Kryptoalgorithmen unter Ausnutzung der Komplexität des Clique-Problems

Dr. R. Winter
Institut für Informatik
Martin-Luther-Universität Halle-Wittenberg
D–06120 Halle (Saale)

Abstract
We consider possibilities of graph constructions to encode messages. The hardness of decoding corresponds to solve a clique problem.

1. Einleitung

Die Sicherheit des Clique-Algorithmus zur Ver- und Entschlüsselung basiert auf der NP-Vollständigkeit des Clique-Problems. Starke NP-Vollständigkeit und Aussichtslosigkeit hinsichtlich eines akzeptablen Ergebnisses bei Verwendung von Approximationsverfahren verdeutlichen die besondere Schwierigkeit beim Finden von Cliquen in allgemeinen Graphen.

Es werden Möglichkeiten angegeben, jeden Klartext in einen Graphen zu verwandeln, so dass die Berechnung der maximalen Clique, enthaltend die zu übertragende Information, aussichtslos erscheint. Der reguläre Nachrichtenempfänger kann den erhaltenen Graphen mittels Privatschlüssel soweit abändern, dass die maximale Clique offensichtlich wird.

2. Grundlegende Definitionen und Sätze

Das Clique-Problem ist wie folgt gegeben: **Definition 2.1:**

CLIQUE 1 (Entscheidungsvariante)
Gegeben sei ein Graph $G = [V, E]$ *mit Knotenmenge* V *und Kantenmenge* $E \subseteq V \times V$, $|V| = n, k \in N, k < n$.
Hat G *eine k-clique, d.h. existiert ein* $V' \subseteq V$ *mit* $|V'| = k$, *so dass für jedes Paar* $v_1, v_2 \in V'$ $[v_1, v_2] \in E$ *gilt ?*

CLIQUE 2 (Zahlvariante)
Gegeben sei ein Graph $G = [V, E]$.
Bestimme die maximale Cliquezahl k !

CLIQUE 3 (Optimierungsvariante)
Gegeben sei ein Graph $G = [V, E]$.
Gib die größte Clique V' *an!*

CLIQUE1 ist ein NP-vollständiges Problem. [WEGENER] Derzeit ist kein Polynomialzeitalgorithmus bekannt zur Entscheidung, ob ein beliebiger Graph G eine k-clique hat. Bekannte Lösungsmethoden laufen darauf hinaus, dass im Wesentlichen jede der $\binom{n}{k}$ k-elementigen Knotenteilmengen auf Clique-Eigenschaft getestet werden. Die Schwierigkeit beruht nicht auf der Größe einzelner Zahlen, da die Knoten fortlaufend von 1 bis n numeriert werden können. Somit ergibt eine Beschränkung der zulässigen Zahlengröße keine Vereinfachung und CLIQUE1 ist stark NP-vollständig. Pseudopolynomielle Algorithmen gibt es für CLIQUE1 nicht, es sei denn $P = NP$ gilt. [WEGENER]
Aus der PCP-Theorie wissen wir, dass das Clique-Problem auch nicht durch Approximationsalgorithmen mit akzeptablem Resultat gelöst werden kann. [MAYR/PRÖMEL/STEGER]

Satz 2.1:

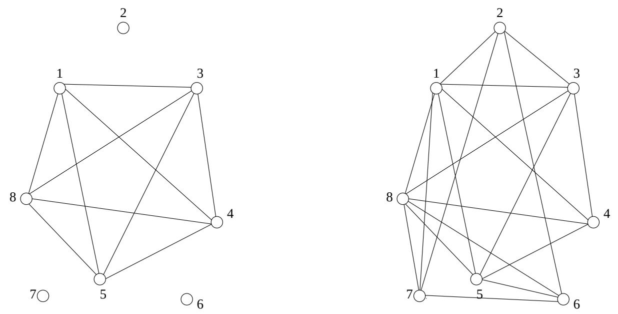
$\varepsilon > 0$ sei eine beliebige Konstante.
Es gibt keinen Polynomialzeit-Approximationsalgorithmus für CLIQUE2 mit Güte $n^{1-\varepsilon}$, falls $NP \neq co - RP$ gilt.
Es gibt keinen Poynomialzeit-Approximationsalgorithmus für CLIQUE2 mit Güte $n^{1/2-\varepsilon}$, falls $NP \neq P$ gilt.[HASTAD]

3. **Prinzip des Clique-Kryptosystems**

Die zu übermittelnde Nachricht (Klartext) $M \in \{0,1\}^*$ wird in Blöcke der Länge n , $n < |M|$ unterteilt. Ein eventuell kürzerer Endblock ist durch ein geeignetes Paddingverfahren aufzufüllen.

