Discovering Colored Network Motifs

Pedro Ribeiro and Fernando Silva

Abstract Network motifs are small overrepresented patterns that have been used successfully to characterize complex networks. Current algorithmic approaches focus essentially on pure topology and disregard node and edge nature. However, it is often the case that nodes and edges can also be classified and separated into different classes. This kind of networks can be modeled by colored (or labeled) graphs. Here we present a definition of colored motifs and an algorithm for efficiently discovering them. We use g-tries, a specialized data-structure created for finding sets of subgraphs. G-Tries encapsulate common sub-structure, and with the aid of symmetry breaking conditions and a customized canonization methodology, we are able to efficiently search for several colored patterns at the same time. We apply our algorithm to a set of representative complex networks, showing that it can find colored motifs and outperform previous methods.

1 Introduction

Network motifs are small overrepresented subgraphs appearing more frequently than what would be expected in randomized networks with similar topological characteristics [1]. They have been extensively used in may domains, such as biological [2] or social [3] networks. The vast majority of past research deals essentially with pure structural motifs, with nodes and edges being of the same type, i.e., with no associated *color* or *label*. This is a restriction that limits the information we can obtain, because it is often the case that nodes or edges are of different nature. For example, in metabolic networks, we can distinguish between two sets of nodes: reactions and chemical compounds [4]. By ignoring these labels we may be missing important patterns, and it has been shown that by associating different colors to the

Pedro Ribeiro · Fernando Silva
CRACS & INESC-TEC, DCC-FCUP, Universidade do Porto, Portugal
e-mail: pribeiro@dcc.fc.up.pt, fds@dcc.fc.up.pt

nodes, their information content is richer [5]. The same can be said about edges, and previous experiments have shown that we would gain if it was possible to distinguish between different types of connections in biological networks [6].

Discovering network motifs is a computationally hard task, intimately connected to the subgraph isomorphism problem. Current methods essentially rely on computing the frequency of subgraphs on both the original graph and on an ensemble of randomized similar graphs. This is traditionally done in one of two distinct approaches: either we enumerate all subgraphs of a certain size and then compute their isomorphisms, or we query individually for one subgraph at a time. The first *network-centric* approach demands that we always look for all possible subgraphs, with no option for looking for specific types. The second *subgraph-centric* approach, considers one subgraph type at a time, not reusing information from previous searches. We recently introduced the g-trie data-structure [7], allowing a *set-centric* approach, in which we can search specifically for a certain set of subgraphs. The algorithms developed show substantial efficiency gains on pure structural motifs, when comparing to previous approaches [8].

In this paper we present an efficient g-tries based algorithm able to discover colored motifs. Our main contribution is two-fold. First, we give a clear definition of what a colored motif can be, including how we can compute its statistical significance. Secondly, we provide an extension of the g-trie data structure and associated algorithms in order to allow the usage of color information on both nodes and edges. This allows for richer searches that do not discard information about the nature of the network constituents. We empirically evaluate our algorithms in a set of representative graphs, showing the feasibility of our approach and how we can outperform past methods.

The remainder of the paper is organized as follows. Section 2 establishes a common graph terminology, explains the problem being tackled, and overviews related work. Section 3 describes the g-trie data structure and details how it can be used for discovering colored motifs. Section 4 shows the results of an empirical evaluation of the developed algorithms, when applied to a set of complex networks, and compares it to the performance obtained by a competing algorithm. Finally, section 5 concludes the paper.

2 Preliminaries

2.1 Graph notation

We briefly review the main concepts and notation that we use. A *graph* G is composed of a set $V(G)$ of *vertices* or *nodes* and a set $E(G)$ of *edges* or *connections*. A k-graph is a graph of size k, i.e., with k nodes. Every edge is composed of a pair (u,v) of two *endpoints* in the set of vertices. A graph is said to be *colored* if we associate colors, or labels, to each of its vertices and/or edges. Note that non-colored graphs can be seen as a simpler case with only one color for nodes and edges. The *degree* of a vertex u is the number of connections it has to other nodes, and its *neighbourhood*, denoted as $N(u)$, is composed by the set of vertices $v \in V(G)$ such that v and u share an edge. The *exclusive neighborhood* of a vertex v of graph G relative to a subgraph G_k is defined as $N_{exclusive}(v, G_k) = \{u : u \in N(v) \wedge u \notin N(G_k) \wedge u \notin G_k\}$.

