Efficient Enumeration of All Chordless Cycles in Graphs

Elisângela Silva Dias *, Diane Castonguay, Humberto Longo, and Walid Abdala Rfaei Jradi **

Instituto de Informática, Universidade Federal de Goiás, Alameda Palmeiras, Quadra D, Campus Samambaia, Caixa Postal 131 - CEP 74001-970 - Goiânia, Goiás, Brasil
{elisangela,diane,longo,walid}@inf.ufg.br
http://www.inf.ufg.br

Abstract. Enumerating chordless cycles is a theoretical important problem in the Graph Theory area. It also can be applied to practical problems, such as discover which predators compete for the same food in ecological networks. Motivated by theoretical interest of the problem and also by its significant practical importance, we show in this paper a new algorithm to enumerate all chordless cycles in undirected graphs. The proposed algorithm is recursive and based on the depth-first search strategy. It is linear in the number of chordless paths.

Keywords: Graphs. Chordless Cycles. Efficient Algorithm.

1 Introduction

A chordless cycle is an induced subgraph which is a cycle of length greater than 3. Finding chordless cycles of a graph efficiently is an important theoretical problem of graphs and also have its practical applications. One application is a better understanding of the structure of ecological networks, such as food web, which goal is to discover the predators that compete for the same prey (see Sokhn et al. [7]). To reach this aim, the directed graph of food web is transformed in a niche overlap graph to pattern the competition between species. The lack of chordless cycles in the last graph means that the species can be rearranged along a single hierarchy.

Other practical application is presented by Pfaltz [6]. The chordless cycles effectively characterize the connectivity structure of the network as a whole. The number of cycles of length 3 ≤ k ≤ max_length can be used to identify the network, revealing global connectivity patterns.

One solution to the problem of determining whether a graph contains a chordless cycle with k vertices or more, for some fixed value of k ≥ 4, is presented

* Partially supported by FAPEG – Fundação de Amparo à Pesquisa do Estado de Goiás.
** Supported by CAPES – Coordenação de Aperfeiçoamento de Pessoal de Nível Superior.
by Hayward [4]. Similarly, a solution to the problem of determining whether a graph contains a chordless cycle of four or more vertices (without setting \( k \)), known as recognizing chordal graphs, was shown by Golumbic [1]. Spinrad [8] developed an improved solution to this problem.

The chordless cycles of length greater than 4 are also called *holes*. The *antihole* is the complement graph of a graph hole. Holes and antiholes have been extensively studied in many different contexts in algorithmic graph theory. The most notable examples are weakly chordal graphs, also known as weakly triangulated graphs [1], [3], which are those containing no holes or antiholes. Nikolopoulos and Palios [5] showed two algorithms, one for hole detection and other for antihole detection in arbitrary graphs.

Uno [9] proposed an algorithm for finding chordless cycles taking \( O(|E|) \) time for each chordless cycle/path. It was evaluated the performance of the algorithm by computational experiments for random graphs, and showed that the computation time is constant per chordless cycle for not so dense random graphs. It also presented that the number of chordless cycles is quite smaller than the number of cycles for those random graphs.

Wild [10] described an algorithm to generate all cycles of cardinality at most five, using the principle of exclusion. It is easy to adapt the proposed algorithm so as to produce only the chordless cycles.

The most efficient algorithm for enumerating all chordless cycles that we know is described in Sokhn et al. [7]. The general principle of this algorithm is to create expanding paths for each vertex using a depth-first search (DFS) strategy and a vertex ordering, so that all chordless cycles are found twice.

In this paper, we present an efficient algorithm to enumerate all chordless cycles in a graph \( G \), taking \( O(P) \) time, where \( P \) is the number of chordless paths in \( G \). The core idea of our algorithm is also to use a vertex ordering, with whom an arbitrary cycle can be described in an unique way. With this in hand, we generate initial triplets and use the idea of depth-first search to find all chordless cycles only once. We want to clarify that our results, even though based on similar ideas of those presented by Sokhn et al. [7], were obtained independently.

The remaining of this text is organized as follows: some preliminaries definitions and comments are presented in Sect. 2; our algorithm is introduced in Sect. 3; Sect. 4 describes the experimental tests and results produced by the new algorithm compared to other methods; finally, in Sect. 5 we draw our conclusions. The proofs are given in the Appendix.

## 2 Preliminaries

Let \( G = (V, E) \) be a finite undirected simple graph with vertex set \( V(G) \) and edge set \( E(G) \). Let \( n = |V(G)| \) and \( m = |E(G)| \). For \( x \in V(G) \), we denote by \( \text{Adj}(x) \) the set of neighbors of \( x \), that is, \( \text{Adj}(x) = \{ y \in V(G) | (x, y) \in E(G) \} \).

