

# ILIGRA: An Efficient Inverse Line Graph Algorithm

Dajie Liu · Stojan Trajanovski · Piet Van Mieghem

Received: 11 December 2012 / Accepted: 3 February 2014 / Published online: 25 April 2014  
© Springer Science+Business Media Dordrecht 2014

**Abstract** This paper presents a new and efficient algorithm, ILIGRA, for inverse line graph construction. Given a line graph  $H$ , ILIGRA constructs its root graph  $G$  with the time complexity being linear in the number of nodes in  $H$ . If ILIGRA does not know whether the given graph  $H$  is a line graph, it firstly assumes that  $H$  is a line graph and starts its root graph construction. During the root graph construction, ILIGRA checks whether the given graph  $H$  is a line graph and ILIGRA stops once it finds  $H$  is not a line graph. The time complexity of ILIGRA with line graph checking is linear in the number of links in the given graph  $H$ . For sparse line graphs of any size and for dense line graphs of small size, numerical results of the running time show that ILIGRA outperforms all currently available algorithms.

**Keywords** Graph algorithm · Line graph · Root graph

## 1 Introduction

A simple graph with  $N$  nodes (vertices) and  $L$  links (edges), denoted by  $G(N, L)$ , is an unweighted, undirected graph containing no self-loops (links starting and ending at the same node) nor multiple links between the same pair of nodes. The *line graph*  $H = l(G)$  of a graph  $G$  is a graph [22] in which every node in  $H$  corresponds to a link in  $G$  and two nodes are adjacent if and only if their corresponding links have a common node in  $G$ . The

---

This research was supported by Next Generation Infrastructures (Bsik).

D. Liu (✉) · S. Trajanovski · P. Van Mieghem  
Faculty of Electrical Engineering, Mathematics and Computer Science,  
Delft University of Technology, P.O. Box 5031, 2600 GA Delft, The Netherlands  
e-mail: totle.liu@gmail.com

S. Trajanovski  
e-mail: s.trajanovski@tudelft.nl

P. Van Mieghem  
e-mail: p.f.a.vanmieghem@tudelft.nl

graph  $G$  is called the *root graph* of  $H$ . The complete graph with three nodes  $K_3$  is a line graph, which has two different root graphs,  $K_3$  and the bipartite graph  $K_{1,3}$ . Except for  $K_3$ , Whitney's theorem [24] states that all connected line graphs have only one root graph up to an isomorphism. Whitney's theorem provides the theoretical basis for the inverse line graph conversion. Cvetković et al. [5] reviewed the state-of-the-art knowledge about line graphs.

There exist plenty of real-world networks that can be modeled by line graphs [16, 17]. A graph is assortative if its low-degree nodes tend to be adjacent with other low-degree nodes and its high-degree nodes tend to be adjacent with other high-degree nodes. Line graphs are assortative and clustered [11, 13, 14, 16]. If two or more communities overlap on a node, it is not feasible to partition nodes to detect communities. In order to detect the overlapping communities, the links are partitioned. However, the algorithms for partitioning links are less efficient than the algorithms for partitioning the nodes. We can transform the networks into their line graphs and partition the nodes of the line graphs [1, 8].

To facilitate the applications of line graphs, the construction of a line graph  $H$  from a root graph  $G$  and the inverse construction from the line graph  $H = l(G)$  to the root graph  $G$  are necessary. The root-to-line graph construction follows straightforwardly from the definition of a line graph [22]. However, the line-to-root graph construction is more complex. Two algorithms for line-to-root graph construction were proposed concurrently by Roussopoulos [20] and Lehot [12]. The algorithm of Roussopoulos is based on the theorem of Krausz [10]: *A graph is a line graph if and only if it is possible to find a collection of cliques in the graph, partitioning all the links, such that each node belongs to at most two of the cliques (some of the cliques can be a single node) and two cliques share at most one node.* Lehot's algorithm employs the principles of van Rooij and Wilf [23]: *A graph is a line graph if and only if it does not have the complete bipartite graph  $K_{1,3}$  as an induced sub-graph, and if two odd triangles<sup>1</sup> have a common link, the sub-graph induced by their nodes is the complete graph  $K_4$ .* Lehot's algorithm first constructs a root graph  $G$  from the given graph  $H$ , and then compares  $l(G)$  and  $H$  to determine whether the given  $H$  is a line graph, unlike Roussopoulos' algorithm, which determines whether the given graph  $H$  is a line graph during the construction of the root graph  $G$ . Naor and Novick [18] proposed a parallel algorithm for line-to-root graph construction based on a divide-and-conquer scheme. Motivated by eigenvectors, Simić [21] proposed an algorithm for recognizing generalized line graphs. Simić's algorithm searches for the maximum degree node in each loop. Degiorgi and Simon [6] proposed a constructive algorithm, based on the Ore's proof [19] of Whitney's theorem [24], which states that two connected and edge-isomorphic graphs with more than four nodes are also node isomorphic and there exists exactly one node isomorphism which generates the given edge isomorphism. The original graph construction examines 2-coloring classes in the input graph components. They showed that their algorithm is more time-efficient than the algorithms of Roussopoulos and Lehot for sparse line graphs and non-line graphs.

In this paper, we propose ILIGRA, an Inverse Line GRaph Algorithm, for line-to-root graph construction. Unlike previous algorithms, ILIGRA checks the connectivity locally. The root graph  $G$  is constructed based on the correspondence between a node in the line graph  $H$  and a link in its root graph  $G$ . Due to the choice of an arbitrary node in the line graph  $H$  and checking the connectivity of its neighbors during the algorithm's execution, ILIGRA is the simplest inverse line graph algorithm proposed so far.

<sup>1</sup>If every node is adjacent to two or zero nodes of a triangle then it is an even triangle.

The paper is organized as follows. ILIGRA is presented in Section 2 and additional details are given in Appendix A. Section 3 demonstrates how ILIGRA works on a descriptive example. Numerical comparisons of ILIGRA with the algorithms of Lehot, Roussopoulos, and Degiorgi and Simon for different types of line graphs are presented in Section 4. Finally, we conclude in Section 5. The link density of line graphs is discussed in Appendix B.

## 2 iligra

### 2.1 Notation

Table 1 summarizes the notation, which is used in the presentation of ILIGRA. According to the definition of the line graph, each node in a line graph  $H(N_H, L_H)$  corresponds to a link in its root graph  $G(N, L)$ . Hence, the number of nodes  $N_H$  in the line graph  $H$  and the number of links  $L$  in the root graph  $G$  are equal,  $N_H = L$ . We always use  $n$  (or  $n$  with subscript) to denote a node in  $H$ . The link in  $G$  corresponding to node  $n$  in  $H$  is denoted by  $l_n$ . In the remainder of the paper, we use  $v$  (or  $v$  with subscript) to denote a node in  $G$ . Denote by  $\mathcal{N}_b(n)$  the set of the nodes in  $H$  which are adjacent to node  $n$  and called the neighbors<sup>2</sup> of node  $n$ . Denote by  $\mathcal{L}_b(l_n)$  the set of the links in  $G$  which corresponds to the nodes in  $\mathcal{N}_b(n)$ . Every link in the root graph  $G$  has two incident nodes.<sup>3</sup> In order to construct the root graph  $G$  from a given line graph  $H$ , we have to determine the two incident nodes of every link in  $G$ . In the root graph  $G$ , we denote by  $v_{l_n}$  the incident node of link  $l_n$  which is first encountered during the algorithm’s execution. The set of the nodes in  $H$ , which corresponds to the links in  $G$  whose incident nodes are not yet determined, is denoted by  $\mathcal{N}_w$ . The set of the nodes in  $H$  corresponding to the links in  $G$  of which one incident node is determined, is denoted by  $\mathcal{N}_h$ .

### 2.2 Concept

The nodes in a line graph  $H(N_H, L_H)$  are denoted by  $n_1, n_2, n_3, \dots, n_{N_H}$ , and the corresponding links in the root graph  $G$  are denoted by  $l_{n_1}, l_{n_2}, l_{n_3}, \dots, l_{n_{N_H}}$ . Initially, it is unknown how the links  $l_{n_1}, l_{n_2}, l_{n_3}, \dots, l_{n_{N_H}}$  connect the nodes in  $G$ , and even the number of nodes  $N$  in  $G$  is unknown.

Suppose that link  $l_{n_1}$  is incident to  $v_1$  and  $v_2$  in  $G$ . From the line graph  $H$ , the set  $\mathcal{N}_b(n_1)$  of the neighbors of node  $n_1$  in  $H$  is known, and the set  $\mathcal{L}_b(l_{n_1})$  of the links in  $G$ , which corresponds to the nodes in  $\mathcal{N}_b(n_1)$ , is also known. By the definition of a line graph, the links in  $\mathcal{L}_b(l_{n_1})$  are the neighboring links of link  $l_{n_1}$ , hence, the links in  $\mathcal{L}_b(l_{n_1})$  should be incident to either  $v_1$  or  $v_2$ . If the links in  $\mathcal{L}_b(l_{n_1})$  which are incident to  $v_1$  are known, the rest of links in  $\mathcal{L}_b(l_{n_1})$  must be incident to  $v_2$ . Unfortunately, it is unknown which links in  $\mathcal{L}_b(l_{n_1})$  are incident to  $v_1$ .

When considering links  $l_{n_2}, l_{n_3}, \dots, l_{n_{N_H}}$ , the same problem appears. The difficulty in constructing the root graph  $G$  lies in partitioning the set of the neighboring links into two complementary subsets of links: the links that are incident to the first incident node of the concerned link, and the other links which are incident to the second incident node of that link.

<sup>2</sup>A node is the neighbor of another node if they are connected.

<sup>3</sup>An incident node of a link is one of the two nodes of that link.