A *subgraph* G_k of a graph G is a graph of size k in which $V(G_k) \subseteq V(G)$ and $E(G_k) \subseteq E(G)$. This subgraph is said to be *induced* when $\forall u, v \in V(G_K)$, $(u, v) \in E(G_k)$ if and only if $(u, v) \in E(G)$. The neighborhood of a subgraph G_k, denoted by $N(G_k)$ is the union of $N(u)$, $\forall u \in V(G_k)$. Two graphs G and H are said to be *isomorphic*, if there is a one-to-one mapping between the vertices of both graphs (including colors) and there is an edge of color c between two vertices of G if and only if their corresponding vertices in H also have an edge of color c.

2.2 Colored Motifs

Milo et al. provided the first definition of network motifs as *"patterns of interconnections occurring in complex networks in numbers that are significantly higher than those in similar randomized networks"* [1]. Here we introduce a similar definition for colored motifs, but with the necessary adaptions. To put it to practice, we need to establish two different concepts: what is the *frequency* of a subgraph and what are *similar randomized networks*.

For the first concept, we resort to the standard definition in the motif detection realm, considering only induced subgraphs, and with two occurrences being different if they have at least one node or edge not shared. As to the second concept, the idea is to test subgraph significance by comparing the frequency of the subgraph in the original network with its frequency on a large number of similar random networks. To be sure that the results are specific to a particular network one should use randomized networks as close as possible to the original one. Milo et al. suggested to keep single-node properties, namely its degree. Adapting this to the colored case, we maintain all color related information. The randomized networks have the same set of nodes and colors, each node keeps the same amount of edges in each color, and each edge connects endpoints of the same colors it was connecting in the original network. We can say that we keep the *colored degree sequence*.

Figure 1 shows an example colored motif following our definition, appearing 4 times on the original network. Different occurrences (indicated with thick lines) may share some of the nodes (e.g., $\{1, 6, 7\}$ and $\{1, 7, 11\}$ share nodes 1 and 7). Colors enrich the information and allow us to distinguish between different types of triangle subgraphs. Note how the randomized networks respect the colored degree sequence of the original, with each node keeping the exact same type of colored connections. For instance, in the original network node 1 has two dashed connections to light nodes, one dashed connection to a black node and one continuous connection to another black node. The same happens at each of the similar randomized networks. However, the subgraph we are considering is much less frequent on these networks

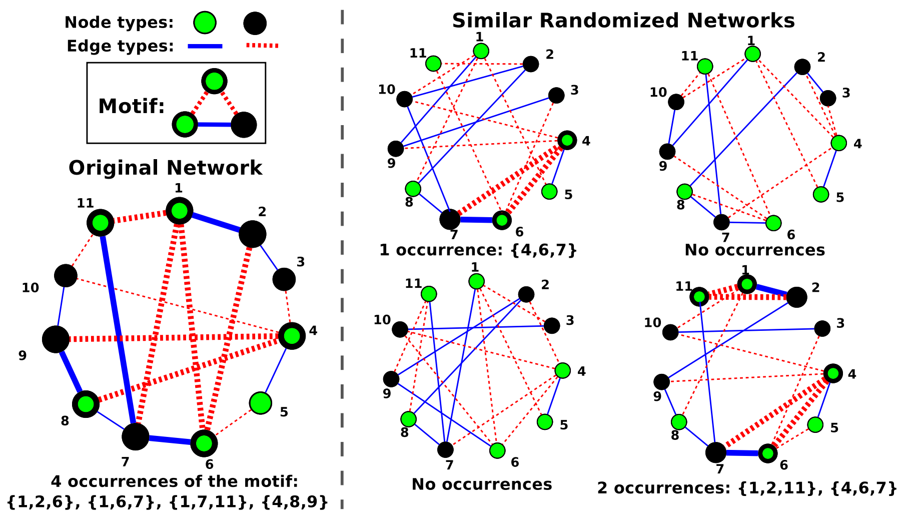
Node types:
Edge types:
Motif:
Original Network
4 occurrences of the motif: {1,2,6}, {1,6,7}, {1,7,11}, {4,8,9}
Similar Randomized Networks
1 occurrence: {4,6,7}
No occurrences
No occurrences
2 occurrences: {1,2,11}, {4,6,7}

