molecules

MDPI

Article

Detection of Network Motif Based on a Novel Graph Canonization Algorithm from Transcriptional Regulation Networks

Jialu Hu [1,2,*] and Xuequn Shang [1]

[1] School of Computer Science, Northwestern Polytechnical University, West Youyi Road 127, Xi'an 710072, China; shang@nwpu.edu.cn

[2] Centre for Multidisciplinary Convergence Computing, School of Computer Science, Northwestern Polytechnical University, Dong Xiang Road 1, Xi'an 710129, China

* Correspondence: jhu@nwpu.edu.cn; Tel.: +86-29-8843-1519

Received: 4 November 2017; Accepted: 5 December 2017; Published: 10 December 2017

Abstract: Network motifs are patterns of complex networks occurring significantly more frequently than those in random networks. They have been considered as fundamental building blocks of complex networks. Therefore, the detection of network motifs in transcriptional regulation networks is a crucial step in understanding the mechanism of transcriptional regulation and network evolution. The search for network motifs is similar to solving subgraph searching problems, which has proven to be NP-complete. To quickly and effectively count subgraphs of a large biological network, we propose a novel graph canonization algorithm based on resolving sets. This method has been implemented in a command line interface (CLI) program sgip using the SeqAn library. Comparing to Babai's algorithm, this approach has a tighter complexity bound, $o(\exp(\sqrt{n}\log^2 n + 4\log n))$, on strongly regular graphs. Results on several simulated datasets and transcriptional regulation networks indicate that sgip outperforms nauty on many graph cases. The source code of sgip is freely accessible in https://github.com/seqan/seqan/tree/master/apps/sgip and the binary code in http://packages.seqan.de/sgip/.

Keywords: network motif; algorithms; graph canonization

1. Introduction

With the advent of high-throughput technologies in biology, the task of obtaining genetics and transcriptional information from specific tissues has become easier and more cost-effective. For example, the yeast two-hybrid (Y2H) system, Chromatin Immunoprecipitation Sequencing (ChIP-seq), and co-immunoprecipitation (coIP) coupled to mass spectrometry allow us to screen molecular interactions of a cell on a large scale. This achievement shed light on the research of understanding the underlying transcriptional regulation and network evolution. It can help us in unraveling the encrypted messages encoded in the structure and topology of protein-protein interaction (PPI) networks. To be analogous with sequence motif, network motif was proposed to characterize different types of complex networks. Network motifs are patterns of interactions occurring in a complex network at numbers that are significantly higher than those in randomized networks. They are considered as basic building blocks which play key roles in processing cellular signals in transcriptional regulatory networks. Currently, several major network motifs have been found in transcriptional regulatory networks, which include feed-forward loop (FFL), single input module (SIM), and dense overlapping regulons (DORs) [1]. The detection of a motif in a PPI network reveals that motif networks substantially influence the evolution conservation.

Many analysis tools have been developed for detecting network motifs in the last decade, including FANMOD [2], mfinder [3], MAVisto [4], etc. There are three major subproblems in the

detection: (1) generating an ensemble of proper random networks; (2) exhaustively enumerating all possible subgraphs in a real network and randomly generated networks; (3) grouping these subgraphs into different categories by the topological structure ; (4) calculating the statistical significance of each graph pattern. Our work mainly focuses on the third subproblem, which is a classical graph canonization problem. Graph canonization is a fundamental problem in theoretical and practical computer science. Its theoretical importance is derived from the relationship with the graph isomorphism problem (GIP). In practice, graph canonization algorithms have many applications in data mining [5,6] and pattern recognition [7,8]. Many of these algorithms were also used in chemistry [9–11]. Therefore, we are motivated to close the gap of computational complexity in the graph canonization problem and develop efficient programs to solve practical problems.

For the graph canonization problem, the complexity of the best known algorithms are moderately exponential $\exp(n^{\frac{1}{2}+o(1)})$ for general graphs, sub-exponential time $n^{n \log n}$ for tournaments [12], polynomial time for bounded valence graphs [13], and $o(\exp(2n^{\frac{1}{2}} \log^2 n))$ for strongly regular graphs [14]. An algorithm proposed in [15] can canonically label d-regular graphs in linear average time cdn. An improved work [16] was done for strongly regular graphs, which can solve the graph isomorphism problem in time $n^{O(n^{\frac{1}{3}} \log n)}$. A more recent work [17] stated that the graph isomorphism for hypergraphs of rank k can be solved in moderately exponential time $\exp(\widetilde{O}(k^2\sqrt{n}))$. In addition to all these theoretical results, practical algorithms were also developed and applied in various applications. One of the most notable and widely used tools is the nauty package [18], which employs a canonical labeling approach to compute automorphism groups. A comprehensive introduction of Mckay's algorithm was described in [19]. However, this algorithm also suffered from an exponential running time on a family of graphs constructed by Miyazaki [20].

