1 Adjacency Matrix

An *adjacency matrix* A is the $|V| \times |V|$ matrix representation of a graph $G = (V, E)$. As we are only considering simple graphs, i.e., no loops exist, the diagonal alues are all 0, i.e., $A_{i,i} = 0, i \in [1, |V|]$.

$A_{i,j}$ indicates if an edge from v_i to v_j exists, i.e., $A_{i,j} = 1 \equiv (v_i, v_j) \in E$. In the case of undirected graphs, we simply set $A_{i,j} = A_{j,i}$ which results in an adjacency matrix mirrored at the diagonal, i.e., $A_{i,j} = A_{j,i} = 1 \equiv \{v_i, v_j\} \in E$.

As an example take the following two graphs:

....

graph / adjacency matrix example

....

The adjacency matrix of a directed graph can be written as a sequence of boolean values of length $|V| \cdot (|V| - 1)$. The adjacency matrix of an undirected graph is mirrored at the diagonal and can therefore be represented as a sequence of length $\frac{|V| \cdot (|V|-1)}{2}$.

$$a = a_1, a_2, \ldots a_{\frac{|V| \cdot (|V|-1)}{2}} = A_{1,2}, A_{1,2}, \ldots A_{1,|V|}, A_{2,3}, A_{2,4}, \ldots A_{|V|-1,|V|}$$

As examples, consider the boolean sequence representation for 3-, 4-, and 5-vertex graphs:

$$\begin{aligned}
3: &\quad A_{1,2}, A_{1,3}, A_{2,3} = a_1, a_2, a_3 \\
4: &\quad A_{1,2}, A_{1,3}, A_{1,4}, A_{2,3}, A_{2,4}, A_{3,4} = a_1, a_2, \ldots a_6 \\
5: &\quad A_{1,2}, A_{1,3}, A_{1,4}, A_{1,5}, A_{2,3}, A_{2,4}, A_{2,5}, A_{3,4}, A_{3,5}, A_{4,5} = a_1, a_2, \ldots a_{10}
\end{aligned}$$

Therefore, all possible adjacency matrices for a given graph size $|V|$ can be represented as the set $\mathcal{A}^{|V|} = \{0, 1\}^{\frac{|V| \cdot (|V|-1)}{2}}$. Each element from this set can also be interpreted as a number $n(a) \in \mathbb{N}$ as follows:

$$n(a) = n(A) = \sum_{i=1}^{\frac{|V| \cdot (|V|-1)}{2}} 2^{i-1} \cdot a_i$$

Hence, we can represent adjacency matrices as simple numbers:

....

example of adjacency matrix and their key

....

---

introduce the nummber as "key" of an adjacency matrix

introduce the induced adjacency matrix of a set of nodes (mentioning the order)

2 Motifs

We consider an undirected graph to be connected if there exists a path between each pair of vertices. We denote the set of connected adjacency matrices of k-vertex graphs ($k \geq 2$) as $\mathcal{A}_{con}^{|V|} \subset \mathcal{A}^{|V|}$.

As motifs of size k, also called k-vertex motifs or k-motifs, we consider the equivalence classes of isomorph connected k-vertex graphs which we denote as $\mathcal{M}_k$.

Each connected adjacency matrix is assigned to exactly one motif $m \in \mathcal{M}$. This assignment can be expressed as a function that maps a connected adjacency matrix to a motif, i.e.,

$$r : \mathcal{A}^k \to \mathcal{M}_k$$

define as mapping from $\mathbb{N}$

This assignment can be easily computed by enumerating all connected adjacency matrices and determining their equivalence class by performing an isomorphism check with all existing motifs.

3 Implementation

For simplicity, we store the function r as integer pairs (a, b) where a is hte key of a connected adjacency matrix and b the equivalence class it belongs to.

4 Statistics

nodes	ams	motifs	maxKey
2	1	1	1
3	4	2	7
4	38	6	63
5	728	21	1023
6	26704	112	32767
7	1866256	853	2097151

5 k-Neighborhood of an edge

As the *k-Neighborhood* $N(a, b)$ of an edge $\{a, b\}$, we denote the set of all $(k-2)$-tuples of vertices that are connected to vertices a or b. Thereby, each element of $N(a, b)$ corresponds to a connected subgraph of G that contains a and b.

For each induced subgraph $G \cap (\{a, b\} \cup n \in N(a, b))$, we can determine the corresponding motif as follows:

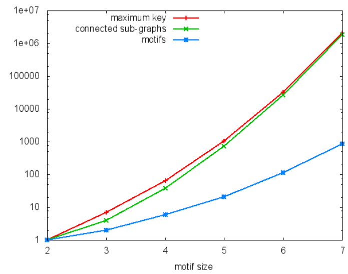

Figure 1: stats of undirected motifs

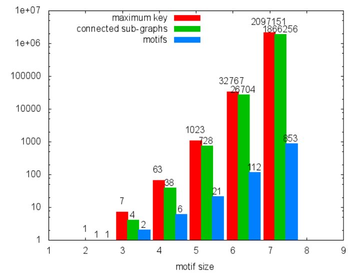

Figure 2: stats of undirected motifs

$$r(n(A_{\{a,b\}\cup N(a,b)}))$$

in case the edge exists already, we can determine the new motif after the edge removal as follows:

$$r(n(A_{\{a,b\}\cup N(a,b)}) - 1)$$

In case the edge is added, the new state of the motif is computed as follows:

$$r(n(A_{\{a,b\}\cup N(a,b)}) + 1)$$

6 Algorithm

Data: $G, \{a,b\}, type \in \{add, rm\}$
begin
  for $n \in N(a,b)$ **do**
    if $type = add$ **then**
      ; `/* edge is added */`
      $\mathcal{F}(r(n(A_{\{a,b\}\cup n}) + 1))$ += 1 ; `/* incr new motif */`
      if $n(A_{\{a,b\}\cup n}) \in \mathcal{N}^k_{con}$ **then**
        $\mathcal{F}(r(n(A_{\{a,b\}\cup n})))$ -= 1 ; `/* decr old motif */`
    else if $type = rm$ **then**
      ; `/* edge is removed */`
      $\mathcal{F}(r(n(A_{\{a,b\}\cup n})))$ -= 1 ; `/* decr old motif */`
      if $n(A_{\{a,b\}\cup n}) - 1 \in \mathcal{N}^k_{con}$ **then**
        $\mathcal{F}(r(n(A_{\{a,b\}\cup n}) - 1))$ += 1 ; `/* incr new motif */`
  end
end

Algorithm 1: *StreaM$_k$* for maintaining $\mathcal{F}$ in dynamic graphs

change set of connected keys to defined version