tracing des Bron/Kerbosch-Algorithmus zur Entdeckung aller maximalen Cliquen in Graphen. Kursskript. 10.08.2006. Darstellungsform 1: http://kontext.fraunhofer.de/haenelt/kurs/folien/Haenelt_TraceCliqueBornKerbosch_Hoch.pdf und Darstellungsform 2: http://kontext.fraunhofer.de/haenelt/kurs/folien/Haenelt_TraceCliqueBornKerbosch_Quer.pdf

Copyright

Versionen: 24.11.2012, 28.11.2011,27.11.2011, 14.11.2009, 10.11.2007, 02.11.2007, 14.11.2006, 10.08.2006