A *path* is a finite sequence of vertices \( \langle v_1, v_2, \ldots, v_k \rangle \) such that \( (v_i, v_{i+1}) \in E \), for all \( i = 1, \ldots, k - 1 \). A *cycle* is a path \( \langle v_1, v_2, \ldots, v_k \rangle \) such that \( (v_k, v_1) \in E \).
A cycle can also be defined as a path of the form \(\langle v_1, v_2, \ldots, v_k, v_1 \rangle\). For the sake of simplicity, we will use this latter description for the algorithm. A path (respectively, a cycle) is said to be \textit{simple} if there is no repetition of vertices. A \textit{chordless path} is an induced subgraph which is a path. Observe that a chordless path is a simple path \(\langle v_1, v_2, \ldots, v_k \rangle\) in which there is no edges between those vertices outside of the ones of the path. A \textit{chordless cycle} is an induced subgraph which is a cycle of length greater than 3. Observe that a chordless cycle is a simple chordless path.

Remark that if \(\langle v_1, v_2, \ldots, v_k \rangle\) is a cycle, so also are \(\langle v_i, v_{i+1}, \ldots, v_k, v_1, v_2, \ldots, v_{i-1} \rangle\) and \(\langle v_i, v_{i-1}, \ldots, v_2, v_1, v_k, \ldots, v_{i+1} \rangle\), for all \(i = 1, \ldots, k - 1\). The minimum degree of \(G\) is denoted by \(\delta(G)\), the maximum degree is denoted by \(\Delta(G)\) and \(\deg_G(v)\) represents the degree of vertex \(v\) in \(G\). We will denote by \(G - u\) the subgraph induced by \(G\) minus vertex \(u\).

In this paper, we will construct an ordering of the set of vertices \(V(G)\), named \textit{degree labeling}. For the sake of simplicity for algorithm and implementation, we consider this ordering as a bijection \(\ell : V(G) \rightarrow \{1, 2, \ldots, n\}\). For this purpose, we also construct a sequences of subgraph and of vertices. Let \(G_1 = G\). We choose inductively \(u_i \in V(G_i)\) such that \(\deg_G(u_i) = \delta(G_i)\) and define \(G_{i+1} = G_i - u_i\). We define \(\ell(u_i) = i\), for all \(i\). Note that the subgraph \(G_i\) is obtained of \(G\) after the elimination of vertices \(\{u_1, \ldots, u_{i-1}\}\).

Denote by \(n_i = |V(G_i)|\) and \(m_i = |E(G_i)|\). Therefore, \(n_i = n - i + 1\) and \(m_{i+1} = m_i - \delta(G_i) = m - \sum_{i=1}^{n} \delta(G_i)\).

Insight of the Lemma \ref{lem:cycle}, we introduce a set of triplets which can begin a possible chordless cycle. Denote by \(T(G)\) the triplets of \(G\) that can be defined as \(T(G) = \{\langle x, u, y \rangle \mid x, u, y \in V(G) \text{ with } x, y \in Adj(u), \ell(u) < \ell(x) < \ell(y) \text{ and } (x, y) \notin E(G)\}\). According to such ordering, we have the following unique way to describe a cycle.

\begin{lemma}
Let \(G\) be an undirected graph and \(\ell : V(G) \rightarrow \{1, 2, \ldots, n\}\) a labeling of \(V(G)\). If \(G\) contains a simple cycle \(\langle v_1, v_2, \ldots, v_k \rangle\), we can suppose that \(\ell(v_2) = \min\{\ell(v_i) \mid i = 1, \ldots, k\}\) and \(\ell(v_1) < \ell(v_3)\). Therefore, this labeling define the cycle in a unique way.
\end{lemma}

One of the interest of this labeling is that if \(G\) a tree, then there is no initial triplets possible, that is, \(T(G) = \emptyset\). Moreover, if \(G\) has a unique cycle, then considering any degree labeling, we have that \(|T(G)| = 1\), avoiding unneeded triplets.

We will need the following property for neighbors of chordless path.

\begin{lemma}
Let \(G\) be an undirected graph. Let \(p = \langle v_1, v_2, \ldots, v_k \rangle\) be a chordless path and \(v \in Adj(v_k)\) such that \(v \neq v_{k-1}\). In this case, observe that \(v \neq v_i\), for all \(i = 1, \ldots, k\). Exactly one of the following occurs:
\begin{itemize}
  \item[(a)] \(p, v\) is a chordless path;
  \item[(b)] \((v, v_{k-1}) \in E\); or
\end{itemize}
\end{lemma}
there exists $i \in \{2, \ldots, k-2\}$ such that $p = \langle v_i, v_{i+1}, \ldots, v_k, v \rangle$ is a chordless cycle.