**Table 1** Notation

$G(N, L)$	The root graph with $N$ nodes and $L$ links
$H(N_H, L_H)$	The line graph of $G$ with $N_H$ nodes and $L_H$ links
$n$	The node $n$ in $H$
$\mathcal{N}$	The set of all the nodes in $H$
$\mathcal{N}_w$	The set of the nodes in $H$ , corresponding to the links in $G$ whose incident nodes are not yet determined
$\mathcal{N}_h$	The set of the nodes in $H$ , corresponding to the links in $G$ of which one incident node is determined
$\mathcal{N}_b(n)$	The set of the neighbors of node $n$ in $H$
$l_n$	The link in $G$ which corresponds to node $n$ in $H$
$\mathcal{L}_b(l_n)$	The set of the links in $G$ which correspond to the nodes in $\mathcal{N}_b(n)$
$v_{l_n}$	The first identified incident node of link $l_n$ in $G$
$\text{ADDNODE}(G, v)$	The function which adds a node $v$ to $G$
$\text{ADDLINK}(G, v_1, v_2)$	The function which adds a link $v_1 \sim v_2$ to $G$

Without loss of generality, suppose that  $\mathcal{L}_b(l_{n_1}) = \{l_{n_2}, l_{n_3}, \dots, l_{n_k}\}$ , where  $k$  is an integer. Suppose that the set  $\mathcal{L}_b(l_{n_1})$  of the neighboring links of  $l_{n_1}$  are partitioned successfully into two subsets:  $\mathcal{L}_{b,v_1}(l_{n_1}) = \{l_{n_2}, l_{n_3}, l_{n_4}\}$  where the links are incident to  $v_1$ , and  $\mathcal{L}_{b,v_2}(l_{n_1}) = \{l_{n_5}, l_{n_6}, \dots, l_{n_k}\}$  where the links are incident to  $v_2$ . Then, the set  $\mathcal{L}_b(l_{n_2})$  of the neighboring links of  $l_{n_2}$  is automatically partitioned: the links  $l_{n_1}, l_{n_3}, l_{n_4}$  are incident to  $v_1$ , and the rest of links in  $\mathcal{L}_b(l_{n_2})$  are incident to the second incident node of  $l_{n_2}$ . Similarly, the sets of the neighboring links of links  $l_{n_3}, l_{n_4}, \dots, l_{n_k}$ , are also automatically partitioned. Assuming  $H$  is a connected line graph, the sets of the neighboring links of all the links in  $G$  can be partitioned by iterating the described process. This is the basic idea of ILIGRA.

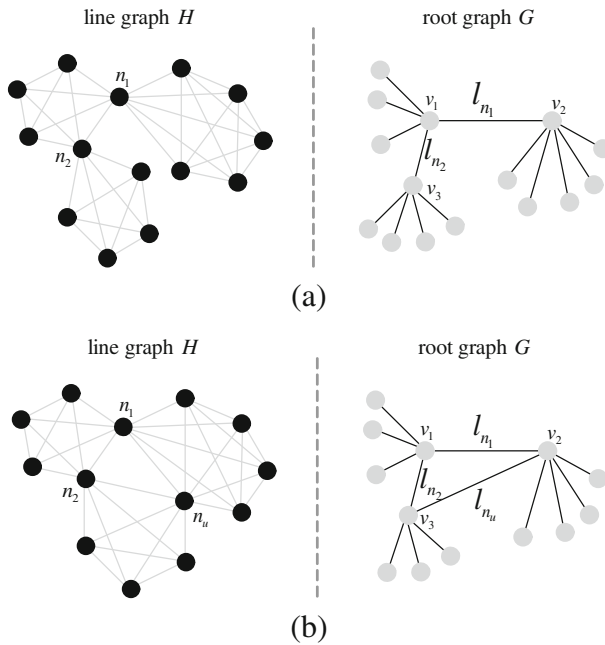
Partitioning the set  $\mathcal{L}_b(l_{n_1})$  of the neighboring links of  $l_{n_1}$  becomes a crucial task in the root graph construction. The theorems in Section 2.3 and Appendix B provide the theoretical basis for this task.

### 2.3 Theoretical Preliminaries

**Theorem 1** *Suppose that two adjacent nodes  $n_1$  and  $n_2$  in  $H$  correspond to links  $l_{n_1}$  and  $l_{n_2}$  in  $G$ , respectively, where  $l_{n_1}$  is incident to  $v_1$  and  $v_2$  and where  $v_1$  is also incident to  $l_{n_2}$ , as shown in Fig. 1a. Then, for each  $n \in \mathcal{N}_b(n_1) \setminus \mathcal{N}_b(n_2)$  in  $H$ , the corresponding link  $l_n$  in  $G$  must be incident to  $v_2$ , and the nodes in  $\mathcal{N}_b(n_1) \setminus \mathcal{N}_b(n_2)$  must form a clique in  $H$ .*

*Proof* For each  $n \in \mathcal{N}_b(n_1) \setminus \mathcal{N}_b(n_2)$ , the corresponding link  $l_n$  in  $G$  has to be incident to either  $v_1$  or  $v_2$ , since  $n$  is adjacent to  $n_1$  in  $H$ . Because  $n$  is not adjacent to  $n_2$ ,  $l_n$  in  $G$  can only be incident to  $v_2$ . Since the corresponding links of all the nodes  $\in \mathcal{N}_b(n_1) \setminus \mathcal{N}_b(n_2)$  are incident to  $v_2$ , the nodes in  $\mathcal{N}_b(n_1) \setminus \mathcal{N}_b(n_2)$  must be fully connected with each other and form a clique in  $H$ . □

Based on Theorem 1, starting with an initial link with nodes  $n_1$  and  $n_2$  in  $H$ , ILIGRA determines the first incident node of the links corresponding to the nodes in  $\mathcal{N}_b(n_1) \setminus \mathcal{N}_b(n_2)$ . The nodes in  $\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$  (i.e. common neighbors of  $n_1$  and  $n_2$  in  $G$ ) may



**Fig. 1** Scenarios in Theorem 1 and 2. Each node (black) in  $H$  corresponds to a link (black) in  $G$

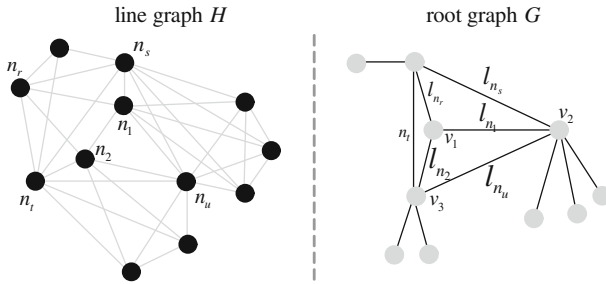
form a clique in  $H$  with the corresponding links being incident to  $v_1$  in  $G$ , as shown in Fig. 1a.

There may also exist a node in  $\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$  which is not adjacent to any other node in  $\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$  and whose corresponding link in  $G$  is incident to  $v_2$  and another node  $v_3$ , as shown in Fig. 1b where the corresponding links of  $n_1$ ,  $n_2$  and that node form a triangle in  $G$ . If there are three or more nodes in the set  $\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$ , we can identify the position of the corresponding link in  $G$  of that node. The last discussion is formalized in Theorem 2.

**Theorem 2** *Suppose that two adjacent nodes  $n_1$  and  $n_2$  in  $H$  correspond to links  $l_{n_1}$  and  $l_{n_2}$  respectively in  $G$ , where  $l_{n_1}$  is incident to  $v_1$  and  $v_2$  and  $l_{n_2}$  is incident to  $v_1$  and  $v_3$ . Suppose that  $|\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)| \geq 3$ . If there exists  $n_u \in \mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$  such that  $n_u$  is not adjacent to any other node in  $\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$ , then link  $l_{n_u}$  must be incident to both  $v_2$  and  $v_3$  in  $G$ .*

*Proof* Since  $n_u \in \mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$ ,  $l_{n_u}$  can be incident to  $v_1$  or be incident to both  $v_2$  and  $v_3$ . If  $l_{n_u}$  is incident to  $v_1$ ,  $n_u$  should be adjacent to at least one other node in  $\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$ , since  $|\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)| \geq 3$ . Because  $n_u$  is not adjacent to any other node in  $\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$ ,  $l_{n_u}$  can only be incident to  $v_2$  and  $v_3$ , as shown in Fig. 1b. The links corresponding to nodes in  $(\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)) \setminus \{n_u\}$  are incident to  $v_1$ . □

If the set of common neighbors of two adjacent nodes  $n_1$  and  $n_2$  have no more than two nodes in  $H$ , then for each node in this set that also has at least 3 additional neighbors



**Fig. 2** Illustration of Theorem 3

different from  $n_1$  and  $n_2$ , the position of its correspondent link in  $G$  can be uniquely determined. This is formalized in Theorem 3.

**Theorem 3** *Suppose that two adjacent nodes  $n_1$  and  $n_2$  in  $H$  correspond to links  $l_{n_1}$  and  $l_{n_2}$  in  $G$ , respectively, where  $l_{n_1}$  is incident to  $v_1$  and  $v_2$  and where  $l_{n_2}$  is incident to  $v_1$  and  $v_3$ , as shown in Fig. 2. If  $|\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)| \leq 2$ , then for each  $n_u \in \mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$ , such that  $|\mathcal{N}_b(n_u) \setminus \{n_1, n_2\}| \geq 3$  and  $\mathcal{N}_b(n_u) \subseteq \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$ , link  $l_{n_u}$  must be incident to both  $v_2$  and  $v_3$  in  $G$ .*

*Proof* Since  $n_u \in \mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$ ,  $l_{n_u}$  can be incident to  $v_1$  or be incident to both  $v_2$  and  $v_3$ . Let us first assume  $l_{n_u}$  is incident to  $v_1$ . Two neighbors  $n_{x1}, n_{x2}$  of  $n_u$  can be adjacent to  $n_1$  or  $n_2$ . However, we have  $|\mathcal{N}_b(n_u) \setminus \{n_1, n_2\}| \geq 3$ , there must be at least one neighbor of  $n_u$  which is not adjacent to either  $n_1$  or  $n_2$ , which contradicts with the fact that  $\mathcal{N}_b(n_u) \subseteq \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$ . Hence,  $l_{n_u}$  can only be incident to  $v_2$  and  $v_3$ .  $\square$

When the set of common nodes of two adjacent nodes  $n_1$  and  $n_2$  in  $H$  has no more than two nodes and a node in this set has no more than two neighbors, different from  $n_1$  and  $n_2$ , then Theorems 2 and 3 are not applicable. We treat those remaining cases in Appendix A.