Fig. 1 An example colored network motif with 3 nodes.

than in the original one, and hence we consider it a motif, i.e., the subgraph is overrepresented.

2.3 *Related work*

There has been some work on motif detection using colors only in nodes [9], but not with colored edges, even if it was acknowledged that incorporating connection types would lead to richer results [6]. The work by Adami et al. [5] considered the case of colors only in nodes and uses an entropy-based measurement to determine significance. Quian et al. [10] also use colors solely in nodes and swap colors in the network, that is, the randomized networks are topological equivalent to the original, but with a different permutation of the colors in the nodes. Schbath et al. [11] also incorporate color in motif detection, but they only consider sets of connected colors, ignoring the exact connections between the respective nodes.

The only work we are aware of that directly supports motif finding with colors both in nodes and edges is the FanMod tool [12], which implements the ESU algorithm ESU [13]. Nevertheless, a clear formal definition of colored motif is not given. ESU was initially created for non-colored topological motif detection, a problem for which there are several possible methodologies. Kavosh [14], FaSE [15] and ESU itself are examples of network-centric approaches, where all subgraphs of a given size are counted by an enumeration followed by isomorphism tests to determine the subgraph class of each found occurrence. On the other hand, algorithms such as the one by Grochow and Kellis [16] only search for one subgraph at a time, with no re-usage of any kind of information between the computation of different subgraphs.

Here we use a set-centric approach, extending our own g-trie data structure [7], which previously did not account for color information. This methodology has been shown to be very competitive with running times that can outperform previous existing methods in the case of uncolored motifs [8].

In this paper we concentrate on exact algorithms, that are able to compute exact frequencies. There are however approximation techniques that trade accuracy for better running times. They are based on methodologies such as sampling [13, 17, 18]. The work we show in this paper can be further extended in the future to support such an approach.

3 Finding colored motifs with g-tries

3.1 The colored g-trie data structure

A g-trie is a multiway tree (with a variable number of children per node) able to store and describe a set of graphs. To avoid further ambiguities, from now on we will use the term node for referring to g-trie tree nodes, and the term vertex to refer to the actual graph vertices stored in the g-trie. Each node contains information about a single vertex and connections to ancestor vertices. In order to support colors, we include in this information the color of the respective node and the color of each edge. A path from the root to a leaf node defines a subgraph.

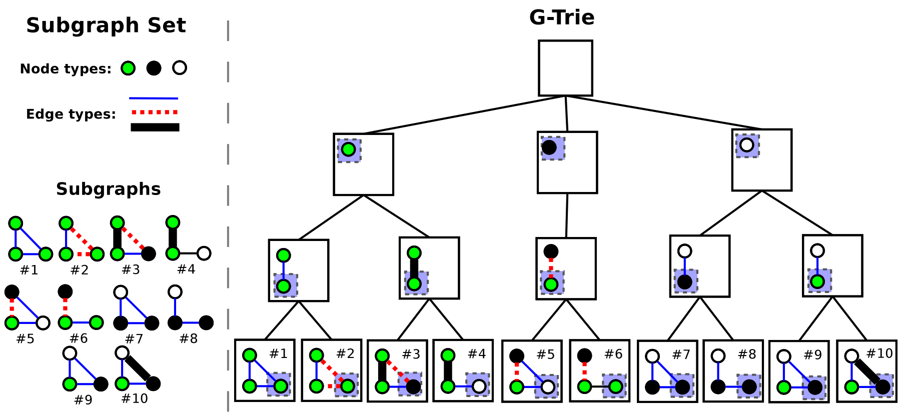

Fig. 2 An example colored g-trie containing 10 different subgraphs.