Backtracking algorithms are also used to determine whether two general graphs are isomorphic by adopting a heuristic function in the search tree, such as SD [21], VF [22] and VF2 [23]. A comparison analysis of the performance of VF2 against VF, nauty, SD, and Ullman [24] was presented in [25]. A new symmetry-detection tool, saucy [26] was developed and tested on large structured graphs generated by CNF formulas, which can outperform nauty by several orders of magnitude. A more recent work, bliss [27], relying on the individualization and refinement scheme in the framework of backtracking, can outperform existing tools in most cases.

Inspired by the concept of the distinguishing set presented in [14], we first introduce the *resolving set* on graphs to solve the graph canonization problem based on backtracking approaches. In this paper, our aim is to design a canonical labeling algorithm that can extend Babai's algorithm to general cases and make it possible to use in practice. Furthermore, we attempt to close the remaining complexity gap of Babai's algorithm [14] on strongly regular graphs with $\lambda = \mu + 1$. It is noted that our aim is to solve the problem in most graph cases, not in the worst case. As far as we know, all the existing algorithms have exponential running time on some types of particularly designed graphs.

Our contribution in this work lies in the following three aspects: (1) we propose a novel algorithm, sgip, for solving the third subproblem in the detection of network motifs; (2) compared to previous algorithms, our approach is characterized by a tighter complexity bound of graph canonization problem; (3) we implemented this novel algorithm in the distribution of SeqAn library [28], which is comparable to the notable package nauty in practice.

2. Preliminary and Definitions

2.1. Subproblem of Grouping Subgraphs

Graph canonization is to find a canonical form for a given graph, which is still an open problem that is neither known to be a polynomial complete nor an NP-complete problem [29]. To group a set of subgraphs into different categories, it is necessary to find a canonical label for each category. Given two graphs $G = (V_1, E_1)$ and $H = (V_2, E_2)$, graph canonization is to find a canonical labeling function $\mathcal{L} : \mathbb{G} \to \mathbb{S}$ such that H is isomorphic to G if and only if $\mathcal{L}(G) = \mathcal{L}(H)$, where $\mathbb{G}$ represents a

set of graphs and $\mathbb{S}$ indicates a collection of strings. Obviously, the graph canonization problem (GCP) is at least as hard as the graph isomorphic problem.

2.2. Definitions

The resolving set concept was first introduced in the literature [30] under the term *locating set*. It is analogous to the distinguishing set in Babai's straightforward algorithm. Assuming each node has a unique position in a given graph, the position can be simply determined by a set of vertices in the graph. This set of vertices is then referred to as a resolving set of this graph. Let $d(u, v)$ be the *distance* (i.e. number of edges in the *shortest path*) between any two vertices $u, v \in V(G)$.

Definition 1. *Given an ordered subset of vertices $W = (w_1, w_2, \cdots, w_s)$ of $V(G)$, the s-tuple $r(v|W) = (d(v, w_1), d(v, w_2), \ldots, d(v, w_s))$ was referred to as the metric representation of v with respect to W. W is called a resolving set if and only if $r(v|W) \neq r(u|W)$ for distinct v and u.*

Definition 2. *The smallest resolving set of a graph G is called a minimum resolving set or* metric basis *of G, denoted as MRS.*

From the definition, we known that a *MRS* gives a *unique metric representation* for each vertex in G. A graph can be seen as an expansion of W in multidimensional space, and all vertices of G have their own unique coordinate positions in the expansion space. The *metric dimension* of G, denoted as $\mu(G)$, is the cardinality of the minimum resolving set of G. For instance, Petersen graph shown in Figure 1 can be determined by *MRS* (A, D, I), and its metric dimension is $\mu(G) = 3$. Each vertex has a unique metric representation with respect to *metric basis* (A, D, I). For example, $r(B|W) = (1, 2, 2)$, $r(E|W) = (1, 1, 2)$ and $r(C|W) = (2, 1, 1)$.