Jeder Klartextblock $B = b_1 b_2 ... b_n \in \{0,1\}^n$wird als Clique-Problem eines Graphen $G = [V, E]$ mit mindestens n Knoten angesehen, wobei jedem $b_i \in \{0,1\}$ ein Knoten $v_i \in V'$ zugeordnet wird. Die Anzahl $k = b_1 + b_2 + ... + b_n$ der Einsen im Block soll die maximale Cliquezahl sein, d.h. $b_i = 1 \leftrightarrow v_i \in V'$ für eine maximale k-clique V'.

Bild 3.1:
$M : 10111001$
$n = 8$
$k = 5$
Graphen G_1 und G_2



Geeignete Verfilzung (vgl. [EGGERS]) des Graphen G_1 durch Hinzufügen von Knoten und Kanten bei Erhalt der maximalen k-clique und eine Knotenpermutation π liefern einen Graphen G_2 mit versteckter k-clique. G_2 ist der verschlüsselte Code, der offen verschickt wird. Ohne Zusatzwissen ist einem Abhörer das Finden einer maximalen Clique nur schwer möglich.

Die effiziente Entfilzung von G_2 durch den regulären Empfänger führt zu einem Graphen G_1 bzw. G_1^*, dessen maximale Clique (aus k Knoten) leicht zu berechnen ist.

$$B \underbrace{\rightarrow G_1 \stackrel{G}{\rightarrow} G_1 \oplus G \stackrel{\pi}{\rightarrow}}_{Codierung} G_2 \underbrace{\stackrel{\pi^{-1}}{\rightarrow} G_2^* \stackrel{G'}{\rightarrow} G_1^* = G_1 \rightarrow}_{Decodierung} B$$

Will man den Verfilzungsmechanismus G öffentlich bekanntgeben, so sollte die Berechnung von G' (zur Entfilzung) aus G schwierig sein. Wir benötigen dazu eine Einwegfunktion f, die in Polynomialzeit G aus G' berechnet.

$G' \xrightarrow{f} G$, wobei f leicht berechenbar, d.h. in Polynomialzeit
$G \xrightarrow{f^{-1}} G'$, wobei f^{-1} schwer berechenbar

Alle bisher bekannten Kandidaten für Einwegfunktionen basieren auf dem Problem DISKRETER LOGARITHMUS bzw. Faktorisierung großer Zahlen [REISCHUK], so dass an dieser Stelle ein Verwenden des RSA-Algorithmus sinnvoll erscheint [SCHNEIER]. Ein Finden anderer Einwegfunktionskandidaten f könnte den RSA-Algorithmus ersetzen.

4. **Verfilzung von Graphen**

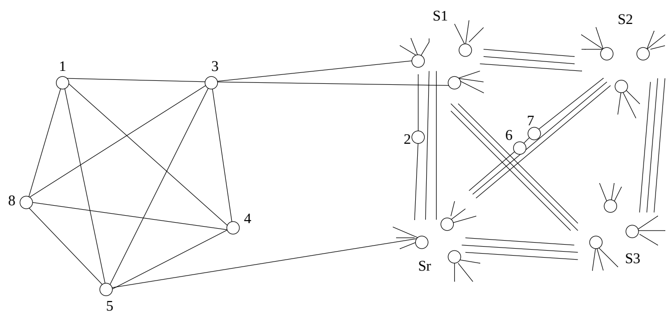
Aus dem Klartextblock der Länge n erhalten wir G_1 bestehend aus einem vollständigen k-elementigen Untergraph und $n-k$ isolierten Knoten.
Eine Knotenpermutation $\pi(12...l) = \pi(1)\pi(2)...\pi(l)$ kann als natürliche Zahl $\pi' = \pi(1)0\pi(2)0...0\pi(l)$ angegeben werden, z.B. $\pi(1\ 2\ 3\ 4\ 5\ 6\ 7...) = 12\ \ 4\ \ 38\ \ 7\ \ 18\ \ 5\ \ 3...$ entspricht $\pi' = 1204038070180503...$, wenn wir o.B.d.A. alle Knotennummern aus $N' = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}^*$ wählen.