Figure 2 exemplifies the concept, with a g-trie storing 10 colored subgraphs, with 3 different vertex colors and 3 different edge colors. Each g-trie node adds a new graph vertex (inside the small square) to the already existing ones in the ancestor nodes. Note how descendants of a node share a common colored subtopology.

3.2 Motif discovery

Our methodology has two input parameters: a network to analyze and k, the size (in vertices) of the motifs we are looking for. The flow of the algorithm is, in its essence, the same as what was done originally by Milo et al [1]. First, compute the frequency of all k-sized subgraphs of the original network, i.e., a *subgraph census*. Secondly, compute the frequency of these subgraphs in the set of similar randomized networks. In the following sections we describe how these two steps are made.

3.3 Census in original network

The original g-trie algorithm for computing subgraph frequencies needs as input a set of subgraphs to store. When no colors are used and the size k is small, it is possible to use as input the set of all possible k-subgraphs. With colors, the number of different possibilities is much larger (due to the possible permutations between colors) and this option becomes unfeasible even for a relatively small k.

Given this, we opted to follow a network-centric approach for this initial step, i.e., we search for all occurrences of k connected nodes and identify their topological type. At the core, we need an algorithm for enumerating the sets of connected nodes. We opted to use ESU [13] as the base algorithm but in order to avoid a large number of isomorphism tests we incorporate the g-trie in the enumeration process, in a similar way to what was done with the FaSE algorithm [15].

Algorithm 1 Census of subgraphs with k vertices in the original network G.

```
1: $T$ := new empty g-trie
2: for all $v \in V(G)$ do
3:     EXTENDSUBGRAPH($\{v\}, \{u \in N(v) : u > v\}, v, T$)
4: for all $n \in T.leaves()$ do
5:     $frequency[CanonicalLabel(n.Graph)]$ += $n.count$
6: procedure EXTENDSUBGRAPH($V_{Subg}, V_{Ext}, v, T$)
7:     $c$ := last vertex in $V_{Subg}$ and its connections to previous nodes
8:     if T.notHasChild($c$) then T.CREATECHILD($c$)
9:     T := T.child($c$)
10:    if $|V_{Subg}| = k$ then $T.count += 1$
11:    else
12:        while $V_{Ext} \neq \emptyset$ do
13:            remove any $w \in V_{Ext}$
14:            $V'_{ext} := V_{ext} \cup \{u \in N_{exclusive}(w, V_{subg}) : u > v\}$
15:            EXTENDSUBGRAPH($V_{Subg} \cup \{w\}, V'_{ext}, v, T$)
```

Algorithm 1 details our approach. At the core we are using the ESU algorithm (lines 2 to 3 and 10 to 15), which enumerates all k-subgraphs of a network once and only once. It works by maintaining two vertex lists: V_{Subg} and V_{ext}. The first one represents the current partial subgraph being constructed (set of connected vertices) and the latter a list of all neighbor vertices that can be added. Initially, it chooses each vertex v in the original graph G to be a possible starting point, and its neighbors as possible extensions (lines 2 and 3). Then it recursively removes each element of the extension list (line 13) and creates a new list of possible extensions (line 14). The usage of the exclusive neighborhood, along with the condition $u > v$, breaks symmetries and guarantees that each occurrence is only found once. When the size of V_{Subg} reaches k it means a new occurrence of a k-subgraph is found.

ESU execution naturally creates an implicit recursive search tree. At each time, one new node is added to the current partial subgraph, and it corresponds to a ramification in the g-trie. If that ramification already existis, we just update our current position in the g-trie (line 9). If not, we first create the ramification (line 8) and then follow that path. In the end, a g-trie path from the root to any leaf corresponds to a different node permutation of a certain graph type. We compute a canonical labeling for each leaf in order to identify its subgraph type (lines 4 and 5) but we avoid the need for that computation on all other occurrences of the same type whose node permutation corresponds to an automorphism, i.e., corresponds to the same path in the g-trie. Note that two leaves may be isomorphic, because the order in which vertices are traversed may implicitly define different paths. Ideally, one wants a single canon path in the g-trie for the same subgraph type but in here we trade memory for better running time, postponing the canonization for the randomized networks. The actual canonical labeling algorithm used is explained in section 3.5.