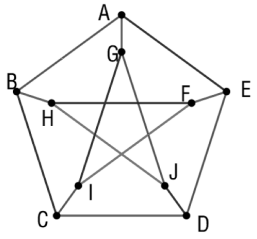


Figure 1. Metric dimension of the Petersen graph. Checking each node, we find that $W = (A, D, I)$ is one possible resolving set, and no smaller resolving set exists. Hence, the metric dimension of the Petersen graph is $\mu(g) = 3$.

Definition 3. *A graph G is said to be k-regular if $\forall v \in V(G)$, the degree of v is k. A k-regular graph G is called a strongly regular graph with parameters (n, k, λ, μ) if all the following conditions hold: (1) G is neither complete nor empty; (2) any two adjacent vertices of G have λ common neighbors; (3) any two nonadjacent vertices of G have μ common neighbors.*

3. Methods

To efficiently distinguish all subgraphs enumerated from biological networks, we propose a novel canonical labeling function based on minimum resovling sets. By definition, each minimum resolving set determines an ordered set of graph nodes. Given a graph, we exhaustively search all possible minimum resolving sets and assign a lexicographical leader as its canonical form.

3.1. Canonical Labeling Function

Let $W = (w_1, w_2, \cdots, w_s)$ be a resolving set for an input graph $\Gamma = (V\Gamma, E)$. As shown in the Definition 1, any pair of $v, u \in V\Gamma$ has distinctive metric representation, $r(v|W) \neq r(u|W)$.

Therefore, by defining that $r(v|W) \prec r(u|W)$ if there exists an integer j such that $d(u, w_j) < d(v, w_j)$ and $d(u, w_i) = d(v, w_i)$ for all integer $i < j$, we can sort all the objects in $V\Gamma$ by the metric representation taking W into account. So, each resolving set uniquely determines an adjacent matrix $Adj_W(\Gamma)$. Then, we give the definition that a canonical label is assigned by the lexicographical leader of all the adjacent matrix yielded by all possible resolving sets. Mathematically, the canonical labeling function can be written in

$$\mathcal{L}(\Gamma) = \min_{\preceq lex}\{Adj_W(\Gamma)|W \in \Omega\}$$

where Ω indicates the collection of minimum resolving sets for graph Γ. So, in order to compute the canonical label, an exhaustive search should be performed on a list of all potential resolving sets. Then, feasible object combinations should be selected and the optimal one should be found, corresponding to the least lexicographical adjacent matrix. Obviously, the complexity of our algorithm only depends on $\mu(\Gamma)$. Supposing $|W| = k$, we can easily draw a conclusion that all $k!$ permutations of W are resolving sets if W is a resolving set through Definition 1. Therefore, if there are C different feasible combinations of vertices in all which yield *minimum resolving sets*, the number of minimum resolving sets is $|\Omega| = C \cdot \mu(\Gamma)!$. Let Γ be a strongly regular graph which is neither the union of disjoint complete graph nor the complement of such a graph. Since the distinguishing set is equivalent to the resolving set for strongly regular graphs, the metric dimension $\mu(\Gamma)$ is bound to $\lfloor 2\sqrt{n}\log n \rfloor - 3$ according to Babai's complexity theory. However, this exhaustive approach also suffers in a large computation for graphs with high metric dimension, such as complete graphs ($\mu(\Gamma) = n - 1$) and stars ($\mu(\Gamma) = n - 2$). Therefore, it is impractical to use on general cases.

3.2. Extended Algorithm on General Cases

In order to find a canonical label for these graphs of high dimension, we introduced the concept of *parity nodes*, also called *twin vertices* [31], to prevent redundant computation, because some distinct resolving sets can deduce to identical adjacent matrix. For simplicity, notations are defined as follows on general graphs. For the undirected case, $\mathcal{N}(v)$ refers to neighborhoods of node v, $\delta(v)$ indicates degree of node v. For the directed case, we denoted with $\mathcal{N}^+(v)$, $\mathcal{N}^-(v)$, and $\mathcal{N}(v)$, respectively, as in, out, and total neighborhoods of v; analogously with $\delta^+(v)$, $\delta^-(v)$, and $\delta(v)$, we denote it as in, out, and total degree of v.

Definition 4. *For undirected graph $\Gamma = (V\Gamma, E)$, two vertices $u, v \in V\Gamma$ are called parity nodes, and denoted as $u \leftrightarrow v$, iff it holds*

$$\mathcal{N}(u)\backslash v = \mathcal{N}(v)\backslash u;$$

otherwise, they are non-parity nodes, denoted as $u \nleftrightarrow v$.