Using induction and the ordering defined previously, we have the following upper bound for $|T(G)|$.

**Lemma 3.** Let $G$ be a graph with a degree labeling. Then $|T(G)| \leq \frac{(\Delta(G)-1)m}{2}$.

## 3 An Algorithm to Find All Chordless Cycles

In the algorithm ChordlessCycles, we initially calculated the degree labeling for the graph $G$. This function permits that the quantity of triplets is optimized. After this, we compute the set of initials triplets $T(G)$. Recall that elements of this set are triple of the form $(x,u,y)$, where $\ell(u) < \ell(x) < \ell(y)$, $x,y \in \text{Adj}(u)$ and $(x,y) \notin E(G)$. This limits the search space.

The general principle of the proposed algorithm is to create chordless paths for each initial triple of vertices using a depth-first search strategy, which complies with the conditions of chordless path and cycle. Other optimization is the labeling of graph vertices, that makes the Algorithm 1 faster. The algorithm CCVisit is called at most once for each triple in $T(G)$ because as soon as it is chosen, it is deleted from $T(G)$.

Remark that each search of algorithm CCVisit is actually done in the graph $G_{\ell(u)}$. The depth of each search is at most the length of the longest chordless path. Moreover, the amount of called of CCVisit is limited by the number of chordless paths in the graph. Therefore, our algorithm is linear in the number of chordless paths. Actually, the algorithm is $O(P(\ell))$, where $P(\ell)$ is the number of chordless paths $p = \langle v_1, v_2, v_3, \ldots, v_k \rangle$ such that $\ell(v_2) = \min\{\ell(v_i) | i = 1, \ldots, k\}$.

We introduced a function, called blocked, to mark neighbors vertex of the chordless path, except for the first vertex of the path. This function blocks the neighbors that could form a chord with vertices in the chordless path and permits to extends chordless path in a faster way. To improve further more, we extend the chordless path to left, with algorithm PrimeExtends, whenever there is only one neighbor possible, that is only one vertex with blocked equal to zero. This extends the chordless path in an unique way and does not change the number of examined chordless paths $P(\ell)$. Moreover, this extension reduces a lot of searches that have chords, since the extension marked the neighbors of the visited vertices. Doing so, we can find all chordless cycles.

**Lemma 4.** Let $p = \langle x, u, y \rangle$ and $q = \langle v_1, v_2, \ldots, v_s = x \rangle$ be paths in $G$, such that the path $(q, u, y)$ is a chordless path. At the beginning of each execution of algorithm PrimeExtends($p, q, C, key$), the function blocked : $V(G) \rightarrow \mathbb{N}$ is such that, for any vertices $v \in V(G)$, blocked($v$) = $i$ if and only if $v$ is neighbor of $i$ vertices of $\{v_2, \ldots, v_s = x, u, y\}$. Remark, that if $s = 1$, this set of vertex is $\{u, y\}$. 
Observe that, combining the Lemma 4 with Lemma 2, we have that if \( \langle q, u, y \rangle = \langle v_1, v_2, \ldots, v_s = x, u, y \rangle \) is a chordless path, and \( v \in Adj(v_1) \), then \( blocked(v) = 0 \) if and only if \( \langle v, q, u, y \rangle \) is a chordless path and \( blocked(v) = 1 \) and \( (v, y) \in E(G) \) if and only if \( \langle x, u, y, v, q \rangle \) is a chordless cycle. Observe that, in this last case, the cycle \( \langle x, u, y, v, q \rangle \) is equivalent to the cycle \( \langle v, q, u, y, v \rangle \).

**Lemma 5.** Let \( p = \langle u_1, u_2, \ldots, u_t \rangle \) and \( q = \langle v_1, v_2, \ldots, v_s \rangle \) be paths in \( G \) such that \( v_s = u_1 \) and \( \langle v_1, \ldots, v_s = u_1, \ldots, u_t \rangle \) is a chordless path. At the beginning and at the end of each execution of algorithm \( CC\text{-}Visit(p, q, C, \text{key}) \), the function \( blocked : V(G) \to \mathbb{N} \) is such that, for any vertices \( v \in V(G) \), \( blocked(v) = i \) if and only if \( v \) is neighbor of \( i \) vertices of \( \{u_2, \ldots, u_{t-1}, v_2, \ldots, v_s\} \).