3.4 Census in the randomized networks

The bulk of the computation work is to discover the subgraphs frequency in the randomized networks. Typically one uses around 100 random networks [1], with each taking an amount of time similar to the time needed for the subgraph census in the original network. Since we know which subgraph types appeared in the original network, we limit our search to those. Our approach is to build a new g-trie containing only a single representation of each subgraph we are interested in. For that, we take all subgraphs found in the original network, apply the canonical labeling described in section 3.5, and insert such canonical representation in the g-trie.

Henceforth, we can use the new g-trie (with a single copy of each subgraph type) to greatly constrain the frequency calculation in the random networks. The core idea is to backtrack through all possible connected sets of nodes, and at the same time only follow the possibilities that map exactly to a g-trie path. We take advantage of common substructures identified in the g-trie in the sense that at a given time we have a partial isomorphic match for several different candidate subgraphs (all the descendants in the g-trie). We also use symmetry breaking conditions to further constrain the search and avoid redundant computation, in a similar way to what we have done with uncolored motifs [8].

Algorithm 2 Census of subgraphs of G-Trie T in graph G

```
1: for all children $c$ of $T.root$ do
2:     MATCH($c$, $\emptyset$)
3: procedure MATCH($T, V_{used}$)
4:     $V$ := MATCHINGVERTICES($T, V_{used}$)
5:     for all node $v$ of $V$ do
6:         if isLeaf($T$) then FOUNDMATCH($V_{used} \cup \{v\}$)
7:         for all children $c$ of $T$ do MATCH($c, V_{used} \cup \{v\}$)
8: function MATCHINGVERTICES($T, V_{used}$)
9:     if $V_{used} = \emptyset$ then $V_{cand} := \{v \in V(G) : v$ respects color of g-trie node$\}$
10:    else $V_{cand} := \{v \in N(V_{used}) : v$ respects color and sym. conditions$\}$
11:    $Vertices := \emptyset$
12:    for all $v \in V_{cand}$ do
13:        if $V_{used} \cup v$ respects connections of $T$ then $Vertices := Vertices \cup \{v\}$
14:    return $Vertices$
```

Algorithm 2 details our method to count all occurrences of the g-trie colored subgraphs on a single randomized network. The idea is to find matches for all possible g-tries paths, i.e., all possible subgraphs. In the beginning we follow all g-trie root children and start with an empty partial match (lines 1 and 2). We then find all candidate vertices to fill the position of that g-trie node (line 4). If we are at a g-trie leaf, we found a complete match to a subgraph and we can increment its frequency (line 6). If not, we continue as before, recursively following all possible g-trie paths from there, i.e., all subgraphs that may start with the current partial match (line 7). In order to find the candidate vertices (lines 8 to 14) we have a look at the neighbors of the current partial match (line 10) and only use those that respect the symmetry breaking conditions (line 10) and that at the same time respect the color connections with the vertices already in the partial match (line 13).

3.5 Canonical labeling of a colored subgraph

For both previous algorithms we need to be able to produce a canonical labeling of a colored subgraph. This allows us to identify the subgraph type at each leaf in the g-trie that is dynamically constructed as we enumerate the subgraphs in the original network. At the same time, this canonical representation is used for choosing the path that will individually represent each isomorphic type in the g-trie that is used for searching in the randomized networks. The choice of this labeling will therefore directly interfere with the g-trie shape for this second part, including how much common-substructure is found, as explained in [8].