Definition 5. *For directed graph $\Gamma = (V\Gamma, E)$, two vertices $u, v \in V\Gamma$ are called parity nodes and denoted as $u \leftrightarrow v$ iff all of the following conditions are satisfied*

$$\begin{cases} \delta^+(u) = \delta^+(v) \\ \delta^-(u) = \delta^-(v) \\ \mathcal{N}^+(u)\backslash v = \mathcal{N}^+(v)\backslash u \\ \mathcal{N}^-(u)\backslash v = \mathcal{N}^-(v)\backslash u \end{cases}$$

otherwise they are non-parity nodes, denoted as $u \nleftrightarrow v$.

Parity nodes define an *equivalence relation* on $V\Gamma$. The *reflexive* property can be easily verified. From the definition of parity nodes, we can easily figure out the symmetric and transitive properties: (1) for any two vertices $u, v \in V\Gamma$, if $u \leftrightarrow v$, then $(u, v) \in E \Leftrightarrow (v, u) \in E$; (2) for any three vertices $u, v, w \in V\Gamma$, if $u \leftrightarrow v$ and $v \leftrightarrow w$ then $u \leftrightarrow w$. The equivalence relation defined by parity nodes is an invariant for all types of graphs. So, the node set $V\Gamma$ can be classified into a set of *equivalent sets* by parity

nodes. The appearance of parity nodes is a major cause of high metric dimension in general graphs, since nodes from one equivalent set cannot be distinguished by any other nodes through metric representation except themselves. For instance, it is clear that all nodes of a complete graph K_q are in one equivalent set. So, $\mu(K_q) = q - 1$ that makes the search task intractable. To extend the former algorithm working on general cases, we propose the concept of the *improved resolving set* that can assign canonical label for input graphs with a smaller metric dimension than the *resolving set*.

For a graph $\Gamma = (V\Gamma, E)$, suppose VS is a subset of $V\Gamma$ such that for any equivalent set S in $V\Gamma$ it implies $|VS \cap S| = 1$, W which is a subset of VS is called *advanced resolving set* if any two elements $v_i, v_j \in VS$, it satisfies $r(v_i|W) \neq r(v_j|W)$. Those with minimum cardinality are called *minimum advanced resolving sets*, and the minimum cardinality is called the *advanced metric dimension* of Γ, denoted as $\nu(\Gamma)$.

According to the definition, the *improved resolving set* only takes one node from each equivalent set into account. Obviously, the *advance metric dimension* is smaller than the *metric dimension*. Then, it concludes that the *improved resolving set* determines a unique order for a part of vertices in $V\Gamma$. However, one may ask the question: does it work for the whole graph? In other words, how does it determine a canonical label for a given graph? This question will be answered in two steps. Before all that, it is notable that given a graph Γ with n vertices, P_n indicates the set of permutations and $\forall\sigma \in P_n$, $Adj_\sigma(\Gamma)$ represents the adjacent matrix of graph Γ.

For a graph Γ and $\forall\sigma \in P_n$, if $u, v \in V\Gamma$ and $u \leftrightarrow v$, then by inverting the order of u and v in σ we obtain a new permutation τ such that $Adj_\sigma(\Gamma) = Adj_\tau(\Gamma)$. Let $\sigma_x = v$, $\sigma_y = u$, $A = Adj_\sigma(\Gamma)$, and $B = Adj_\tau(\Gamma)$. Then, $\tau_x = u$ and $\tau_y = v$, and there must exist a permuted matrix P, such that $A = P^{-1}BP$. By Definition 5, we know that $\mathcal{N}^+(v)\backslash u = \mathcal{N}^+(u)\backslash v$. This implies that $A(x, j) = A(y, j)$, $j \notin \{x, y\}$. Similarly, $\mathcal{N}^-(v)\backslash u = \mathcal{N}^-(u)\backslash v$ implies that $A(j, x) = A(j, y)$, $j \notin \{x, y\}$. Additionally, $v \leftrightarrow u$ implies $A(x, y) = A(y, x)$, and clearly, $A(x, x) = A(y, y) = 0$, hence it follows $A = PAP^{-1} = B$.

Through the theorem stated above, the *advanced resolving set* is able to determine a canonical label for input graphs. All of the permutations within the equivalent set yield the same adjacent matrix. Hence, our algorithm for assigning a canonical label for a given graph Γ can be done in the following four steps: (1) determine equivalent sets over $V\Gamma$; (2) calculate $\nu(\Gamma)$ on the graph Γ; (3) give a brute force search over a set of non-equivalent nodes; (4) choose the optimal one which has the lexicographical leader. This improved algorithm provides a better performance on these graphs with rich equivalent sets. The *advanced resolving set* enables our algorithm to work on many types of graphs.