For a given node  $n_1$ , ILIGRA uses Theorem 1, 2, 3 and Table 3 from Appendix A to determine which links in  $\mathcal{L}_b(l_{n_1})$  are incident to  $v_1$  and which else are incident to  $v_2$ , where  $v_1$  and  $v_2$  are the nodes of  $l_{n_1}$  in  $G$ . Then, for each link in  $\mathcal{L}_b(l_{n_1})$ , the first incident node has been determined.

### 2.4 Algorithm Description

ILIGRA starts by setting  $G$  to an empty graph (line 1). Initially, nothing in  $G$  is determined, hence  $\mathcal{N}_w = \mathcal{N}$  and  $\mathcal{N}_h = \emptyset$  (line 2), where  $\mathcal{N}_w$  and  $\mathcal{N}_h$  are defined in Table 1. Then ILIGRA picks an arbitrary node  $n_1$  in the set  $\mathcal{N}_w$  and picks an arbitrary neighbor  $n_2$  of  $n_1$  in the set  $\mathcal{N}_b(n_1)$  (lines 3–4). Two nodes  $v_1$  and  $v_2$  are added to the root graph  $G$  (line 5), and link  $l_{n_1} = v_1 \sim v_2$  is added to the root graph  $G$  (line 6). Since the incident nodes of link  $l_{n_1}$  have been determined in  $G$ , node  $n_1$  is removed from  $\mathcal{N}_w$  (line 6). Then  $v_1$  is chosen<sup>4</sup> to be

<sup>4</sup>ILIGRA arbitrarily chooses a node from  $v_1$  and  $v_2$  and lets it be incident to  $l_{n_2}$ .

incident to link  $l_{n_2}$  (line 7). Since the first incident node of link  $l_{n_2}$  is determined, node  $n_2$  is moved from  $\mathcal{N}_w$  to  $\mathcal{N}_h$  (line 7).

According to the definition of the line graph, the links in  $\mathcal{L}_b(l_{n_1})$  have a node in common with link  $l_{n_1}$  in  $G$ . Since  $l_{n_1}$  is incident to  $v_1$  and  $v_2$ , the links in  $\mathcal{L}_b(l_{n_1})$  should also be incident to either  $v_1$  or  $v_2$ . By Theorem 1, ILIGRA determines that the links in  $\mathcal{L}_b(l_{n_1}) \setminus \mathcal{L}_b(l_{n_2})$ , corresponding to the nodes in  $n \in \mathcal{N}_b(n_1) \setminus \mathcal{N}_b(n_2)$ , are incident to  $v_2$ . For each node  $n$  in  $\mathcal{N}_b(n_1) \setminus \mathcal{N}_b(n_2)$ , ILIGRA sets the first identified incident node  $v_{l_n}$  of the corresponding link  $l_n$  to  $v_2$ , and moves  $n$  from  $\mathcal{N}_w$  to  $\mathcal{N}_h$  (lines 8–9).

---

**Algorithm 1:** ILIGRA( $H$ )

---

**Input:** A line graph  $H$

**Output:** The root graph  $G$  of  $H$  if  $H$  is a line graph

```

1   $G \leftarrow$  an empty graph;
2   $\mathcal{N} \leftarrow$  the set of nodes in  $H$ ;  $\mathcal{N}_w \leftarrow \mathcal{N}$ ;  $\mathcal{N}_h \leftarrow \emptyset$ ;
3   $n_1 \leftarrow$  an arbitrary node  $\in \mathcal{N}_w$ ;
4   $n_2 \leftarrow$  an arbitrary node  $\in \mathcal{N}_b(n_1)$ ;
5  ADDNODE( $G, v_1$ ); ADDNODE( $G, v_2$ );
6  ADDLINK( $G, v_1, v_2$ );  $\mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_1\}$ ;
7   $v_{l_{n_2}} \leftarrow v_1$ ;  $\mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_2\}$ ;  $\mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_2\}$ ;
8  for each  $n \in \mathcal{N}_b(n_1) \setminus \mathcal{N}_b(n_2)$  do
9     $v_{l_n} \leftarrow v_2$ ;  $\mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n\}$ ;  $\mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n\}$ ;
10  $J \leftarrow \mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$ ;
11 if  $1 \leq |J| \leq 2$  then
12   if  $\exists n_u \in J$  such that  $\mathcal{N}_b(n_u) \subseteq \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $|\mathcal{N}_b(n_u) \setminus \{n_1, n_2\}| \geq 3$ 
13     then
14        $v_{l_{n_u}} \leftarrow v_2$ ;  $\mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}$ ;
15        $\mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}$ ;  $J \leftarrow J \setminus \{n_u\}$ ;
16     else
17       INITSPECCASES( $H, n_1, n_2, n_u$ )
18 else if  $|J| \geq 3$  and  $\exists n_u \in J$  such that  $n_u$  is not adjacent to any other node in  $J$ 
19   then
20      $v_{l_{n_u}} \leftarrow v_2$ ;  $\mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}$ ;
21      $\mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}$ ;  $J \leftarrow J \setminus \{n_u\}$ ;
22 for each  $n \in J$  do
23    $v_{l_n} \leftarrow v_1$ ;  $\mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n\}$ ;  $\mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n\}$ ;
24 if  $J \neq \emptyset$  and  $J$  is not a clique in  $H$  then
25    $H$  is not a line graph. Exit.
26 if  $|\mathcal{N}_b(n_1) \setminus J| \neq 0$  and  $\mathcal{N}_b(n_1) \setminus J$  is not a clique then
27    $H$  is not a line graph. Exit.
28 while  $\mathcal{N}_h \neq \emptyset$  do
29    $n \leftarrow$  an arbitrary node in  $\mathcal{N}_h$ ;
30   ADDNODE( $G, v$ ); ADDLINK( $G, v_{l_n}, v$ );
31    $\mathcal{N}_h \leftarrow \mathcal{N}_h \setminus \{n\}$ ;  $\mathcal{C} \leftarrow \emptyset$ ;
32   for each  $n_r \in \mathcal{N}_b(n)$  do
33     if  $n_r \in \mathcal{N}_h$  and  $v_{l_n} \neq v_{l_{n_r}}$  then
34        $\mathcal{C} \leftarrow \mathcal{C} \cup \{n_r\}$ ;
35       ADDLINK( $G, v_{l_{n_r}}, v$ );  $\mathcal{N}_h \leftarrow \mathcal{N}_h \setminus \{n_r\}$ ;
36     else if  $n_r \in \mathcal{N}_w$  then
37        $\mathcal{C} \leftarrow \mathcal{C} \cup \{n_r\}$ ;  $v_{l_{n_r}} \leftarrow v$ ;
38        $\mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_r\}$ ;  $\mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_r\}$ ;
39 if  $\mathcal{C} \neq \emptyset$  and  $\mathcal{C}$  is not a clique in  $H$  then
40    $H$  is not a line graph. Exit.

```

---

ILIGRA sets  $J$  to the intersection of  $\mathcal{N}_b(n_1)$  and  $\mathcal{N}_b(n_2)$  (line 10). If there are only 1 or 2 nodes in  $J$ , and if there exists  $n_u \in J$  such that any neighbor of  $n_u$  is also a neighbor of either  $n_1$  or  $n_2$ , and node  $n_u$  satisfies  $|\mathcal{N}_b(n_u) \setminus \{n_1, n_2\}| \geq 3$ , according to Theorem 3, link  $l_{n_u}$  should be incident to  $v_2$ . ILIGRA sets  $v_{l_{n_u}}$  to  $v_2$ , and adds  $n_u$  to  $\mathcal{N}_h$  and removes  $n_u$  from  $\mathcal{N}_w$  and removes  $n_u$  from  $J$  (lines 11–14). If  $|J| \leq 2$  and  $|\mathcal{N}_b(n_u) \setminus \{n_1, n_2\}| \leq 2$ , the special cases are handled by the subroutine INITSPECcases (lines 15–16). The subroutine INITSPECcases is stated in Appendix A. If  $|J| \geq 3$  and if there exists  $n_u \in J$  such that  $n_u$  is not adjacent to any other node in  $J$ , according to Theorem 2, link  $l_{n_u}$  should be incident to  $v_2$ . ILIGRA sets  $v_{l_{n_u}}$  to  $v_2$ , and adds  $n_u$  to  $\mathcal{N}_h$  and removes  $n_u$  from  $\mathcal{N}_w$  and removes  $n_u$  from  $J$  (lines 17–19).

Since node  $n_u$  has been removed from  $J$ , the rest of links in  $J$  should be incident to  $v_1$ . For each  $n$  in  $J$  which is adjacent with both  $n_1$  and  $n_2$ , ILIGRA sets  $v_{l_n}$  to  $v_1$ , and adds  $n$  to  $\mathcal{N}_h$  and removes  $n$  from  $\mathcal{N}_w$  (lines 20–21). The nodes in  $J$  should be fully connected to each other, and remove the corresponding links are all incident to  $v_1$ . If the nodes in  $J$  do not form a clique in  $H$ , then  $H$  is not a line graph (lines 22–23). The nodes in  $\mathcal{N}_b(n_1) \setminus J$  should also be fully connected to each other, since the corresponding links are all incident to  $v_2$ . If the nodes in  $\mathcal{N}_b(n_1) \setminus J$  do not form a clique in  $H$ , then  $H$  is not a line graph (lines 24–25).