Observe that, combining the Lemma 5 with Lemma 2, after calling algorithm \( BlockNeighbors(u_t) \) in \( CC\text{-}Visit(p, q, C, \text{key}) \), we have that if \( \langle v_1, \ldots, v_s = u_1, \ldots, u_t \rangle \) is a chordless path, and \( v \in Adj(u_t) \), then \( blocked(v) = 1 \) if and only if \( \langle p, v \rangle \) is a chordless path or a chordless cycle.

**Theorem 1.** – Correctness of ChordlessCycles(G) algorithm. The algorithm ChordlessCycles(G) finds all the chordless cycles of a graph G.
Algorithm 2: DegreeLabeling($G$)

**Input:** Graph $G$.

**Output:** A labeling of vertices of $G$.

1. foreach $v \in V(G)$ do
   2. $\text{degree}(v) \leftarrow 0$;
   3. $\text{color}(v) \leftarrow \text{white}$;
   4. foreach $u \in \text{Adj}(v)$ do
      5. $\text{degree}(v) \leftarrow \text{degree}(v) + 1$;

6. for $i = 1$ to $n$ do
   7. $\text{min}\_\text{degree} \leftarrow n$;
   8. foreach $x \in V(G)$ do
      9. if (($\text{color}(x) = \text{white}$) and ($\text{degree}(x) < \text{min}\_\text{degree}$)) then
         10. $v \leftarrow x$;
         11. $\text{min}\_\text{degree} \leftarrow \text{degree}(x)$;
   12. $\ell(v) \leftarrow i$;
   13. $\text{color}(v) \leftarrow \text{black}$;
   14. foreach $u \in \text{Adj}(v)$ do
      15. if $\text{color}(u) = \text{white}$ then
         16. $\text{degree}(u) \leftarrow \text{degree}(u) - 1$;

17. return $\ell$.

Algorithm 3: Triplets($G$)

**Input:** Undirected simple graph $G$.

**Output:** Set $T(G)$ of inicial chordless paths of length 3.

1. $T \leftarrow \emptyset$.
2. foreach $u \in V(G)$ do
   3. // Generate all triplets on form $\langle x, u, y \rangle$.
   4. foreach $x, y \in \text{Adj}(u)$ such that $\ell(u) < \ell(y) < \ell(x)$ do
      5. if $(x, y) \notin E(G)$ then
         6. $T \leftarrow T \cup \{ \langle x, u, y \rangle \}$.

6. return $T$.

Algorithm 4: BlockNeighbors($v$)

**Input:** A vertex $v \in V(G)$.

**Output:** Blockade of all vertices on neighborhood of $v$.

1. foreach $u \in \text{Adj}(v)$ do
   2. $\text{blocked}(u) \leftarrow \text{blocked}(u) + 1$. 
Algorithm 5: CC_Visit(p, q, C, key)

Input: Paths $p = \langle u_1, u_2, \ldots, u_t \rangle$ and $q = \langle v_1, v_2, \ldots, v_s \rangle$
such that $v_s = u_1$, $\langle v_1, \ldots, v_s \rangle = u_1, \ldots, u_t$ is a
chordless path; set $C$ of chordless cycles; and
key = $\ell(u_2)$, that is the least value of this
chordless path.

Output: Set $C$ of chordless cycles.

1. BlockNeighbors($u_t$).
2. foreach $v \in \text{Adj}(u_t)$ do
3.  
4.    if $(\ell(v) > \text{key})$ and $(\text{blocked}(v) = 1)$ then
5.      $p \leftarrow \langle p, v \rangle$;
6.      if $(v, v_1) \in E(G)$ then
7.        $C \leftarrow C \cup \{p, q\}$;
8.      else
9.        $C \leftarrow \text{CC_Visit}(p, q, C, \text{key})$;
10.     UnblockNeighbors($u_t$).
11. return $C$.

Algorithm 6: PrimeExtends(p, q, C, key)

Input: Paths $p = \langle x, u, y \rangle$ and $q = \langle v_1, v_2, \ldots, v_s \rangle$
such that the path $\langle q, u, y \rangle$ is a chordless path; set
$C$ of chordless cycles; and key = $\ell(u)$.

Output: New path $q = \langle z, v_1, v_2, \ldots, v_s \rangle$, such that
$\langle q, u, y \rangle$ is a chordless path.