3.3. Complexity Analysis

As stated above, the complexity of the straightforward algorithm and the extended algorithm primarily depend on *metric dimension* and *advanced metric dimension*, respectively. According to Babai's theorem [14], the complexity of his algorithm on strongly regular graphs is bounded by $o(\exp(2\sqrt{n}\log^2 n))$. Here, we proposed a statistical model revealing that a tighter bound of complexity exists on strongly regular graph with $\mu = \lambda + 1$.

Supposing $\nu(\Gamma) = s$ for a general graph Γ, it takes time $O(n^2)$ to compute equivalent sets, $O(n^2)$ to calculate the distance matrix, and $O(sn^2)$ time to test whether it is a resolving set. So, the complexity of our algorithm is simply bounded by $o(n^{s+3})$. Since $\mu(\Gamma) \leq \sqrt{n-1}\log n + 1$ for strongly regular graphs with $\mu = \lambda + 1$, we set a new tighter bound $o(\exp(\sqrt{n-1}\log^2 n + 4\log n))$ in recognizing this kind of particularly designed graphs.

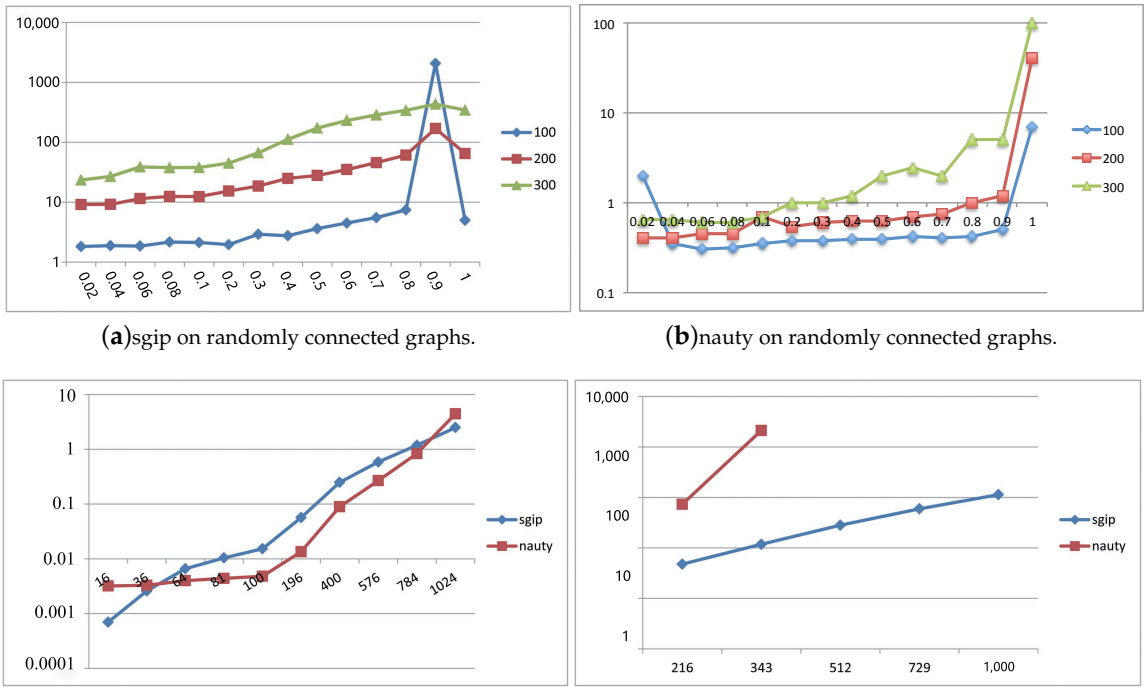
4. Results and Discussion

All of our experiments were carried out on a Linux PC with 2.2 GHz CPU, 4 G memory. The algorithm was implemented in a package named sgip in the distribution of the C++ library SeqAn. The source code of sgip is freely accessible at https://github.com/seqan/seqan/tree/master/apps/sgip and the binary code at http://packages.seqan.de/sgip/.