The loop (lines 26–38) runs until  $\mathcal{N}_h$  is an empty set. ILIGRA picks an arbitrary node  $n$  in  $\mathcal{N}_h$  (line 27). ILIGRA adds a node  $v$  and a link  $l_n$  between  $v_{l_n}$  and  $v$  to  $G$  (line 28), and removes  $n$  from  $\mathcal{N}_h$  (line 29). ILIGRA sets  $\mathcal{C}$  to an empty set (line 29). For each neighbor  $n_r$  of  $n$ , if  $n_r \in \mathcal{N}_h$  and  $v_{l_n} \neq v_{l_{n_r}}$ , ILIGRA adds link  $l_{n_r}$  between  $v_{l_{n_r}}$  and  $v$  to  $G$ , and removes  $n_r$  from  $\mathcal{N}_h$ , and adds  $n_r$  to  $\mathcal{C}$  (lines 30–33); If  $n_r \in \mathcal{N}_w$ , ILIGRA sets  $v_{l_{n_r}}$  to  $v$ , and moves  $n_r$  from  $\mathcal{N}_w$  to  $\mathcal{N}_h$ , and adds  $n_r$  to  $\mathcal{C}$  (lines 34–36). The nodes in  $\mathcal{C}$  should be fully connected with each other, since the corresponding links are all incident to  $v$ . If the nodes in  $\mathcal{C}$  do not form a clique in  $H$ ,  $H$  is not a line graph (lines 37–38). If  $H$  is a connected graph,  $\mathcal{N}_w$  should be an empty set when  $\mathcal{N}_h$  becomes an empty set. While  $\mathcal{N}_w \neq \emptyset$ , repeat lines 3–38. For each component of a given disconnected line graph, lines 3–38 will be executed once. If the input graphs are line graphs, lines 22–25 and 37–38 can be skipped, which are used to check whether the given graph is a line graph.

### 2.5 Complexity

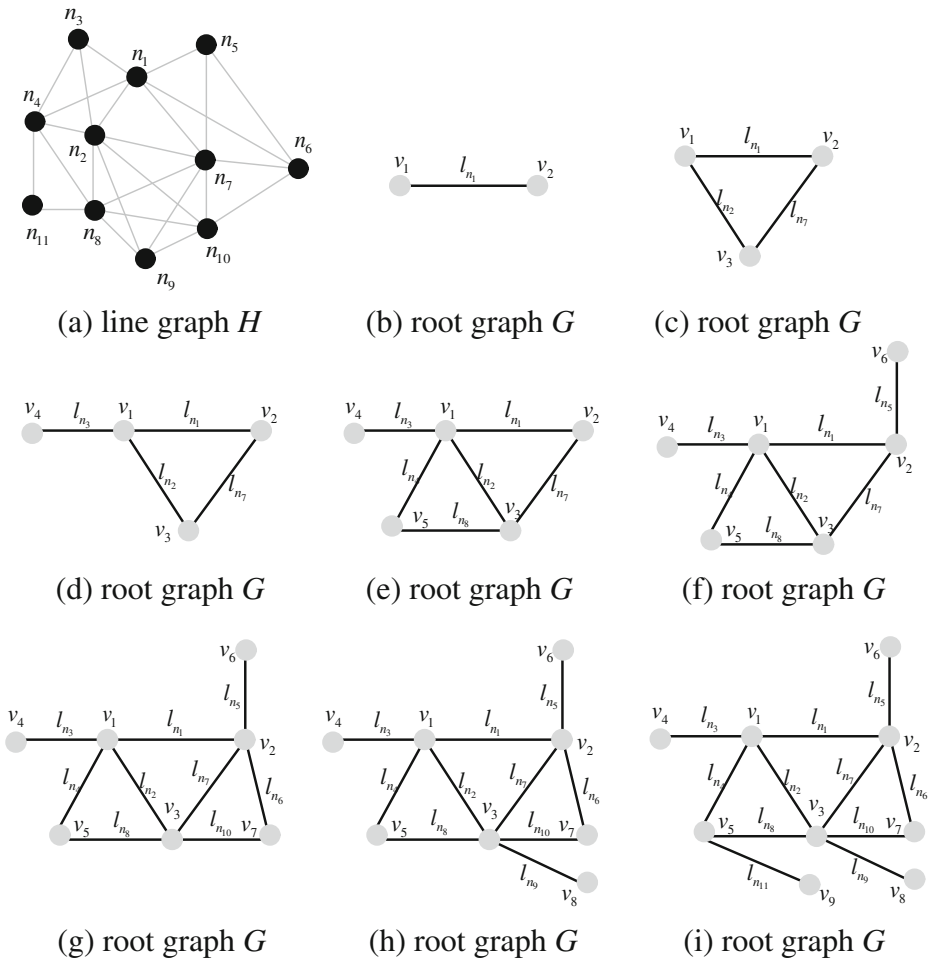
The lines 1–21 of ILIGRA examine all the neighbors of the  $n_1$  in  $H$ , with the complexity  $O(N_H)$ , where  $N_H$  is the number of nodes in  $H$ . The lines 22–25, which check whether  $H$  is a line graph, have the complexity  $O(N_L)$ , where  $N_L$  is the number of links in  $H$ . The lines 26–36 have the complexity  $O(N_H)$ . The lines 37–38 check whether  $H$  is a line graph and have the complexity  $O(N_L)$ . Hence, the overall complexity of ILIGRA with checking if  $H$  is a line graph is  $O(N_L)$ , and the complexity of ILIGRA without checking is  $O(N_H)$ .

### 3 An Example

In this section, we use an example depicted in Fig. 3 to show how ILIGRA works. Given a line graph  $H$  shown in Fig. 3a, ILIGRA constructs its root graph  $G$  incrementally as shown in Fig. 3b–i.

Initially, set  $G$  to an empty graph. We have  $\mathcal{N}_w = \{n_1, n_2, \dots, n_{11}\}$  and  $\mathcal{N}_h = \emptyset$ . Add nodes  $v_1$  and  $v_2$  to  $G$ , and add link  $l_{n_1}$  between  $v_1$  and  $v_2$  to  $G$  as shown in Fig. 3b, and  $\mathcal{N}_w = \{n_2, n_3, \dots, n_{11}\}$ . Set  $v_{l_{n_2}}$  to  $v_1$ ,  $\mathcal{N}_w = \{n_3, n_4, \dots, n_{11}\}$  and  $\mathcal{N}_h = \{n_2\}$ . Since



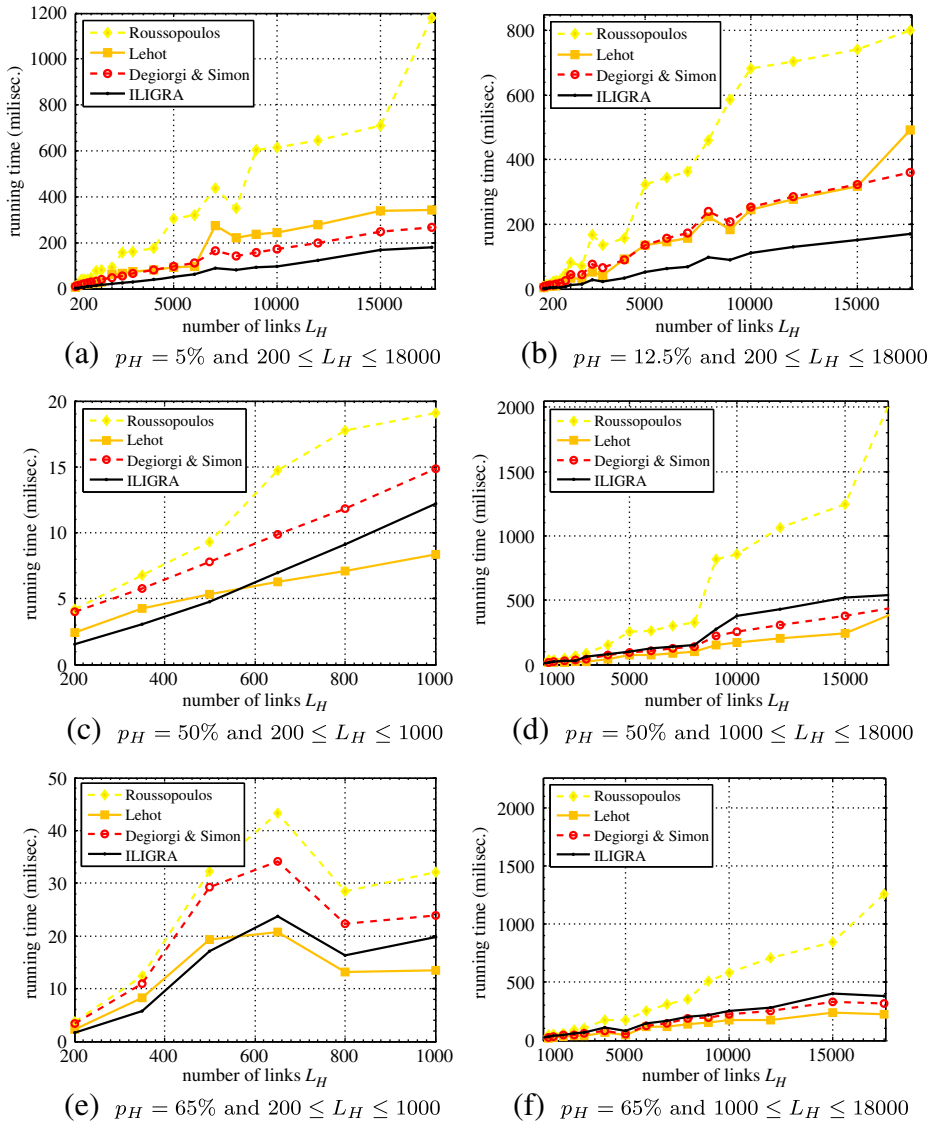


**Fig. 3** An example shows how ILIGRA constructs  $G$  from a given  $H$

$\mathcal{N}_b(n_1) \setminus \mathcal{N}_b(n_2) = \{n_5, n_6\}$ , according to Theorem 1, set  $v_{l_{n_5}}$  to  $v_2$  and also set  $v_{l_{n_6}}$  to  $v_2$ . We have  $\mathcal{N}_w = \{n_3, n_4, n_7, n_8, \dots, n_{11}\}$  and  $\mathcal{N}_h = \{n_2, n_5, n_6\}$ . Since  $\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2) = \{n_3, n_4, n_7\}$  and none of  $n_3$  and  $n_4$  is not adjacent to  $n_7$ , according to Theorem 1, set  $v_{l_{n_7}}$  to  $v_2$ . Now  $\mathcal{N}_w = \{n_3, n_4, n_8, n_9, n_{10}, n_{11}\}$  and  $\mathcal{N}_h = \{n_2, n_5, n_6, n_7\}$ . For the two nodes  $n_3$  and  $n_4$  in  $\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2) \setminus \{n_7\}$ , the corresponding links should be incident to  $v_1$ . Hence, set both  $v_{l_{n_3}}$  and  $v_{l_{n_4}}$  to  $v_1$ . Now  $\mathcal{N}_w = \{n_8, n_9, n_{10}, n_{11}\}$  and  $\mathcal{N}_h = \{n_2, n_3, n_4, n_5, n_6, n_7\}$ .