We opted to use the `GTCanon` labeling, which was created precisely for usage with g-tries [8]. At the core, this labeling uses `nauty`, a very efficient graph isomorphism program [19]. However, natively, `nauty` supports colors only in nodes and therefore we had to adapt it so that edge colors are also supported. In practice, for the labeling part, we convert each subgraph to an equivalent subgraph that uses colors only in vertices. The idea is to use several layers as is, for instance, exemplified in [20]. If we have k colors in the edges, we can substitute each vertex $v_j \in V(G)$ in the graph by a connected graph of k vertices $v_j^0, v_j^1, \ldots, v_j^k$ (we use cycles of size k). If there is an edge of color i between vertices v_j and $v_{j'}$, we add an edge between v_j^i and $v_{j'}^i$. This would imply a new graph with $|V(G)| \times k$ nodes, that is, with k layers. We reduce this to $|V(G)| \times log(k)$ by using the binary expansion of each color number and letting each layer represent a single bit in that binary representation.

4 Experimental evaluation

In order to evaluate our approach, we implemented the described algorithms in C++. All experiments were run on an Intel Core 2 6600 (2.40GHz) with 2GB of memory. We use four different networks, with varied topological features, as summarized in Table 1. All networks are undirected and simple, i.e., with no self-loops or multiple edges between the same pair of nodes.

Table 1 The set of four networks used for experimentation.

Name	Nodes	Edges	Avg. Degree	Nr of colors in: nodes	Nr of colors in: edges	Brief Description	Ref
`blogs`	1,490	16,715	22.4	2	1	links between political blogs	[21]
`dblp`	2,878	11,324	7.9	3	3	co-authorship of papers	[22]
`flights`	7,976	15,677	3.9	1	2	flights between airports	[23]
`elections`	8.297	100.753	24.3	2	2	elections for wikipedia admin	[24]

Given the nature of this paper, we always apply some kind of coloring in the networks used, even if the original dataset did not natively come with that color assignment. In `blogs`, all edges are of the same type and there are 2 colors in the nodes, indicating political leaning (left/liberal or right/conservative). By contrast, in `flights` all nodes are of the same type and there are 2 colors for the connections, indicating domestic and international flights. `dblp` is a co-authorship network in which we created 3 categories for the nodes (bottom 10%, top 1% and the rest) to differentiate the number of different co-authors each one has (the same was done for edges and number of co-authorships). Finally, `elections` is a signed network where there are 2 node colors (users being voted on and user casting votes) and 2 edge colors (supporting or opposing).

The only publicly available competing algorithm that performs a similar task is ESU, through its Fanmod tool [12], and therefore we directly compare our work with it, turning on colors on both nodes and connections. We experimented to compute motifs as we increase the size k of the subgraphs. We consider a typical usage of 100 randomized networks, generated with a markov chain process that keeps the same colored degree sequence of the original network, both on ESU and g-tries. We measured the execution time for each subgraph census, as shown in Table 2. For practical reasons, we limited the size k so that it would be feasible to run an entire motif computation with Fanmod.

Table 2 Experimental results.

network	k	Execution Time (seconds)						Speedup
		ESU (via Fanmod)			G-Tries			G-Tries
		Original	Avg.Random	Total	Original	Avg.Random	Total	vs ESU
`blogs`	3	2.1	2.1	209.06	0.73	0.29	29.73	7.0x
	4	232.10	263.45	26,577.10	53.04	15.10	1,563.04	17.0x
`dblp`	3	0.50	0.25	25.50	0.15	0.02	2.15	11.9x
	4	8.11	11.80	1,188.11	1.90	0.17	18.90	62.9x
	5	276.03	479.57	48,233.03	70.02	5.50	620.02	77.8x
`flights`	3	1.59	1.63	164.59	0.48	0.05	5.48	30.0x
	4	139.36	187.00	18,839.36	35.01	4.23	458.01	41.1x
`elections`	3	23.02	33.55	3,378.02	7.51	1.70	177.51	19.0x
	4	6,987.34	7,434.25	750,412.02	800.86	256.68	26,468.85	28.4x