1. counter $\leftarrow 0$;
2. foreach $v \in \text{Adj}(v_1)$ do
3.    if $(\ell(v) > \text{key})$ then
4.      if $(\text{blocked}(v) = 1)$ then
5.        if $(v, y) \in E(G)$ then
6.          $C \leftarrow C \cup \{p, v, q\}$;
7.        else
8.          if $(\text{blocked}(v) = 0)$ then
9.            counter $\leftarrow$ counter + 1;
10.           $z \leftarrow v$;
11.      if (counter = 1) then
12.         BlockNeighbors($v_1$);
13.         $q \leftarrow \langle z, q \rangle$;
14.         $q \leftarrow \text{PrimeExtends}(p, q, \text{key})$.
15. return $q$.
Algorithm 7: UnblockNeighbors($v$)  

**Input**: A vertex $v ∈ V(G)$.  
**Output**: Unblockade of all vertices on neighborhood of $v$.

1. **for each** $u ∈ Adj(v)$ **do**  
   2. **if** $(blocked(u) > 0)$ **then**  
      3. $blocked(u) ← blocked(u) - 1$.

Algorithm 8: UnblockNeighborsPE($q$)  

**Input**: Chordless path $q = ⟨v_1, v_2, \ldots, v_s⟩$.  
**Output**: Unblockade of all vertices on neighborhood of $q$.

1. **for** $i = 2$ **to** $s$ **do**  
   2. UnblockNeighbors($v_i$);

4 Experimental Results

In the implementation, we used an adjacency matrix, which allows the verification of adjacency between two vertices in constant time, and also a compact representation of graphs as proposed by Harish and Narayanan [2].

The implementation and execution of algorithm was performed using C++ and the g++ compiler on Linux Ubuntu 64 bits operating system, a HP Proliant DL380 G7 Xeon Quad Core E5506 2.13GHZ with 40GB of RAM memory and 1.6 TB of disc.

The running time $T$ (in seconds) for the graphs presented in [7], representing some ecological networks, named food web and known graphs are shown in Tables 1 and 2, respectively. Column labeled “Name” is the dataset name, $n$ is the number of vertices, $m$ is the number of edges, $#clc$ refers to the number of chordless cycles and $C_3$ refers to the number of cycles of size 3 in the graph. The time $T_1$ refers to that found by Sokhn et al. [7], $T_2$ refers to the running time of the algorithm of Sokhn et al. [7] in our machine and $T_3$ refers to the running time of our algorithm.

We can observe that the column in Tables 1 and 2 that has the symbol “−” is the graph that not was tested on computer of Sokhn et al. [7], and we present just result of their algorithm running in our computer.

The graphs used provide from known databases of ecological studies, in which the directed graph food web is transformed into undirected graph niche overlap according to the definitions of Wilson and Watkins [11]. However, so that we could make a comparison with the times obtained by Sokhn et al. [7], we also exclude the vertices of degree 0 in the graphs of the Table 1.

In Figure 1 we present the graph Goiânia downtown, which it runtime is in first line in the Table 2. The figure highlights 3 of 9311 chordless cycles present on this graph.
Table 1. Running time to enumerate all chordless cycles on niche-overlap graphs.

| Name       | n   | m   | #clc | C3  | Time T1 | Time T2 | Time T3 |
|------------|-----|-----|------|-----|---------|---------|---------|
| CrystalD   | 16  | 86  | 0    | 293 | 0.000   | 0.000   |
| ChesUpper  | 24  | 85  | 0    | 167 | 0.001   | 0.000   |
| Narragan   | 26  | 168 | 0    | 586 | 0.003   | 0.000   |
| Chesapeake | 27  | 90  | 0    | 157 | 0.000   | 0.000   |
| Michigan   | 29  | 175 | 0    | 587 | 0.002   | 0.000   |
| Mondego    | 30  | 206 | 0    | 886 | 0.003   | 0.001   |
| Cyprwt     | 53  | 842 | 0    | 8946| 0.023   | 0.005   |
| Everglades | 58  | 1214| 0    | 15627| 0.054  | 0.015   |
| Mangrovedry| 86  | 2132| 0    | 30659| 0.054  | 0.015   |
| Floridabay | 107 | 3249| 710  | 15627| 7.054   | 0.015   |

Table 2. Running time to enumerate all chordless cycles on known graphs.