4.1. Tests on Simulated Data

To test the performance, our algorithm and the nauty algorithm were tested on a benchmark database [32] containing 72,800 couples of simple graphs, of which 18,200 couples are isomorphic graphs and 54,600 couples have subgraph isomorphism mapping among them. These graphs are in several different categories, which include randomly connected graphs, regular meshes, bounded valence graphs, irregular meshes, and irregular bounded valence graphs. In addition, 3400 new couples of isomorphic random graphs were generated.

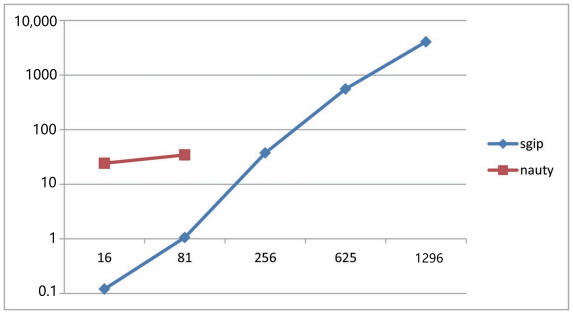
To provide a comparison of sgip with nauty, Figure 2 consists of five plots sketching the running time of two packages over various kinds of graphs. Average running time was estimated by sampling 100 couples of general graphs for each density. Figure 2a,b give an overview of the growing trends of sgip and nauty with respect to a density series. It is true that the running time of the two tools basically depends on both the density and size (number of nodes) of given graphs. It is worth noting that sgip took approximately 20 min to figure it out on graphs with 100 nodes and density of 0.9. We think that this was caused by some special graphs of highly advanced metric dimension. Overall, all kinds of tested general graphs could be distinguished in a reasonably short time, including both sparse and dense graphs, in spite of a slightly longer running time than that of nauty in general usage. However, there are also many solid pieces of evidences showing that sgip outperforms nauty in some other cases. For instance, Figure 2c–e depicts plots of running time cost on mesh graphs versus graph size. It is shown that nauty had a dramatic increase when the size of the mesh graphs got larger, and it became impractical to use when the size of the 3D mesh graphs exceeded a certain limit (around 300). More seriously, on 4D mesh graphs, it was unable to compute when the size was only above 81. In contrast, sgip was more than 100 times faster than nauty on 3D mesh graphs. The advantage of sgip became increasingly conspicuous with the growth of size and dimension on mesh graphs. Apparently, sgip is rather feasible for use in multi-dimensional mesh graphs.



(**a**)sgip on randomly connected graphs. (**b**)nauty on randomly connected graphs.

(**c**) run-time comparison on 2D mesh. (**d**) run-time comparison on 3D mesh.

Figure 2. *Cont.*



(e) run-time comparison on 4D mesh.

Figure 2. Performance comparison of sgip and nauty on different types of graph. Each experiment was tested on 100 couples of isomorphic graphs. (**a**,**b**) Horizontal axis represents the density of tested graphs (density is the ratio of the number of existing directed edges and $n(n-1)$), vertical axis represents the running time of the responding tool. (**c**,**d**,**e**) Horizontal axis represents the number of vertices of the tested graphs, and the vertical axis is run-time on mesh graphs.

4.2. Tests on Transcriptional Regulation Networks

To apply our algorithm in the detection of network motifs from transcriptional regulation networks, we employed the *ESU* algorithm to search for all possible subgraphs in PPI networks. In the following step, the program sgip was adapted to solve the subproblem of grouping subgraphs into different categories. Our approach was performed on transcriptional regulation networks of *Saccharomyces cerevisiae* and *Escherichia coli*. One thousand random graphs were generated for each species by randomly switching any two interactions of the real network $10 * |E|$ times, which can guarantee the same degree distribution with the real network. Finally, p-value was used to measure the statistical significance of each pattern. As shown in Table 1, we found three types of network motif which frequently occur in real biological networks. These patterns are feed-forward loop, single input module, and pairs of operons controlled by the same two transcriptional factors. All of their p-value scores were less than 1×10^{-13}.These results indicate the practicability of our algorithm in the detection of network motifs in transcriptional regulation networks.

Table 1. Statistically significant patterns appearing in transcriptional regulation networks. Pattern A refers to a coherent feed-forward loop whose connections are x→y→z and x→z. Pattern B refers to these subgraphs with single input module (>10 nodes). Pattern C refers to pairs of operons controlled by the same two transcription factors.