We can observe that our algorithm consistently outperforms ESU. The exact speedup is not easy to predict and highly depends on the topology of the analyzed network. Nevertheless, the results show a tendency to achieve better speedups as k increases, being one to two orders of magnitude faster than ESU. Given the nature of the associated computational task, the time to compute grows exponentially as k increases. However, even an increase of just one node in the size of the motifs that we are able to calculate may be very important, because patterns previously unknown may become "visible". Furthermore, faster execution times may allow the analysis of larger networks. Regarding the memory usage, the census on the original network is more expensive on our method than with ESU, since as explained in section 3.3 we opted to trade memory for better running time. However, it is possible to use any other algorithm for this initial census, including ESU itself. Moreover, the task that dominates the total running time is the census on the ensemble of random networks and in that part g-tries are not more costly in terms of memory than ESU, because they natively provide a compact representation of the set of subgraphs in which we can store their respective frequencies.

The main focus of this paper is the algorithm in itself, but we feel it is important to show the usefulness of colored motifs and figure 3 shows some of the motifs we found. The significance of a subgraph is computed as in the original Milo et al. paper [1]. For instance, a subgraph is overrepresented if the probability that it appears more often in the randomized networks than in the original network is smaller than a

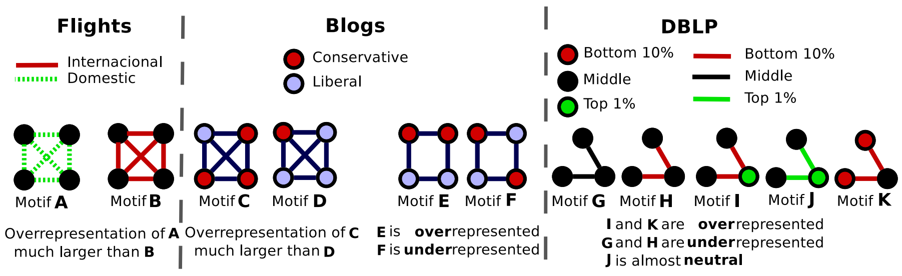

Fig. 3 Some examples of motifs found on the networks used for experimenting.

certain threshold. Here we are not worried about defining what the exact value of the threshold should be, but rather we want to show some general trends and our goal is to give the reader a feel of the kind of information colored motifs provide. For instance, without colors on both nodes and edges, the 5 patterns indicated for `dblp` would be indistinguishable. However, we can see that some of them are overrepresented (appearing more often than expected), some are underrepresented (appearing less than expected) and others are neutral (appearing in expected frequencies). The same happens on the other networks. Finding explanations for the depicted frequencies would be another (interesting) task, but what remains is that by considering colors we have a richer mining environment, able to expose patterns that would be invisible when only using the traditional uncolored analysis.

5 Conclusions

In this paper we gave a definition of colored network motifs and described a complete methodology to find such characteristic patterns. The main computational problem we tackled was calculation of subgraphs frequency and at the core or our algorithms lies the g-trie data structure, responsible for representing sets of subgraphs. We adapted existing algorithms and combined them in order to produce a solution that we have shown to be not only feasible, but also substantially faster than a direct competing algorithm. We also tried to show some of the potential in the usage of colored motifs.

In the near future we plan to publicly release the software created for the described algorithms and to explore extensions of these algorithms such as an approximation version with sampling capabilities or a parallel approach. We also intend to apply the concept of colored motifs on several real-world networks, and to experiment with different types of randomized networks

Acknowledgements Pedro Ribeiro is funded by an FCT Research Grant (SFRH/BPD/81695/2011). This work is partly funded by NSRF and ERDF through the COMPETE and ON.2 Programmes and by National Funds through FCT, within projects FCOMP-01-0124-FEDER-037281 and NORTE-07-0124-FEDER-000059.