Take  $n_2$  from  $\mathcal{N}_h$ . Add a node  $v_3$  to  $G$  and add link  $l_{n_2}$  between  $v_3$  and  $v_{l_{n_2}}$  ( $v_{l_{n_2}}$  has been found to be  $v_1$  previously), as shown in Fig. 3c. Now  $\mathcal{N}_h = \{n_3, n_4, n_5, n_6, n_7\}$ . We have  $\mathcal{N}_b(n_2) = \{n_1, n_3, n_4, n_7, n_8, n_9, n_{10}\}$ . Since  $n_7 \in \mathcal{N}_h$  and  $v_{l_{n_7}} = v_2 \neq v_{l_{n_2}} = v_1$ , add  $l_{n_7}$  between  $v_2$  and  $v_3$  to  $G$ . Now  $\mathcal{N}_h = \{n_3, n_4, n_5, n_6\}$ . Since  $n_8, n_9$  and  $n_{10}$  belong to  $\mathcal{N}_w$ , set  $v_{l_{n_8}}, v_{l_{n_9}}$  and  $v_{l_{n_{10}}}$  to  $v_3$ . Now  $\mathcal{N}_w = \{n_{11}\}$  and  $\mathcal{N}_h = \{n_3, n_4, n_5, n_6, n_8, n_9, n_{10}\}$ .

Take  $n_3$  from  $\mathcal{N}_h$ . Add a node  $v_4$  to  $G$  and add link  $l_{n_3}$  between  $v_4$  and  $v_{l_{n_3}}$ , which is namely  $v_1$ , as shown in Fig. 3d. Now  $\mathcal{N}_h = \{n_4, n_5, n_6, n_8, n_9, n_{10}\}$ .



**Fig. 4** Algorithms' running times for *line graphs* with different  $p_H$  and  $L_H$

Take  $n_4$  from  $\mathcal{N}_h$ . Add a node  $v_5$  to  $G$  and add link  $l_{n_4}$  between  $v_5$  and  $v_{l_{n_4}}$ , which is also  $v_1$ , as shown in Fig. 3e. Now  $\mathcal{N}_h = \{n_5, n_6, n_8, n_9, n_{10}\}$ . We have  $\mathcal{N}_b(n_4) = \{n_1, n_2, n_3, n_8, n_{11}\}$ . Since  $n_8 \in \mathcal{N}_h$  and  $v_{l_{n_8}} = v_3 \neq v_{l_{n_4}} = v_1$ , add  $l_{n_8}$  between  $v_5$  and  $v_3$  to  $G$ . Now  $\mathcal{N}_h = \{n_5, n_6, n_9, n_{10}\}$ . Since  $n_{11} \in \mathcal{N}_w$ , set  $v_{l_{n_{11}}}$  to  $v_5$ . Now  $\mathcal{N}_w = \emptyset$  and  $\mathcal{N}_h = \{n_5, n_6, n_9, n_{10}, n_{11}\}$ .

Take  $n_5$  from  $\mathcal{N}_h$ . Add a node  $v_6$  to  $G$  and add link  $l_{n_5}$  between  $v_6$  and  $v_{l_{n_5}}$ , which is also  $v_2$ , as shown in Fig. 3f. Now  $\mathcal{N}_h = \{n_6, n_9, n_{10}, n_{11}\}$ .

**Table 2** The fastest algorithm for different input line graphs

Link density $p_H$	Number of links $L_H$	
	$\leq 500$	$> 500$
0.05	ILIGRA	ILIGRA
0.125	ILIGRA	ILIGRA
0.5	ILIGRA	Lehot
0.65	ILIGRA	Lehot

Take  $n_6$  from  $\mathcal{N}_h$ . Add a node  $v_7$  to  $G$  and add link  $l_{n_6}$  between  $v_7$  and  $v_{l_{n_6}}$ , which is also  $v_2$ , as shown in Fig. 3g. Now  $\mathcal{N}_h = \{n_9, n_{10}, n_{11}\}$ . We have  $\mathcal{N}_b(n_6) = \{n_1, n_5, n_7, n_{10}\}$ . Since  $n_{10} \in \mathcal{N}_h$  and  $v_{l_{n_{10}}} = v_3 \neq v_{l_{n_6}} = v_2$ , add  $l_{n_{10}}$  between  $v_7$  and  $v_3$  to  $G$ . Now  $\mathcal{N}_h = \{n_9, n_{11}\}$ .

Take  $n_9$  from  $\mathcal{N}_h$ . Add a node  $v_8$  to  $G$  and add link  $l_{n_9}$  between  $v_8$  and  $v_{l_{n_9}}$ , which is also  $v_3$ , as shown in Fig. 3h. Now  $\mathcal{N}_h = \{n_{11}\}$ .

Take the only node  $n_9$  from  $\mathcal{N}_h$ . Add a node  $v_9$  to  $G$  and add link  $l_{n_{11}}$  between  $v_9$  and  $v_{l_{n_{11}}}$ , which is also  $v_5$ , as shown in Fig. 3i. Now  $\mathcal{N}_h = \emptyset$ . Since  $\mathcal{N}_w$  is also an empty set, the construction of  $G$  is accomplished.

### 4 Evaluation

We compare ILIGRA’s running time with the running times of three published line graph reconstruction algorithms: Roussopoulos’ algorithm [20], Lehot’s algorithm [12], and Degiorgi and Simon’s algorithm [6]. All algorithms have been implemented in the same programming language (C++) and the same data structures and libraries [15] have been used.<sup>5</sup> The evaluation of all the algorithms has been conducted on the same machine.<sup>6</sup>

The performances of the above-mentioned algorithms have been compared using the same input graphs  $H$ . All the algorithms are able to construct the root graph  $G$  if the given graph  $H$  is a line graph, and can tell *non-line graph* when  $H$  is not a line graph. The line graphs with link density<sup>7</sup>  $p_H = 0.05$  and  $0.125$  are generated by converting random graphs [3, 7] with a fixed link density into line graphs. However, the line graphs of these random graphs can never have high link densities (explained with details in Appendix B). Therefore, the line graphs with  $p_H = 0.5$  and  $0.65$  have been generated by converting the scale-free graphs [2] into line graphs.

<sup>5</sup>LEDA: <http://www.algorithmic-solutions.com/leda/>

<sup>6</sup>Intel(R) Core(TM) 2 Duo CPU T9600 on  $2 \times 2.80$  GHz; 4 GB RAM memory

<sup>7</sup>The *link density* of a given line graph  $H(L, L_H)$  is defined by  $p_H = L_H / \binom{L}{2}$ , where  $L$  is the number of nodes in  $H$  and  $L_H$  is the number of links in  $H$ .

Figure 4 reflects the trends for the running times of all the algorithms when the input graphs are line graphs with different link density  $p_H$  and different number of links  $L_H$ . Figure 4a and b show the running times for line graphs with small link density  $p_H = 0.05, 0.125$ , where ILIGRA performs faster than all the other algorithms. Figure 4c, d, e and f illustrate the trends for the algorithms' running times for line graphs with high link density  $p_H = 0.50$  and  $p_H = 0.65$ . ILIGRA is the fastest algorithm for line graphs with small number of links  $200 \leq L_H \leq 500$  and Lehot's algorithm has the shortest running time for line graphs with high number of links  $650 \leq L_H \leq 18000$ . The best algorithms, reflected by the running time for all the cases, are summarized in Table 2.

## 5 Conclusion

We present ILIGRA algorithm for inverse line graph construction. Given a line graph  $H$ , ILIGRA constructs its root graph  $G$  and checks whether the given graph is a line graph during the construction. ILIGRA also works for disconnected line graphs by iterating through the connected components of the input line graph. The time complexity of ILIGRA is linear in the number of nodes in the input graph  $H$  without checking if the given graph is a line graph. The time complexity of ILIGRA with full functionality is linear in the number of links in the given line graphs. Numerical comparisons with the algorithms of Lehot, Roussopoulos, and Degiorgi and Simon have been demonstrated. Given line graphs with small link density (i.e. sparse graphs), ILIGRA is the fastest algorithm in root graph construction, as shown in Table 2.

**Acknowledgments** We would like to thank Prof. Dr. Klaus Simon from ETH Zürich for providing us with the initial implementations of the algorithms of Lehot, Roussopoulos, and Degiorgi and Simon. We are grateful to two anonymous reviewers who provided us with valuable comments.

## Appendix A: Special cases

We start with two adjacent nodes  $n_1$  and  $n_2$  in  $H$  that correspond to links  $l_{n_1}$  and  $l_{n_2}$  in  $G$ , respectively, where  $l_{n_1}$  is incident to  $v_1$  and  $v_2$  and where  $l_{n_2}$  is incident to  $v_1$  and  $v_3$ . We denote by  $J = \mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2)$ ,  $C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $L$  is the number of links in  $G$ . For each  $n_u \in J$ , link  $l_{n_u}$  is either incident to  $v_1$ , or incident to both  $v_2$  and  $v_3$ . For each  $n_u \in J$ , we denote  $Z = \mathcal{N}_b(n_u) \setminus \{n_1, n_2\}$ . In the remainder of this appendix and Table 3, we continue with the case analysis for the remaining cases:  $|J| \leq 2$  and  $|Z| \leq 2$ .

1.  $Z = \emptyset$

(a)  $|J| = 1$

i.  $L = 3$

The root graph  $G$  is  $K_3$  or  $K_{1,3}$ , as shown in Fig. 5a.

ii.  $L \geq 4$

$l_{n_u}$  is incident to  $v_1$ .