π' und $r, s \in N$ (oder $r00s00\pi' \in N$, wobei r und s o.B.d.A. $\in N'$ sind,) werden geheim übertragen.

Wir geben einige Möglichkeiten der Verfilzung von G_1 an:

1.
   - Ein anzufügender Graph G enthält die Knotenmengen $S_i = \{v_{ij}; j = 1(1)s\}$ für $i = 1(1)r$. Knoten der gleichen Menge S_i sind nicht verbunden, Knoten verschiedener Mengen S_i, S_j werden jeweils paarweise verbunden.
   - Die isolierten Knoten von G_1 werden beliebig auf soeben erzeugten Kanten platziert.
   - Jeder Clique-Knoten kann wahlweise probabilistisch mit Knoten aus einer Menge S_i verbunden werden, wobei höchstens $k-1$ Clique-Knoten auf dieselbe Menge S_i zielen dürfen.

Bild 4.1:



Lemma 4.1:

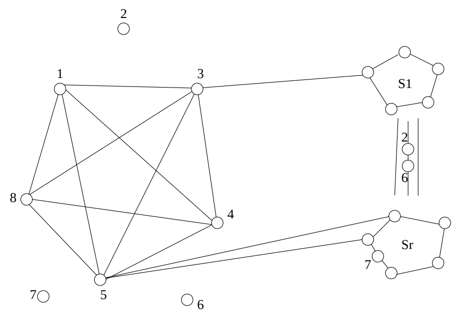
Für $r < k < s\ r$ ist der soeben konstruierte Graph invariant bzgl. maximaler Clique.

2. • Der anzufügende Graph enthält die Knotenmengen $S_i = \{v_{ij}; j = 1(1)s\}$ für $i = 1(1)r$. Knoten einer Menge S_i bilden einen Ring. Knoten verschiedener Mengen S_i, S_j sind paarweise verbunden.

   • Die isolierten Knoten aus G_1 werden beliebig auf soeben erzeugten Kanten (einschließlich der Ringe) platziert.

   • Jeder Clique-Knoten kann wahlweise probabilistisch mit Knoten eines Ringes S_i verbunden werden, wobei höchstens $k-2$ Clique-Knoten auf denselben S_i-Ring zielen.

Bild 4.2:



Lemma 4.2:

Für $2r < k < s\ r$ ist der konstruierte Graph invariant bzgl. maximaler Clique.

Damit die Verschlüsselung von r, s und π' mittels RSA-Verfahren nicht bereits aufwendiger wird als eine direkte Verschlüsselung des ursprünglichen Klartextes mit RSA, empfehlen wir folgendes Vorgehen:

Betrachten wir zunächst r und s.

Lemma 4.3:

> *Gegeben sei ein RSA-Algorithmus mit Laufzeit $T(n) = p(n)$ für ein Polynom p zur Verschlüsselung und geheimen Entschlüsselung durch den regulären Empfänger. Dann können Zahlen $r, s \leq 2^{c \log n}$ für eine Konstante $c > 0$ in polylogarithmischer Zeit mit RSA verschlüsselt und privat entschlüsselt werden.*

Beweis:
$|r|, |s| = \log(2^{c \log n}) = c \log n \log 2 = c' \log n$
$T(|r|), T(|s|) \leq T(c' \log n) = p(c' \log n)$

□

Durch einen Klartextblock der Länge n wird im allgemeinen eine Zahl $\Theta(2^n)$ dargestellt.

Die Verwendung von RSA für r und s gemäß Lemma 4.2 ist also sinnvoll. Aus Sicherheitsgründen wird empfohlen, s dabei möglichst groß zu wählen.