References

1. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., Alon, U.: Network motifs: simple building blocks of complex networks. Science **298**(5594) (October 2002) 824–827
2. Albert, I., Albert, R.: Conserved network motifs allow protein-protein interaction prediction. Bioinformatics **20**(18) (December 2004) 3346–3352
3. Wu, G., Harrigan, M., Cunningham, P.: Characterizing wikipedia pages using edit network motif profiles. In: 3rd Int. workshop on search and mining user-generated contents (SMUC), New York, NY, USA, ACM (2011) 45–52
4. Schbath, S., Lacroix, V., Sagot, M.: Assessing the exceptionality of coloured motifs in networks. EURASIP Journal on Bioinformatics and Systems Biology (2008)
5. Adami, C., Qian, J., Rupp, M., Hintze, A.: Information content of colored motifs in complex networks. Artificial Life **17**(4) (2011) 375–390
6. Yeger-Lotem, E., Sattath, S., Kashtan, N., Itzkovitz, S., Milo, R., Pinter, R.Y., Alon, U., Margalit, H.: Network motifs in integrated cellular networks of transcription-regulation and protein-protein interaction. Proc. of the National academy of Sciences of the United States of America **101**(16) (2004) 5934–5939
7. Ribeiro, P., Silva, F.: G-tries: an efficient data structure for discovering network motifs. In: 25th ACM Symposium on Applied Computing (SAC), ACM (March 2010) 1559–1566
8. Ribeiro, P., Silva, F.: G-tries: a data structure for storing and finding subgraphs. Data Mining and Knowledge Discovery (2013)
9. Bruno, F., Palopoli, L., Rombo, S.E.: New trends in graph mining: Structural and node-colored network motifs. IJKDB **1**(1) (2010) 81–99
10. Qian, J., Hintze, A., Adami, C.: Colored Motifs Reveal Computational Building Blocks in the C. elegans Brain. PLoS ONE **6**(3) (March 2011) e17013+
11. Schbath, S., Lacroix, V., Sagot, M.F.: Assessing the exceptionality of coloured motifs in networks. EURASIP J. Bioinformatics and Systems Biology **2009** (2009)
12. Wernicke, S., Rasche, F.: Fanmod: a tool for fast network motif detection. Bioinformatics **22**(9) (May 2006) 1152–1153
13. Wernicke, S.: Efficient detection of network motifs. IEEE/ACM Transactions on Computational Biology and Bioinformatics **3**(4) (2006) 347–359
14. Kashani, Z., Ahrabian, H., Elahi, E., Nowzari-Dalini, A., Ansari, E., Asadi, S., Mohammadi, S., Schreiber, F., Masoudi-Nejad, A.: Kavosh: a new algorithm for finding network motifs. BMC Bioinformatics **10**(1) (2009) 318
15. Paredes, P., Ribeiro, P.: Towards a faster network-centric subgraph census. In: International Conference on Advances in Social Networks Analysis and Mining, IEEE (2013) 264–271
16. Grochow, J., Kellis, M.: Network motif discovery using subgraph enumeration and symmetry-breaking. Research in Computational Molecular Biology (2007) 92–106
17. Ribeiro, P., Silva, F.: Efficient subgraph frequency estimation with g-tries. In: International Workshop on Algorithms in Bioinformatics, Springer (2010) 238–249
18. Kashtan, N., Itzkovitz, S., Milo, R., Alon, U.: Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. Bioinformatics **20**(11) (2004) 1746–1758
19. McKay, B.D., Piperno, A.: Practical graph isomorphism, {II}. Journal of Symbolic Computation **60**(0) (2013) 94–112
20. Kao, M.Y., ed.: Encyclopedia of Algorithms. Springer (2008)
21. Adamic, L.A., Glance, N.: The political blogosphere and the 2004 U.S. election: divided they blog. In: 3rd international workshop on Link discovery (LinkKDD), New York, NY, USA, ACM (2005) 36–43
22. Kang, U., Papadimitriou, S., Sun, J., Tong, H.: Centralities in large networks: Algorithms and observations. In: SIAM International Conference on Data Mining. (2011) 119–130
23. Opsahl, T.: Why anchorage is not (that) important: Binary ties and sample selection. `http://toreopsahl.com/2011/08/12/` (August 2011)
24. Leskovec, J., Huttenlocher, D., Kleinberg, J.: Signed networks in social media. In: SIGCHI Conference on Human Factors in Computing Systems, ACM (2010) 1361–1370