| Name                  | n   | m   | #clc | C3  | Time T1 | Time T2 | Time T3 |
|-----------------------|-----|-----|------|-----|---------|---------|---------|
| Goiânia downtown      | 43  | 75  | 9311 | 5  | 0.536   | 0.076   |
| C100                  | 100 | 100 | 1    | 0  | 0.005   | 0.000   |
| Wheel 100             | 101 | 200 | 1    | 100| 0.000   | 0.000   |
| K50,50                | 50  | 1250| 1500625| 0  | 1.870   | 0.296   |
| Grid 4 × 10           | 40  | 66  | 1823 | 0  | 0.165   | 0.023   |
| Grid 5 × 10           | 50  | 85  | 52620| 0  | 2.759   | 0.365   |
| Grid 6 × 10           | 60  | 104 | 800139| 0  | 44.337  | 5.113   |
| Grid 7 × 10           | 70  | 123 | 8136453| 0  | 735.852 | 66.716  |
| Grid 8 × 10           | 80  | 142 | 71535910| 0 | 9965.260| 837.548 |
| Grid 9 × 10           | 90  | 161 | 280245916| 0| 10643.100| 10643.100|

Fig. 1. Simple representation of Goiânia downtown, Goiás, Brasil.
5 Conclusions

This paper presented an algorithm linear in the number of chordless paths, that finds and lists all chordless cycles of a given undirected graph $G$. The algorithm taking $O(P(\ell))$ time, where $P(\ell)$ is the number of chordless paths $p = \langle v_1, v_2, v_3, \ldots, v_k \rangle$ such that $\ell(v_2) = \min\{\ell(v_i) | i = 1, \ldots, k\}$.

We also performed some experiments using graphs of real applications in ecological studies, namely food web graphs, to know whether some species can be rearranged along a single hierarchy.

The algorithm is based on some depth-first search strategy optimizations, including restrictions on chordless paths and cycles. An important advantage of the algorithm that it finds the same chordless cycle only once and limits the search by constructing initial triplets that can lead to chordless cycles.

Acknowledgments. We would like to thank Nayla Sokhn, which kindly helped us in understanding of the practical applications of chordless cycles in ecological networks and provided her database and algorithm for comparison.

References

1. Golumbic, M.C.: Algorithm Graph Theory and Perfect Graphs. Academic Press (1980)
2. Harish, P., Narayanan, P.J.: Accelerating large graph algorithms on the GPU using CUDA. In: Proceedings of the 14th International Conference on High Performance Computing, HiPC’07, pp. 197–208. Springer-Verlag, Goa, India (2007)
3. Hayward, R.B.: Weakly Triangulated Graphs. J. Comb. Theory Ser. B 39, 200–208 (1985)
4. Hayward, R. B.: Two Classes of Perfect Graphs. PhD Thesis, School of Computer Science, McGill Univ. (1987)
5. Nikolopoulos, S.D., Palios, L.: Hole and Anti-hole Detection in Graphs. In: Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete algorithms, SODA ’04 27, 850–859 (2004)
6. Pfaltz, J.L.: Chordless Cycles in Networks. In: ICDE Workshops (2013)
7. Sokhn, N., Baltensperger, R., Bersier, L.F., Hennebert, J., Nitsche, U.U.: Identification of Chordless Cycle in Ecological Networks. In: COMPLEX 2012 (2012)
8. Spinrad, J.P.: Finding Large Holes. Inform. Process. Letters 39, 227–229 (1991)
9. Uno, T.: An Output Linear Time Algorithm for Enumerating Chordless Cycles. In: 92nd SIGAL of Information Processing Society Japan, pp. 47–53 (2003)
10. Wild, M.: Generating all Cycles, Chordless Cycles, and Hamiltonian Cycles with the Principle of Exclusion. J. Discrete Algorithms, 6(1): 93–102 (2008)
11. Wilson, R.J., Watkins, J.J.: Graphs: An Introductory Approach. Wiley, Michigan University (1990)
Appendix: Proofs of the Lemmas

Lemma 1. Let $G$ be an undirected graph and $\ell : V(G) \to \{1, 2, \ldots, n\}$ a labeling of $V(G)$. If $G$ contains a simple cycle $\langle v_1, v_2, \ldots, v_k \rangle$, we can suppose that $\ell(v_2) = \min\{\ell(v_i) \mid i = 1, \ldots, k\}$ and $\ell(v_1) < \ell(v_2)$. Therefore, this labeling define the cycle in an unique way.