Patterns	Appearances in Real Network		Appearances in Randomized Network		*p*-Value	
	Yeast	*E. coli*	Yeast	*E. coli*	Yeast	*E. coli*
A	65	34	10.2 ± 5	4.5 ± 2	0	0
B	122	79	33.5 ± 12	30 ± 6	8.2×10^{-14}	1.1×10^{-16}
C	396	203	108 ± 29	55 ± 10	0	0

5. Conclusions

In this paper, we proposed a novel graph canonization algorithm sgip based on resolving sets to count all possible subgraphs of a large transcriptional regulation network. It has been proven that there exists a tighter bound of metric dimension on strongly regular graphs with $\mu = \lambda + 1$. The algorithm was implemented in the distribution of C++ library SeqAn. To test the performance, both nauty and sgip were performed on the same benchmark datasets with identical computational resources. The result shows that sgip is efficient and practical to use in most general cases and outperformed nauty in some particular designed graph structures. Applied in transcriptional regulation networks, our approach successfully found three typical network motifs which were significantly more frequent than those in randomized networks. This proves the efficiency of sgip in the detection of network

motifs from large biological networks. Graph canonization problems are widely used in many other scientific and engineering fields, such as pattern recognition, chemical structure, etc. Hopefully, the practicability of our algorithm can benefit more researchers in their work and studies.

Acknowledgments: This project has been funded by the National Natural Science Foundation of China (Grant No. 61332014 and 61702420); the China Postdoctoral Science Foundation (Grant No. 2017M613203); the Natural Science Foundation of Shaanxi Province (Grant No. 2017JQ6037); the Fundamental Research Funds for the Central Universities (Grant No. 3102015QD013).

Author Contributions: Jialu Hu and Xuequn Shang conceived and designed the experiments; Jialu Hu performed the experiments; Jialu Hu analyzed the data; Jialu Hu contributed analysis tools; Jialu Hu and Xuequn Shang wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest. The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

References

1. Shen-Orr, S.S.; Milo, R.; Mangan, S.; Alon, U. Network motifs in the transcriptional regulation network of *Escherichia coli*. *Nat. Genet.* **2002**, *31*, 64–68.
2. Wernicke, S.; Rasche, F. FANMOD: A tool for fast network motif detection. *Bioinformatics* **2006**, *22*, 1152–1153.
3. Kashtan, N.; Itzkovitz, S.; Milo, R.; Alon, U. Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics* **2004**, *20*, 1746–1758.
4. Schreiber, F.; Schwöbbermeyer, H. MAVisto: A tool for the exploration of network motifs. *Bioinformatics* **2005**, *21*, 3572–3574.
5. Washio, T.; Motoda, H. State of the art of graph-based data mining. *ACM SIGKDD Explor. Newsl.* **2003**, *5*, 59–68.
6. Rückert, U.; Kramer, S. Frequent free tree discovery in graph data. In Proceedings of the 2004 ACM Symposium on Applied Computing, Nicosia, Cyprus, 14–17 March 2004; Association for Computing Machinery (ACM): New York, NY, USA, 2004; pp. 564–570.
7. Conte, D.; Foggia, P.; Sansone, C.; Vento, M. Thirty years of graph matching in pattern recognition. *Int. J. Pattern Recognit. Artif. Intell.* **2004**, *18*, 265–298.
8. Von der Malsburg, C. Pattern recognition by labeled graph matching. *Neural Netw.* **1988**, *1*, 141–148.
9. Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 31–36, doi:10.1021/ci00057a005.
10. McNaught, A. The IUPAC International Chemical Identifier: InChl—A New Standard for Molecular Informatics. In *Chemistry International*; International Union of Pure and Applied Chemistry (IUPAC): Durham, NC, USA, 2006; Volume 28.
11. Hähnke, V.; Rupp, M.; Krier, M.; Rippmann, F.; Schneider, G. Pharmacophore alignment search tool: Influence of canonical atom labeling on similarity searching. *J. Comput. Chem.* **2010**, *31*, 2810–2826.
12. Babai, L.; Luks, E. Canonical labeling of graphs. In Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing, Boston, MA, USA, 25–27 April 1983; pp. 171–183.
13. Luks, E. Isomorphism of graphs of bounded valence can be tested in polynomial time. *J. Comput. Syst. Sci.* **1982**, *25*, 42–65.
14. Babai, L. On the complexity of canonical labeling of strongly regular graphs. *SIAM J. Comput.* **1980**, *9*, 212.
15. Kucera, L. Canonical labeling of regular graphs in linear average time. In Proceedings of the 28th Annual Symposium on Foundations of Computer Science, Los Angeles, CA, USA, 12–14 October 1987; IEEE Computer Society: Washington, DC, USA, 1987; pp. 271–279.
16. Spielman, D.A. Faster isomorphism testing of strongly regular graphs. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory Of Computing, Philadelphia, PA, USA, 22–24 May 1996; Association for Computing Machinery (ACM): New York, NY, USA, 1996; pp. 576–584.
17. Babai, L.; Codenotti, P. Isomorhism of Hypergraphs of Low Rank in Moderately Exponential Time. In Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science, Philadelphia, PA, USA, 25–28 October 2008; IEEE Computer Society: Washington, DC, USA, 2008; pp. 667–676.