Nun sehen wir uns die Übertragung von π' an.

Aus Klartextblocklänge n folgen n Graphknoten in G_1, so dass mit $n!$ möglichen Permutationen $|\pi'| = \log(n!) \leq n \log n$ gilt. Offenbar ist eine RSA-Anwendung auf π' bei jeder Klartextblockübertragung aus komplexitätstheoretischer Sicht unsinnig.

Folgende Alternativen werden angeboten:

- Beibehaltung eines π' für mehr als $O(\log n)$ Klartextblockübertragungen (Sicherheit problematisch), Verwendung des RSA-Algorithmus zur Übertragung von π' (public-key)
- Verwendung eines symmetrischen Verfahrens zur Übertragung von π', d.h. π' bleibt geheim

- π' wird durch eine geheime analytische Funktion beschrieben (symmetrisch) oder geändert und nur eventuelle Parameter dieser Funktion oder ein anfängliches π' werden per RSA übertragen.

Andererseits kann man neben π' die Zahl k des k-elementigen vollständigen Untergraphen in G_1 geheim, z.B. mittels RSA-Algorithmus, übertragen und anschließend G_1 wie folgt verfilzen:

3.
- Der Graph G_1 wird durch zufälliges Knoten- und Kantenanfügen verfilzt mit der Bedingung, dass keine größere Clique entstehen darf.
- Wir lassen wahlweise maximal $k-2$ Kanten aus den isolierten Knoten in G_1 austreten mit beliebigem Zielknoten, jedoch so, dass keine k-clique neu gebildet wird. (Isolierte Knoten können dann auch einen Grad $> k-2$ erhalten.)
- Durch beliebiges Hinzufügen weiterer Knoten und Kanten unter Berücksichtigung, dass keine k-clique entsteht, ist eine weitere Graph-Verfilzung möglich.
- Die Knotennummern werden beliebig permutiert. Es entsteht ein Graph G_2 mit Cliquezahl k, dessen k-clique bei Unkenntnis von k im allgemeinen schwer zu berechnen ist.

5. **Entschlüsselung**

Die maximale Clique als Information für den Geheimtextblock ist aus dem verfilzten Graph G_2 schwer erhältlich. Bei Kenntnis des geheimen Schlüssels (π', r, s) und Verfilzungsmethode 1 oder 2 wendet man zunächst π^{-1} (aus π') auf den verfilzten Graphen G_2 an. Im entstehenden Graph G_2^* interessiert nur noch der Teilgraph $G_1^* = [V_1^*, E_1^*]$ der ersten n Knoten und zugehörigen Kanten. In G_1^* testen wir nacheinander die Knoten $1, 2, ..., n$ bzgl. Dreierverbindung.
Für $[i,j] \in E_1^*, [j,m] \in E_1^*, [i,m] \in E_1^*$ sind i, j, m in der gesuchten Clique.
Isolierte Knoten haben nach Konstruktion in G_1^* zu keinem Cliquepunkt Verbindung, und es sind keine drei isolierten Knoten paarweise untereinander verbunden. Nach $O(n^2)$ Tests ist die maximale Clique in G_1^*, d.h. in G_1 gefunden. (O.B.d.A. sei n durch 3 teilbar, ansonsten werden noch $n-1$ bzw. n auf Dreiecksbeziehung mit 1 bzw. 2 getestet.)

Bei Verfilzungsmethode 3 gehen wir wie folgt vor:
Bei Kenntnis des Geheimschlüssels k kann die maximale Clique V_2' von G_2 leicht ermittelt werden. Nach Anwendung von π^{-1} auf V_2' entsteht aus der k-clique der gesuchte Klartextblock.

Lemma 5.1:

> *Ist in einem Graph G_2 die Cliquezahl k bekannt, so läßt sich die maximale Clique in Polynomialzeit berechnen.*

Beweis:
Graph G_2 und maximale Cliquezahl k seien gegeben.
Für $|V_2| = n$ sind maximal $\binom{n}{m} = \frac{n!}{k!(n-k)!} \leq (n-k+1)^k$ k-elementige Teilmengen $V_2' \in V_2$ auf k-clique-Eigenschaft zu testen. Es genügt, Teilmengen V_2' mit Knoten vom Grad $\geq k-1$ zu untersuchen.