Proof. As said before, any cycle $\langle v_1, v_2, \ldots, v_k \rangle$ can also be described as $\langle v_i, v_{i+1}, v_{i+2}, \ldots, v_k, v_1, v_2, \ldots, v_{i-1} \rangle$ or $\langle v_i, v_{i+1}, v_{i+2}, \ldots, v_k, v_2, v_3, \ldots, v_{i-2} \rangle$ or $\langle v_{i+1}, v_{i+2}, v_{i+3}, \ldots, v_k, v_1, v_2, \ldots, v_{i-2} \rangle$ or $\langle v_{i+1}, v_{i+2}, v_{i+3}, \ldots, v_k, v_2, v_3, \ldots, v_{i-1} \rangle$ for all $i = 1, \ldots, k$. Let $i$ be such that $\ell(v_i) = \min\{\ell(v_j) \mid j = 1, \ldots, k\}$. For this condition, we have two possible labels left $\langle v_{i-1}, \ldots, v_k, v_1, v_2, \ldots, v_{i-2} \rangle$ or $\langle v_{i+1}, \ldots, v_k, v_1, v_2, \ldots, v_{i+2} \rangle$. Since the neighbors of $v_i$, in the cycle, are $v_{i-1}$ and $v_{i+1}$, exactly one of the above label satisfies the second condition.

Lemma 2. Let $G$ be an undirected graph. Let $p = \langle v_1, v_2, \ldots, v_k \rangle$ be a chordless path and $v \in \text{Adj}(v_k)$ such that $v \neq v_{k-1}$. In this case, observe that $v \neq v_i$, for all $i = 1, \ldots, k$. Exactly one of the following occurs:

(a) $(p, v)$ is a chordless path;
(b) $(v, v_{k-1}) \in E$; or
(c) there exists $i \in \{2, \ldots, k-2\}$ such that $p = \langle v_i, v_{i+1}, \ldots, v_k, v \rangle$ is a chordless cycle.

Proof. First, suppose that there exists $i \in \{1, \ldots, k-2\}$ such that $v = v_i$, but, in this case, we have an edge between $v_i$ and $v_k$ which is a contradiction to the fact that $p$ is a chordless path. Suppose that $(p, v)$ is a not a chordless path and that $(v, v_{k-1}) \notin E$. Since $p$ is a chordless path, there exists $i \in \{1, \ldots, k-2\}$ such that $(v, v_i) \in E$. Choosing the biggest $i$ with this property, we have the desire chordless cycle.

Lemma 3. Let $G$ be a graph that use degree labeling. Then $|T(G)| \leq \frac{(\Delta(G)-1)m}{2}$.

Proof. Recall that $T(G) = \{(x, u, y) \text{ such that } x, u, y \in V(G) \text{ with } x, y \in \text{Adj}(u), \ell(x) < \ell(u) < \ell(y) \text{ and } (x, y) \notin E(G)\}$. We will proof the result by induction on $n = |V(G)|$.

Clearly, if $n = 1$ the result holds true. Suppose that $n > 1$. Recall that $\text{degree}_G(u_1) = \delta(G)$, $G_2 = G - u_1$ and $m_2 = m - \delta(G)$. It is not hard to see that $T(G) = T(G_2) \cup \{(x, u_1, y) \text{ such that } x, y \in \text{Adj}(u_1) \text{ with } \ell(x) < \ell(y) \text{ and } (x, y) \notin E(G)\}$.

By induction, we have that $|T(G_2)| \leq \frac{\delta(G_2)-1)\cdot m_2}{2}$. Since $\delta(G_2) \leq \Delta(G)$ and $m_2 = m - \delta(G)$, we got that

$$|T(G_2)| \leq \frac{(\Delta(G) - 1) \cdot (m - \delta(G))}{2} \quad \text{and}$$

$$|T(G)| \leq \frac{(\Delta(G) - 1) \cdot (m - \delta(G))}{2} + \frac{\delta(G) \cdot (\Delta(G) - 1)}{2} = \frac{(\Delta(G) - 1) \cdot (m - \delta(G) + \delta(G))}{2} = \frac{(\Delta(G) - 1) \cdot m}{2}.$$

Therefore, $|T(G)| \leq \frac{(\Delta(G)-1)m}{2}$ and the proof is complete.
Lemma 4. Let \( p = \langle x, u, y \rangle \) and \( q = \langle v_1, v_2, \ldots, v_s \rangle \) be paths in \( G \), such that the path \( \langle q, u, y \rangle \) is a chordless path. At the beginning of each \( \text{PrimeExtends}(p, q, C, \text{key}) \), the function \( \text{blocked} : V(G) \to \mathbb{N} \) is such that, for any vertices \( v \in V(G) \), \( \text{blocked}(v) = i \) if and only if \( v \) is the neighbors of \( i \) of the vertex of \( \{v_2, \ldots, v_s = x, u, y\} \). Remark, that if \( s = 1 \), this set of vertex is \( \{u, y\} \).