18. McKay, B. Practical Graph Isomorphism. *J. Symb. Comput.* **1981**, *60*, 94–112.
19. Hartke, S.G.; Radcliffe, A.J. *Communicating Mathematics: A Conference in Honor of Joseph A. Gallian's 65th Birthday, July 16–19, 2007, University of Minnesota, Duluth, Minnesota*; Chapter Mckay's Canonical Graph Labeling Algorithm; American Mathematical Society: Providence, RI, USA, 2009; p. 99.
20. Miyazaki, T. Groups and computation II: Workshop on groups and computation. In *Groups and Computation II Workshop on Groups and Computation June 710 1995*; Chapter the Complexity Of McKay'S Canonical Labeling Algorithm; American Mathematical Society: Providence, RI, USA, 1997; p. 239.
21. Schmidt, D.C.; Druffel, L.E. A Fast Backtracking Algorithm to Test Directed Graphs for Isomorphism Using Distance Matrices. *J. ACM* **1976**, *23*, 433–445.
22. Cordella, L.; Foggia, P.; Sansone, C.; Vento, M. Performance evaluation of the VF graph matching algorithm. In Proceedings of the International Conference on Image Analysis and Processing, Venice, Italy, 27–29 September 1999; pp. 1172–1177.
23. Cordella, L.; Foggia, P.; Sansone, C.; Vento, M. An improved algorithm for matching large graphs. In Proceedings of the 3rd IAPR-TC15 Workshop on Graph-Based Representations in Pattern Recognition, Ischia, Italy, 23–25 May 2001; pp. 149–159.
24. Ullmann, J.R. An Algorithm for Subgraph Isomorphism. *J. ACM* **1976**, *23*, 31–42.
25. Foggia, P.; Sansone, C.; Vento, M. A Performance Comparison of Five Algorithms for Graph Isomorphism. In Proceedings of the 3rd IAPR-TC15 Workshop on Graph-Based Representations in Pattern Recognition, Ischia, Italy, 23–25 May 2001; pp. 188–199.
26. Darga, P.T.; Liffiton, M.H.; Sakallah, K.A.; Markov, I.L. Exploiting structure in symmetry detection for CNF. In Proceedings of the 41st Annual Design Automation Conference, San Diego, CA, USA, 7–11 June 2004; Association for Computing Machinery (ACM): New York, NY, USA, 2004; pp. 530–534.
27. Junttila, T.; Kaski, P. Engineering an efficient canonical labeling tool for large and sparse graphs. In Proceedings of the 9th Workshop on Algorithm Engineering and Experiments and the 4th Workshop on Analytic Algorithms and Combinatorics, New Orleans, LA, USA, 6 January 2007; pp. 135–149.
28. Döring, A.; Weese, D.; Rausch, T.; Reinert, K. SeqAn An efficient, generic C++ library for sequence analysis. *BMC Bioinform.* **2008**, *9*, 11.
29. Garey, M.R.; Johnson, D.S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*; W.H. Freeman & Co.: New York, NY, USA, 1979.
30. Slater, P.J. Leaves of trees. In Proceedings of the Sixth Southeastern Conference on Combinatorics, Graph Theory, and Computing, Boca Raton, FL, USA, 17–20 February 1975; pp. 549–559.
31. Carmen, H.; Mercé, M.; Pelayo, I.M.; Carlos, S.; Wood, D.R. Extremal Graph Theory for Metric Dimension and Diameter. *Electron. J. Comb.* **2010**, *17*, R30.
32. Foggia, P.; Sansone, C.; Vento, M. A database of graphs for isomorphism and sub-graph isomorphism benchmarking. In Proceedings of the 3rd IAPR-TC15 International Workshop on Graph-based Representations, Ischia, Italy, 23–25 May 2001; pp. 176–187.

Sample Availability: Samples of the compounds are available from the authors.