*Proof* Let us assume  $l_{n_u}$  is incident to  $v_2$  and  $v_3$ . Since  $H$  has more than 3 nodes and  $\mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2) = \{n_u\}$ , the extra links must be incident to

**Table 3** All cases for identifying the special node in  $J$ , whose corresponding link is incident to  $v_2$

Conditions		$l_{n_u}$ is incident to
$ J  \geq 3$		Use Theorem 2.
$ Z  \geq 3$		Use Theorem 3.
$ Z  = 0$	$ J  = 1$ $L = 3$	$v_1$ or $v_2$ ( $G$ is $K_3$ or $K_1;3$ )
	$ J  = 1$ $L \geq 4$	$v_1$
	$ J  = 2$ and $nr \notin Nb(nu)$ $L = 4$	$v_1$ or $v_2$ (isomorphic)
$ J  \leq 2$	$ J  = 1$ $L \geq 5$	$v_2$ if $nx \notin C$ ; $v_1$ if $nx \in C$
	$ J  = 1$ $L = 4$	$v_1$ or $v_2$ (isomorphic)
	$ J  = 1$ $L \geq 5$	$v_2$ if $nx \notin C$ ; $v_1$ if $nx \in C$
	$ J  = 2$ and $nr \notin Nb(nu)$ $ns \notin Nb(nr)$	$v_2$ if $ns \in C$ ; $v_1$ if $ns \notin C$
	$ J  = 2$ and $nr \notin Nb(nu)$ $ns \in Nb(nr)$ $L = 5$	$v_1$ or $v_2$ (isomorphic)
	$ J  = 2$ and $nr \notin Nb(nu)$ $ns \in Nb(nr)$ $L \geq 6$	$v_2$ if $nx \notin C$ ; $v_1$ if $nx \in C$
$ Z  = 2$	$ns \notin Nb(nt)$ $ J  = 1$	Figure 6 (a)
	$ns \notin Nb(nt)$ $ J  = 2$ and $nr \notin Nb(nu)$	$v_2$
	$ns \in Nb(nt), ns, nt, n_1$ pairwise adjacent	$v_2$
	$ns \in Nb(nt),  J  = 1$ $L = 5$	$v_1$ or $v_2$ (isomorphic)
$ns; nt; n_1$ are not pairwise adjacent	$ J  = 1$ $L \geq 6$	$v_2$ if $nx \notin C$ ; $v_1$ if $nx \in C$
	$ J  = 1$ and $L = 6$	$v_1$ or $v_2$ (isomorphic)
	$nr \notin Nb(nu)$ $L \geq 7$	$v_2$ if $nx \notin C$ ; $v_1$ if $nx \in C$

Notation:  $J = \mathcal{N}_b(n_1) \cap \mathcal{N}_b(n_2), n_u \in J, C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2), Z = \mathcal{N}_b(n_u) \setminus \{n_1, n_2\}$

$v_2$  or  $v_3$  in  $G$ , which means  $n_u$  must have other neighbors different from  $n_1$  and  $n_2$ , contradicting the fact that  $Z = \mathcal{N}_b(n_u) \setminus \{n_1, n_2\} = \emptyset$ . Hence,  $l_{n_u}$  can only be incident to  $v_1$ . □

(b)  $|J| = 2$  ( $J = \{n_u, n_r\}$ ) and  $n_r \notin \mathcal{N}_b(n_u)$

i.  $L = 4$

$l_{n_u}$  is incident to  $v_1$  or  $v_2$ , as shown in Fig. 5b. The resulting root graphs are isomorphic.

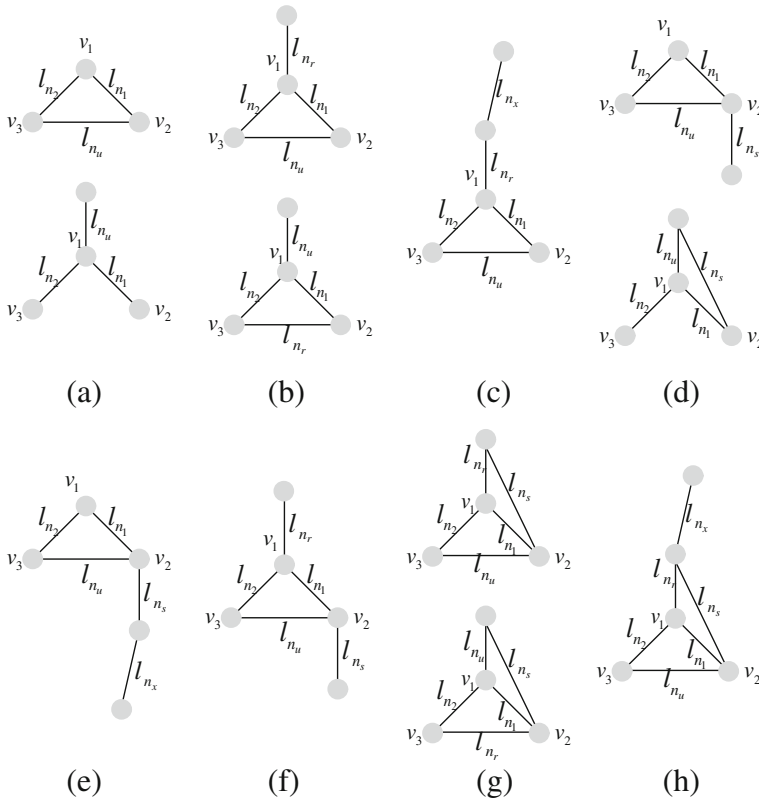
ii.  $L \geq 5$  and  $n_x$  is the node in  $H$  different from  $n_1, n_2, n_u,$  and  $n_r$ .

If  $n_x \notin C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$ , then  $l_{n_u}$  is incident to  $v_2$  (Fig. 5c), otherwise  $l_{n_u}$  is incident to  $v_1$ .

*Proof* Let us first assume  $n_x \notin C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $l_{n_u}$  is incident to  $v_1$ . Since  $|Z| = 0, l_{n_x}$  must be incident to either  $v_2$  or  $v_3$ , which contradicts the fact that  $n_x \notin C$ . Hence,  $l_{n_u}$  is incident to  $v_2$ . Let us now assume  $n_x \in C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $l_{n_u}$  is incident to  $v_2$ . Since  $|Z| = 0, l_{n_x}$  must be incident to  $v_1$ , which contradicts the fact that  $|J| = 2$ . Hence,  $l_{n_u}$  is incident to  $v_1$ . □

2.  $|Z| = 1$  ( $Z = \{n_s\}$ )

(a)  $|J| = 1$



**Fig. 5** Scenarios for (a)  $|Z| = 0, |J| = 1, L = 3$ ; (b)  $|Z| = 0, |J| = 2, L = 4$ ; (c)  $|Z| = 0, |J| = 2, L = 5, n_x \notin C$ ; (d)  $|Z| = 1, |J| = 1, L = 4$ ; (e)  $|Z| = 1, |J| = 1, L = 5, n_x \notin C$ ; (f)  $|Z| = 1, |J| = 2, L = 5, n_s \notin \mathcal{N}_b(n_r), n_s \in C$ ; (g)  $|Z| = 1, |J| = 2, L = 5, n_s \in \mathcal{N}_b(n_r)$ ; and (h)  $|Z| = 1, |J| = 2, L = 6, n_s \in \mathcal{N}_b(n_r), n_x \notin C$

- i.  $L = 4$   
 $l_{n_u}$  is incident to  $v_1$  or  $v_2$ , as shown in Fig. 5d. The resulting root graphs are isomorphic.
- ii.  $L \geq 5$  and  $n_x$  is the node in  $H$  different from  $n_1, n_2, n_u$ , and  $n_s$ .  
 If  $n_x \notin C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$ ,  $l_{n_u}$  is incident to  $v_2$  (Fig. 5e), otherwise  $l_{n_u}$  is incident to  $v_1$ .

*Proof* Let us first assume  $n_x \notin C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $l_{n_u}$  is incident to  $v_1$ . Since  $|J| = 1$  and  $|Z| = 1$ ,  $l_{n_x}$  must be incident to either  $v_2$  or  $v_3$ , which contradicts the fact that  $n_x \notin C$ . Hence,  $l_{n_u}$  is incident to  $v_2$ . Let us now assume  $n_x \in C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $l_{n_u}$  is incident to  $v_2$  and  $v_3$ . Since  $|Z| = 1$ ,  $l_{n_x}$  must be incident to  $v_1$ , which contradicts the fact that  $|J| = 1$ . Hence,  $l_{n_u}$  is incident to  $v_1$ . □

(b)  $|J| = 2$  ( $J = \{n_u, n_r\}$ ) and  $n_r \notin \mathcal{N}_b(n_u)$

i.  $n_s \notin \mathcal{N}_b(n_r)$

If  $n_s \in C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$ ,  $l_{n_u}$  is incident to  $v_2$  (Fig. 5f), otherwise  $l_{n_u}$  is incident to  $v_1$ .

*Proof* Let us first assume  $n_s \in C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $l_{n_u}$  is incident to  $v_1$ . Since  $l_{n_u}$  is incident to  $v_1$ ,  $l_{n_r}$  must be incident to  $v_2$  and  $v_3$ . Since  $n_s \notin \mathcal{N}_b(n_r)$ ,  $l_{n_s}$  must be incident to  $v_1$ , contradicting the fact that  $|J| \leq 2$ . Hence,  $l_{n_u}$  is incident to  $v_2$ . Let us now assume  $n_s \notin C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $l_{n_u}$  is incident to  $v_2$  and  $v_3$ . Since  $n_s \in \mathcal{N}_b(n_u)$ ,  $n_s$  is incident to either  $v_2$  or  $v_3$ , contradicting the fact that  $n_s \notin C$ . Hence,  $l_{n_u}$  is incident to  $v_1$ .  $\square$

ii.  $n_s \in \mathcal{N}_b(n_r)$

A.  $L = 5$

$l_{n_u}$  is incident to  $v_1$  or  $v_2$ , as shown in Fig. 5g. The resulting root graphs are isomorphic.

B.  $L \geq 6$  and  $n_x$  is the node in  $H$  different from  $n_1, n_2, n_u, n_r$ , and  $n_s$ .

If  $n_x \notin C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$ ,  $l_{n_u}$  is incident to  $v_2$  (Fig. 5h), otherwise  $l_{n_u}$  is incident to  $v_1$ .