□

Bei Kryptosystemen mit einer der vorgestellten Verfilzungstechniken für Graphen sind zur leichten Entschlüsselung durch den Empfänger

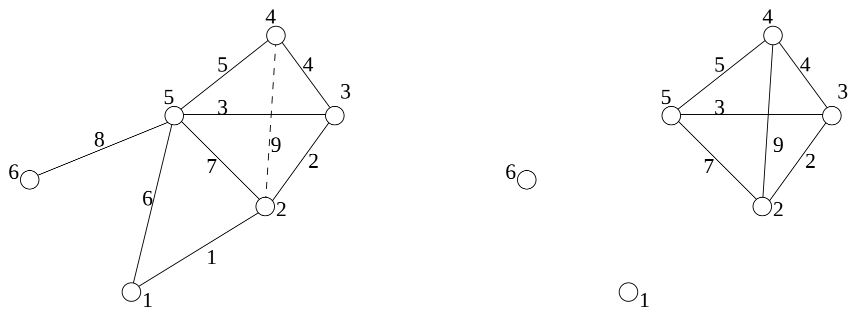
private Schlüssel π', r, s bzw. k verwendbar.
Bei symmetrischer Verschlüsselungstechnik [SCHNEIER] [WOBST] wird der private Schlüssel bereits zur Verschlüsselung des Klartextes verwendet. Nur mit dessen Kenntnis ist eine Codierung möglich.
Soll das Kryptosystem als Public-key-Verfahren [SCHNEIER] [WOBST] laufen, so sind die Zahlen π', r, s bzw. k gemäß Einwegfunktion f zu verschlüsseln (z.B. mit RSA-Verfahren [SCHNEIER]). Die Ergebnisse $f(\pi'), f(r), f(s)$ bzw. $f(k)$ stehen öffentlich zur Verschlüsselung von Klartext für jedermann zur Verfügung. Jedoch kann wahrscheinlich nur der reguläre Empfänger π', r, s bzw. k zurückgewinnen (mittels eines privaten Schlüssels).

6. **Ein weiteres Kryptosystem**

Wir stellen ein symmetrisches Kryptosystem vor. Als geheimen Schlüssel verwenden wir einen zufällig erstellten Graphen $G = [V, E]$, in dem ca. 50 % der möglichen Kanten vorhanden sind. $|V| = n$ sei beliebig gewählt. Eine geheime Ordnung aller möglichen (vorhandenen und nicht vorhandenen) Kanten wird erstellt. Wir betrachten einen Klartextstring aus $\{0, 1\}^*$ (oder über einem anderen endlichen Alphabet) und verschlüsseln ihn wie folgt:

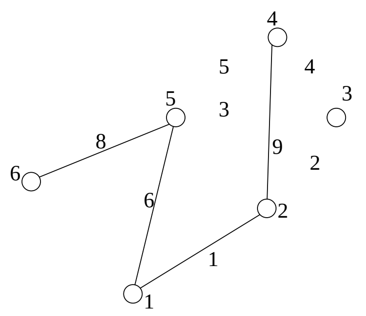
- Eine (nicht notwendig geheime) Funktion g berechnet für jedes Klartextzeichen eine Zahl k , die die Größe der Clique angibt.
- In einem Graphen G^0 der Größe n werden zufällig k Knoten für eine k-clique ausgewählt.

Bild 6.1:
G und G^0



- G^0 wird über den geheimen Schlüssel G gelegt, wobei wir jede Kante aus G mit der korrespondierenden Kante aus G^0 addieren (unter Verwendung der XOR-Funktion).