Proof. In the first call of \( \text{PrimeExtends}(p, q, C, \text{key}) \), we have increased by 1 the function blocked for all neighbors of \( u \) and \( y \). Thus, the result holds at any call of the algorithm \( \text{PrimeExtends}(p, q, C, \text{key}) \), we have increased by 1 the function blocked for all neighbors of each vertex in \( \{v_2, \ldots, v_s = x, u, y\} \). The next call of \( \text{PrimeExtends}(p, q, C, \text{key}) \) is done after increased by 1 the function blocked for all neighbors of \( v_1 \) and augmenting \( q \) to \( \langle z, q \rangle \), and thus the results still holds.

Lemma 5. Let \( p = \langle u_1, u_2, \ldots, u_t \rangle \) and \( q = \langle v_1, v_2, \ldots, v_s \rangle \) be paths in \( G \) such that \( v_s = v_1 \) and \( \langle v_1, \ldots, v_s = u_1, \ldots, u_t \rangle \) is a chordless path. At the beginning and at the end of each \( \text{CCVisit}(p, q, C, \text{key}) \), the function \( \text{blocked} : V(G) \to \mathbb{N} \) is such that, for any vertices \( v \in V(G) \), \( \text{blocked}(v) = i \) if and only if \( v \) is the neighbors of \( i \) of the vertex of \( \{v_2, \ldots, u_{t-1}, v_2, \ldots, v_s\} \).

Proof. As in the previous lemma, is not hard to see that at any beginning of \( \text{CCVisit}(p, q, C, \text{key}) \), we have increased by 1 the function blocked for all neighbors of each vertex in \( \{u_2, \ldots, u_{t-1}, v_2, \ldots, v_s\} \), as so in the end of it, since we increased by 1 the function blocked for all neighbors of \( u_t \) at the beginning and decreased at the end.

Theorem 2. Correctness of \( \text{ChordlessCycles}(G) \) algorithm.

The algorithm \( \text{ChordlessCycles}(G) \) finds all the chordless cycles of a graph \( G \).

Proof. Let \( C = \langle u_1, \ldots, u_k, u_1 \rangle \) be a chordless cycle of \( G \). By lemma 1 we can assume that \( \ell(u_2) = \min\{\ell(u_i) \mid i = 1, \ldots, k\} \) and \( \ell(u_1) < \ell(u_k) \). Therefore, the triple \( \langle u_1, u_2, u_3 \rangle \) is generated in the Algorithm Triplets(\( G \)). Thus, the algorithm \( \text{ChordlessCycles} \), will perform lines 2 to 17 with \( p = \langle u_1, u_2, u_3 \rangle \).

The algorithm \( \text{PrimeExtends} \) will increased the chordless cycle \( p \) until there is no more vertex to extends or there is more than one possible vertex. In this case, since we have a chordless cycle, we will extends until there is more than one possible vertex. Observe that, combining Lemma 3 with Lemma 2, we have that, if \( \langle q, v_2, u_3 \rangle = \langle v_1, v_2, \ldots, v_s = u_1, u_2, u_3 \rangle \) is a chordless path, and \( v \in \text{Adj}(v_1) \), then \( \text{blocked}(v) = 0 \) if and only if \( v \) is no other neighbors in \( \langle q, u_2, u_3 \rangle \), that is \( \langle v, q, u, y \rangle \) is a chordless path and \( \text{blocked}(v) = 1 \) and \( \langle v, y \rangle \) \( E(G) \) if and only if \( \langle x, u, y, v, q \rangle \) is a chordless cycle. Observe that, in this last case, the cycle \( \langle x, u, y, v, q \rangle \) is equivalent to the cycle \( \langle v, q, u, y, v \rangle \). Therefore, if we did not find the cycle \( C \), there exists \( t \), such that \( q = \langle u_{t+1}, \ldots, u_k, u_1 \rangle \). Let define \( v_i \) such that \( q = \langle v_1, \ldots, v_s \rangle \).

Observe that, combining Lemma 3 with Lemma 2 after calling \( \text{BlockNeighbors}(u_i) \) in \( \text{CCVisit}(p, q, C, \text{key}) \), we have that if \( \langle v_1, \ldots, v_s = u_1, \ldots, u_t \rangle \) is a chordless path, and \( v \in \text{Adj}(u_2) \), then \( \text{blocked}(v) = 1 \) if and only if \( \langle p, v \rangle \) is a chordless path or a chordless cycle. The algorithm will call \( \text{CCVisit} \) until \( i = t \), where it finds the cycle \( C \).