*Proof* Assume that  $n_x \notin C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $l_{n_u}$  is incident to  $v_1$ . Since  $n_r \notin \mathcal{N}_b(n_u)$ ,  $l_{n_r}$  is incident to  $v_2$  and  $v_3$ . Since  $n_s \in \mathcal{N}_b(n_r)$  and  $n_s \in \mathcal{N}_b(n_u)$ ,  $l_{n_s}$  is incident to the node of  $l_{n_u}$  different from  $v_1$  and either  $v_2$  or  $v_3$ . Since  $|J| = 2$  and  $|Z| = 1$ ,  $l_{n_x}$  must be incident to either  $v_2$  or  $v_3$ , which contradicts the fact that  $n_x \notin C$ . Hence,  $l_{n_u}$  is incident to  $v_2$ . Now, assume that  $n_x \in C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $l_{n_u}$  is incident to  $v_2$  and  $v_3$  and  $l_{n_r}$  is incident to  $v_1$ . Since  $n_s \in \mathcal{N}_b(n_u)$ ,  $l_{n_s}$  is incident to either  $v_2$  or  $v_3$ . Since  $|Z| = 1$ ,  $l_{n_x}$  must be incident to  $v_1$ , which contradicts the fact that  $|J| = 2$ . Hence,  $l_{n_u}$  is incident to  $v_1$ .  $\square$

3.  $|Z| = 2$  ( $Z = \{n_s, n_t\}$ )

(a)  $n_s \notin \mathcal{N}_b(n_t)$

i.  $|J| = 1$  ( $J = \{n_u\}$ )

$l_{n_u}$  is incident to  $v_2$ .

*Proof* Assume that  $l_{n_u}$  is incident to  $v_1$  and  $v_u$ . Since  $|J| = 1$ , both  $l_{n_s}$  and  $l_{n_t}$  must be incident to  $v_u$ , then  $n_s \in \mathcal{N}_b(n_t)$ , contradicting the assumption that  $n_s \notin \mathcal{N}_b(n_t)$ . Hence,  $l_{n_u}$  is incident to  $v_2$ .  $\square$

- ii.  $|J| = 2$  ( $J = \{n_u, n_r\}$ ) and  $n_r \notin \mathcal{N}_b(n_u)$   
 $l_{n_u}$  is incident to  $v_2$ , as shown in Fig. 6a.

*Proof* Assume that  $l_{n_u}$  is incident to  $v_1$  and  $v_u$ . Since  $l_{n_u}$  is incident to  $v_1$ ,  $l_{n_r}$  must be incident to  $v_2$ . Since  $|J| = 2$ , both  $l_{n_s}$  and  $l_{n_t}$  must be incident to  $v_u$ , then  $n_s \in \mathcal{N}_b(n_t)$ , contradicting with the assumption that  $n_s \notin \mathcal{N}_b(n_t)$ . Hence,  $l_{n_u}$  is incident to  $v_2$ . □

- (b)  $n_s \in \mathcal{N}_b(n_t)$  and  $n_s, n_t, n_1$  (or  $n_2$ ) are pairwise adjacent  
 $l_{n_u}$  is incident to  $v_2$ , as shown in Fig. 6b.

*Proof* If  $l_{n_u}$  is incident to  $v_1$ , neither  $n_s, n_t, n_1$  nor  $n_s, n_t, n_2$  can be pairwise adjacent, hence  $l_{n_u}$  is incident to  $v_2$ . □

- (c)  $n_s \in \mathcal{N}_b(n_t)$  and  $n_s, n_t, n_1$  are not pairwise adjacent

- i.  $|J| = 1$

- A.  $L = 5$

$l_{n_u}$  is incident to  $v_1$  or  $v_2$ , as shown in Fig. 6c. The resulting root graphs are isomorphic.

- B.  $L \geq 6$  and  $n_x$  is the node in  $H$  different from  $n_1, n_2, n_u, n_s$ , and  $n_t$ .

If  $n_x \notin C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$ ,  $l_{n_u}$  is incident to  $v_2$  (Fig. 6d), otherwise  $l_{n_u}$  is incident to  $v_1$ .

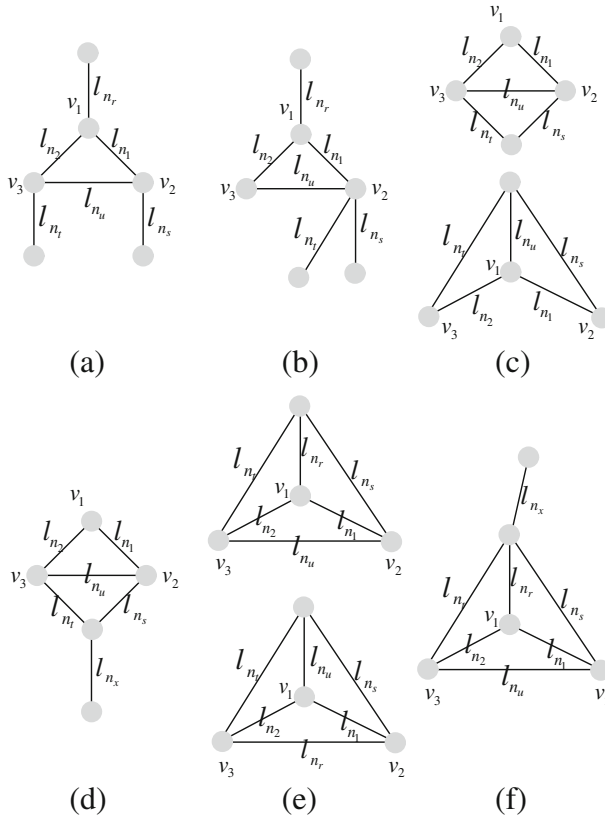
*Proof* Assume  $n_x \notin C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $l_{n_u}$  is incident to  $v_1$ . Since  $|J| = 1$  and  $|Z| = 2$ , there is no link incident to both  $v_2$  and  $v_3$ , hence  $l_{n_s}$  and  $l_{n_t}$  are incident to the node of  $l_{n_u}$  different from  $v_1$  and one of the nodes  $v_2$  and  $v_3$ . Now,  $l_{n_x}$  cannot be adjacent to  $v_1$  as  $|J| = 1$ ; it cannot be adjacent to none of  $v_2, v_3$  and the node of  $l_{n_u}$  different from  $v_1$ , because  $|Z| = 1$ , hence there will not be a space for link  $l_{n_x}$ , which contradicts the assumption of the existence of  $n_x$  and  $L \geq 6$ . Hence,  $l_{n_u}$  is incident to  $v_2$ . Now, assume  $n_x \in C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $l_{n_u}$  is incident to  $v_2$  and  $v_3$ . Since  $|J| = 1$ , no other link is incident to  $v_1$ , therefore  $l_{n_s}, l_{n_t}$  and  $l_{n_x}$  are all incident to either  $v_2$  or  $v_3$ , which contradicts the assumption  $|Z| = 2$ . Hence,  $l_{n_u}$  is incident to  $v_1$ . □

- ii.  $|J| = 2$  ( $J = \{n_u, n_r\}$ ) and  $n_r \notin \mathcal{N}_b(n_u)$

- A.  $L = 6$

$l_{n_u}$  is incident to  $v_1$  or  $v_2$ , as shown in Fig. 6e. The resulting root graphs are isomorphic.





**Fig. 6** Scenarios for  $|Z| = 2$ : (a)  $n_s \notin \mathcal{N}_b(n_t)$ ,  $|J| = 2$ ; (b)  $n_s \in \mathcal{N}_b(n_t)$  and  $n_s, n_t, n_1$  (or  $n_2$ ) are pairwise adjacent;  $n_s \in \mathcal{N}_b(n_t)$  and  $n_s, n_t, n_1$  (or  $n_2$ ) are not pairwise adjacent: (c)  $|J| = 1$ ,  $L = 5$ ; (d)  $|J| = 1$ ,  $L = 6$ ,  $n_x \notin C$ ; (e)  $|J| = 2$ ,  $L = 6$ ; and (f)  $|J| = 2$ ,  $L = 7$ ,  $n_x \notin C$

- B.  $L \geq 7$  and  $n_x$  is the node in  $H$  different from  $n_1, n_2, n_u, n_r, n_s$ , and  $n_t$ .  
 If  $n_x \notin C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$ ,  $l_{n_u}$  is incident to  $v_2$  (Fig. 6f), otherwise  $l_{n_u}$  is incident to  $v_1$ .

*Proof* Assume  $n_x \notin C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $l_{n_u}$  is incident to  $v_1$ . Since  $|J| = 2$  and  $|Z| = 2$ ,  $l_{n_r}$  is incident to both  $v_2$  and  $v_3$ ;  $l_{n_s}$  and  $l_{n_t}$  are incident to the node of  $l_{n_u}$  different from  $v_1$  and one of the nodes  $v_2$  and  $v_3$ . Since  $n_x \notin C$ , link  $l_{n_x}$  is incident to the node of  $l_{n_u}$  different from  $v_1$ , which contradicts the assumption of  $|Z| = 2$ . Hence,  $l_{n_u}$  is incident to  $v_2$ . Now, assume  $n_x \in C = \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  and  $l_{n_u}$  is incident to  $v_2$  and  $v_3$ . Since  $|J| = 2$ ,  $l_{n_r}$  is incident to  $v_1$ . Since  $n_x \in C$  and  $|J| = 1$ , link  $l_{n_x}$  cannot be incident to  $v_1$ . Finally,  $l_{n_s}$ ,  $l_{n_r}$  and  $l_{n_x}$  are all incident to either  $v_2$  or  $v_3$ ,

which contradicts the fact that  $|Z| = 2$ . Hence,  $l_{n_u}$  is incident to  $v_1$ .  $\square$

## Appendix B: The Link Density of Line Graphs

The link density is an important characteristic for the topology of line graphs. This section discusses the relation between the link density of line graph  $H$ , the number of nodes  $N$  and the number of links  $L$  in the root graph  $G(N, L)$ .