Bild 6.2:
G XOR G^0

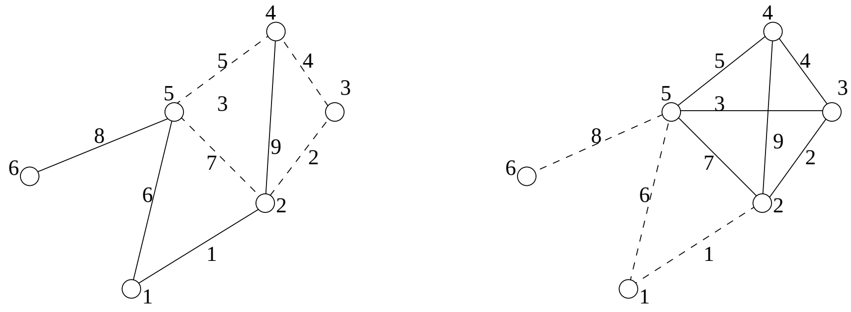


- Wir erstellen einen Bitstring b, der die Kanten des Graphen G XOR G^0 nach der geheimen Ordnung mit 1 oder 0 beschreibt, je nachdem, ob die Kante existiert oder nicht.

1	2	3	4	5	6	7	8	9	10	11	...	15
1	0	0	0	0	1	0	1	1	0	0	...	0

- b wird dem Empfänger zugesandt.
- Zur Entschlüsselung baut der Empfänger den Graphen G^0 mit Kenntnis der Kantenordnung wieder auf. Dabei wird der geheime Schlüssel G über G XOR G^0 gelegt (mit XOR).

Bild 6.3:
$G\textsc{xor}G^0$ und $(G\textsc{xor}G^0)\textsc{xor}G$



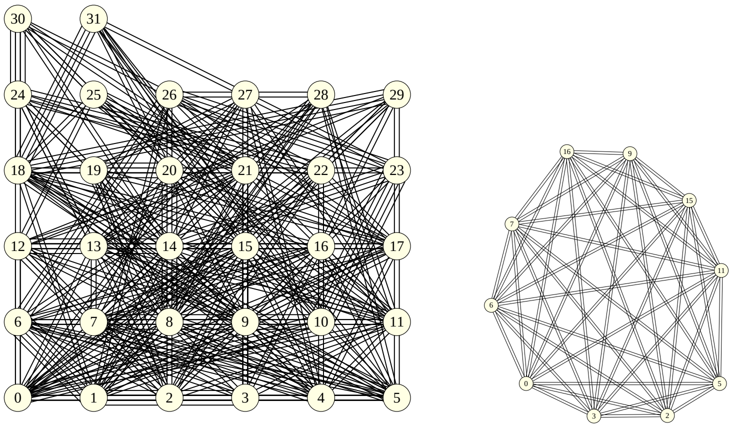
- Wir haben so wieder den k-clique-Graph G gefunden. Man braucht nur noch von einem Knoten mit Grad ≥ 1 die Gradzahl bestimmen und hat damit $k-1$. Aus k ist mit g^{-1} das Klartextzeichen zurückgewonnen.

Unter der Voraussetzung, dass die Grösse n und der Graph G geheim bleiben, ist es schwer, die Nachricht zu entschlüsseln, da man von der abgefangenen Nachricht nicht auf die Größe des geheimen Graphen schließen kann. Die geheime Kantenordung verhindert ein leichtes Erkennen der Clique. Das mehrfache Versenden der gleichen Nachricht führt zu verschiedenen Bitströmen, so dass eine Kryptanalyse weiter erschwert wird.
Alternativ kann man auch mehrere Cliquen in den Graph G^0 einbetten. Um dem Entschlüsseler die Clique-Suche zu erleichtern, sollten alle Cliquen eine eigene Zusammenhangskomponente [SIMON] darstellen.