The number of nodes  $N_H$  in the line graph  $H$  is equal to the number of links  $L$  in the root graph  $G$ . For the number of links  $L_H$  in the line graph  $H$ , we have

$$L_H = \frac{1}{2} \sum_{i=1}^N d_i^2 - L \quad (1)$$

where  $d = [d_1, d_2, \dots, d_N]$  is the degree sequence of  $G$ .

By using (1), the link density  $p_H$  of  $H$  equals

$$p_H = \frac{L_H}{\binom{L}{2}} = \frac{\frac{1}{2} \sum_{i=1}^N d_i^2 - L}{\binom{L}{2}} = \frac{\sum_{i=1}^N d_i^2 - 2L}{L^2 - L} \quad (2)$$

Using the basic law of degrees,  $\sum_{i=1}^N d_i = 2L$ , and Cauchy's inequality [4, 9]

$$\sum_{i=1}^N d_i^2 \geq \frac{\left(\sum_{i=1}^N d_i\right)^2}{N} = \frac{4L^2}{N} \quad (3)$$

and using (3) into (2), we obtain

$$p_H = \frac{\sum_{i=1}^N d_i^2 - 2L}{L^2 - L} \geq \frac{2L \left(\frac{2L}{N} - 1\right)}{L^2 - L} = \frac{2}{N} \frac{2L - N}{L - 1} \quad (4)$$

Equality in (4) holds for regular root graphs  $G$ , where  $d_i = \frac{2L}{N}$ , for  $i = 1, 2, \dots, N$ . When  $L \gg N$ , the link density  $p_H$  asymptotically tends to  $\frac{4}{N}$ . Hence, the line graphs of dense root graphs with  $L \gg N$  have small link densities.

**Algorithm 2:** INITSPEC CASES( $H, n_1, n_2, n_u$ )

```

1  if  $\mathcal{N}_b(n_u) \setminus \{n_1, n_2\} = \emptyset$  then
2    if  $J = \{n_u\}$  then
3      if  $L = 3$  then
4        |  $G$  is  $K_{1,3}$  or  $K_3$ . Exit.
5      else if  $L \geq 4$  then
6        |  $v_{l_{n_u}} \leftarrow v_1; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
7    else if  $J = \{n_u, n_r\}$  and  $n_r \notin \mathcal{N}_b(n_u)$  then
8      if  $L = 4$  then
9        |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
10       else if  $L \geq 5$  and  $n_x \notin \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  then
11         |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
12 else if  $\mathcal{N}_b(n_u) \setminus \{n_1, n_2\} = \{n_s\}$  then
13   if  $J = \{n_u\}$  then
14     if  $L = 4$  then
15       |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
16     else if  $L \geq 5$  and  $n_x \notin \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  then
17       |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
18   else if  $J = \{n_u, n_r\}$  and  $n_r \notin \mathcal{N}_b(n_u)$  then
19     if  $n_s \notin \mathcal{N}_b(n_r)$  and  $n_s \in \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  then
20       |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
21     else if  $n_s \in \mathcal{N}_b(n_r)$  then
22       if  $L = 5$  then
23         |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
24       else if  $L \geq 6$  and  $n_x \notin \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  then
25         |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
26 else if  $\mathcal{N}_b(n_u) \setminus \{n_1, n_2\} = \{n_s, n_t\}$  then
27   if  $n_s \notin \mathcal{N}_b(n_t)$  then
28     if  $J = \{n_u\}$  then
29       |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
30     else if  $J = \{n_u, n_r\}$  and  $n_r \notin \mathcal{N}_b(n_u)$  then
31       |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
32   else if  $n_s \in \mathcal{N}_b(n_t)$  and  $n_s, n_t, n_1$  or  $n_2$  form a  $K_3$  then
33     |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
34   else if  $n_s \in \mathcal{N}_b(n_t)$  and neither  $n_s, n_t, n_1$  nor  $n_s, n_t, n_2$  form a  $K_3$  then
35     if  $J = \{n_u\}$  then
36       if  $L = 5$  then
37         |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
38       else if  $L \geq 6$  and  $n_x \notin \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  then
39         |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
40     else if  $J = \{n_u, n_r\}$  and  $n_r \notin \mathcal{N}_b(n_u)$  then
41       if  $L = 6$  then
42         |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 
43       else if  $L \geq 7$  and  $n_x \notin \mathcal{N}_b(n_1) \cup \mathcal{N}_b(n_2)$  then
44         |  $v_{l_{n_u}} \leftarrow v_2; \mathcal{N}_h \leftarrow \mathcal{N}_h \cup \{n_u\}; \mathcal{N}_w \leftarrow \mathcal{N}_w \setminus \{n_u\}; J \leftarrow J \setminus \{n_u\};$ 

```

We derive an upper bound for the link density  $p_H$ . Using  $L = \left(\sum_{i=1}^N d_i\right)/2$  and the inequality  $\left(\sum_{i=1}^N x_i\right)^2 \geq \sum_{i=1}^N x_i^2$  for  $x_i = d_i - 1 \geq 0$ , we obtain

$$\begin{aligned} \sum_{i=1}^N d_i^2 &= \sum_{i=1}^N (d_i - 1)^2 - N + 2 \sum_{i=1}^N d_i = \sum_{i=1}^N (d_i - 1)^2 - N + 4L \\ &\leq 4L - N + \left(\sum_{i=1}^N (d_i - 1)\right)^2 = (2L - N + 1)^2 + N - 1 \end{aligned} \quad (5)$$

Finally,  $p_H$  is bounded by

$$\frac{4L - 2N}{N(L - 1)} \leq p_H \leq \frac{(2L - N + 1)^2 + N - 2L - 1}{L^2 - L}$$

Equality in (5) is achieved if and only if  $(d_i - 1)(d_j - 1) = 0$  for all  $i, j \in 1, 2, \dots, N$ . The star graph  $K_{1,N}$  satisfies the condition for equality in (5), indicating that the line graph of  $K_{1,N}$  reaches the upper bound of link density  $p_H$ . In fact, the line graph of  $K_{1,N}$  is complete graph  $K_{N-1}$  with maximum link density of 1. In conclusion, dense line graphs can be obtained if the original graph has one node with a high degree and the other nodes have relatively small degrees. On the other hand, the line graph of a regular graph has the minimum link density. Hence, the line graphs with  $p_H = 0.5$  and  $0.65$  in Section 4 are generated by converting the scale-free graphs into line graphs.

## References

1. Ahn, Y.Y., Bagrow, J.P., Lehmann, S.: Link communities reveal multiscale complexity in networks. *Nature* **466**(7307), 761–764 (2010)
2. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. *Science* **286**(5439), 509–512 (1999)
3. Bollobás, B.: *Random Graphs*. Cambridge University Press, Cambridge (2001)
4. Cauchy, A.L.: *Cours d'analyse de l'Ecole Royale Polytechnique*, vol. 3 (1821). Imprimerie royale, Paris (reissued by Cambridge University Press), Cambridge (2009)
5. Cvetković, D., Rowlinson, P., Simić, S.: *Spectral Generalizations of Line Graphs*. Cambridge University Press, Cambridge (2004)
6. Degiorgi, D.G., Simon, K.: A dynamic algorithm for line graph recognition. In: *Proceedings of 21st International Workshop on Graph-Theoretic Concepts in Computer Science (Lecture Notes in Computer Science 1017)*, pp. 37–48. Springer-Verlag (1995)
7. Erdős, P., Rényi, A.: On random graphs, I. *Publ. Math. (Debr.)* **6**, 290–297 (1959)
8. Evans, T., Lambiotte, R.: Line graphs, link partitions, and overlapping communities. *Phys. Rev. E* **80**(1), 016105 (2009)
9. Hardy, G.H., Littlewood, J.E., Pólya, G.: *Inequalities*, 2nd edn. Cambridge University Press, Cambridge (1988)
10. Krausz, J.: Démonstration nouvelle d'un théorème de Whitney sur les réseaux. *Mat. Fiz. Lapok* **50**, 75–85 (1943)
11. Krawczyk, M.J., Muchnik, L., Manka-Krason, A., Kulakowski, K.: Line graphs as social networks. *Phys. A* **390**, 2611–2618 (2011)
12. Lehot, P.G.H.: An optimal algorithm to detect a line graph and output its root graph. *J. ACM* **21**, 569–575 (1974)
13. Manka-Krason, A., Kulakowski, K.: Assortativity in random line graphs. *Acta Phys. Pol. B Proc. Suppl.* **3**(2), 259–266 (2010)
14. Manka-Krason, A., Mwijjage, A., Kulakowski, K.: Clustering in random line graphs. *Comput. Phys. Commun.* **181**(1), 118–121 (2010)

15. Mehlhorn, K., Näher, S.: LEDA: A Platform for Combinatorial and Geometric Computing. Cambridge University Press, Cambridge (1999)
16. Nacher, J.C., Ueda, U., Yamada, T., Kanehisa, M., Akutsu, T.: Line graphs as social networks. BMC Bioinfo. **24**(207), 2611–2618 (2004)
17. Nacher, J.C., Yamada, T., Goto, S., Kanehisa, M., Akutsu, T.: Two complementary representations of a scale-free network. Phys. A **349**, 349–363 (2005)
18. Naor, J., Novick, M.B.: An efficient reconstruction of a graph from its line graph in parallel. J. Algorith. **11**, 132–143 (1990)
19. Ore, O.: Theory of Graphs, vol. 21. American Mathematical Society Colloquium Publications (1962)
20. Roussopoulos, N.D.: A  $\max\{m, n\}$  algorithm for detecting the graph  $h$  from its line graph  $g$ . Info. Process. Lett. **2**, 108–112 (1973)
21. Simić, S.: An algorithm to recognize a generalized line graphs and output its root graph. Publ. Math. Inst. (Belgrade) **49**(63), 21–26 (1990)
22. Van Mieghem, P.: Graph Spectra for Complex Networks. Cambridge University Press, Cambridge (2011)
23. van Rooij, A.C.M., Wilf, H.S.: The interchange graph of a finite graph. Acta Math. Acad. Sci. Hung. **16**, 263–269 (1965)
24. Whitney, H.: Congruent graphs and the connectivity of graphs. Am. J. Math. **54**, 150–168 (1932)