7. Implementierung von Clique-Algorithmen mit LEDA

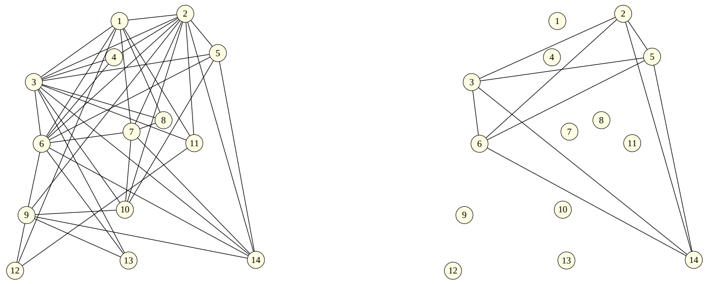
Gemäß Methode 3 zur Verfilzung wurde das Kryptosystem in LEDA [MEHLHORN, NÄHER] implementiert. Ein Klartextblock aus $\{0,1\}^{32}$ (4 Byte) ist einzulesen (Abwandlung der Blocklänge im Programmcode möglich).
Es wird ein Graph G_1 konstruiert, in dem diejenigen Knoten eine Clique bilden, deren korrespondierende Bits im Code 1 gesetzt sind. Die anderen Knoten werden probabilistisch so verknüpft, dass keine neue Clique dieser Größe entsteht.
Die Entschlüsselung durch Finden der maximalen Clique erfolgt unter Zuhilfenahme des Geheimschlüssels Cliquezahl.
Auf die Vergrößerung des Graphen G_1 zur verstärkten Verfilzung wurde aus Platzgründen bei der Implementierung verzichtet.

Bild 7.1:
Nachricht $M = 101101, |M| = 6$
Graph G_1 (Unter den ersten 6 Knoten bilden $0, 2, 3, 5$ die Clique, woraus der Klartext M zurückgewonnen wird.)



Das im Abschnitt 6 beschriebene Kryptosystem wurde ebenfalls mit LEDA implementiert.
Bei der Erstellung des geheimen Schlüssels nehmen wir keine zusätzliche Kantenordnung, wie im Algorithmus beschrieben, vor, da der Graph ohnehin zufällig erstellt wird.

Bild 7.2:
$k = 5$ in einem Graphen G^0 als Clique verschlüsselt



8. Schlussfolgerungen

Die reine Verwendung eines Public-key-Algorithmus (z.B. RSA) zur Chiffrierung eines Klartextes erfordert viel Zeit.
Wir übertragen stattdessen nur kurze private Schlüselelemente (z.B. r, s, k) mit einem RSA-Algorithmus. Längere private Schlüsselteile werden geheim (mit symmetrischen Verfahren) übertragen.
Versteckt in große Graphen versenden wir den Klartext, wobei ein Knacken des Codes dem Finden einer maximalen Clique in einem großen Graph oder der Entfilzung des Graphen zu einem einfachen Graph mit offensichtlicher maximaler Clique entspricht.
Insgesamt nutzt das Kryptosystem Public-key- wie auch symmetrische Verfahren einerseits, und die Komplexität des Clique-Problems wie auch der Faktorisierung großer Zahlen andererseits.
Der Zusammenhang zwischen umfangreicher Übertragungsmenge (Graph G_2), entsprechender Zeit- und Platzkomplexität und erhöhter Sicherheit sowie die daraus resultierende Wahl der geeigneten Graph-Verfilzungsmethode ist entsprechend dem jeweiligen Verwendungszweck abzuwägen. Weitere Verfilzungsmethoden sind vorstellbar.

Dank

Für die Erstellung der LEDA-Quellcodes bedanke ich mich bei Herrn Gregor Erz und Herrn Ralf Hoffmann herzlich.

Literatur

[EGGERS] B. Eggers. *Ordering on Graphs and Computations of CLIQUE.* Technical Report No.91-03, TU Berlin, FB 20, 1991

[HASTAD] J. Hastad. *Clique is hard to approximate within* $n^{1-\varepsilon}$. Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science. p 627-636. 1996

[MAYR/PRÖMEL/STEGER] E. W. Mayr, H. J. Prömel, A. Steger. *Lectures on Proof Verification and Approximation Algorithms.* Springer 1998

[MEHLHORN/NÄHER] K. Mehlhorn, S. Näher. *LEDA.* Cambridge University Press 1999

[REISCHUK] K. R. Reischuk. *Komplexitätstheorie 1.* Teubner 1999

[SCHNEIER] B. Schneider. *Angewandte Kryptographie.* Addison-Wesley 1997

[SIMON] K. Simon. *Effiziente Algorithmen für perfekte Graphen.* Teubner 1992

[WEGENER] I. Wegener. *Theoretische Informatik.* Teubner 1993

[WOBST] R. Wobst. *Abenteuer Kryptologie.* Addison-Wesley 1997